Analyzing Pima Indians Diabetes Database Using Supervised Machine Learning Algorithms

**Introduction:**

Diabetes is a prevalent disease around the world. Many studies are conducted in different samples to identify and prevent the disease. Around globe, studies are made for the prediction and prevention of diabetes. There are several reasons for the onset of diabetes in adults, which includes genetics, nutrition, stress, etc. Prediction of the possibility in specific population could lead to the prevention of this disease in the population. In this study, the data selected is “Pima Indians Diabetes Database”, with the supervised machine learning methods of machine learning algorithms the possible outcome of persons to develop diabetes are predicted, which could be inevitable for current researchers of Diabetes prevention studies.

**Objective:**

Supervised machine learning algorithms are used to predict the possible outcome in Pima population to develop diabetes, which is in turn based on the diagnostic measurement available in the dataset. Objective is to predict the outcome and prevent the development of diabetes in those samples.

**Dataset Description:**

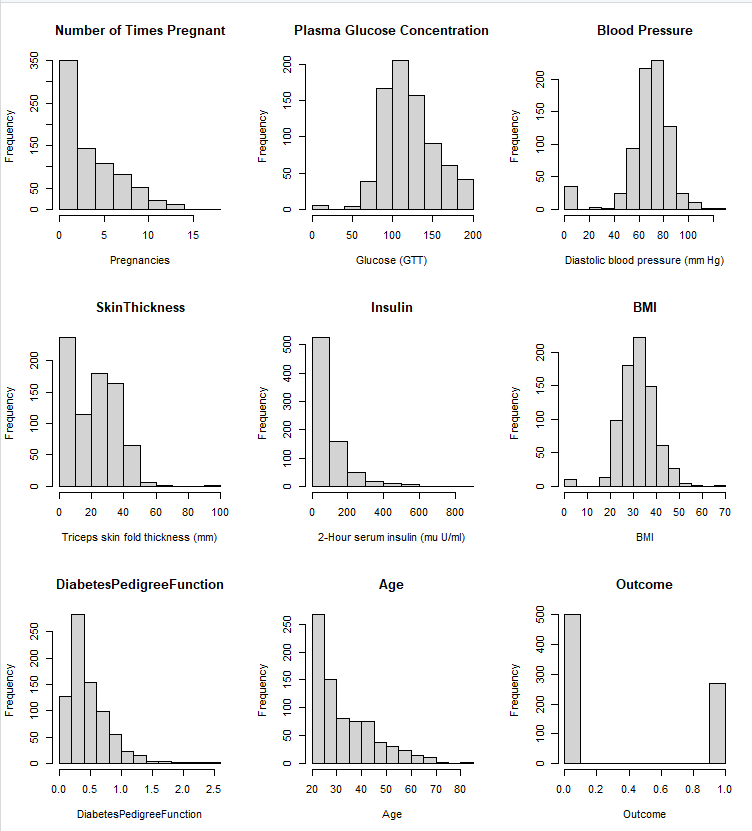
“Pima Indians Diabetes Database” dataset is used for this study (Source: <https://www.kaggle.com/uciml/pima-indians-diabetes-database>). This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases with 768 observations and 9 columns and the attributes are:

Those input variables were:

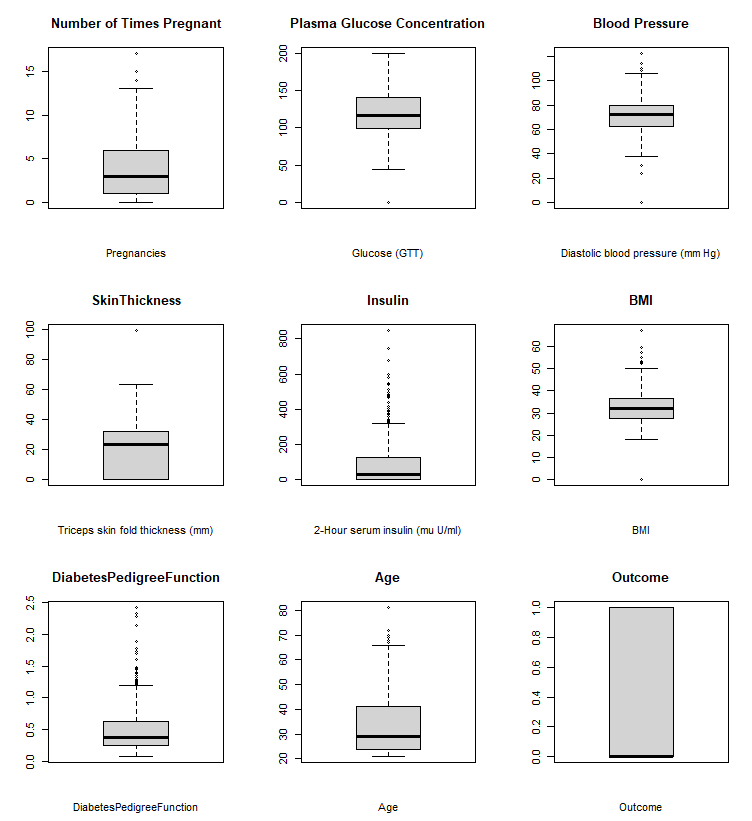
1. Pregnancies - Number of times pregnant
2. Glucose - Plasma Glucose Concentration
3. BloodPressure - Blood Pressure (mm Hg)
4. SkinThickness - Triceps Skin Fold Thickness (mm)
5. Insulin - Serum Insulin (µU/ml)
6. BMI - Body Mass Index (Weight in kg / (Height in m)2)
7. DiabetesPedigreeFunction - Diabetes Pedigree Function
8. Age - Age (years)
9. Outcome - Outcome of test (diabetic, 1; nondiabetic, 0).

In this data, among 768 in the sample about 268 where identified as diabetic and 500 are nondiabetic. Now, let look at the variable in below Figure 1, which is based on frequency distribution. Only few observations could be made from this plot. The pregnancies, skin thickness, diabetes pedigree function, age, 2- serum insulin plots show right skewed or positive skewness. On other hand, blood pressure and BMI show left skewed or negative skewness. And in attributes pregnancy, blood pressure, insulin, diabetes pedigree function, BMI and age outlier are found in comparatively higher ratio, Figure 2.

**Figure. 1: The frequency distribution of the features in the “Pima Indians Diabetes Database”.**



**Figure 2: Outliers in the attribute of the “Pima Indians Diabetes Database”.**



This dataset is free of NA.

**Methodologies:**

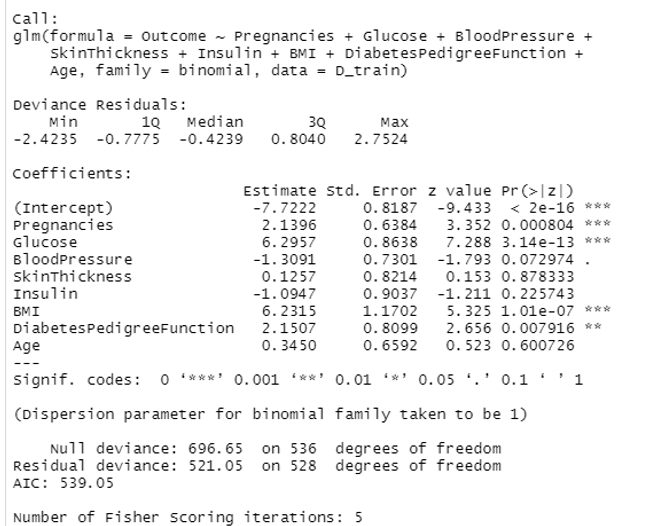
Supervised machine learning algorithms are used to predict for this dataset. In this dataset, attribute “Outcome” is used the label and Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, and Age are used for input variable.

In the supervised machine learning algorithms, the methodologies used to predict the outcome are K nearest neighborhood method (KNN), Decision Tree, Logistic regression, and Neutral network.

**Logistic Regression:**

The logistic binominal classification method is used for the study. The logistic regression is based on assumption such as, the outcome is binary, there is a linear relationship between logit outcome and predictor variables, no influence of outliners, and no high correlation among predictor. Further, in this classification the prediction of results could be based on the p value. In this study, the attributes pregnancy, glucose, BMI, and Diabetes pedigree function has greater significance based on p value. In this method 79% of the accuracy is obtained (Appendix I for detailed methodologies).

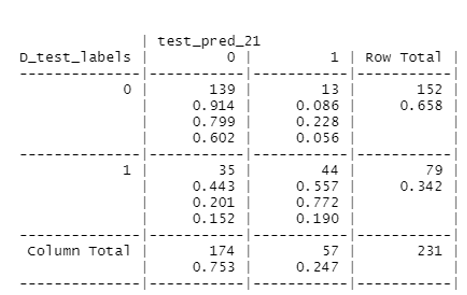
**Figure 3. Logistic regression results**



**K Nearest Neighborhood Method (KNN):**

KNN is an effective algorithm as it can handle realistic data. It is based on the closeness between the data point, which is calculated by Euclidean distance method. Hence, the k tuning parameter is significant in this study we have tuned the dataset with the k value 21, 10 and 5. Highest accuracy is obtained with k21 and the accuracy for this model is 79%. Further, cross table evaluation has predicted 139 no diabetic (TP) persons, 44 diabetic (TN) persons, 13 predicted diabetic but aren’t diabetic (FP) and 35 persons are diabetics but not detected (FN) **Figure. 4**. Because of the high accuracy it is mostly used (Appendix I for detailed methodologies).

