

---

# Implementing Diabetes Prediction using Various Classification Algorithm in Machine Learning

---

Saturday, 27.08.2022

**Thesis Submitted to -**  
**Exposys Data Labs**

**Internship Report by**

**Abhishek Karmakar**  
**M.Tech - Indian Institute of Information Technology, Allahabad**

---

# Abstract

Diabetes is a group of metabolic disorders in which there are high blood sugar levels over a prolonged period. Symptoms of high blood sugar include frequent urination, increased thirst, and increased hunger. If left untreated, diabetes can cause many complications. Acute complications can include diabetic ketoacidosis, hyperosmolar hyglycemic state, or death. Serious long-term complications include cardiovascular disease, stroke, chronic kidney disease, foot ulcers, and damage to the eyes.

Diabetes is a type of chronic disease which is more common among the people of all age groups. Predicting this disease at an early stage can help a person to take the necessary precautions and change his/her lifestyle accordingly to either prevent the occurrence of this disease or control the disease.

We will be taking the Pima Indians diabetes dataset and pre-process it with an objective to employ machine learning for prediction of diabetes. We will use 8 Machine Learning algorithms like Naive Bayes, Decision Tree, Random Forest, XgBoost, Support Vector Machine (SVM ), Logistic regression , K-Nearest Neighbors (KNN) and Neural Network and apply these algorithms on the dataset to predict the disease. Then we will compare the result of these algorithms according to classification metrics so that best out of these can be found out. The Pima indian dataset has 8 features and one output column which is discrete in nature (0/1).

For Naive Bayes Model, the accuracy is 79.22%, Decision Tree is 72.07%, Random Forest is 82.46%, XgBoost is 75.97%, SVM is 75.97%, Logistic Regression is 82.46%, KNN is 81.16% and Neural Network is 75.32%.

# Table of Contents

## 1. Introduction

- 1.1) Objective
- 1.2) Details About the Dataset
- 1.3) Proposed Overall Architecture - Proposed Word
  - A) Data Collection
  - B) Preprocessing
  - C) Machine Learning (Data Modeling)
  - D) Evaluating Different Models
  - E) Result
- 1.4) Introduction to Evaluation Matrix

## 2. Existing Methods

## 3. Proposed Methods with Architecture

- 3.1) Naive Bayes
- 3.2) Decision Tree
- 3.3) Random Forest
- 3.4) XgBoost
- 3.5) Support Vector Machine
- 3.6) Logistic Regression
- 3.7) K-Nearest Neighbors
- 3.8) Neural Network
- 3.9) Best Model Selection

## 4. Methodology

- 4.1) Dataset Collection
- 4.2) Exploratory Data Analysis

- 4.3) Data Visualization**
- 4.4) Data Preprocessing**
- 4.5) Model Building**
- 4.6) Evaluation of Models**

## **5. Implementation**

- 5.1) Setting up the Environment**
- 5.2) Important Libraries and their versions**
- 5.3) Explanation**
  - 5.3.1) Load the dataset**
  - 5.3.2) Data Manipulation and Analysis**
  - 5.3.3) Data Splitting**
  - 5.3.4) Model Training and Testing**

## **6. Conclusion**

- 6.1) Result and Comparison Chart**
- 6.2) Future Work**

# Chapter 1

## Introduction

Diabetes mellitus is a group of metabolic disorders due to less insulin in the blood and it can also create several kinds of different diseases as well. Warning signs of this diabetes result in more hunger, feeling thirsty, weight loss and if not medicated it will lead to death sometimes. There are three types of diabetes Type1, Type2 and Type3. Type1 is due to the failure of pancreas that produces little or no insulin, Type2 when the whole body either doesn't produce enough insulin or it resists insulin and Type3 occurs when neurons in the brain are unable to respond to insulin. In these type2 is the most common type, nearly 90% cases of total diabetes patients.

Diabetes mellitus is a group of metabolic disorders due to less insulin in the blood. Diabetes can create several kinds of different diseases as well. Diabetes can cause a worldwide health care crisis because according to some research around 600 million people will be affected by this disease by the end of 2035. It can lead to several disorders for example urinary organ diseases, blindness etc. So patients need to go to the hospital to get reports after consulting with the doctor and sometimes it will be time consuming so by using machine learning and deep learning methods we can have an answer for this issue. The aim of this project is to predict type2 diabetes at an early stage so that it can lead to improved treatment for a patient. So in this study we used Naive Bayes, Decision Tree, Random Forest, XgBoost, Support Vector Machine (SVM), Logistic regression, K-Nearest Neighbors (KNN) and Neural Network to predict diabetes.

## 1.1 Objective

We will try to build a machine learning model to accurately predict whether or not the patients in the dataset have diabetes or not. Early prediction of type2 diabetes is tough for doctors due to interdependence on various factors such as insulin level, skin thickness, blood pressure, pregnancies, glucose level, BMI, diabetes pedigree and age so using machine learning we will develop a system which can perform early prediction of type2 diabetes for a patient with a higher accuracy using different machine learning algorithms. This project aims to predict type2 diabetes using 7 Machine Learning algorithms and one deep learning algorithm namely Neural Network. It has been found that Random forest algorithm demonstrated the highest accuracy of 82.46 % of accuracy.

## 1.2 Details About the Dataset

The datasets consist of several medical predictor variables and one target variable, Outcome. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

- Pregnancies: Number of times pregnant
- Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- BloodPressure: Diastolic blood pressure (mm Hg)
- SkinThickness: Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin ( $\mu$ U/ml)
- BMI: Body mass index ( $\text{weight in kg}/(\text{height in m})^2$ )
- DiabetesPedigreeFunction: Diabetes pedigree function
- Age: Age (years)
- Outcome: Class variable (0 or 1)

Number of Observation Units: 768

Variable Number: 9

## 1.3 Proposed Overall Architecture

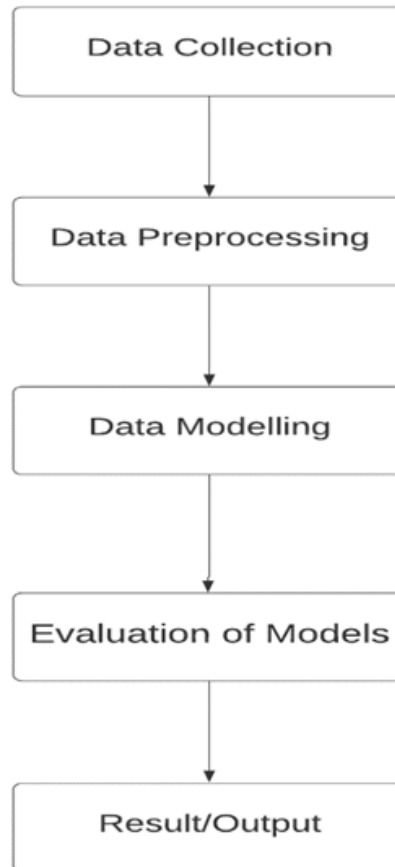


Figure 1.1: Overall Architecture

### **Proposed Work-**

#### **A) Dataset collection**

We will use the Pima Indians dataset which has 9 features.

