Question 1

2023-10-17

1.Data Inspection and exploration:

```r
#Reading the data set diabetes.csv
datasetdiabetesoriginal= read.csv('diabetes.csv', header=TRUE)
head(datasetdiabetesoriginal)
```

```
##   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
sum(datasetdiabetesoriginal$Outcome==0)/nrow(datasetdiabetesoriginal)
```

```
## [1] 0.6510417
```

It is observed that the columns: Glucose, BloodPressure, SkinThichkness, Insulin and BMI, have entries with 0's.Since values of 0 for these variables are not logical, they will become missing values (NA).

```r
#install.packages("tidyverse")
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

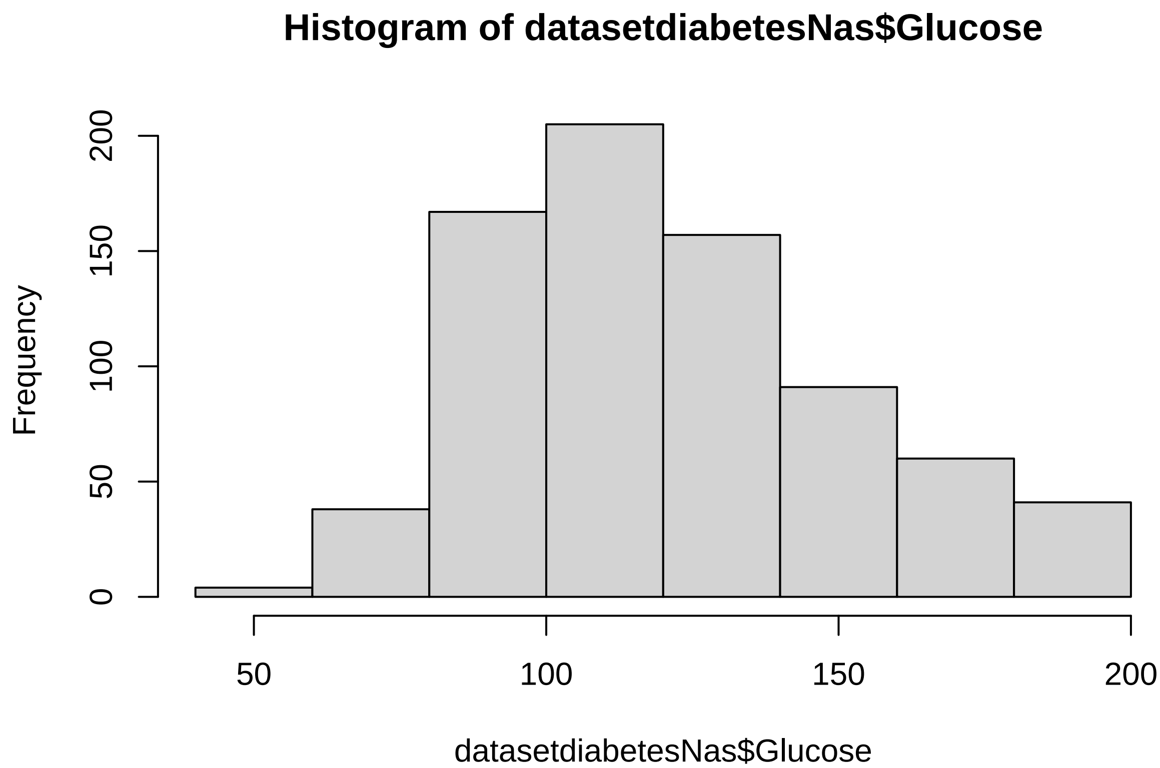
```r
datasetdiabetesNas <-mutate(datasetdiabetesoriginal,
                         Glucose = na_if(Glucose, 0),
                         BloodPressure = na_if(BloodPressure, 0),
                         SkinThickness = na_if(SkinThickness, 0),
                         Insulin = na_if(Insulin, 0),
                         BMI = na_if(BMI, 0))
head(datasetdiabetesNas)
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35      NA 33.6
## 2           1      85            66            29      NA 26.6
## 3           8     183            64            NA      NA 23.3
## 4           1      89            66            23      94 28.1
## 5           0     137            40            35     168 43.1
## 6           5     116            74            NA      NA 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
```

Plots/figures with descriptive statistics

- Histogram for Glucose and Age

```r
hist(datasetdiabetesNas$Glucose)
```



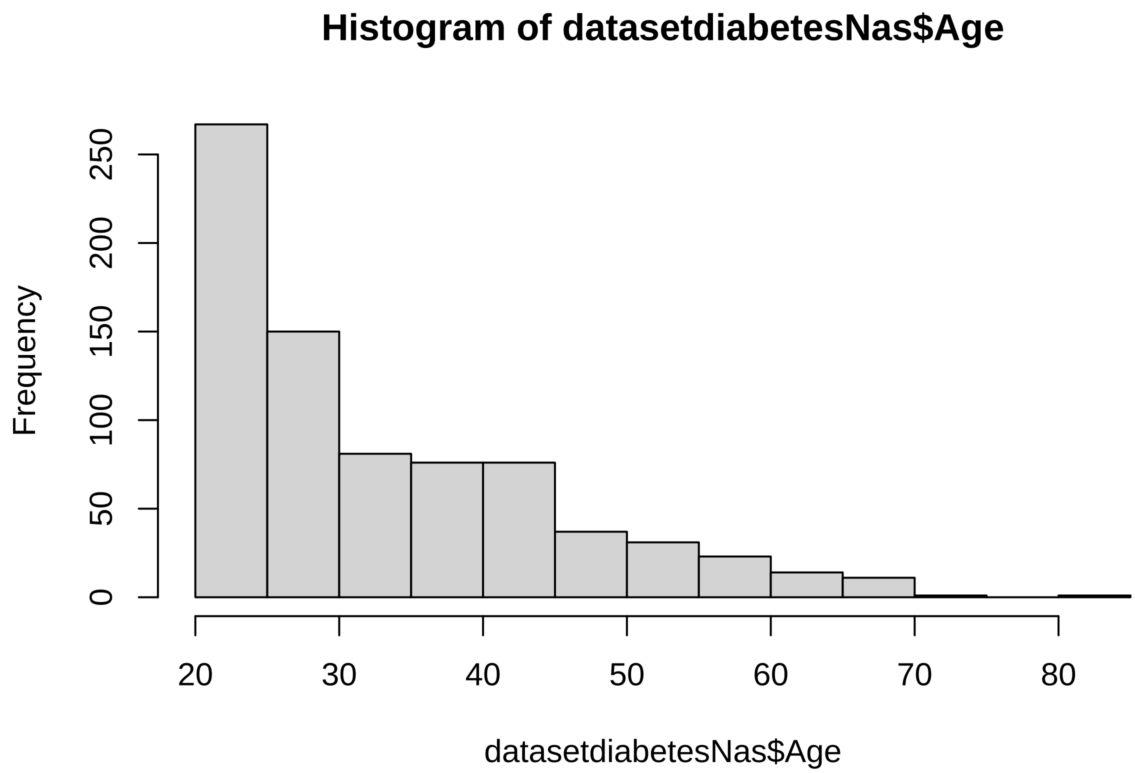

```
mean_glucose=mean(datasetdiabetesNas$Glucose)
sd_glucose=sd(datasetdiabetesNas$Glucose)
print(mean_glucose)
```

```
## [1] NA
```

```
print(sd_glucose)
```

```
## [1] NA
```

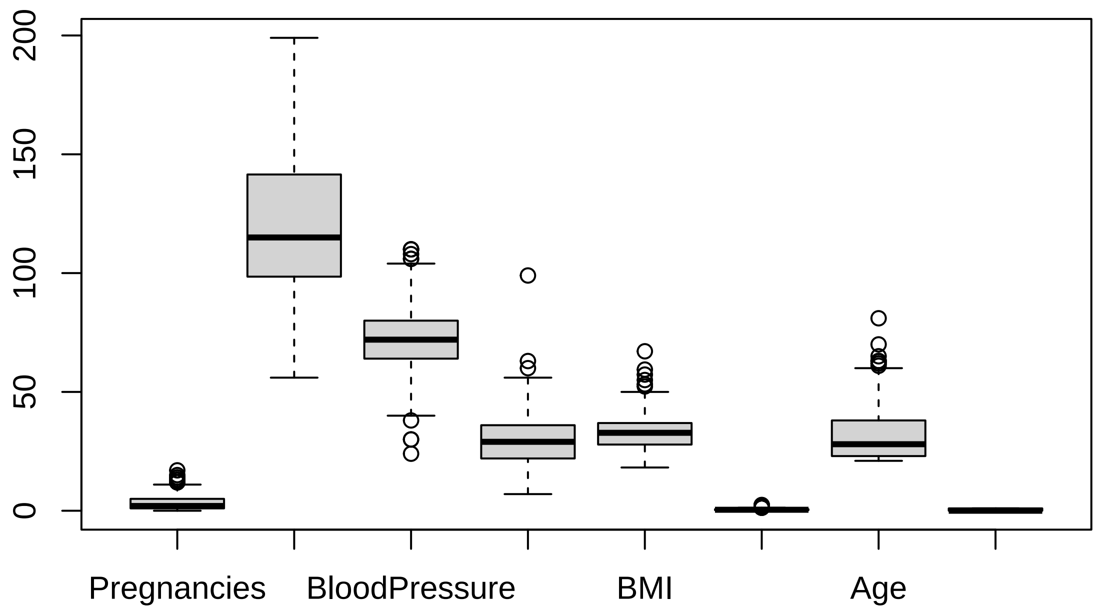
```
hist(datasetdiabetesNas$Age)
```



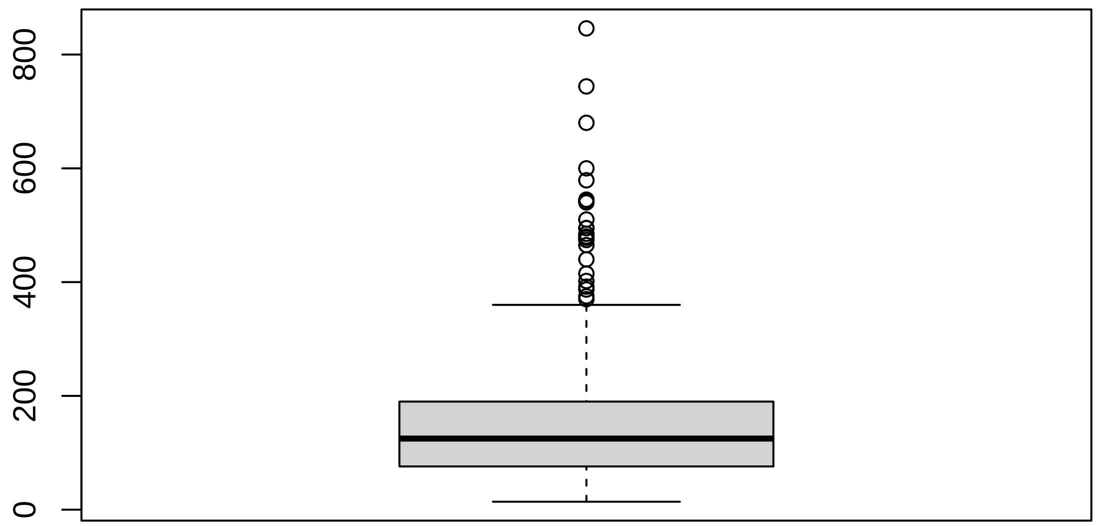
It is observed that the most frequent values for the variable Glucose are between (90,140), and for the variable Age the most frequent values are between (20,30).

-Boxplot for all the variables

```
dataclean=na.omit(subset(datasetdiabetesNas, select = -Insulin))
Insulin=na.omit(subset(datasetdiabetesNas, select = Insulin))
attach(dataclean)
boxplot(dataclean)
```



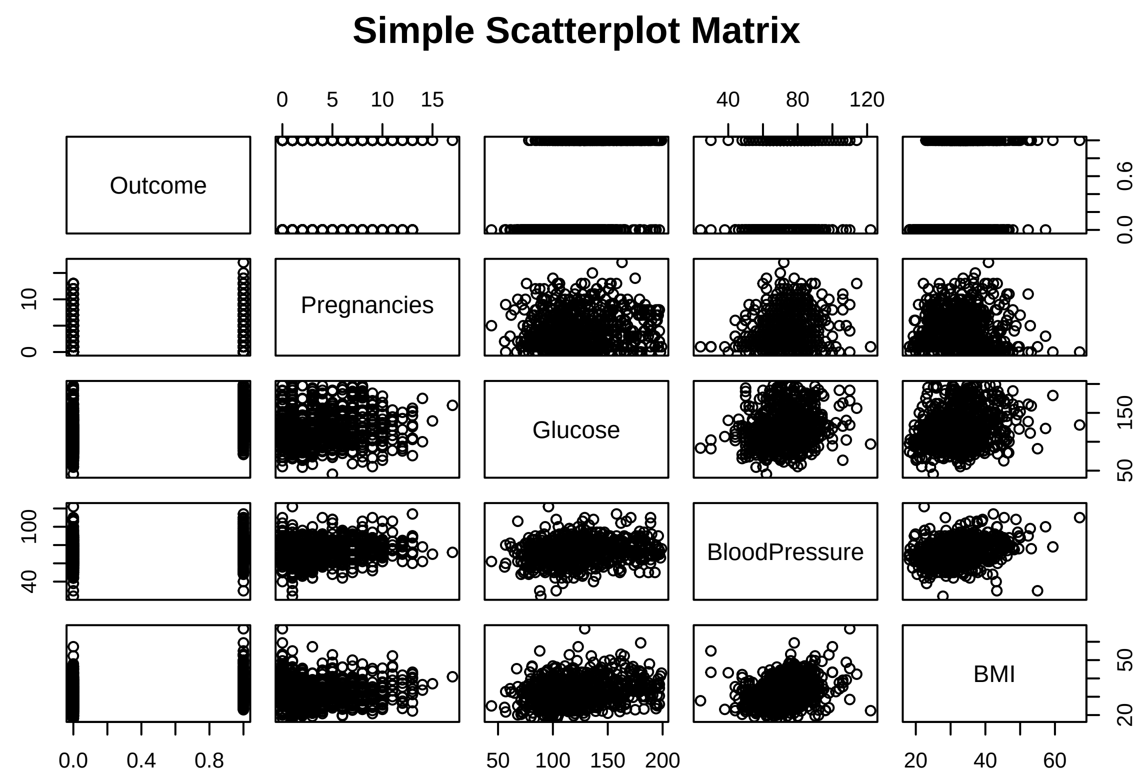

```r
boxplot(Insulin)
```



The outlines for all the variables are observed and NAs are omitted. Because the range of the insulin variable is greater than the other variables, it was graphed separately.

- Scatter Plot:

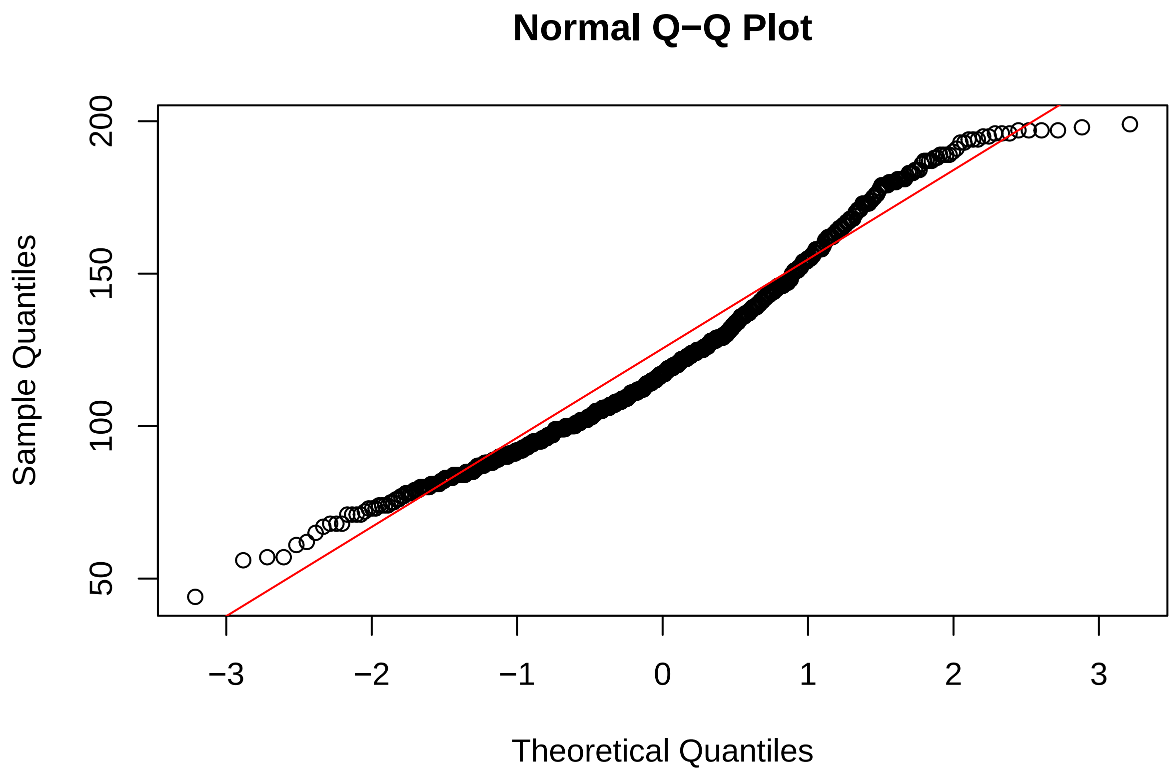
```
pairs(~Outcome+Pregnancies+Glucose+BloodPressure+BMI,data=datasetdiabetesNas,
   main="Simple Scatterplot Matrix")
```



2.

In the QQ Plot, we can observe that the data deviates from normal and is right skewed:

```
##QQ Plot
glucose=na.omit(datasetdiabetesNas$Glucose)
mean_glucose=mean(glucose)
sd_glucose=sd(glucose)
qqnorm(glucose)
qqline(rnorm(100,mean_glucose,sd_glucose), col="red")
```



The Shapiro Wilk Test and the Kolmogorov-Smirnov test, return a significant p-value, then the null hypothesis is rejected and we can conclude that the Glucose variable is not normal.

```
## Shapiro Wilks Test
shapiro.test(glucose) ##not normal
```

```
##
##  Shapiro-Wilk normality test
##
## data:  glucose
## W = 0.96964, p-value = 1.72e-11
```

```
mean_glucose=mean(glucose)
sd_glucose=sd(glucose)
#Kolmogorov Smirnov Test
ks.test(glucose,rnorm(5000,mean_glucose,sd_glucose)) ##not normal
```

```
## Warning in ks.test.default(glucose, rnorm(5000, mean_glucose, sd_glucose)):
## p-value will be approximate in the presence of ties
```

```
##
##  Asymptotic two-sample Kolmogorov-Smirnov test
##
## data:  glucose and rnorm(5000, mean_glucose, sd_glucose)
## D = 0.080831, p-value = 0.0003502
## alternative hypothesis: two-sided
```

```r
ks.test(glucose, "pnorm", mean=mean_glucose, sd=sd_glucose )
```

```
## Warning in ks.test.default(glucose, "pnorm", mean = mean_glucose, sd =
## sd_glucose): ties should not be present for the Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  glucose
## D = 0.072695, p-value = 0.0006291
## alternative hypothesis: two-sided
```

3.

```r
data_output_1=datasetdiabetesNas[datasetdiabetesNas$Outcome==1,]$Glucose
data_output_0=datasetdiabetesNas[datasetdiabetesNas$Outcome==0,]$Glucose
t.test(data_output_1,data_output_0)
```

```
##
##  Welch Two Sample t-test
##
## data:  data_output_1 and data_output_0
## t = 14.884, df = 466.02, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  27.49380 35.85757
## sample estimates:
## mean of x mean of y
##  142.3195  110.6439
```

The P-value is significant, the confidence interval at 95% is (27.4,35.8), then the difference of the means is not equal to 0 and the t value is equal to 14.88 showing how far appart are the means from each other. In conclusion the means are different between glucose with outcome = 1 and Outcome = 0 and the difference is statistically significant.

4.

```r
## Convert the outcome to factor
datasetdiabetesNas$Outcome <- factor(datasetdiabetesNas$Outcome)
is.factor(datasetdiabetesNas$Outcome)
```

```
## [1] TRUE
```

```r
## Training and Testing
set.seed(21)
n=nrow(datasetdiabetesNas)
sample <- sample(c(TRUE, FALSE), n, replace=TRUE, prob=c(0.7,0.3))
train  <- datasetdiabetesNas[sample, ]
test   <- datasetdiabetesNas[!sample, ]

##probabilities of outcome in both datasets
prop.table(table(train$Outcome))
```

```
##
##         0         1
## 0.6666667 0.3333333
```

```r
prop.table(table(test$Outcome))
```

```
##
##        0        1
## 0.617284 0.382716
```

```r
# Dataset without insulin
datasetdiabetes_no_insulin <-select(datasetdiabetesNas,-Insulin)


## Training and Testing without Insulin
train_no_insulin <- datasetdiabetes_no_insulin[sample, ]
test_no_insulin <- datasetdiabetes_no_insulin[!sample, ]
```

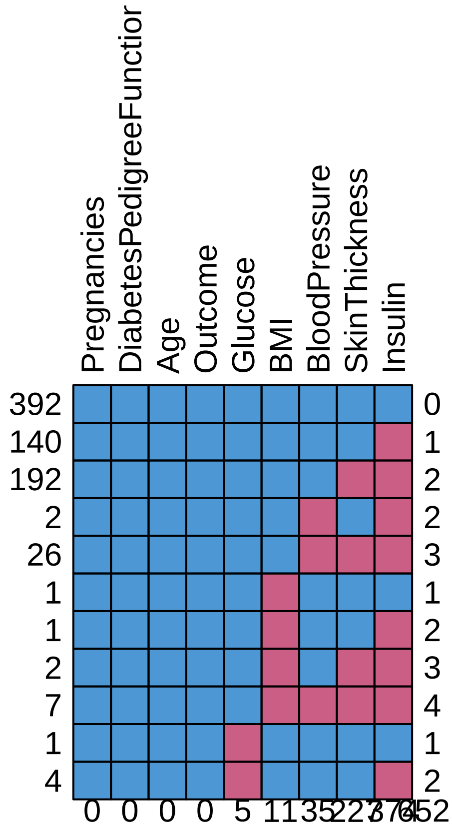
5. Logistic Regresion Model

Multiple imputation for the Diabetes Dataset

```r
#install.packages("mice")
library(mice)
```

```
##
## Attaching package: 'mice'

## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```r
md.pattern(datasetdiabetesNas,rotate.names=TRUE)
```




```
##     Pregnancies DiabetesPedigreeFunction Age Outcome Glucose BMI BloodPressure
## 392           1                        1   1       1       1   1             1
## 140           1                        1   1       1       1   1             1
## 192           1                        1   1       1       1   1             1
## 2             1                        1   1       1       1   1             0
## 26            1                        1   1       1       1   1             0
## 1             1                        1   1       1       1   0             1
## 1             1                        1   1       1       1   0             1
## 2             1                        1   1       1       1   0             1
## 7             1                        1   1       1       1   0             0
## 1             1                        1   1       1       0   1             1
## 4             1                        1   1       1       0   1             1
##               0                        0   0       0       5  11            35
##     SkinThickness Insulin
## 392             1       1   0
## 140             1       0   1
## 192             0       0   2
## 2               1       0   2
## 26              0       0   3
## 1               1       1   1
## 1               1       0   2
## 2               0       0   3
## 7               0       0   4
```

```
## 1                 1        1   1
## 4                 1        0   2
##                 227      374 652
```

```r
nrow(datasetdiabetesNas)
```

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

```r
summary(datasetdiabetesNas)
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness
##  Min.   : 0.000   Min.   : 44.0   Min.   : 24.00   Min.   : 7.00
##  1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 64.00   1st Qu.:22.00
##  Median : 3.000   Median :117.0   Median : 72.00   Median :29.00
##  Mean   : 3.845   Mean   :121.7   Mean   : 72.41   Mean   :29.15
##  3rd Qu.: 6.000   3rd Qu.:141.0   3rd Qu.: 80.00   3rd Qu.:36.00
##  Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
##                   NA's   :5       NA's   :35       NA's   :227
##     Insulin            BMI        DiabetesPedigreeFunction      Age
##  Min.   : 14.00   Min.   :18.20   Min.   :0.0780           Min.   :21.00
##  1st Qu.: 76.25   1st Qu.:27.50   1st Qu.:0.2437           1st Qu.:24.00
##  Median :125.00   Median :32.30   Median :0.3725           Median :29.00
##  Mean   :155.55   Mean   :32.46   Mean   :0.4719           Mean   :33.24
##  3rd Qu.:190.00   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
##  Max.   :846.00   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##  NA's   :374      NA's   :11
##  Outcome
##  0:500
##  1:268
##
##
##
##
##
```

We can observe that 51% of rows have 0 missing values, 18.4% have 1 missing value, 25.9% have two missing values and 4% have 3 or 4 missing values.

It is also observed that the most common missing values are: 1. Insulin 2. Skin Thickness 3. Blood Pressure.

To solve the problem of missing data, I'll use Multiple Imputation with the mice package.