**Figure 4: K Nearest Neighborhood Method cross table reference.**

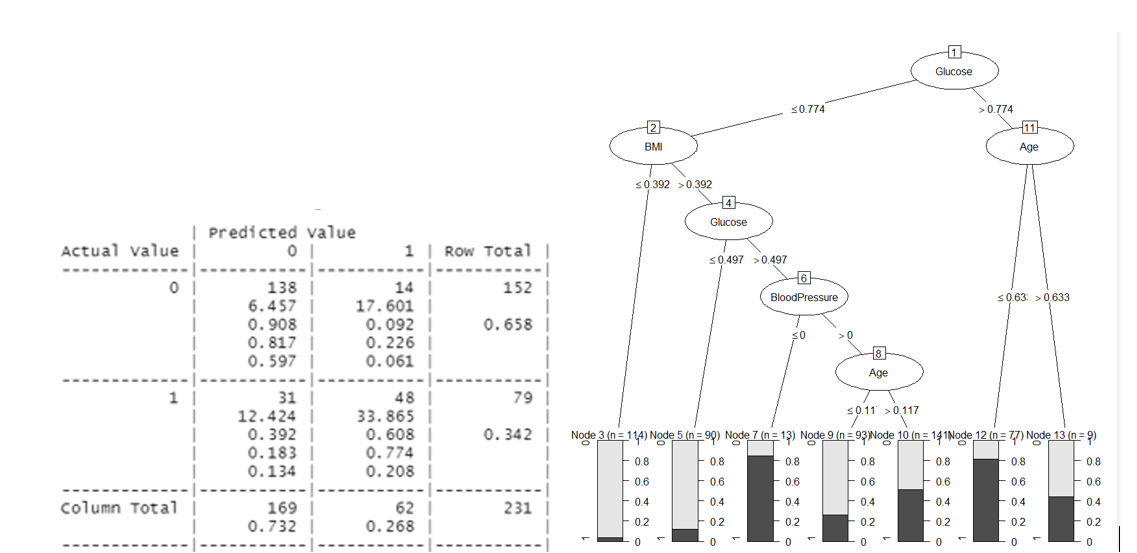


**Decision Trees Method (C5.0):**

The Decision tree is based on recursive partitioning, where the start is the root node, which represents out entire dataset. The algorithm chooses the attribute that is most predictive to the label, which in turn form the tree branches and the process continues as each node branches based recursive partitioning.

The tuning parameter used here is trials. The one with lest trials produces better result. Hence the train sample with trial 5 is taken as it has highest accuracy, 0.8051. Further, the cross-reference table has predicted 138 no diabetic (TP) persons, 48 diabetes (TN) persons, 14 persons predicted diabetes but aren’t diabetic (FP) and 31 persons diabetics but not detected (FN). Decision tree C5.0 method has achieved 81% of accuracy (Appendix I for detailed methodologies).

**Figure 5: Decision Trees Method cross table reference and decision tree plot (trial = 5)**

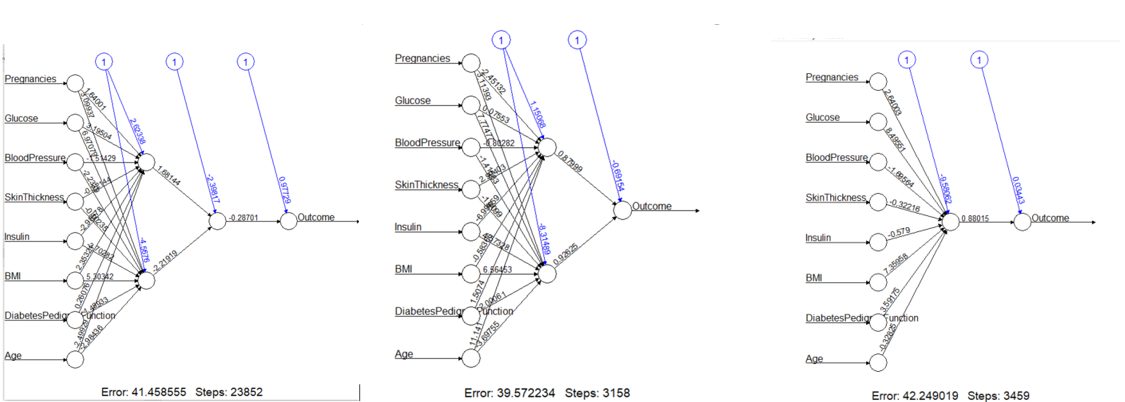


**Neural Network:**

Neural Network or Artificial Neural Network is based on biological neuron system, which modifies the internal structure by adjusting weights of inputs. By pattern recognition it solves the complexities of the machine processing.

The neural network is of input layer (dataset input), hidden layer (optimize weight of input variable), and output layer (output of the prediction). In this study, the model without any hidden layer produces highest accuracy of 61%, compared to the 2 hidden layer model and the model with softplus activation factor (Appendix I for detailed methodologies).

**Figure 6: Neural Network plot (a) soft max activation, (b) 2 hidden layer and (c) no hidden layer**



**Results:**

In this study, Decision tree and KNN methods provide better results in predicting the outcome in the given sample. In the given test sample, Decision tree method has predicted 138 no diabetic (TP) persons, 48 diabetes (TN) persons, 14 persons predicted diabetes but aren’t diabetic (FP) and 31 persons diabetics but not detected (FN). Similarly, KNN has predicted 139 no diabetic (TP) persons, 44 diabetic (TN) persons, 13 predicted diabetic but aren’t diabetic (FP) and 35 persons are diabetics but not detected (FN). Hence, both Decision tree method and KNN method could be efficient for the dataset because of their high accuracy of 81% and 79%.

|  |  |  |
| --- | --- | --- |
| **Table 1. Accuracy for the machine learning algorithms for Pima Indians Diabetes Database** | | |
| **Machine Learning Algorithms** | **Parameters** | **Accuracy** |
| K nearest neighborhood method (KNN) | k = 21 | 0.7922 |
|  | k = 10 | 0.7575 |
|  | k = 5 | 0.7532 |
| Logistic regression | default | 0.7922 |
| Decision Tree | trials = 1 | 0.7835 |
|  | trials = 5 | 0.8051 |
|  | trials = 20 | 0.7878 |
| Neutral network | default | 0.6131 |
|  | hidden layer = 2 | 0.589 |
|  | hidden layer = 2,1 | 0.5979 |

**Conclusion:**

For this dataset, KNN and Decision tree method where proved to more efficient when compared to the logistic regression and neural network method. Neural network method has more black boxes and required more development and it requires a huge set of data. Finally, the prediction of diabetes in a population could be highly effective to prevent disease in this population. The required accuracy to predict is obtained through the above supervised machine learning methods. Hence, the identified population susceptible to the disease could be studied and measures can be applied in order to prevent the onset of the disease.

**References:**

Jack W. Smith, J. E. Everhart, W. C. Dickson, et al. Using the ADAP Learning Algorithm to Forcast the Onset of Diabetes Mellitus. Johns Hopkins Apl Technical Digest, November 1988.

Meherwar Fatima, Maruf Pasha. Survey of Machine Learning Algorithms for Disease Diagnostic. Journal of Intelligent Learning Systems and Applications, 2017, 9, 1-16.

Quan Zou, Kaiyang Qu, Yamei Luo, et al. Predicting Diabetes Mellitus with Machine Learning Techniques. Front. Genet., 06 November 2018 | <https://doi.org/10.3389/fgene.2018.00515>

**Appendix I:**

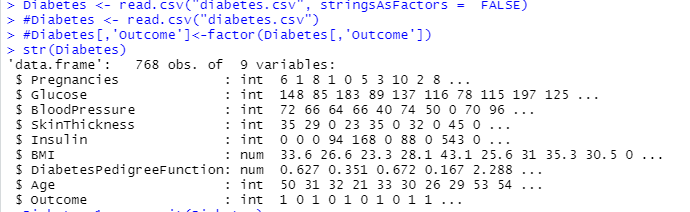
**Procedure For this Study:**

1. Importing Data
2. Preprocessing the Data:
3. Supervised learning methodologies
4. KNN
5. Logistic classification
6. Tree
7. Neural Network

The programing language used is R.