#### **B) Preprocessing**

We will preprocess the raw data into an understandable dataset so that we can understand positive class (Diabetes=true(1)) and negative class (Diabetes=false(0)).

After that divide each features of the dataset into training, cross-validation and testing set in ratio 80%:20%

### C) Machine Learning (Data Modeling)

In this we will train 8 models (Naive Bayes, Decision Tree, Random Forest, XgBoost, Support Vector Machine (SVM ), Logistic regression , K-Nearest Neighbors (KNN) and Neural Network) by applying training sets (80%) and (20%) data will be used to test the model.

### D) Evaluating different models

We will define classification metrics or confusion metrics to evaluate the model. It is a table by which we can evaluate the performance of our model.

#### Classification metrics-

- 1) True positive if (predicted=positive and actual=positive)
- 2) False positive if (predicted=positive and actual=negative)
- 3) True negative if (predicted=negative and actual=negative)
- 4) False negative if (predicted=negative and actual=positive)

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

$$\text{Fscore} = (2 * \text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

$$\text{Specificity} = \text{TN} / (\text{FP} + \text{TN})$$

### E) Result-

On comparing these models on the basis of the recall, precision, Fscore, Accuracy we will select the model which has highest accuracy, a greater F1 score because a classifier of high precision and high recall is a good classifier.




## 1.4 Introduction to Evaluation Matrix

Evaluation metrics explain performance of the model. We compare the actual values in the test set with predicted values by the model to calculate the accuracy. Evaluation metrics provide a key role in analyzing and development of models so that we can improve our model. By using confusion matrix, models ability can be shown to correctly separate the classes. Each confusion matrix row shows actual labels in test set and columns show the predicted label by the classifier. We can interpret four boxes of matrix as count of true positives, false negatives, true negatives and false positives. Based on these values we can calculate precision, recall, f1 score and accuracy. Recall is TP divided by total actual positives  $[TP/(TP+FN)]$ , Precision is TP divided by total predicted positives  $[TP/(TP+FP)]$ , F-score is weighted average of recall and precision  $[(2*Recall*Precision)/(Recall+Precision)]$ , accuracy is totally correctly predicted divided by total values  $[(TP+TN)/(TP+TN+FP+FN)]$ . So comparison between models will be made on the basis of these values.

# Chapter 2

## Existing Methods

- [1] Lejla Alic et al. elaborates a prediction model using support vector machine algorithm for predicting type2 diabetes disease.
- [2] Rahul Pradhan et al. elaborates prediction of diabetes using logistic regression, Knn, SVM, decision tree in which SVM and decision tree produces best result out of these with higher accuracy and a greater F1 score.
- [3] Muhammad Azeem Sarwar et al. elaborates prediction of diabetes on pima Indian dataset using logistic regression, knn, SVM and random forest in which SVM and Knn produces best result with higher accuracy.
- [4] Priyanka Sonar et al. presented algorithms including decision tree, Support vector machine, Naïve bayes and artificial neural network in which decision tree and SVM produces best results with higher accuracy for the prediction of diabetes.
- [5] Rahul Barhate et al. presented various classifiers used include logistic regression, decision tree, knn, SVM, random forest and neural network on Pima Indian dataset and for optimum performance various parameters of the dataset varied. From the result it was seen that random forests performed well but the accuracy difference between them is minimal.
- [6] Naveen Kishore G et al. presented various classifiers such as SVM, Decision tree, KNN, Logistic regression and random forest on pima Indian dataset out of these Random forest and SVM produces better results .
- [7] Deepti Sisodia et al. discussed prediction of diabetes using three algorithms including Decision tree, SVM and Naïve bayes out of these naïve bayes performs better with higher accuracy.



[8] Chinmayi Chitnis et al. discussed prediction of diabetes using five algorithms including Artificial neural network, Logistic regression, Knn , decision tree and random forest out of these Ann performs better with an accuracy of 80.2 %. We will be predicting whether the patient has diabetes or not on the basis of the features we will provide to our machine learning model, and for that, we will be using the famous Pima Indians Diabetes Database.

# Chapter 3

## Proposed Method with Architecture

We will be predicting whether the patient has diabetes or not on the basis of the features we will provide to our machine learning model, and for that, we will be using the famous Pima Indians Diabetes Dataset. The steps that we follow, basically starts with EDA (Exploratory Data Analysis), Data Visualization, Data Preprocessing and lastly model training. The figure shown below shows the proposed block diagram for the procedure to approach the classification problem. In a total of 8 models used for training purposes, Random forest showed the best outcome of 82.46% accuracy rate, reason could be as its a bagging technique.

The aim of this project is to predict type2 diabetes at an early stage so that it can lead to improved treatment for a patient. Algorithms used namely Naive Bayes, Decision Tree, Random Forest, XgBoost, Support Vector Machine (SVM ), Logistic regression , K-Nearest Neighbors (KNN) and Neural Network and the algorithm with highest accuracy and f score will be selected.

Machine learning can be used to recognise complex patterns in the data and make meaningful relationship so that it can be very useful in predicting several diseases. Several machine learning algorithms we used in this paper such as Naive Bayes, Decision Tree, Random Forest, XgBoost, Support Vector Machine (SVM ), Logistic regression , K-Nearest Neighbors (KNN) and Neural Network for predicting type2 diabetes.

### 3.1 Naive Bayes

**Principle of Naive Bayes Classifier:** A Naive Bayes classifier is a probabilistic machine learning model that's used for classification tasks. The crux of the classifier is based on the Bayes theorem.

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

Figure 3.1: Bayes Theorem

In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features (see Bayes classifier). They are among the simplest Bayesian network models,<sup>[1]</sup> but coupled with kernel density estimation, they can achieve high accuracy levels.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem.

Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

The confusion matrix, accuracy and the classification report are as follows.

```
[[93 14]
 [18 29]]
0.7922077922077922
```

	precision	recall	f1-score	support
0	0.84	0.87	0.85	107
1	0.67	0.62	0.64	47
accuracy			0.79	154
macro avg	0.76	0.74	0.75	154
weighted avg	0.79	0.79	0.79	154

Figure 3.2: Classification Report of Naive Bayes

## 3.2 Decision Tree

They are built by splitting training data into distinct nodes where one node contains all of one category data. Branching of cases is made from the result of testing an attribute and each leaf node holds a class label. Decision tree can be constructed by considering attributes one by one. Process starts with choose an attribute from the data set and calculate the significance of an attribute in splitting of data, next split data based on the value of best attribute then go to each branch and repeat for rest of the attributes in the dataset. Method uses recursive partitioning to split data into segments by minimizing the impurity at each step and impurity is calculated by entropy of data. Entropy is the amount of randomness in the data. In decision trees the best tree is which has the smallest entropy in their nodes. If the samples are homogeneous then entropy is 0 and if the samples are equally divided then the entropy is 1.