```r
#Multiple imputation
m=5
diabetes.imp.data <-mice(data=train,m=m,maxit=10,print=FALSE)
diabetes.imp.data
```

```
## Class: mids
## Number of multiple imputations:  5
## Imputation methods:
##              Pregnancies                  Glucose            BloodPressure
##                       ""                    "pmm"                    "pmm"
##            SkinThickness                  Insulin                      BMI
```

```
##                    "pmm"                    "pmm"                    "pmm"
## DiabetesPedigreeFunction                      Age                  Outcome
##                       ""                       ""                       ""
## PredictorMatrix:
##                 Pregnancies Glucose BloodPressure SkinThickness Insulin BMI
## Pregnancies               0       1             1             1       1   1
## Glucose                   1       0             1             1       1   1
## BloodPressure             1       1             0             1       1   1
## SkinThickness             1       1             1             0       1   1
## Insulin                   1       1             1             1       0   1
## BMI                       1       1             1             1       1   0
##                 DiabetesPedigreeFunction Age Outcome
## Pregnancies                            1   1       1
## Glucose                                1   1       1
## BloodPressure                          1   1       1
## SkinThickness                          1   1       1
## Insulin                                1   1       1
## BMI                                    1   1       1
```

```
#plot(diabetes.imp.data)
#stripplot(diabetes.imp.data)
```

The method used for all the variables was pmm, predictive mean matching.

Logistic Regression Model

```
## Ordinary Logistic regression with missing data

model_out <- glm(Outcome ~ ., data = train, family = binomial() )
summary(model_out)
```

```
##
## Call:
## glm(formula = Outcome ~ ., family = binomial(), data = train)
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -9.7723144  1.4523891  -6.728 1.71e-11 ***
## Pregnancies              0.0141226  0.0739732   0.191  0.84859
## Glucose                  0.0408724  0.0070718   5.780 7.49e-09 ***
## BloodPressure           -0.0104526  0.0138985  -0.752  0.45201
## SkinThickness            0.0176740  0.0209982   0.842  0.39996
## Insulin                 -0.0008474  0.0015001  -0.565  0.57214
## BMI                      0.0542818  0.0339766   1.598  0.11013
## DiabetesPedigreeFunction  0.6933936  0.5436851   1.275  0.20218
## Age                      0.0612131  0.0234404   2.611  0.00902 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 347.54  on 272  degrees of freedom
## Residual deviance: 232.01  on 264  degrees of freedom
##   (252 observations deleted due to missingness)
## AIC: 250.01
##
## Number of Fisher Scoring iterations: 5
```

```r
## Ordinary Logistic regression without Insulin
model_out_no_insulin <- glm(Outcome ~ ., data = train_no_insulin, family = binomial() )
summary(model_out_no_insulin)
```

```
##
## Call:
## glm(formula = Outcome ~ ., family = binomial(), data = train_no_insulin)
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -9.211504   1.170602  -7.869 3.57e-15 ***
## Pregnancies               0.080962   0.057653   1.404   0.1602
## Glucose                   0.037250   0.005098   7.306 2.75e-13 ***
## BloodPressure            -0.020781   0.012132  -1.713   0.0867 .
## SkinThickness             0.009927   0.018117   0.548   0.5837
## BMI                       0.076829   0.029499   2.604   0.0092 **
## DiabetesPedigreeFunction  1.086194   0.470849   2.307   0.0211 *
## Age                       0.047688   0.017522   2.722   0.0065 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 473.12  on 374  degrees of freedom
## Residual deviance: 317.01  on 367  degrees of freedom
##   (150 observations deleted due to missingness)
## AIC: 333.01
##
## Number of Fisher Scoring iterations: 5
```

```r
## Logistic Regression with multiple imputation

models <- lapply(1:m, function(i) {
  model <- glm(Outcome ~ ., data = complete(diabetes.imp.data, i), family = "binomial")
  return(model)
})

pooled_model <- pool(models)
summary(pooled_model)
```

```
##                       term      estimate   std.error  statistic        df
## 1              (Intercept) -9.1543668146 1.037998902 -8.8192452 352.57464
## 2              Pregnancies  0.1253654019 0.042542323  2.9468396 477.98570
## 3                  Glucose  0.0381349827 0.005536734  6.8876318 119.51405
## 4            BloodPressure -0.0111268182 0.010262699 -1.0842000 486.93335
## 5            SkinThickness  0.0037547189 0.017664404  0.2125585  42.14727
```

```
## 6                  Insulin -0.0007588956 0.001438466 -0.5275730  24.49300
## 7                      BMI  0.0922064519 0.027317256  3.3753922 130.18166
## 8 DiabetesPedigreeFunction  0.7711316967 0.376289769  2.0493029 508.27392
## 9                      Age  0.0169225306 0.011906417  1.4212950 446.95173
##        p.value
## 1 5.396548e-17
## 2 3.367284e-03
## 3 2.829052e-10
## 4 2.788127e-01
## 5 8.326961e-01
## 6 6.025427e-01
## 7 9.718508e-04
## 8 4.094529e-02
## 9 1.559283e-01
```

```r
AIC <- lapply(1:m,function(i) {
  aic_i <-models[[i]]$aic
  return(aic_i)
})
print(AIC)
```

```
## [[1]]
## [1] 494.5971
##
## [[2]]
## [1] 491.4894
##
## [[3]]
## [1] 494.8755
##
## [[4]]
## [1] 490.7071
##
## [[5]]
## [1] 492.4589
```

```r
##Size of data without NAs
dimensions_without_Nas=dim(na.omit(datasetdiabetesNas))
dimenstios_without_Insulin=dim(na.omit(datasetdiabetes_no_insulin))
print(dimensions_without_Nas)
```

```
## [1] 392   9
```

```r
print(dimenstios_without_Insulin)
```

```
## [1] 532   8
```

- In the first model 252 rows were deleted due NAs, although this model is the one with the lowest AIC, because it has less data is less reliable that the model without the variable Insulin. The low AIC could indicate that the model is overfiting the data.
- In the model without the variable insulin the AIC is 333, the model lays in the middle of the other two models. -In the model with multiple imputation due to the large number of outliers and missing data, the AIC is higher that indicates that worse regression results were generated (even for larger numbers of multiple imputations like m=20).

6. Predictions, Results and analysis

Model Without Insulin:

```r
#install.packages("caret")
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
#Accuracy for training data
train_no_insulin <-na.omit(train_no_insulin)
predicted_probs <- predict(model_out_no_insulin, type="response")
predicted_class <- ifelse(predicted_probs > 0.5, 1, 0)
confusionMatrix(as.factor(predicted_class), as.factor(train_no_insulin$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 228  50
##          1  25  72
##
##                Accuracy : 0.8
##                  95% CI : (0.7559, 0.8393)
##     No Information Rate : 0.6747
##     P-Value [Acc > NIR] : 4.642e-08
##
##                   Kappa : 0.5189
##
##  Mcnemar's Test P-Value : 0.005584
##
##             Sensitivity : 0.9012
##             Specificity : 0.5902
##          Pos Pred Value : 0.8201
##          Neg Pred Value : 0.7423
##              Prevalence : 0.6747
##          Detection Rate : 0.6080
##    Detection Prevalence : 0.7413
##       Balanced Accuracy : 0.7457
##
##        'Positive' Class : 0
##
```

```r
#Accuracy for testing data
test_no_insulin <-na.omit(test_no_insulin)
predicted_probs <- predict(model_out_no_insulin, newdata=test_no_insulin, type="response")
predicted_class <- ifelse(predicted_probs > 0.5, 1, 0)
confusionMatrix(as.factor(predicted_class), as.factor(test_no_insulin$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 87 26
##          1 15 29
##
##                Accuracy : 0.7389
##                  95% CI : (0.6628, 0.8056)
##     No Information Rate : 0.6497
##     P-Value [Acc > NIR] : 0.01072
##
##                   Kappa : 0.3986
##
##  Mcnemar's Test P-Value : 0.11835
##
##             Sensitivity : 0.8529
##             Specificity : 0.5273
##          Pos Pred Value : 0.7699
##          Neg Pred Value : 0.6591
##              Prevalence : 0.6497
##          Detection Rate : 0.5541
##    Detection Prevalence : 0.7197
##       Balanced Accuracy : 0.6901
##
##        'Positive' Class : 0
##
```

The accuracy of the model for the training dataset is 80% and for the testing dataset is 73.8%, in both cases it is observed that the quantity for false negatives is larger than false positives, i.e. several people with diabetes were mistakenly classified as non-diabetics by the model.

Model With Insulin

```r
train_noNas <-na.omit(train)
predicted_probs_complete <- predict(model_out, type="response")
predicted_class_complete <- ifelse(predicted_probs_complete > 0.5, 1, 0)
confusionMatrix(as.factor(predicted_class_complete), as.factor(train_noNas$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 163  36
```

```
##          1  19  55
##
##                Accuracy : 0.7985
##                  95% CI : (0.746, 0.8445)
##     No Information Rate : 0.6667
##     P-Value [Acc > NIR] : 1.029e-06
##
##                   Kappa : 0.5245
##
##  Mcnemar's Test P-Value : 0.03097
##
##             Sensitivity : 0.8956
##             Specificity : 0.6044
##          Pos Pred Value : 0.8191
##          Neg Pred Value : 0.7432
##              Prevalence : 0.6667
##          Detection Rate : 0.5971
##    Detection Prevalence : 0.7289
##       Balanced Accuracy : 0.7500
##
##        'Positive' Class : 0
##
```

```r
#Accuracy for testing data
test_noNas <-na.omit(test)
predicted_probs_complete_test <- predict(model_out, newdata=test_noNas, type="response")
predicted_class_complete_test <- ifelse(predicted_probs_complete_test > 0.5, 1, 0)
confusionMatrix(as.factor(predicted_class_complete_test), as.factor(test_noNas$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 70 19
##          1 10 20
##
##                Accuracy : 0.7563
##                  95% CI : (0.6691, 0.8303)
##     No Information Rate : 0.6723
##     P-Value [Acc > NIR] : 0.02949
##
##                   Kappa : 0.4122
##
##  Mcnemar's Test P-Value : 0.13739
##
##             Sensitivity : 0.8750
##             Specificity : 0.5128
##          Pos Pred Value : 0.7865
##          Neg Pred Value : 0.6667
##              Prevalence : 0.6723
##          Detection Rate : 0.5882
##    Detection Prevalence : 0.7479
##       Balanced Accuracy : 0.6939
##
```

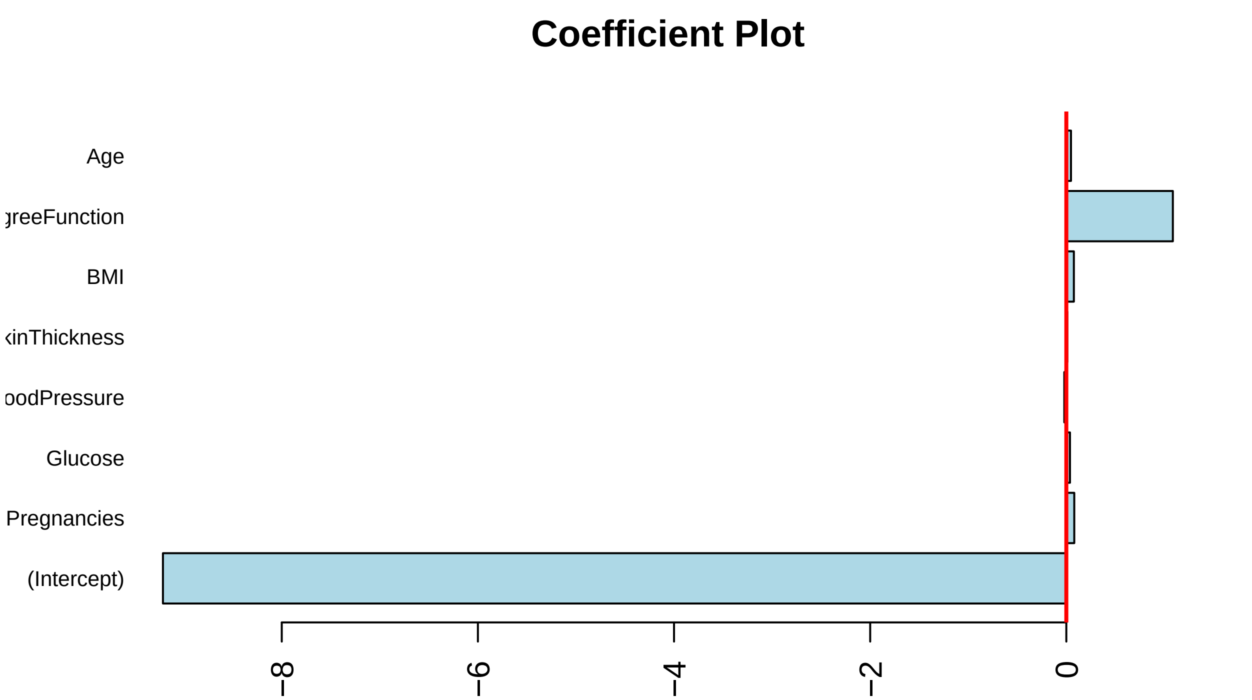
```
##        'Positive' Class : 0
##
```

The models have similar results in accuracy, and because the model without insulin has more data, I decided to pick the model without insulin.

Analyzing coefficients

The significant variables for the model are Glucose, BMI and Age.

```
##The significant variables for the model are Pregnancies, glucose, BMI and Diabetes Pedigree Function

coef_data <- coef(summary(model_out_no_insulin))
barplot(coef_data[, 1], beside=TRUE, las=2, main="Coefficient Plot", horiz=TRUE, col="lightblue", border
abline(v=0, col="red", lwd=2)
```

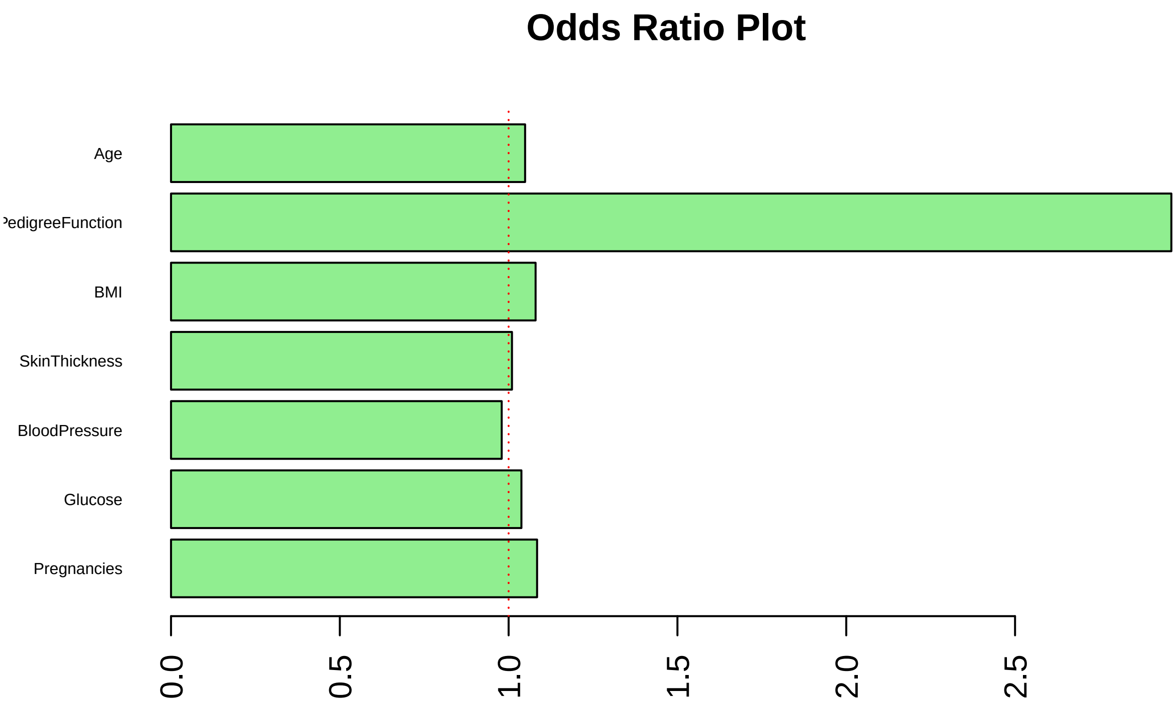


It is observed that the coefficient of the variable is larger than the other ones, but it's p-value is 0.0211, which is not as significant as other variables, this can be attributed to the fact that the data is not scaled.

Analyzing odds ratio

Odds ratio: 1 no effect on response, >1 predictor produces higher odds of the outcome, <1 odds ratio is associated with lower odds of the outcome occurring,

```
odds_ratios <- exp(coef(model_out_no_insulin))
# no intercept from the odds_ratios
odds_ratios_noint= odds_ratios[-1]
barplot(odds_ratios_noint, beside=TRUE, las=2, main="Odds Ratio Plot", horiz=TRUE, col="lightgreen", bo
abline(v = 1, col = "red", lty = 3)
```



the analysis of the observed odd ratios are not reliable since the data is not scaled.

7. Reduced Model:

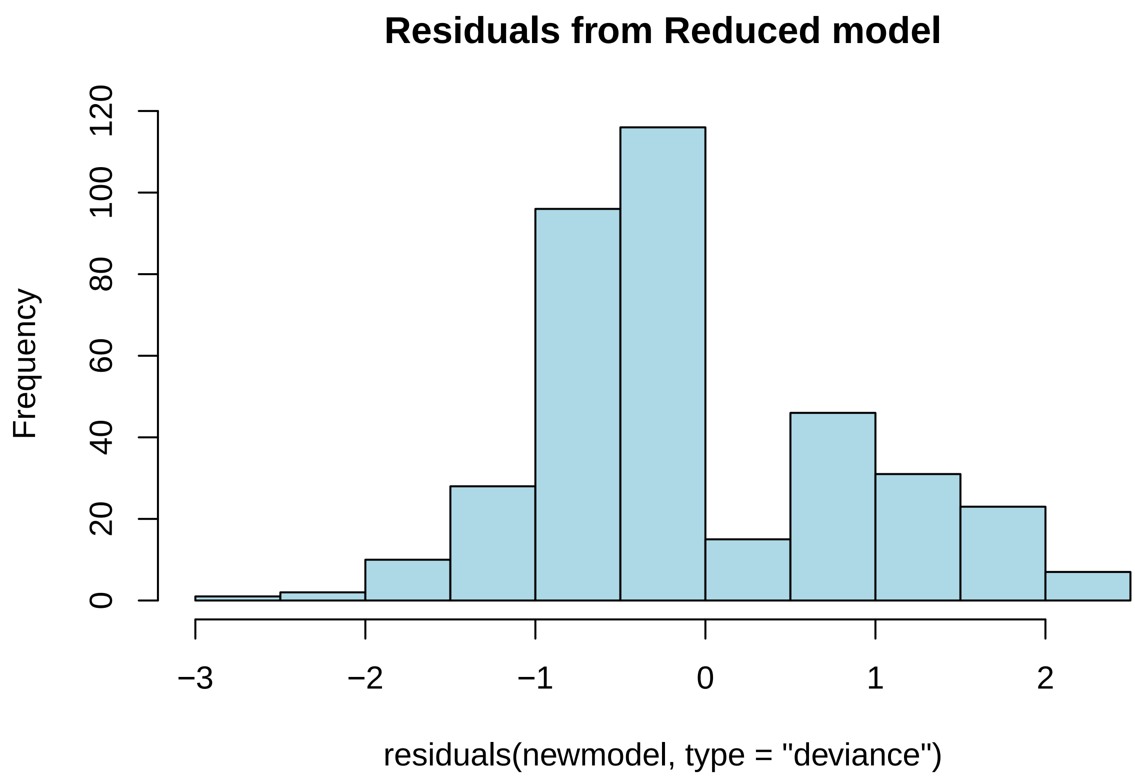
I decided to select the variables: Glucose, BMI and Age.Since those showed to be the most significant variables of the model.

```
newmodel <- glm(Outcome ~ Glucose+BMI+Age, family=binomial(link='logit'),
                data=train_no_insulin)
summary(newmodel)
```

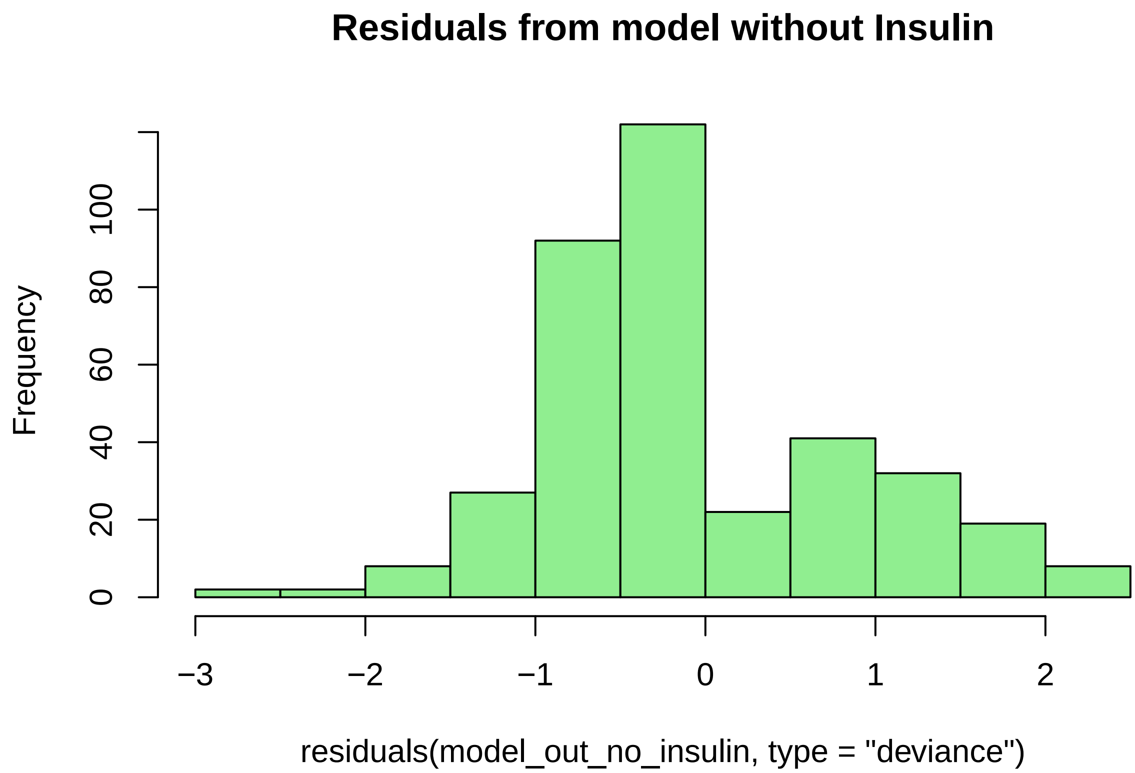
```
##
## Call:
## glm(formula = Outcome ~ Glucose + BMI + Age, family = binomial(link = "logit"),
##     data = train_no_insulin)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -9.656787   1.067566  -9.046  < 2e-16 ***
## Glucose      0.036312   0.004858   7.475 7.72e-14 ***
## BMI          0.073727   0.021904   3.366 0.000763 ***
## Age          0.056961   0.012834   4.438 9.06e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 473.12  on 374  degrees of freedom
## Residual deviance: 328.60  on 371  degrees of freedom
## AIC: 336.6
##
## Number of Fisher Scoring iterations: 5
```

```
hist(residuals(newmodel, type="deviance"), main="Residuals from Reduced model", col="lightblue")
```




```
hist(residuals(model_out_no_insulin, type="deviance"), main="Residuals from model without Insulin", col=
```



Predictions for the Reduced Model

```
##Acuracy for training data
pred_probs <- predict(newmodel, type="response")
pred_class <- ifelse(pred_probs > 0.5, 1, 0)
confusionMatrix(as.factor(pred_class), as.factor(train_no_insulin$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 227  47
##          1  26  75
##
##                Accuracy : 0.8053
##                  95% CI : (0.7616, 0.8442)
##     No Information Rate : 0.6747
##     P-Value [Acc > NIR] : 1.179e-08
##
##                   Kappa : 0.5359
##
##  Mcnemar's Test P-Value : 0.01924
##
##             Sensitivity : 0.8972
##             Specificity : 0.6148
##          Pos Pred Value : 0.8285
```

```
##          Neg Pred Value : 0.7426
##              Prevalence : 0.6747
##          Detection Rate : 0.6053
##    Detection Prevalence : 0.7307
##       Balanced Accuracy : 0.7560
##
##        'Positive' Class : 0
##
```

```
#Accuracy for testing data
pred_probs_test <- predict(newmodel, newdata=test_no_insulin, type="response")
pred_class_test <- ifelse(pred_probs_test > 0.5, 1, 0)
confusionMatrix(as.factor(pred_class_test), as.factor(test_no_insulin$Outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 87 29
##          1 15 26
##
##                Accuracy : 0.7197
##                  95% CI : (0.6426, 0.7884)
##     No Information Rate : 0.6497
##     P-Value [Acc > NIR] : 0.03779
##
##                   Kappa : 0.346
##
##  Mcnemar's Test P-Value : 0.05002
##
##             Sensitivity : 0.8529
##             Specificity : 0.4727
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.6341
##              Prevalence : 0.6497
##          Detection Rate : 0.5541
##    Detection Prevalence : 0.7389
##       Balanced Accuracy : 0.6628
##
##        'Positive' Class : 0
##
```

Conclusions: The accuracy and the residuals for the reduced model and the model without insulin are similar, the AIC for the reduced model increased 4 points. Even so, the reduced model is less complex and has similar results.

AUC-ROC

```
#install.packages("pROC")
#install.packages("randomForest")
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(randomForest)
```

```
## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

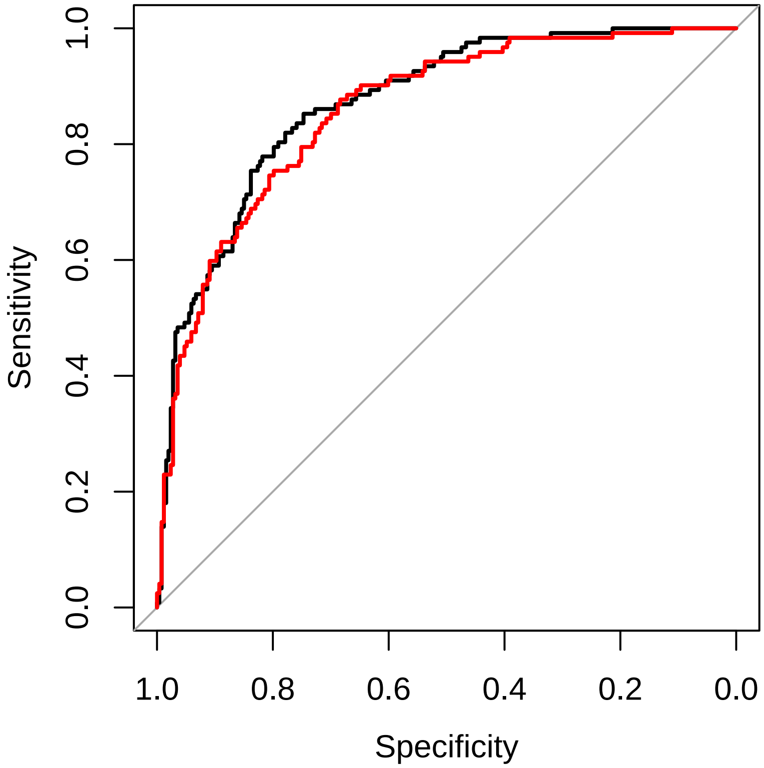
```r
set.seed(420)

#ROC for Model without insulin
Outcome_train_noinsulin=train_no_insulin[,8]
predictions1=predict(model_out_no_insulin)
roc1=roc(Outcome_train_noinsulin ~ predictions1)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```r
##ROC for Reduced Model
predictions2=predict(newmodel)
roc2=roc(Outcome_train_noinsulin ~ predictions2)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```r
##Graph
par(pty="s")
plot(roc1)
plot(roc2, add=TRUE, col='red')
```




```r
names(roc2)
```

```
##  [1] "percent"            "sensitivities"      "specificities"
##  [4] "thresholds"         "direction"          "cases"
##  [7] "controls"           "fun.sesp"           "auc"
## [10] "call"               "original.predictor" "original.response"
## [13] "predictor"          "response"           "levels"
## [16] "predictor.name"     "response.name"
```

```r
##Area under the curve
AUC_noinsulin=roc1$auc
AUC_reducedmodel=roc2$auc
print(paste("The AUC for the model without Insulin is", AUC_noinsulin))
```

```
## [1] "The AUC for the model without Insulin is 0.871444307652433"
```

```r
print(paste("The AUC for the reduced model is",AUC_reducedmodel))
```

```
## [1] "The AUC for the reduced model is 0.857966694745027"
```

It can be seen that the curve of the reduced model in red does not deviate too much from the curve of the original model. Moreover, the AUC values for both models are similar. In conclussion the reduced model give me a better trade-off between accuracy and complexity.

