Diabetes Prediction Using Machin Learning

Wail

2023-09-20

Project Scope

The project aim to predict whether the patient has diabetes or not based on the provided dataset from the National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK) for Pima Indian heritage *Females Only*, the dataset needed some cleaning and very high statistical analysis due to the sensitivity of the outcomes after cleaning and preparation i've applied machine learning modeling using three different modules linear regression, decision tree, and naive Bayes in order to have valid accurate and data- driven result.

Install The Required Packages and Load Libraries

```
install.packages('ggplot2', repos = "http://cran.us.r-project.org") #for data visualization
```

```
## package 'ggplot2' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Acc\AppData\Local\Temp\RtmpeWeXAN\downloaded_packages
```

```
install.packages('grid', repos = "http://cran.us.r-project.org") # for grids
install.packages('gridExtra', repos = "http://cran.us.r-project.org") # for arranging the grids
```

```
## package 'gridExtra' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Acc\AppData\Local\Temp\RtmpeWeXAN\downloaded_packages
```

```
install.packages('corrplot', repos = "http://cran.us.r-project.org") # for Correlation plot
```

```
## package 'corrplot' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Acc\AppData\Local\Temp\RtmpeWeXAN\downloaded_packages
```

```
install.packages('caret', repos = "http://cran.us.r-project.org") # for confusion matrix
```

```
## package 'caret' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\Acc\AppData\Local\Temp\RtmpeWeXAN\downloaded_packages
```

```
install.packages('e1071', repos = "http://cran.us.r-project.org") # for naive bayes
```

```
## Error in download.file(url, destfile, method, mode = "wb", ...) :
##   cannot open URL 'http://cran.us.r-project.org/bin/windows/contrib/4.3/e1071_1.7-13.zip'
```

```
install.packages('lattice', repos = "http://cran.us.r-project.org")
```

```
## Error in download.file(url, destfile, method, mode = "wb", ...) :
##   cannot open URL 'http://cran.us.r-project.org/bin/windows/contrib/4.3/lattice_0.21-8.zip'
```

```
library('ggplot2') #for data visualization
library('grid') # for grids
library('gridExtra') # for arranging the grids
library('corrplot') # for Correlation plot
library('lattice')
library('caret') # for confusion matrix
library('e1071') # for naive bayes
library('rpart')
```

Step 1 : Collect Data (upload Diabetes csv file)

```
getwd()
```

```
## [1] "C:/Users/Acc/Desktop/Meri SKILL Internship/Projects/Project 2 - Diabetes Data"
```

```
setwd("C:/Users/Acc/Desktop/Meri SKILL Internship/Projects/Project 2 - Diabetes Data")
diabetes <- read.csv("diabetes.csv")
```

Step 2 : Clean up and Prepare fo Analysis

```
colnames(diabetes) #List of columns name
```

```
## [1] "Pregnancies"              "Glucose"
## [3] "BloodPressure"            "SkinThickness"
## [5] "Insulin"                  "BMI"
## [7] "DiabetesPedigreeFunction" "Age"
## [9] "Outcome"
```

```
nrow(diabetes) #How many rows are in the data frame ?
```

```
## [1] 768
```

```
dim(diabetes) #Dimension of the data frame ?
```

```
## [1] 768   9
```

```r
str(diabetes) #See list of columns and data types(numeric, character, etc)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ Pregnancies             : int  6 1 8 1 0 5 3 10 2 8 ...
##  $ Glucose                 : int  148 85 183 89 137 116 78 115 197 125 ...
##  $ BloodPressure           : int  72 66 64 66 40 74 50 0 70 96 ...
##  $ SkinThickness           : int  35 29 0 23 35 0 32 0 45 0 ...
##  $ Insulin                 : int  0 0 0 94 168 0 88 0 543 0 ...
##  $ BMI                     : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ DiabetesPedigreeFunction: num  0.627 0.351 0.672 0.167 2.288 ...
##  $ Age                     : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ Outcome                 : int  1 0 1 0 1 0 1 0 1 1 ...
```

```r
head(diabetes) #See the first 6 rows of data frame
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35       0 33.6
## 2           1      85            66            29       0 26.6
## 3           8     183            64             0       0 23.3
## 4           1      89            66            23      94 28.1
## 5           0     137            40            35     168 43.1
## 6           5     116            74             0       0 25.6
##   DiabetesPedigreeFunction Age Outcome
## 1                    0.627  50       1
## 2                    0.351  31       0
## 3                    0.672  32       1
## 4                    0.167  21       0
## 5                    2.288  33       1
## 6                    0.201  30       0
```

```r
summary(diabetes) #Statistical Summary of data
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness
##  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
##  Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
##  3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##     Insulin           BMI        DiabetesPedigreeFunction      Age
##  Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
##  1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
##  Median : 30.5   Median :32.00   Median :0.3725           Median :29.00
##  Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24
##  3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
##  Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##     Outcome
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.349
##  3rd Qu.:1.000
##  Max.   :1.000
```