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

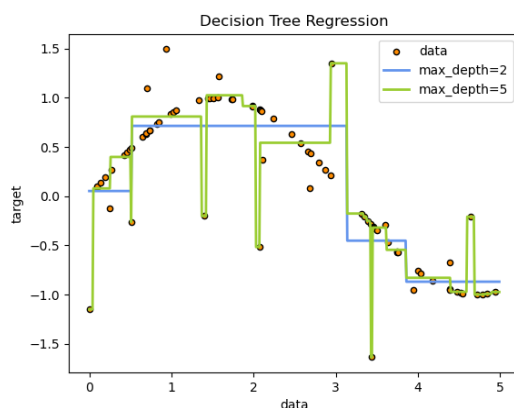


Figure 3.3: Decision Tree Intuition

The confusion matrix, accuracy and the classification report are as follows.

```

[[82 25]
 [18 29]]
0.7207792207792207

```

	precision	recall	f1-score	support
0	0.82	0.77	0.79	107
1	0.54	0.62	0.57	47
accuracy			0.72	154
macro avg	0.68	0.69	0.68	154
weighted avg	0.73	0.72	0.73	154

Figure 3.4: Classification Report of Decision Tree Algorithm

### 3.3 Random Forest

Random forest is a type of supervised machine learning algorithm based on ensemble learning. Ensemble learning is a type of learning where you join different types of algorithms or the same algorithm multiple times to form a more powerful prediction model. The random forest algorithm combines multiple algorithms of the same type i.e. multiple decision trees, resulting in a forest of trees, hence the name "Random Forest".

The random forest algorithm can be used for both regression and classification tasks. Random forest is a supervised learning algorithm. It consists of a large number of decision trees that operate as a group. Each individual tree in random forest classifier output a class prediction and selects best solution by voting. Process starts with first select random samples from dataset, second construct a decision tree for each sample and get a prediction result from each decision tree, third perform voting for each result and forth selects best solution with the most votes as the final prediction. Advantage of using random forest not decision tree is that sometimes decision tree may suffer from overfitting but not random forest because it takes the average of all the predictions. Random forest classifier is considered as a highly accurate method because of many decision trees in the process.

The confusion matrix, accuracy and the classification report are as follows.

```

[[94 13]
 [14 33]]
0.8246753246753247
      precision    recall  f1-score   support

     0       0.87       0.88       0.87       107
     1       0.72       0.70       0.71        47

 accuracy          0.82       154
 macro avg       0.79       0.79       0.79       154
 weighted avg    0.82       0.82       0.82       154

```

Figure 3.5: Classification Report of Random Forest Algorithm

### 3.4 XgBoost

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solves many data science problems in a fast and accurate way. The same code runs on major distributed environments (Hadoop, SGE, MPI) and can solve problems beyond billions of examples.

The confusion matrix, accuracy and the classification report are as follows.

```

[[86 21]
 [16 31]]
Accuracy Score = 0.7597402597402597
      precision    recall  f1-score   support

     0       0.84       0.80       0.82       107
     1       0.60       0.66       0.63        47

 accuracy          0.76       154
 macro avg       0.72       0.73       0.72       154
 weighted avg    0.77       0.76       0.76       154

```

Figure 3.6: Classification Report of XgBoost Algorithm



### 3.5 Support Vector Machine

Support vector machines are a set of supervised learning methods used for classification, regression, and outlier detection. All of these are common tasks in machine learning.

You can use them to detect cancerous cells based on millions of images or you can use them to predict future driving routes with a well-fitted regression model.

There are specific types of SVMs you can use for particular machine learning problems, like support vector regression (SVR) which is an extension of support vector classification (SVC).

The main thing to keep in mind here is that these are just math equations tuned to give you the most accurate answer possible as quickly as possible.

SVMs are different from other classification algorithms because of the way they choose the decision boundary that maximizes the distance from the nearest data points of all the classes. The decision boundary created by SVMs is called the maximum margin classifier or the maximum margin hyperplane.

Machine learning technique support vector machine is also used for classification. It is a supervised algorithm that can classify cases by finding a hyper-plane. It works by mapping data to higher dimensional space so that hyper-plane could be drawn. This mapping is called kernelling and mathematical function used for this mapping is known as kernel function and this function can be of different types such as sigmoid, linear, polynomial and radial basis function(RBF). We can't know which function provides best result with any given dataset but we can choose different function and compare the result. Hyperplane is a separator between different categorical values (Classes). From this hyperplane we can classify unknown cases. Therefore SVM outputs an optimal hyperplane that categorizes new cases. Optimal hyperplane is the one that represents the largest separation between two classes. Examples that are closest to hyperplane are support vectors. Advantage of svm is that they are accurate in high-dimensional spaces and it's also memory efficient because they use subset of training points in the function. Disadvantage is that the algorithm is prone for

over-fitting if the number of features is greater than the number of samples, in our case number of features is less than the number of samples.

The confusion matrix, accuracy and the classification report are as follows.

```

Accuracy Score = 0.7597402597402597
[[92 15]
 [22 25]]

              precision    recall  f1-score   support

     0         0.81         0.86         0.83         107
     1         0.62         0.53         0.57          47

 accuracy              0.76         154
 macro avg              0.72         0.70         0.70         154
 weighted avg           0.75         0.76         0.75         154

```

Figure 3.7: Classification Report of SVM Algorithm

## 3.6 Logistic Regression

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

Machine learning technique logistic regression is used for classification. Logistic regression is similar to linear regression but in a logistic regression we predict a variable which is categorical or discrete like binary(0/1, Yes/No, true/false). In logistic regression independent variables should be continuous but if independent variables are categorical in nature then we can convert it to continuous or leave it as a dummy variable. Logistic regression can be used for both binary classification and multi-class classification but in this project binary classification is used because output variable is 0 or 1. Logistic regression returns probability score between 0 and 1. The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function