8. Decision Tree:

```r
#install.packages("rpart")
#install.packages("rpart.plot")
library(rpart)
library(rpart.plot)

outcome_tree = rpart(Outcome ~ .,data=train_no_insulin)
summary(outcome_tree)
```

```
## Call:
## rpart(formula = Outcome ~ ., data = train_no_insulin)
##   n= 375
##
##           CP nsplit rel error    xerror       xstd
## 1 0.35245902      0 1.0000000 1.0000000 0.07436434
## 2 0.03688525      1 0.6475410 0.6803279 0.06589539
## 3 0.03278689      3 0.5737705 0.6967213 0.06645582
## 4 0.01639344      5 0.5081967 0.6721311 0.06560928
## 5 0.01000000      7 0.4754098 0.7131148 0.06700086
##
## Variable importance
##                  Glucose                      Age DiabetesPedigreeFunction
##                       52                       19                        9
##                      BMI            SkinThickness              Pregnancies
##                        8                        5                        4
##            BloodPressure
##                        3
##
## Node number 1: 375 observations,    complexity param=0.352459
##   predicted class=0  expected loss=0.3253333  P(node) =1
##     class counts:   253   122
##    probabilities: 0.675 0.325
##   left son=2 (282 obs) right son=3 (93 obs)
##   Primary splits:
##       Glucose                  < 143.5  to the left,  improve=40.740380, (0 missing)
##       Age                      < 42.5   to the left,  improve=19.323490, (0 missing)
##       Pregnancies              < 6.5    to the left,  improve=12.414070, (0 missing)
##       BMI                      < 29.65  to the left,  improve=11.023950, (0 missing)
##       DiabetesPedigreeFunction < 0.3185 to the left,  improve= 8.354083, (0 missing)
##   Surrogate splits:
##       DiabetesPedigreeFunction < 1.149  to the left,  agree=0.765, adj=0.054, (0 split)
##       Age                      < 50.5   to the left,  agree=0.763, adj=0.043, (0 split)
##       SkinThickness            < 55     to the left,  agree=0.757, adj=0.022, (0 split)
##       BMI                      < 48.05  to the left,  agree=0.757, adj=0.022, (0 split)
##       Pregnancies              < 12.5   to the left,  agree=0.755, adj=0.011, (0 split)
##
## Node number 2: 282 observations,    complexity param=0.03688525
##   predicted class=0  expected loss=0.1914894  P(node) =0.752
##     class counts:   228    54
##    probabilities: 0.809 0.191
##   left son=4 (246 obs) right son=5 (36 obs)
```

```
##   Primary splits:
##       Age           < 42.5   to the left,  improve=9.334054, (0 missing)
##       Glucose       < 123.5  to the left,  improve=4.909785, (0 missing)
##       Pregnancies   < 5.5    to the left,  improve=4.706317, (0 missing)
##       BMI           < 45.05  to the left,  improve=4.198270, (0 missing)
##       SkinThickness < 23.5   to the left,  improve=4.085162, (0 missing)
##   Surrogate splits:
##       Pregnancies   < 9.5    to the left,  agree=0.897, adj=0.194, (0 split)
##       BloodPressure < 101    to the left,  agree=0.879, adj=0.056, (0 split)
##       SkinThickness < 7.5    to the right, agree=0.879, adj=0.056, (0 split)
##       Glucose       < 141.5  to the left,  agree=0.876, adj=0.028, (0 split)
##
## Node number 3: 93 observations,    complexity param=0.03278689
##   predicted class=1  expected loss=0.2688172  P(node) =0.248
##     class counts:    25    68
##    probabilities: 0.269 0.731
##   left son=6 (27 obs) right son=7 (66 obs)
##   Primary splits:
##       DiabetesPedigreeFunction < 0.332  to the left,  improve=4.744325, (0 missing)
##       Glucose                  < 157.5  to the left,  improve=4.466476, (0 missing)
##       Age                      < 40.5   to the left,  improve=2.449469, (0 missing)
##       SkinThickness            < 43     to the left,  improve=1.803042, (0 missing)
##       Pregnancies              < 6.5    to the left,  improve=1.443838, (0 missing)
##   Surrogate splits:
##       SkinThickness < 18.5   to the left,  agree=0.731, adj=0.074, (0 split)
##       BMI           < 25.7   to the left,  agree=0.731, adj=0.074, (0 split)
##
## Node number 4: 246 observations,    complexity param=0.01639344
##   predicted class=0  expected loss=0.1422764  P(node) =0.656
##     class counts:   211    35
##    probabilities: 0.858 0.142
##   left son=8 (238 obs) right son=9 (8 obs)
##   Primary splits:
##       BMI                      < 45.05  to the left,  improve=3.853676, (0 missing)
##       Glucose                  < 101.5  to the left,  improve=3.506432, (0 missing)
##       Age                      < 24.5   to the left,  improve=3.113500, (0 missing)
##       SkinThickness            < 28.5   to the left,  improve=2.783393, (0 missing)
##       DiabetesPedigreeFunction < 0.2445 to the left,  improve=1.351403, (0 missing)
##
## Node number 5: 36 observations,    complexity param=0.03688525
##   predicted class=1  expected loss=0.4722222  P(node) =0.096
##     class counts:    17    19
##    probabilities: 0.472 0.528
##   left son=10 (7 obs) right son=11 (29 obs)
##   Primary splits:
##       Age                      < 59     to the right, improve=4.840996, (0 missing)
##       DiabetesPedigreeFunction < 0.628  to the left,  improve=3.836752, (0 missing)
##       SkinThickness            < 23.5   to the left,  improve=1.587302, (0 missing)
##       Glucose                  < 107.5  to the left,  improve=1.388889, (0 missing)
##       BMI                      < 28.1   to the left,  improve=1.018336, (0 missing)
##   Surrogate splits:
##       SkinThickness < 17.5   to the left,  agree=0.861, adj=0.286, (0 split)
##
## Node number 6: 27 observations,    complexity param=0.03278689
```

```
##   predicted class=0  expected loss=0.4814815  P(node) =0.072
##     class counts:    14    13
##    probabilities: 0.519 0.481
##   left son=12 (14 obs) right son=13 (13 obs)
##   Primary splits:
##       Glucose                  < 159.5  to the left,  improve=4.1518110, (0 missing)
##       DiabetesPedigreeFunction < 0.269  to the right, improve=2.1671960, (0 missing)
##       BMI                      < 34.55  to the left,  improve=1.5167760, (0 missing)
##       Age                      < 40.5   to the left,  improve=1.0243390, (0 missing)
##       SkinThickness            < 32     to the left,  improve=0.9259259, (0 missing)
##   Surrogate splits:
##       Pregnancies   < 6.5    to the left,  agree=0.667, adj=0.308, (0 split)
##       BloodPressure < 67     to the left,  agree=0.667, adj=0.308, (0 split)
##       SkinThickness < 30.5   to the left,  agree=0.667, adj=0.308, (0 split)
##       BMI           < 33.95  to the left,  agree=0.667, adj=0.308, (0 split)
##       Age           < 30.5   to the left,  agree=0.667, adj=0.308, (0 split)
##
## Node number 7: 66 observations
##   predicted class=1  expected loss=0.1666667  P(node) =0.176
##     class counts:    11    55
##    probabilities: 0.167 0.833
##
## Node number 8: 238 observations
##   predicted class=0  expected loss=0.1260504  P(node) =0.6346667
##     class counts:   208    30
##    probabilities: 0.874 0.126
##
## Node number 9: 8 observations
##   predicted class=1  expected loss=0.375  P(node) =0.02133333
##     class counts:     3     5
##    probabilities: 0.375 0.625
##
## Node number 10: 7 observations
##   predicted class=0  expected loss=0  P(node) =0.01866667
##     class counts:     7     0
##    probabilities: 1.000 0.000
##
## Node number 11: 29 observations,    complexity param=0.01639344
##   predicted class=1  expected loss=0.3448276  P(node) =0.07733333
##     class counts:    10    19
##    probabilities: 0.345 0.655
##   left son=22 (16 obs) right son=23 (13 obs)
##   Primary splits:
##       Glucose                  < 107.5  to the left,  improve=3.3822940, (0 missing)
##       DiabetesPedigreeFunction < 0.4345 to the left,  improve=3.3822940, (0 missing)
##       BloodPressure            < 75     to the left,  improve=0.9858012, (0 missing)
##       BMI                      < 33.45  to the right, improve=0.9418321, (0 missing)
##       Age                      < 44.5   to the right, improve=0.7527989, (0 missing)
##   Surrogate splits:
##       BloodPressure            < 77     to the left,  agree=0.690, adj=0.308, (0 split)
##       DiabetesPedigreeFunction < 1.0265 to the left,  agree=0.690, adj=0.308, (0 split)
##       BMI                      < 35.75  to the left,  agree=0.655, adj=0.231, (0 split)
##       Age                      < 48.5   to the left,  agree=0.655, adj=0.231, (0 split)
##       Pregnancies              < 8.5    to the right, agree=0.621, adj=0.154, (0 split)
```

```
##
## Node number 12: 14 observations
##   predicted class=0  expected loss=0.2142857  P(node) =0.03733333
##     class counts:    11     3
##    probabilities: 0.786 0.214
##
## Node number 13: 13 observations
##   predicted class=1  expected loss=0.2307692  P(node) =0.03466667
##     class counts:     3    10
##    probabilities: 0.231 0.769
##
## Node number 22: 16 observations
##   predicted class=0  expected loss=0.4375  P(node) =0.04266667
##     class counts:     9     7
##    probabilities: 0.562 0.437
##
## Node number 23: 13 observations
##   predicted class=1  expected loss=0.07692308  P(node) =0.03466667
##     class counts:     1    12
##    probabilities: 0.077 0.923
```

```r
##Predictions training and testing

predictions_Outcome_trainig = predict(outcome_tree, newdata=train_no_insulin, type='class')
head( predictions_Outcome_trainig )
```

```
##  2  4  7 14 15 17
##  0  0  0  1  1  1
## Levels: 0 1
```

```r
predictions_Outcome_testing = predict(outcome_tree, newdata=test_no_insulin, type='class')

##Accuracy trainig and testing

accuracytraining <- mean(predictions_Outcome_trainig == train_no_insulin$Outcome)
accuracytesting <- mean(predictions_Outcome_testing == test_no_insulin$Outcome)
accuracytraining
```

```
## [1] 0.8453333
```

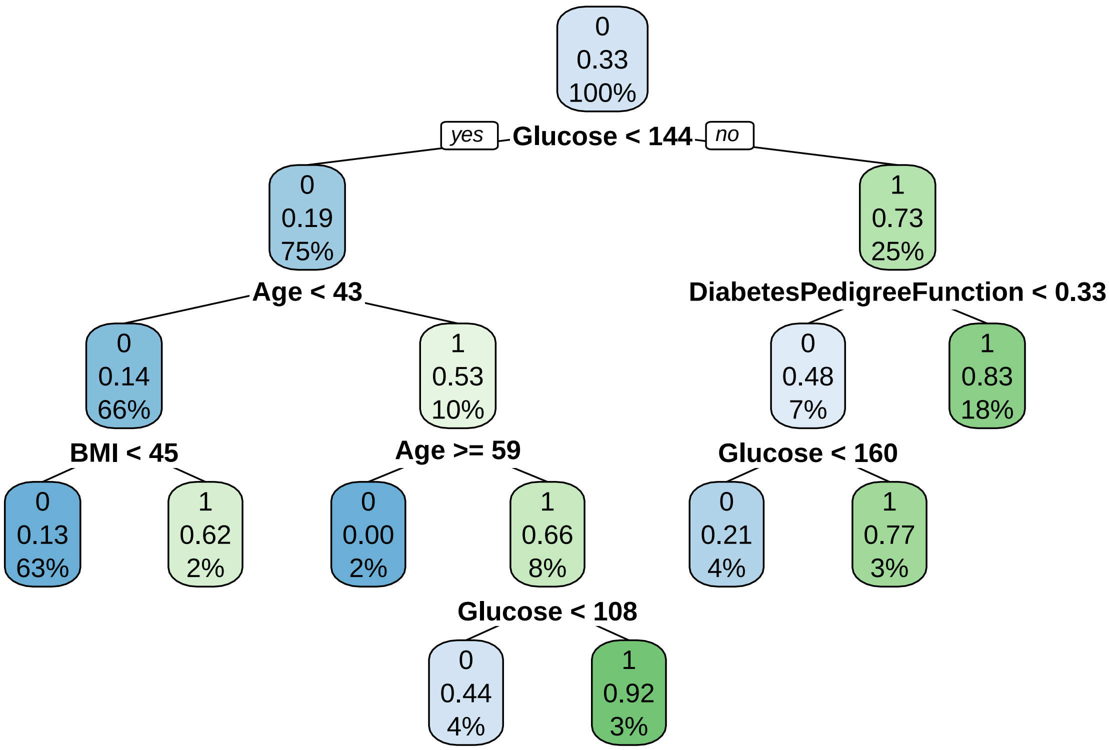
```r
accuracytesting
```

```
## [1] 0.7261146
```

```r
## Confusion Matrix for training

confusionMatrix(predictions_Outcome_trainig, train_no_insulin$Outcome)$table
```

```
##           Reference
## Prediction   0   1
##          0 235  40
##          1  18  82
```

```r
##Draw the tree model

#library(rattle)
#fancyRpartPlot(outcome_tree)

rpart.plot(outcome_tree, fallen.leaves = FALSE)
```




```r
##Variable Importance
outcome_tree$variable.importance
```

```
##                  Glucose                      Age DiabetesPedigreeFunction
##                48.533763                17.985334                 7.975374
##                      BMI            SkinThickness              Pregnancies
##                 7.139254                 4.406749                 4.050857
##            BloodPressure
##                 2.836745
```

The Training and testing accuracy are 84.5% and 72.6% respectively. In the graph it is observed that 33% of the observations in the training dataset are diabetic and the first decision node is glucose, if glucose is > 144 the probability of being diabetic is 73%.

9.

The decision tree is as accurate as the logistic regression model in this case, because of it's simplicity it is very useful to understand how to categorize each patient and the probabilities of a patient having diabetes depending on his values for Glucose, Age, Diabetes Pedigree Function and BMI.

In the decision tree, the variable Diabetes Pedigree Function is more important that the variable BMI, in comparison with the logistic regression model where the removal of Diabetes Pedigree Function didn't affect the outcome.

Since decisions can be visualized in such an intuitive way, it is a pragmatic method for diagnosing the patient.

10.

Although both models are similar in accuracy, with the decision tree performing 5% better on training data. Because the simplicity of the Decision Tree, the Decision Tree model gives a better trade off between accuracy and complexity.

11. Scaling the data:

```
#Data scaled without insulin
numeric_cols <- datasetdiabetes_no_insulin[sapply(datasetdiabetes_no_insulin, is.numeric)]

# Scale the numeric columns
scaled_data_noinsulin <- as.data.frame(scale(numeric_cols))
scaled_data_noinsulin$Outcome <- datasetdiabetes_no_insulin[["Outcome"]]
head(scaled_data_noinsulin)
```

```
##   Pregnancies    Glucose BloodPressure SkinThickness        BMI
## 1   0.6395305  0.8617221   -0.03272323    0.55804049  0.1649875
## 2  -0.8443348 -1.2014407   -0.51729142   -0.01464349 -0.8458446
## 3   1.2330766  2.0079237   -0.67881415            NA -1.3223797
## 4  -0.8443348 -1.0704463   -0.51729142   -0.58732747 -0.6292377
## 5  -1.1411079  0.5014873   -2.61708691    0.55804049  1.5368309
## 6   0.3427574 -0.1862336    0.12879950            NA -0.9902491
##   DiabetesPedigreeFunction         Age Outcome
## 1                0.4681869  1.42506672       1
## 2               -0.3648230 -0.19054773       0
## 3                0.6040037 -0.10551539       1
## 4               -0.9201630 -1.04087112       0
## 5                5.4813370 -0.02048305       1
## 6               -0.8175458 -0.27558007       0
```

```
names(scaled_data_noinsulin)
```

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

```r
#Is Outcome a factor?
is.factor(scaled_data_noinsulin$Outcome)
```

```
## [1] TRUE
```

```r
#training and testing scaled

set.seed(21)
n2=nrow(scaled_data_noinsulin)
sample2 <- sample(c(TRUE, FALSE), n2, replace=TRUE, prob=c(0.7,0.3))
training_scaled <-scaled_data_noinsulin[sample2,]
testing_scaled <-scaled_data_noinsulin[!sample2,]
```

12. Using Scaled Data

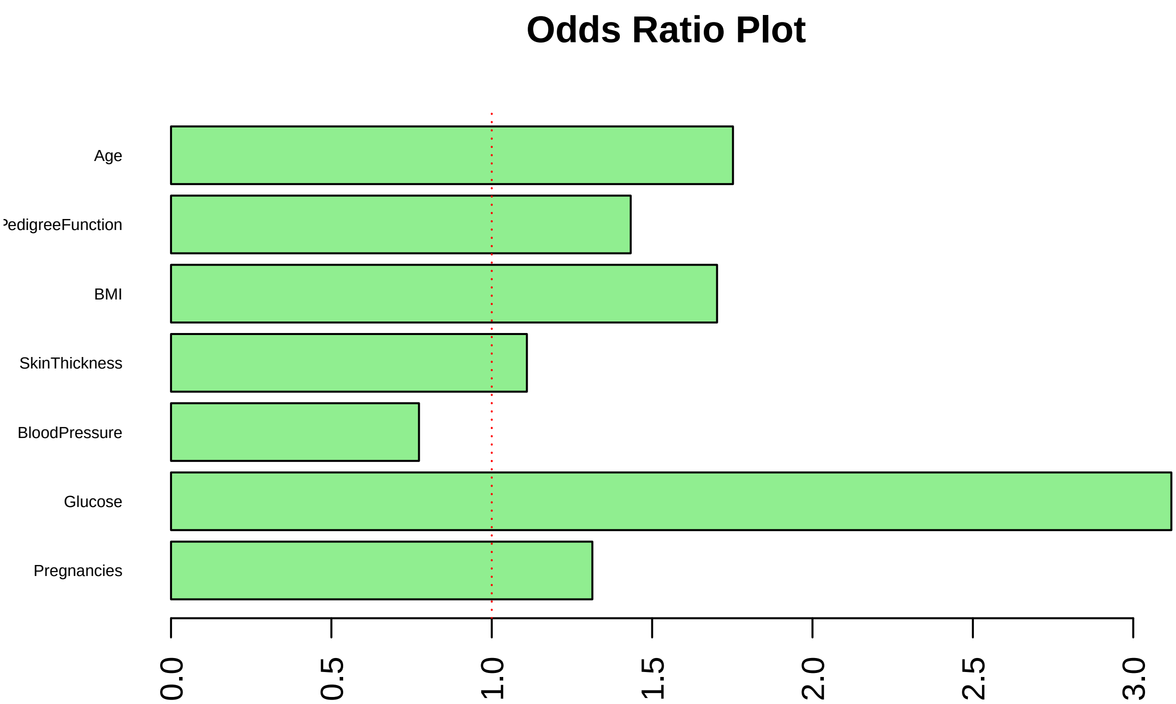
Logistic Regression Model

```r
#Logistic Regression Model:
Complete_model_scaled <- glm(Outcome ~ ., data = training_scaled, family = binomial() )
summary(Complete_model_scaled)
```

```
##
## Call:
## glm(formula = Outcome ~ ., family = binomial(), data = training_scaled)
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -0.9911     0.1507  -6.576 4.84e-11 ***
## Pregnancies                0.2728     0.1943   1.404   0.1602
## Glucose                    1.1375     0.1557   7.306 2.75e-13 ***
## BloodPressure             -0.2573     0.1502  -1.713   0.0867 .
## SkinThickness              0.1040     0.1898   0.548   0.5837
## BMI                        0.5320     0.2043   2.604   0.0092 **
## DiabetesPedigreeFunction   0.3599     0.1560   2.307   0.0211 *
## Age                        0.5608     0.2061   2.722   0.0065 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 473.12  on 374  degrees of freedom
## Residual deviance: 317.01  on 367  degrees of freedom
##   (150 observations deleted due to missingness)
## AIC: 333.01
##
## Number of Fisher Scoring iterations: 5
```

```r
# Analyzing Odds Ratios
odds_ratios <- exp(coef(Complete_model_scaled))
# no intercept from the odds_ratios
```

```
odds_ratios_noint= odds_ratios[-1]
barplot(odds_ratios_noint, beside=TRUE, las=2, main="Odds Ratio Plot", horiz=TRUE, col="lightgreen", bo
abline(v = 1, col = "red", lty = 3)
```




```
#Accuracy for testing data
predicted_probs_s <- predict(Complete_model_scaled, newdata=testing_scaled, type="response")
predicted_class_s <- ifelse(predicted_probs_s > 0.5, 1, 0)
confusionMatrix(as.factor(predicted_class_s), as.factor(testing_scaled$Outcome))$table
```

```
##           Reference
## Prediction  0  1
##          0 87 26
##          1 15 29
```

```
confusionMatrix(as.factor(predicted_class_s), as.factor(testing_scaled$Outcome))$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##     0.73885350     0.39857984     0.66283844     0.80562264     0.64968153
## AccuracyPValue  McnemarPValue
##     0.01071629     0.11834981
```

There isn't an observed change in the levels of significance of the p-values, the variables: Glucose, BMI and Age are the most statistically significant. The coefficients changed and now the odd ratio table is accurate, selecting Glucose as the variable that produces higher odds instead of Diabetes Pedigree Function.

The AIC, accuracy and confusion matrix remain the same.

Reduced Regression Model

```r
#Reduced Regression Model:
reduced_model_scaled <- glm(Outcome ~ Glucose+BMI+Age, family=binomial(link='logit'),
             data=training_scaled)
summary(reduced_model_scaled)
```

```
##
## Call:
## glm(formula = Outcome ~ Glucose + BMI + Age, family = binomial(link = "logit"),
##     data = training_scaled)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.9041     0.1178  -7.674 1.66e-14 ***
## Glucose       1.0895     0.1263   8.627  < 2e-16 ***
## BMI           0.5913     0.1234   4.792 1.65e-06 ***
## Age           0.3521     0.1114   3.161  0.00157 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 660.44  on 516  degrees of freedom
## Residual deviance: 488.03  on 513  degrees of freedom
##   (8 observations deleted due to missingness)
## AIC: 496.03
##
## Number of Fisher Scoring iterations: 5
```

```r
#Accuracy for testing data
predicted_probs_r <- predict(reduced_model_scaled, newdata=testing_scaled, type="response")
predicted_class_r <- ifelse(predicted_probs_r > 0.5, 1, 0)
confusionMatrix(as.factor(predicted_class_r), as.factor(testing_scaled$Outcome))$table
```

```
##           Reference
## Prediction   0   1
##          0 128  42
##          1  17  48
```

```r
confusionMatrix(as.factor(predicted_class_r), as.factor(testing_scaled$Outcome))$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   0.7489361702   0.4392315470   0.6884300140   0.8030452910   0.6170212766
## AccuracyPValue  McnemarPValue
##   0.0000129118   0.0017808702
```

The AIC for this model keeps being larger than the complete model, but it is accurate and less complex.

