# Predictive Analytics in Healthcare for Diabetes Prediction

Final Year Project Thesis

by

Faizan Zafar (111528)

Saad Raza (111400)

Muhammad Umair Khalid (111835)

In Partial Fulfilment

Of the Requirements for the degree

Bachelor of Engineering in Software Engineering
(BESE)


School of Electrical Engineering and Computer Science

National University of Sciences and Technology

Islamabad, Pakistan

(2018)

# DECLARATION

We hereby declare that this project thesis entitled **"Predictive Analytics in Healthcare for Diabetes Prediction"** submitted to the **"School of Electrical Engineering and Computer Science"**, is a record of an original work done by the authors under the guidance of Supervisor **"Dr. Muhammad Ali Tahir"** and that no part has been plagiarized without citations. Also, this project work is submitted in the partial fulfilment of the requirements for the degree of Bachelor of Software Engineering.

| **Team Members** | **Signature** |
|---|---|
| Faizan Zafar | _____. |
| Saad Raza | _____. |
| M. Umair Khalid | _____. |

| **Supervisor:** | **Signature** |
|---|---|
| Dr. Muhammad Ali Tahir | _____. |

**Date:**
03 May, 2018

**Place:**

HPC Lab, School of EE & CS, NUST, Islamabad

# DEDICATION

To God The Almighty

&

To our Parents, Colleagues, Advisors and Faculty Members

# ACKNOWLEDGEMENTS

First and foremost, we would like to express our sincere thanks to The God Almighty for the gift of life, wisdom and understanding He had given to us, a reason for our existence. And to our families for the love and support they had provided throughout our life.

**Dr. Muhammad Ali Tahir** and **Dr. Muhammad Imran Malik** whom we regard as our mentors and supervisors, we thank them for the expertise and intelligence they have displayed while supervising this project. We believe this work is a result of their good guidance and cooperation.

We cannot forget our friends in the Faculty of Computing for the academic interactions and company they have accorded to us.

Lastly, we would like to convey our gratitude to the lecturers in our faculty for the good job done during the 4-year period of our course. May the good Lord bless them and keep them safe.

# Table of Contents

# ABSTRACT

Diabetes is a chronic disease that poses a great challenge for human health worldwide. On average, about 8.3% of people are diagnosed with diabetes around the world. In Pakistan alone, 35.3 million individuals had diabetes in 2017, and millions more were at a high risk to develop the disease.

The aim of this project is to aid in the early detection and efficient diagnosis of Type 2 diabetes by computing a probability of them having diabetes based on their clinical data. For this, we have used machine learning techniques and statistical modelling measures. The project has focused on providing the most accurate results in diabetes prediction based on certain diagnostic measurements present in a particular dataset obtained from Kaggle. For conducting literature review and understanding the healthcare topic, relevant papers about healthcare analytics were searched in popular databases such as google scholar and springer using specific keywords.

The most significant and obvious result of using such technology within the healthcare sector is its positive results on costs and yielding of instant diagnosis. Because of reduced cost, electronic information is one of the main aspects that has a dominant impact on healthcare predictive analytics.

For the implementation phase, we have used JetBrains Pycharm IDE for development in Python 3.6 and RStudio for statistical modelling in R 3.4. The python libraries which proved to be most useful for our experiments were *numpy*, *pandas* and *sklearn*.

*Chapter 1*

# 1. INTRODUCTION

Applications of computing in healthcare carry the potential to revolutionize the industry and provide benefits of immense magnitudes. Diabetes is one of the critical diseases, which has long-term complications associated with it and follows with various health problems. It is estimated that diabetes currently affects 425 million people worldwide. This figure is expected to rise to over 642 million by 2040. The overwhelming scale of the problem presents significant challenges to healthcare systems and clinical practices. Both the prevalence and the incidence of Type 2 diabetes rise with increasing age and is observed more in women due to physiological reasons. Hence, we have solely targeted it for our thesis.

Predictive analytics uses statistical or machine learning methods to make a prediction about future or unknown outcomes. These methods are gaining increasing momentum and attracting a lot of attention in the field of medical research. They have shown their capabilities to effectively deal with large numbers of variables while producing powerful prediction models. They also embed variable selection mechanisms, which can detect complex relationships in the data.

## 1.1 Background

According to the first WHO Global report on diabetes published on World Health Day April 7, 2016, the number of adults living with diabetes has almost increased four times since 1980 to 422 million. It caused about 1.5 million deaths in 2012. This threatening numbers necessitates the development of effective and accurate diagnosis tools that doctors and practitioners must make use of.

Diabetes can be divided into several distinct types. However, there are two major clinical types, type 1 diabetes (T1D) and type 2 diabetes (T2D), according to the etiopathology of the disorder. T2D appears to be the most common form of diabetes (90% of all diabetic patients), mainly characterized by insulin resistance. The project considers data and features for T2D.

With the help of technology, it is necessary to build a system that stores and analyses clinical data of diabetic patients and predicts its outcome accordingly. Applying machine learning methods in diabetes research is a key approach to utilizing the volumes of available diabetes-related data for extracting knowledge. Machine learning is the scientific field dealing with the ways in which machines learn from experience. The purpose of machine learning is the construction of computer systems that can adapt and learn from their experience.

## 1.2 Problem Statement

Patients with the potential of diabetes have to go through a series of tests and exams to diagnose the disease properly. These tests include some redundant or unnecessary medical procedures, which lead to intricate complications and wastage of time & resources.

The average lifetime costs of direct medical treatment for a diabetic patient were estimated to be approximately $85,000 in 2012 in the United States. The burden of this disease on the economy far exceeds the direct medical costs in the healthcare sector because diabetes reduces the quality of life and labour productivity. This happens because the symptoms are neglected and there is no proper diagnosis scheme. Hence, preventing the disease altogether through early detection could potentially reduce burden on the economy and help in diabetes management of the patient.

## 1.3 Project Objectives

### 1.3.1 Specific Objectives

The specific objectives of our project include the following:

- Reviewing literature on related systems and techniques and analyze the existing solutions.
- Gaining knowledge of the relevant medical discipline
- Gathering the dataset
- Preprocessing and refining of dataset
- Extracting the useable features
- Implementing the system
- Testing and validation of the system

## 1.4 Scope

Our project scope encompasses the domain of data science. We aim to make use of data analytics and machine learning algorithms to provide concrete statistics in the field of medicine. We have chosen this particular domain because of our own personal interest and the rising need for bridging the gap between medicine and technology. Our project can help in future research and development related to this field. Our project aims to provide results in the form of predictions which have a certain probability of occurrence.

## 1.5 Motivation

Machine learning and statistical modelling in the medical field have a lot of scope in upcoming technological era. New methodologies and systems are being developed to ease and improve the diagnosis process. Disease prediction is an area

of which is growing day by day and it is highly recommended for a software engineer to have the understanding and knowledge of this domain.

The security and confidentiality concerns of medical datasets put a lot of constraints to the development of healthcare systems. While studying about these constraints, we faced several problems and tackled them head on, the details of which are mentioned later in this document.

## 1.6 Significance

Many external factors including lack of experienced medical professionals or unavailability of resources may lead to incorrect diagnosis of diabetes. Therefore, a computational approach provides a strong alternative for diabetes prediction and diagnosis. The computational tools and methods may help clinicians to make accurate diagnosis. Also, it will help individuals to get acquainted about their health status and future possible diabetic condition so that they can get the chance to adopt a better lifestyle to prevent the disease.

## 1.7 How is Our Solution Different?

- Use of Gradient Boosting machine learning algorithm
- Use of F1 score
- Easy to interpret the results (Probabilistic intuition)
- Comparison and contrast of the results of various known machine learning algorithms
- Space for expansion in future

## 1.8 Deadline and Schedule

The timeline following in order to complete the project successfully is as following:

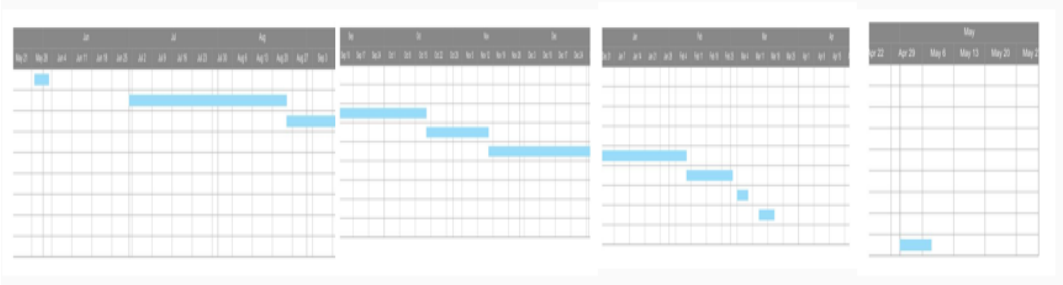

*Figure 1-Gantt Chart*

## 1.9 Report Organization

The report is organized in nine chapters:

**Chapter 1** consists of the introduction, scope and explains the major need of the project

**Chapter 2** consists of the Literature Review and explains the Background of the problem and work being done in this domain.

**Chapter 3** defines the problem to which we are proposing our solution.

**Chapter 4** consists of the methods, approaches, tools, techniques, algorithms, or other aspects of the solution.

**Chapter 5** consists of the high-level overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components.

**Chapter 6** consists of the methods, tools and techniques used to develop the software

**Chapter 7** consists of a comprehensive evaluation of the solution is presented with supporting figures and graphics.

**Chapter 8** includes a brief summary of how the proposed solution has addressed the problem statement.

**Chapter 9** consists of the references.

*Chapter 2*

# 2. LITERATURE REVIEW

## Introduction

In this section of the document, the research, location and analysis of the existing knowledge related to the subject of inquiry are exploded and cited. It also sells at the relationship of the proposed research for purposes of good representation and critical review of the existing literature.

Given the current situation, there is much that needs to be done in the domain of diabetes prediction. There is a dire need to overcome the constraints of diabetic dataset management. A lot of work is done and is being done in the field of Predictive Analytics in Healthcare in general.

## 2.1 Related Work

The review on prior work gives several results on analysis of healthcare data which was carried out by different methods and techniques. Design of prediction models for diabetes diagnosis has been an active research area for the past decade. The advanced set of models found in literature are based on clustering algorithms and artificial neural networks. We have cited different articles and research papers and presented our findings in correlation with them.

Following are the few projects that have been done in our domain which we studied to learn about this domain:

### 2.1.1 Diagnosis of diabetes using classification mining techniques

*Diagnosis of diabetes using classification mining techniques. Aiswarya Iyer et al. (2015),* used classification technique to find out patterns from the datasets of

diabetic patients. They deployed Naive Bayes and Decision Tree algorithms by using Weka tool. Authors also compared performance of both algorithms on Pima dataset. Experimental results showed effectiveness of each proposed classification model.

### 2.1.2 The Evolution of Boosting Algorithms – From Machine Learning to Statistical Modelling

*The Evolution of Boosting Algorithms – From Machine Learning to Statistical Modelling. Mayr A, Binder H, Gefeller O, Schmid M*, present the idea of boosting to increase the accuracy of a weak classifying tool by combining various instances into a more accurate prediction model. They highlighted the concepts such as gradient boosting, adaptive boosting and likelihood-based boosting in classification and regression problems.

### 2.1.3 An Expert Clinical Decision Support System to Predict Disease Using Classification Technique

*An Expert Clinical Decision Support System to Predict Disease Using Classification Technique, Emrana Kabir Hashi, Md. Shahid Uz Zaman and Md. Rokibul Hasan (2017).* The authors have developed an expert healthcare decision support system that predicts diabetes in a patient. The model is trained on Pima dataset. C4.5 decision tree and K-nearest Neighbour algorithms are used to develop the model and the former achieved a remarkable accuracy of 90.43%.

### 2.1.4 Predicting diabetes mellitus using SMOTE and ensemble machine learning approach: The Henry Ford Exercise Testing (FIT) project

*Predicting diabetes mellitus using SMOTE and ensemble machine learning approach: The Henry Ford Exercise Testing (FIT) project, Manal Alghamdi, Mouaz Al-Mallah, Steven Keteyian, Clinton Brawner, Jonathan Ehrman, Sherif Sakr (2017).* Researchers developed an ensembling-based predictive model using thirteen attributes that were selected based on their clinical importance. The study shows the potential of ensembling and SMOTE approaches for predicting incident diabetes using cardiorespiratory fitness data.

### 2.1.5 Machine Learning and Data Mining Methods in Diabetes Research

*Machine Learning and Data Mining Methods in Diabetes Research, Ioannis Kavakiotis, Olga Tsave, Athanasios Salifoglou, Nicos Maglaveras, Ioannis Vlahavas, Ioanna Chouvarda.* This paper provides computational insight into diabetes research. It gives a detailed outline of machine learning and knowledge discovery and discusses the previous established approaches, their outcomes and limitations. The applications in the selected article project the usefulness of extracting valuable knowledge leading to new hypotheses targeting deeper understanding and further investigation in diabetes.

### 2.1.6 Predictive Analysis of Diabetic Patient Data Using Machine Learning and Hadoop

*Predictive Analysis of Diabetic Patient Data Using Machine Learning and Hadoop. Gauri D. Kalyankar, Shivananda R. Poojara, Nagaraj V. Dharwadkar (2017),* work on decision tree algorithm in Hadoop MapReduce environment to find out missing values and discover patterns from it. This work predicts types of diabetes widespread, related future risks and the type of treatment that can be provided, based on the risk level of the patient.

*Chapter 3*

# 3. PROBLEM DEFINITION

## 3.1 Background

Effects of diabetes have been reported to have a more fatal and worsening impact on women than on men because of their lower survival rate and poorer quality of life. WHO reports state that almost one-third of the women who suffer from diabetes have no knowledge about it. The effect of diabetes is unique in case of mothers because the disease is transmitted to their unborn children. Strokes, miscarriages, blindness, kidney failure and amputations are just some of the complications that arise from this disease. For the purposes of this project, the analyses of diabetes cases have been restricted to women of age 21 and above.