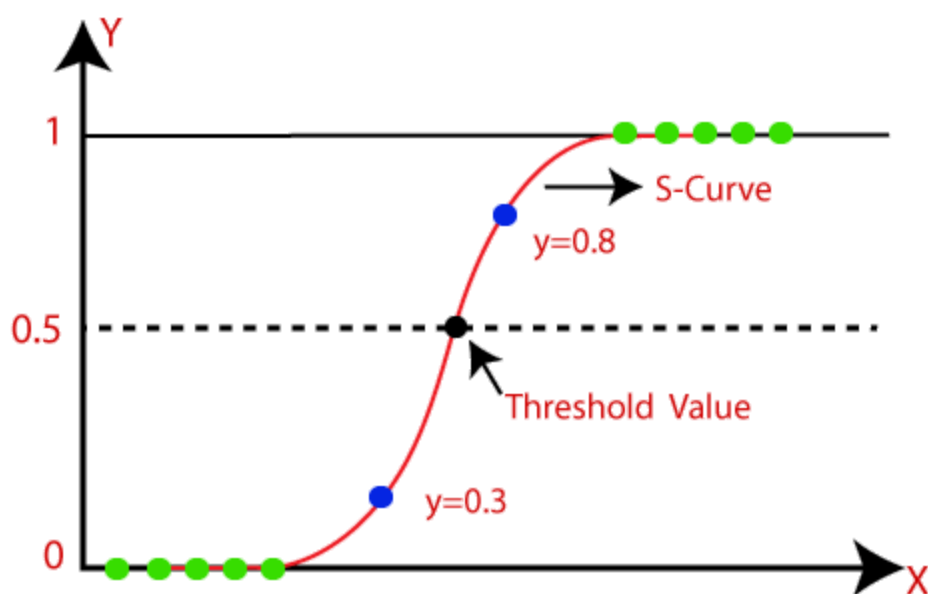


Figure 3.8: Logistic Regression Intuition

The confusion matrix, accuracy and the classification report are as follows

```
[[98  9]
 [18 29]]
0.8246753246753247
```

	precision	recall	f1-score	support
0	0.84	0.92	0.88	107
1	0.76	0.62	0.68	47
accuracy			0.82	154
macro avg	0.80	0.77	0.78	154
weighted avg	0.82	0.82	0.82	154

Figure 3.9: Classification Report of Logistic RegressionAlgorithm

## 3.7 K-Nearest Neighbors

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique. The K-NN algorithm assumes the similarity between the new case/data and available cases and puts the new case into the category that is most similar to the available categories.

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suited category by using K- NN algorithm.

K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

Machine learning technique Knn is also used for classification. Takes a bundle of labeled points and uses them to learn and then label other unknown points. Knn classifies cases based on their similarity to other cases. There are different ways to calculate similarity or dissimilarity between data points and one of them is distance metrics which calculates Euclidean distance. So data points that are close to each other are called neighbors. Algorithm starts with picking a value for k. Second calculate the distance of unknown case from all other cases in the dataset. Three select k observations in training set that are nearest to unknown case and fourth predict output for unknown case using the most popular response (i.e One with most count) value from the k nearest neighbors. For our model we keep changing k starting from 1 and analyze for which value of k algorithm performs best.

KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.



Figure 3.10: KNN Algorithm

### Need of KNN Algorithm

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point  $x_1$ , so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:

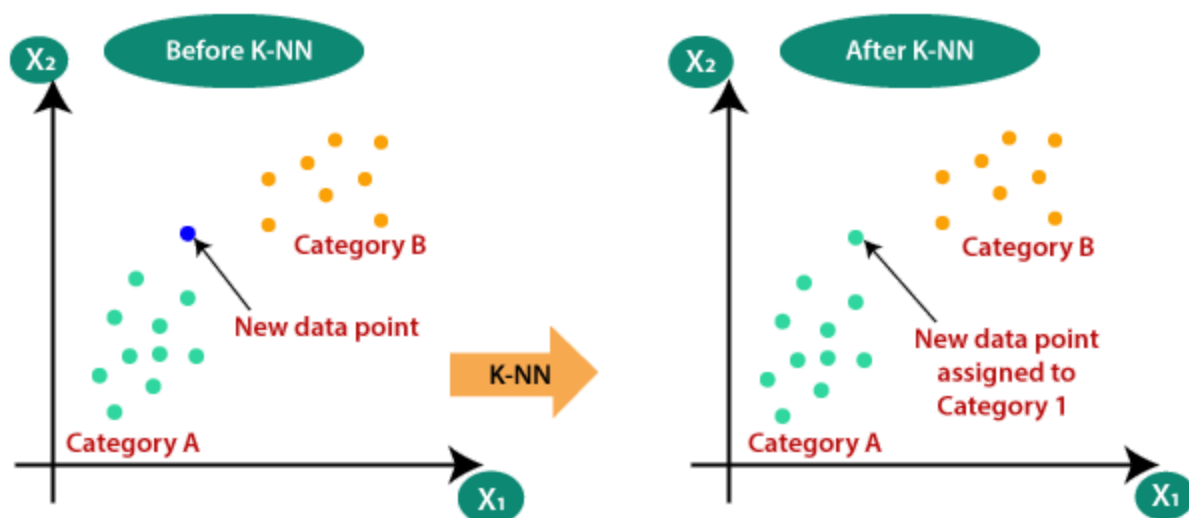


Figure 3.11: KNN Algorithm Intuition

For different values of K, plots of error rates reflects differently:

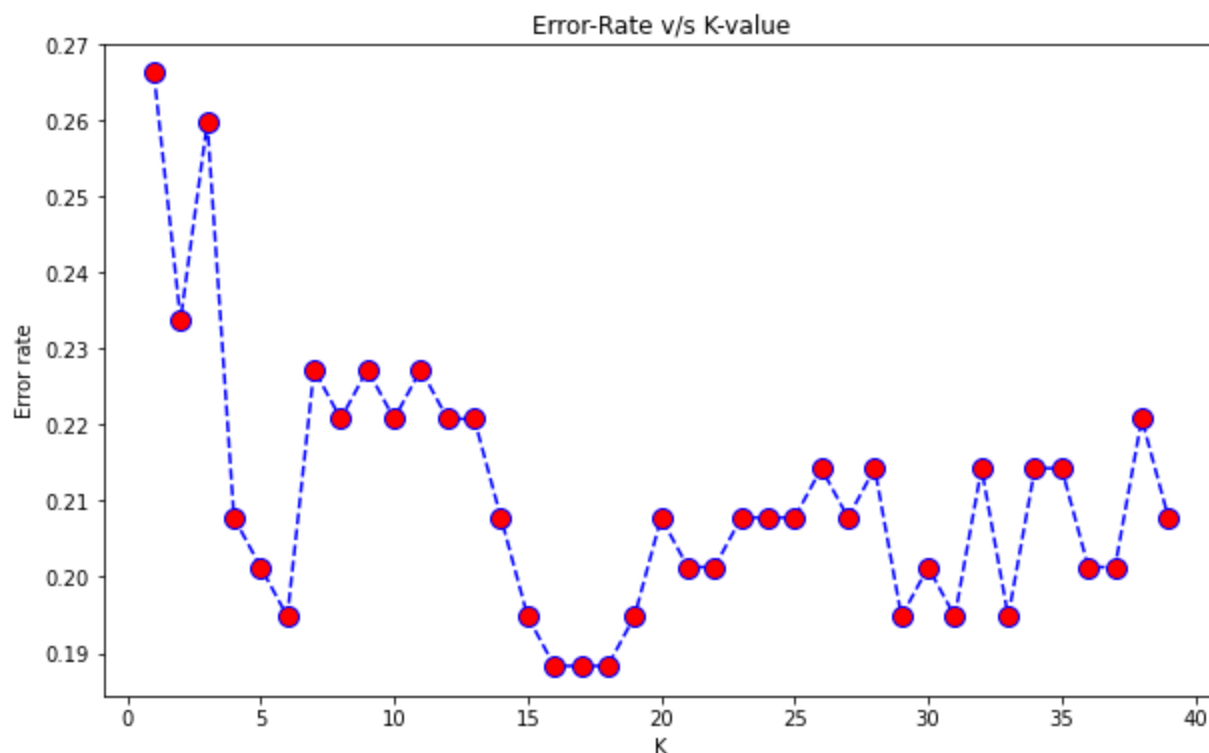


Figure 3.12: K-value Prediction along with the error rates.