Decision Tree

```r
#Decision Tree:
outcome_tree_scaled = rpart(Outcome ~ .,data=training_scaled)
summary(outcome_tree_scaled)
```

```
## Call:
## rpart(formula = Outcome ~ ., data = training_scaled)
##   n= 525
##
##           CP nsplit rel error    xerror       xstd
## 1 0.29142857      0 1.0000000 1.0000000 0.06172134
## 2 0.03142857      1 0.7085714 0.8114286 0.05816020
## 3 0.02285714      3 0.6457143 0.7828571 0.05749876
## 4 0.01714286      8 0.5314286 0.7371429 0.05636698
## 5 0.01142857     10 0.4971429 0.7428571 0.05651354
## 6 0.01000000     14 0.4514286 0.7714286 0.05722443
##
## Variable importance
##                  Glucose                      Age DiabetesPedigreeFunction
##                       47                       18                       13
##                      BMI              Pregnancies            BloodPressure
##                        9                        9                        4
##
## Node number 1: 525 observations,    complexity param=0.2914286
##   predicted class=0  expected loss=0.3333333  P(node) =1
##     class counts:   350   175
##    probabilities: 0.667 0.333
##   left son=2 (388 obs) right son=3 (137 obs)
##   Primary splits:
##       Glucose       < 0.5833589   to the left,  improve=46.239450, (3 missing)
##       Age           < -0.2330639  to the left,  improve=21.859380, (0 missing)
##       BMI           < -0.3765297  to the left,  improve=18.000060, (5 missing)
##       Pregnancies   < 0.787917    to the left,  improve=13.811540, (0 missing)
##       SkinThickness < -0.5396038  to the left,  improve= 8.648043, (145 missing)
##   Surrogate splits:
##       DiabetesPedigreeFunction < 2.043662    to the left,  agree=0.749, adj=0.044, (3 split)
##       Age                      < 2.062809    to the left,  agree=0.749, adj=0.044, (0 split)
##
## Node number 2: 388 observations,    complexity param=0.02285714
##   predicted class=0  expected loss=0.2087629  P(node) =0.7390476
##     class counts:   307    81
##    probabilities: 0.791 0.209
##   left son=4 (241 obs) right son=5 (147 obs)
##   Primary splits:
##       Age                      < -0.2330639  to the left,  improve=9.948781, (0 missing)
##       Glucose                  < -0.6610886  to the left,  improve=9.268308, (3 missing)
##       BMI                      < -0.8819457  to the left,  improve=8.141677, (5 missing)
##       Pregnancies              < 0.1943709   to the left,  improve=4.883465, (0 missing)
##       DiabetesPedigreeFunction < 0.5647677   to the left,  improve=4.608357, (0 missing)
##   Surrogate splits:
##       Pregnancies              < 0.1943709   to the left,  agree=0.809, adj=0.497, (0 split)
##       DiabetesPedigreeFunction < 1.357033    to the left,  agree=0.644, adj=0.061, (0 split)
##       BloodPressure            < -0.1134846  to the left,  agree=0.637, adj=0.041, (0 split)
##       Glucose                  < 0.02663239  to the left,  agree=0.629, adj=0.020, (0 split)
```

```
##
## Node number 3: 137 observations,    complexity param=0.03142857
##   predicted class=1  expected loss=0.3138686  P(node) =0.2609524
##     class counts:    43    94
##    probabilities: 0.314 0.686
##   left son=6 (67 obs) right son=7 (70 obs)
##   Primary splits:
##       Glucose                  < 1.238331    to the left,  improve=7.031606, (0 missing)
##       DiabetesPedigreeFunction < -0.4734765  to the left,  improve=5.119659, (0 missing)
##       BMI                      < -0.3765297  to the left,  improve=4.441118, (0 missing)
##       SkinThickness            < 1.273895    to the left,  improve=2.568562, (33 missing)
##       Age                      < 2.487971    to the right, improve=2.397924, (0 missing)
##   Surrogate splits:
##       DiabetesPedigreeFunction < -0.6470202  to the left,  agree=0.577, adj=0.134, (0 split)
##       BloodPressure            < 0.04803814  to the right, agree=0.562, adj=0.104, (0 split)
##       BMI                      < 0.04224358  to the left,  agree=0.562, adj=0.104, (0 split)
##       Age                      < -0.2330639  to the left,  agree=0.562, adj=0.104, (0 split)
##       Pregnancies              < 1.381463    to the right, agree=0.555, adj=0.090, (0 split)
##
## Node number 4: 241 observations
##   predicted class=0  expected loss=0.120332  P(node) =0.4590476
##     class counts:   212    29
##    probabilities: 0.880 0.120
##
## Node number 5: 147 observations,    complexity param=0.02285714
##   predicted class=0  expected loss=0.3537415  P(node) =0.28
##     class counts:    95    52
##    probabilities: 0.646 0.354
##   left son=10 (29 obs) right son=11 (118 obs)
##   Primary splits:
##       BMI                      < -0.8025232  to the left,  improve=9.041393, (0 missing)
##       Glucose                  < -0.8248316  to the left,  improve=6.849108, (1 missing)
##       Age                      < 1.977777    to the right, improve=4.180581, (0 missing)
##       DiabetesPedigreeFunction < -0.7752917  to the left,  improve=3.574521, (0 missing)
##       SkinThickness            < -1.112288   to the left,  improve=1.389305, (58 missing)
##   Surrogate splits:
##       Age < 2.062809    to the right, agree=0.816, adj=0.069, (0 split)
##
## Node number 6: 67 observations,    complexity param=0.03142857
##   predicted class=1  expected loss=0.4776119  P(node) =0.127619
##     class counts:    32    35
##    probabilities: 0.478 0.522
##   left son=12 (27 obs) right son=13 (40 obs)
##   Primary splits:
##       DiabetesPedigreeFunction < -0.4221679  to the left,  improve=4.623577, (0 missing)
##       Age                      < 0.7022918   to the left,  improve=4.312465, (0 missing)
##       BloodPressure            < 1.057555    to the left,  improve=4.156510, (1 missing)
##       Pregnancies              < 1.08469     to the left,  improve=3.538228, (0 missing)
##       BMI                      < -0.3693095  to the left,  improve=3.360060, (0 missing)
##   Surrogate splits:
##       BMI     < -0.6725591  to the left,  agree=0.687, adj=0.222, (0 split)
##       Glucose < 1.107337    to the right, agree=0.642, adj=0.111, (0 split)
##       Age     < 2.445455    to the right, agree=0.642, adj=0.111, (0 split)
##
```

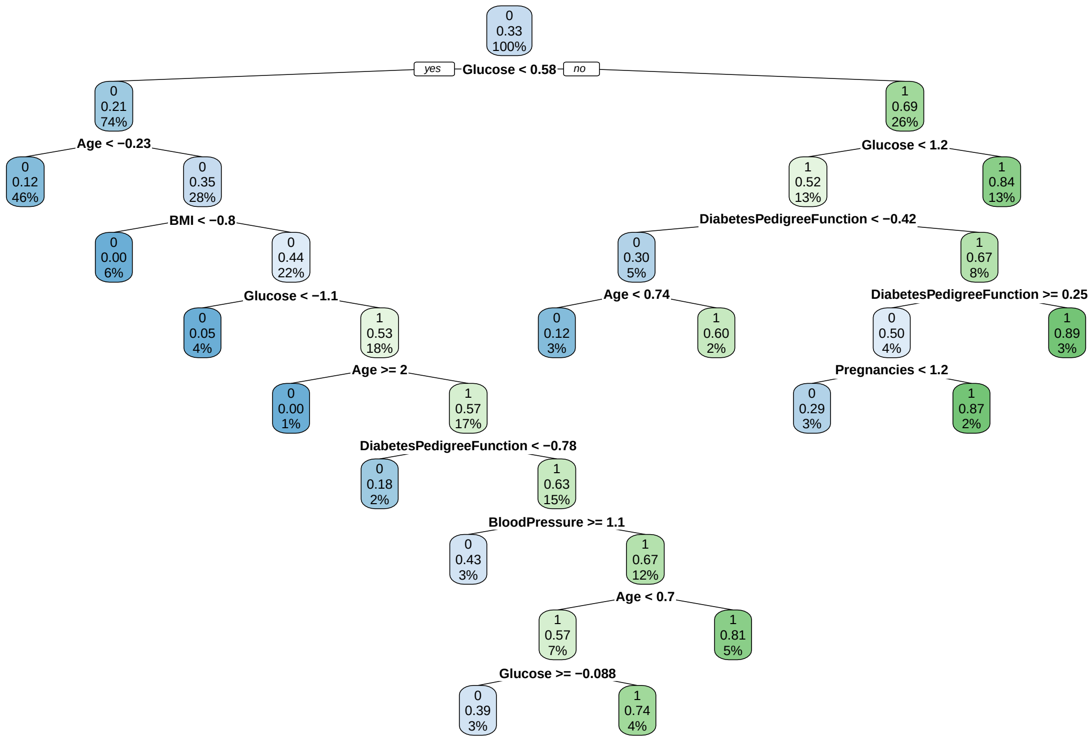
```
## Node number 7: 70 observations
##   predicted class=1  expected loss=0.1571429  P(node) =0.1333333
##     class counts:    11    59
##    probabilities: 0.157 0.843
##
## Node number 10: 29 observations
##   predicted class=0  expected loss=0  P(node) =0.0552381
##     class counts:    29     0
##    probabilities: 1.000 0.000
##
## Node number 11: 118 observations,    complexity param=0.02285714
##   predicted class=0  expected loss=0.440678  P(node) =0.2247619
##     class counts:    66    52
##    probabilities: 0.559 0.441
##   left son=22 (22 obs) right son=23 (96 obs)
##   Primary splits:
##       Glucose                  < -1.054072   to the left,  improve=8.2609500, (1 missing)
##       DiabetesPedigreeFunction < 0.1693898   to the left,  improve=3.5295180, (0 missing)
##       Age                      < 1.977777    to the right, improve=2.8902120, (0 missing)
##       BloodPressure            < 1.057555    to the right, improve=1.7254570, (4 missing)
##       BMI                      < -0.5064938  to the right, improve=0.8659296, (0 missing)
##
## Node number 12: 27 observations,    complexity param=0.01142857
##   predicted class=0  expected loss=0.2962963  P(node) =0.05142857
##     class counts:    19     8
##    probabilities: 0.704 0.296
##   left son=24 (17 obs) right son=25 (10 obs)
##   Primary splits:
##       Age           < 0.744808    to the left,  improve=2.9298470, (0 missing)
##       Pregnancies   < 0.787917    to the left,  improve=2.4566280, (0 missing)
##       BloodPressure < 0.6941291   to the left,  improve=2.4566280, (0 missing)
##       BMI           < 0.403255    to the left,  improve=0.9434698, (0 missing)
##       SkinThickness < -0.1578145  to the right, improve=0.5275120, (8 missing)
##   Surrogate splits:
##       Pregnancies              < 0.1943709   to the left,  agree=0.815, adj=0.5, (0 split)
##       BloodPressure            < 0.3710836   to the left,  agree=0.815, adj=0.5, (0 split)
##       DiabetesPedigreeFunction < -0.7119105  to the right, agree=0.704, adj=0.2, (0 split)
##
## Node number 13: 40 observations,    complexity param=0.01714286
##   predicted class=1  expected loss=0.325  P(node) =0.07619048
##     class counts:    13    27
##    probabilities: 0.325 0.675
##   left son=26 (22 obs) right son=27 (18 obs)
##   Primary splits:
##       DiabetesPedigreeFunction < 0.2463527   to the right, improve=2.9944440, (0 missing)
##       Age                      < 0.9573889   to the left,  improve=2.3705130, (0 missing)
##       Pregnancies              < -0.6959483  to the left,  improve=2.2880950, (0 missing)
##       BMI                      < -0.3259881  to the left,  improve=1.0305190, (0 missing)
##       BloodPressure            < -0.4365301  to the left,  improve=0.9672619, (1 missing)
##   Surrogate splits:
##       Pregnancies   < 0.491144    to the right, agree=0.675, adj=0.278, (0 split)
##       BMI           < 1.370766    to the left,  agree=0.625, adj=0.167, (0 split)
##       Age           < 1.935261    to the left,  agree=0.625, adj=0.167, (0 split)
##       Glucose       < 0.6161075   to the right, agree=0.575, adj=0.056, (0 split)
```

```
##       BloodPressure < 1.34022     to the left,  agree=0.575, adj=0.056, (0 split)
##
## Node number 22: 22 observations
##   predicted class=0  expected loss=0.04545455  P(node) =0.04190476
##     class counts:    21     1
##    probabilities: 0.955 0.045
##
## Node number 23: 96 observations,    complexity param=0.02285714
##   predicted class=1  expected loss=0.46875  P(node) =0.1828571
##     class counts:    45    51
##    probabilities: 0.469 0.531
##   left son=46 (7 obs) right son=47 (89 obs)
##   Primary splits:
##       Age                      < 1.977777    to the right, improve=4.261938, (0 missing)
##       DiabetesPedigreeFunction < -0.7752917  to the left,  improve=3.645833, (0 missing)
##       BloodPressure            < 1.057555    to the right, improve=2.372191, (4 missing)
##       Glucose                  < -0.4973455  to the left,  improve=1.772136, (1 missing)
##       Pregnancies              < -0.1024022  to the right, improve=0.712500, (0 missing)
##
## Node number 24: 17 observations
##   predicted class=0  expected loss=0.1176471  P(node) =0.03238095
##     class counts:    15     2
##    probabilities: 0.882 0.118
##
## Node number 25: 10 observations
##   predicted class=1  expected loss=0.4  P(node) =0.01904762
##     class counts:     4     6
##    probabilities: 0.400 0.600
##
## Node number 26: 22 observations,    complexity param=0.01714286
##   predicted class=0  expected loss=0.5  P(node) =0.04190476
##     class counts:    11    11
##    probabilities: 0.500 0.500
##   left son=52 (14 obs) right son=53 (8 obs)
##   Primary splits:
##       Pregnancies              < 1.233077    to the left,  improve=3.5357140, (0 missing)
##       DiabetesPedigreeFunction < 0.7669839   to the left,  improve=1.5714290, (0 missing)
##       BMI                      < 0.9519924   to the right, improve=0.9428571, (0 missing)
##       Age                      < 0.8298403   to the left,  improve=0.9428571, (0 missing)
##       BloodPressure            < 0.8556518   to the left,  improve=0.5723443, (1 missing)
##   Surrogate splits:
##       DiabetesPedigreeFunction < 1.269506    to the left,  agree=0.773, adj=0.375, (0 split)
##       Age                      < -0.1905477  to the left,  agree=0.773, adj=0.375, (0 split)
##       BloodPressure            < 1.178697    to the left,  agree=0.727, adj=0.250, (0 split)
##       Glucose                  < 0.9435937   to the left,  agree=0.682, adj=0.125, (0 split)
##
## Node number 27: 18 observations
##   predicted class=1  expected loss=0.1111111  P(node) =0.03428571
##     class counts:     2    16
##    probabilities: 0.111 0.889
##
## Node number 46: 7 observations
##   predicted class=0  expected loss=0  P(node) =0.01333333
##     class counts:     7     0
```

```
##    probabilities: 1.000 0.000
##
## Node number 47: 89 observations,    complexity param=0.02285714
##   predicted class=1  expected loss=0.4269663  P(node) =0.1695238
##     class counts:    38    51
##    probabilities: 0.427 0.573
##   left son=94 (11 obs) right son=95 (78 obs)
##   Primary splits:
##       DiabetesPedigreeFunction < -0.7752917  to the left,  improve=3.8419370, (0 missing)
##       BloodPressure            < 1.057555    to the right, improve=1.6125220, (4 missing)
##       Glucose                  < -0.4973455  to the left,  improve=1.2437700, (1 missing)
##       Age                      < 0.7022918   to the left,  improve=1.0599960, (0 missing)
##       BMI                      < -0.5064938  to the right, improve=0.9271384, (0 missing)
##
## Node number 52: 14 observations
##   predicted class=0  expected loss=0.2857143  P(node) =0.02666667
##     class counts:    10     4
##    probabilities: 0.714 0.286
##
## Node number 53: 8 observations
##   predicted class=1  expected loss=0.125  P(node) =0.0152381
##     class counts:     1     7
##    probabilities: 0.125 0.875
##
## Node number 94: 11 observations
##   predicted class=0  expected loss=0.1818182  P(node) =0.02095238
##     class counts:     9     2
##    probabilities: 0.818 0.182
##
## Node number 95: 78 observations,    complexity param=0.01142857
##   predicted class=1  expected loss=0.3717949  P(node) =0.1485714
##     class counts:    29    49
##    probabilities: 0.372 0.628
##   left son=190 (14 obs) right son=191 (64 obs)
##   Primary splits:
##       BloodPressure            < 1.057555    to the right, improve=1.9795170, (4 missing)
##       DiabetesPedigreeFunction < 0.1693898   to the left,  improve=1.8692310, (0 missing)
##       Age                      < 0.7022918   to the left,  improve=1.8692310, (0 missing)
##       Glucose                  < -0.1698593  to the right, improve=0.8963847, (1 missing)
##       BMI                      < -0.4631724  to the right, improve=0.8466117, (0 missing)
##   Surrogate splits:
##       BMI < 2.027806    to the right, agree=0.865, adj=0.231, (4 split)
##
## Node number 190: 14 observations
##   predicted class=0  expected loss=0.4285714  P(node) =0.02666667
##     class counts:     8     6
##    probabilities: 0.571 0.429
##
## Node number 191: 64 observations,    complexity param=0.01142857
##   predicted class=1  expected loss=0.328125  P(node) =0.1219048
##     class counts:    21    43
##    probabilities: 0.328 0.672
##   left son=382 (37 obs) right son=383 (27 obs)
##   Primary splits:
```

```
##       Age                      < 0.7022918   to the left,  improve=1.9084400, (0 missing)
##       DiabetesPedigreeFunction < 0.1693898   to the left,  improve=1.3469550, (0 missing)
##       SkinThickness            < 0.4625932   to the left,  improve=1.1283480, (24 missing)
##       Glucose                  < -0.1698593  to the right, improve=0.9142857, (1 missing)
##       BMI                      < 0.937552    to the left,  improve=0.5686642, (0 missing)
##   Surrogate splits:
##       Glucose                  < -0.6283399  to the right, agree=0.641, adj=0.148, (0 split)
##       DiabetesPedigreeFunction < 1.957645    to the left,  agree=0.625, adj=0.111, (0 split)
##       BMI                      < -0.4631724  to the right, agree=0.609, adj=0.074, (0 split)
##       Pregnancies              < 1.08469     to the left,  agree=0.594, adj=0.037, (0 split)
##       BloodPressure            < -0.1134846  to the left,  agree=0.594, adj=0.037, (0 split)
##
## Node number 382: 37 observations,    complexity param=0.01142857
##   predicted class=1  expected loss=0.4324324  P(node) =0.07047619
##     class counts:    16    21
##    probabilities: 0.432 0.568
##   left son=764 (18 obs) right son=765 (19 obs)
##   Primary splits:
##       Glucose                  < -0.08798776 to the right, improve=2.0000000, (1 missing)
##       SkinThickness            < 0.1762512   to the left,  improve=1.1244150, (14 missing)
##       BloodPressure            < -0.4365301  to the right, improve=1.0317460, (2 missing)
##       DiabetesPedigreeFunction < 0.1693898   to the left,  improve=0.9696156, (0 missing)
##       BMI                      < -0.3765297  to the left,  improve=0.7695696, (0 missing)
##   Surrogate splits:
##       DiabetesPedigreeFunction < -0.0222628  to the left,  agree=0.694, adj=0.389, (1 split)
##       Pregnancies              < 1.381463    to the right, agree=0.667, adj=0.333, (0 split)
##       Age                      < -0.06299922 to the right, agree=0.667, adj=0.333, (0 split)
##       BMI                      < -0.3981904  to the left,  agree=0.583, adj=0.167, (0 split)
##       BloodPressure            < 0.2095609   to the right, agree=0.556, adj=0.111, (0 split)
##
## Node number 383: 27 observations
##   predicted class=1  expected loss=0.1851852  P(node) =0.05142857
##     class counts:     5    22
##    probabilities: 0.185 0.815
##
## Node number 764: 18 observations
##   predicted class=0  expected loss=0.3888889  P(node) =0.03428571
##     class counts:    11     7
##    probabilities: 0.611 0.389
##
## Node number 765: 19 observations
##   predicted class=1  expected loss=0.2631579  P(node) =0.03619048
##     class counts:     5    14
##    probabilities: 0.263 0.737
```

```r
rpart.plot(outcome_tree_scaled, fallen.leaves = FALSE)
```




```r
#Variable Importance
outcome_tree_scaled$variable.importance
```

```
##                  Glucose                      Age DiabetesPedigreeFunction
##                 65.13983                 25.43765                 17.94039
##                      BMI              Pregnancies            BloodPressure
##                 12.23408                 12.14003                  5.92835
```

```r
# Accuracy for testing data
predictions_Outcome_testing = predict(outcome_tree_scaled, newdata=testing_scaled, type='class')
accuracytesting_t <- mean(predictions_Outcome_testing == testing_scaled$Outcome)
accuracytesting_t
```

```
## [1] 0.7078189
```

```r
confusionMatrix(predictions_Outcome_testing, testing_scaled$Outcome)$table
```

```
##           Reference
## Prediction   0   1
##          0 129  50
##          1  21  43
```

The decision tree has a higher depth for the scaled data, this is due the less variability of the variables, that makes more difficult to find an accurate split. Then the model with scaled data is more complex and prone to overfitting.

13. K-means

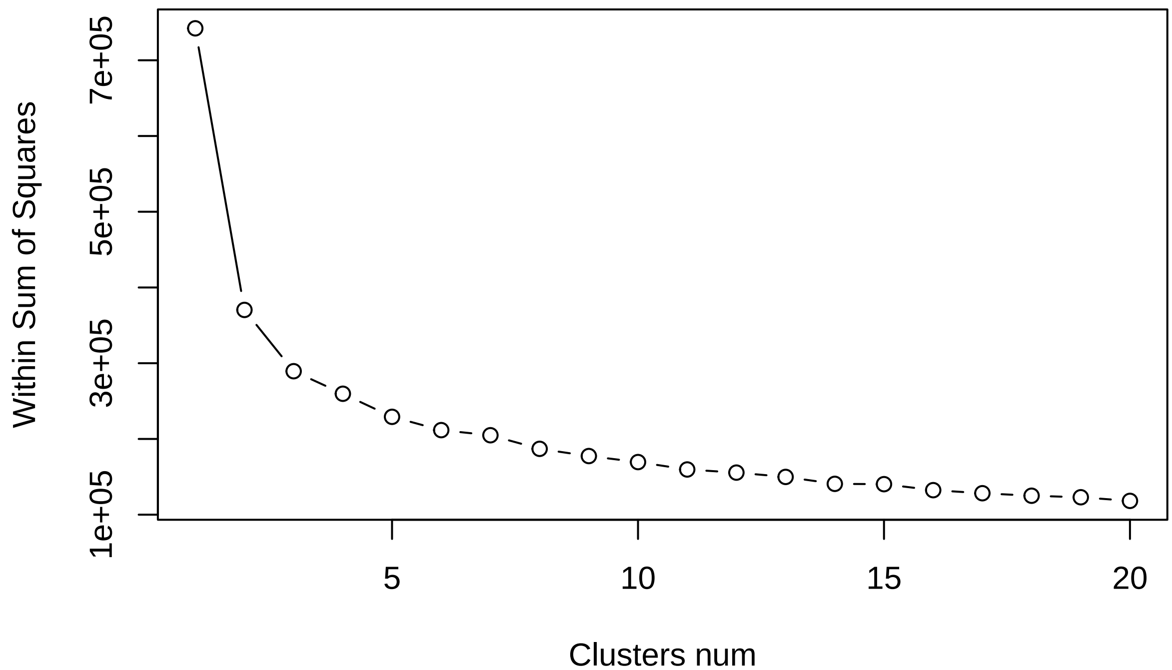
K-means on diabetes dataset:

```
datasetdiabetesNoNAs <-na.omit(datasetdiabetes_no_insulin) #Data without Nas and without insulin
numeric_datasetdiabetes=datasetdiabetesNoNAs[,1:7]

#install.packages("MASS")
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

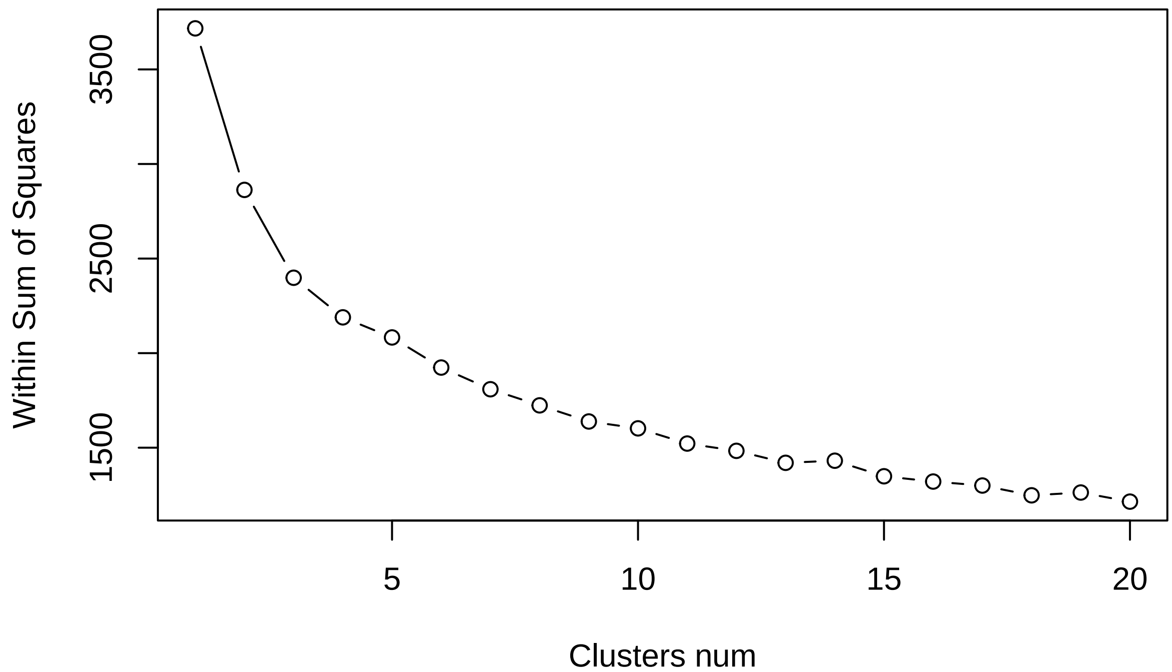
```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
clust_max = 20

#Within total sum of squares, for data
wss = c()

for( k in 1:clust_max) {
   kmeans_model = kmeans(datasetdiabetesNoNAs, centers=k)
   wss[k] = kmeans_model$tot.withinss
}

#Elbow - Plot
plot(1:clust_max, wss, type="b", xlab="Clusters num", ylab="Within Sum of Squares")
```



For the scaled data:

```
#Within total sum of squares, for scale data
scale_datasediabetes <-scale(numeric_datasetdiabetes)
wss = c()

for( k in 1:clust_max) {
   kmeans_model = kmeans(scale_datasediabetes, centers=k)
   wss[k] = kmeans_model$tot.withinss
}
#Elbow - Plot
plot(1:clust_max, wss, type="b", xlab="Clusters num", ylab="Within Sum of Squares")
```



Conclusions: In the MSS plot for the data, we can observe an elbow, which indicates I should use 3 clusters, also because of the different measurements between the variables we can observe larger numbers in the MSS plot compared with the MSS plot for non scaled data. In the scaled data MSS plot, there is not a clear elbow in the graph, it indicates that the MSS will slowly decrease even for more than 3 clusters.

14.

I decided to use 2 and 4 clusters because the variable outcome has 2 factors:

Non-Scaled Dataset:

```
##for all the variables included insulin
kmeans_result_2_complete = kmeans(na.omit(datasetdiabetesNas[,1:8]), centers=2)
t1 = table(na.omit(datasetdiabetesNas)$Outcome, kmeans_result_2_complete$cluster)
t1
```

```
##
##        1   2
##   0   27 235
##   1   29 101
```

The data set without the column insulin has a better performance in Kmeans that the whole dataset:

```r
kmeans_result_2 = kmeans(datasetdiabetesNoNAs[,1:7], centers=2)
t2 = table(datasetdiabetesNoNAs$Outcome, kmeans_result_2$cluster)
t2
```

```
##
##       1   2
##   0  61 294
##   1 110  67
```

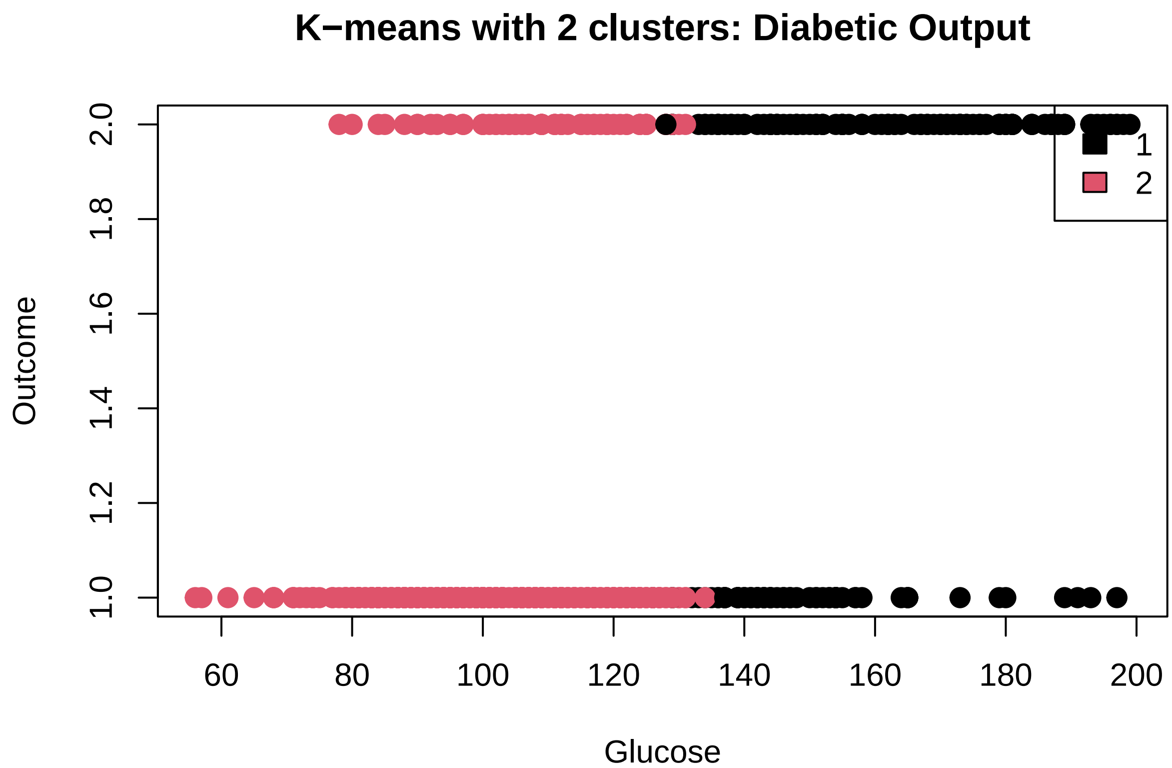
```r
kmeans_result_4 = kmeans(datasetdiabetesNoNAs[,1:7], centers=4)
t3 = table(datasetdiabetesNoNAs$Outcome, kmeans_result_4$cluster)
t3
```

```
##
##       1   2   3   4
##   0  83 135 112  25
##   1  27  11  61  78
```

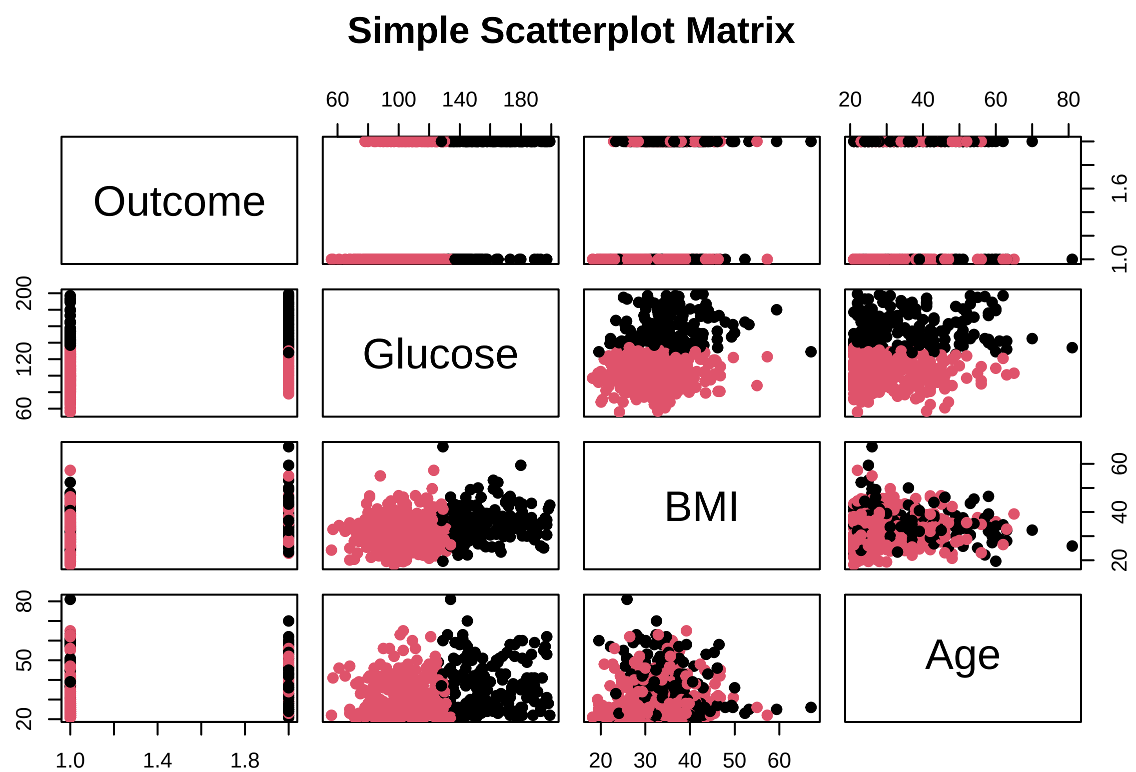
```r
## acuraccy
n=nrow(datasetdiabetesNoNAs)
accuracy=c((110+294)/n*100,(168+96+78+38)/n*100)
print(accuracy)
```

```
## [1] 75.93985 71.42857
```

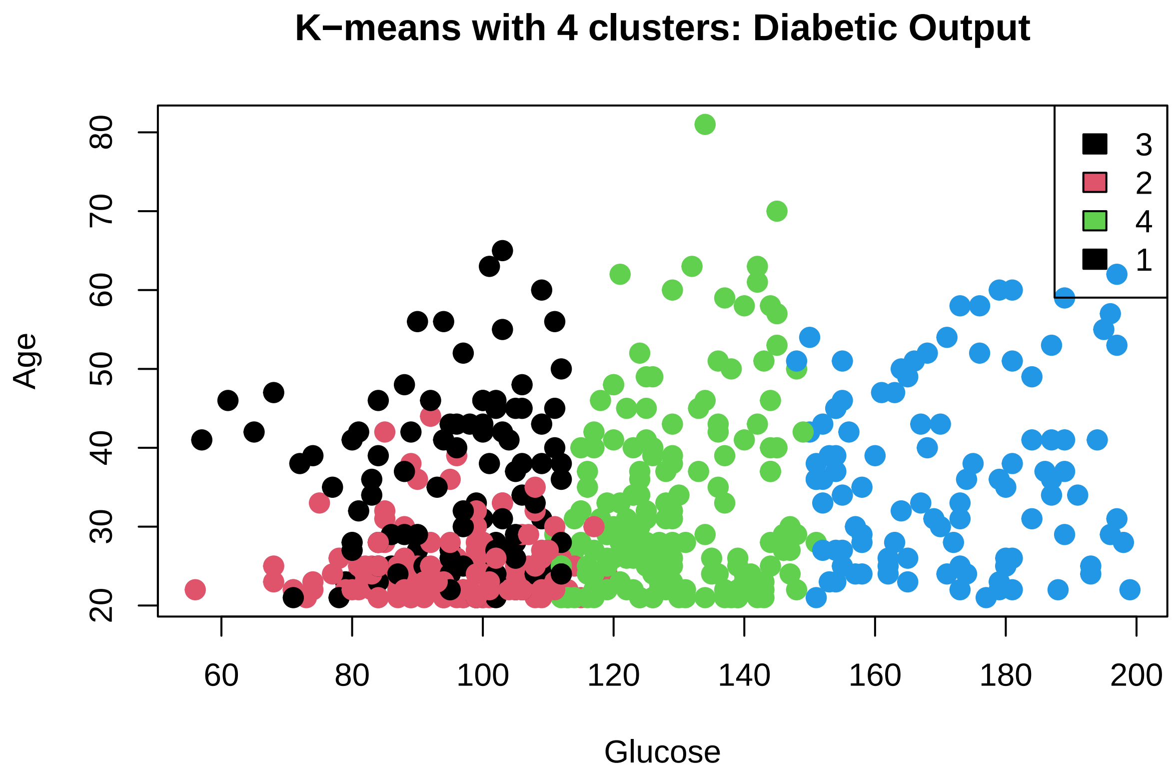
```r
##Graphs:

##Graph for 2 clusters:
##Plot
plot(datasetdiabetesNoNAs$Glucose, datasetdiabetesNoNAs$Outcome, col=kmeans_result_2$cluster,
     main="K-means with 2 clusters: Diabetic Output",
     xlab="Glucose", ylab="Outcome", pch=20, cex=2)
legend("topright", legend=unique(kmeans_result_2$cluster), fill=1:3)
```



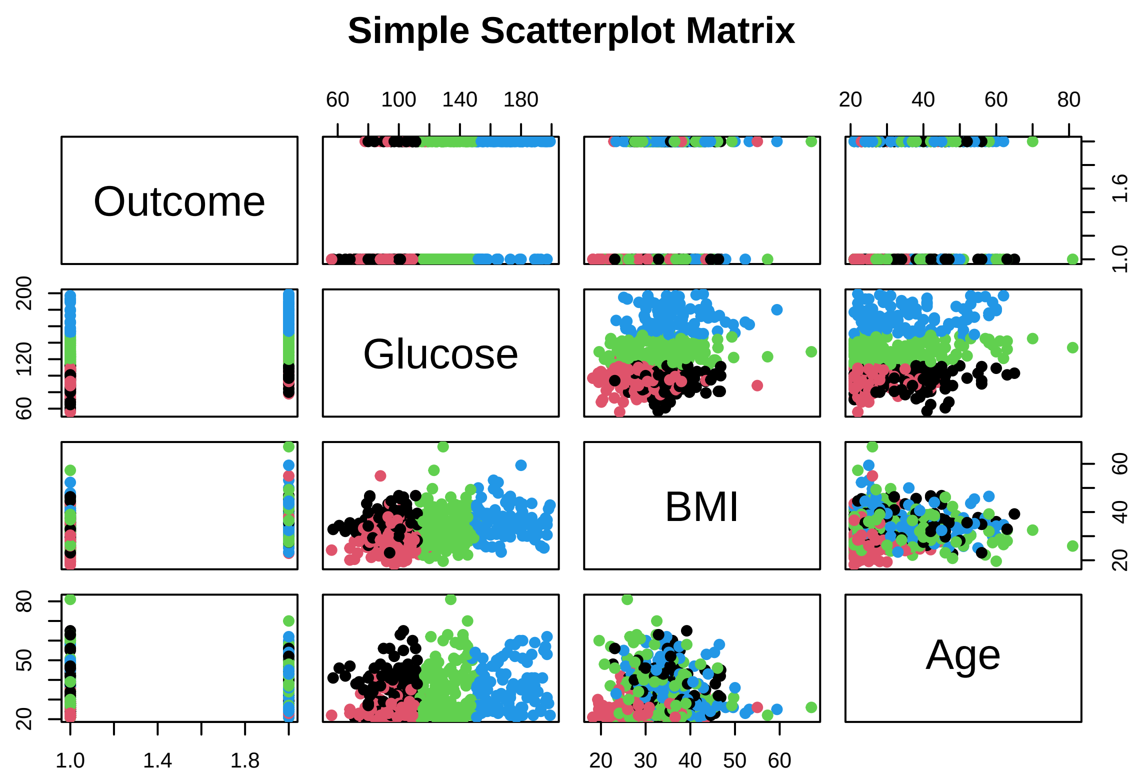

```
## Scatter Plot
#use of dataset diabetes nas because there the Output is not a factor
pairs(~Outcome+Glucose+BMI+Age,data=na.omit(datasetdiabetesNas[,-5]),
   main="Simple Scatterplot Matrix",col=kmeans_result_2$cluster, pch=20, cex=1.5)
```




```
##Graph for 4 clusters:
##Plot:
plot(datasetdiabetesNoNAs$Glucose, datasetdiabetesNoNAs$Age, col=kmeans_result_4$cluster,
     main="K-means with 4 clusters: Diabetic Output",
     xlab="Glucose", ylab="Age", pch=20, cex=2)
legend("topright", legend=unique(kmeans_result_4$cluster), fill=1:3)
```




```
##Scatter Plot:
pairs(~Outcome+Glucose+BMI+Age,data=na.omit(datasetdiabetesNas[,-5]),
   main="Simple Scatterplot Matrix",col=kmeans_result_4$cluster, pch=20, cex=1.5)
```



For the non-scaled data it is observed that the the accuracy for the variable outcome is better for 2 than for 4 clusters, and for 4 clusters there is more observed variability in the kmeans.

Scaled Dataset:

```
kmeans_result_2_scaled = kmeans(scale_datasediabetes, centers=2)
t1 = table(datasetdiabetesNoNAs$Outcome, kmeans_result_2_scaled$cluster)
t1
```

```
##
##       1   2
##   0 105 250
##   1 126  51
```

```
kmeans_result_4_scaled = kmeans(scale_datasediabetes, centers=4)
t1 = table(datasetdiabetesNoNAs$Outcome, kmeans_result_4_scaled$cluster)
t1
```

```
##
##       1   2   3   4
##   0 193  55  24  83
##   1  20  33  55  69
```

```
## acuraccy
n=nrow(scale_datasediabetes)
accuracy=c((249+127)/n*100,(160+115+72+40)/n*100)
print(accuracy)
```

```
## [1] 70.67669 72.74436
```

There is more variation in the scaled data set with 2 clusters because the WSS does not decrease as fast as the non-scaled data. The accuracy doesn't improve for the outcome with 4 clusters.

15. K-medoids

For k=2 clusters:

```
#install.packages("cluster")
library(cluster)
#install.packages("factoextra")
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
# K-medoids for k=2
nrow(datasetdiabetesNoNAs)
```

```
## [1] 532
```

```
diabetes.pam=pam(na.omit(datasetdiabetesNoNAs), 2)
diabetes.pam
```

```
## Medoids:
##      ID Pregnancies Glucose BloodPressure SkinThickness  BMI
## 339 233           9     152            78            34 34.2
## 390 268           3     100            68            23 31.6
##     DiabetesPedigreeFunction Age Outcome
## 339                    0.893  33       2
## 390                    0.949  28       1
## Clustering vector:
##   1   2   4   5   7   9  14  15  17  19  20  21  24  25  26  28  29  31  32  33
##   1   2   2   1   2   1   1   1   2   2   2   1   2   1   2   2   1   2   1   2
##  35  36  38  39  40  41  43  44  46  48  49  51  52  53  54  55  56  57  58  60
##   2   2   2   2   2   1   2   1   1   2   2   2   2   2   1   1   2   1   2   2
##  64  66  67  69  70  71  72  74  75  78  80  81  83  84  86  87  88  89  90  92
##   1   2   2   2   1   2   1   1   2   2   2   2   2   2   2   2   2   1   2   2
##  93  95  96  97  98  99 100 104 106 108 109 110 111 112 113 115 119 120 121 122
##   2   1   1   2   2   2   1   2   2   1   2   2   1   1   2   1   2   2   1   2
## 123 126 127 128 129 131 133 134 135 136 137 138 140 142 143 145 147 148 150 151
##   2   2   2   2   2   1   1   2   2   2   2   2   2   2   2   1   2   2   2   1
## 153 154 156 157 158 159 160 161 162 163 164 166 167 170 172 174 175 176 178 182
##   1   1   1   2   2   2   1   1   2   2   2   2   1   2   1   2   2   1   1   2
```

```
## 186 187 188 189 190 192 195 196 198 199 200 201 203 204 205 206 207 209 210 211
##   1   1   1   2   1   1   2   1   2   2   1   2   2   2   2   2   1   2   1   2
## 212 213 214 215 216 217 218 219 221 224 225 226 228 229 230 232 233 235 237 238
##   1   1   1   2   1   2   2   2   1   1   2   2   1   1   2   1   2   2   1   1
## 239 241 242 244 245 246 248 249 250 253 254 255 256 257 258 259 260 261 263 264
##   1   2   2   2   1   1   1   2   2   2   2   2   2   2   2   1   1   1   2   1
## 266 268 271 272 274 276 277 278 280 282 283 286 287 288 289 290 291 292 293 294
##   2   1   2   2   2   2   2   2   2   1   1   1   1   2   2   2   2   2   1   2
## 296 297 298 299 302 303 306 307 308 309 310 311 312 313 314 315 316 317 319 321
##   1   1   1   2   1   2   2   1   1   2   2   2   2   1   2   2   2   2   2   2
## 322 323 324 325 326 327 329 330 331 332 335 336 339 341 342 346 347 349 353 354
##   2   2   1   2   1   2   2   2   2   2   2   1   1   2   2   1   1   2   2   2
## 357 359 360 361 363 365 366 368 369 370 371 373 374 375 376 377 378 380 381 382
##   2   2   1   1   2   1   2   2   2   1   1   2   2   2   1   2   2   2   2   2
## 383 384 385 386 387 388 389 390 391 393 394 396 397 398 400 403 404 406 410 411
##   2   2   2   2   2   2   1   2   2   2   2   2   2   1   1   1   2   2   1   2
## 412 413 414 415 416 417 418 420 421 422 423 424 425 426 428 429 430 432 433 435
##   2   1   1   1   1   2   1   1   2   2   2   2   1   1   1   1   2   2   2   2
## 437 439 441 442 443 445 446 447 448 449 450 451 453 455 456 458 459 460 461 463
##   1   2   1   2   2   2   1   2   2   2   2   2   2   2   1   2   1   1   2   2
## 464 466 467 468 470 471 472 473 476 477 478 479 480 481 482 483 484 486 487 488
##   2   2   2   2   1   1   1   2   1   2   2   1   1   1   1   2   2   1   1   1
## 489 491 492 493 494 498 499 500 501 502 504 505 507 508 509 511 512 515 516 517
##   2   2   2   2   2   2   1   1   2   2   2   2   1   2   2   2   1   2   1   1
## 520 521 522 526 527 528 529 531 533 535 539 540 541 542 543 544 545 546 547 548
##   1   2   2   2   2   2   2   2   2   2   1   1   2   1   2   2   2   1   1   1
## 549 550 551 552 554 555 556 557 559 562 563 564 566 567 568 569 570 573 574 575
##   1   1   2   2   2   2   2   2   2   1   2   2   2   2   2   1   2   2   2   1
## 576 577 580 581 582 583 585 586 589 591 592 594 595 596 598 600 601 603 604 606
##   2   2   1   1   2   2   2   2   1   2   2   2   1   1   2   2   2   2   1   2
## 607 608 609 610 611 612 613 614 615 618 619 621 622 624 626 630 632 634 638 639
##   1   2   1   2   2   1   1   2   1   2   2   2   2   2   2   2   2   2   2   2
## 640 641 645 646 647 648 649 650 651 652 653 655 656 657 658 660 662 663 664 665
##   2   2   2   1   1   1   1   2   2   2   2   2   1   2   1   2   1   1   1   2
## 666 667 668 669 670 671 672 673 674 680 681 682 683 686 688 689 690 693 694 696
##   2   1   2   2   1   1   2   2   1   2   2   1   2   1   2   1   1   2   1   1
## 697 699 701 702 703 705 706 708 710 711 712 713 714 716 717 718 719 720 721 722
##   1   2   2   1   1   2   2   2   2   1   1   1   1   1   1   2   2   2   2   2
## 723 724 726 727 728 731 733 734 736 737 738 739 741 742 743 745 746 747 748 749
##   1   2   2   2   1   1   1   2   2   1   2   2   2   2   2   1   2   1   2   1
## 752 753 754 755 756 757 761 762 764 765 766 768
##   2   2   1   1   1   1   2   1   2   2   2   2
## Objective function:
##    build     swap
## 26.59346 25.12324
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```
str(diabetes.pam)
```

```
## List of 10
##  $ medoids   : num [1:2, 1:8] 9 3 152 100 78 68 34 23 34.2 31.6 ...
```

```
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "339" "390"
##   .. ..$ : chr [1:8] "Pregnancies" "Glucose" "BloodPressure" "SkinThickness" ...
##  $ id.med    : int [1:2] 233 268
##  $ clustering: Named int [1:532] 1 2 2 1 2 1 1 1 2 2 ...
##   ..- attr(*, "names")= chr [1:532] "1" "2" "4" "5" ...
##  $ objective : Named num [1:2] 26.6 25.1
##   ..- attr(*, "names")= chr [1:2] "build" "swap"
##  $ isolation : Factor w/ 3 levels "no","L","L*": 1 1
##   ..- attr(*, "names")= chr [1:2] "1" "2"
##  $ clusinfo  : num [1:2, 1:5] 196 336 84.9 56.9 28.8 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:5] "size" "max_diss" "av_diss" "diameter" ...
##  $ silinfo   :List of 3
##   ..$ widths         : num [1:532, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:532] "717" "762" "210" "176" ...
##   .. .. ..$ : chr [1:3] "cluster" "neighbor" "sil_width"
##   ..$ clus.avg.widths: num [1:2] 0.32 0.47
##   ..$ avg.width      : num 0.415
##  $ diss      : NULL
##  $ call      : language pam(x = na.omit(datasetdiabetesNoNAs), k = 2)
##  $ data      : num [1:532, 1:8] 6 1 1 0 3 2 1 5 0 1 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:532] "1" "2" "4" "5" ...
##   .. ..$ : chr [1:8] "Pregnancies" "Glucose" "BloodPressure" "SkinThickness" ...
##  - attr(*, "class")= chr [1:2] "pam" "partition"
```

```
#Medoids and pam algorithm
as.vector(diabetes.pam$medoids[1,])
```

```
## [1]   9.000 152.000  78.000  34.000  34.200   0.893  33.000   2.000
```

```
## Graph for K = 2
summary(diabetes.pam)
```

```
## Medoids:
##      ID Pregnancies Glucose BloodPressure SkinThickness  BMI
## 339 233           9     152            78            34 34.2
## 390 268           3     100            68            23 31.6
##     DiabetesPedigreeFunction Age Outcome
## 339                    0.893  33       2
## 390                    0.949  28       1
## Clustering vector:
##   1   2   4   5   7   9  14  15  17  19  20  21  24  25  26  28  29  31  32  33
##   1   2   2   1   2   1   1   1   2   2   2   1   2   1   2   2   1   2   1   2
##  35  36  38  39  40  41  43  44  46  48  49  51  52  53  54  55  56  57  58  60
##   2   2   2   2   2   1   2   1   1   2   2   2   2   2   1   1   2   1   2   2
##  64  66  67  69  70  71  72  74  75  78  80  81  83  84  86  87  88  89  90  92
##   1   2   2   2   1   2   1   1   2   2   2   2   2   2   2   2   2   1   2   2
##  93  95  96  97  98  99 100 104 106 108 109 110 111 112 113 115 119 120 121 122
##   2   1   1   2   2   2   1   2   2   1   2   2   1   1   2   1   2   2   1   2
```