Step 3 : Conduct Statistical Analysis

```
min_glucose <- min(diabetes$Glucose)
print(paste("Minimum Glucose Value :",min_glucose)) #The minimum value of Glucose
```

```
## [1] "Minimum Glucose Value : 0"
```

```
max_glucose <- max(diabetes$Glucose)
print(paste("Maximum Glucose Value :",max_glucose)) #The maximum value of Glucose
```

```
## [1] "Maximum Glucose Value : 199"
```

```
range_Glucose <- range(diabetes$Glucose) #another method to find the minimum and maximum
print(range_Glucose)
```

```
## [1]   0 199
```

```
print(paste("Minimum Glucose Value :",range_Glucose[1]))
```

```
## [1] "Minimum Glucose Value : 0"
```

```
print(paste("Maximum Glucose Value :",range_Glucose[2]))
```

```
## [1] "Maximum Glucose Value : 199"
```

```
Mean_Glucose <- mean(diabetes$Glucose) #The mean value of Glucose
print(paste("Mean of Glucose :",Mean_Glucose))
```

```
## [1] "Mean of Glucose : 120.89453125"
```

```
Median_Glucose <- median(diabetes$Glucose) #The median of Glucose
print(paste("Median of Glucose :",Median_Glucose))
```

```
## [1] "Median of Glucose : 117"
```

```
Mode_Glucose <- table(diabetes$Glucose) #The mode of Glucose using table and sort functions
sort(Mode_Glucose,decreasing = TRUE)
```

```
##
##  99 100 106 111 125 129  95 102 105 108 112 109 122  90 107 114 117 119 120 124
##  17  17  14  14  14  14  13  13  13  13  13  12  12  11  11  11  11  11  11  11
## 128  84 115  88  91  92  97 101 103 123 126 146  96 136 137 139 158  85  87  93
##  11  10  10   9   9   9   9   9   9   9   9   9   8   8   8   8   8   7   7   7
##  94 116 130 144 147  80  81  83  89 104 110 118 121 134 143 151 154 162 173   0
##   7   7   7   7   7   6   6   6   6   6   6   6   6   6   6   6   6   6   6   5
## 113 127 131 132 133 138 140 141 142 145 155 179 180 181  71  74  78 135 148 152
##   5   5   5   5   5   5   5   5   5   5   5   5   5   5   4   4   4   4   4   4
```

```
## 165 168 187 189 197  68  73  79  82  86  98 150 156 161 163 164 166 167 171 183
##   4   4   4   4   4   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
## 184 194 196  57  75  76  77 153 157 159 170 174 175 176 188 193 195  44  56  61
##   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   1   1   1
##  62  65  67  72 149 160 169 172 177 178 182 186 190 191 198 199
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

```
q1 <- quantile(diabetes$Glucose,0.25) #First quartile
print(paste("First Quartile :",q1))
```

```
## [1] "First Quartile : 99"
```

```
q3 <- quantile(diabetes$Glucose,0.75) #Third quartile
print(paste("Third Quartile :",q3))
```

```
## [1] "Third Quartile : 140.25"
```

```
IQR_Glucose <- IQR(diabetes$Glucose) #Interquartile range
print(paste("Interquartile range for Glucose :",IQR_Glucose))
```