The confusion matrix, accuracy and the classification report are as follows for k=17 value as it has the smallest Error rate.

```
[[96 11]
 [18 29]]
0.8116883116883117
```

	precision	recall	f1-score	support
0	0.84	0.90	0.87	107
1	0.72	0.62	0.67	47
accuracy			0.81	154
macro avg	0.78	0.76	0.77	154
weighted avg	0.81	0.81	0.81	154

Figure 3.13: Classification Report of KNN Algorithm

## 3.8 Neural Network

Neurons (aka Nerve Cells) are the fundamental units of our brain and nervous system.

The neurons are responsible for receiving input from the external world, for sending output (commands to our muscles), and for transforming the electrical signals in between.

Artificial Neural Networks are normally called Neural Networks (NN). Neural networks are in fact multi-layer Perceptrons. The perceptron defines the first step into multi-layered neural networks.

Input data (Yellow) are processed against a hidden layer (Blue) and modified against another hidden layer (Green) to produce the final output (Red).

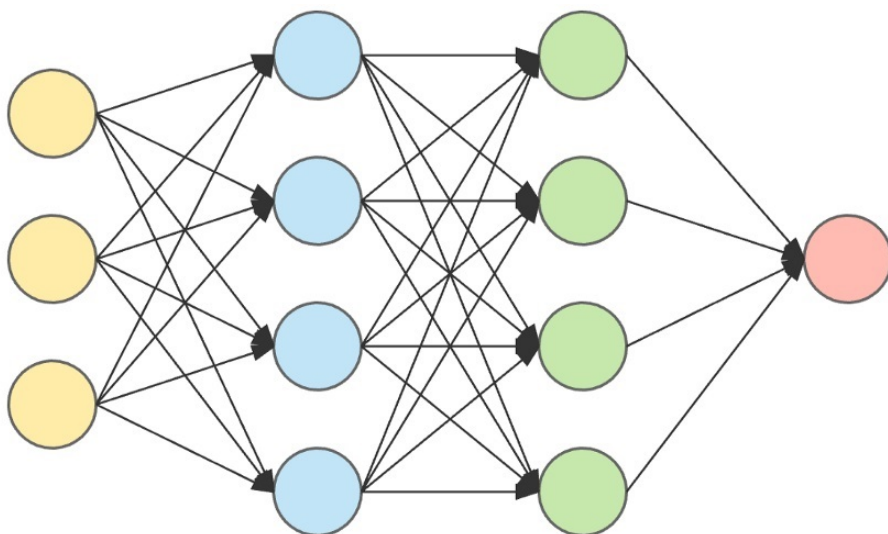



Figure 3.14: Neural Network Architecture

The confusion matrix, accuracy and the classification report are as follows: model trained on 50 epochs.



```
Out[167]: <keras.callbacks.History at 0x1c6d6688df0>

In [168]: model.evaluate(X_test, y_test)
          5/5 [=====] - 0s 6ms/step - loss: 0.5703 - accuracy: 0.7532

Out[168]: [0.5703256726264954, 0.7532467246055603]
```

Figure 3.15: Accuracy of Neural Network

## 3.9 Best Model

It is found that the best model for this diabetes dataset is Random forest as it has the best accuracy and the F-score.



# Chapter 4

## Methodology

The aim of this study was to create classification models for the diabetes data set and to predict whether a person is sick by establishing models and to obtain maximum validation scores in the established models.

The system architecture has been demonstrated here in this image below.

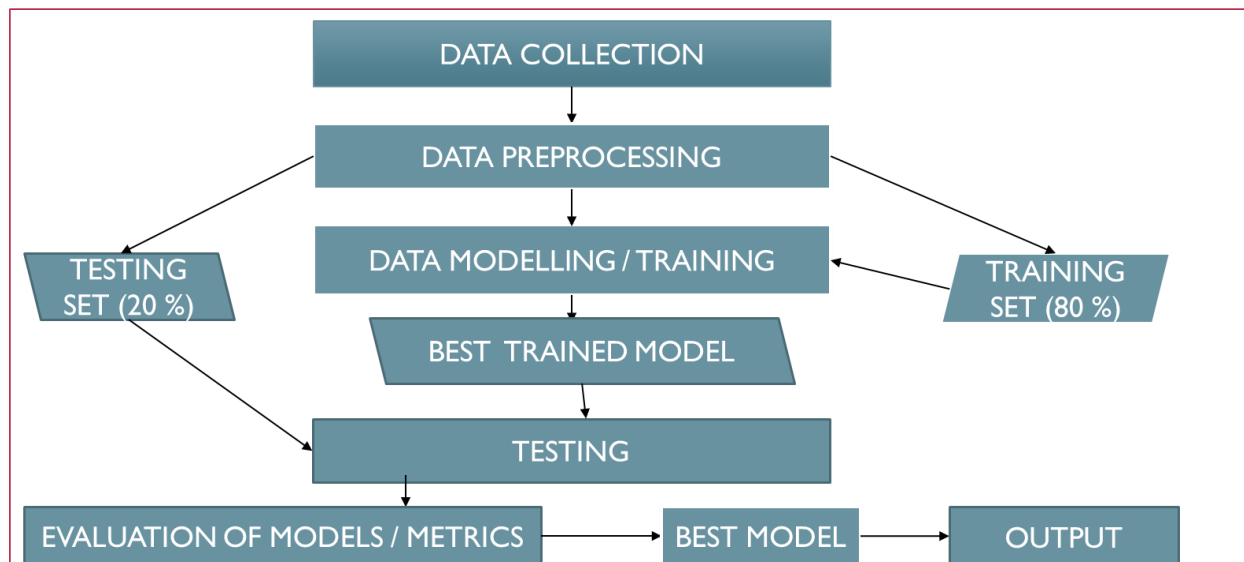



Figure 4.1: Methodology

### 4.1 Dataset Collection

The Pima Indian diabetes dataset (PIDDD) will be taken from online Kaggle website which has 768 instances and 8 features. The dataset features are:

- a) Pregnancies

- 
- b) Glucose
  - c) Blood pressure
  - d) Skin thickness
  - e) Insulin
  - f) BMI
  - g) Diabetes pedigree
  - h) Age

## 4.2 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations. With Exploratory Data Analysis; The data set's structural data were checked. The types of variables in the dataset were examined. Size information of the dataset was accessed. The 0 values in the data set are missing values. Primarily these 0 values were replaced with NaN values. Descriptive statistics of the data set were examined.