```
## 123 126 127 128 129 131 133 134 135 136 137 138 140 142 143 145 147 148 150 151
##   2   2   2   2   2   1   1   2   2   2   2   2   2   2   2   1   2   2   2   1
## 153 154 156 157 158 159 160 161 162 163 164 166 167 170 172 174 175 176 178 182
##   1   1   1   2   2   2   1   1   2   2   2   2   1   2   1   2   2   1   1   2
## 186 187 188 189 190 192 195 196 198 199 200 201 203 204 205 206 207 209 210 211
##   1   1   1   2   1   1   2   1   2   2   1   2   2   2   2   2   1   2   1   2
## 212 213 214 215 216 217 218 219 221 224 225 226 228 229 230 232 233 235 237 238
##   1   1   1   2   1   2   2   2   1   1   2   2   1   1   2   1   2   2   1   1
## 239 241 242 244 245 246 248 249 250 253 254 255 256 257 258 259 260 261 263 264
##   1   2   2   2   1   1   1   2   2   2   2   2   2   2   2   1   1   1   2   1
## 266 268 271 272 274 276 277 278 280 282 283 286 287 288 289 290 291 292 293 294
##   2   1   2   2   2   2   2   2   2   1   1   1   1   2   2   2   2   2   1   2
## 296 297 298 299 302 303 306 307 308 309 310 311 312 313 314 315 316 317 319 321
##   1   1   1   2   1   2   2   1   1   2   2   2   2   1   2   2   2   2   2   2
## 322 323 324 325 326 327 329 330 331 332 335 336 339 341 342 346 347 349 353 354
##   2   2   1   2   1   2   2   2   2   2   2   1   1   2   2   1   1   2   2   2
## 357 359 360 361 363 365 366 368 369 370 371 373 374 375 376 377 378 380 381 382
##   2   2   1   1   2   1   2   2   2   1   1   2   2   2   1   2   2   2   2   2
## 383 384 385 386 387 388 389 390 391 393 394 396 397 398 400 403 404 406 410 411
##   2   2   2   2   2   2   1   2   2   2   2   2   2   1   1   1   2   2   1   2
## 412 413 414 415 416 417 418 420 421 422 423 424 425 426 428 429 430 432 433 435
##   2   1   1   1   1   2   1   1   2   2   2   2   1   1   1   1   2   2   2   2
## 437 439 441 442 443 445 446 447 448 449 450 451 453 455 456 458 459 460 461 463
##   1   2   1   2   2   2   1   2   2   2   2   2   2   2   1   2   1   1   2   2
## 464 466 467 468 470 471 472 473 476 477 478 479 480 481 482 483 484 486 487 488
##   2   2   2   2   1   1   1   2   1   2   2   1   1   1   1   2   2   1   1   1
## 489 491 492 493 494 498 499 500 501 502 504 505 507 508 509 511 512 515 516 517
##   2   2   2   2   2   2   1   1   2   2   2   2   1   2   2   2   1   2   1   1
## 520 521 522 526 527 528 529 531 533 535 539 540 541 542 543 544 545 546 547 548
##   1   2   2   2   2   2   2   2   2   2   1   1   2   1   2   2   2   1   1   1
## 549 550 551 552 554 555 556 557 559 562 563 564 566 567 568 569 570 573 574 575
##   1   1   2   2   2   2   2   2   2   1   2   2   2   2   2   1   2   2   2   1
## 576 577 580 581 582 583 585 586 589 591 592 594 595 596 598 600 601 603 604 606
##   2   2   1   1   2   2   2   2   1   2   2   2   1   1   2   2   2   2   1   2
## 607 608 609 610 611 612 613 614 615 618 619 621 622 624 626 630 632 634 638 639
##   1   2   1   2   2   1   1   2   1   2   2   2   2   2   2   2   2   2   2   2
## 640 641 645 646 647 648 649 650 651 652 653 655 656 657 658 660 662 663 664 665
##   2   2   2   1   1   1   1   2   2   2   2   2   1   2   1   2   1   1   1   2
## 666 667 668 669 670 671 672 673 674 680 681 682 683 686 688 689 690 693 694 696
##   2   1   2   2   1   1   2   2   1   2   2   1   2   1   2   1   1   2   1   1
## 697 699 701 702 703 705 706 708 710 711 712 713 714 716 717 718 719 720 721 722
##   1   2   2   1   1   2   2   2   2   1   1   1   1   1   1   2   2   2   2   2
## 723 724 726 727 728 731 733 734 736 737 738 739 741 742 743 745 746 747 748 749
##   1   2   2   2   1   1   1   2   2   1   2   2   2   2   2   1   2   1   2   1
## 752 753 754 755 756 757 761 762 764 765 766 768
##   2   2   1   1   1   1   2   1   2   2   2   2
## Objective function:
##    build     swap
## 26.59346 25.12324
##
## Numerical information per cluster:
##      size max_diss  av_diss  diameter separation
## [1,]  196 84.87845 28.81575 117.20499   5.912326
## [2,]  336 56.88739 22.96927  98.27525   5.912326
```

```
##
## Isolated clusters:
##  L-clusters: character(0)
##  L*-clusters: character(0)
##
## Silhouette plot information:
##     cluster neighbor    sil_width
## 717       1        2  0.5444252223
## 762       1        2  0.5327900917
## 210       1        2  0.5317113033
## 176       1        2  0.5310541845
## 426       1        2  0.5307788419
## 613       1        2  0.5279579757
## 111       1        2  0.5214098555
## 428       1        2  0.5168352047
## 507       1        2  0.5166370488
## 546       1        2  0.5155046228
## 416       1        2  0.5152781386
## 57        1        2  0.5149011649
## 133       1        2  0.5147218314
## 589       1        2  0.5112127095
## 481       1        2  0.5100360337
## 749       1        2  0.5075847380
## 733       1        2  0.5066796827
## 646       1        2  0.5065595314
## 607       1        2  0.5064081350
## 32        1        2  0.5051900519
## 371       1        2  0.5048161165
## 703       1        2  0.5046892097
## 549       1        2  0.5040305533
## 697       1        2  0.5029692203
## 196       1        2  0.5025412073
## 456       1        2  0.5019943107
## 46        1        2  0.5009546012
## 237       1        2  0.5006837213
## 754       1        2  0.4997050346
## 671       1        2  0.4996872595
## 239       1        2  0.4980044271
## 186       1        2  0.4960000620
## 547       1        2  0.4957513671
## 41        1        2  0.4957026678
## 361       1        2  0.4952629621
## 360       1        2  0.4951767414
## 153       1        2  0.4936485262
## 336       1        2  0.4928850706
## 755       1        2  0.4913053080
## 400       1        2  0.4912104206
## 410       1        2  0.4910550152
## 500       1        2  0.4894882020
## 238       1        2  0.4892763262
## 287       1        2  0.4888626663
## 229       1        2  0.4887711397
## 160       1        2  0.4876914714
## 54        1        2  0.4874561041
```

```
## 682         1         2  0.4857189317
## 569         1         2  0.4851951568
## 488         1         2  0.4841703578
## 670         1         2  0.4839814472
## 339         1         2  0.4819911686
## 131         1         2  0.4798145193
## 612         1         2  0.4776167003
## 246         1         2  0.4758538491
## 187         1         2  0.4739481318
## 562         1         2  0.4733701779
## 207         1         2  0.4716170490
## 647         1         2  0.4715089898
## 261         1         2  0.4701118838
## 221         1         2  0.4699758860
## 516         1         2  0.4689043885
## 425         1         2  0.4684101007
## 15          1         2  0.4676722011
## 213         1         2  0.4673285108
## 745         1         2  0.4673251761
## 499         1         2  0.4662997776
## 260         1         2  0.4645965777
## 9           1         2  0.4644990870
## 662         1         2  0.4642172332
## 307         1         2  0.4609341461
## 470         1         2  0.4596740621
## 248         1         2  0.4588715293
## 441         1         2  0.4573650393
## 596         1         2  0.4570127152
## 609         1         2  0.4555487178
## 716         1         2  0.4508035742
## 161         1         2  0.4481855761
## 154         1         2  0.4428837068
## 216         1         2  0.4420362147
## 663         1         2  0.4413286854
## 14          1         2  0.4409195699
## 550         1         2  0.4374326254
## 324         1         2  0.4358100325
## 156         1         2  0.4323228910
## 648         1         2  0.4301017327
## 112         1         2  0.4297963039
## 115         1         2  0.4263810841
## 604         1         2  0.4230318654
## 326         1         2  0.4215744647
## 55          1         2  0.4206180609
## 121         1         2  0.4157402398
## 1           1         2  0.4134417155
## 723         1         2  0.4088226557
## 446         1         2  0.4080088927
## 44          1         2  0.4079016124
## 313         1         2  0.4059419280
## 245         1         2  0.4046483452
## 145         1         2  0.4039217857
## 228         1         2  0.4031954655
## 296         1         2  0.3983935456
```

```
## 259         1         2  0.3979016779
## 418         1         2  0.3974147672
## 459         1         2  0.3948098581
## 365         1         2  0.3932193213
## 664         1         2  0.3926676140
## 581         1         2  0.3898596008
## 711         1         2  0.3747825158
## 471         1         2  0.3705911449
## 517         1         2  0.3683281080
## 70          1         2  0.3675072321
## 297         1         2  0.3663583551
## 96          1         2  0.3656864899
## 690         1         2  0.3603618184
## 747         1         2  0.3597880199
## 656         1         2  0.3472117154
## 212         1         2  0.3438313736
## 25          1         2  0.3395615853
## 200         1         2  0.3351671392
## 167         1         2  0.3344234872
## 389         1         2  0.3319299337
## 437         1         2  0.3266437015
## 696         1         2  0.3222234562
## 575         1         2  0.3115813509
## 580         1         2  0.3005295451
## 29          1         2  0.2908242162
## 376         1         2  0.2900922493
## 413         1         2  0.2883709883
## 108         1         2  0.2743290514
## 757         1         2  0.2705760930
## 667         1         2  0.2683685070
## 190         1         2  0.2642736451
## 728         1         2  0.2632148641
## 302         1         2  0.2557501998
## 403         1         2  0.2525261116
## 615         1         2  0.2482142964
## 264         1         2  0.2453718347
## 414         1         2  0.2453703582
## 649         1         2  0.2394127838
## 224         1         2  0.2358589067
## 476         1         2  0.2178235208
## 95          1         2  0.2177357797
## 232         1         2  0.2076091654
## 689         1         2  0.2044269967
## 89          1         2  0.2027088649
## 214         1         2  0.1988845331
## 429         1         2  0.1979386234
## 72          1         2  0.1942188879
## 286         1         2  0.1880593184
## 487         1         2  0.1793327863
## 151         1         2  0.1787342185
## 64          1         2  0.1775600909
## 472         1         2  0.1703505838
## 370         1         2  0.1642231444
## 480         1         2  0.1430040117
```

```
## 486         1         2  0.1347018959
## 415         1         2  0.1318195603
## 460         1         2  0.1250641531
## 283         1         2  0.1000010264
## 540         1         2  0.0965737308
## 178         1         2  0.0851914785
## 172         1         2  0.0776899501
## 512         1         2  0.0737459967
## 188         1         2  0.0683164278
## 756         1         2  0.0625112952
## 694         1         2  0.0508618408
## 5           1         2  0.0485446699
## 308         1         2  0.0463466554
## 346         1         2  0.0398518205
## 347         1         2  0.0277660753
## 520         1         2  0.0206857056
## 282         1         2  0.0133603174
## 293         1         2  0.0008114586
## 731         1         2 -0.0107743897
## 398         1         2 -0.0130986234
## 21          1         2 -0.0188852554
## 713         1         2 -0.0243175882
## 674         1         2 -0.0390360899
## 74          1         2 -0.0401788143
## 100         1         2 -0.0479313031
## 714         1         2 -0.0517988318
## 548         1         2 -0.0550563835
## 702         1         2 -0.0664889348
## 539         1         2 -0.0691568344
## 479         1         2 -0.0801711797
## 268         1         2 -0.0830680538
## 686         1         2 -0.0834794100
## 712         1         2 -0.0942471191
## 298         1         2 -0.1158240320
## 482         1         2 -0.1203435525
## 737         1         2 -0.1224627310
## 420         1         2 -0.1240407290
## 542         1         2 -0.1256829578
## 192         1         2 -0.1381167170
## 658         1         2 -0.1388938646
## 595         1         2 -0.1537410638
## 97          2         1  0.6331469168
## 241         2         1  0.6313287825
## 554         2         1  0.6303215967
## 458         2         1  0.6300399145
## 4           2         1  0.6296105531
## 768         2         1  0.6267718599
## 552         2         1  0.6244697662
## 209         2         1  0.6240962482
## 138         2         1  0.6240402009
## 263         2         1  0.6230353867
## 390         2         1  0.6229460938
## 53          2         1  0.6228934969
## 417         2         1  0.6221654712
```

```
## 630         2          1  0.6218155977
## 2           2          1  0.6214528932
## 242         2          1  0.6212839549
## 442         2          1  0.6202022460
## 384         2          1  0.6186560649
## 254         2          1  0.6176836629
## 119         2          1  0.6174889352
## 761         2          1  0.6169461467
## 555         2          1  0.6167341366
## 150         2          1  0.6160585630
## 422         2          1  0.6160389762
## 491         2          1  0.6150666211
## 71          2          1  0.6146460270
## 164         2          1  0.6142753513
## 483         2          1  0.6136083751
## 159         2          1  0.6123767138
## 219         2          1  0.6123111348
## 710         2          1  0.6117505613
## 373         2          1  0.6110261144
## 736         2          1  0.6106598023
## 137         2          1  0.6102775845
## 211         2          1  0.6087942151
## 563         2          1  0.6085225328
## 453         2          1  0.6083896024
## 502         2          1  0.6077950518
## 545         2          1  0.6072703212
## 608         2          1  0.6072321547
## 78          2          1  0.6051712605
## 622         2          1  0.6049913399
## 88          2          1  0.6047662912
## 66          2          1  0.6044478251
## 335         2          1  0.6043197964
## 109         2          1  0.6026822648
## 332         2          1  0.6022666515
## 638         2          1  0.6017285070
## 104         2          1  0.6009738378
## 113         2          1  0.6006794509
## 226         2          1  0.6001069244
## 84          2          1  0.6000777512
## 651         2          1  0.5984174335
## 349         2          1  0.5981674091
## 489         2          1  0.5977794322
## 526         2          1  0.5976089631
## 354         2          1  0.5965948374
## 498         2          1  0.5953310304
## 564         2          1  0.5952255062
## 28          2          1  0.5945347520
## 342         2          1  0.5944732571
## 574         2          1  0.5941162844
## 278         2          1  0.5936426860
## 468         2          1  0.5932660806
## 75          2          1  0.5925639284
## 645         2          1  0.5911250999
## 175         2          1  0.5891596737
```

```
## 225          2         1  0.5891527138
## 235          2         1  0.5885925472
## 567          2         1  0.5876995884
## 378          2         1  0.5858154077
## 83           2         1  0.5849784268
## 135          2         1  0.5844423418
## 624          2         1  0.5841920110
## 594          2         1  0.5841870112
## 464          2         1  0.5841692154
## 504          2         1  0.5833182460
## 432          2         1  0.5830900831
## 527          2         1  0.5827277824
## 739          2         1  0.5820936225
## 204          2         1  0.5817085510
## 535          2         1  0.5807927314
## 69           2         1  0.5807223255
## 451          2         1  0.5802945062
## 366          2         1  0.5799250953
## 515          2         1  0.5791551880
## 289          2         1  0.5790190014
## 680          2         1  0.5778548906
## 33           2         1  0.5774060148
## 493          2         1  0.5765090871
## 509          2         1  0.5749284515
## 233          2         1  0.5745037096
## 368          2         1  0.5737814645
## 311          2         1  0.5736642331
## 48           2         1  0.5729904677
## 566          2         1  0.5728259412
## 134          2         1  0.5716642423
## 99           2         1  0.5714431563
## 39           2         1  0.5711282556
## 660          2         1  0.5709144736
## 433          2         1  0.5695729622
## 253          2         1  0.5684192363
## 655          2         1  0.5684008806
## 455          2         1  0.5681736899
## 447          2         1  0.5679070641
## 439          2         1  0.5674637028
## 140          2         1  0.5668620740
## 484          2         1  0.5663172854
## 639          2         1  0.5658214086
## 557          2         1  0.5655305072
## 49           2         1  0.5642933613
## 277          2         1  0.5637706119
## 586          2         1  0.5635995740
## 120          2         1  0.5611610193
## 90           2         1  0.5609187122
## 7            2         1  0.5604630413
## 650          2         1  0.5592205668
## 753          2         1  0.5581638325
## 391          2         1  0.5580798766
## 381          2         1  0.5576404612
## 36           2         1  0.5572468498
```

```
## 435         2          1  0.5566971192
## 706         2          1  0.5551541893
## 382         2          1  0.5551286946
## 521         2          1  0.5544019784
## 734         2          1  0.5535084169
## 449         2          1  0.5529753362
## 657         2          1  0.5528283294
## 672         2          1  0.5524711611
## 292         2          1  0.5510064077
## 397         2          1  0.5506919331
## 195         2          1  0.5504675017
## 157         2          1  0.5504286549
## 721         2          1  0.5471418237
## 266         2          1  0.5468855006
## 110         2          1  0.5466453296
## 568         2          1  0.5462315431
## 404         2          1  0.5460945248
## 123         2          1  0.5457938186
## 683         2          1  0.5451816213
## 611         2          1  0.5448013233
## 203         2          1  0.5446585786
## 330         2          1  0.5437069487
## 640         2          1  0.5431329777
## 369         2          1  0.5410477638
## 148         2          1  0.5407364344
## 511         2          1  0.5404156440
## 377         2          1  0.5394310085
## 174         2          1  0.5364065948
## 463         2          1  0.5363101952
## 618         2          1  0.5353554719
## 467         2          1  0.5350741715
## 614         2          1  0.5350627966
## 52          2          1  0.5340412512
## 582         2          1  0.5340316290
## 669         2          1  0.5329812890
## 272         2          1  0.5328673342
## 748         2          1  0.5300116593
## 291         2          1  0.5282582648
## 198         2          1  0.5252723860
## 743         2          1  0.5250491388
## 430         2          1  0.5248701232
## 303         2          1  0.5248056895
## 98          2          1  0.5237366375
## 544         2          1  0.5222470915
## 56          2          1  0.5221783821
## 738         2          1  0.5215935161
## 158         2          1  0.5213013068
## 505         2          1  0.5206760918
## 632         2          1  0.5192017330
## 86          2          1  0.5187904969
## 374         2          1  0.5163902387
## 312         2          1  0.5159319332
## 166         2          1  0.5153172444
## 317         2          1  0.5148285525
```

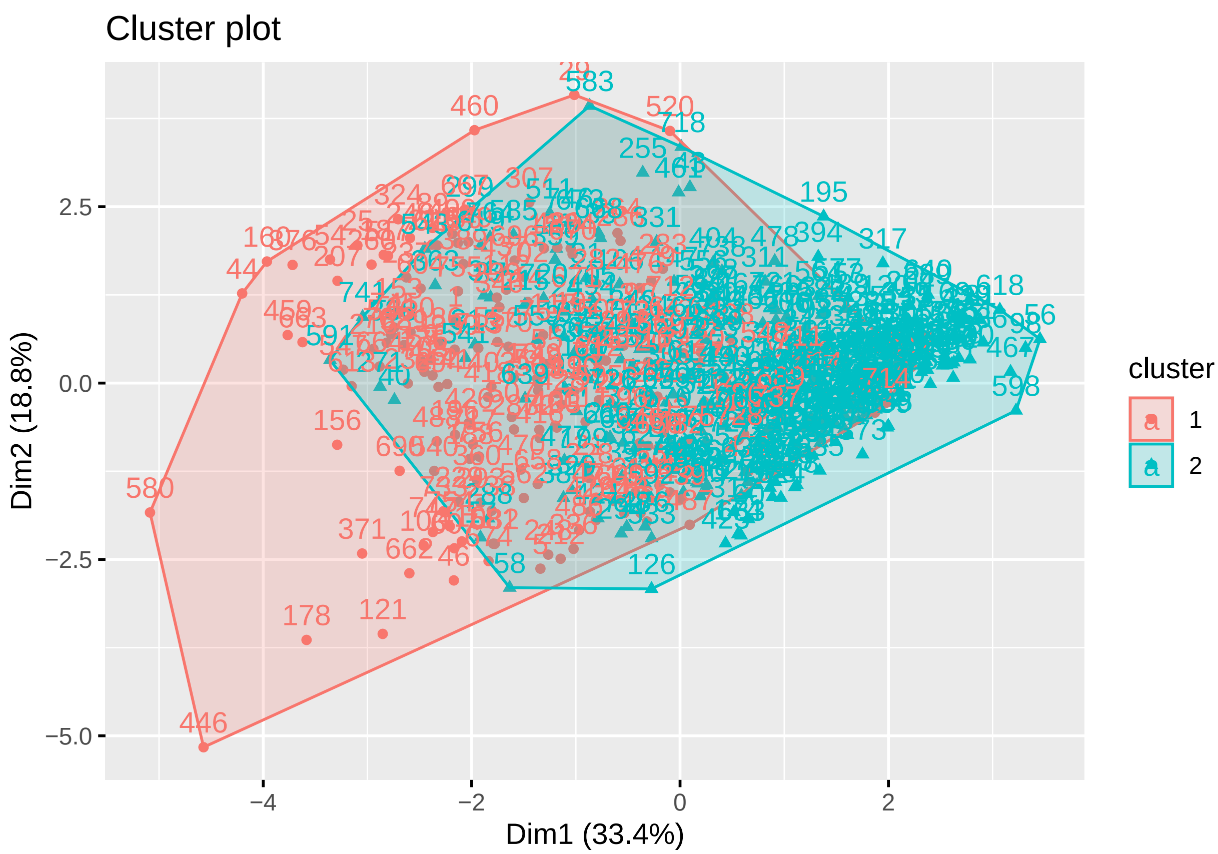
```
## 143         2         1  0.5135369493
## 448         2         1  0.5107926577
## 80          2         1  0.5087374573
## 316         2         1  0.5084579940
## 688         2         1  0.5075119426
## 742         2         1  0.5066391199
## 705         2         1  0.5064524605
## 280         2         1  0.5042183358
## 641         2         1  0.5038817603
## 255         2         1  0.5026911384
## 206         2         1  0.5018372256
## 573         2         1  0.5015749139
## 681         2         1  0.4999300358
## 533         2         1  0.4975118179
## 492         2         1  0.4952196649
## 93          2         1  0.4951942978
## 359         2         1  0.4947084595
## 60          2         1  0.4943625815
## 412         2         1  0.4919906906
## 51          2         1  0.4909565195
## 274         2         1  0.4905849703
## 322         2         1  0.4879716744
## 610         2         1  0.4876845241
## 423         2         1  0.4840145575
## 258         2         1  0.4839705515
## 541         2         1  0.4798588164
## 353         2         1  0.4795505039
## 383         2         1  0.4782316885
## 217         2         1  0.4765237875
## 720         2         1  0.4758653046
## 299         2         1  0.4735114213
## 147         2         1  0.4721848677
## 122         2         1  0.4688235186
## 626         2         1  0.4687987147
## 424         2         1  0.4645979041
## 38          2         1  0.4644751774
## 256         2         1  0.4634858062
## 325         2         1  0.4618951350
## 162         2         1  0.4605117198
## 142         2         1  0.4603609237
## 189         2         1  0.4585198168
## 199         2         1  0.4584514875
## 290         2         1  0.4539845282
## 257         2         1  0.4537262095
## 668         2         1  0.4530347381
## 276         2         1  0.4521852467
## 718         2         1  0.4517609323
## 477         2         1  0.4497872035
## 329         2         1  0.4476447255
## 746         2         1  0.4471667990
## 719         2         1  0.4471039107
## 577         2         1  0.4416835531
## 411         2         1  0.4391605164
## 601         2         1  0.4380305355
```

```
## 20         2        1  0.4353094718
## 559        2        1  0.4348307468
## 551        2        1  0.4338875941
## 722        2        1  0.4295911444
## 543        2        1  0.4289818511
## 443        2        1  0.4286164252
## 652        2        1  0.4252879244
## 201        2        1  0.4241040118
## 478        2        1  0.4226096554
## 271        2        1  0.4221314380
## 314        2        1  0.4205739869
## 250        2        1  0.4188522771
## 598        2        1  0.4175851831
## 529        2        1  0.4171862735
## 315        2        1  0.4128794977
## 528        2        1  0.4111196309
## 600        2        1  0.4097385349
## 205        2        1  0.4069952130
## 445        2        1  0.3991095137
## 319        2        1  0.3969280580
## 81         2        1  0.3961196798
## 67         2        1  0.3932061941
## 387        2        1  0.3924931582
## 673        2        1  0.3923675138
## 215        2        1  0.3907129076
## 380        2        1  0.3892409269
## 727        2        1  0.3879299359
## 126        2        1  0.3815463776
## 182        2        1  0.3811139733
## 128        2        1  0.3703614228
## 394        2        1  0.3684862839
## 170        2        1  0.3677589864
## 473        2        1  0.3661420312
## 666        2        1  0.3625032220
## 726        2        1  0.3624733525
## 19         2        1  0.3595639096
## 386        2        1  0.3594526688
## 163        2        1  0.3567310359
## 665        2        1  0.3561498490
## 244        2        1  0.3459916886
## 43         2        1  0.3425882680
## 450        2        1  0.3410133356
## 621        2        1  0.3328831652
## 531        2        1  0.3305678491
## 592        2        1  0.3245978482
## 87         2        1  0.3237944705
## 619        2        1  0.3203062802
## 766        2        1  0.3202560067
## 58         2        1  0.3093880654
## 570        2        1  0.3085917570
## 31         2        1  0.3068983768
## 230        2        1  0.3067507962
## 764        2        1  0.2993299552
## 127        2        1  0.2974953998
```

```
## 693       2        1  0.2966419478
## 501       2        1  0.2937688054
## 765       2        1  0.2920580001
## 331       2        1  0.2900962233
## 327       2        1  0.2891316631
## 24        2        1  0.2808901666
## 466       2        1  0.2771234375
## 388       2        1  0.2763388418
## 591       2        1  0.2693451406
## 701       2        1  0.2663467403
## 306       2        1  0.2588173619
## 129       2        1  0.2563703342
## 576       2        1  0.2558626959
## 606       2        1  0.2543463537
## 40        2        1  0.2525065272
## 310       2        1  0.2438312213
## 136       2        1  0.2418120920
## 406       2        1  0.2409425905
## 385       2        1  0.2363522100
## 724       2        1  0.2363311890
## 375       2        1  0.2357574289
## 461       2        1  0.2322284061
## 323       2        1  0.2309361016
## 106       2        1  0.2286540118
## 752       2        1  0.2158542363
## 396       2        1  0.2112008834
## 92        2        1  0.2097978014
## 708       2        1  0.2038043673
## 218       2        1  0.2001782293
## 357       2        1  0.1925120874
## 653       2        1  0.1890981541
## 603       2        1  0.1875178949
## 288       2        1  0.1873935001
## 556       2        1  0.1861011013
## 249       2        1  0.1830341946
## 363       2        1  0.1787678949
## 17        2        1  0.1764394322
## 421       2        1  0.1738360306
## 522       2        1  0.1683378968
## 309       2        1  0.1642521639
## 35        2        1  0.1576723897
## 494       2        1  0.1532902744
## 321       2        1  0.1518988105
## 26        2        1  0.1483604559
## 508       2        1  0.1362427295
## 741       2        1  0.1301810740
## 341       2        1  0.1215635294
## 393       2        1  0.1191546608
## 583       2        1  0.1114325051
## 634       2        1  0.1091886013
## 294       2        1  0.1047763836
## 585       2        1  0.0977107762
## 699       2        1  0.0846426200
## Average silhouette width per cluster:
```