## 3.1 Problem Definition

Delayed diagnosis and control of diabetes significantly increases the risk of secondary complications, such as cardiovascular diseases, kidney failure, and sight problems, as well as lower extremity and foot problems. However, half of people with diabetes are not aware of their disease. In the majority of cases, diabetes is diagnosed in the advanced stage requiring medications. Therefore, timely diagnosis at an early stage of disease using clinical and lifestyle indicators such as BMI (body mass index) can save lives and health resources.

The diagnosis of diabetes in the elderly and women also presents challenges, and it is estimated that half of the mentioned population with diabetes are not diagnosed correctly. This is due to many factors, including the observation that this cohort rarely presents with the typical symptoms of hyperglycaemia. Moreover,

fasting blood glucose measurements or oral glucose tests are not routinely performed. Common symptoms such as fatigue, blurred vision, and polyuria are often not recognized as abnormal in this population, and polydipsia can go unnoticed because of the decreased thirst usually associated with advanced age or pregnancies.

The domain of our project includes Hospitals, Clinics, Governments, Research Institutes, IT industries and Medical Industries. These organizations have been facing the issues and problems related to diagnosis of diabetes for quite long and have deployed various solutions to overcome them. Our technological solution aims to provide accurate results based on specified clinical and physiological factors. Though there are abundant factors which actually lead to diabetes in a human, all of them could not be quantified or explained within the scope of this project. Hence, only considerable components were taken into account.

*Chapter 4*

# 4. METHODOLOGY

## 4.1 Introduction

This section contains a description of the methods, approaches, tools, techniques and algorithms chosen to achieve the objectives of the proposed system. It will go on to describe the techniques of data collection, data processing and presentation of results that will be employed in the research study of the proposed systems. The tools used to develop the prototype application are JetBrains PyCharm, RStudio, WEKA, Open Refine, Microsoft Excel integrated with specific libraries.

## 4.2 Approach for the development of predictive model for diabetes prediction

The nature of the project is a research-based one, hence a traditional approach for software design and development was a not a feasible one. We were determined to improve upon previous findings in diabetes prediction and present new discoveries on known datasets but the uncertainty of those results was a hindrance for the project as a whole. Moreover, since this domain was new to us, we had to gain an understanding of it fully and overcome initial problems to participate in the implementation processes. For these reasons, we engaged in a procedural and systematic approach where extensive resources were allocated on studying the need, benefits and throughput of the project, and then we spent efforts on the actual execution and output of the project. The project contains aspects of both core programming and reviewing literature.

The steps involved in the overall methodology of the project are highlighted as follows:

- **Information gathering:**

  The project is aligned with our interests to make an impact via the applications of data science and machine learning in the healthcare sector. Hence, we had to collect information on and review related projects in the field, their success rates and gain domain knowledge.

- **Research on project feasibility:**

  Another important aspect was to assess the viability of the research and to what extent it was beneficial and useable for end users. Upon interviewing and questioning concerned medical authorities, we concluded that this research area has vast applications for different stakeholders and they were attracted towards the promise of automated diagnosis of diabetes.

- **Research on type of data required:**

  Next, we collected information on the clinical and lifestyle indicators which contribute to the prevalence of diabetes. Prioritizing these features was a rigorous task since there is no concrete proof that a set of symptoms will provide causation for diabetes.

- **Data gathering:**

  We searched for datasets containing values within the boundaries defined by clinical experts. This phase was particularly time consuming. In the initial phases, we visited different hospitals and clinics in the locations of Islamabad and Lahore for collection and gathering of the dataset directly. Our efforts did not prove successful as electronic health records were close to non-existent in Pakistan and there were legal and confidential complications in sharing the data of patients, where they were actually available. Learning from this experience and upon the suggestion of medical professionals, we set out to develop a questionnaire for patients to collect variables from them

which were only required for our research but nothing fruitful came of this in helping us achieving our goals for the research. Harsh attitudes of patients and general unawareness of computer science were the reasons we believe this happened. Lastly, we searched thoroughly on Kaggle to find datasets used for similar experiments. This attempt was fortunate as we discovered the PIMA dataset whose integrity and authenticity was up voted by majority of the research community. The details of the attributes and specifications of the dataset are given in Chapter 6.

- **Data pre-processing:**

  This step included tempering with and clarifying the medical records for null/zero values, missing and inconsistent values and broken links. Integrity and validation checks were placed on the dataset to ensure authenticity. Data Augmentation was performed to reduce overfitting and increase prediction precision. Moreover, the source of the dataset and its citations were fully explored before using it in our project.

- **Research on machine learning algorithms:**

  As part of developing a prediction model, we explored various machine learning classification and regression techniques. Since the data is of numerical nature for some fields and category-based for other fields, it was appropriate to implement algorithms, which work best for both data types. The functional details of these algorithms have been mentioned in Section 4.4.

- **Implementing algorithms:**

  This is the actual implementation procedure where we dedicated significant efforts in trying to achieve most accurate results while studying in detail the training and test set, data processes and output of the algorithms.

- **Comparing results of different algorithms:**

  To intuitively measure accuracy of prediction, we used the F1 score for different algorithms on the same dataset. The values were analysed and the algorithm with the highest F1 score was chosen for further improvement.

- **Improving the findings from the results:**

  Different experiments were conducted with the gradient boosting algorithm to realize the importance of certain features in the prediction of the decision. Overfitting of the was minimized because of the 'boosting' effect combined with introduction of 'noise' in the training dataset.

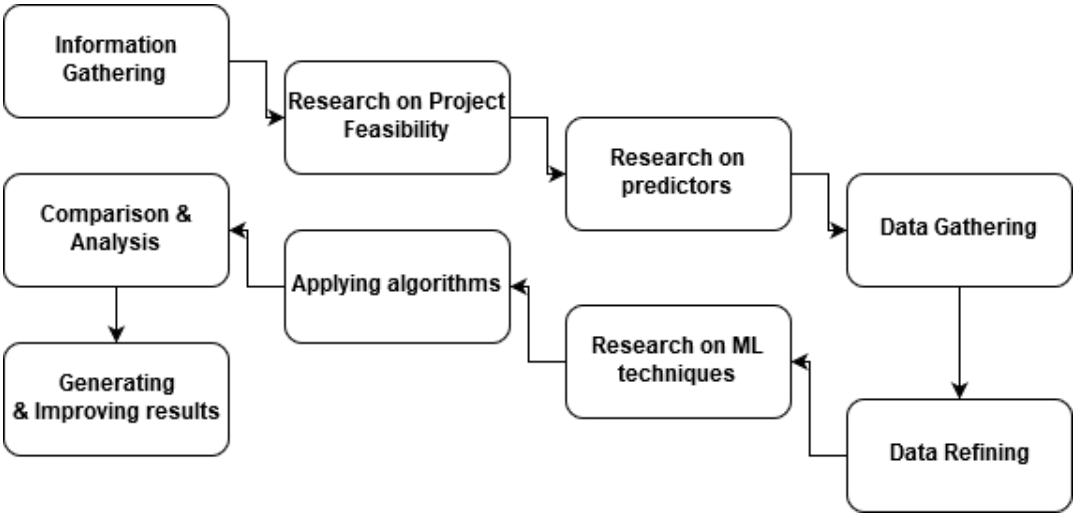


*Figure 2-Approach for project development*

The figure below shows a generalized approach we adopted for extracting useful information from varied data points. We utilized the concept of knowledge discovery which is a field encompassing theories, methods and techniques, trying to make sense of data and extract useful knowledge from them. It maps out to be a multistep process (selection, pre-process, transformation, data mining, interpretation, and evaluation) depicted in Figure 3.

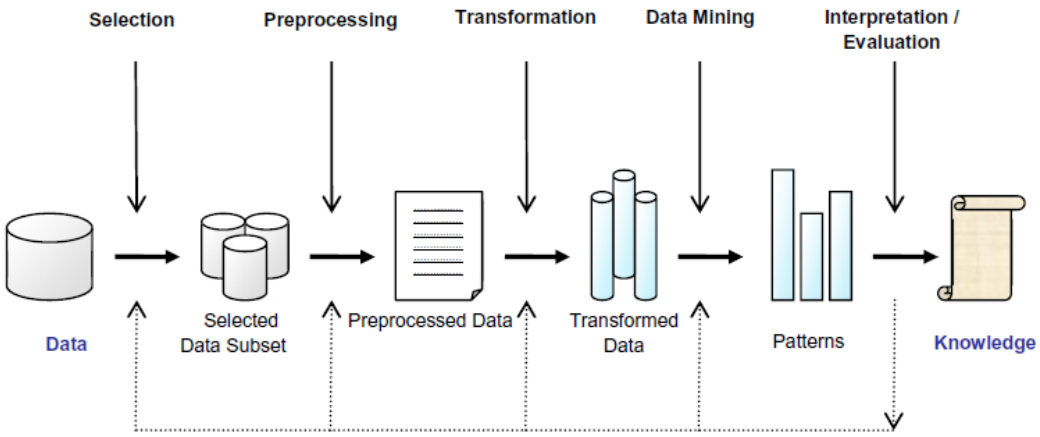


*Figure 3-A generalized approach*

The clinical prediction model is very valuable as it can be applied to a various scenario like screening, prediction, medical decision making and education in health.

## 4.3 Research Methodology

A research methodology is a systematic plan for conducting research. Sociologists draw on a variety of both qualitative and quantitative research methods, including experiments, survey research, participant observation, and secondary data. Quantitative methods aim to classify features, count them, and create statistical models to test hypotheses and explain observations. Qualitative methods aim for a complete, detailed description of observations, including the context of events and circumstances.

The PIMA dataset, which the project uses, consists of 768 instances of a heterogeneous sample of diabetic and non-diabetic patients. Diabetic patients represent 34.9% of the whole sample while non-diabetic patients represent 65.1%. The variance between the two classes is considerably large and could possibly lead to lower out of sample accuracy of the classifiers.

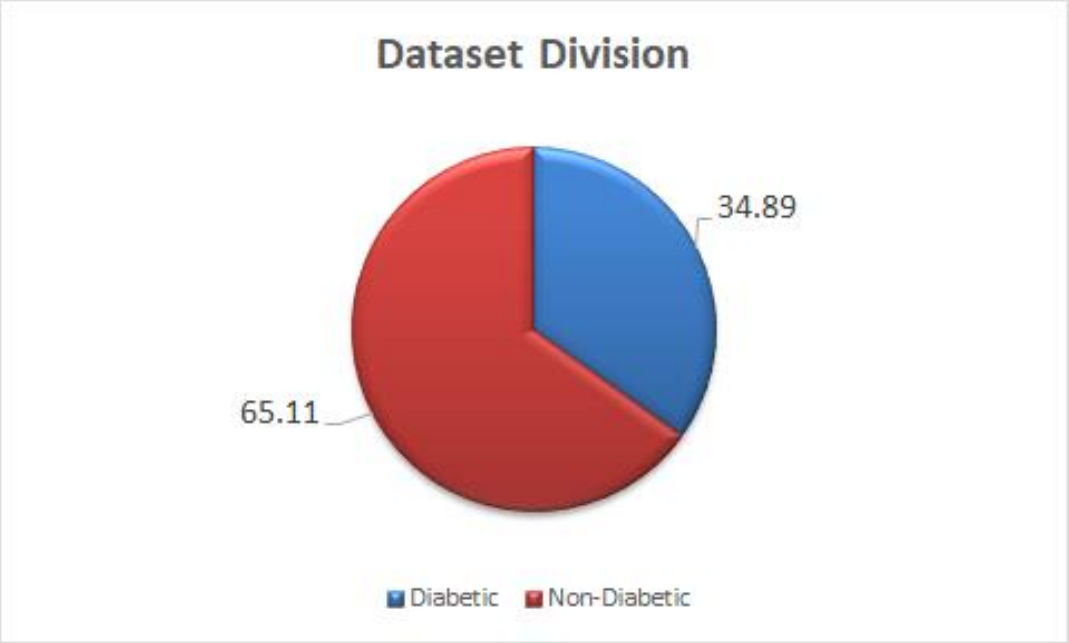


*Figure 4-Dataset division*

In most cases, the real-world data is imbalanced in many applications such as fraud detection, prevalence of diseases, credit scoring, or medical diagnosis. Class imbalance is a supervised learning problem and is very popular in the community of data science. The class imbalance problem occurs when there is a big difference between the number of majority class and the minority class and mostly in classes with binary values. The disparity caused in the values of the target class could have an extremely negative impact on the performance of the machine learning algorithms. Most of the time, it would lead to false classification and the prediction result will be either over-fitted because the model does not attenuate the bias for the majority class or under-performed due to the very few instances of positive class.

## 4.4 Algorithms

The findings of the research-based project are developed and documented according to Software Engineering principles. Multiple machine learning algorithms (classification and regression techniques) have been to develop the eventual prediction model for the dataset. We engaged in a comparative analysis where the

F1 scores of the algorithms were considered. The reason for using F1 score as a measure of accuracy over other measures is presented in section 4.4.8.