```
## [1] "Interquartile range for Glucose : 41.25"
```

```
sd_Glucose <- sd(diabetes$Glucose) #Standard Deviation
print(paste("Standard Deviation for Glucose Column :",sd_Glucose))
```

```
## [1] "Standard Deviation for Glucose Column : 31.9726181951362"
```

```
var_Glucose <- var(diabetes$Glucose) #Variance
print(paste("Variance for Glucose Column :",var_Glucose))
```

```
## [1] "Variance for Glucose Column : 1022.24831425196"
```

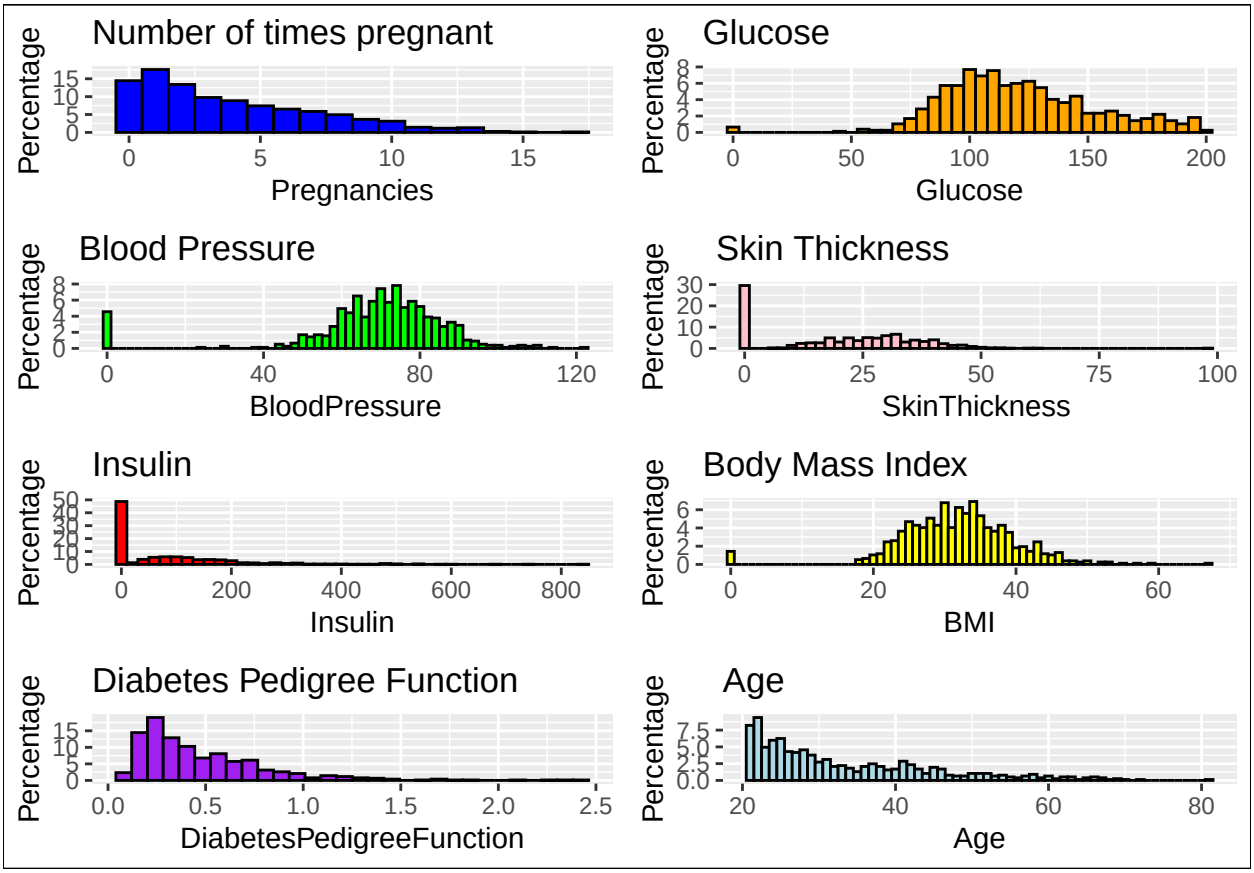
Share Findings and plotting