```
## [1] 0.3200952 0.4698977
## Average silhouette width of total data set:
## [1] 0.4147073
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```
fviz_cluster(diabetes.pam)
```



For k= 4 clusters:

```
# K-medoids for k=4

diabetes.pam4=pam(datasetdiabetesNoNAs, 4)
diabetes.pam4
```

```
## Medoids:
##      ID Pregnancies Glucose BloodPressure SkinThickness  BMI
## 418 287           4     144            82            32 38.5
## 241 162           1      91            64            24 29.2
## 20   11           1     115            70            30 34.6
## 426 294           4     184            78            39 37.0
##     DiabetesPedigreeFunction Age Outcome
## 418                    0.554  37       2
```

```
## 241                     0.192  21        1
## 20                      0.529  32        2
## 426                     0.264  31        2
## Clustering vector:
##   1   2   4   5   7   9  14  15  17  19  20  21  24  25  26  28  29  31  32  33
##   1   2   2   3   2   4   4   1   3   2   3   1   3   1   3   2   1   3   1   2
##  35  36  38  39  40  41  43  44  46  48  49  51  52  53  54  55  56  57  58  60
##   3   3   3   2   3   4   3   1   4   2   3   2   2   2   4   1   2   4   3   3
##  64  66  67  69  70  71  72  74  75  78  80  81  83  84  86  87  88  89  90  92
##   1   2   3   2   1   2   1   1   2   2   3   2   2   2   3   3   2   1   2   3
##  93  95  96  97  98  99 100 104 106 108 109 110 111 112 113 115 119 120 121 122
##   2   1   1   2   2   2   1   2   3   1   2   2   4   1   2   1   2   2   4   3
## 123 126 127 128 129 131 133 134 135 136 137 138 140 142 143 145 147 148 150 151
##   3   2   3   3   3   4   4   2   2   3   2   2   3   3   2   1   2   3   2   1
## 153 154 156 157 158 159 160 161 162 163 164 166 167 170 172 174 175 176 178 182
##   1   1   1   2   2   2   1   1   3   3   2   3   1   3   1   2   2   4   1   3
## 186 187 188 189 190 192 195 196 198 199 200 201 203 204 205 206 207 209 210 211
##   4   4   1   3   1   3   2   1   2   3   1   3   3   2   3   3   4   2   4   2
## 212 213 214 215 216 217 218 219 221 224 225 226 228 229 230 232 233 235 237 238
##   1   4   1   3   1   3   3   2   4   1   2   2   4   4   3   1   2   2   4   4
## 239 241 242 244 245 246 248 249 250 253 254 255 256 257 258 259 260 261 263 264
##   1   2   2   3   1   4   4   3   3   2   2   2   3   3   3   4   1   4   2   1
## 266 268 271 272 274 276 277 278 280 282 283 286 287 288 289 290 291 292 293 294
##   3   3   3   3   2   3   3   2   2   1   1   1   1   3   2   3   2   3   3   3
## 296 297 298 299 302 303 306 307 308 309 310 311 312 313 314 315 316 317 319 321
##   1   1   3   3   1   2   3   1   3   3   3   2   3   1   3   3   3   2   3   3
## 322 323 324 325 326 327 329 330 331 332 335 336 339 341 342 346 347 349 353 354
##   3   3   1   3   1   3   3   3   3   2   2   4   1   3   2   1   3   2   2   2
## 357 359 360 361 363 365 366 368 369 370 371 373 374 375 376 377 378 380 381 382
##   3   3   4   4   3   1   2   2   2   1   4   2   3   3   1   2   2   3   3   2
## 383 384 385 386 387 388 389 390 391 393 394 396 397 398 400 403 404 406 410 411
##   2   2   3   3   3   3   1   2   3   3   3   3   2   3   4   1   2   3   4   3
## 412 413 414 415 416 417 418 420 421 422 423 424 425 426 428 429 430 432 433 435
##   3   1   1   3   4   2   1   3   3   2   3   3   1   4   4   1   3   2   2   2
## 437 439 441 442 443 445 446 447 448 449 450 451 453 455 456 458 459 460 461 463
##   1   2   4   2   3   3   4   2   3   3   3   2   2   2   4   2   1   1   3   2
## 464 466 467 468 470 471 472 473 476 477 478 479 480 481 482 483 484 486 487 488
##   2   3   2   2   1   1   1   3   1   3   3   3   1   1   3   2   2   1   1   4
## 489 491 492 493 494 498 499 500 501 502 504 505 507 508 509 511 512 515 516 517
##   2   2   2   3   3   2   4   1   3   2   2   3   4   3   2   2   3   2   1   1
## 520 521 522 526 527 528 529 531 533 535 539 540 541 542 543 544 545 546 547 548
##   1   2   3   2   2   3   3   3   2   2   3   1   3   3   3   2   2   4   4   3
## 549 550 551 552 554 555 556 557 559 562 563 564 566 567 568 569 570 573 574 575
##   1   4   3   2   2   2   3   3   3   4   2   2   2   2   2   1   3   3   2   1
## 576 577 580 581 582 583 585 586 589 591 592 594 595 596 598 600 601 603 604 606
##   3   2   4   1   3   3   3   2   4   3   3   2   3   4   2   2   3   3   1   3
## 607 608 609 610 611 612 613 614 615 618 619 621 622 624 626 630 632 634 638 639
##   4   2   1   2   2   4   4   3   1   2   3   3   2   2   2   2   3   3   2   3
## 640 641 645 646 647 648 649 650 651 652 653 655 656 657 658 660 662 663 664 665
##   2   3   3   1   4   4   1   2   2   3   3   3   1   2   3   2   4   4   1   3
## 666 667 668 669 670 671 672 673 674 680 681 682 683 686 688 689 690 693 694 696
##   3   1   3   3   1   1   2   2   1   2   2   1   2   3   2   1   1   3   3   1
## 697 699 701 702 703 705 706 708 710 711 712 713 714 716 717 718 719 720 721 722
##   4   3   3   3   1   3   2   3   2   1   3   3   3   4   4   3   3   3   2   3
```

```
## 723 724 726 727 728 731 733 734 736 737 738 739 741 742 743 745 746 747 748 749
##   1   3   3   3   1   3   4   2   2   3   2   2   3   2   2   1   3   1   2   4
## 752 753 754 755 756 757 761 762 764 765 766 768
##   3   3   4   1   1   1   2   4   3   3   3   2
## Objective function:
##    build     swap
## 21.36960 21.08895
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```r
str(diabetes.pam4)
```

```
## List of 10
##  $ medoids   : num [1:4, 1:8] 4 1 1 4 144 91 115 184 82 64 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:4] "418" "241" "20" "426"
##   .. ..$ : chr [1:8] "Pregnancies" "Glucose" "BloodPressure" "SkinThickness" ...
##  $ id.med    : int [1:4] 287 162 11 294
##  $ clustering: Named int [1:532] 1 2 2 3 2 4 4 1 3 2 ...
##   ..- attr(*, "names")= chr [1:532] "1" "2" "4" "5" ...
##  $ objective : Named num [1:2] 21.4 21.1
##   ..- attr(*, "names")= chr [1:2] "build" "swap"
##  $ isolation : Factor w/ 3 levels "no","L","L*": 1 1 1 1
##   ..- attr(*, "names")= chr [1:4] "1" "2" "3" "4"
##  $ clusinfo  : num [1:4, 1:5] 109 168 194 61 47.8 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:5] "size" "max_diss" "av_diss" "diameter" ...
##  $ silinfo   :List of 3
##   ..$ widths         : num [1:532, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:532] "418" "1" "339" "425" ...
##   .. .. ..$ : chr [1:3] "cluster" "neighbor" "sil_width"
##   ..$ clus.avg.widths: num [1:4] 0.196 0.293 0.172 0.284
##   ..$ avg.width      : num 0.228
##  $ diss      : NULL
##  $ call      : language pam(x = datasetdiabetesNoNAs, k = 4)
##  $ data      : num [1:532, 1:8] 6 1 1 0 3 2 1 5 0 1 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:532] "1" "2" "4" "5" ...
##   .. ..$ : chr [1:8] "Pregnancies" "Glucose" "BloodPressure" "SkinThickness" ...
##  - attr(*, "class")= chr [1:2] "pam" "partition"
```

```r
#Medoids and pam algorithm
as.vector(diabetes.pam4$medoids[1,])
```

```
## [1]   4.000 144.000  82.000  32.000  38.500   0.554  37.000   2.000
```

```r
## Graph for K = 2
summary(diabetes.pam4)
```

```
## Medoids:
##      ID Pregnancies Glucose BloodPressure SkinThickness  BMI
## 418 287           4     144            82            32 38.5
## 241 162           1      91            64            24 29.2
## 20   11           1     115            70            30 34.6
## 426 294           4     184            78            39 37.0
##     DiabetesPedigreeFunction Age Outcome
## 418                    0.554  37       2
## 241                    0.192  21       1
## 20                     0.529  32       2
## 426                    0.264  31       2
## Clustering vector:
##   1   2   4   5   7   9  14  15  17  19  20  21  24  25  26  28  29  31  32  33
##   1   2   2   3   2   4   4   1   3   2   3   1   3   1   3   2   1   3   1   2
##  35  36  38  39  40  41  43  44  46  48  49  51  52  53  54  55  56  57  58  60
##   3   3   3   2   3   4   3   1   4   2   3   2   2   2   4   1   2   4   3   3
##  64  66  67  69  70  71  72  74  75  78  80  81  83  84  86  87  88  89  90  92
##   1   2   3   2   1   2   1   1   2   2   3   2   2   2   3   3   2   1   2   3
##  93  95  96  97  98  99 100 104 106 108 109 110 111 112 113 115 119 120 121 122
##   2   1   1   2   2   2   1   2   3   1   2   2   4   1   2   1   2   2   4   3
## 123 126 127 128 129 131 133 134 135 136 137 138 140 142 143 145 147 148 150 151
##   3   2   3   3   3   4   4   2   2   3   2   2   3   3   2   1   2   3   2   1
## 153 154 156 157 158 159 160 161 162 163 164 166 167 170 172 174 175 176 178 182
##   1   1   1   2   2   2   1   1   3   3   2   3   1   3   1   2   2   4   1   3
## 186 187 188 189 190 192 195 196 198 199 200 201 203 204 205 206 207 209 210 211
##   4   4   1   3   1   3   2   1   2   3   1   3   3   2   3   3   4   2   4   2
## 212 213 214 215 216 217 218 219 221 224 225 226 228 229 230 232 233 235 237 238
##   1   4   1   3   1   3   3   2   4   1   2   2   4   4   3   1   2   2   4   4
## 239 241 242 244 245 246 248 249 250 253 254 255 256 257 258 259 260 261 263 264
##   1   2   2   3   1   4   4   3   3   2   2   2   3   3   3   4   1   4   2   1
## 266 268 271 272 274 276 277 278 280 282 283 286 287 288 289 290 291 292 293 294
##   3   3   3   3   2   3   3   2   2   1   1   1   1   3   2   3   2   3   3   3
## 296 297 298 299 302 303 306 307 308 309 310 311 312 313 314 315 316 317 319 321
##   1   1   3   3   1   2   3   1   3   3   3   2   3   1   3   3   3   2   3   3
## 322 323 324 325 326 327 329 330 331 332 335 336 339 341 342 346 347 349 353 354
##   3   3   1   3   1   3   3   3   3   2   2   4   1   3   2   1   3   2   2   2
## 357 359 360 361 363 365 366 368 369 370 371 373 374 375 376 377 378 380 381 382
##   3   3   4   4   3   1   2   2   2   1   4   2   3   3   1   2   2   3   3   2
## 383 384 385 386 387 388 389 390 391 393 394 396 397 398 400 403 404 406 410 411
##   2   2   3   3   3   3   1   2   3   3   3   3   2   3   4   1   2   3   4   3
## 412 413 414 415 416 417 418 420 421 422 423 424 425 426 428 429 430 432 433 435
##   3   1   1   3   4   2   1   3   3   2   3   3   1   4   4   1   3   2   2   2
## 437 439 441 442 443 445 446 447 448 449 450 451 453 455 456 458 459 460 461 463
##   1   2   4   2   3   3   4   2   3   3   3   2   2   2   4   2   1   1   3   2
## 464 466 467 468 470 471 472 473 476 477 478 479 480 481 482 483 484 486 487 488
##   2   3   2   2   1   1   1   3   1   3   3   3   1   1   3   2   2   1   1   4
## 489 491 492 493 494 498 499 500 501 502 504 505 507 508 509 511 512 515 516 517
##   2   2   2   3   3   2   4   1   3   2   2   3   4   3   2   2   3   2   1   1
## 520 521 522 526 527 528 529 531 533 535 539 540 541 542 543 544 545 546 547 548
##   1   2   3   2   2   3   3   3   2   2   3   1   3   3   3   2   2   4   4   3
## 549 550 551 552 554 555 556 557 559 562 563 564 566 567 568 569 570 573 574 575
##   1   4   3   2   2   2   3   3   3   4   2   2   2   2   2   1   3   3   2   1
## 576 577 580 581 582 583 585 586 589 591 592 594 595 596 598 600 601 603 604 606
##   3   2   4   1   3   3   3   2   4   3   3   2   3   4   2   2   3   3   1   3
```

```
## 607 608 609 610 611 612 613 614 615 618 619 621 622 624 626 630 632 634 638 639
##   4   2   1   2   2   4   4   3   1   2   3   3   2   2   2   2   3   3   2   3
## 640 641 645 646 647 648 649 650 651 652 653 655 656 657 658 660 662 663 664 665
##   2   3   3   1   4   4   1   2   2   3   3   3   1   2   3   2   4   4   1   3
## 666 667 668 669 670 671 672 673 674 680 681 682 683 686 688 689 690 693 694 696
##   3   1   3   3   1   1   2   2   1   2   2   1   2   3   2   1   1   3   3   1
## 697 699 701 702 703 705 706 708 710 711 712 713 714 716 717 718 719 720 721 722
##   4   3   3   3   1   3   2   3   2   1   3   3   3   4   4   3   3   3   2   3
## 723 724 726 727 728 731 733 734 736 737 738 739 741 742 743 745 746 747 748 749
##   1   3   3   3   1   3   4   2   2   3   2   2   3   2   2   1   3   1   2   4
## 752 753 754 755 756 757 761 762 764 765 766 768
##   3   3   4   1   1   1   2   4   3   3   3   2
## Objective function:
##    build     swap
## 21.36960 21.08895
##
## Numerical information per cluster:
##      size max_diss  av_diss diameter separation
## [1,]  109 47.80971 22.76098 79.33763   5.332534
## [2,]  168 55.59405 19.27082 89.42799   4.378139
## [3,]  194 52.57576 20.63621 86.25179   4.378139
## [4,]   61 69.30647 24.54836 95.19087   7.867720
##
## Isolated clusters:
##  L-clusters: character(0)
##  L*-clusters: character(0)
##
## Silhouette plot information:
##     cluster neighbor     sil_width
## 418       1        3  0.3908018456
## 1         1        4  0.3865314721
## 339       1        4  0.3744714434
## 425       1        4  0.3718468870
## 245       1        3  0.3695032457
## 723       1        3  0.3682931311
## 664       1        3  0.3657866303
## 517       1        3  0.3621689619
## 365       1        3  0.3561312938
## 161       1        4  0.3530137352
## 604       1        4  0.3516871045
## 96        1        3  0.3438741532
## 70        1        3  0.3376528106
## 471       1        3  0.3348413439
## 25        1        3  0.3347767069
## 755       1        4  0.3342526776
## 500       1        4  0.3318876926
## 670       1        4  0.3312099926
## 459       1        4  0.3293853484
## 55        1        4  0.3284444730
## 324       1        4  0.3262301217
## 437       1        3  0.3254682201
## 690       1        3  0.3253692839
## 569       1        4  0.3244260260
## 216       1        4  0.3238625972
```

```
## 389       1       3  0.3226039665
## 696       1       3  0.3202134042
## 745       1       4  0.3145023411
## 609       1       4  0.3122115663
## 747       1       4  0.3080852367
## 297       1       3  0.3079568624
## 296       1       4  0.2962989213
## 153       1       4  0.2818908744
## 212       1       3  0.2782518788
## 260       1       4  0.2778070652
## 29        1       3  0.2734853327
## 575       1       3  0.2722780988
## 376       1       3  0.2690517291
## 156       1       4  0.2689418895
## 287       1       4  0.2674541357
## 154       1       4  0.2655344357
## 757       1       3  0.2583768062
## 167       1       3  0.2569711941
## 200       1       3  0.2566338949
## 581       1       4  0.2526770366
## 470       1       4  0.2488836753
## 667       1       3  0.2453999178
## 413       1       3  0.2407303730
## 615       1       3  0.2333330835
## 264       1       3  0.2295879301
## 649       1       3  0.2271897985
## 112       1       4  0.2269204023
## 313       1       4  0.2266646309
## 403       1       3  0.2238447498
## 646       1       4  0.2207380370
## 145       1       4  0.2185648180
## 728       1       3  0.2168127835
## 190       1       3  0.2158735186
## 476       1       3  0.2114273038
## 481       1       4  0.2050136063
## 108       1       3  0.2010076378
## 32        1       4  0.2007145603
## 224       1       3  0.1906343496
## 196       1       4  0.1900994164
## 326       1       4  0.1805904791
## 232       1       3  0.1795374222
## 414       1       3  0.1772584170
## 286       1       3  0.1687541387
## 95        1       3  0.1650760185
## 370       1       3  0.1624525794
## 89        1       3  0.1617837231
## 429       1       3  0.1594216815
## 302       1       3  0.1588966988
## 480       1       3  0.1415736825
## 689       1       3  0.1335032973
## 656       1       4  0.1313117537
## 711       1       4  0.1149795148
## 307       1       4  0.1129848904
## 460       1       3  0.1106108187
```

```
## 214        1        3  0.1091075459
## 72         1        3  0.1000888297
## 151        1        3  0.0935704990
## 283        1        3  0.0849049842
## 64         1        3  0.0758507182
## 487        1        3  0.0724030077
## 472        1        3  0.0718931667
## 115        1        4  0.0710221613
## 178        1        3  0.0626091703
## 540        1        3  0.0566972751
## 188        1        3  0.0453172807
## 160        1        4  0.0430366006
## 520        1        3  0.0404831170
## 239        1        4  0.0403761464
## 549        1        4  0.0301405038
## 486        1        3  0.0297856789
## 516        1        4  0.0250044851
## 346        1        3  0.0222591049
## 682        1        4  0.0171052985
## 756        1        3  0.0160139457
## 671        1        4 -0.0098434555
## 15         1        4 -0.0142284144
## 172        1        3 -0.0154010962
## 674        1        3 -0.0472328031
## 282        1        3 -0.0546161004
## 44         1        4 -0.0567830172
## 703        1        4 -0.0674437946
## 21         1        3 -0.0792338302
## 100        1        3 -0.0830642179
## 74         1        3 -0.0852132896
## 211        2        3  0.4821766365
## 442        2        3  0.4746275544
## 554        2        3  0.4652057332
## 483        2        3  0.4624283104
## 104        2        3  0.4565479751
## 4          2        3  0.4558976046
## 384        2        3  0.4553923892
## 526        2        3  0.4550780169
## 373        2        3  0.4534155314
## 451        2        3  0.4511920737
## 235        2        3  0.4477053759
## 761        2        3  0.4466883985
## 332        2        3  0.4465953079
## 498        2        3  0.4458621278
## 594        2        3  0.4434700143
## 175        2        3  0.4419580658
## 555        2        3  0.4395287733
## 458        2        3  0.4388540969
## 241        2        3  0.4381513180
## 53         2        3  0.4360586741
## 48         2        3  0.4348852434
## 491        2        3  0.4347516242
## 552        2        3  0.4340956180
## 354        2        3  0.4334446948
```

```
## 150         2         3  0.4314676871
## 33          2         3  0.4303706513
## 535         2         3  0.4283904023
## 509         2         3  0.4216347661
## 2           2         3  0.4208907589
## 109         2         3  0.4195716976
## 618         2         3  0.4180415080
## 159         2         3  0.4170369453
## 433         2         3  0.4161803390
## 467         2         3  0.4146017850
## 138         2         3  0.4095690624
## 651         2         3  0.4067476649
## 75          2         3  0.4061170306
## 56          2         3  0.4022675939
## 630         2         3  0.4020422908
## 521         2         3  0.4017294260
## 335         2         3  0.4016431348
## 98          2         3  0.4000686361
## 422         2         3  0.3998800203
## 219         2         3  0.3976126749
## 97          2         3  0.3956830660
## 608         2         3  0.3936394178
## 7           2         3  0.3914893508
## 233         2         3  0.3882990302
## 586         2         3  0.3871212322
## 254         2         3  0.3854308734
## 566         2         3  0.3762733845
## 681         2         3  0.3677840683
## 69          2         3  0.3638182927
## 502         2         3  0.3617162609
## 289         2         3  0.3583786719
## 28          2         3  0.3580458387
## 119         2         3  0.3555692318
## 563         2         3  0.3468519045
## 226         2         3  0.3454575581
## 135         2         3  0.3454388233
## 311         2         3  0.3454306699
## 417         2         3  0.3444790835
## 404         2         3  0.3425959638
## 738         2         3  0.3420875226
## 527         2         3  0.3412023589
## 622         2         3  0.3393572550
## 435         2         3  0.3337612688
## 195         2         3  0.3328829226
## 242         2         3  0.3319441008
## 369         2         3  0.3316517294
## 253         2         3  0.3302902908
## 378         2         3  0.3292950497
## 83          2         3  0.3283983375
## 660         2         3  0.3269484728
## 209         2         3  0.3236910536
## 99          2         3  0.3221975246
## 768         2         3  0.3215284209
## 515         2         3  0.3202976912
```

```
## 574         2         3  0.3198678357
## 672         2         3  0.3190918483
## 349         2         3  0.3185142040
## 638         2         3  0.3156366316
## 157         2         3  0.3156280828
## 439         2         3  0.3155463046
## 545         2         3  0.3149628408
## 432         2         3  0.3097490451
## 113         2         3  0.2975900834
## 680         2         3  0.2971661329
## 710         2         3  0.2953958823
## 174         2         3  0.2941784970
## 225         2         3  0.2930511666
## 453         2         3  0.2925715376
## 484         2         3  0.2886789440
## 463         2         3  0.2881523119
## 721         2         3  0.2868081992
## 739         2         3  0.2854311336
## 291         2         3  0.2850330100
## 368         2         3  0.2838390624
## 164         2         3  0.2788523037
## 706         2         3  0.2785731005
## 204         2         3  0.2768601993
## 353         2         3  0.2744165485
## 147         2         3  0.2721423774
## 564         2         3  0.2717105549
## 52          2         3  0.2682249543
## 736         2         3  0.2612985056
## 134         2         3  0.2600742177
## 263         2         3  0.2548244783
## 489         2         3  0.2530484934
## 464         2         3  0.2510829227
## 274         2         3  0.2483694852
## 120         2         3  0.2403873930
## 342         2         3  0.2387811640
## 624         2         3  0.2385862561
## 447         2         3  0.2372301734
## 71          2         3  0.2363830028
## 303         2         3  0.2362155333
## 255         2         3  0.2331698530
## 598         2         3  0.2328329952
## 544         2         3  0.2312651617
## 511         2         3  0.2208224644
## 748         2         3  0.2189891327
## 640         2         3  0.2188145615
## 390         2         3  0.2088565607
## 84          2         3  0.2088457483
## 504         2         3  0.2083894430
## 366         2         3  0.2081091115
## 742         2         3  0.2069812676
## 137         2         3  0.2045173260
## 39          2         3  0.1982936899
## 377         2         3  0.1969697868
## 455         2         3  0.1955943843
```

```
## 317          2          3  0.1737392405
## 78           2          3  0.1729852022
## 397          2          3  0.1710883847
## 278          2          3  0.1632451666
## 468          2          3  0.1618572959
## 568          2          3  0.1575060995
## 657          2          3  0.1562938768
## 198          2          3  0.1517308804
## 88           2          3  0.1513474312
## 93           2          3  0.1498782999
## 673          2          3  0.1446913752
## 533          2          3  0.1391036411
## 611          2          3  0.1386004339
## 110          2          3  0.1341144473
## 126          2          3  0.1336272930
## 683          2          3  0.1329623114
## 280          2          3  0.1328609177
## 382          2          3  0.1321310898
## 51           2          3  0.1309261256
## 383          2          3  0.1226647576
## 567          2          3  0.1225109312
## 688          2          3  0.1185198354
## 66           2          3  0.1124231931
## 600          2          3  0.1054648785
## 734          2          3  0.0988796062
## 492          2          3  0.0869726728
## 743          2          3  0.0797011467
## 158          2          3  0.0760851022
## 650          2          3  0.0744159613
## 577          2          3  0.0739576217
## 81           2          3  0.0659079261
## 90           2          3  0.0629438171
## 143          2          3  0.0525421325
## 19           2          3  0.0467488587
## 610          2          3  0.0448338522
## 626          2          3  0.0309245341
## 387          3          2  0.3806377980
## 570          3          1  0.3790988136
## 327          3          1  0.3670412900
## 765          3          1  0.3662139832
## 693          3          1  0.3660621173
## 127          3          1  0.3603042977
## 20           3          2  0.3589417386
## 766          3          2  0.3539520611
## 727          3          2  0.3508122631
## 726          3          2  0.3444414718
## 319          3          2  0.3399652848
## 701          3          1  0.3375802377
## 306          3          1  0.3364826454
## 310          3          1  0.3328722517
## 24           3          1  0.3319366132
## 230          3          2  0.3296801083
## 215          3          2  0.3290056691
## 163          3          2  0.3265530205
```

