```
library(corrplot)
```

```
## corrplot 0.90 loaded
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.1.2
```

```
library(e1071)
library(randomForest)
```

```
## randomForest 4.6-14
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
library(partykit)
```

```
## Warning: package 'partykit' was built under R version 4.1.2
```

```
## Loading required package: grid
```

```
## Loading required package: libcoin
```

```
## Warning: package 'libcoin' was built under R version 4.1.2
```

```
## Loading required package: mvtnorm
```

```
library(tidyverse)
```

```
## Registered S3 method overwritten by 'cli':
##   method     from
##   print.tree tree
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::combine()       masks randomForest::combine()
## x dplyr::filter()        masks stats::filter()
## x dplyr::lag()           masks stats::lag()
## x purrr::lift()          masks caret::lift()
## x randomForest::margin() masks ggplot2::margin()
```

```
library(ggplot2)
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.1.2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(dplyr)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

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

```
## The following object is masked from 'package:randomForest':
##
##     combine
```

```
library(rpart.plot)
```

```
## Loading required package: rpart
```

```
library(e1071)
library(mice)
```

```
## Warning: package 'mice' was built under R version 4.1.2
```

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

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

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

```
library(caTools)
```

```
## Warning: package 'caTools' was built under R version 4.1.2
```

```
library(pROC)
```

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

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

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

```
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.1.2
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following object is masked from 'package:e1071':
##
##     impute
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
data <- read.csv('D:/MS Materials/Books - Learning Materials/Lecture Notes/Sem 2/DPA/Project/dia
betes.csv')
```

```
str(data)
```

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

Variable Description

pregnant : Number of times pregnant glucose : Plasma glucose concentration (glucose tolerance test) triceps : Triceps skin fold thickness (mm Hg) insulin : 2-hour serum insulin (mu U/ml) mass : Body mass index (weight in kg/(height in m)^2) pedigree : Diabetes pedigree function age : Age (years) diabetes : Test for diabetes

We inspect whether there is any missing value of our observation

```
colSums(is.na(data))
```

```
##              Pregnancies                  Glucose            BloodPressure
##                        0                        0                        0
##            SkinThickness                  Insulin                      BMI
##                        0                        0                        0
## DiabetesPedigreeFunction                      Age                  Outcome
##                        0                        0                        0
```

There is no missing data of our dataframe so we could proceed to the next step.

Basic Exploratory Data Analysis

```
data$Outcome <- factor(make.names(data$Outcome))
biological_data <- data[,setdiff(names(data), c('Outcome', 'Pregnancies'))]
features_miss_num <- apply(biological_data, 2, function(x) sum(x<=0))
features_miss <- names(biological_data)[ features_miss_num > 0]
features_miss_num
```

```
##                  Glucose            BloodPressure            SkinThickness
##                        5                       35                      227
##                  Insulin                      BMI DiabetesPedigreeFunction
##                      374                       11                        0
##                      Age
##                        0
```

```
rows_errors <- apply(biological_data, 1, function(x) sum(x<=0)>1)
sum(rows_errors)
```

```
## [1] 234
```

```
sum(rows_errors)/nrow(data)
```

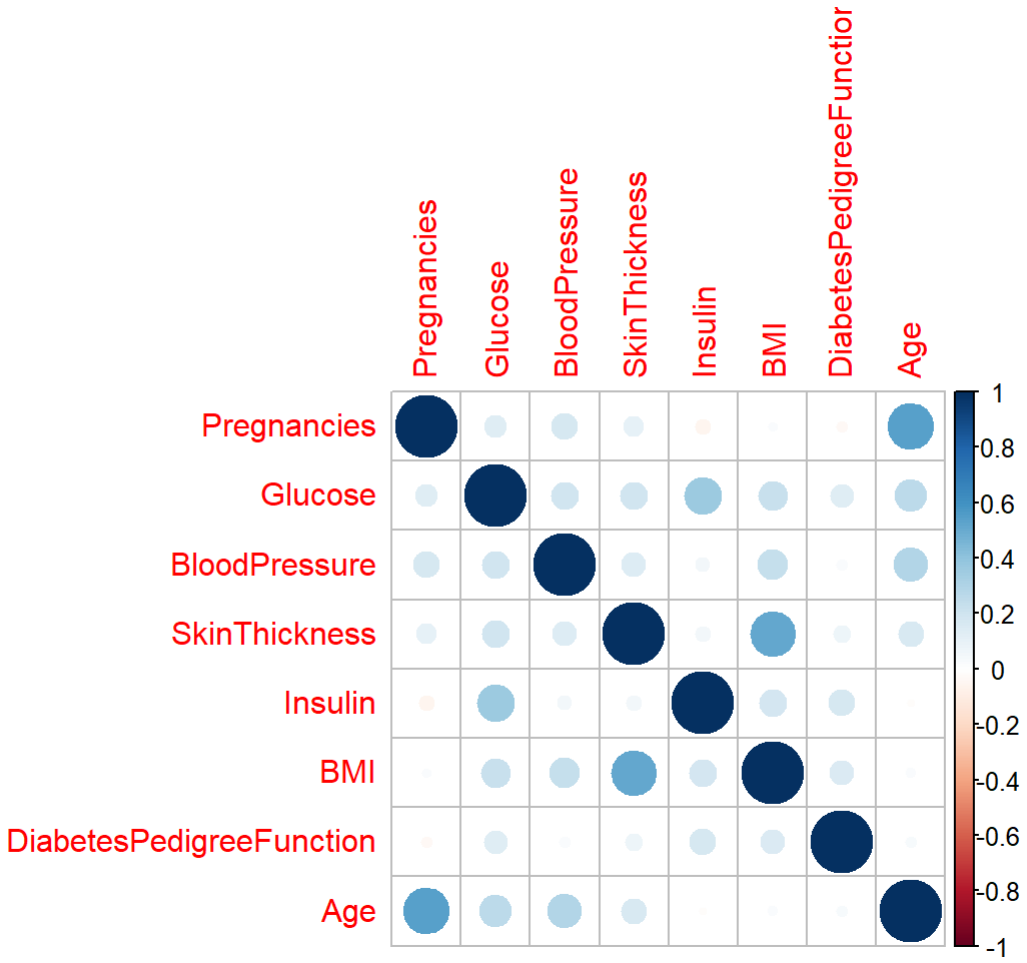
```
## [1] 0.3046875
```

```
biological_data[biological_data<=0] <- NA
data[, names(biological_data)] <- biological_data
data_original <- data
data[,-9] <- with(data[,-9], impute(data[,-9],fun=median))
```

```
prop.table(table(data$Outcome))
```

```
##
##        X0        X1
## 0.6510417 0.3489583
```

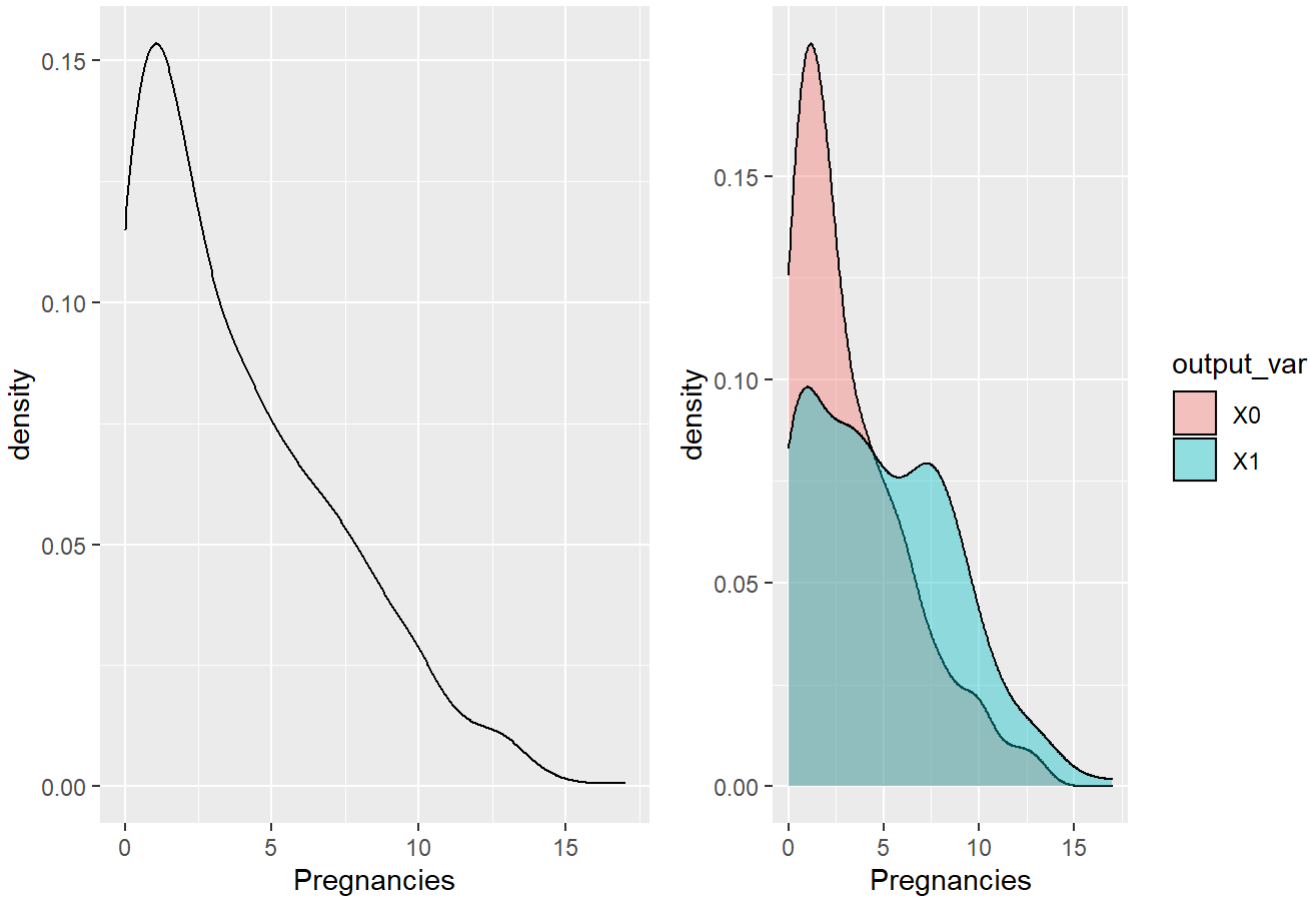
```
correlat <- cor(data[, setdiff(names(data), 'Outcome')])
corrplot(correlat)
```



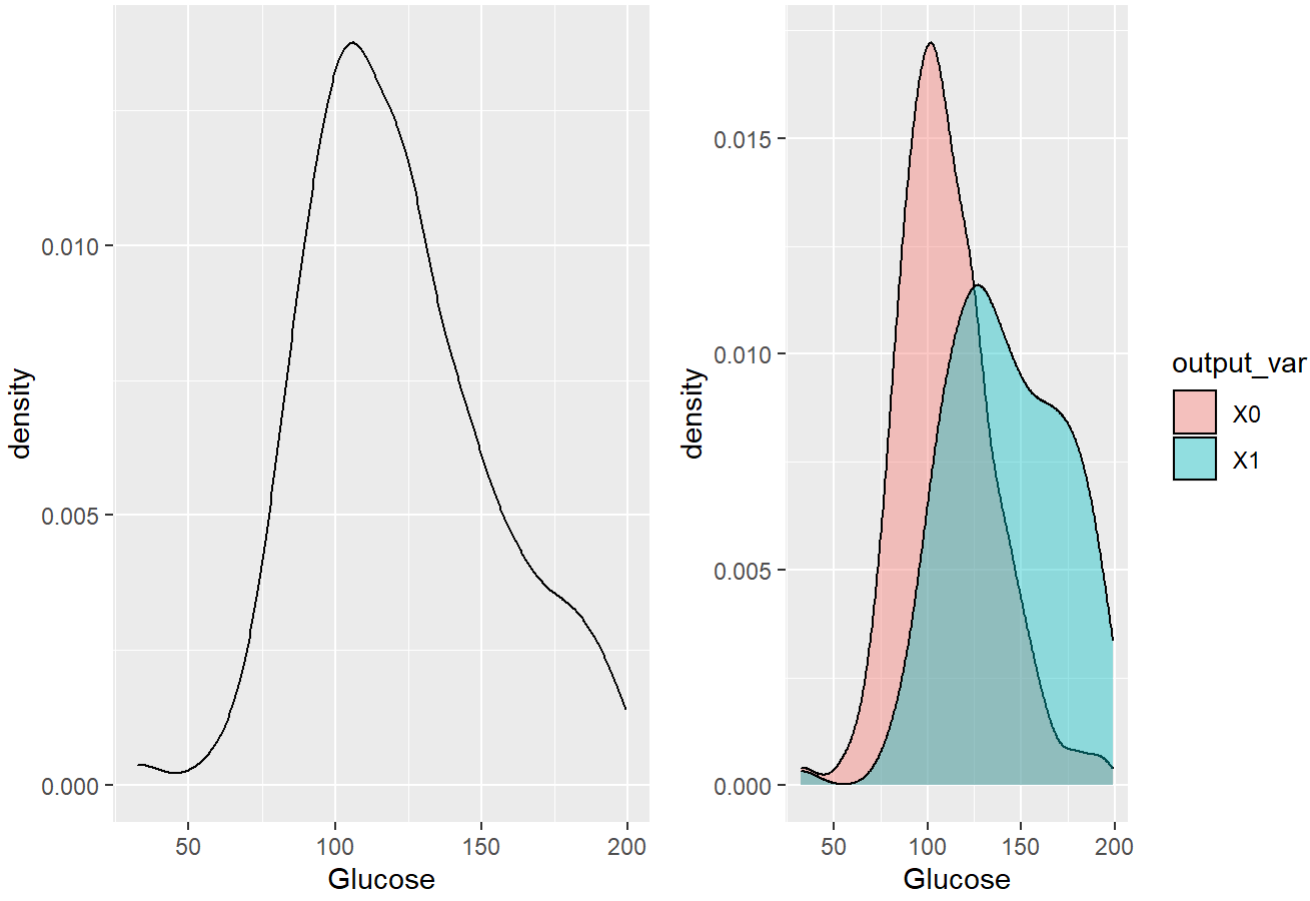