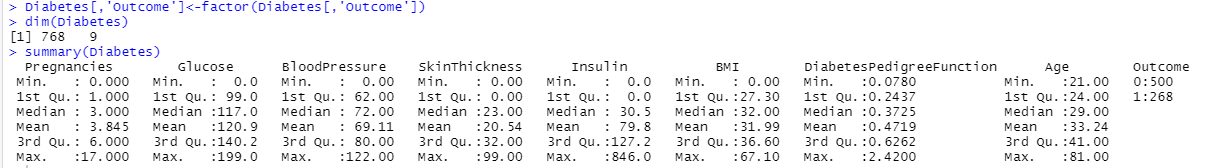
1. **Importing Data:**

The data is imported as csv format.



As the outcome attribute is binomial it is change to factor when illustrating the summary.

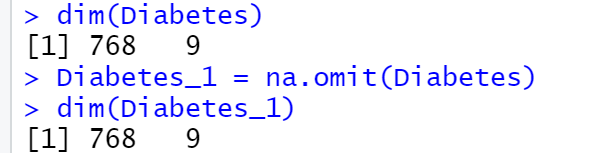
**Summary OF Data:**



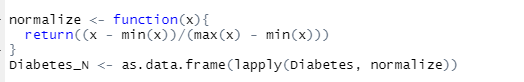
**Preprocessing Data:**

The data set is normalized, and NA are checked in the dataset. This dataset does not have NA.

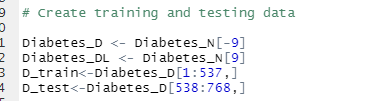
**Removing NA:**



**Data normalized:**



**Training and testing data are prepared.**

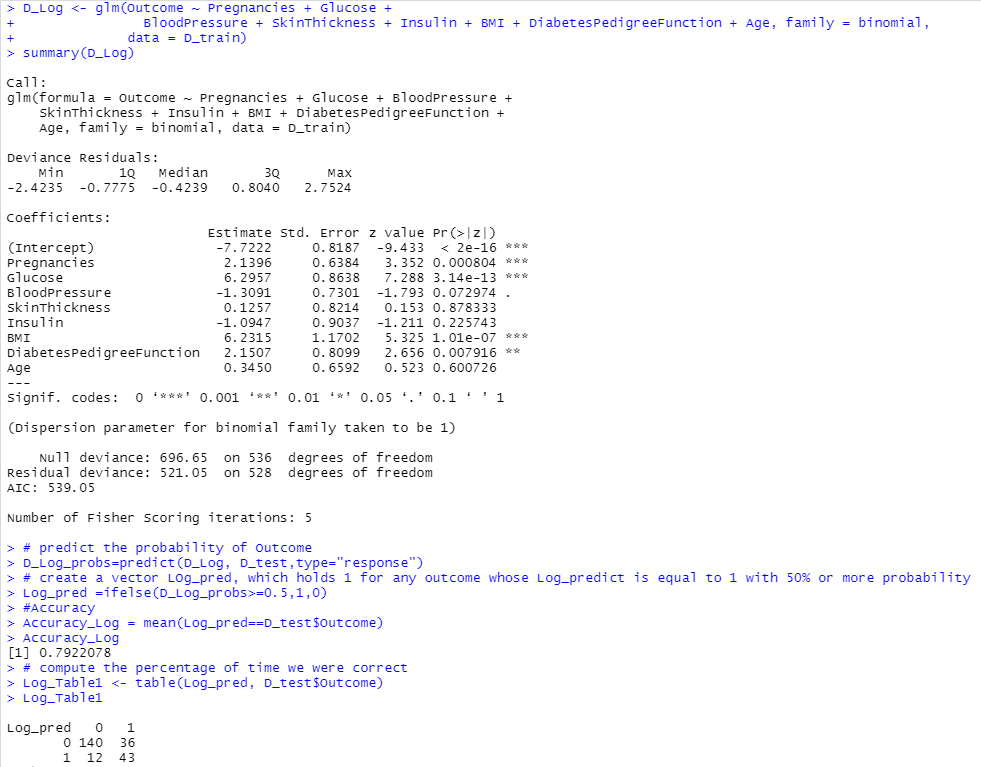


**Creating label for training and testing data**



**Logistic Classification Method:**

The logistic classification binominal method 79% of accuracy is obtained.

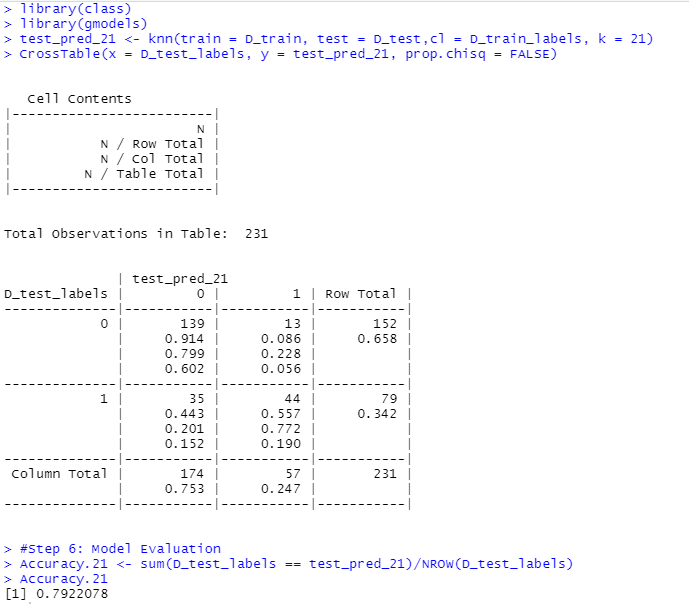


TP – 140 no diabetic, TN – 43 - diabetes, FP – 12 predicted diabetes but aren’t diabetic and FN – 36 diabetics not detected.

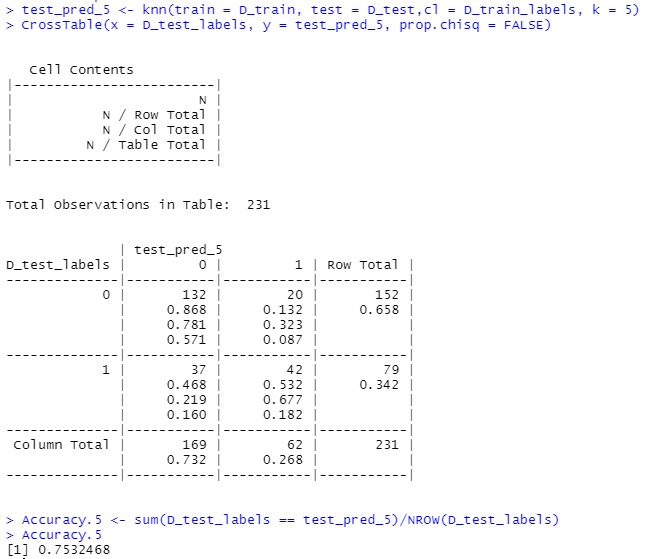
**K nearest neighborhood method (KNN):**

KNN is method is an effective algorithm as it can handle realistic data. It is based on the closeness between the data point which is calculated by Euclidean distance method. Hence the k tuning parameter are 21, 10 and 5.

**Model – 1: KNN (Parameter: k=21)**

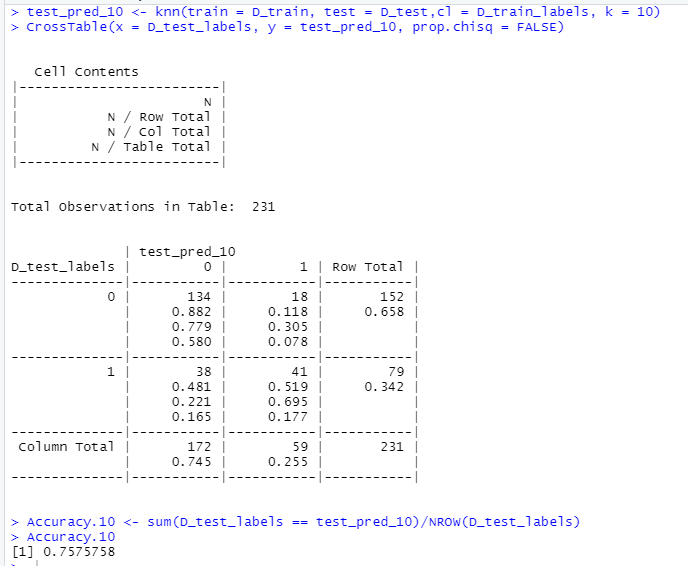


TP – 139 no diabetic, TN – 44 - diabetes, FP – 13 predicted diabetes but aren’t diabetic and FN – 30 diabetics not detected.

**Model – 2: KNN (Parameter: k=5)**

TP – 132 no diabetic, TN – 42 - diabetes, FP – 20 predicted diabetes but aren’t diabetic and FN – 37 diabetics not detected.