## 4.3 Data Visualization

From data visualization we can analyze the data by looking at some plots such as histograms, correlation matrix, countplot etc. It also tells how features correspond to output. Real world data can be incomplete, inconsistent and noisy so this can lead to poor model training. Preprocessing required in order to address these issues. The data description is shown below.

	count	mean	std	min	10%	25%	50%	75%	90%	95%	99%	max
Pregnancies	768.0	3.845052	3.369578	0.000	0.000	1.00000	3.0000	6.00000	9.0000	10.00000	13.00000	17.00
Glucose	768.0	120.894531	31.972618	0.000	85.000	99.00000	117.0000	140.25000	167.0000	181.00000	196.00000	199.00
BloodPressure	768.0	69.105469	19.355807	0.000	54.000	62.00000	72.0000	80.00000	88.0000	90.00000	106.00000	122.00
SkinThickness	768.0	20.536458	15.952218	0.000	0.000	0.00000	23.0000	32.00000	40.0000	44.00000	51.33000	99.00
Insulin	768.0	79.799479	115.244002	0.000	0.000	0.00000	30.5000	127.25000	210.0000	293.00000	519.90000	846.00
BMI	768.0	31.992578	7.884160	0.000	23.600	27.30000	32.0000	36.60000	41.5000	44.39500	50.75900	67.10
DiabetesPedigreeFunction	768.0	0.471876	0.331329	0.078	0.165	0.24375	0.3725	0.62625	0.8786	1.13285	1.69833	2.42
Age	768.0	33.240885	11.760232	21.000	22.000	24.00000	29.0000	41.00000	51.0000	58.00000	67.00000	81.00
Outcome	768.0	0.348958	0.476951	0.000	0.000	0.00000	0.0000	1.00000	1.0000	1.00000	1.00000	1.00

Figure 4.2: Data Description

The distribution of the outcome variable in the data was examined and visualized.

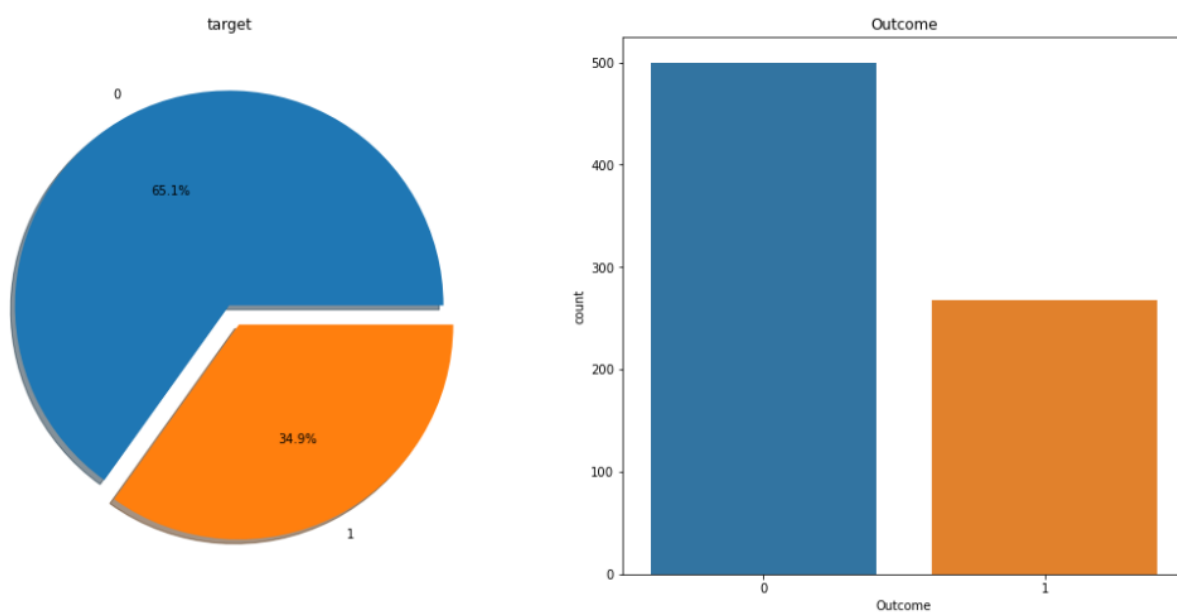


Figure 4.3: Outcome Visualization

Number of zero and one outcomes are as follows:-

```
True cases / diabetes=YES : 268  
False cases / diabetes=NO : 500  
True cases in percentage : 34.89583333333333 %  
False cases in percentage: 65.10416666666666 %
```