Following are the details of the core algorithms and accuracy measure we considered:

### 4.4.1 Gaussian Naïve Bayes

***Sci-kit source:***

*class* `sklearn.naive_bayes.GaussianNB`(*priors=None*)

The Gaussian Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. When dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a Gaussian distribution. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

$$p(x = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}}$$

*Figure 5-Mathematical functional underlying Gaussian Naïve Bayes*

### 4.4.2 K Nearest Neighbour (kNN)

***Sci-kit source:***

*class* `sklearn.neighbors.KNeighborsClassifier`(*n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=1, **kwargs*)

The kNN algorithm is arguably the simplest machine learning algorithm. Building the model consists only of storing the training set. To make a prediction for a new data point, the algorithm finds the closest data points in the training data

set—its “nearest neighbours.” The proposed work has used Euclidean distance to define the closeness:

$$d(X,Y) = \sqrt{\sum_{i=0}^{n} (Xi - Yi)^2}$$

Where, X=(x1, x2……,xn) and Y=(y1,y2…..yn)

Step 1: Input: D= $\{(x_1, c_1),\ldots.(x_n, c_n)\}$
x=$(x_1,\ldots..,x_n)$ new instance to be classified
Step 2: For each labeled instance $(x_i,c_i)$
*Calculate* $d(x_i, x)$
Step 3: *Order* $d(x_i, x)$ from lowest to highest, (i=1,…..,N)
Step 4*: Select* the K nearest instances to x : $D_x^K$
Step 5: *Assign* to x the most frequent class in $D_x^K$

*Figure 6-Algorithm for kNN classifier*

### 4.4.3 AdaBoosting

***Sci-kit source:***

*class* `sklearn.ensemble.AdaBoostClassifier`(*base_estimator=None, n_estimators=50, learning_rate=1.0, algorithm='SAMME.R', random_state=None*)

Adaptive Boosting (AdaBoosting) converts a set of weak learners into a single strong learner. It initializes a strong learner (usually a decision tree) and iteratively creates a weak learner that is added to the strong learner. At each iteration, adaptive boosting changes the sample distribution by modifying the weights attached to each of the instances. It increases the weights of the wrongly predicted instances and decreases the ones of the correctly predicted instances. The weak learner thus focuses more on the difficult instances. After being trained, the weak learner is added to the strong one according to his performance (so-called alpha weight). The higher it performs, the more it contributes to the strong learner.

### 4.4.4 Keras Neural Network

Keras is a high-level neural networks API (application programming interface) capable of running on top of TensorFlow. It helps with fast experimentation which is essential for this project. It allows to go from idea to result with the least possible delay and is proven to being key to good research. A prediction model is understood as a sequence or a graph of standalone, fully-configurable modules that can be plugged together with as little restrictions as possible. In particular, neural layers, cost functions, optimizers, initialization schemes, activation functions, regularization schemes are all standalone modules that can be combined in Keras to create new models.

### 4.4.5 Random Forest Classifier

***Sci-kit source:***

*class* `sklearn.ensemble.RandomForestClassifier`(*n_estimators=10, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=1, random_state=None, verbose=0, warm_start=False, class_weight=None*)

Random forest classifier creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object. The random-forest algorithm brings extra randomness into the model, when it is growing the trees. Instead of searching for the best feature while splitting a node, it searches for the best feature among a random subset of features. This process creates a wide diversity, which generally results in a better model. Another great quality of the random forest algorithm is that it is very easy to measure the relative importance of each feature on the prediction.

### 4.4.6 Logistic Regression (LR)

***Sci-kit source:***

*class sklearn.linear_model.**LogisticRegression**(penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='warn', max_iter=100, multi_class='warn', verbose=0, warm_start=False, n_jobs=None)*

Logistic Regression (LR) is a statistical classifier that provides the probability for predicting the labeled class of categorical type by using a number of attributes. The goal of logistic regression is to find the best fitting (yet biologically reasonable) model to describe the relationship between the dichotomous characteristic of interest (dependent variable i.e. the response or outcome variable) and a set of independent (predictor or explanatory) variables. Logistic regression generates the coefficients (and its standard errors and significance levels) of a formula to predict a logit transformation of the probability of presence of the characteristic of interest:

$$logit(p) = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + \ldots + b_kX_k$$

*Figure 7-Underlying equation for LR*

### 4.4.7 Gradient Boosting

***Sci-kit source:***

*class* sklearn.ensemble.**GradientBoostingClassifier**(*loss='deviance', learning_rate=0.1, n_estimators=100, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=3, min_impurity_decrease=0.0, min_impurity_split=None, init=None, random_state=None, max_features=None, verbose=0, max_leaf_nodes=None, warm_start=False, presort='auto', validation_fraction=0.1, n_iter_no_change=None, tol=0.0001*)

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models. It involves three critical elements:

- A loss function to be optimized.
- A weak learner to make predictions.
- An additive model to add weak learners to minimize the loss function.

Gradient boosting generates learners during the learning process. It builds the first learner to predict the labels of samples, and calculate the loss i.e. the difference between the outcome of the first learner and the real value. Then, it builds a second learner to predict the loss after the first step. The step continues to learn until a certain threshold. Because of its mathematical nature, gradient boosting outputs the same value of prediction for the corresponding set of input values, unlike keras neural network, where the final output is dependent on the outputs of the activation functions in each of the layers.

**Algorithm 1: Gradient_Boost**

1 $F_0(\mathbf{x}) = \arg\min_\rho \sum_{i=1}^{N} L(y_i, \rho)$

2 For $m = 1$ to $M$ do:

3 $\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}, \; i = 1, N$

4 $\mathbf{a}_m = \arg\min_{\mathbf{a},\beta} \sum_{i=1}^{N} [\tilde{y}_i - \beta h(\mathbf{x}_i; \mathbf{a})]^2$

5 $\rho_m = \arg\min_\rho \sum_{i=1}^{N} L(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m))$

6 $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$

7 endFor

end Algorithm

*Figure 8-Mathematical model for gradient boosting*

### 4.4.8 F1 Score

A number of metrics are used in Machine Learning to measure the predictive accuracy of a model. The choice of accuracy metric depends on the Machine Learning task. We reviewed several metrics to decide the correctness of our

prediction model. *Accuracy* is the proportion of correct results that a classifier achieves. However, it does not cater to false positives and false negatives, where the accuracy of a classifier may increase while simultaneously giving erroneous prediction outputs. We were interested in a judging criterion for a model which made use of precision and recall as these are less misleading measures. *Precision* highlights the correct fraction out of all the examples the classifier labelled as positive. *Recall* is simply the true positive rate. We made use of the *F1 score* which is the harmonic mean of these values. F1 score has an intuitive meaning. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances). We were inclined to use it because our dataset was imbalanced i.e. it had unequal number of positive and negative outcomes. Accuracy is not a good measure for imbalanced data. This problem is recurrent with a class imbalance when classification accuracy alone cannot be trusted to select a well-performing model.

$$F_1 = \frac{2}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

*Figure 9-Equation for F1 score*

## 4.5 Tools and Techniques

The model is implemented using python and its libraries such as *numpy*, *ensemble*, *sklearn*, *pandas* and *matplotlib* which are amongst the most advanced technologies in machine learning domain. The results are visualized in RStudio and Microsoft Excel. The reason for choosing them is the ease of use and availability of powerful features which make data visualization quite an interesting and meaningful task.

## 4.6 Tools

- JetBrains Pycharm
- RStudio
- WEKA
- OpenRefine

*Chapter 5*

# 5. DETAILED DESIGN AND ARCHITECTURE

## 5.1 System Architecture

### 5.1.1 Architecture Design

Research on Predictive Analytics in Healthcare for Diabetes Prediction is self-contained and being developed for individuals and organizations to assist them in diagnosis and ultimately prevention of the disease. It is to be designed and developed from the detailed descriptions of algorithms and new user requirements.

The architecture of the diabetes prediction model includes various modules such as data collection, pre-processing, predictive analysis, post-processing, extracting meaningful results. The overall system architecture is shown as a block diagram given below:

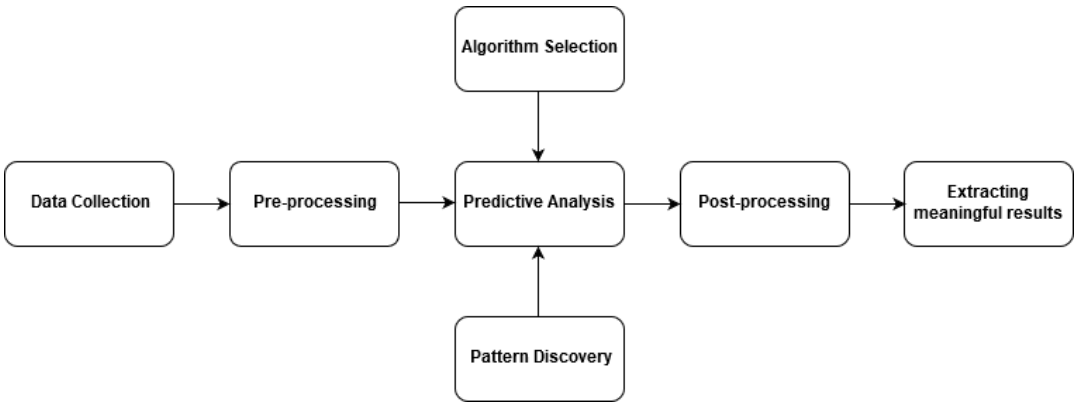


*Figure 10-Project Architecture*

The architecture of the model development is shown below in the architecture diagram:

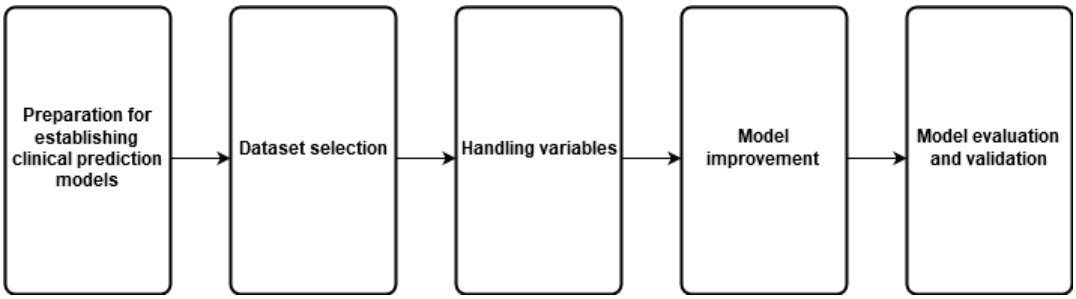


*Figure 11-High-level Model Architecture Diagram*

Concrete results and information of the data are expected from the experimentation phase. There are some major responsibilities that the project must undertake and the various roles that the diagnosis system must play. These responsibilities and roles include:

- User input validation
- Connectivity to backend model
- Diagnosis as a probabilistic output

The architecture of the prototype application is shown below in the architecture diagram:

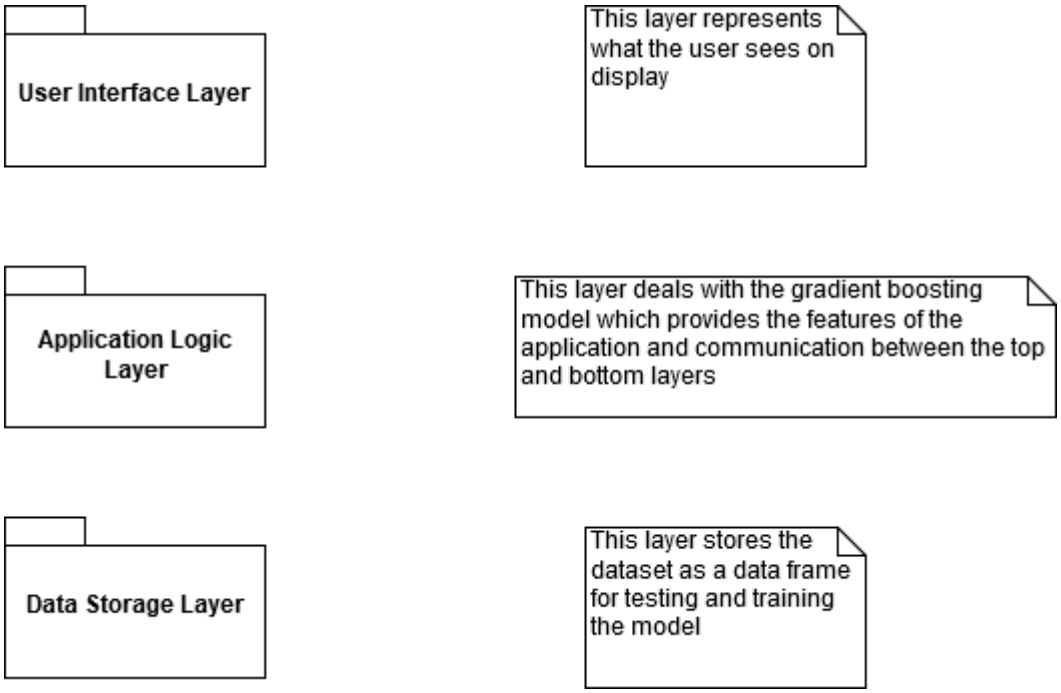


*Figure 12-Architecture Diagram*

### 5.1.2 Architecture Design Approach

There are five types of architecture design approaches. These approaches include:

- Linear Approach
- **Divisions Approach**
- Centralized Approach
- Cycle Approach
- Investigative Approach

The architecture design approach we chose for our application is Divisions approach. In this approach, the design process is choosing the best solution out of several divisions of design solutions. We chose this approach because our project was a research-based one and we had several designs in mind from which we chose the most doable and optimal design.

### 5.1.3 Subsystem Architecture

There are multiple subcomponents of our project. These subcomponents are:

- User input validation
- Connectivity to backend model
- Diagnosis as a probabilistic output

#### 5.1.3.1 User input validation

The user input to the application is validated to check for any missing entries.

#### 5.1.3.2 Connectivity to backend prediction model

The graphical user interface links the prediction model and the end user. The user can obtain the results of diabetes diagnosis via the interface. The technical details of the link between the model and interface have been explained in Chapter 6.

#### 5.1.3.3 Diagnosis as a probabilistic output

The result of diagnosis is a probability of a person having type 2 diabetes based on his clinical information. A conclusive and essential part of the architecture of the project was to display and present the results and findings in a meaningful and interpretable way. Bar graphs, line graphs, pie charts, point plots aided us in this phase. These metrics helped us in comparative analysis of the underlying algorithms which was a main objective of the project.

## 5.2 Detailed System Design

The prototype application and research on Predictive Analytics in Healthcare for Diabetes Prediction consists of a number of modules and processes. These modules combine to make the complete research output and enable the developer to generate results and visualize them accordingly.

- User input validation
- Connectivity to backend model
- Diagnosis as a probabilistic output

This section includes the detailed description of all these components.

### 5.2.1 User input validation

#### 5.2.1.1 Classification

The classification of the component is that it is a module.

#### 5.2.1.2 Definition

Validation process on user entered information

#### 5.2.1.3 Responsibilities

The user input to the application is validated to check for any missing entries.