**Model – 2: KNN (Parameter: k=10)**

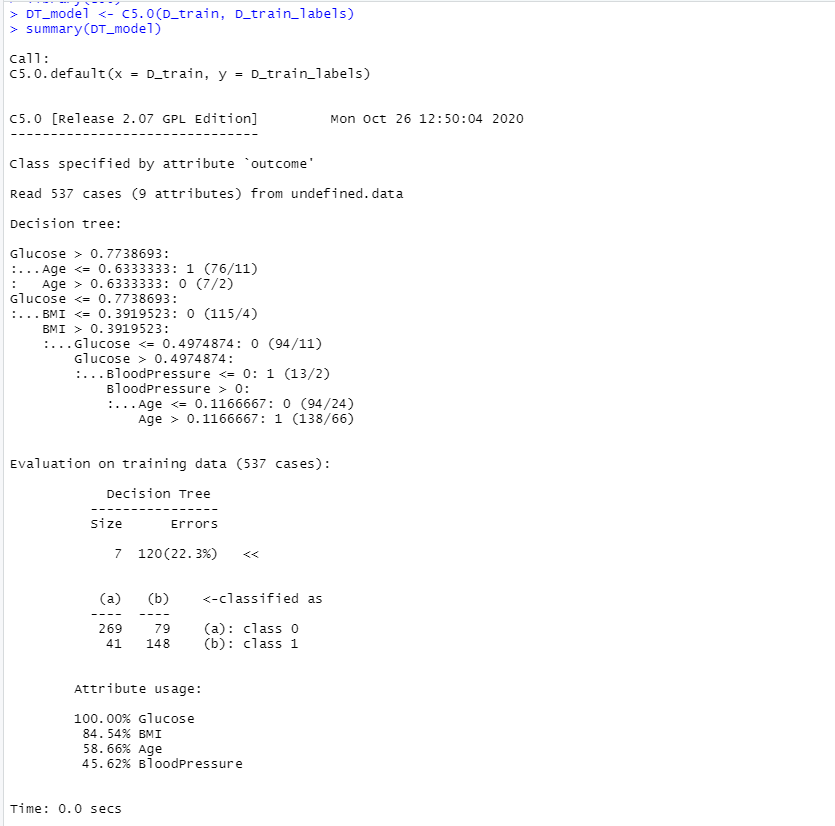


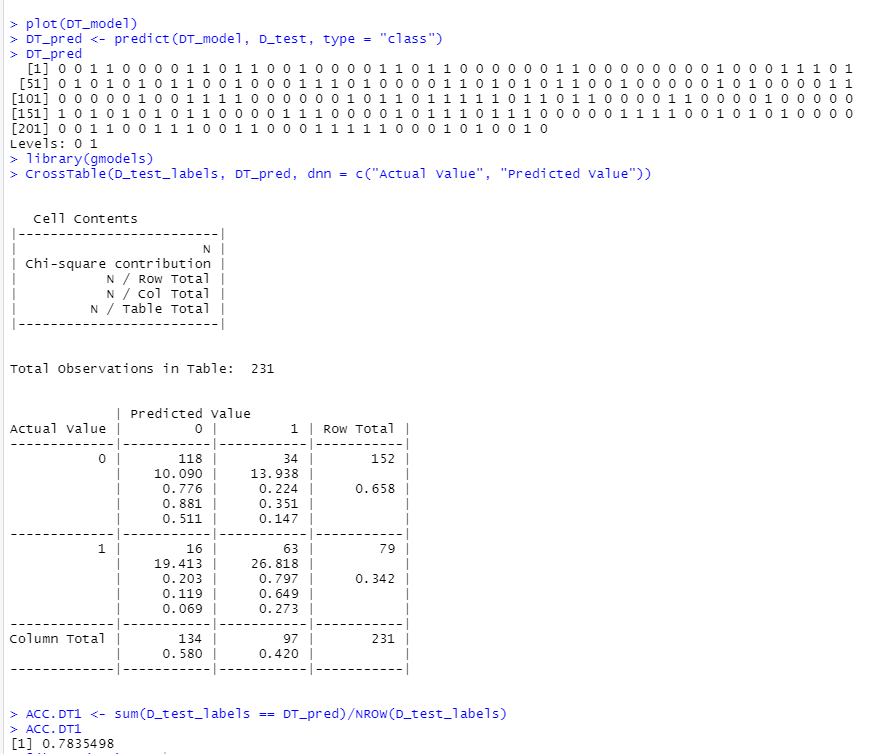
TP – 132 no diabetic, TN – 42 - diabetes, FP – 20 predicted diabetes but aren’t diabetic and FN – 37 diabetics not detected.

**Decision Trees Method (C5.0):**

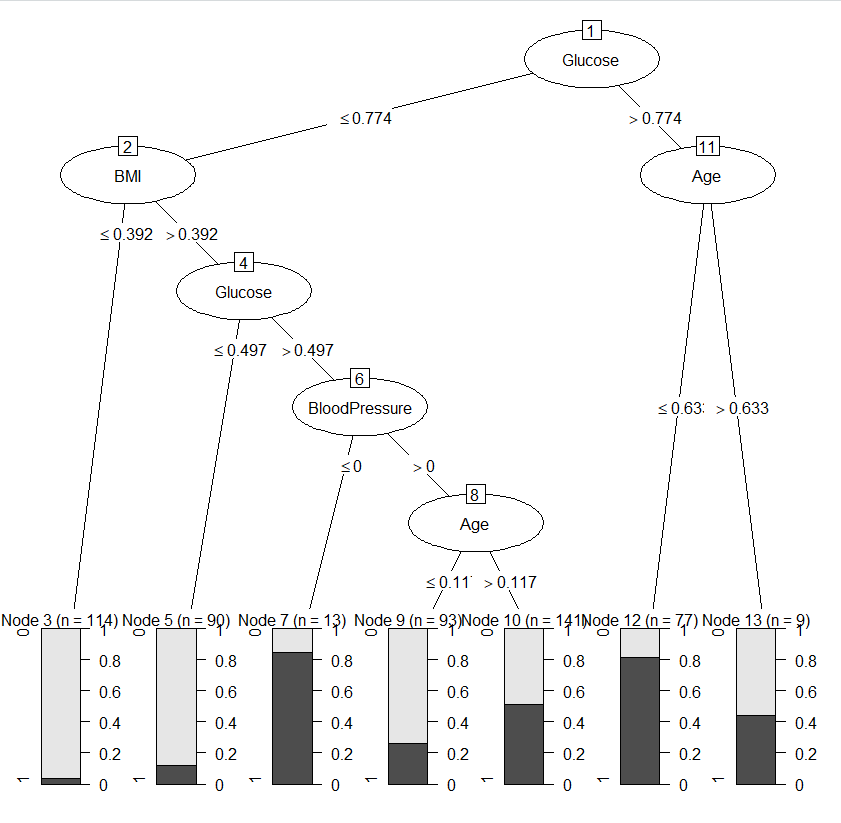
The Decision tree is based on recursive partitioning, where the start is the root node, which represents out entire dataset. The tuning parameter used here is trials. Hence the train sample with trial 5 is taken as it has highest accuracy.