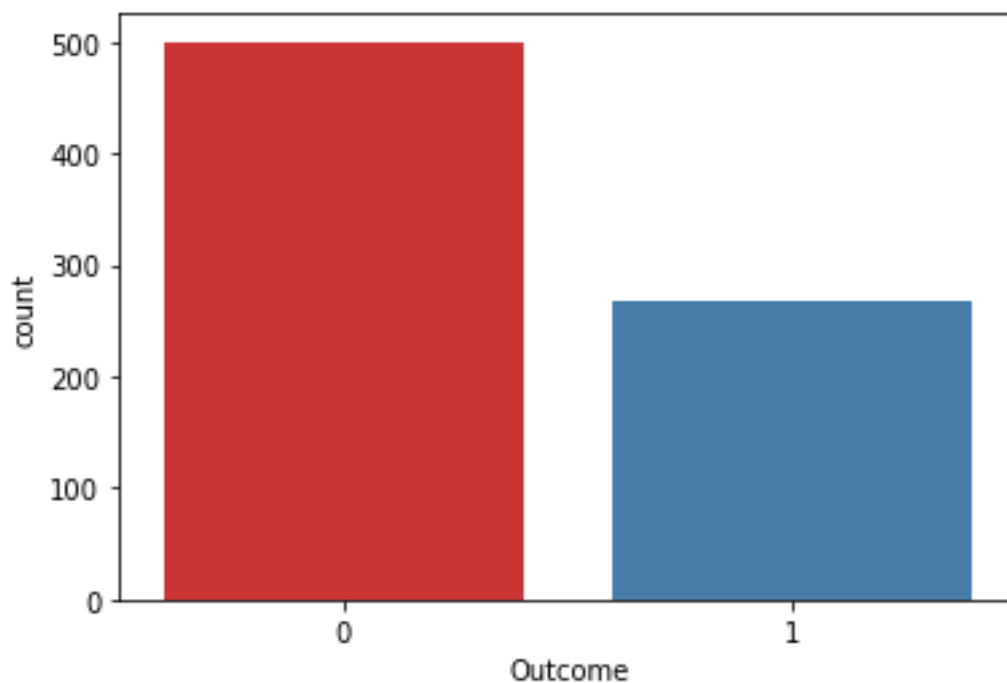


Figure 4.4: Hissing Values

Adding NaN to the zero values:-

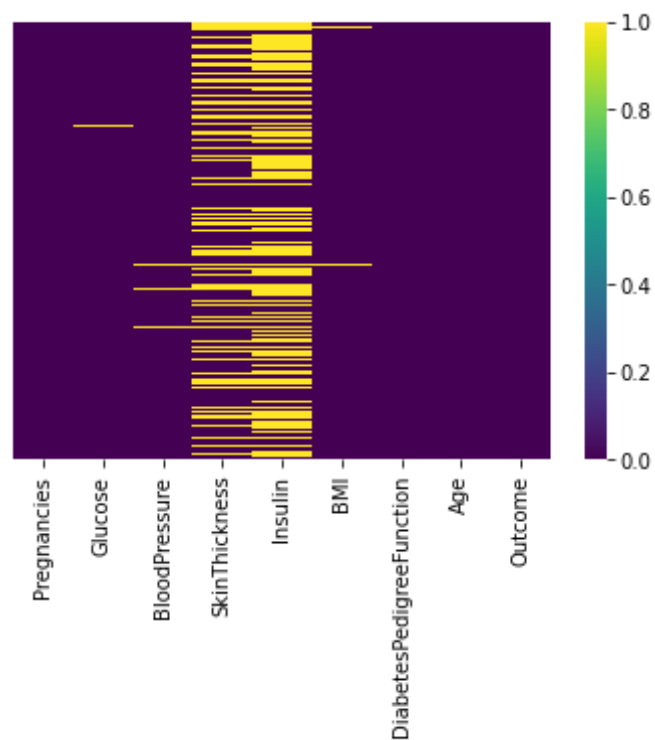


Figure 4.5: Missing Values Visualization

Correlation of different attributes:-

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.221898
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.466581
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.065068
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.074752
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.130548
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.292695
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.173844
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.238356
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.000000

Figure 4.6: Missing Values Visualization

## Distribution Plot

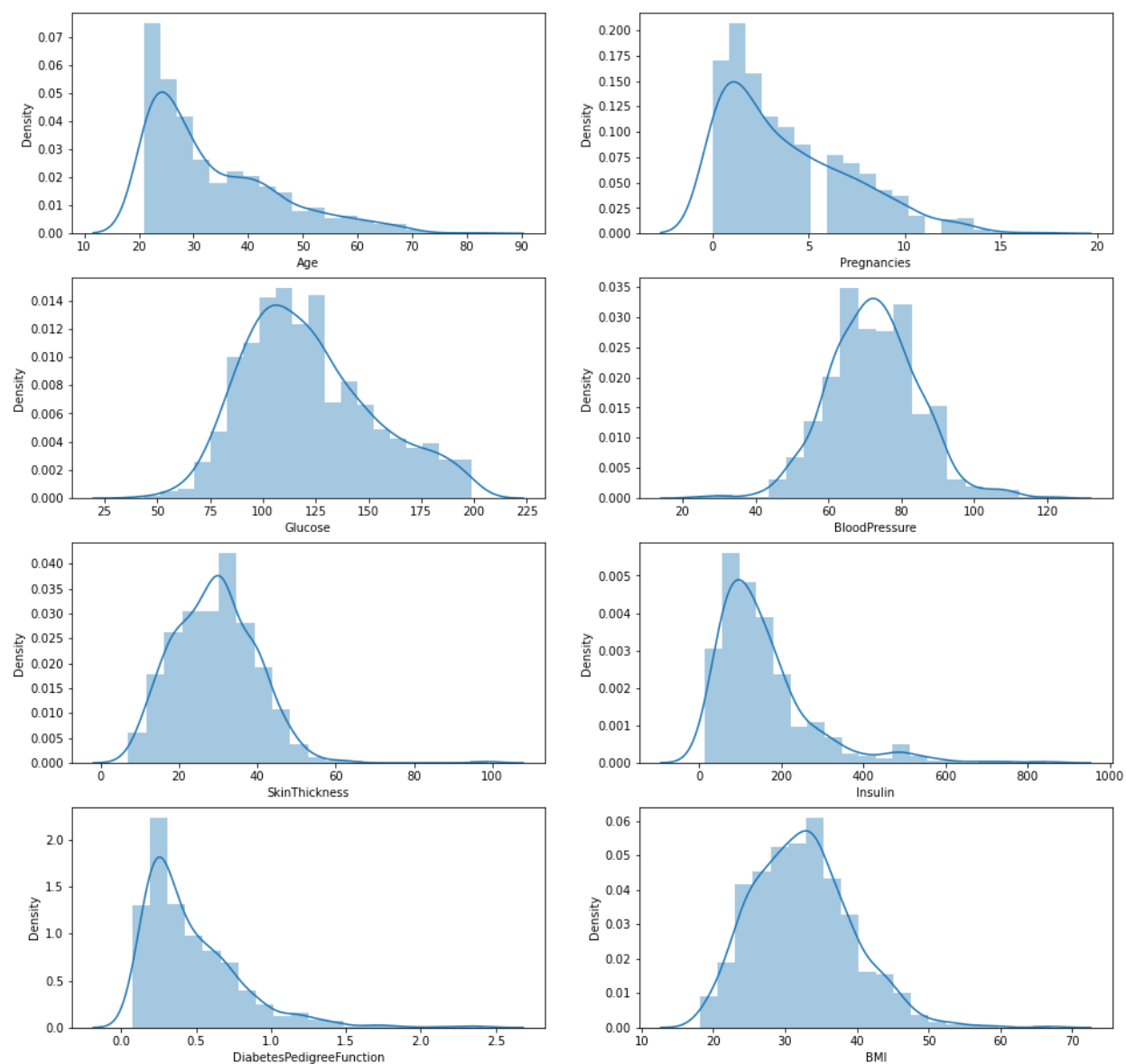


Figure 4.7: Distribution Plot

PairPlots are as follows :-



Figure 4.8: Pair Plot

## 4.4 Data Preprocessing

Preprocessing technique is used so that the raw data can be converted into an understandable format which is used for training and testing. We will use various efficient preprocessing techniques like data imputation, handling null values etc. After that the dataset will be divided into training and testing sets, percentage split could be 80 percent of data is for training and remaining 20 percent of data is for

testing. Afterwards we Normalize the data to make it within a range. Additionally we replace the NaN values to the mean and median values accordingly. Data Preprocessing section, the NaN values missing observations were filled with the median values of whether each variable was sick or not. The X variables were standardized with the Scalarization method.