#### 5.2.1.4 Constraints

The prototype does not check for the normal range of the indicators with respect to the age of that the user.

#### 5.2.1.5 Composition

This module is composed of the higher-level user interface layer only.

#### 5.2.1.6 Uses/Interactions

Interactions with the application for this module can be done via keyboard input.

#### 5.2.1.7 Resources

The validation logic built with the *Tkinter* application is the primary resource for this module. Secondary resource is the model which actually provides the accurate output.

#### 5.2.1.8 Processing

The data is passed into the *def validate_click(data):* to check for the complete and valid data.

#### 5.2.1.9 Interface/Exports

The main interface of the application gives access to this feature.

#### 5.2.1.10 Detailed Subsystem Design

The subsystem is designed as a validation model to check for the incomplete or missing entries in the user input. Application will not proceed forward with its intended functionality if the data entries are missing and will give an error to correct the action.

### 5.2.2 Connectivity to backend model

#### 5.2.2.1 Classification

The classification of the component is that it is a module.

#### 5.2.2.2 Definition

Traversal of data from interface through prediction model and finally obtaining output.

#### 5.2.2.3 Responsibilities

The graphical user interface links the prediction model and the end user. The user can obtain the results of diabetes diagnosis.

#### 5.2.2.4 Constraints

Only the gradient boosting classifier model can be accessed and data can be passed through it.

#### 5.2.2.5 Composition

This module is composed of the lower level application logic layer only.

#### 5.2.2.6 Uses/Interactions

User makes use of this module indirectly as the data processing is explicitly not visible to the user.

#### 5.2.2.7 Resources

The *pickle* library is the primary resource for this module since it encompasses the file open and load operations.

#### 5.2.2.8 Processing

The function *def runmodel(data):* processes the user input through the gradient boosting model file and finally produces the output as a probability.

#### 5.2.2.9 Interface/Exports

The main interface of the application gives access to this feature.

#### 5.2.2.10 Detailed Subsystem Design

The subsystem is designed to link the gradient boosting model and user input. Basically, the user input in the form of numerical data passes through the model saved as a file and then the output is generated and displayed in the same main interface.

### 5.2.3 Diagnosis as a probabilistic output

#### 5.2.3.1 Classification

The classification of the component is that it is a submodule.

#### 5.2.3.2 Definition

Obtaining and interpreting the end results of the application.

#### 5.2.3.3 Responsibilities

The result of diagnosis is a probability of a person having type 2 diabetes based on his clinical information.

#### 5.2.3.4 Constraints

The application does not a give a definitive indication in the form of ‘Positive’ or ‘Negative’ since there are a range of indicators that may be responsible for diabetes and it is medically inaccurate to do so as well.

#### 5.2.3.5 Composition

This module is composed of the higher-level interface layer only.

#### 5.2.3.6 Uses/Interactions

User presses the ‘Diagnose’ button on the application and gets a response in the form of a probability of having diabetes based on the information entered.

#### 5.2.3.7 Resources

The statement

#### 5.2.3.8 Processing

The statement *model.predict_proba(new_patient)[:, 1]* outputs the prediction result as a decimal value between 1 and 0 so it can be interpreted as a value for probability.

#### 5.2.3.9 Interface/Exports

The main interface of the application gives access to this feature.

#### 5.2.3.10 Detailed Subsystem Design

The system is designed so that it gives a clearly visible output to the user in terms of the probability that the person has type 2 diabetes. A text box displaying the output in clearly understandable language is present on the application interface.

*Chapter 6*

# 6. IMPLEMENTATION AND TESTING

## 6.1 Introduction

The chapter describes how the solution has been implemented and what are the main challenges encountered in development of the system and how the proposed software was evaluated.

## 6.2 System Analysis and Challenges

During the research phase, requirements of the users were categorized into user requirements, system and hardware requirements.

### 6.2.1 Requirement Specifications

After the research phase, number of requirements were formulated namely user requirements, hardware requirements and software attributes. These were grouped as user, functional, non-functional and systems requirements formally in the Software Requirements Specification document.

### 6.2.2 User Requirements

During requirements gathering phase, the requirements were investigated and we explored the need and feasibility of the project along with its operating environment. Not only that but we also tried out which possible problems could the system face and probable solutions.

### 6.2.3 Hardware Requirements

The hardware interfaces required are as follows:

- A PC or laptop with processing power equivalent to core i5 or core i7.
- Communication protocols for internet connectivity

### 6.2.4 Software Requirements

- Windows OS or Ubuntu OS

### 6.2.5 System Development

We deployed a classic machine learning pipeline starting with the loading and pre-processing of the dataset. The development of the prediction model required intensive experimentation; implementing various models from a range of algorithms and then tuning one (or multiple ones) to achieve the highest possible F1 score. For this purpose, we went through a number of steps in the development phase.

- **Load the dataset:**

```
data = pd.read_csv('C:\\Users\\User\Desktop\FYP\input\diabetes.csv')
```

*Figure 13-Loading dataset via Pandas library*

- **Pre-processing of dataset:**

The first thing we did was to see if there were any zero value entries in the dataset. As expected, there were quite a few as can be seen in each field:

| Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeFunction | Age | Outcome |
|---|---|---|---|---|---|---|---|---|
| 6 | 148 | 72 | 35 | **0** | 33.6 | 0.627 | 50 | 1 |
| 1 | 85 | 66 | 29 | **0** | 26.6 | 0.351 | 31 | 0 |
| 8 | 183 | 64 | **0** | **0** | 23.3 | 0.672 | 32 | 1 |
| 1 | 89 | 66 | 23 | 94 | 28.1 | 0.167 | 21 | 0 |
| 0 | 137 | 40 | 35 | 168 | 43.1 | 2.288 | 33 | 1 |
| 5 | 116 | 74 | 0 | 0 | 25.6 | 0.201 | 30 | 0 |
| 3 | 78 | 50 | 32 | 88 | 31 | 0.248 | 26 | 1 |
| 10 | 115 | **0** | 0 | 0 | 35.3 | 0.134 | 29 | 0 |
| 2 | 197 | 70 | 45 | 543 | 30.5 | 0.158 | 53 | 1 |
| 8 | 125 | 96 | **0** | **0** | **0** | 0.232 | 54 | 1 |
| 4 | 110 | 92 | **0** | **0** | 37.6 | 0.191 | 30 | 0 |
| 10 | 168 | 74 | **0** | **0** | 38 | 0.537 | 34 | 1 |
| 10 | 139 | 80 | **0** | **0** | 27.1 | 1.441 | 57 | 0 |

*Figure 14-Zero value entries in dataset*

So, to cater to this problem we used a function *def replace_zero(df, field, target):* to replace the zeroes with mean of the entries of that field grouped by the outcome field, so that the replaced values do not have any effect on the outcome. This function also displayed the number of zero value entries that each feature contained.

```
Field Glucose : num 0-entries: 5
Field BloodPressure : num 0-entries: 35
Field SkinThickness : num 0-entries: 227
Field Insulin : num 0-entries: 374
Field BMI : num 0-entries: 11
Field Age : num 0-entries: 0
Field DiabetesPedigreeFunction : num 0-entries: 0
```

*Figure 15-Output of replace_zero function*

After replacing the zero values the next step was to remove the outliers from the dataset. This was done by the function *def TurkyOutliers(df_out,nameOfFeature,drop=False):* which implemented the interquartile range method. After removing the outliers from the dataset, we lost 141 rows from our data frame.

```
New dataset with removed outliers has 627 samples with 9 features each.
df shape: 768, new df shape: 627, we lost 141 rows, 18.359375% of our data
```

*Figure 16-Output of outliers' removal*

- **Implementing the algorithms and selecting the best model:**

Since our problem falls under the regression category, we ran seven machine learning models on first, the raw dataset and then, on the pre-processed dataset, and immediately calculated their respective F1 scores.

- kNN

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=1, n_neighbors=5, p=2,
           weights='uniform'):
F1 score for training set is: 0.831
F1 score for testing set is: 0.800
```

*Figure 17-F1 score of kNN*

- Logistic Regression

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False):
F1 score for training set is: 0.714
F1 score for testing set is: 0.778
```

*Figure 18-F1 score of LR*

- C4.5 Decision Trees

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=0,
            splitter='best'):
F1 score for training set is: 1.000
F1 score for testing set is: 0.806
```

*Figure 19-F1 score of Decision Trees*

- Random Forest Classifier

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
            oob_score=False, random_state=0, verbose=0, warm_start=False):
Fl score for training set is: 0.979
Fl score for testing set is: 0.844
```

*Figure 20-F1 score of Random Forest*

- AdaBoosting

```
AdaBoostClassifier(algorithm='SAMME.R', base_estimator=None,
          learning_rate=1.0, n_estimators=50, random_state=0):
Fl score for training set is: 0.908
Fl score for testing set is: 0.836
```

*Figure 21-F1 score of AdaBoosting*

- Gaussian Naïve Bayes

```
GaussianNB(priors=None):
Fl score for training set is: 0.750
Fl score for testing set is: 0.794
```

*Figure 22-F1 score of Gaussian Naïve Bayes*

- Gradient Boosting

```
GradientBoostingClassifier(criterion='friedman_mse', init=None,
              learning_rate=0.01, loss='deviance', max_depth=3,
              max_features=None, max_leaf_nodes=None,
              min_impurity_decrease=0.0, min_impurity_split=None,
              min_samples_leaf=1, min_samples_split=2,
              min_weight_fraction_leaf=0.0, n_estimators=310,
              presort='auto', random_state=0, subsample=0.5, verbose=0,
              warm_start=False):
Fl score for training set is: 0.910
Training Accuracy is 0.9452054794520548

Fl score for testing set is: 0.853
Testing Accuracy is 0.8994708994708994
```

*Figure 23-F1 score of Gradient Boosting*

```
Importance of Each feature
[0.03760781 0.14498305 0.02733982 0.07294285 0.51773388 0.06684927
 0.06547279 0.06707054]
```

*Figure 24-Relative Importance of each feature*

- Keras Neural Network

```
Accuracy ---> 0.8518518496442724
```

*Figure 25-Accuracy measure of neural network*

It is to be noted that the algorithms were implemented on the pre-processed dataset for our research and all similar conditions were kept constant, with the exception of gradient boosting, which was put to parameter tuning. The reason being that prediction by gradient boosting was intuitively highly accurate as it had the highest F1 score on raw dataset first. Hence, we deemed it fit for further research.

- **Tuning of Gradient Boosting Algorithm**

As gradient boosting is a boosting algorithm, so it is robust to overfitting. It has an attribute of "*n_estimator*" which defines how many boosting stages or iterations to perform. We used the cross-validation method to find the optimal number of "n_estimators" attribute. The graph below shows the iterations which produce the loss for the respective method used. The x-axis encompasses the number of iterations or boosting stages while the y-axis displays the normalized loss. Comparison of Out-of-bag loss, Cross Validation loss and Test loss was done to find iterations with minimum loss.

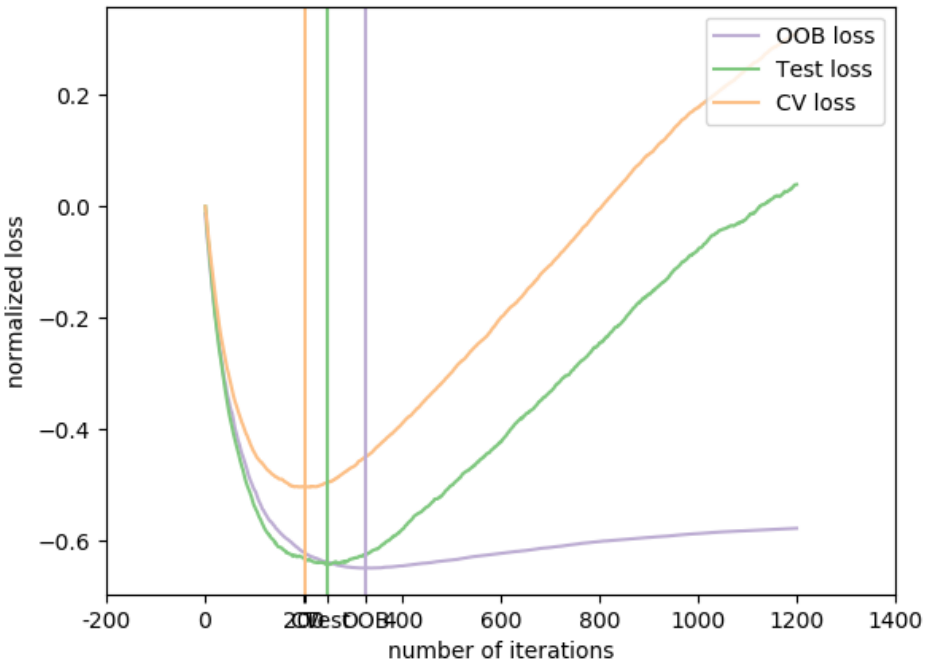


*Figure 26-Finding n_estimators*

The next parameter that we changed was "*max_depth*" which is individual depth of each regression tree involved in the boosting process. Increasing this might cause the model to slightly overfit however by setting it to 3, our F1 score went up 0.040 (0.824 to 0.828).

An interesting discovery was that by dropping the fields of *BloodPressure* and *Pregnancies* from the test set and train set, the F1 score increases by 0.029 (0.824 to 0.853). However, we avoided removing any columns from our data frame to keep it from overfitting. Another insightful discovery was that the algorithm was giving more importance to *SkinThickness* than *BloodPressure.*

- **Save the prediction model as an external file**

The contents of *gbc.fit(X_train, y_train)* and all related libraries are swapped to a binary file using *pickle.dump*. It basically serializes a Python object into a byte stream. This file was later used as part of the *Tkinter* module.

```
y_pred = clf_.predict(X_test)
import pickle
Gradientboost = open("gradientboost.pkl","wb")
pickle.dump(clf_,Gradientboost)
Gradientboost.close()
```

*Figure 27-Export gradient boosting model to external file*

- **Setting up Tkinter GUI module**

*Tkinter* module of Python was setup with the default site packages and libraries. The file *gradientboost.pkl* obtained in the previous step was placed in the home directory of the setup. The user interface was developed in the file *test.py*, including all labels, entries and buttons. The function *runmodel(data)* instantiated the gradient boosting model.