**Model – 1: Decision Trees Method (Parameter: trails =1)**

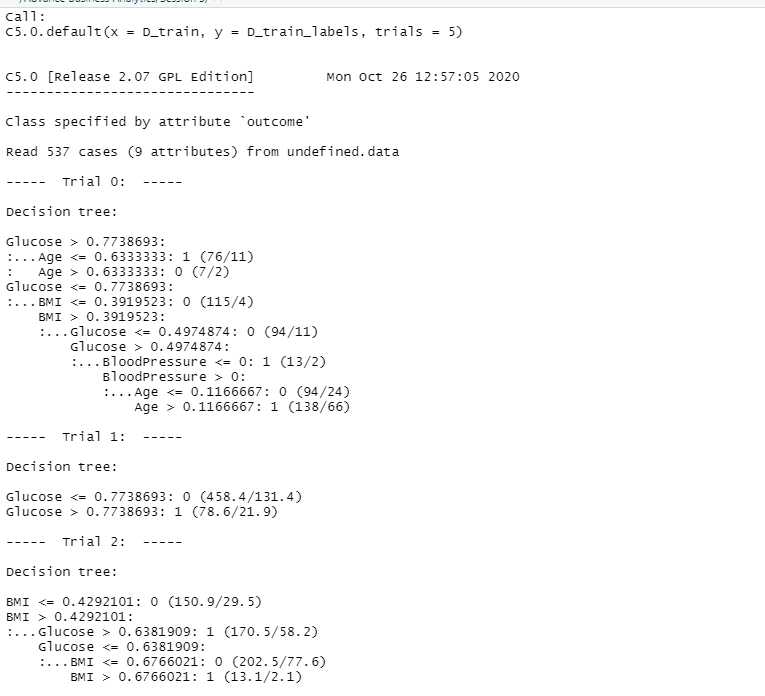


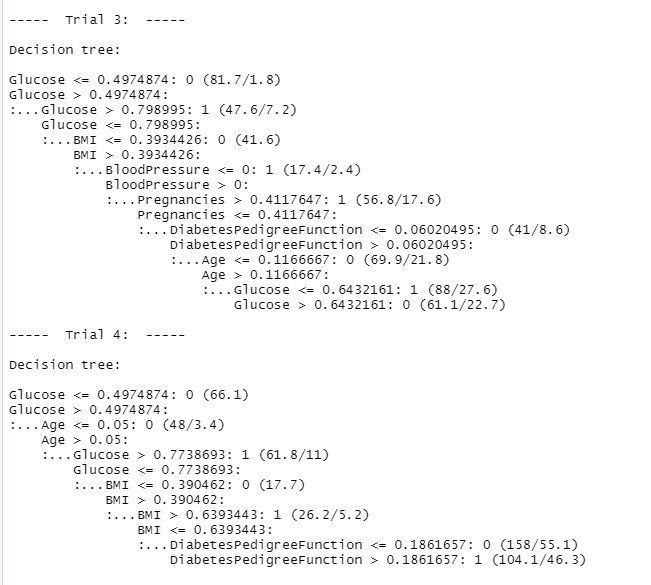


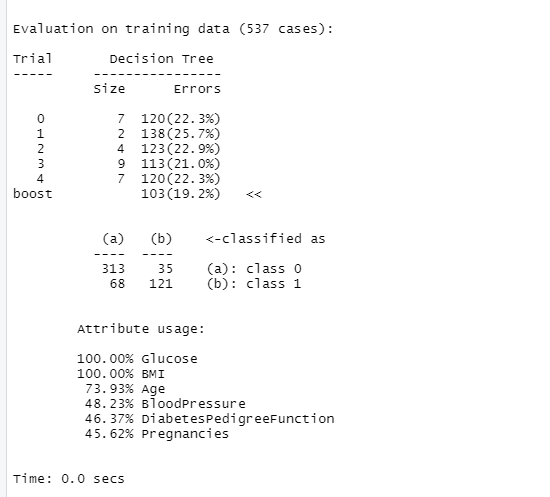
TP – 118 no diabetic, TN – 63 - diabetes, FP – 34 predicted diabetes but aren’t diabetic and FN – 16 diabetics not detected.