## 4.5 Model Building

For the Model Naive Bayes Model, the accuracy is 79.22%, Decision Tree is 72.07%, Random Forest is 82.46%, XgBoost is 75.97%, SVM is 75.97%, Logistic Regression is 82.46%, KNN is 81.16% and Neural Network is 75.32%. The outcome for Random Forest came out to be the best for being a bagging technique.

Decision Tree Parameters on tuning: DecisionTreeClassifier(max\_depth=2, max\_features=5, min\_samples\_leaf=6)

RandomForestClassifier Parameters on tuning:

RandomForestClassifier(max\_depth=5, n\_estimators=80)

XgBoost Parameters on tuning:

XGBClassifier(base\_score=0.5, booster='gbtree', callbacks=None, colsample\_bylevel=1, colsample\_bynode=1, colsample\_bytree=1, early\_stopping\_rounds=None, enable\_categorical=False, eval\_metric=None, gamma=0, gpu\_id=-1, grow\_policy='depthwise', importance\_type=None, interaction\_constraints='', learning\_rate=0.300000012, max\_bin=256, max\_cat\_to\_onehot=4, max\_delta\_step=0, max\_depth=6, max\_leaves=0, min\_child\_weight=1, missing=nan, monotone\_constraints='()', n\_estimators=100, n\_jobs=0, num\_parallel\_tree=1, predictor='auto', random\_state=0, reg\_alpha=0, reg\_lambda=1)

Naive Bayes and Logistic Regression on Parameters tuning: Default Values

KNN on Parameters tuning: KNeighborsClassifier(n\_neighbors = 17)

SVC on Parameters tuning: SVC(C=3)



### Neural Network on Parameters tuning:

```
tf.keras.models.Sequential([tf.keras.layers.Flatten(input_shape=(8,)),  
                             tf.keras.layers.Dense(12, activation=tf.nn.relu),  
                             tf.keras.layers.Dense(8, activation=tf.nn.relu),  
                             tf.keras.layers.Dense(128, activation=tf.nn.relu),  
                             tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)])
```

## 4.6 Evaluation of models

For performance evaluation of our models classification metrics or confusion metrics should be made to calculate recall, precision, Fscore, and accuracy.

Classification metrics:

- i. True Positive if (predicted=positive and actual=positive) -> TP
- ii. False Positive if (predicted=positive and actual=negative) -> FP
- iii. True Negative if (predicted=negative and actual=negative) -> TN
- iv. False Negative if (predicted=negative and actual=positive) -> FN

# Chapter 5

## Implementation

So, till now we explored how all five machine learning algorithms work, information about pima indian dataset, how preprocessing works and some information about evaluation metrics.

### 5.1 Setting up the Environment

This project is done on the Windows 10 system. We will begin by downloading all relevant machine learning libraries including pandas, numpy, sklearn, seaborn, matplotlib etc and import them at the start of the code written in python3 language. Ide used for writing the code is jupyter notebook.

### 5.2 Important Libraries and their versions

The packages used are :-

Pandas==1.3.4

Seaborn==0.11.2

Matplotlib==3.1.0

Numpy==1.20.3


Sklearn==1.1.2

tensorflow==2.7.0

### 5.3 Explanation

#### 5.3.1 Load the dataset

The dataset used in this project is pima indian dataset and it is present in csv format. The dataset consists of 8 features and one column for output. Features include number of pregnancies, glucose level, blood pressure, skin thickness, insulin, bmi, diabetes pedigree function and age. There are a total of 768 samples so the dimension of the dataset is 768 X 9.



```
pd.read_csv(filename)- For loading data from csv file
dataset.shape- Outputs dimension of dataset
dataset.head()- For returning first 10 rows of dataset.
```

### 5.3.2 Data Manipulation and Analysis

The countplot, histogram, correlation matrix, handling missing data and data imputation has been done on the loaded dataframe.

`dataset.isnull().values.any()`- In our project it returns false so there are no null values so no need to handle this problem.

In this process zero value in any column is replaced with statistical value such as mean or median of that particular column.

In our project zero values of five features that is glucose, skin thickness, blood pressure, bmi and insulin are marked with NaN and then replaced with mean of the column because glucose=0 or skin thickness=0 or blood pressure=0 or bmi=0 or insulin=0 is not possible(not real). So the trained model is more realistic.

```
dataset[attribute].replace(0,np.NaN)- attribute=0 marked with NaN
dataset[attribute].fillna(dataset[attribute].mean())- NaN value replaced by mean
of that column.
```

### 5.3.3 Data Splitting

```
variables=train.test.split(independent variable vector, dependent variable
vector, test.size=0.3) - Function for splitting the data.
```

### 5.3.3 Model Training and Testing

All the data training models and its parameters have been mentioned in the above segments. We have decided to move forward with the Random Forest Model which gave us the highest f score and accuracy.

# Chapter 6

## Conclusion

After using all these patient records, we are able to build a machine learning model (random forest – best one) to accurately predict whether or not the patients in the dataset have diabetes or not along with that we were able to draw some insights from the data via data analysis and visualization. For Naive Bayes Model, the accuracy is 79.22%, Decision Tree is 72.07%, Random Forest is 82.46%, XgBoost is 75.97%, SVM is 75.97%, Logistic Regression is 82.46%, KNN is 81.16% and Neural Network is 75.32%.

Therefore we can conclude that the model Random Forest is the most suitable model.

## 6.1 Result and Comparison Chart

**\*\* Paper[1], Paper[2] and Paper[3] has been mentioned in Chapter2 - Existing Methods**

Algorithms	Our	Paper 1	Paper 2	Paper 3
Logistic Regression	82.46 %	77.5 %	72.39 %	74 %
Support vector machine	75.97 %	77.9 %	73.43 %	77 %
K nearest neighbor	72.07 %	76 %	71.3 %	77 %
Decision tree	72.07 %	75.8 %	72.91 %	71 %
Neural Network	75.32 %	78.4 %	-----	-----
Naïve Bayes	79.22%	-----	-----	-----
Random Forest	82.46 %	79.7 %	74.4 %	71 %

Figure 6.1: Comparison Chart

## 6.2 Future Work

Till now we have successfully execute all five machine learning algorithms along with some preprocessing techniques and in the next phase of the project we will on expanding the dataset to some deep learning methods and selects the best model from machine learning and deep learning models, we will also add some more efficient preprocessing techniques so that our models accuracy will increase if possible. We will also include GUI so that users can interact with the software.

Expanding the dataset to deep learning algorithms .

Make a program in the end for the comparison between machine learning and deep learning and one model will be selected for output which has highest accuracy. Add some extra preprocessing techniques. Make a GUI for interaction of users with the software.