```
univar_graph <- function(univar_name, univar, data, output_var) {
  g_1 <- ggplot(data, aes(x=univar)) + geom_density() + xlab(univar_name)
  g_2 <- ggplot(data, aes(x=univar, fill=output_var)) + geom_density(alpha=0.4) + xlab(univar_na
me)
  grid.arrange(g_1, g_2, ncol=2, top=paste(univar_name,"variable", "/ [ Skew:",skewness(univar),
"]"))
}

for (x in 1:(ncol(data)-1)) {
  univar_graph(names(data)[x], data[,x], data, data[,'Outcome'])
}
```

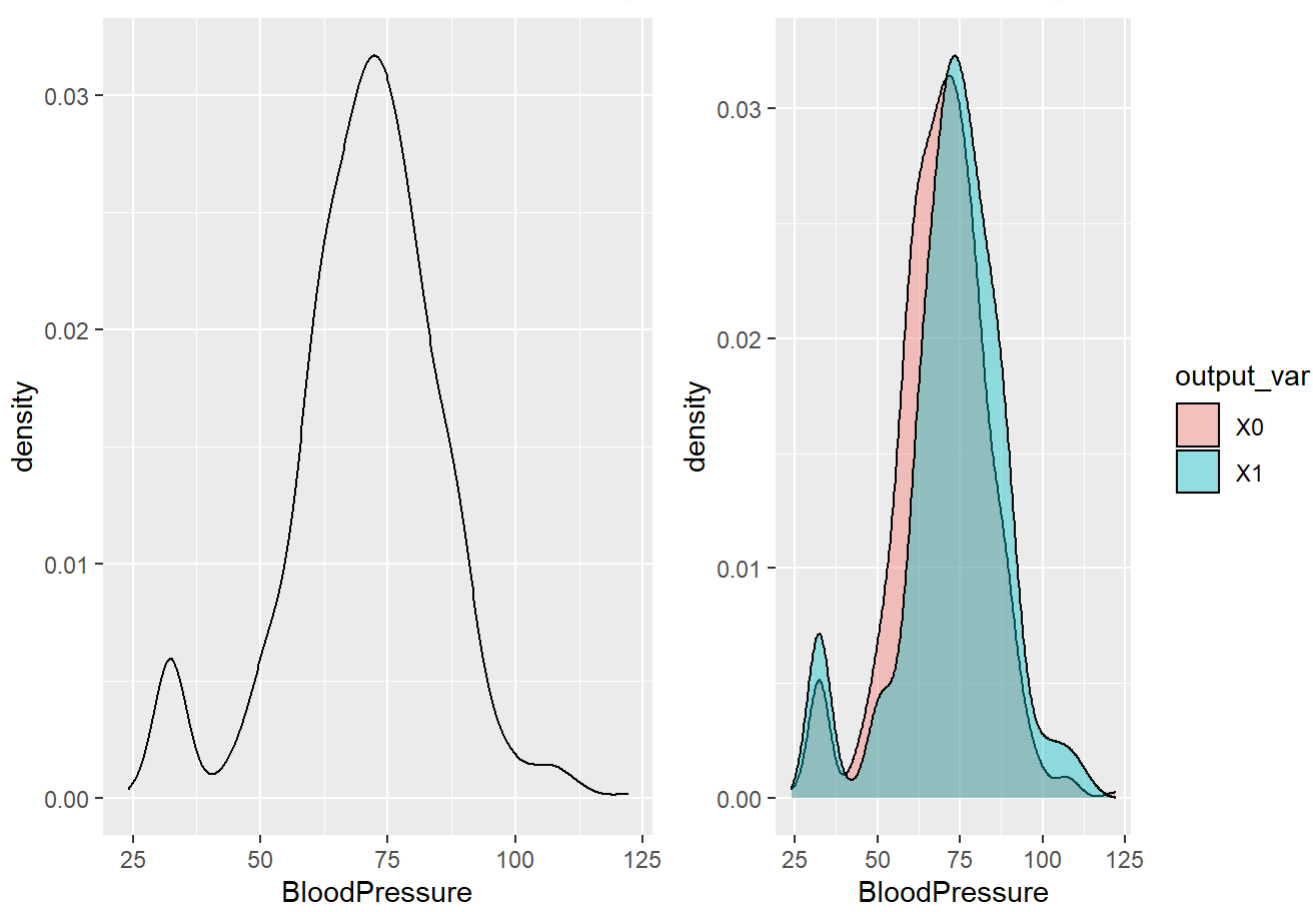
Pregnancies variable / [Skew: 0.898154872604808]



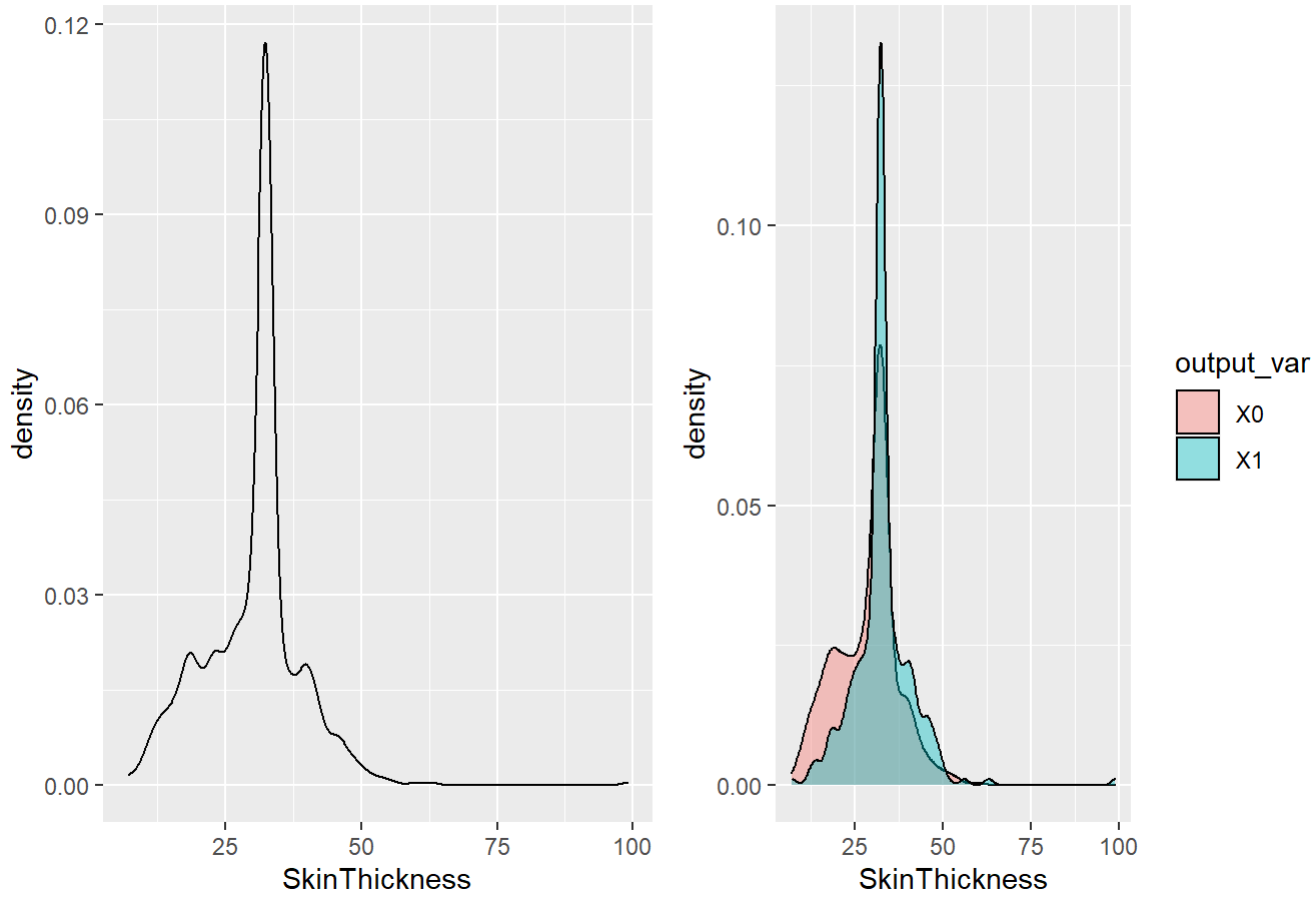
Glucose variable / [Skew: 0.393352849269315]



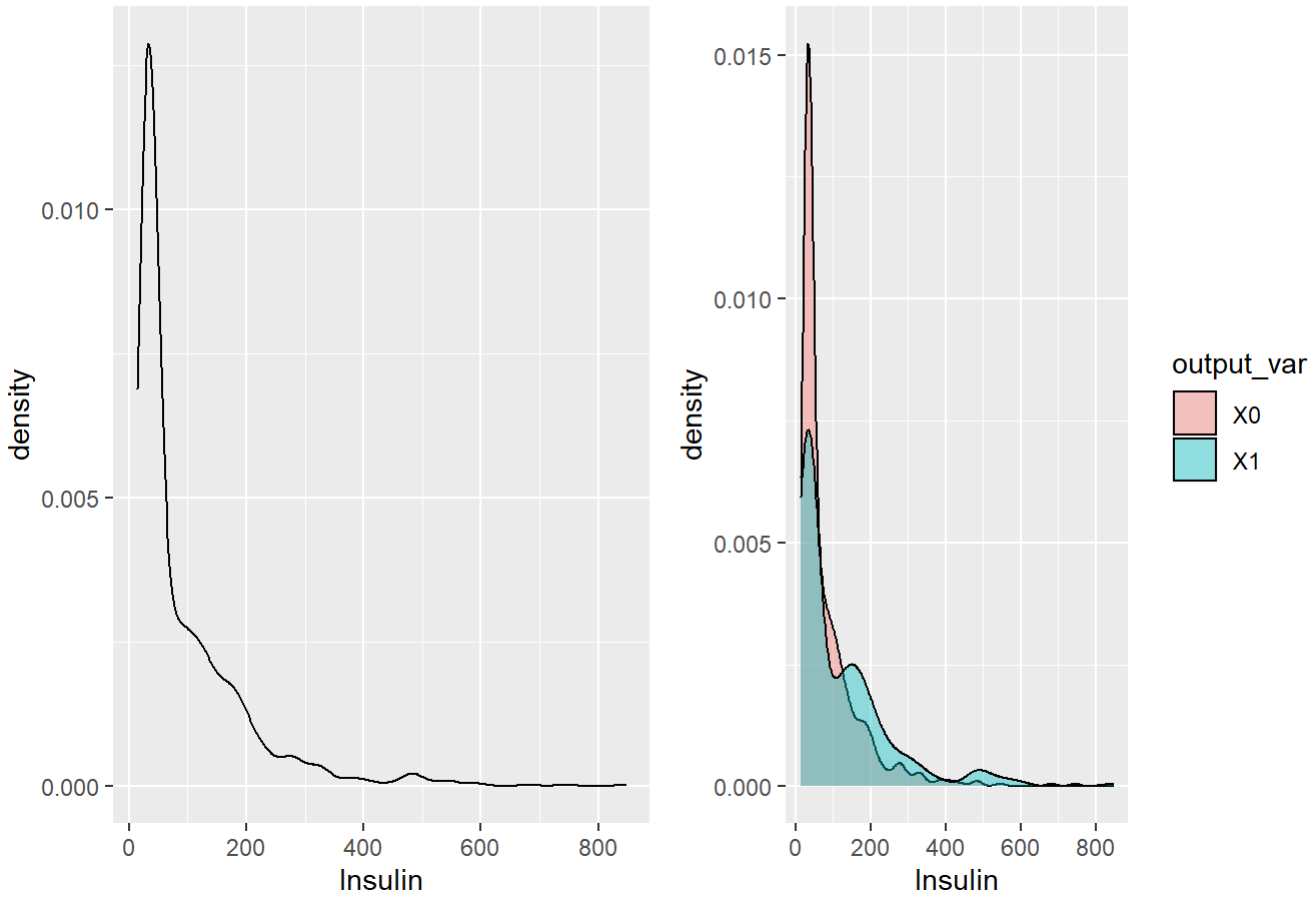
BloodPressure variable / [Skew: -0.469176890812277]



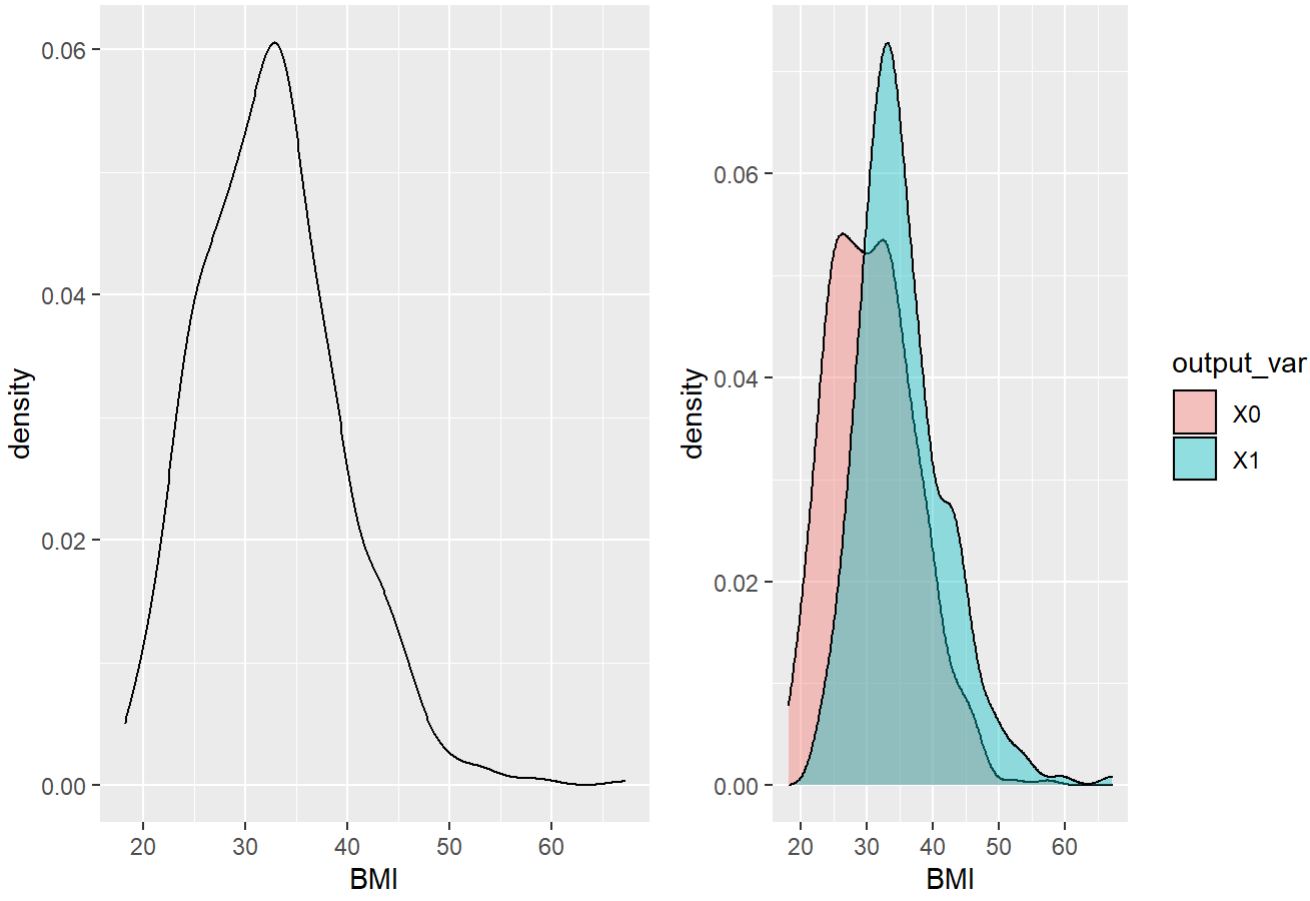