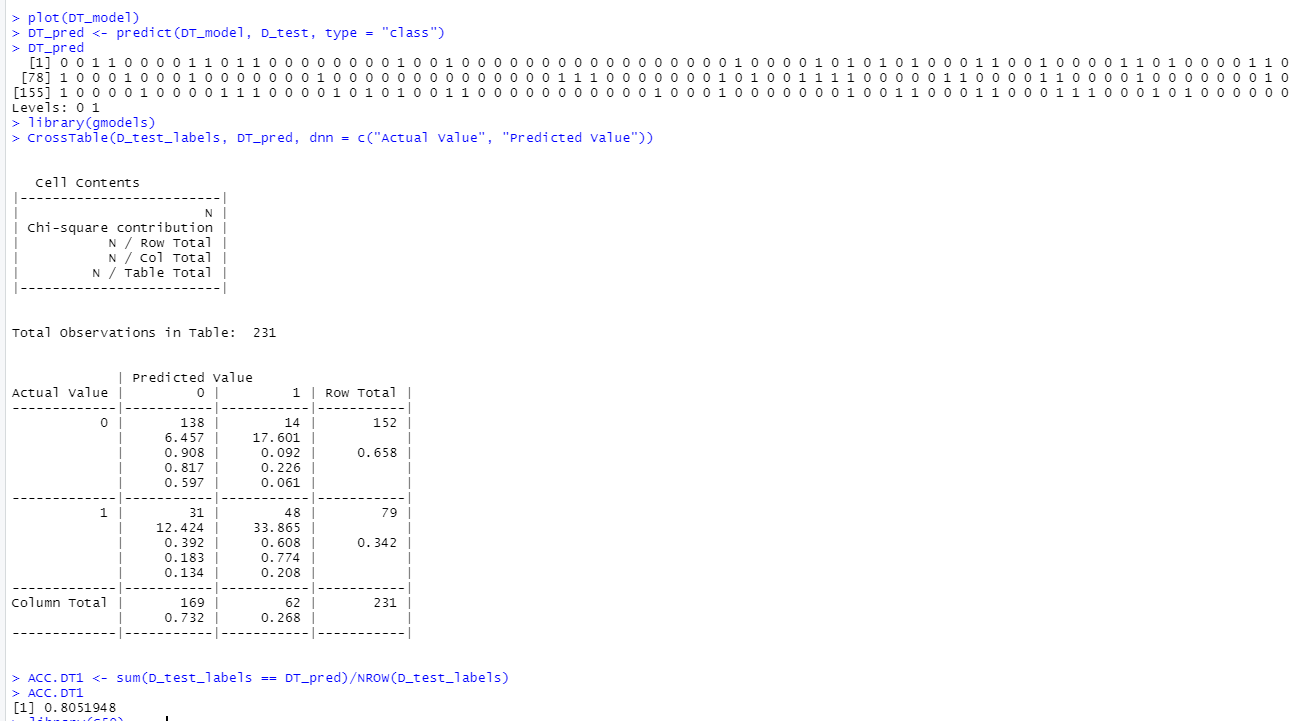


**Mode2 – 1: Decision Trees Method (Parameter: trails =5)**

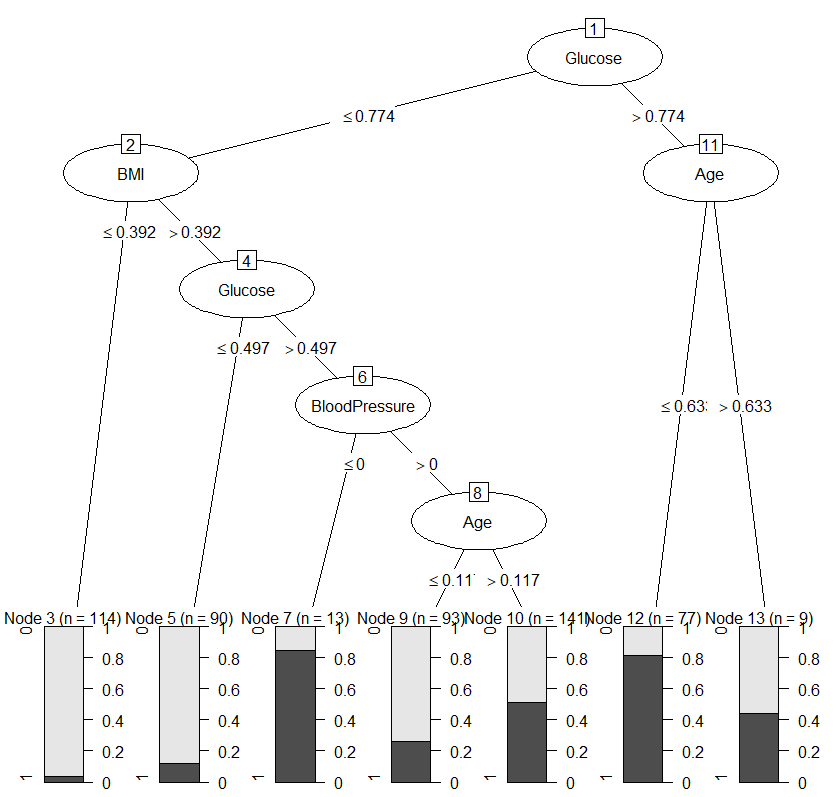




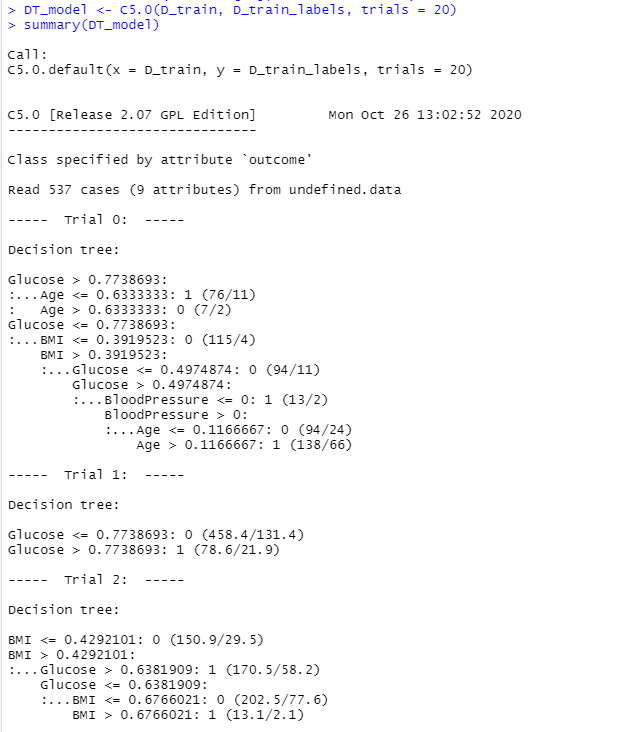


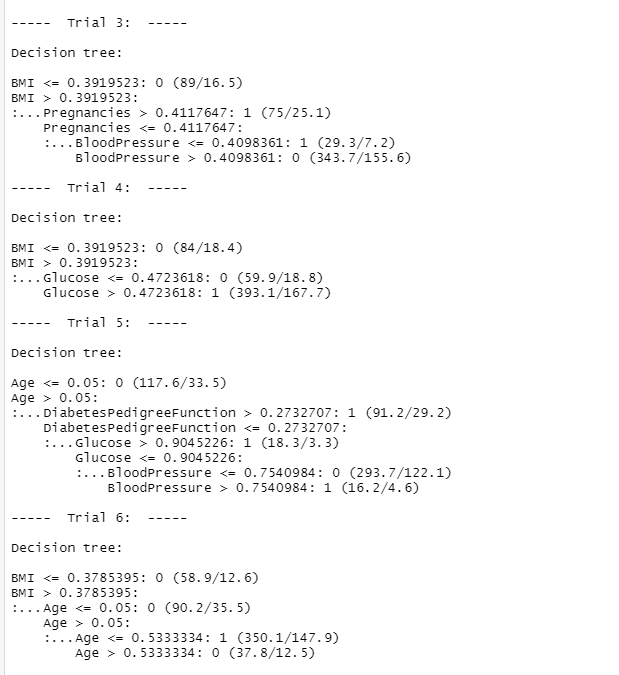


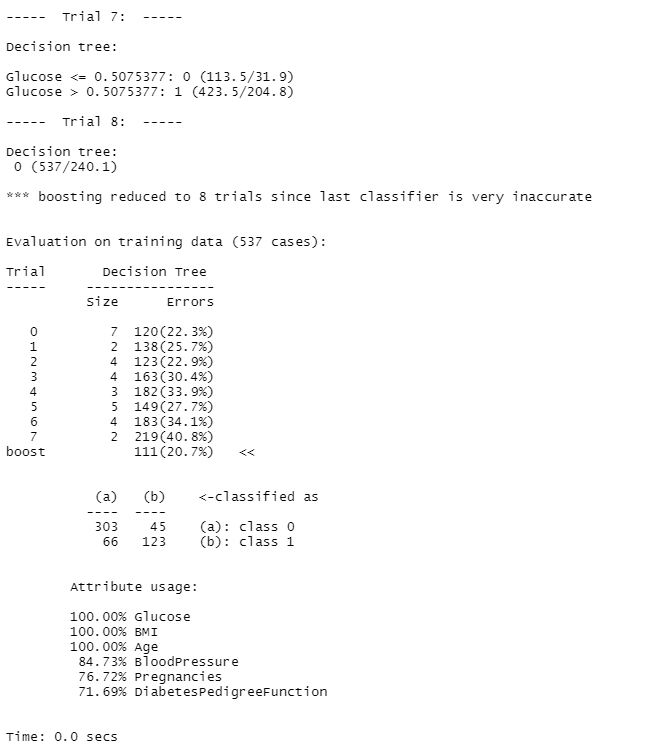
TP – 138 no diabetic, TN – 48 - diabetes, FP – 14 predicted diabetes but aren’t diabetic and FN – 31 diabetics not detected.

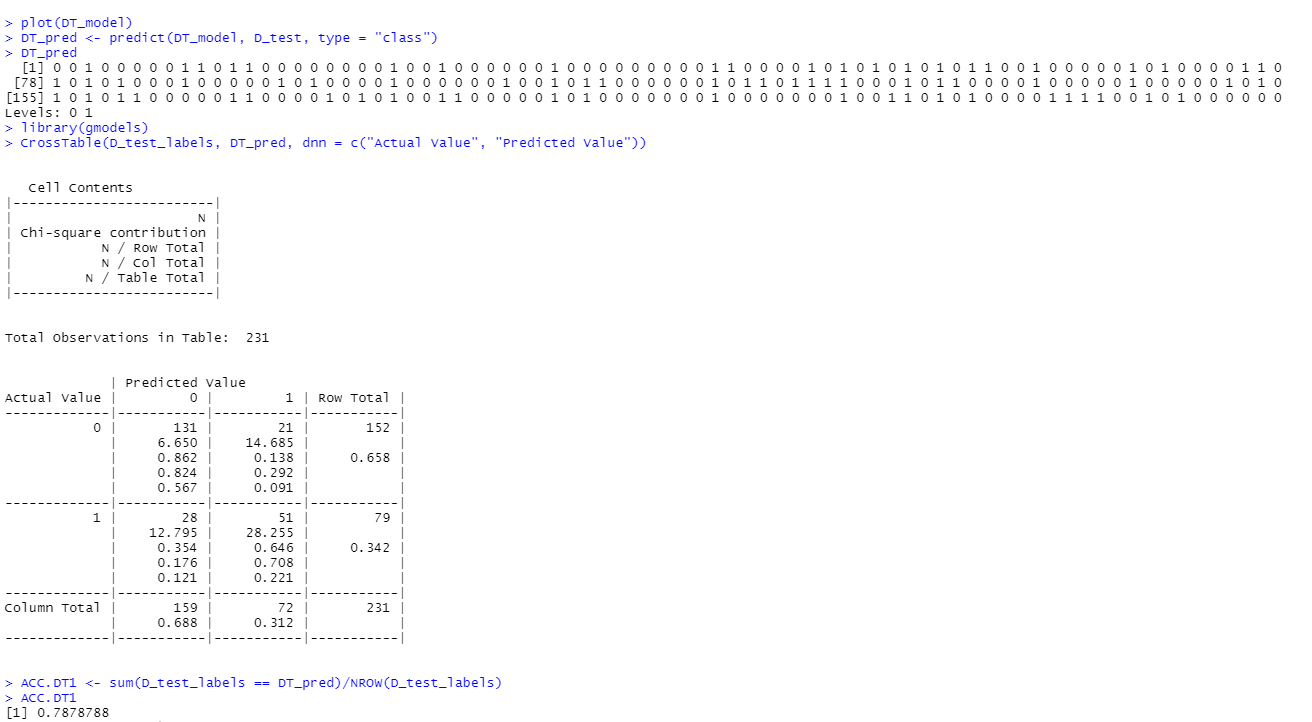


**Model – 3: Decision Trees Method (Parameter: trails =20)**

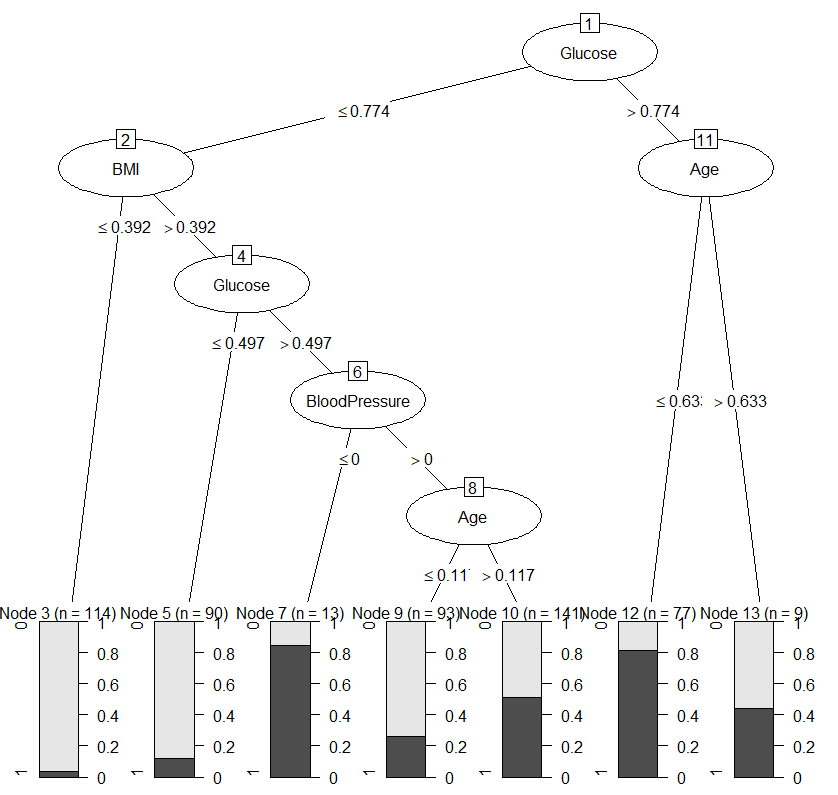








TP – 131 no diabetic, TN – 51 - diabetes, FP – 21 predicted diabetes but aren’t diabetic and FN – 28 diabetics not detected.