```
p1 <- ggplot(diabetes, aes(x=Pregnancies)) + ggtitle("Number of times pregnant") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), binwidth = 1, colour="black", fill="blue") + y
p2 <- ggplot(diabetes, aes(x=Glucose)) + ggtitle("Glucose") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), binwidth = 5, colour="black", fill="orange") +
p3 <- ggplot(diabetes, aes(x=BloodPressure)) + ggtitle("Blood Pressure") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), binwidth = 2, colour="black", fill="green") +
p4 <- ggplot(diabetes, aes(x=SkinThickness)) + ggtitle("Skin Thickness") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), binwidth = 2, colour="black", fill="pink") + y
p5 <- ggplot(diabetes, aes(x=Insulin)) + ggtitle("Insulin") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), binwidth = 20, colour="black", fill="red") + y
p6 <- ggplot(diabetes, aes(x=BMI)) + ggtitle("Body Mass Index") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), binwidth = 1, colour="black", fill="yellow") +
p7 <- ggplot(diabetes, aes(x=DiabetesPedigreeFunction)) + ggtitle("Diabetes Pedigree Function") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), colour="black", fill="purple") + ylab("Percen
p8 <- ggplot(diabetes, aes(x=Age)) + ggtitle("Age") +
  geom_histogram(aes(y = 100*(..count..)/sum(..count..)), binwidth=1, colour="black", fill="lightblue")
grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, ncol=2)
```

```
## Warning: The dot-dot notation (‘..count..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(count)‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.

## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```

```
grid.rect(width = 1, height = 1, gp = gpar(lwd = 1, col = "black", fill = NA))
```