SkinThickness variable / [Skew: 0.475768658146166]



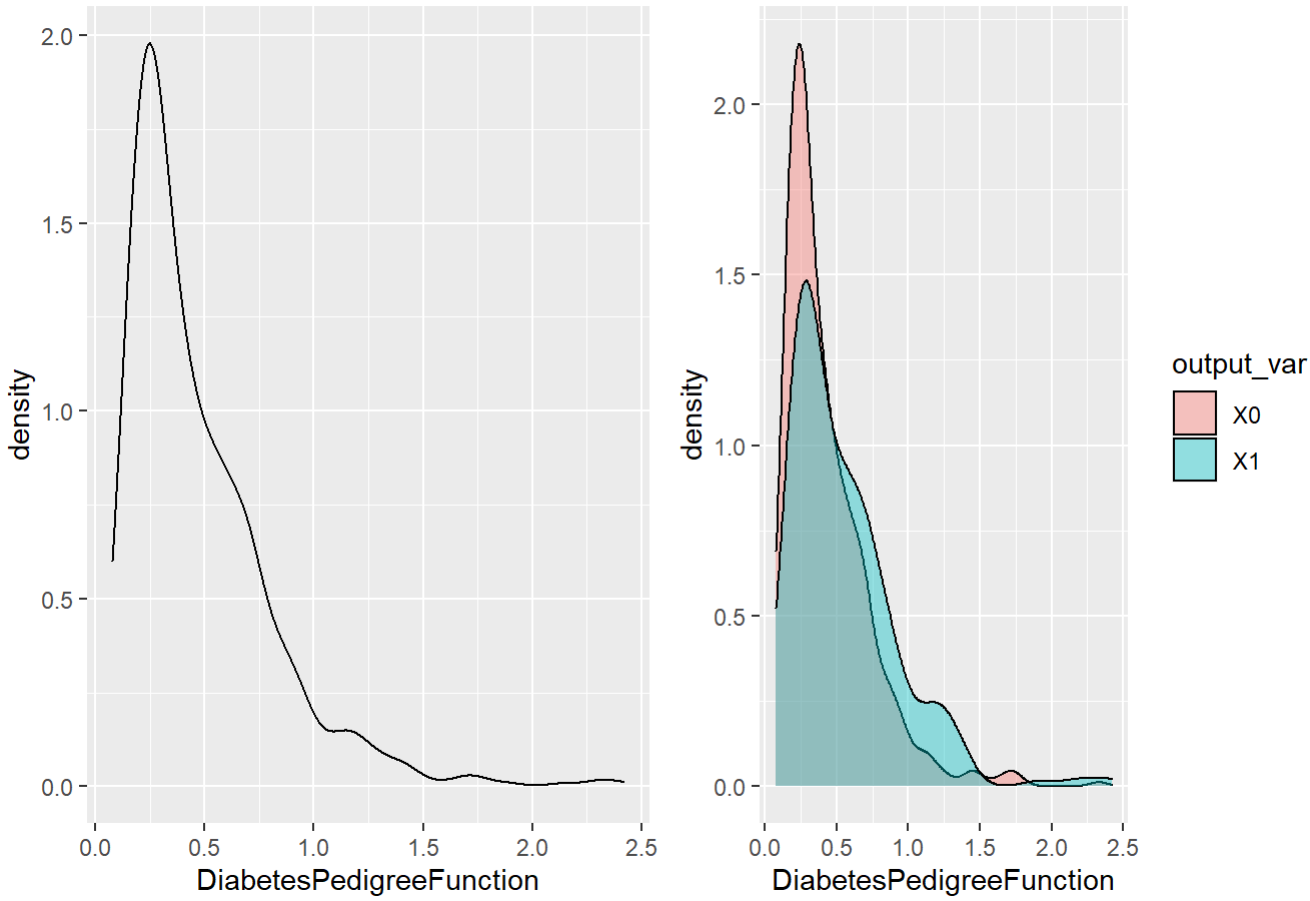
Insulin variable / [Skew: 2.70902864789091]



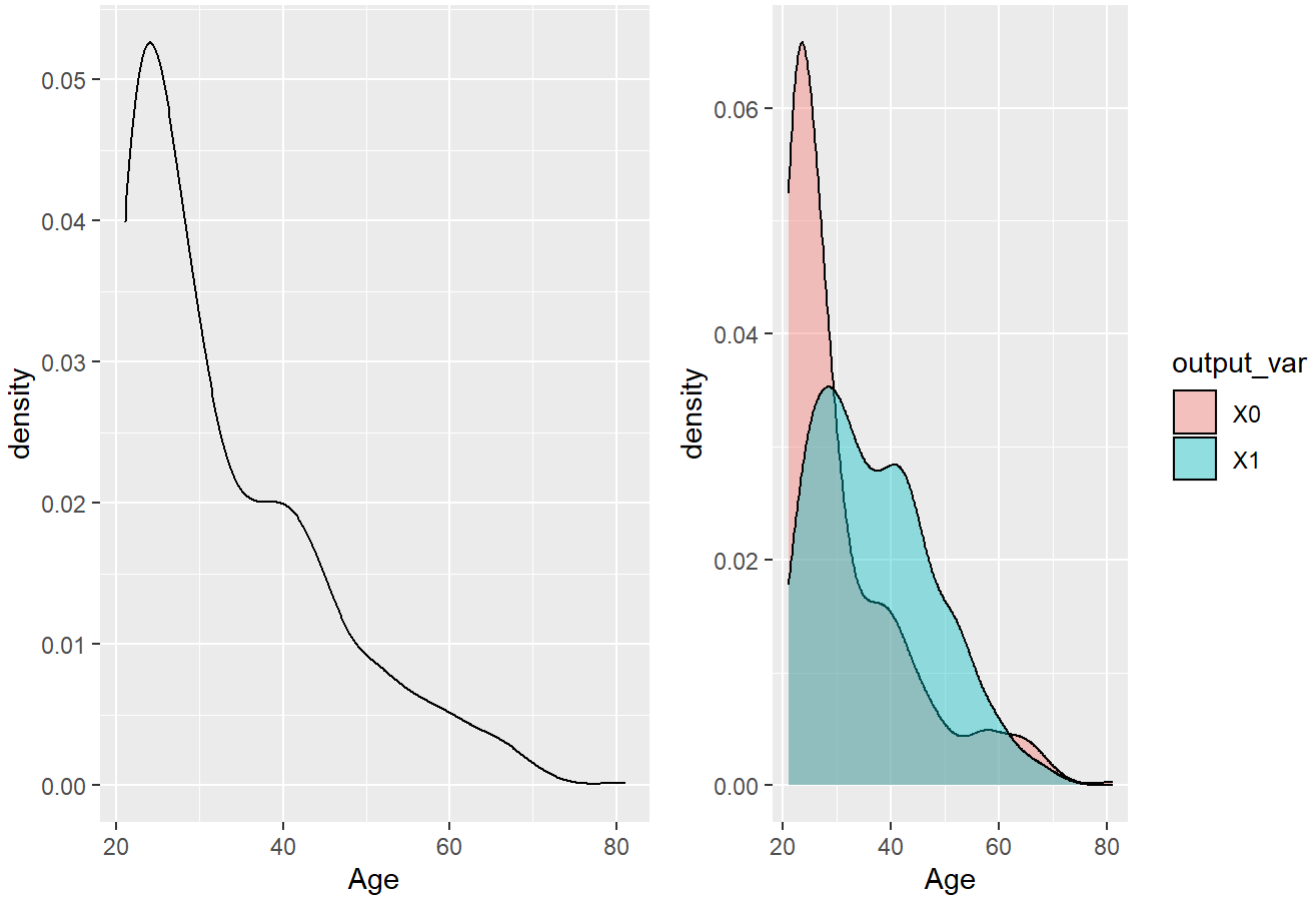
BMI variable / [Skew: 0.596275540402581]



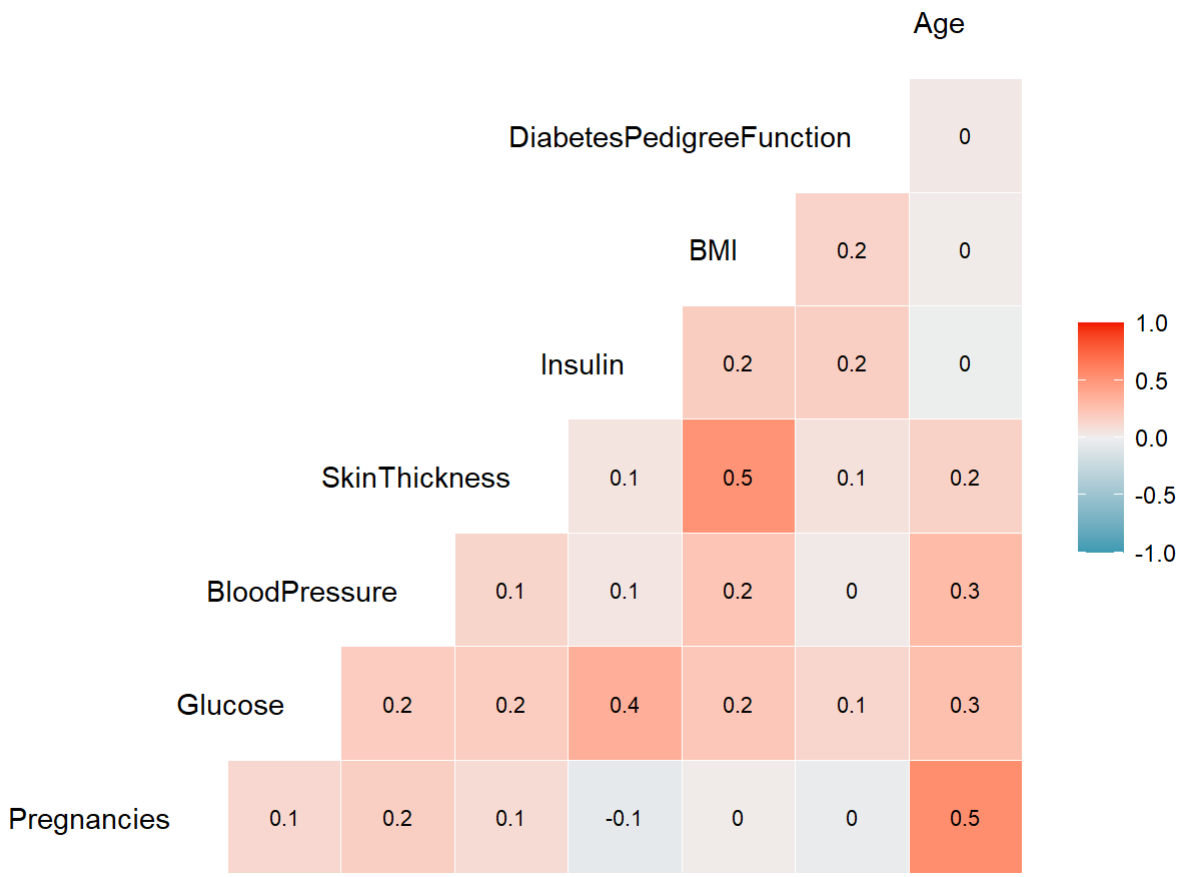
DiabetesPedigreeFunction variable / [Skew: 1.91241792381955]



Age variable / [Skew: 1.12518804431459]

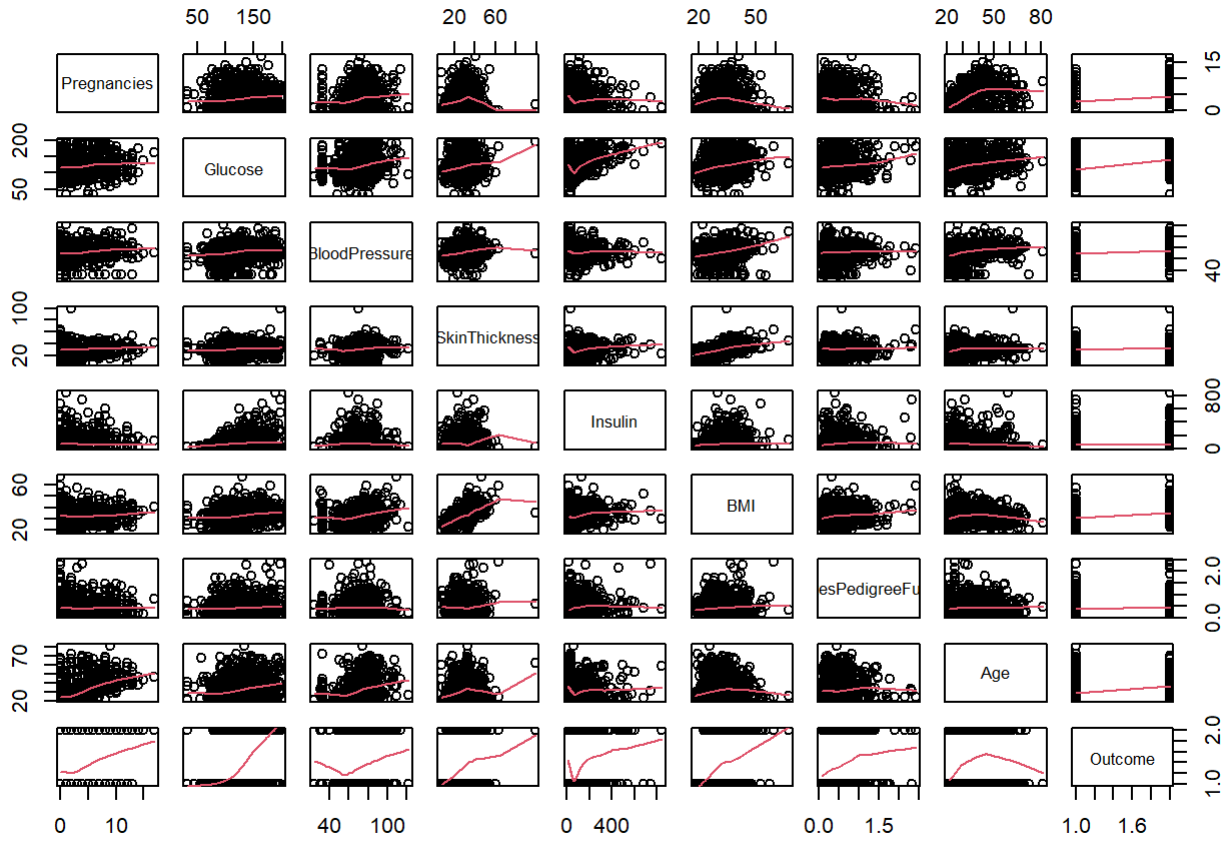