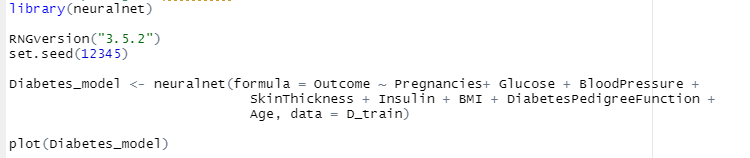


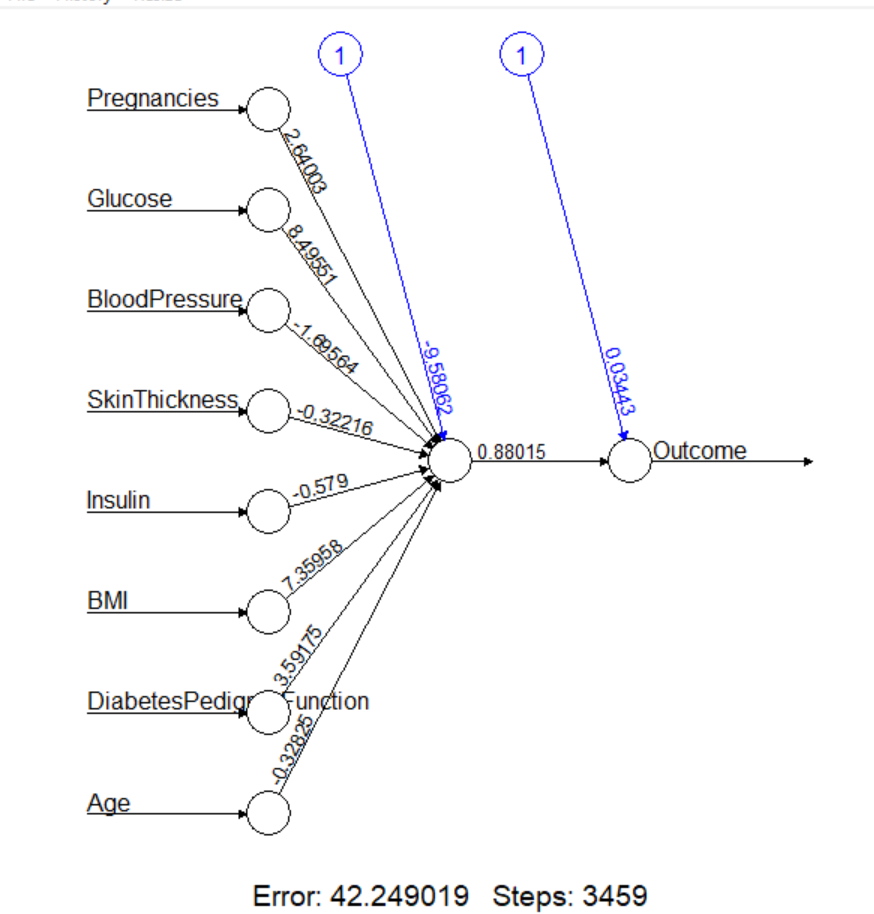
Neural Network:

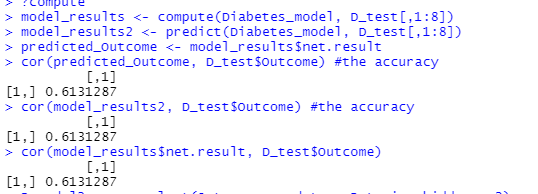
Neural Network or Artificial Neural Network is based on biological neuron system, which modifies the internal structure by adjusting weights of inputs. By pattern recognition it solves the complexities of the machine processing.

The neural network is of input layer (dataset input), hidden layer (optimize weight of input variable), and output layer (output of the prediction). In this study,

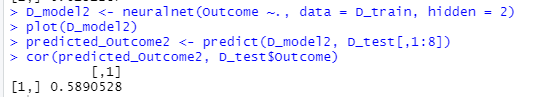
**Model - 1: Neural Network plot (no hidden layer)**

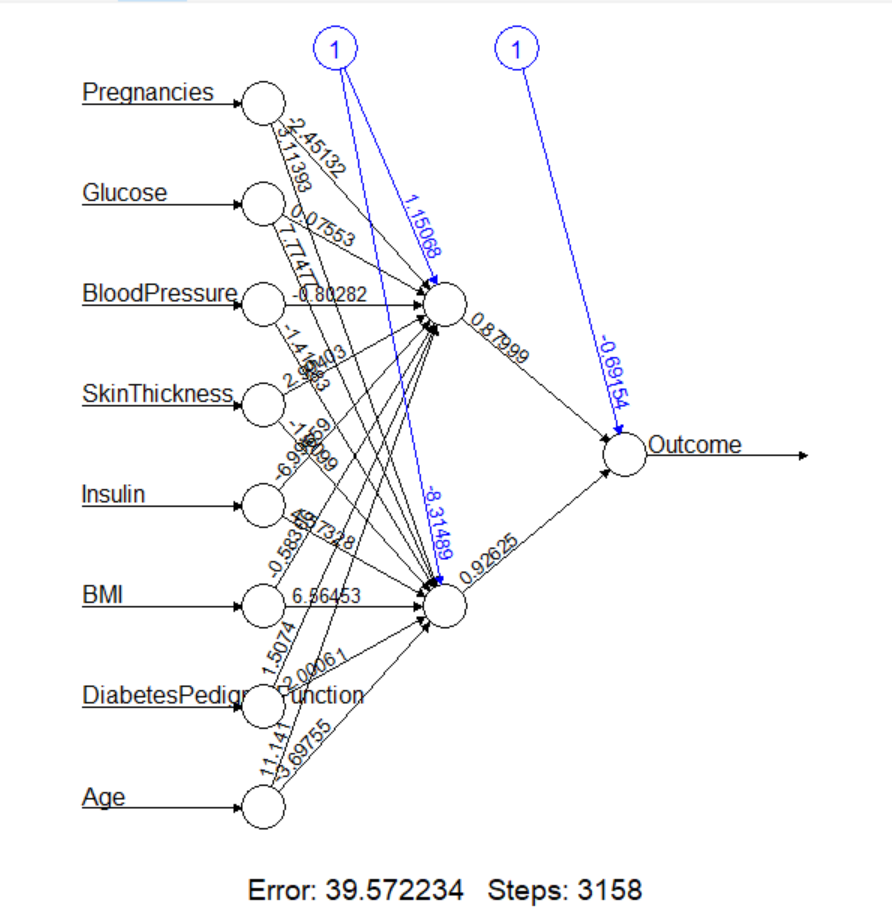




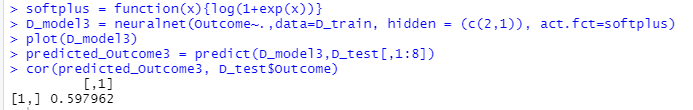


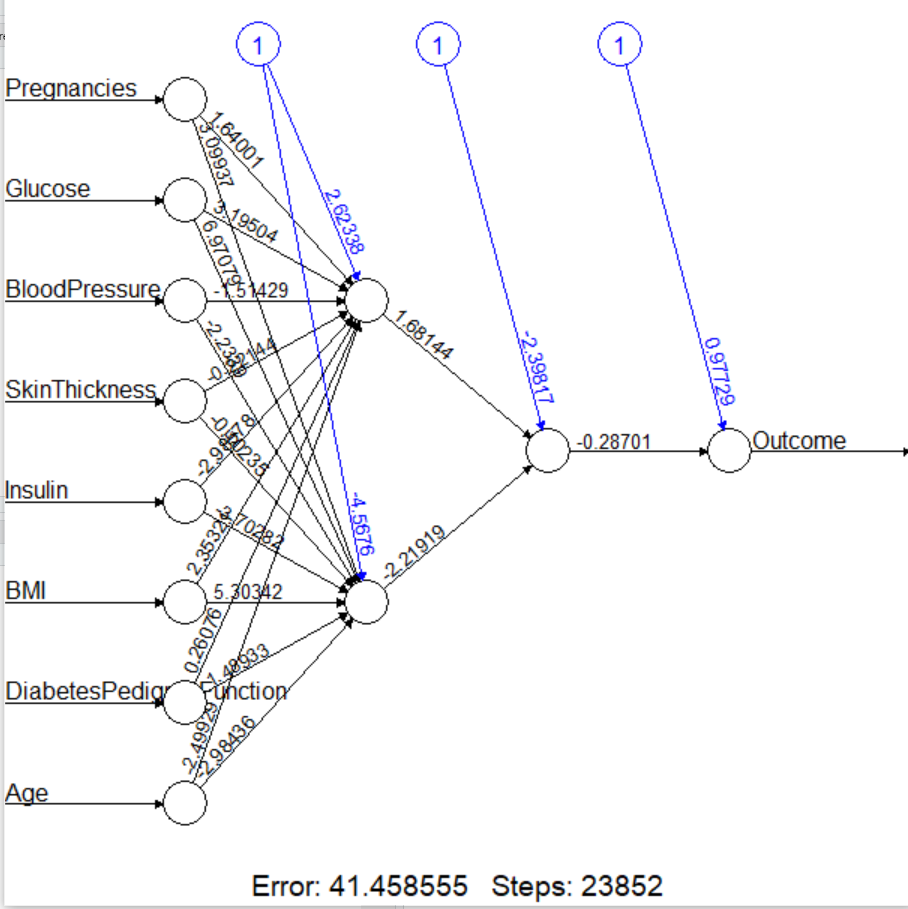
**Model - 2: Neural Network plot (2 hidden layer)**