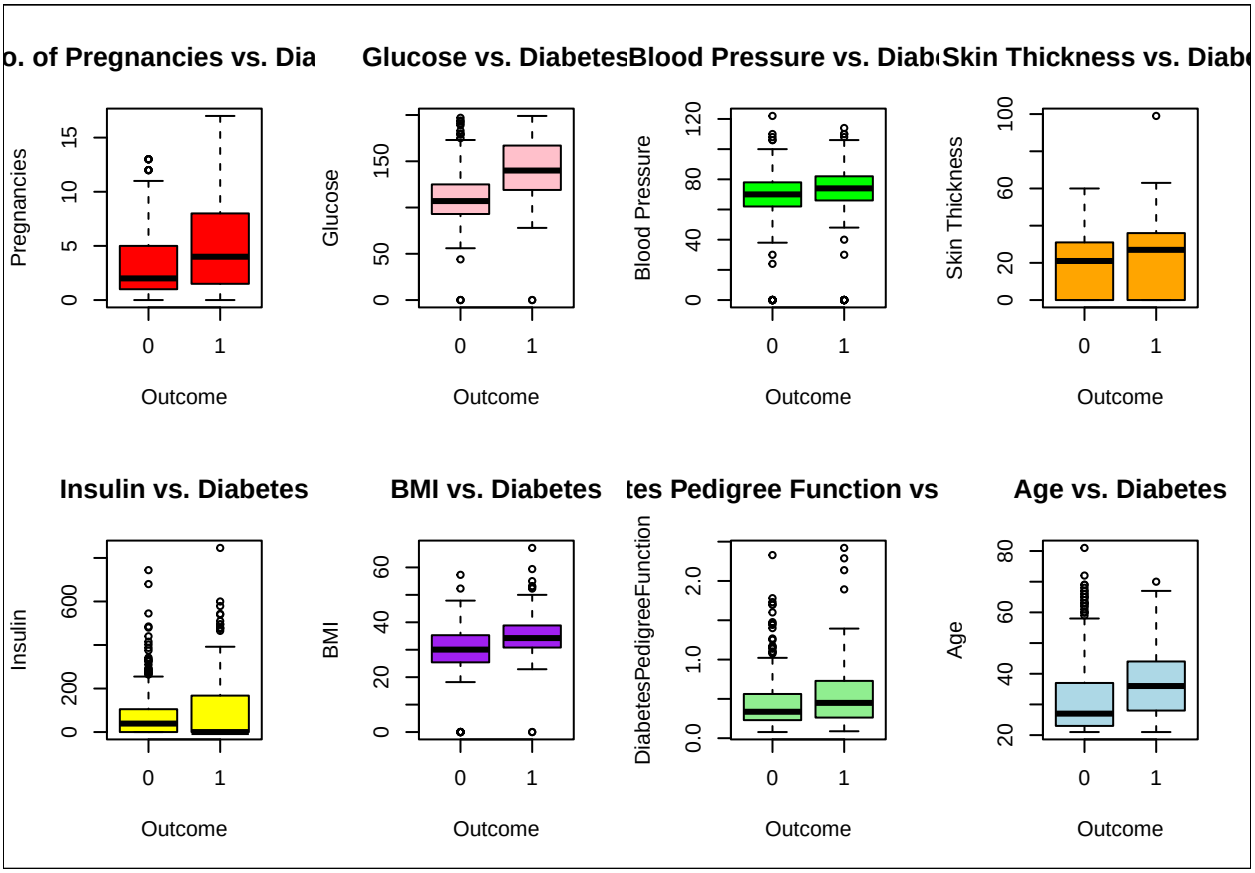


Identify correlation between Numeric Variables and Outcomes to if it's correlated or not

```
numeric.var <- sapply(diabetes, is.numeric)
corr.matrix <- cor(diabetes[,numeric.var])
corrplot(corr.matrix, main="\n\nCorrelation Plot for Numerical Variables", order = "hclust", tl.col = "
box(which = "outer", lty = "solid")
```

```
attach(diabetes)
par(mfrow=c(2,4))
boxplot(Pregnancies~Outcome, main="No. of Pregnancies vs. Diabetes",
        xlab="Outcome", ylab="Pregnancies",col="red")
boxplot(Glucose~Outcome, main="Glucose vs. Diabetes",
        xlab="Outcome", ylab="Glucose",col="pink")
boxplot(BloodPressure~Outcome, main="Blood Pressure vs. Diabetes",
```

```
        xlab="Outcome", ylab="Blood Pressure",col="green")
boxplot(SkinThickness~Outcome, main="Skin Thickness vs. Diabetes",
        xlab="Outcome", ylab="Skin Thickness",col="orange")
boxplot(Insulin~Outcome, main="Insulin vs. Diabetes",
        xlab="Outcome", ylab="Insulin",col="yellow")
boxplot(BMI~Outcome, main="BMI vs. Diabetes",
        xlab="Outcome", ylab="BMI",col="purple")
boxplot(DiabetesPedigreeFunction~Outcome, main="Diabetes Pedigree Function vs. Diabetes", xlab="Outcome
boxplot(Age~Outcome, main="Age vs. Diabetes",
        xlab="Outcome", ylab="Age",col="lightblue")
box(which = "outer", lty = "solid")
```



Create Linear Regression Analysis Model to Check Cross Validiation and Accuracy

```
diabetes$BloodPressure <- NULL
diabetes$SkinThickness <- NULL
train <- diabetes[1:540,]
test <- diabetes[541:768,]
model <-glm(Outcome ~.,family=binomial(link='logit'),data=train)
summary(model)
```

```
##
## Call:
```

```
## glm(formula = Outcome ~ ., family = binomial(link = "logit"),
##     data = train)
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -8.3461752  0.8157916 -10.231  < 2e-16 ***
## Pregnancies               0.1246856  0.0373214   3.341 0.000835 ***
## Glucose                   0.0315778  0.0042497   7.431 1.08e-13 ***
## Insulin                  -0.0013400  0.0009441  -1.419 0.155781
## BMI                       0.0881521  0.0164090   5.372 7.78e-08 ***
## DiabetesPedigreeFunction  0.9642132  0.3430094   2.811 0.004938 **
## Age                       0.0018904  0.0107225   0.176 0.860053
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 700.47  on 539  degrees of freedom
## Residual deviance: 526.56  on 533  degrees of freedom
## AIC: 540.56
##
## Number of Fisher Scoring iterations: 5
```