- **Building the executable file**

The command *python setup.py build* was invoked in Windows command line within the current folder of the *Tkinter* setup. This led to the creation of *.exe* file of the prototype in the *build* folder.

## 6.3 Diagrams

### 6.3.1 Basic Modules

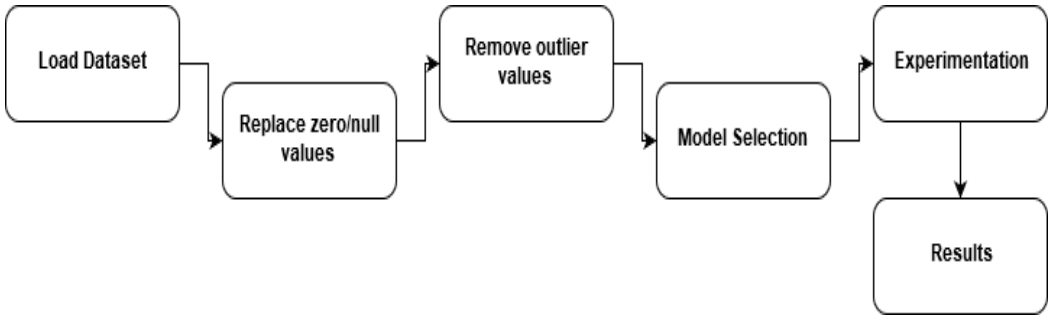


*Figure 28-Module Diagram*

### 6.3.2 Use Case Diagram

A defined use case of the end system is direct interaction with the user for diabetes diagnosis and related information retrieval.

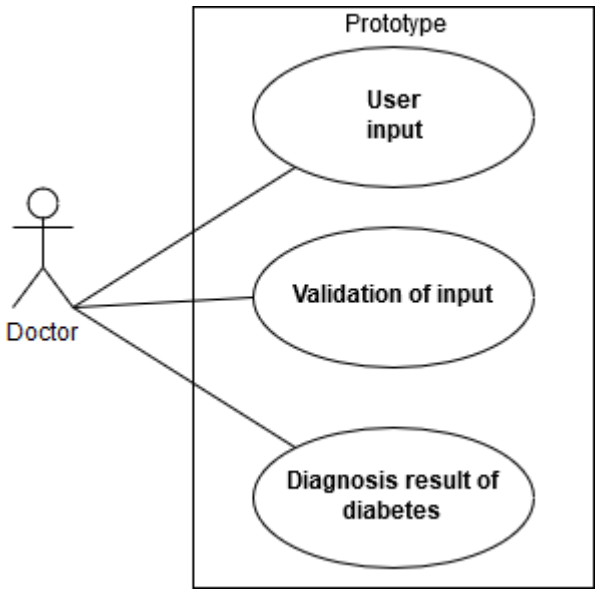


*Figure 29-Use Case Diagram*

## 6.4 Core Functionalities of Prototype

The prototype is developed for testing the results on end users according to Software engineering principles. Following are the core modules of the system:

- User input validation
- Connectivity to backend model
- Diagnosis as a probabilistic output

### 6.4.1 User input validation

The user input to the application is validated to check for any missing entries.

### 6.4.2 Connectivity to backend prediction model

The graphical user interface links the prediction model and the end user. The user can obtain the results of diabetes diagnosis via the interface. The technical details of the link between the model and interface have been explained in Chapter 6.

### 6.4.3 Diagnosis as a probabilistic output

The result of diagnosis is a probability of a person having type 2 diabetes based on his clinical information. The application does not a give a definitive indication in the form of simply ‘Yes’ or ‘No’ since there are a range of indicators that may be responsible for diabetes and it is medically too high a risk to do so as well.

## 6.5 Tools and Techniques

The model is implemented using python and its libraries such as *numpy*, *ensemble*, *sklearn*, *pandas* and *matplotlib* which are amongst the most advanced technologies in machine learning domain. The results are visualized in RStudio and Microsoft Excel. The reason for choosing them is the ease of use and availability of powerful features which make data visualization quite an interesting and meaningful task.

## 6.6 Tools

- JetBrains Pycharm
- RStudio
- WEKA
- OpenRefine

## 6.7 Testing

Testing of the results was done after the system was put in practice. Understanding of the graphical results by medical practitioners and doctors was essential for this phase. The authenticity of the results is guaranteed to a great extent since the dataset is of real patients and originates from a well-established repository. Traditional software testing methods were studied in detail and a subset of them were applied as part of the project. A noticeable component is that the training and test set is from the same dataset with a variation in the percentage split. The reason for this is to engage in successful black-box and white-box testing.

Testing was done mainly in two ways:

### 6.7.1 Validation Testing

Validation testing was carried out on individual features of the system to ensure that they are fully functional units. The outcomes and findings were presented to doctors and medical practitioners from the Community Medicine Department at The Foundation University Medical College, Rawalpindi, Pakistan. 6 out of the 8 professionals were able to fully interact with the system and make use of the results. In fact, they found the research largely appealing and truly believed that the discoveries of a similar nature on a dataset known to them would provide ease in the overall diagnosis process. The success of each individual experiment and outcome

gave us the go ahead to carryout integration testing.

### 6.7.2 Black-box Testing

Black box testing is a software testing method in which the internal structure and design of the system being tested is not known to the tester. The tests were functional in our case. We deployed equivalence partitioning method which involved dividing input values of the dataset of different parameters into valid and invalid partitions (dependent upon the algorithm being used) and then selecting representative values from each partition as test data. Developers were involved in the testing phase as well. It was a result-focused phase and emphasis was on the difference between expected and actual outcome.

*Chapter 7*

# 7. RESULTS AND DISCUSSION

This section discusses the end results of the proposed system, along with graphical displays and visualizations of the dataset and its features. We have focused on providing comparative analysis of the results of the machine learning experiments via various graphs and charts. Screenshots of the console and charts showing the F1 scores are also included. Different technical and logistic problems faced during the whole project development process have been highlighted as well.

## 7.1 Results

### 7.1.1 Visualization of dataset

As explained in Chapter 6, we first engaged in data visualization of the individual features present in the dataset. The purpose of this activity was to get a better and visual understanding of the attributes, and possibly determine their statistical characteristics. Each graph displays all the data points for that individual feature mentioned in the title chart. The x-axis encompasses the number of records (number of people of whose indicators are present) while the y-axis has the values for that respective feature.