**Model - 3: Neural Network plot (soft max activation)**





**Appendix II:**

**R programming Code for the Study:**

Diabetes <- read.csv("diabetes.csv", stringsAsFactors = FALSE)

summary(Diabetes[,c(1,2,3,4,5,6,7,8)])

# Preparing the dataset

Diabetes\_1 = na.omit(Diabetes)

dim(Diabetes\_1)

# Frequency Plot for attributes

Diabetes.P <- par(mfrow=c(3, 3))

hist(Diabetes$Pregnancies, main = "Number of Times Pregnant",

xlab = "Pregnancies")

hist(Diabetes$Glucose, main = "Plasma Glucose Concentration",

xlab = "Glucose")

hist(Diabetes$BloodPressure, main = "Blood Pressure",

xlab = "Diastolic blood pressure (mm Hg)")

hist(Diabetes$SkinThickness, main = "SkinThickness",

xlab = "Triceps skin fold thickness (mm)")

hist(Diabetes$Insulin, main = "Insulin",

xlab = "2-Hour serum insulin (mu U/ml)")

hist(Diabetes$BMI, main = "BMI",

xlab = "BMI")

hist(Diabetes$DiabetesPedigreeFunction, main = "DiabetesPedigreeFunction",

xlab = "DiabetesPedigreeFunction")

hist(Diabetes$Age, main = "Age",

xlab = "Age")

hist(Diabetes$Outcome, main = "Outcome",

xlab = "Outcome")

# Boxplot for attributes

Diabetes.BP <- par(mfrow=c(3, 3))

boxplot(Diabetes$Pregnancies, main = "Number of Times Pregnant",

xlab = "Pregnancies")

boxplot(Diabetes$Glucose, main = "Plasma Glucose Concentration",

xlab = "Glucose")

boxplot(Diabetes$BloodPressure, main = "Blood Pressure",

xlab = "Diastolic blood pressure (mm Hg)")

boxplot(Diabetes$SkinThickness, main = "SkinThickness",

xlab = "Triceps skin fold thickness (mm)")

boxplot(Diabetes$Insulin, main = "Insulin",

xlab = "2-Hour serum insulin (mu U/ml)")

boxplot(Diabetes$BMI, main = "BMI",

xlab = "BMI")

boxplot(Diabetes$DiabetesPedigreeFunction, main = "DiabetesPedigreeFunction",

xlab = "DiabetesPedigreeFunction")

boxplot(Diabetes$Age, main = "Age",

xlab = "Age")

boxplot(Diabetes$Outcome, main = "Outcome",

xlab = "Outcome")

#Normalization

normalize <- function(x){

return((x - min(x))/(max(x) - min(x)))

}

Diabetes\_N <- as.data.frame(lapply(Diabetes, normalize))

# Create training and testing data

Diabetes\_D <- Diabetes\_N[-9]

Diabetes\_DL <- Diabetes\_N[9]

D\_train<-Diabetes\_D[1:537,]

D\_test<-Diabetes\_D[538:768,]

# Create labels for training and testing data

D\_train\_labels <- Diabetes\_DL[1:537, ]

D\_test\_labels <- Diabetes\_DL[538:768, ]

D\_train\_labels <- as.factor(D\_train\_labels)

D\_test\_labels <- as.factor(D\_test\_labels)

################################################################################

# KNN Method

library(class)

library(gmodels)

test\_pred\_21 <- knn(train = D\_train, test = D\_test,cl = D\_train\_labels, k = 21)

CrossTable(x = D\_test\_labels, y = test\_pred\_21, prop.chisq = FALSE)

Accuracy.21 <- sum(D\_test\_labels == test\_pred\_21)/NROW(D\_test\_labels)

Accuracy.21

test\_pred\_10 <- knn(train = D\_train, test = D\_test,cl = D\_train\_labels, k = 10)

CrossTable(x = D\_test\_labels, y = test\_pred\_10, prop.chisq = FALSE)

Accuracy.10 <- sum(D\_test\_labels == test\_pred\_10)/NROW(D\_test\_labels)

Accuracy.10

test\_pred\_5 <- knn(train = D\_train, test = D\_test,cl = D\_train\_labels, k = 5)

CrossTable(x = D\_test\_labels, y = test\_pred\_5, prop.chisq = FALSE)

Accuracy.5 <- sum(D\_test\_labels == test\_pred\_5)/NROW(D\_test\_labels)

Accuracy.5

#############################################################################

# Decision Tree

#install.packages("C50")

library(C50)

#help("C5.0")

library(gmodels)

DT\_model <- C5.0(D\_train, D\_train\_labels)

summary(DT\_model)

plot(DT\_model)

DT\_pred <- predict(DT\_model, D\_test, type = "class")

DT\_pred

CrossTable(D\_test\_labels, DT\_pred, dnn = c("Actual Value", "Predicted Value"))

ACC.DT1 <- sum(D\_test\_labels == DT\_pred)/NROW(D\_test\_labels)

ACC.DT1

DT\_model <- C5.0(D\_train, D\_train\_labels, trials = 5)

summary(DT\_model)

plot(DT\_model)

DT\_pred <- predict(DT\_model, D\_test, type = "class")

DT\_pred

CrossTable(D\_test\_labels, DT\_pred, dnn = c("Actual Value", "Predicted Value"))

ACC.DT1 <- sum(D\_test\_labels == DT\_pred)/NROW(D\_test\_labels)

ACC.DT1

DT\_model <- C5.0(D\_train, D\_train\_labels, trials = 20)

summary(DT\_model)

plot(DT\_model)

DT\_pred <- predict(DT\_model, D\_test, type = "class")

DT\_pred

CrossTable(D\_test\_labels, DT\_pred, dnn = c("Actual Value", "Predicted Value"))

ACC.DT1 <- sum(D\_test\_labels == DT\_pred)/NROW(D\_test\_labels)

ACC.DT1

######################################################

#Neural Network

Diabetes\_norm <- as.data.frame(lapply(Diabetes, normalize))

dim(Diabetes)

D\_train <- Diabetes\_N[1:537,]

D\_test <- Diabetes\_N[538:768,]

Diabetes[,'Outcome']<-factor(Diabetes[,'Outcome'])

summary(Diabetes)

# neuralnetwork

#install.packages("neuralnet")

library(neuralnet)

RNGversion("3.5.2")

set.seed(12345)

Diabetes\_model <- neuralnet(formula = Outcome ~ Pregnancies+ Glucose + BloodPressure +

SkinThickness + Insulin + BMI + DiabetesPedigreeFunction +

Age, data = D\_train)

plot(Diabetes\_model)

#predicting method

?compute

model\_results <- compute(Diabetes\_model, D\_test[,1:8])

model\_results2 <- predict(Diabetes\_model, D\_test[,1:8])

predicted\_Outcome <- model\_results$net.result

cor(predicted\_Outcome, D\_test$Outcome) #the accuracy

cor(model\_results2, D\_test$Outcome) #the accuracy

cor(model\_results$net.result, D\_test$Outcome)

D\_model2 <- neuralnet(Outcome ~., data = D\_train, hidden = 2)

plot(D\_model2)

predicted\_Outcome2 <- predict(D\_model2, D\_test[,1:8])

cor(predicted\_Outcome2, D\_test$Outcome)

softplus = function(x){log(1+exp(x))}

D\_model3 = neuralnet(Outcome~.,data=D\_train, hidden = (c(2,1)), act.fct=softplus)

plot(D\_model3)

predicted\_Outcome3 = predict(D\_model3,D\_test[,1:8])

cor(predicted\_Outcome3, D\_test$Outcome)

################################################################################

# Logistic regression

D\_Log <- glm(Outcome ~ Pregnancies + Glucose +

BloodPressure + SkinThickness + Insulin + BMI + DiabetesPedigreeFunction + Age, family = binomial,

data = D\_train)

summary(D\_Log)

# predict the probability of Outcome

D\_Log\_probs=predict(D\_Log, D\_test,type="response")

Log\_pred =ifelse(D\_Log\_probs>=0.5,1,0)

#Accuracy

Accuracy\_Log = mean(Log\_pred==D\_test$Outcome)

Accuracy\_Log

Log\_Table1 <- table(Log\_pred, D\_test$Outcome)

Log\_Table1

D\_Log2 <- glm(Outcome ~ Pregnancies + Glucose + BloodPressure +

BMI + DiabetesPedigreeFunction, family = binomial, data = D\_train)

summary(D\_Log2)

D\_Log\_probs2=predict(D\_Log2, D\_test,type="response")

Log\_pred2 =ifelse(D\_Log\_probs2>=0.5,1,0)

Accuracy\_Log2 = mean(Log\_pred2==D\_test$Outcome)

Accuracy\_Log2

Log\_Table2 <- table(Log\_pred, D\_test$Outcome)

Log\_Table2

#################################################################################