```
## 606        3         1  0.3184404296
## 473        3         2  0.3156808642
## 621        3         2  0.3069536710
## 529        3         2  0.3029762721
## 450        3         2  0.3009549423
## 665        3         2  0.2958397483
## 443        3         2  0.2948217312
## 752        3         1  0.2943091531
## 385        3         1  0.2936811007
## 666        3         2  0.2922483251
## 218        3         1  0.2918275912
## 136        3         1  0.2883611609
## 322        3         2  0.2838069428
## 412        3         2  0.2825603888
## 653        3         1  0.2825572159
## 315        3         2  0.2824222210
## 106        3         1  0.2810046681
## 375        3         2  0.2798375783
## 722        3         2  0.2778402143
## 128        3         2  0.2774853817
## 592        3         2  0.2771516151
## 603        3         1  0.2770807772
## 323        3         1  0.2748387456
## 249        3         1  0.2739873471
## 325        3         2  0.2707994969
## 290        3         2  0.2701781517
## 551        3         2  0.2690949225
## 396        3         1  0.2681257433
## 591        3         1  0.2649091343
## 189        3         2  0.2624880915
## 129        3         1  0.2593200728
## 556        3         1  0.2591393552
## 331        3         2  0.2581093274
## 724        3         1  0.2546837587
## 67         3         2  0.2529789537
## 522        3         1  0.2515161813
## 86         3         2  0.2472450409
## 357        3         1  0.2471479161
## 40         3         1  0.2462595186
## 619        3         2  0.2459569925
## 668        3         2  0.2401890959
## 652        3         2  0.2391347024
## 256        3         2  0.2383329854
## 122        3         2  0.2364297805
## 595        3         1  0.2342572634
## 406        3         2  0.2342054093
## 182        3         2  0.2339859998
## 309        3         1  0.2304672729
## 531        3         2  0.2302253180
## 142        3         2  0.2283464246
## 92         3         1  0.2280358524
## 461        3         1  0.2265699397
## 288        3         1  0.2262091095
## 542        3         1  0.2149435764
```

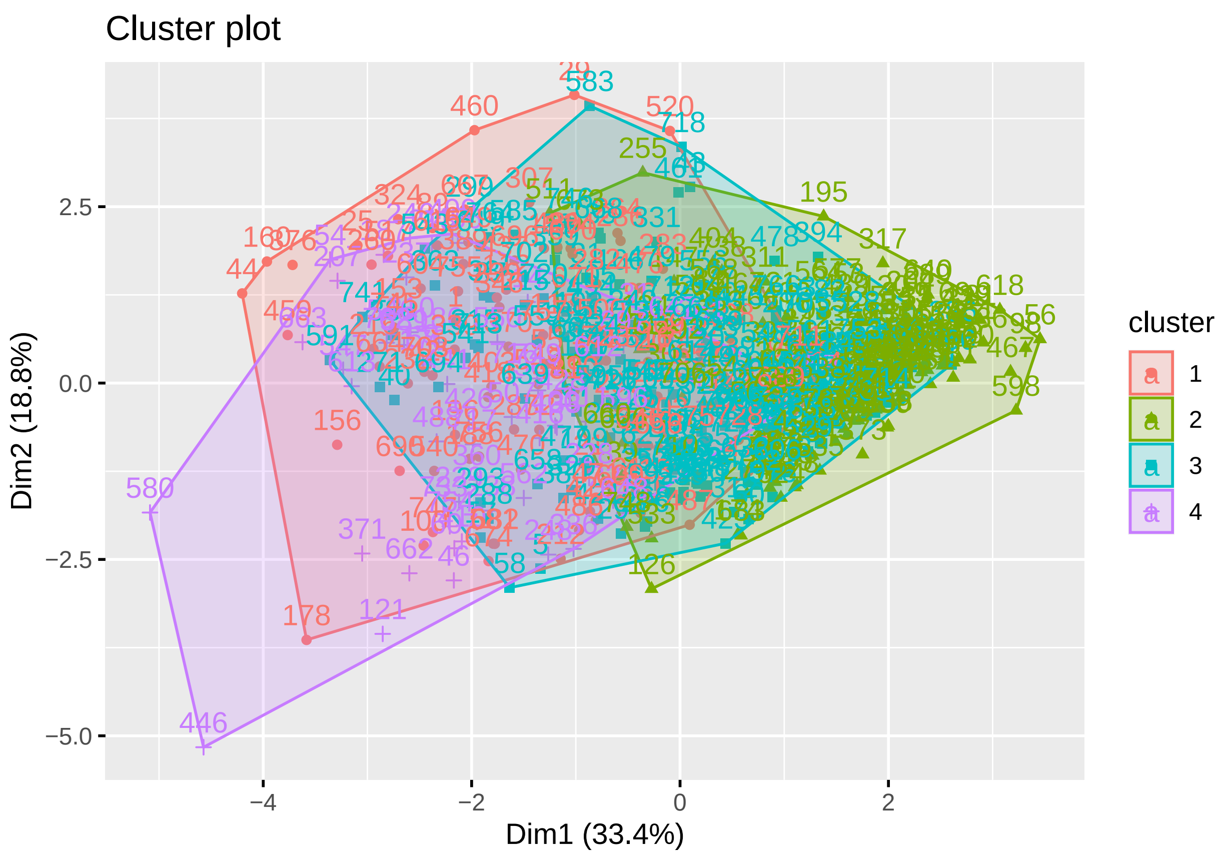
```
## 199         3          2  0.2142340959
## 17          3          1  0.2135385942
## 420         3          1  0.2106468572
## 501         3          2  0.2103273550
## 258         3          2  0.2075820189
## 508         3          1  0.2073685178
## 477         3          2  0.2070838616
## 192         3          1  0.2070485755
## 421         3          1  0.2058208932
## 26          3          1  0.2052797899
## 87          3          2  0.2050771859
## 206         3          2  0.1970292681
## 478         3          2  0.1949822152
## 35          3          1  0.1936623267
## 217         3          2  0.1936233816
## 31          3          2  0.1935119007
## 708         3          2  0.1918899773
## 316         3          2  0.1894567923
## 388         3          2  0.1884211437
## 528         3          2  0.1836771734
## 658         3          1  0.1835265638
## 321         3          1  0.1820970884
## 424         3          2  0.1783141054
## 268         3          1  0.1779793265
## 576         3          2  0.1766348746
## 257         3          2  0.1765435924
## 494         3          1  0.1764556553
## 686         3          1  0.1739573693
## 298         3          1  0.1729557184
## 162         3          2  0.1716753432
## 394         3          2  0.1712930722
## 294         3          1  0.1698975461
## 482         3          1  0.1663038205
## 393         3          1  0.1625766012
## 244         3          2  0.1624774505
## 341         3          1  0.1623802853
## 38          3          2  0.1607239398
## 737         3          1  0.1599859302
## 250         3          2  0.1592534566
## 741         3          1  0.1577947724
## 271         3          2  0.1569172899
## 559         3          2  0.1568101184
## 719         3          2  0.1540171275
## 201         3          2  0.1513637249
## 330         3          2  0.1512564160
## 539         3          1  0.1502864658
## 712         3          1  0.1480345486
## 479         3          1  0.1478321541
## 445         3          2  0.1476451639
## 205         3          2  0.1469908917
## 466         3          2  0.1459568610
## 123         3          2  0.1443352319
## 705         3          2  0.1439848112
## 634         3          1  0.1439750845
```

```
## 312        3         2  0.1435532471
## 411        3         2  0.1433427901
## 548        3         1  0.1372760707
## 764        3         2  0.1365674671
## 541        3         2  0.1331428158
## 58         3         2  0.1322060446
## 148        3         2  0.1318062492
## 170        3         2  0.1285772518
## 363        3         1  0.1232519918
## 381        3         2  0.1229200221
## 714        3         1  0.1203243262
## 398        3         1  0.1189928164
## 43         3         2  0.1176353823
## 614        3         2  0.1170046342
## 746        3         2  0.1139697441
## 329        3         2  0.1132372279
## 140        3         2  0.1122593033
## 585        3         1  0.1086728399
## 713        3         1  0.1055820513
## 80         3         2  0.1052632201
## 573        3         2  0.1045436076
## 60         3         2  0.1015160385
## 601        3         2  0.0971526237
## 632        3         2  0.0955453421
## 702        3         1  0.0918303310
## 699        3         1  0.0905367271
## 293        3         1  0.0816890099
## 583        3         1  0.0808267038
## 276        3         2  0.0803367964
## 386        3         2  0.0781495335
## 731        3         1  0.0712528354
## 299        3         2  0.0697503228
## 292        3         2  0.0658322362
## 49         3         2  0.0552977156
## 374        3         2  0.0547404894
## 655        3         2  0.0508876307
## 423        3         2  0.0495495849
## 347        3         1  0.0426402397
## 203        3         2  0.0412774402
## 380        3         2  0.0386244974
## 582        3         2  0.0375393154
## 505        3         2  0.0361780530
## 166        3         2  0.0357927817
## 5          3         1  0.0293957054
## 272        3         2  0.0290355055
## 720        3         2  0.0265177563
## 694        3         1  0.0221228910
## 449        3         2  0.0210586143
## 308        3         1  0.0171813144
## 512        3         1 -0.0005624917
## 543        3         2 -0.0128314442
## 645        3         2 -0.0176119819
## 448        3         2 -0.0184549389
## 493        3         2 -0.0193618945
```

```
## 753        3        2 -0.0285253216
## 415        3        1 -0.0293975095
## 391        3        2 -0.0333046015
## 639        3        2 -0.0388982141
## 641        3        2 -0.0417757155
## 557        3        2 -0.0453541340
## 314        3        2 -0.0456059752
## 36         3        2 -0.0543842811
## 430        3        2 -0.0555976778
## 277        3        2 -0.0711235960
## 669        3        2 -0.0758661633
## 359        3        2 -0.0853872864
## 718        3        2 -0.0893093229
## 266        3        2 -0.1322367518
## 360        4        1  0.4802540878
## 229        4        1  0.4773573451
## 400        4        1  0.4669340756
## 186        4        1  0.4664219762
## 562        4        1  0.4625250351
## 361        4        1  0.4481232642
## 57         4        1  0.4427051878
## 426        4        1  0.4404504953
## 749        4        1  0.4386295992
## 662        4        1  0.4311890924
## 210        4        1  0.4087521812
## 261        4        1  0.4000037413
## 428        4        1  0.3953491649
## 9          4        1  0.3936697697
## 207        4        1  0.3923852304
## 546        4        1  0.3903336595
## 499        4        1  0.3834489564
## 547        4        1  0.3751025887
## 46         4        1  0.3713071250
## 607        4        1  0.3689829430
## 176        4        1  0.3636647233
## 41         4        1  0.3607961421
## 716        4        1  0.3598886816
## 596        4        1  0.3396888663
## 754        4        1  0.3295323230
## 507        4        1  0.3257506336
## 259        4        1  0.3199781953
## 14         4        1  0.3139310889
## 237        4        1  0.2984133271
## 221        4        1  0.2974735687
## 441        4        1  0.2928487059
## 246        4        1  0.2912773080
## 238        4        1  0.2871255077
## 648        4        1  0.2805143106
## 717        4        1  0.2690074071
## 187        4        1  0.2683464130
## 456        4        1  0.2621442753
## 550        4        1  0.2532747095
## 612        4        1  0.2400721527
## 733        4        1  0.2376230192
```

```
## 416       4        1  0.2307880174
## 589       4        1  0.2299595187
## 371       4        1  0.2241312811
## 410       4        1  0.2235364672
## 111       4        1  0.2186823052
## 446       4        1  0.2102513863
## 133       4        1  0.2023235739
## 131       4        1  0.1963673368
## 213       4        1  0.1952225958
## 54        4        1  0.1834357016
## 488       4        1  0.1648187236
## 762       4        1  0.1620987298
## 580       4        1  0.1551503451
## 697       4        1  0.1227568984
## 613       4        1  0.0764160724
## 336       4        1  0.0565414979
## 647       4        1  0.0454970934
## 248       4        1  0.0200554267
## 228       4        1  0.0022720182
## 663       4        1 -0.0004721703
## 121       4        1 -0.0068171513
## Average silhouette width per cluster:
## [1] 0.1960544 0.2929732 0.1720970 0.2842343
## Average silhouette width of total data set:
## [1] 0.2280349
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```r
fviz_cluster(diabetes.pam4)
```



16.

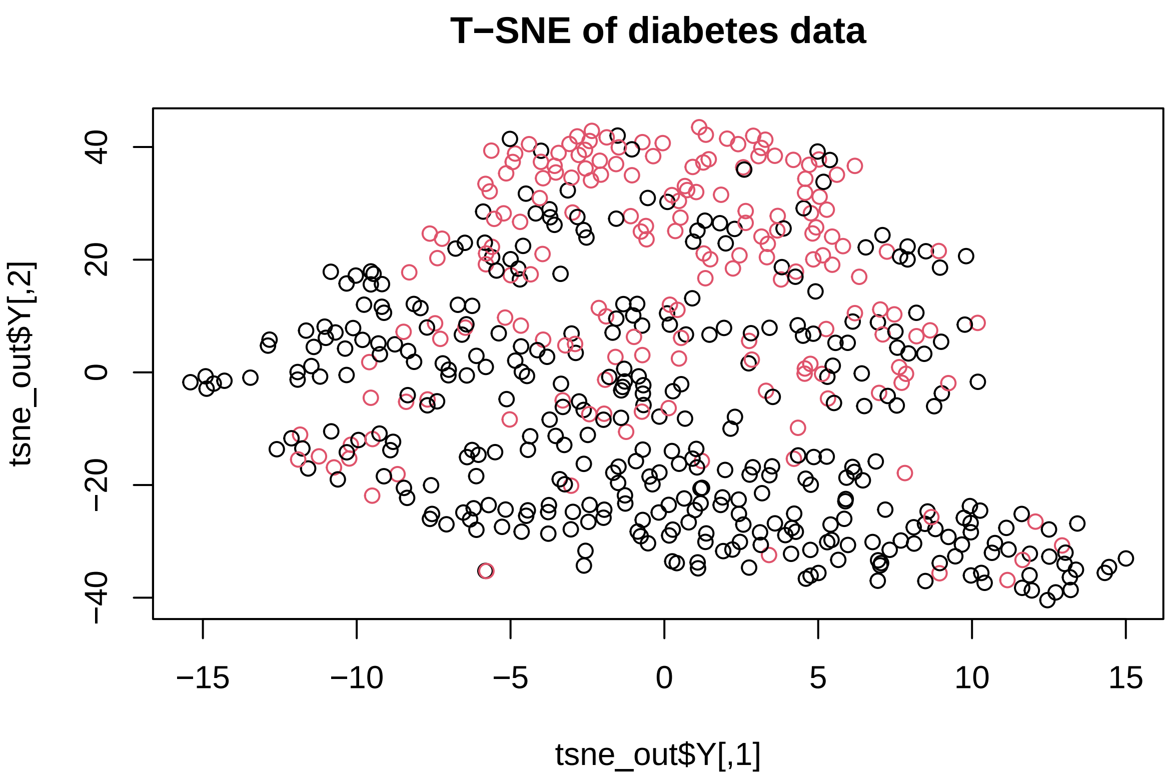
```r
#install.packages("Rtsne")
library('Rtsne')
diabetes_unique = unique(datasetdiabetesNoNAs)
diabetes_matrix = as.matrix(diabetes_unique[,1:7])
print( head( diabetes_matrix ) )
```

```
##   Pregnancies Glucose BloodPressure SkinThickness  BMI DiabetesPedigreeFunction
## 1           6     148            72            35 33.6                    0.627
## 2           1      85            66            29 26.6                    0.351
## 4           1      89            66            23 28.1                    0.167
## 5           0     137            40            35 43.1                    2.288
## 7           3      78            50            32 31.0                    0.248
## 9           2     197            70            45 30.5                    0.158
##   Age
## 1  50
## 2  31
## 4  21
## 5  33
## 7  26
## 9  53
```

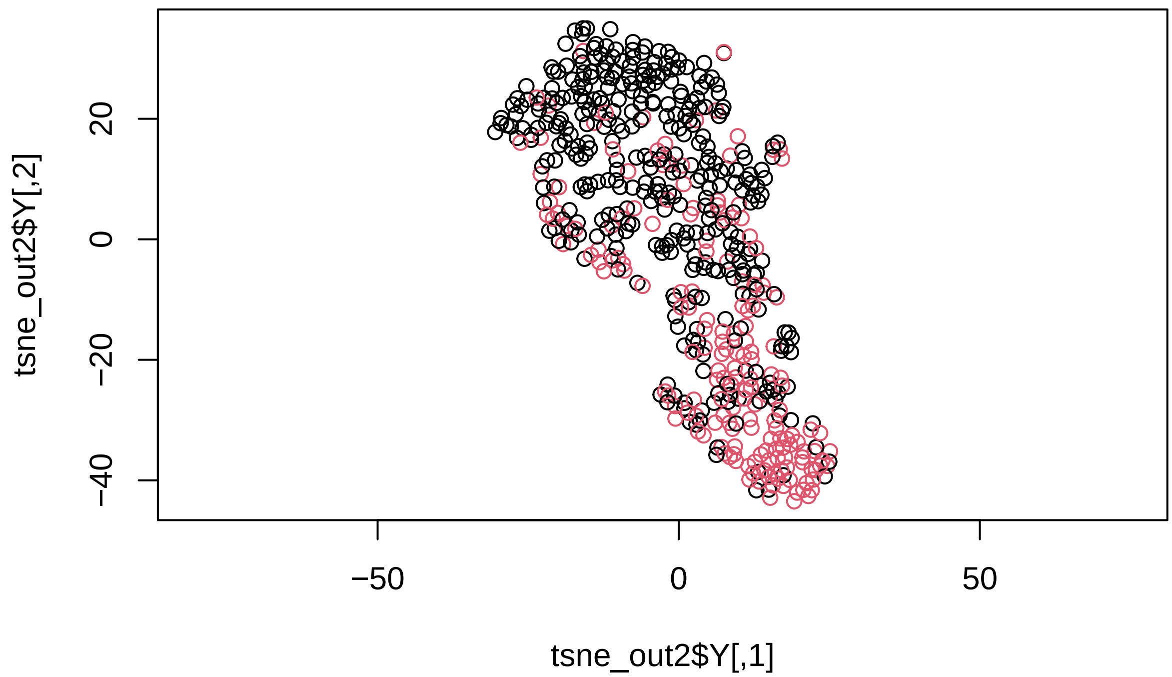
```r
tsne_out = Rtsne(diabetes_unique,pca=TRUE,perplexity=30,theta=0.2)
names(tsne_out)
```

```
##  [1] "N"                   "Y"                     "costs"
##  [4] "itercosts"           "origD"                 "perplexity"
##  [7] "theta"               "max_iter"              "stop_lying_iter"
## [10] "mom_switch_iter"     "momentum"              "final_momentum"
## [13] "eta"                 "exaggeration_factor"
```

```r
plot(tsne_out$Y,col=diabetes_unique$Outcome,main='T-SNE of diabetes data')
```




```r
tsne_out2 = Rtsne(diabetes_unique,pca=FALSE, theta=0.0)

tsne_out2 = Rtsne(dist(normalize_input(diabetes_matrix)), theta=0.0)
plot(tsne_out2$Y,col=diabetes_unique$Outcome, asp=1)
```

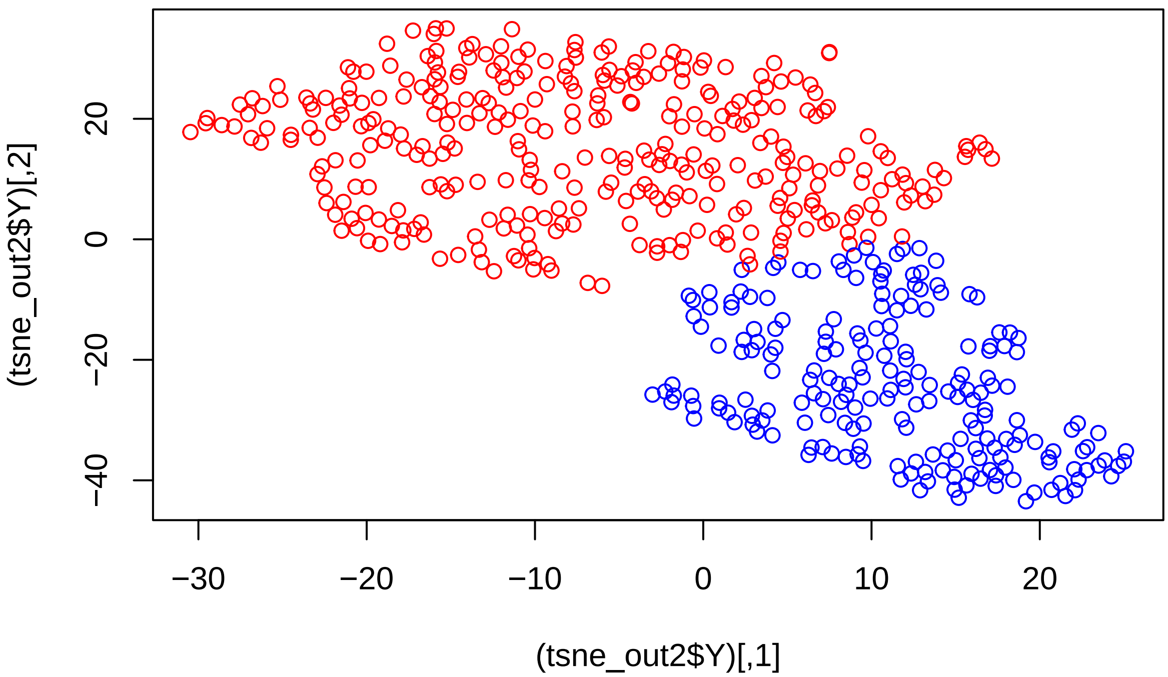


With K -Means

```r
tk = kmeans((tsne_out2$Y),2)
tk$cluster
```

```
##   [1] 2 1 1 1 1 2 2 2 2 1 1 2 2 2 2 1 2 1 2 1 2 1 1 1 1 2 1 2 2 1 1 1 1 1 2 2 1
##  [38] 2 1 1 2 1 1 1 2 1 2 2 1 1 1 1 1 1 1 1 1 2 1 1 1 2 2 1 1 1 2 1 1 2 1 1 2 2
##  [75] 1 2 1 1 2 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 2 1 1 1 2 2 2 2 1 1 1 2 2 1 2 1
## [112] 1 2 1 1 1 1 2 2 1 2 2 2 1 2 2 1 2 1 1 2 1 1 1 1 1 2 1 2 1 2 2 2 1 2 1 1 1
## [149] 2 2 1 1 2 2 2 2 1 1 2 2 2 1 1 1 2 2 2 1 1 1 1 1 1 1 1 2 2 2 1 2 1 2 1 1 1
## [186] 1 1 1 1 2 2 2 2 2 1 1 1 1 2 1 2 2 2 1 2 1 2 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1
## [223] 2 1 2 1 1 1 2 1 1 2 2 1 1 2 1 1 1 1 1 1 2 2 1 2 1 1 1 2 2 1 1 1 2 1 1 1 1
## [260] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 2 2 2 1 1 2 1 1 2 2 2 2 1 2 1 2 1 1 1 2 2 2 2
## [297] 1 1 1 1 2 1 2 1 1 1 2 1 1 1 1 1 1 1 2 1 2 2 2 1 1 1 1 1 2 2 2 1 2 1 1 2 2
## [334] 2 2 1 1 2 2 2 1 1 1 1 2 1 2 2 1 1 1 1 2 1 1 1 1 1 2 2 2 1 2 1 1 1 1 1 1 1
## [371] 2 2 1 1 1 1 1 2 2 1 2 2 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 2 1 1 2 2 1 2 2
## [408] 1 2 1 1 1 2 2 1 1 1 2 2 1 2 1 2 1 1 2 2 1 2 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2
## [445] 2 2 2 1 1 1 2 1 2 1 2 1 2 2 2 1 1 2 1 1 2 2 1 1 2 1 1 2 1 1 1 2 2 1 2 2 2
## [482] 1 1 2 2 1 1 1 1 2 2 1 1 2 2 1 1 1 1 1 2 1 1 1 2 1 2 1 1 2 1 1 1 1 1 2 1 2
## [519] 1 2 2 1 2 2 2 2 1 2 1 1 1 1
```

```r
colors = c("red","blue")
colors = colors[tk$cluster]
plot( (tsne_out2$Y), col=colors )
```



17.

```
#library(cluster.stats)

#clustering vector:
clustering_vector <- as.data.frame(diabetes.pam$clustering)

#silhouette score
silhouette_scores <- as.data.frame(diabetes.pam$silinfo$widths)
head(silhouette_scores)
```

```
##     cluster neighbor sil_width
## 717       1        2 0.5444252
## 762       1        2 0.5327901
## 210       1        2 0.5317113
## 176       1        2 0.5310542
## 426       1        2 0.5307788
## 613       1        2 0.5279580
```

```
names(silhouette_scores)
```

```
## [1] "cluster"   "neighbor"  "sil_width"
```

```r
#most ambiguous data
y=min(silhouette_scores$sil_width)
y
```

```
## [1] -0.1537411
```

```r
#row with minimum value for silhouette score
row_with_minvalue <- which(silhouette_scores$sil_width == y)
row_with_minvalue
```

```
## [1] 196
```

```r
myrow=row_with_minvalue[[1]][1]
myrow
```

```
## [1] 196
```

```r
silohouette_scores_myrow <- as.data.frame(diabetes.pam$silinfo)[myrow,]
silohouette_scores_myrow #Belongs to cluster 1
```

```
##     widths.cluster widths.neighbor widths.sil_width clus.avg.widths avg.width
## 595              1               2       -0.1537411       0.4698977 0.4147073
```

```r
target_row_index <- which(rownames(clustering_vector) == myrow)
target_row_index
```

```
## [1] 128
```

```r
#select the row

prob <- predict(newmodel, newdata=datasetdiabetesNoNAs[target_row_index,], type="response")
prob
```

```
##       196
## 0.6541096
```

```r
#prob <- predict(newmodel, newdata=datasetdiabetesNoNAs, type="response")
#prob

#Real Outcome
datasetdiabetesNoNAs[target_row_index,]
```

```
##     Pregnancies Glucose BloodPressure SkinThickness  BMI
## 196           5     158            84            41 39.4
##     DiabetesPedigreeFunction Age Outcome
## 196                    0.395  29       1
```

I looked for the row that had the lowest silohouette score, which means that point makes a better fit in the other cluster and the average of distances between the other cluster and that point, is smaller that the average of distances between that point and all the other points on its cluster. The most ambiguous data is the row 128 of the datasetdiabetesNoNas.