```
GGally::ggcorr(data[,-9], hjust = 1, layout.exp = 2, label = T, label_size = 2.9)
```



There is no strong correlation among predictor variables

```
pairs(data, panel = panel.smooth)
```



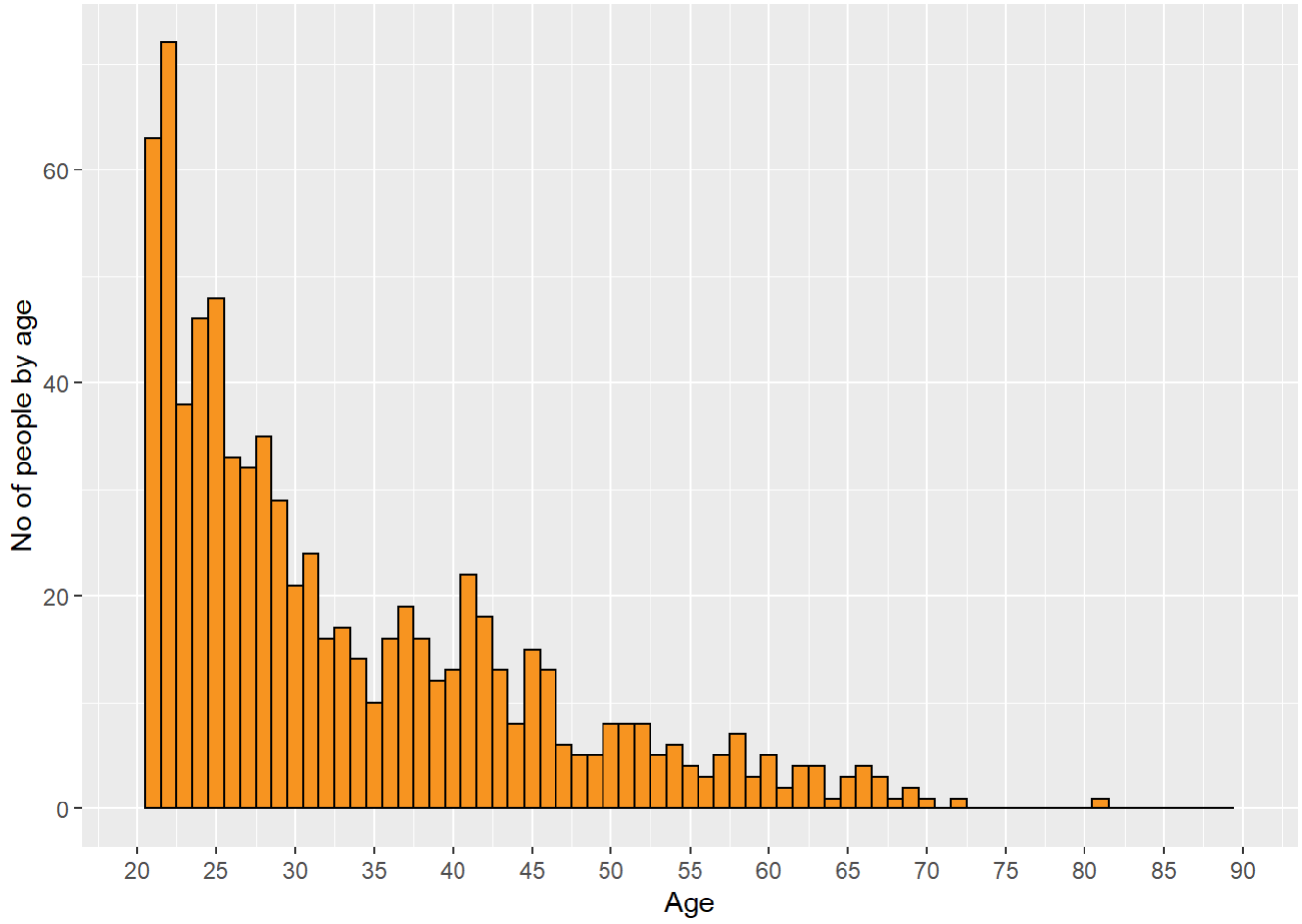

```
data$Age_Cat <- ifelse(data$Age < 21, "<21",
                  ifelse((data$Age>=21) & (data$Age<=25), "21-25",
                  ifelse((data$Age>25) & (data$Age<=30), "25-30",
                  ifelse((data$Age>30) & (data$Age<=35), "30-35",
                  ifelse((data$Age>35) & (data$Age<=40), "35-40",
                  ifelse((data$Age>40) & (data$Age<=50), "40-50",
                  ifelse((data$Age>50) & (data$Age<=60), "50-60",">60")))))))

data$Age_Cat <- factor(data$Age_Cat, levels = c('<21','21-25','25-30','30-35','35-40','40-50','5
0-60','>60'))
table(data$Age_Cat)
```

```
##
##   <21 21-25 25-30 30-35 35-40 40-50 50-60   >60
##     0   267   150    81    76   113    54    27
```

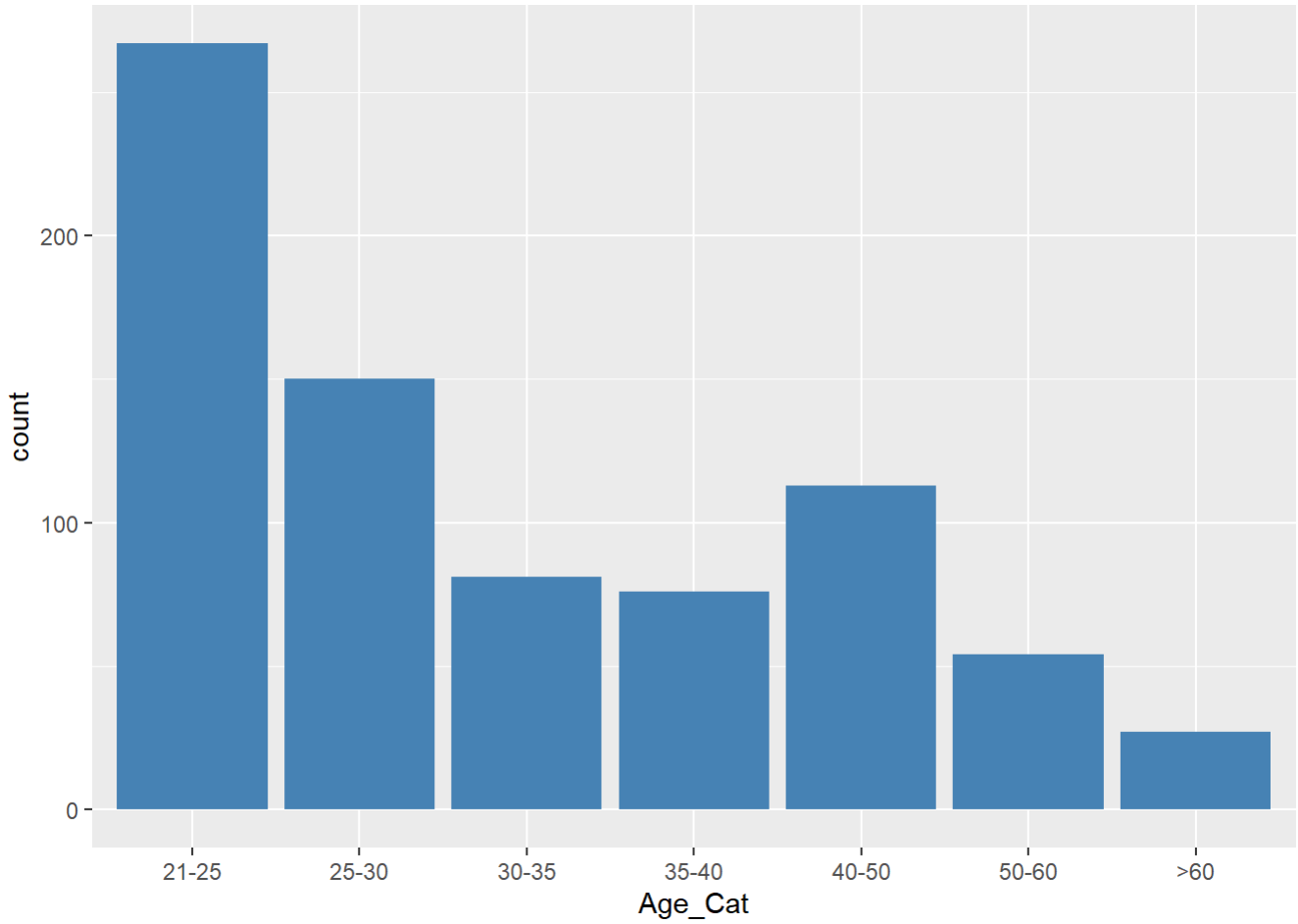
```
# Histogram of Age
ggplot(aes(x = Age), data=data) +
        geom_histogram(binwidth=1, color='black', fill = "#F79420") +
        scale_x_continuous(limits=c(20,90), breaks=seq(20,90,5)) +
        xlab("Age") +
        ylab("No of people by age")
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