Cross Validation

```
fitted.results <- predict(model,newdata=test,type='response')
fitted.results <- ifelse(fitted.results > 0.5,1,0)
(conf_matrix_logi<-table(fitted.results, test$Outcome))
```

```
##
## fitted.results   0   1
##              0 136  34
##              1  14  44
```

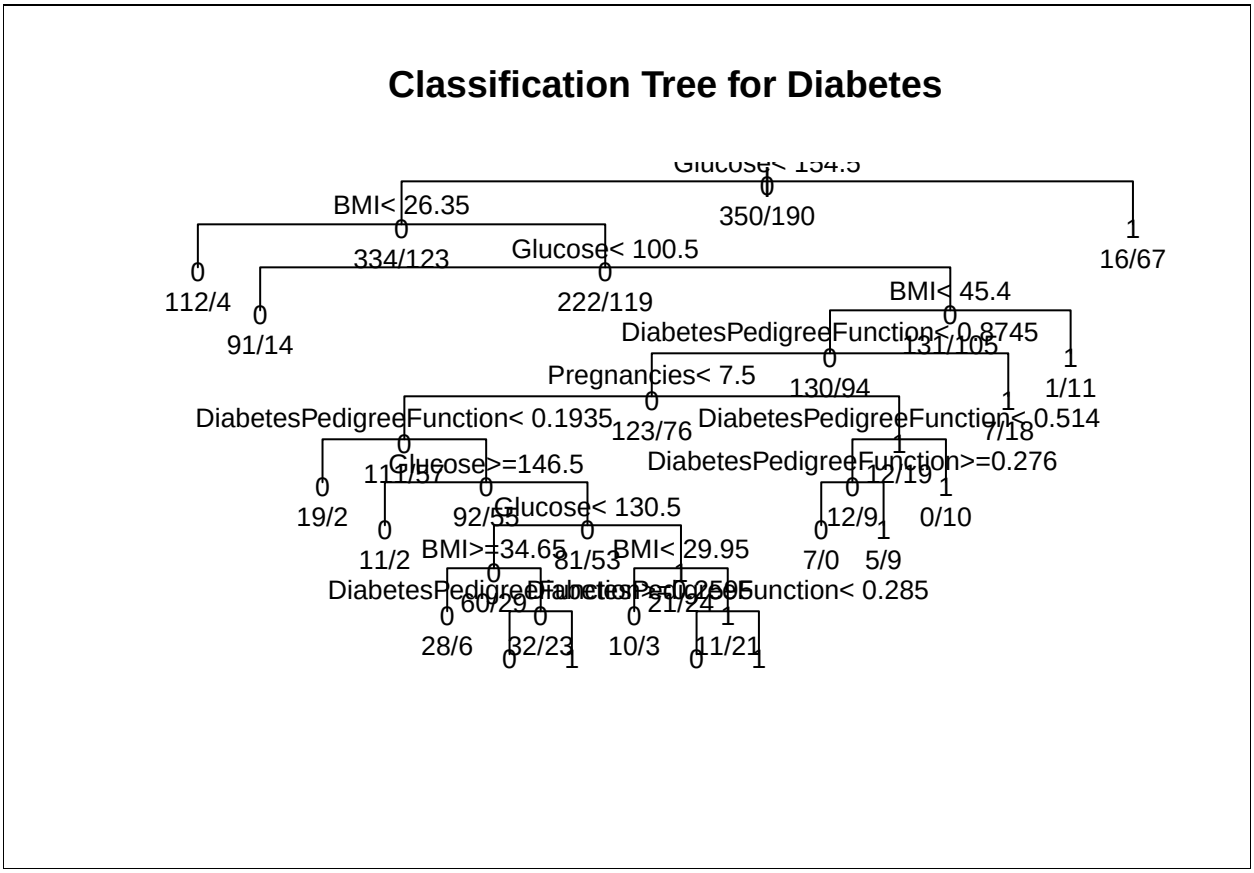
Accuracy

```
misClasificError <- mean(fitted.results != test$Outcome)
print(paste('Accuracy',1-misClasificError))
```

```
## [1] "Accuracy 0.789473684210526"
```