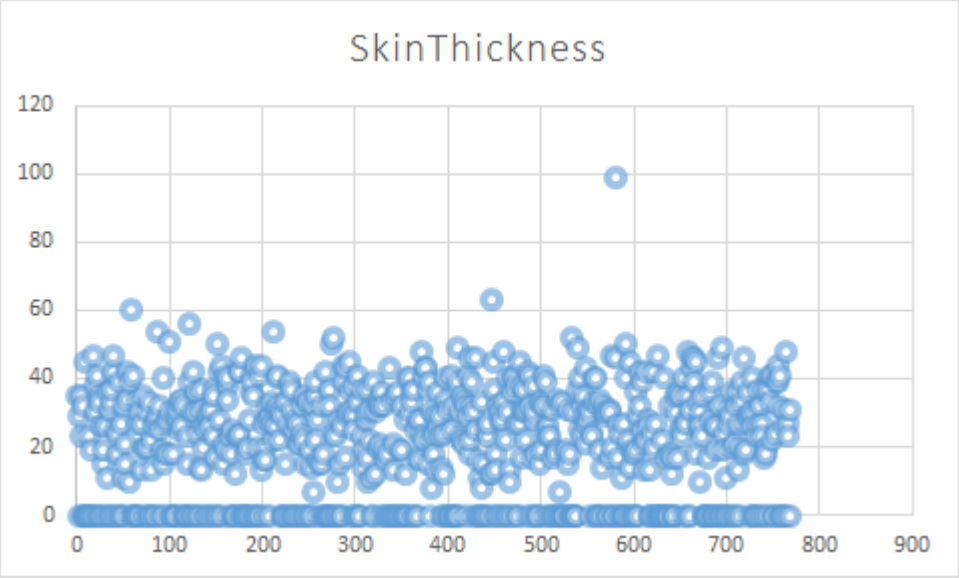


*Figure 30-Visualization for SkinThickness*

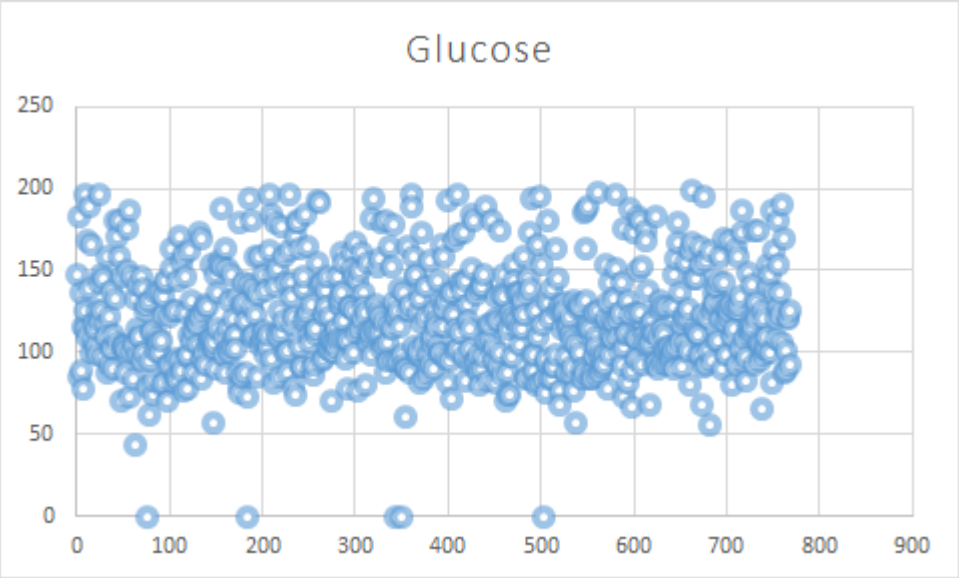


*Figure 31-Visualization for Glucose*

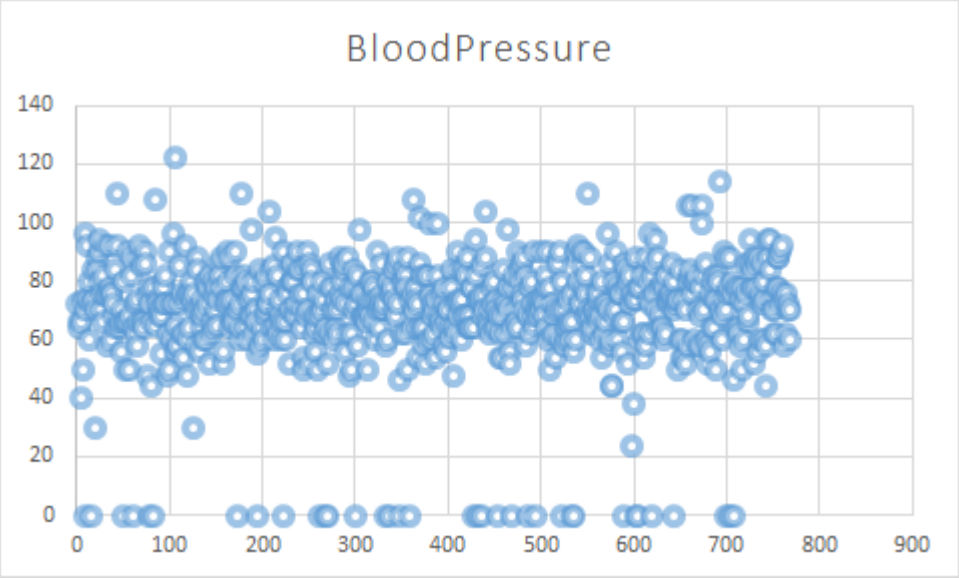


*Figure 32-Visualization for BloodPressure*

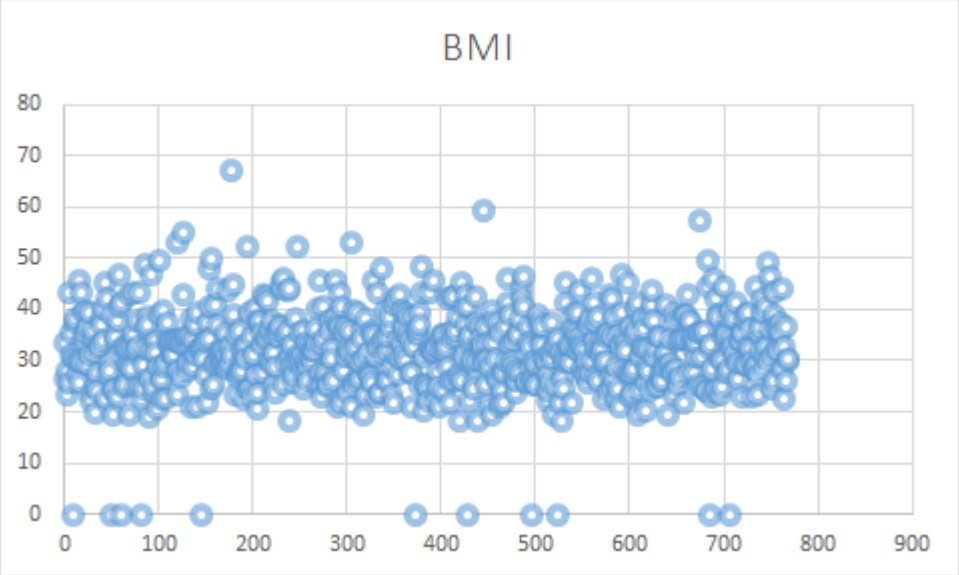


*Figure 33-Visualization for BMI*

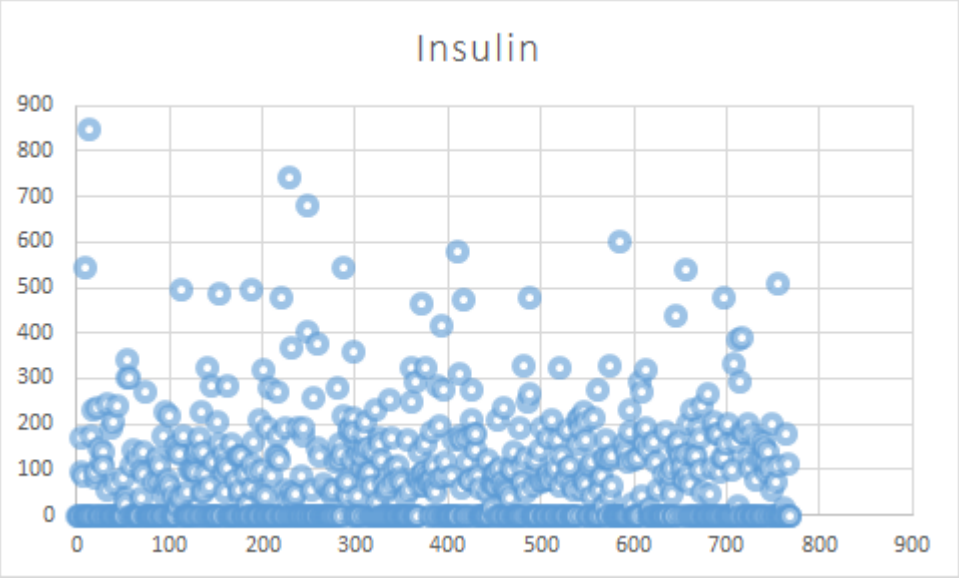


*Figure 34-Visualization for Insulin*

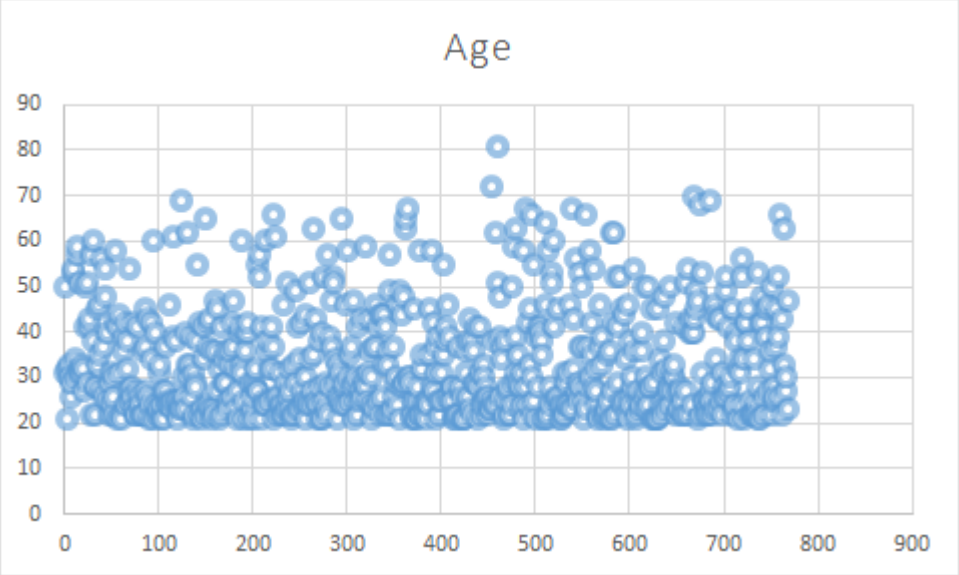


*Figure 35-Visualization for Age*

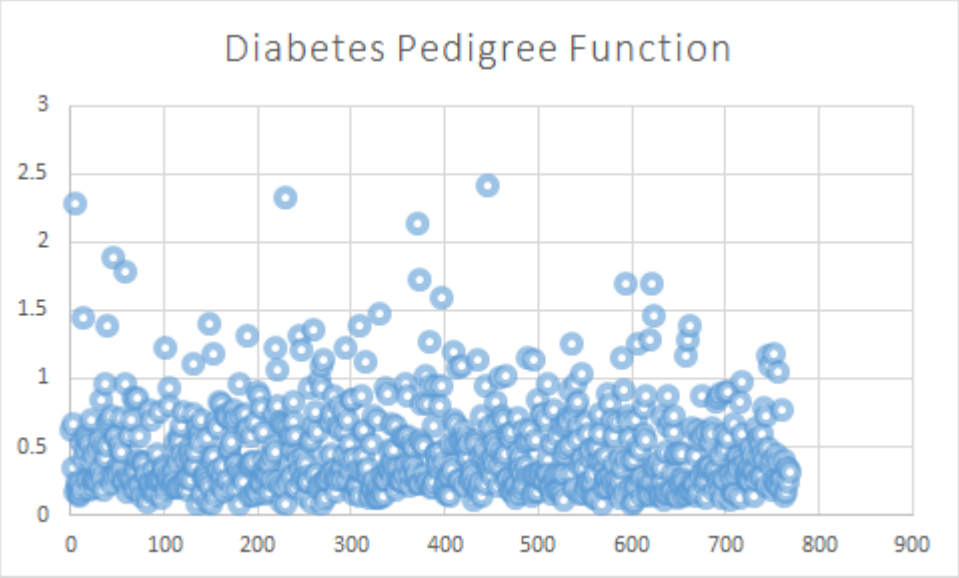


*Figure 36-Visualization for DiabetesPedigreeFunction*

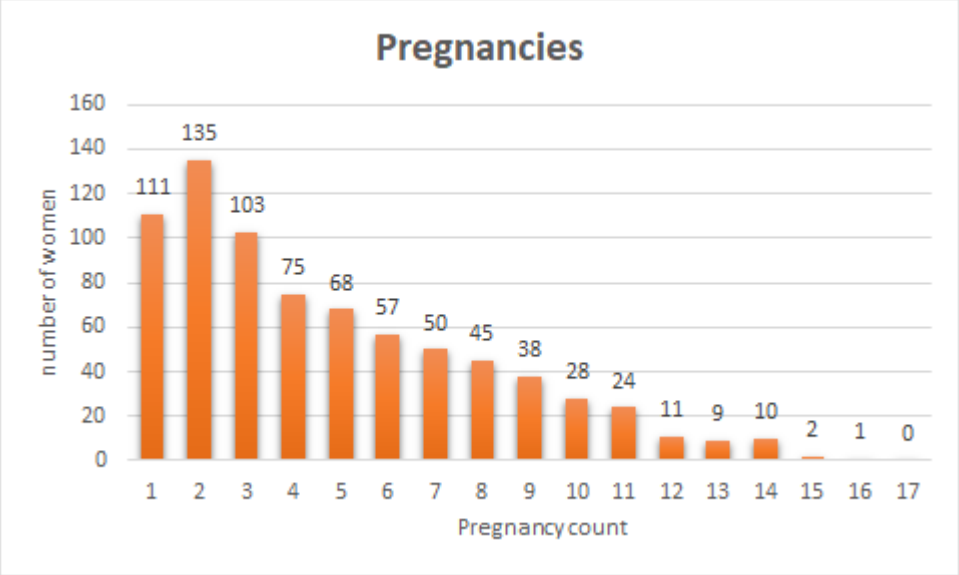


*Figure 37-Visualization for Pregnancies*

### 7.1.2 Visualization of Outliers

The next immediate step was to visualize the outlier data points for each attribute contained in the dataset. The plots below show the outliers for each field of the dataset. A key is appended with each plot for better understanding.