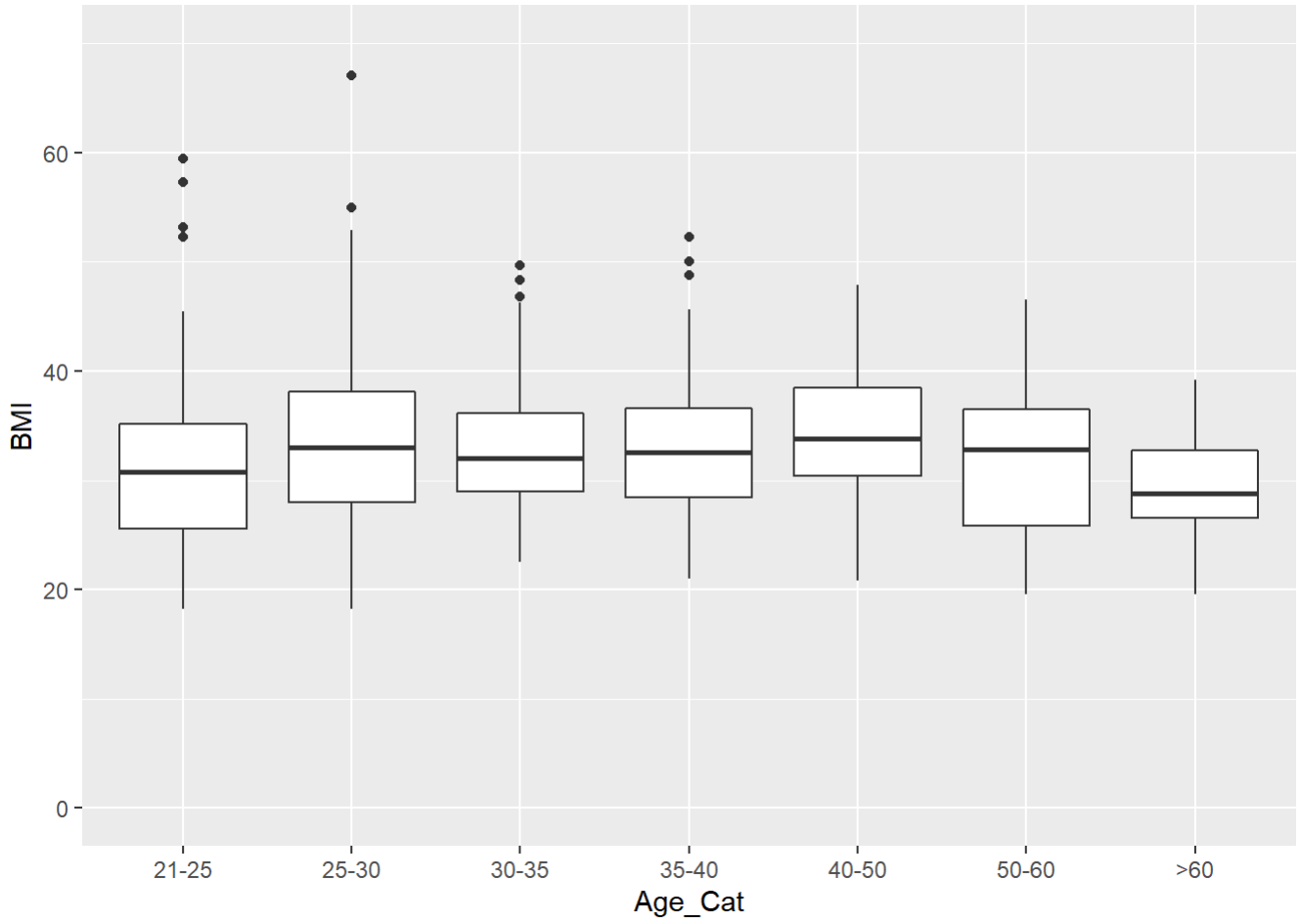
Most of the subjects are in between the ages 21 - 30

```
# Barplot by Age_Cat
ggplot(aes(x = Age_Cat), data = data) +
            geom_bar(fill='steelblue')
```




```
# box plot of Age_Cat vs BMI

ggplot(aes(x=Age_Cat, y = BMI), data = data) +
        geom_boxplot() +
        coord_cartesian(ylim = c(0,70))
```




```
by(data$BMI, data$Age_Cat, summary)
```

```
## data$Age_Cat: <21
## NULL
## ------------------------------------------------------------
## data$Age_Cat: 21-25
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.20   25.60   30.80   31.21   35.20   59.40
## ------------------------------------------------------------
## data$Age_Cat: 25-30
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.20   28.00   33.00   33.47   38.10   67.10
## ------------------------------------------------------------
## data$Age_Cat: 30-35
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   22.50   29.00   32.00   32.81   36.10   49.70
## ------------------------------------------------------------
## data$Age_Cat: 35-40
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   28.48   32.60   32.97   36.58   52.30
## ------------------------------------------------------------
## data$Age_Cat: 40-50
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    20.8    30.4    33.8    34.5    38.5    47.9
## ------------------------------------------------------------
## data$Age_Cat: 50-60
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.60   25.85   32.85   31.71   36.52   46.50
## ------------------------------------------------------------
## data$Age_Cat: >60
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.60   26.55   28.80   29.60   32.70   39.20
```