Create Decision Tree Model to check Accuracy

```
model2 <- rpart(Outcome ~ Pregnancies + Glucose + BMI + DiabetesPedigreeFunction, data=train,method="cla
plot(model2, uniform=TRUE,
    main="Classification Tree for Diabetes")
text(model2, use.n=TRUE, all=TRUE, cex=.8)
box(which = "outer", lty = "solid")
```



Accuracy Calculation

```
treePred <- predict(model2, test, type = 'class')
(conf_matrix_dtree<-table(treePred, test$Outcome))
```

```
##
## treePred   0   1
##        0 121  29
##        1  29  49
```

```
mean(treePred==test$Outcome)
```

```
## [1] 0.745614
```

Create Naive Bayes Model

```
model_naive <- naiveBayes(Outcome ~., data = train)
```

Confusion Table and Accuracy Calculation

```r
# predicting target
toppredict_set <- test[1:6]
dim(toppredict_set)
```

```
## [1] 228   6
```

```r
preds_naive <- predict(model_naive, newdata = toppredict_set)
(conf_matrix_naive <- table(preds_naive, test$Outcome))
```

```
##
## preds_naive   0   1
##           0 129  29
##           1  21  49
```

```r
mean(preds_naive==test$Outcome)
```

```
## [1] 0.7807018
```

Compare Accuracy and Sensitivity Level of Our Three Models (Linear Regression, Decision Tree, Naive Byes) to See The Highest Value.

```r
confusionMatrix(conf_matrix_logi)
```

```
## Confusion Matrix and Statistics
##
##
## fitted.results   0   1
##              0 136  34
##              1  14  44
##
##                Accuracy : 0.7895
##                  95% CI : (0.7307, 0.8405)
##     No Information Rate : 0.6579
##     P-Value [Acc > NIR] : 9.506e-06
##
##                   Kappa : 0.5016
##
##  Mcnemar's Test P-Value : 0.006099
##
##             Sensitivity : 0.9067
##             Specificity : 0.5641
##          Pos Pred Value : 0.8000
##          Neg Pred Value : 0.7586
##              Prevalence : 0.6579
##          Detection Rate : 0.5965
##    Detection Prevalence : 0.7456
##       Balanced Accuracy : 0.7354
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(conf_matrix_dtree)
```

```
## Confusion Matrix and Statistics
##
##
## treePred   0   1
##        0 121  29
##        1  29  49
##
##                Accuracy : 0.7456
##                  95% CI : (0.6839, 0.8008)
##     No Information Rate : 0.6579
##     P-Value [Acc > NIR] : 0.002723
##
##                   Kappa : 0.4349
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.8067
##             Specificity : 0.6282
##          Pos Pred Value : 0.8067
##          Neg Pred Value : 0.6282
##              Prevalence : 0.6579
##          Detection Rate : 0.5307
##    Detection Prevalence : 0.6579
##       Balanced Accuracy : 0.7174
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(conf_matrix_naive)
```

```
## Confusion Matrix and Statistics
##
##
## preds_naive   0   1
##           0 129  29
##           1  21  49
##
##                Accuracy : 0.7807
##                  95% CI : (0.7213, 0.8326)
##     No Information Rate : 0.6579
##     P-Value [Acc > NIR] : 3.562e-05
##
##                   Kappa : 0.5005
##
##  Mcnemar's Test P-Value : 0.3222
##
##             Sensitivity : 0.8600
##             Specificity : 0.6282
##          Pos Pred Value : 0.8165
##          Neg Pred Value : 0.7000
##              Prevalence : 0.6579
```

```
##          Detection Rate : 0.5658
##    Detection Prevalence : 0.6930
##       Balanced Accuracy : 0.7441
##
##        'Positive' Class : 0
##
```