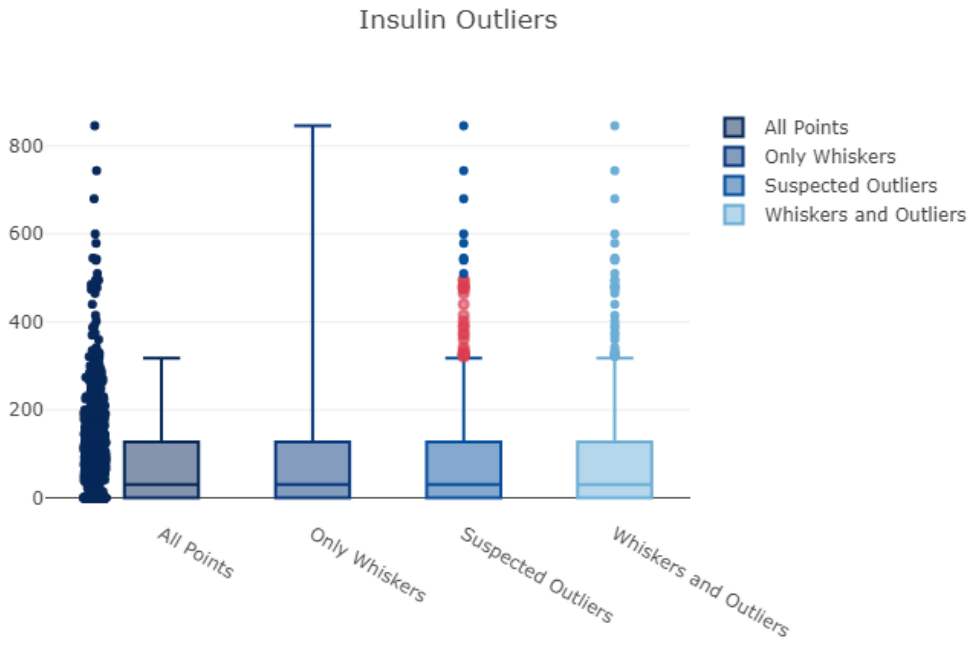


*Figure 38-Outliers for Insulin*

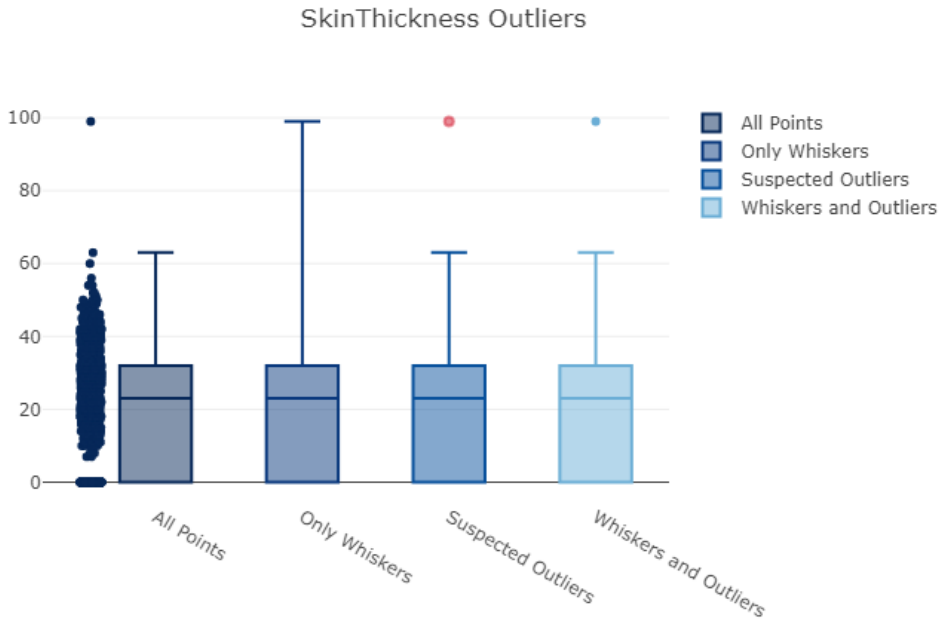


*Figure 39-Outliers for SkinThickness*

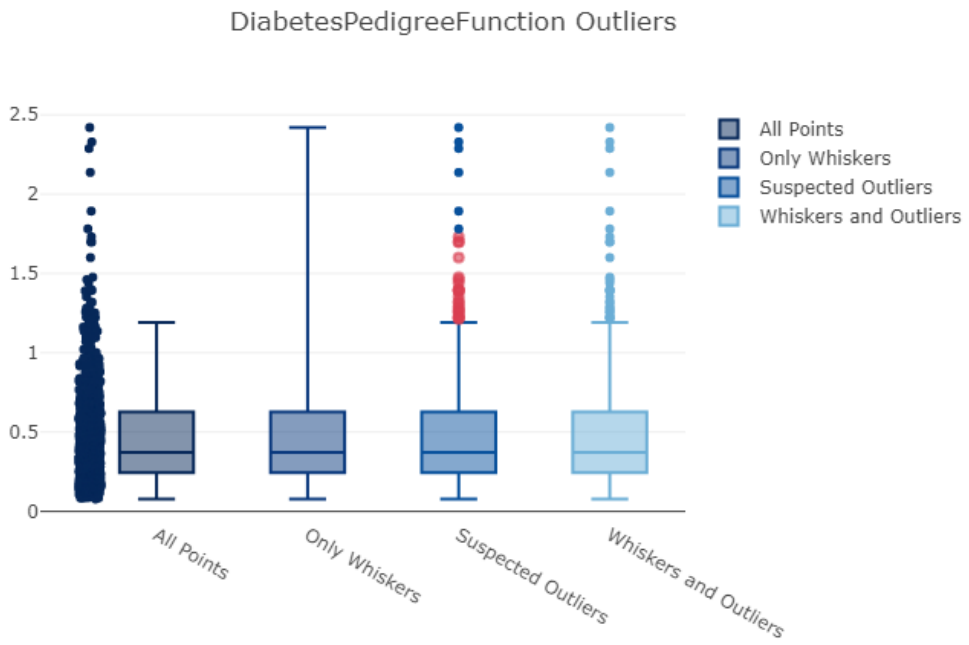


*Figure 40-Outliers for DiabetesPedigreeFunction*

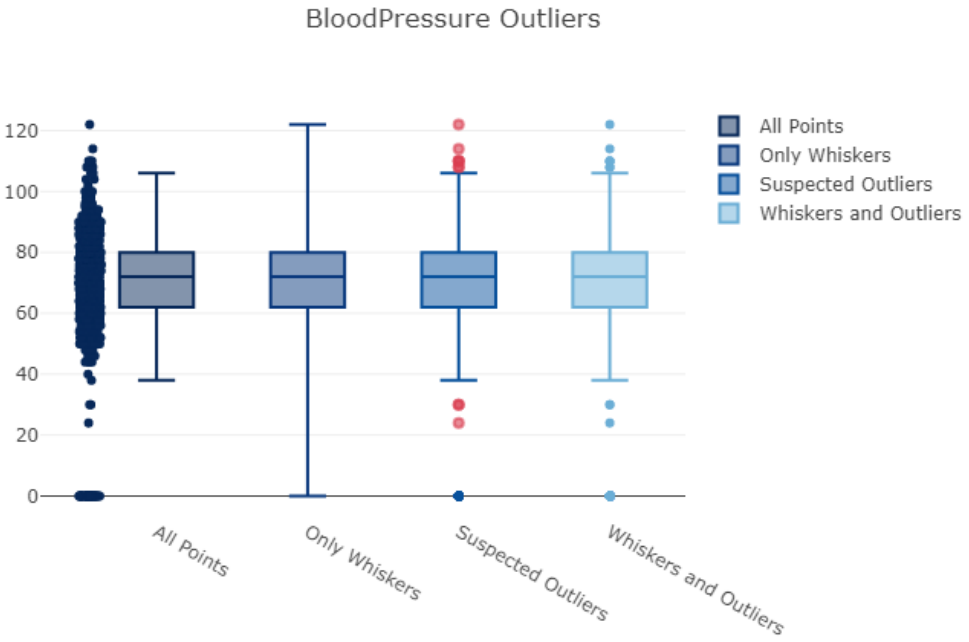


*Figure 41-Outliers for BloodPressure*

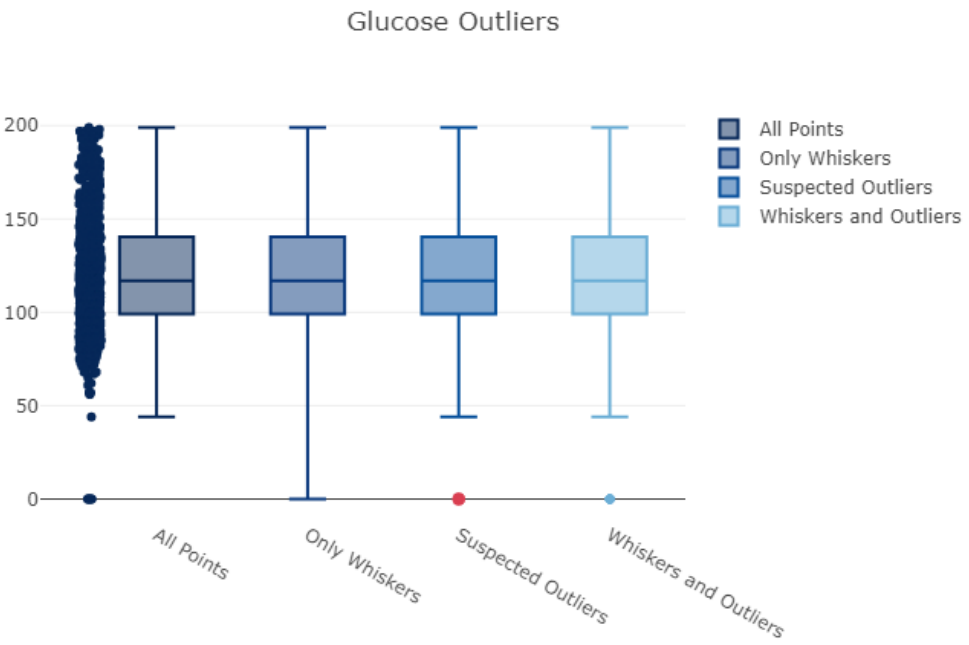


*Figure 42-Outliers for Glucose*

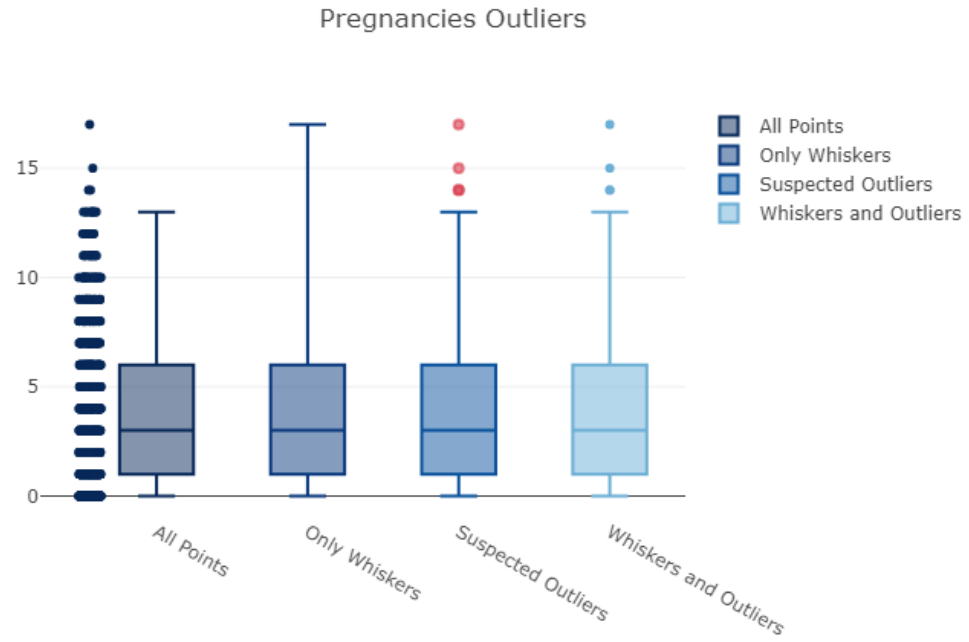


*Figure 43-Outliers for Pregnancies*

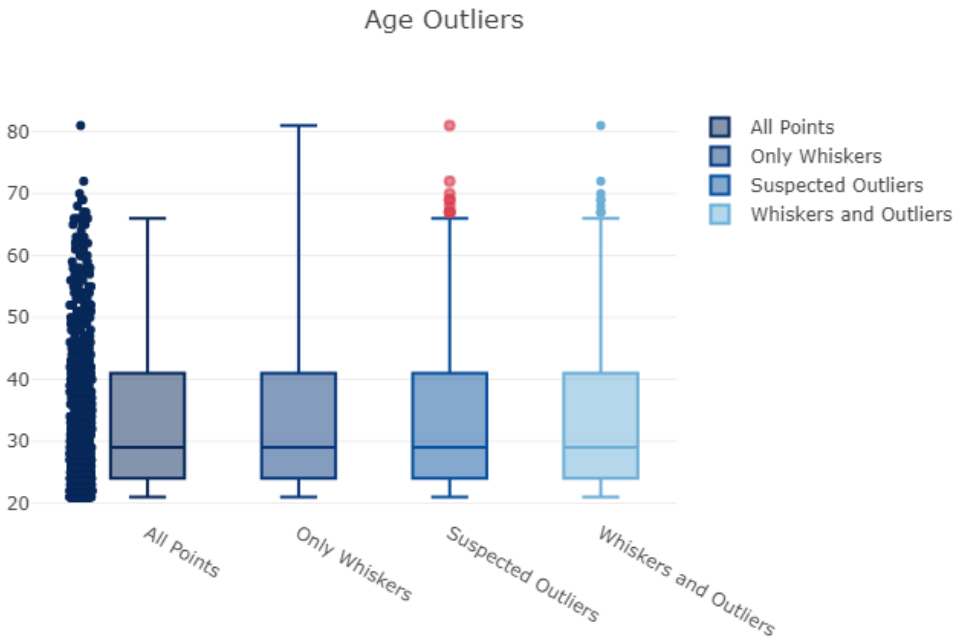


*Figure 44-Outliers for Age*

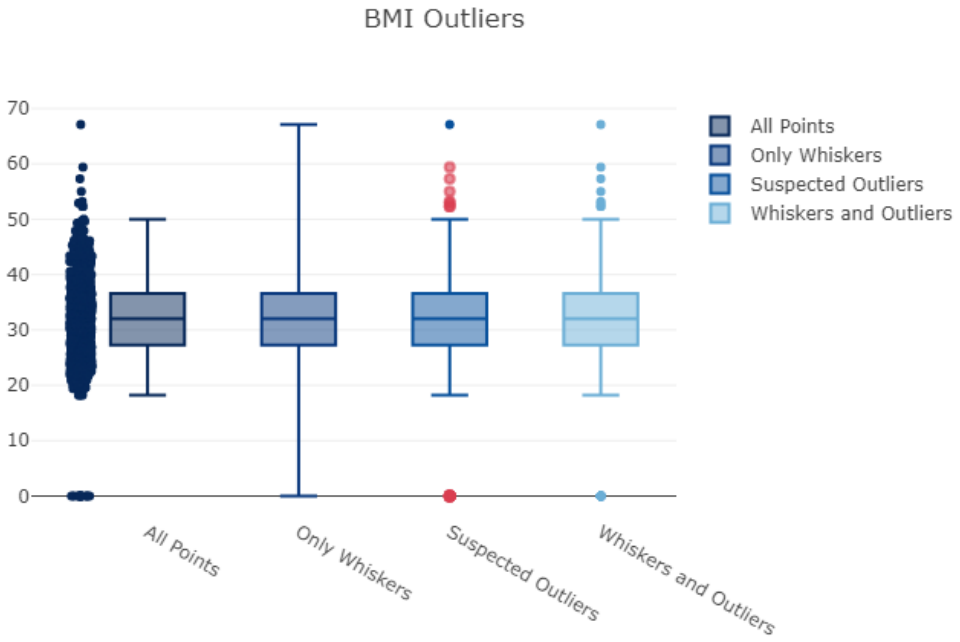


*Figure 45-Outliers for BMI*

### 7.1.3 Summary of prediction models' results

The tabular summary shows the evaluation of the various prediction models used in our research. It is to be noted that in the column, F1 Score (A), the F1 scores are obtained from the test set of raw dataset. The models are trained and tested using 70:30 percentage split method to evaluate the regression result. The last column, F1 Score (B), provides F1 scores on pre-processed dataset with parameter tuning.

| Prediction Model | F1 Score (A) | F1 Score (B) |
|---|---|---|
| k Nearest Neighbour | 0.807 | 0.800 |
| Logistic Regression | 0.775 | 0.778 |
| Decision Tree | 0.772 | 0.806 |
| Random Forest | 0.793 | 0.844 |
| AdaBoosting | 0.790 | 0.836 |
| Gaussian Naïve Bayes | 0.640 | 0.794 |
| Keras Neural Network | 0.760 | 0.851 |
| Gradient Boosting | 0.824 | 0.853 |

*Table 1-Prediction models with F1 scores*

### 7.1.4 Experimentation

Different experiments were conducted with the dataset, to check for the relative contribution and the effect of various features on the outcome of diagnosis. These phenomena were realized with the respective F1 scores for the experiments.

- In the first experiment, each field was used one by one to predict the outcome. The F1 score of the outcome is represented on the y-axis while the predictors are represented on the x-axis.

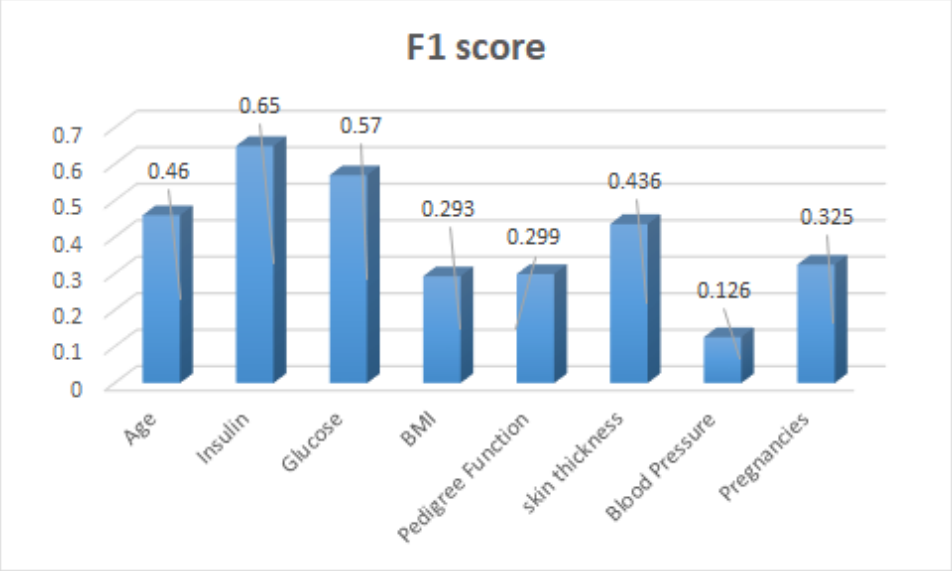


*Figure 46-F1 score per feature*

- In another experiment, fields were dropped i.e. excluded from the test set and train set to observe how the outcome would be affected. The F1 score of the outcome is represented on the y-axis while the x-axis represents the indicators which were dropped in each particular experiment.