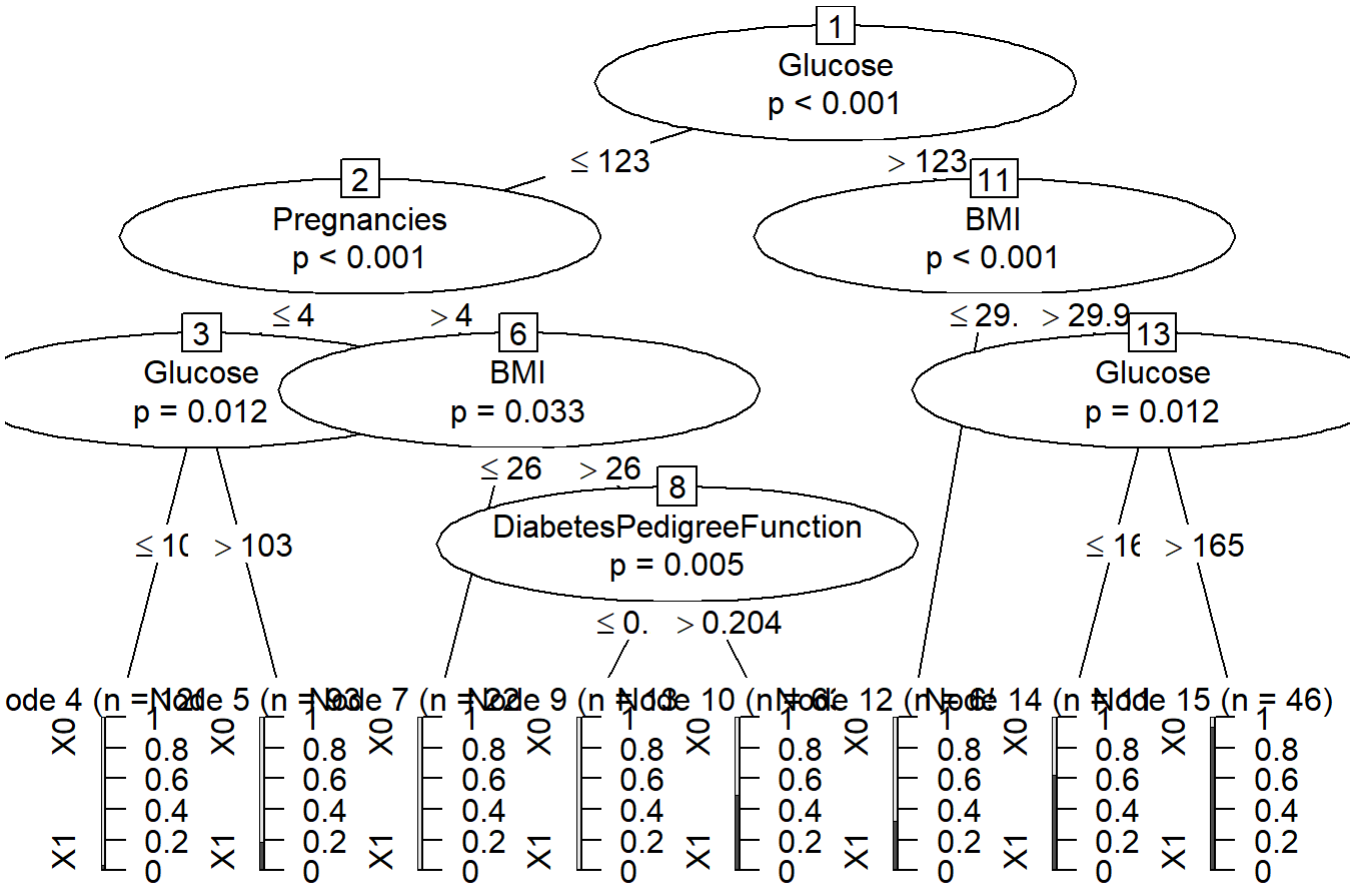
Machine Learning Model

1. Decision Tree Model

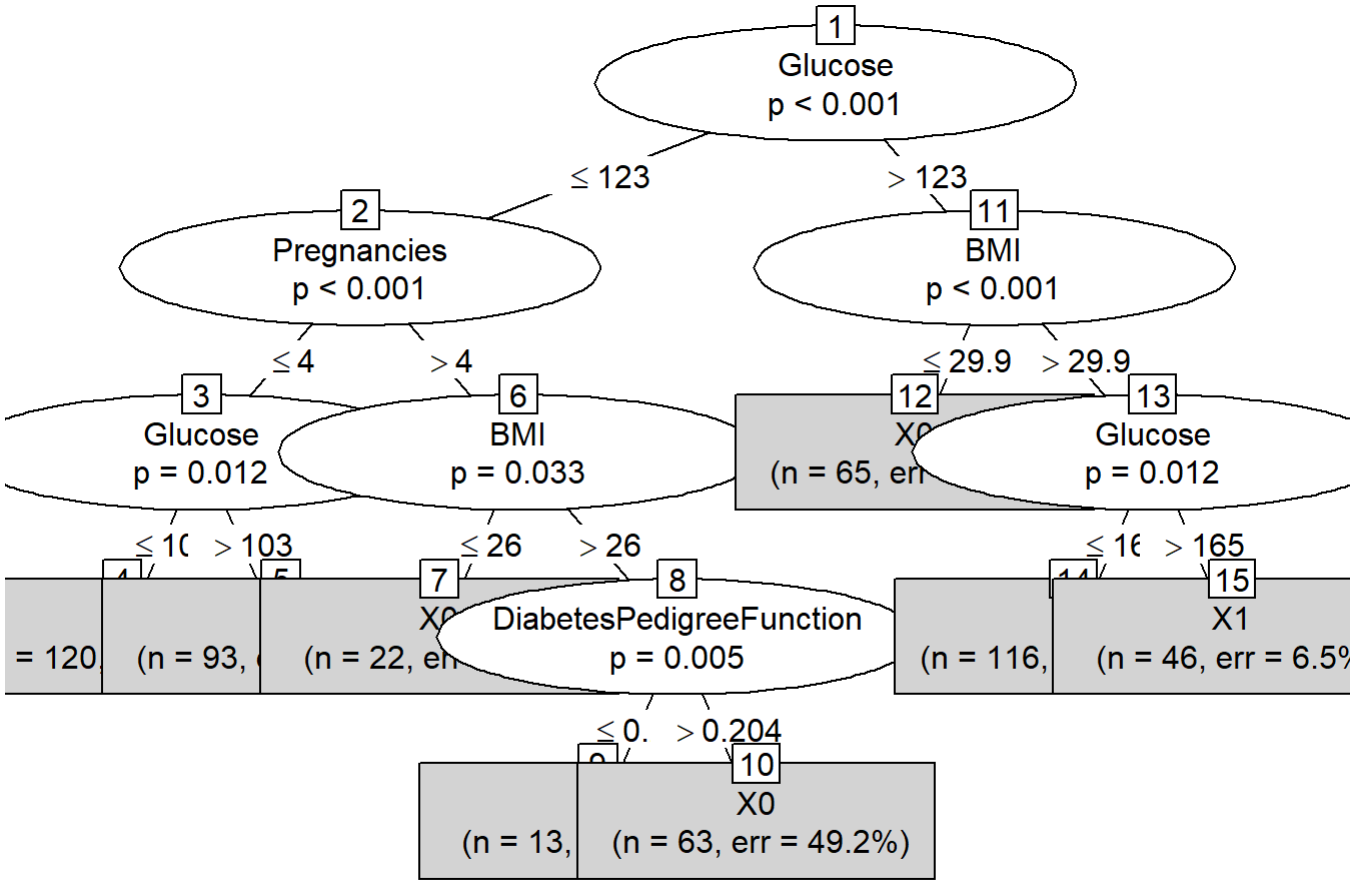
```
data = subset(data, select=-c(Age_Cat))
```

```
set.seed(7)
dindex <- createDataPartition(data$Outcome, p=0.7, list=FALSE)
data_train <- data[dindex,]
data_test <- data[-dindex,]
```

```
model_dt <- ctree(Outcome~., data_train)
plot(model_dt)
```




```
plot(model_dt, type = "simple")
```



we can see the number of nodes and its distribution. In which :

[1] is root node

[2],[3],[6],[8], [11], and [13] are internal nodes or branch. Internal nodes shown by arrow pointing to/from them.

[4],[5],[7],[9], [10], [12], [14], [15] are leaf nodes or leaf.

The model above we can apply to our data test.

```
pred_dt <- predict(model_dt, data_test)
```

```
(conf_matrix_dtree <- table(pred_dt, data_test$Outcome))
```

```
##
## pred_dt  X0  X1
##      X0 125  36
##      X1  25  44
```

Result of confusion Matrix shows that decision tree predicts 125 cases negative diabetes correctly and 36 cases with wrong prediction. At the same time, this model predicts that there are 44 positive diabetes correctly and 25 cases of wrong prediction.

```
caret::confusionMatrix(pred_dt, data_test[,9])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  X0  X1
##         X0 125  36
##         X1  25  44
##
##                Accuracy : 0.7348
##                  95% CI : (0.6728, 0.7906)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : 0.004551
##
##                   Kappa : 0.396
##
##  Mcnemar's Test P-Value : 0.200415
##
##             Sensitivity : 0.8333
##             Specificity : 0.5500
##          Pos Pred Value : 0.7764
##          Neg Pred Value : 0.6377
##              Prevalence : 0.6522
##          Detection Rate : 0.5435
##    Detection Prevalence : 0.7000
##       Balanced Accuracy : 0.6917
##
##        'Positive' Class : X0
##
```

The Accuracy of Model is 71% for Decision Tree.

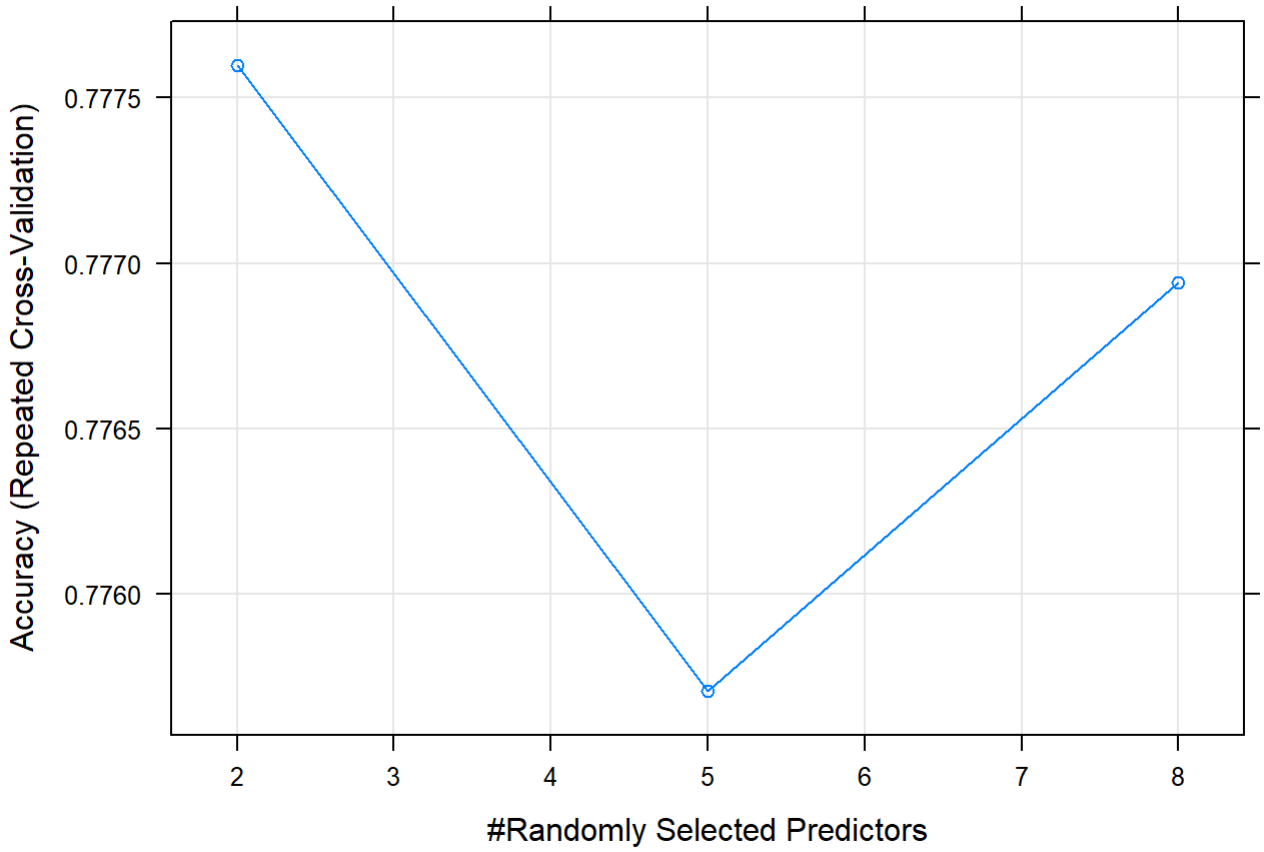
2. Random Forest Model

The model will be built using 5-fold cross validation, and 3 repeats

```
# train() to make model, method = to use k-fold, repeats= to show the best 3 value of mytr
ctrl <- trainControl(method = "repeatedcv", number =5, repeats = 3)


model_forest <- train(Outcome~., data=data_train, method="rf", trControl=ctrl)
```

```
plot(model_forest)
```