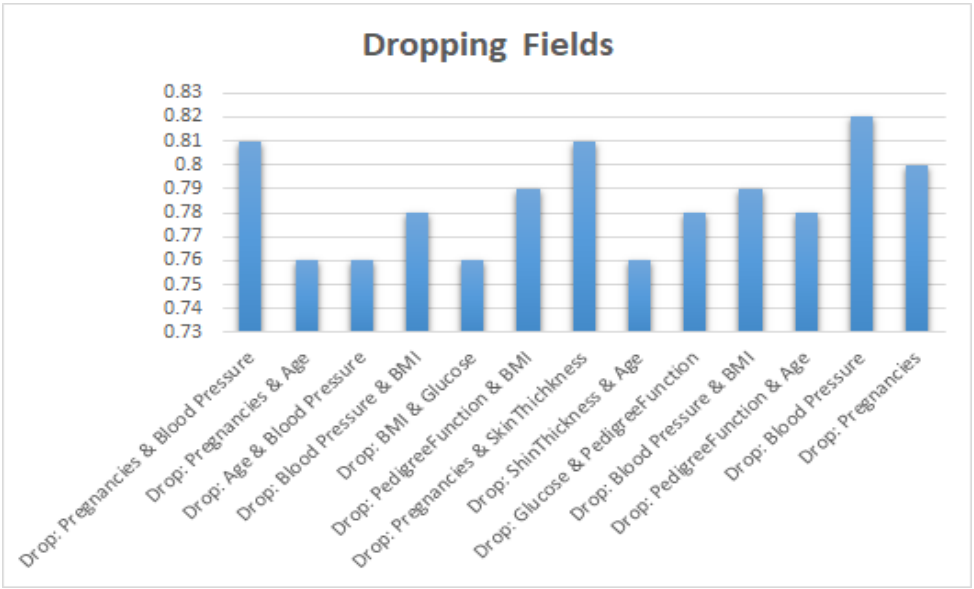


*Figure 47- Effect on F1 score after dropping fields*

- Lastly, an experiment was conducted to check for the total number of entries required to functionally train the model. The y-axes represent the F1 scores for the training set while the x-axis represent the number of records or entries.

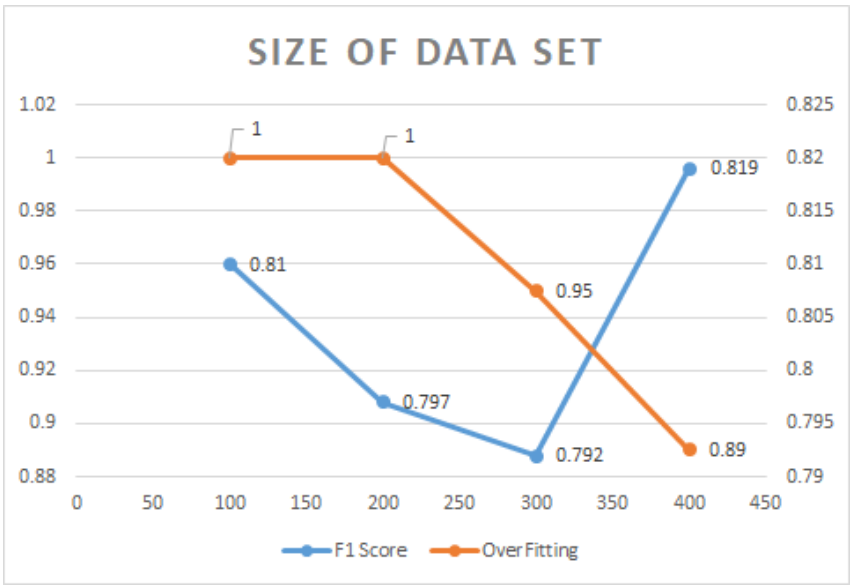


*Figure 48- Number of records required to train the model*

### 7.1.5 Application Results

To test our results, we developed a prototype application for displaying the patient's possibility of diabetes as a percentage, which the user can interpret meaningfully. The user can enter the values of indicators in the input fields. They are then run through the pre-trained model and a final output is obtained.

- After entering the data, validation of input data takes places by pressing the 'Validate' button to check for any missing entries.

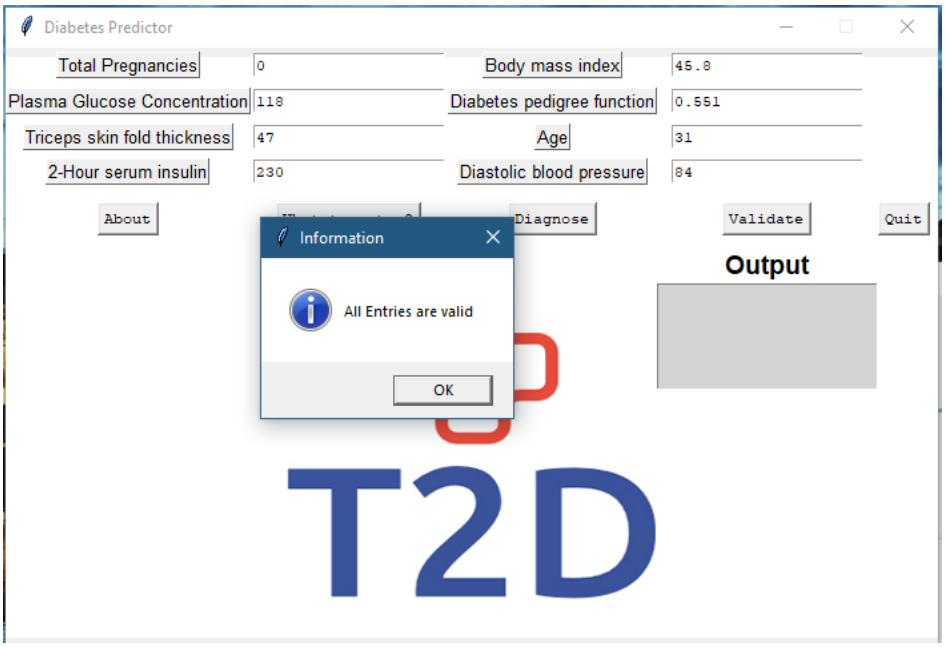


*Figure 49-Interface of prototype application*

- The final output is obtained by clicking on the 'Diagnose' button.

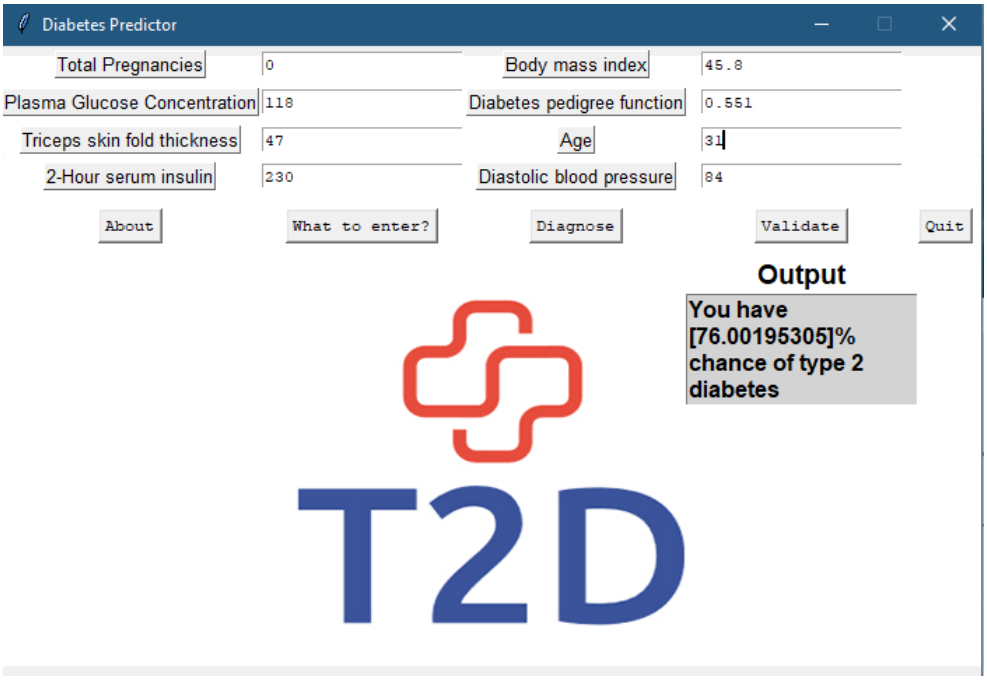


*Figure 50-Diagnosis Output*

## 7.2 Problems Faced

While working on this project, we faced several problems.

- Data Collection from local sources:

  As stated in the Requirements Specification document, we faced problems in collecting clinical information from hospitals and clinics in Pakistan. These sources were reluctant to provide the required data and they confronted us with a harsh attitude. The samples of paper-based records that we did manage to obtain proved to be unusable. This was a primary problem of this project that we managed to eliminate by using Kaggle dataset.

- Setting up python frameworks and libraries:

  Setting up PyCharm and all the dependencies, frameworks and libraries in Python was taxing and troublesome since it was time consuming and often there were compatibility issues. Troubleshooting these errors and setting up a development environment was intricate.

*Chapter 8*

# 8. CONCLUSION AND FUTURE WORK

## 8.1 Conclusion

The automated diagnosis of diabetes is an important real-world medical problem. Detection of diabetes in its early stages is the key for treatment. Our project shows how Gradient Boosting model contributes to the actual diagnosis of Type 2 diabetes for local and systematic treatment, along with presenting related work in the field. Experimental results show the effectiveness of the proposed model with an out of sample accuracy of 89.9% and F1 score of 0.853. The performance of the different techniques was investigated for the diabetes diagnosis problem. Experimental results demonstrate the adequacy of the proposed model.

## 8.2 How does this project impact our society?

By transforming various health records of diabetic patients to useful analyzed results, this analysis will make the patients understand the complications to occur. Patient lifestyle can be affected due to this in a drastic way. Moreover, it will provide the doctors with metrics like most or least probable cause of diabetes in a certain human being. Hospitals and clinics are the major stakeholders of this research, and through testing the results, we believe that this research segment will prove beneficial for them. The end system with the underlying predictive model we have proposed may be deployed in regions where medical facilities are limited.

## 8.3 How does this project improve our current understanding?

By engaging in research in this field, we have gained sufficient understanding and new perspective of a real-life problem. We got to deeply understand and improve the Gradient boosting algorithm, interpret its parameters, analyze similar algorithms, and lastly define the architecture and the ecosystem of

the prediction model. We also got an understanding of the domain of machine learning in practice and of its vast applications. We learned how to perform experimental operations by manipulating data points, input values, parameters etc. and how to deploy the model in a fully functional development environment. The whole process of obtaining and improving the results and providing visualizations alongside was an opportunistic challenge.

## 8.4 Recommendations

The domain of Predictive Analytics in Healthcare specifically for prediction of diabetes has produced some discoveries in the past but there is significant area of improvement and an abundance of research that needs to be done. We would highly recommend computer scientists and engineers to work in this domain and contribute what they can in the area of data science and machine learning.

## 8.5 Future Work

There is always room for improvement in technology and we believe in evolution of a system resulting in improved efficiency of a design. Moreover, there is much to be discovered and explored in the domain of machine learning in healthcare. Lastly, ways to meet the modern needs of the industry have to be improved over a certain period of time.

Following are the future recommendations that can be held as a ground to work upon:

- Moving towards a market ready product for diabetes diagnosis
- Research into datasets for local regions
- Research into more intricate features which affect the prevalence of diabetes
- Build security measures of database of medical records
- An interactive graphical user interface for the system
- Automated data visualization
- Optimization of the system

*Chapter 9*

# 9. REFERENCES


[1] Aiswarya Iyer et al. Jeyalatha and Ronak Sumbaly, Diagnosis of diabetes using classification mining techniques, (2015)

[2] Mostafa Fathi Ganji and Mohammad Saniee Abadeh, “Using fuzzy Ant Colony Optimization for Diagnosis of Diabetes Disease”, Proceedings of ICEE 2010, May 11-13, 2010

[3] T. Jayalakshmi and Dr. A. Santhakumaran, “A Novel Classification Method for Diagnosis of Diabetes Mellitus Using Artificial Neural Networks”, International Conference on Data Storage and Data Engineering, 2010, pp. 159-163

[4] Keating, B.J. Advances in Risk Prediction of Type 2 Diabetes: Integrating Genetic Scores with Framingham Risk Models. (2015) Diabetes 64(5): 1495-1497.

[5] S. Vijayarani and S. Dhayanand, “Data mining classification algorithms for kidney disease prediction,” Int. J. on Cyber. & Informatics, Vol. 4, pp. 13-25, August 2015.

[6] E. Alickovic and A. Subasi, “Medical decision support system for diagnosis of heart arrhythmia using DWT and random forests classifier,” Patient Facing System, J Med Syst, Vol. 40, pp. 108-120, 2016.

[7] Emrana Kabir Hashi, Md. Shahid Uz Zaman and Md. Rokibul Hasan, An Expert Clinical Decision Support System to Predict Disease Using Classification Techniques, February 2017

[8] Gauri D. Kalyankar Shivananda R. Poojara Nagaraj V. Dharwadkar, Predictive Analysis of Diabetic Patient Data Using Machine Learning and Hadoop, 2017

[9] Mohamed, Ehab I., et al. "Predicting Type 2 diabetes using an Electronic nose-based artificial neural network analysis." Diabetes, nutrition & metabolism 15.4 (2002): 215-221.

[10] Meo SA, Inam Zia, Ishfaq A Bukhari, Shoukat Ali Arain, Type 2 diabetes mellitus in Pakistan: Current prevalence and future forecast, Journal of the Pakistan Medical Association 66(12):1637-1642 · December 2016

[11] Guo, Yang, Guohua Bai, and Yan Hu. "Using Bayes Network for Prediction of Type-2 Diabetes." Internet Technology and Secured Transactions, 2012 International Conference for. IEEE, 2012.

[12] 2. Shankaracharya et al. Java-based diabetes type 2 prediction tools for better diagnosis. (2011) Diabetes Technol Ther14 (3): 251–256.

[13] Chandvanya JR, Aluvalu R. Ranking with distance-based Outlier Detection Techniques: A survey. International Journal of Computer Applications. 2014; 89(6):8–11.

[14] Logistic Regression - An Overview. Available from: http:// www.math.umt.edu/graham/stat452/logistic.pdf

[15] Goteti VS, Pannucci M, Zhang J. Logistic Regression Analysis of the occurrence of diabetes in pima Indian Women. 2004. Available from: http://www.rci.rutgers.edu/~cabrera/587/pima.pdf

[16] Gorunescu F. Data mining concepts, models and techniques. Intelligent Systems Reference Library. Berlin Heidelberg, Springer-Verlag: 2011. p. 256–60.

[17] Mafuratidze E, Chako K, Phillipo H, Zhou DT. Over 27% of Type 2 diabetic patients studied at Parirenyatwa Diabetic Clinic in Zimbabwe have evidence of impaired renal function. International Journal of Scientific and Technology Research. 2014 Mar; 3(3):14–8.