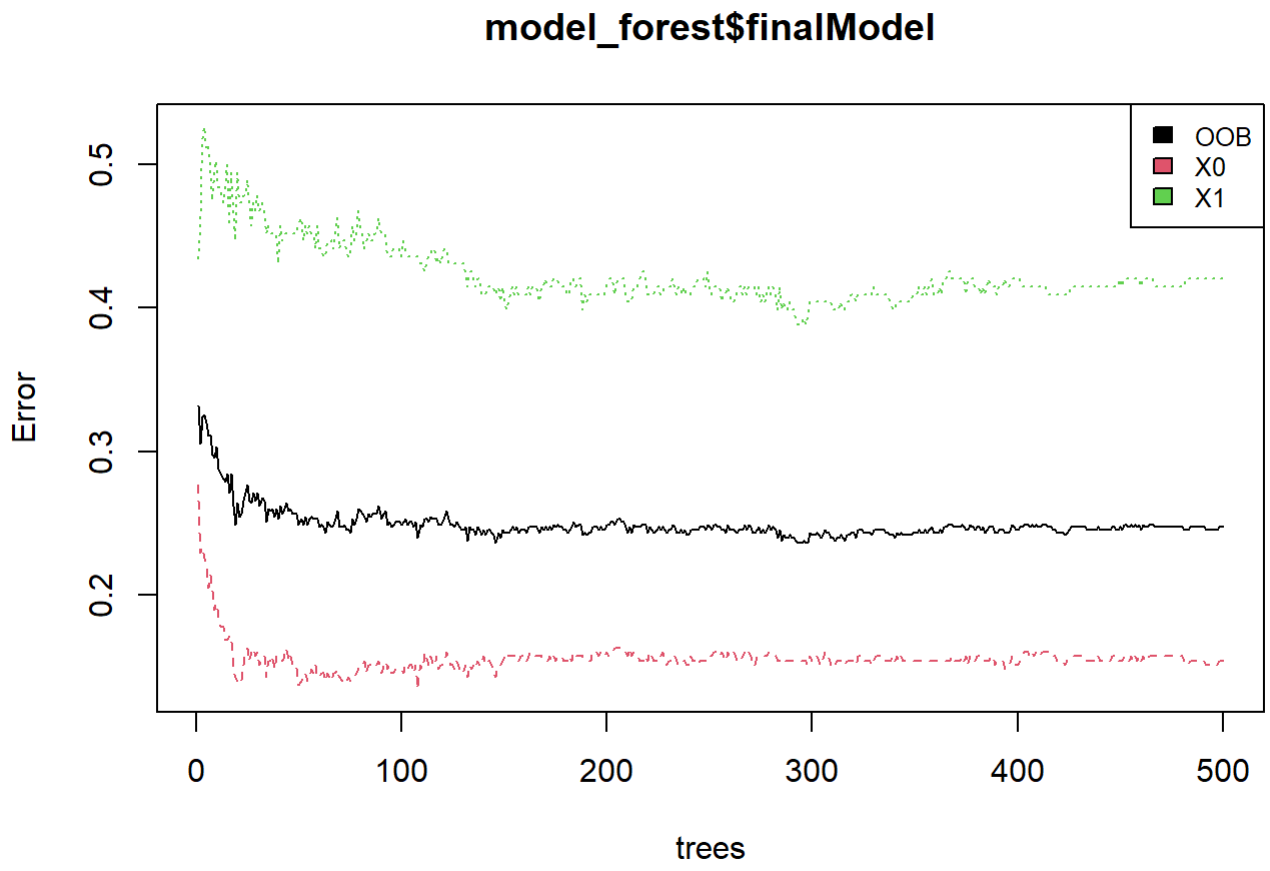



```
varImp(model_forest)
```

```
## rf variable importance
##
##                           Overall
## Glucose                   100.000
## BMI                        51.783
## DiabetesPedigreeFunction   29.490
## Age                        27.784
## BloodPressure               7.253
## Pregnancies                 5.932
## SkinThickness               4.346
## Insulin                     0.000
```

Based on result above, we know that glucose rate has the highest impact to the result while the other variables are only 50% or less than it.

```
plot(model_forest$finalModel)
legend("topright", colnames(model_forest$finalModel$err.rate),
       col=1:6, cex= 0.8, fill=1:6)
```



Based on visualization above comparison of OOB and targeted variable. It depicts that from tree number around 100 the error of model has been better, yet we can still use more than 400 trees to reduce our OOB.

```
model_forest$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = min(param$mtry, ncol(x)))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 24.72%
## Confusion matrix:
##     X0  X1 class.error
## X0 296  54   0.1542857
## X1  79 109   0.4202128
```

```
predict_forest <- predict(model_forest, data_test)
```

```
(conf_matrix_forest1 <- table(predict_forest, data_test$Outcome))
```

```
##
## predict_forest  X0  X1
##             X0 124  33
##             X1  26  47
```

Result of confusion Matrix shows that decision tree predicts 124 cases negative diabetes correctly and 33 cases with wrong prediction. At the same time, this model predicts that there are 47 positive diabetes correctly and 26 cases of wrong prediction.

```
confusionMatrix(conf_matrix_forest1)
```

```
## Confusion Matrix and Statistics
##
##
## predict_forest  X0  X1
##             X0 124  33
##             X1  26  47
##
##                Accuracy : 0.7435
##                  95% CI : (0.6819, 0.7986)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : 0.001872
##
##                   Kappa : 0.4228
##
##  Mcnemar's Test P-Value : 0.434724
##
##             Sensitivity : 0.8267
##             Specificity : 0.5875
##          Pos Pred Value : 0.7898
##          Neg Pred Value : 0.6438
##              Prevalence : 0.6522
##          Detection Rate : 0.5391
##    Detection Prevalence : 0.6826
##       Balanced Accuracy : 0.7071
##
##        'Positive' Class : X0
##
```

Glucose is variable that most impact to diabetes, and followed by age, pedigree and pressure. The Accuracy of Random Forest Model is 74%

3. Support Vector Machine

```
svm.model <- svm(Outcome~., data = data_train, kernel="sigmoid")
svm.pred <- predict(svm.model, data_test)
```

```
cm_svm <- confusionMatrix(svm.pred, data_test$Outcome, positive = "X1")
cm_svm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  X0  X1
##         X0 126  33
##         X1  24  47
##
##                Accuracy : 0.7522
##                  95% CI : (0.6912, 0.8066)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : 0.0007075
##
##                   Kappa : 0.439
##
##  Mcnemar's Test P-Value : 0.2893148
##
##             Sensitivity : 0.5875
##             Specificity : 0.8400
##          Pos Pred Value : 0.6620
##          Neg Pred Value : 0.7925
##              Prevalence : 0.3478
##          Detection Rate : 0.2043
##    Detection Prevalence : 0.3087
##       Balanced Accuracy : 0.7137
##
##        'Positive' Class : X1
##
```

Result of confusion Matrix shows that Support Vector Machine predicts 126 cases negative diabetes correctly and 33 cases with wrong prediction. At the same time, this model predicts that there are 47 positive diabetes correctly and 24 cases of wrong prediction.

The Accuracy of Support Vector Machine is 75%

4. Logistic Regression

```
fitControl <- trainControl(method = "cv", number = 10, classProbs = TRUE, summaryFunction = twoC
lassSummary)
model_glm <- train(Outcome~., data_train,
                   method = "glm",
                   metric = "ROC",
                   preProcess = c('center','scale'),
                   trControl=fitControl)
```

```
pred_glm <- predict(model_glm, data_test)
cm_glm <- confusionMatrix(pred_glm, data_test$Outcome, positive = "X1")
cm_glm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  X0  X1
##         X0 129  29
##         X1  21  51
##
##                Accuracy : 0.7826
##                  95% CI : (0.7236, 0.8341)
##     No Information Rate : 0.6522
##     P-Value [Acc > NIR] : 1.156e-05
##
##                   Kappa : 0.5094
##
##  Mcnemar's Test P-Value : 0.3222
##
##             Sensitivity : 0.6375
##             Specificity : 0.8600
##          Pos Pred Value : 0.7083
##          Neg Pred Value : 0.8165
##              Prevalence : 0.3478
##          Detection Rate : 0.2217
##    Detection Prevalence : 0.3130
##       Balanced Accuracy : 0.7488
##
##        'Positive' Class : X1
##
```

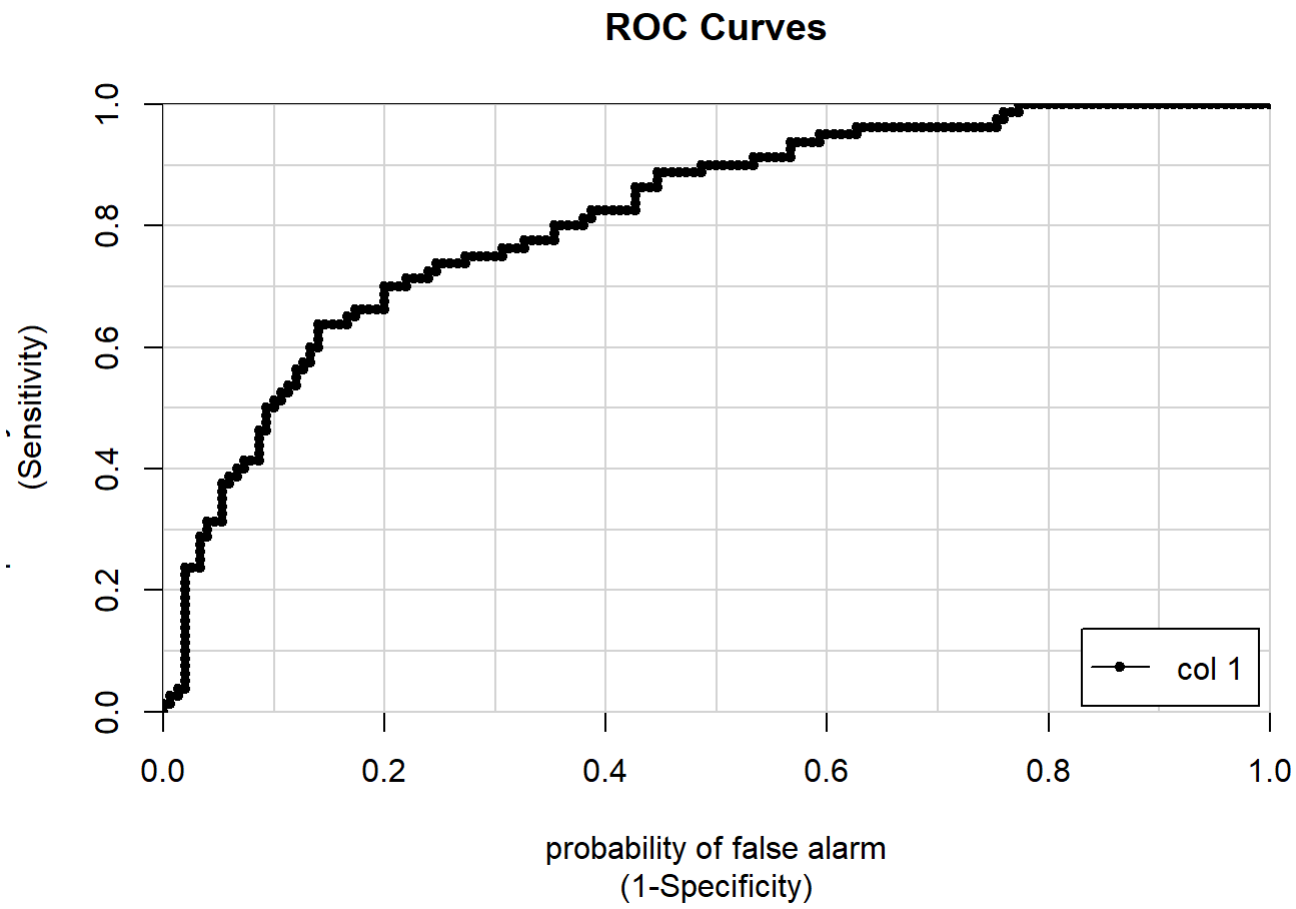
Result of confusion Matrix shows that Logistic Regression predicts 129 cases negative diabetes correctly and 29 cases with wrong prediction. At the same time, this model predicts that there are 51 positive diabetes correctly and 21 cases of wrong prediction

```
pred_prob_glm <- predict(model_glm, data_test, type="prob")
roc_glm <- roc(data_test$Outcome, pred_prob_glm$X1)
```

```
## Setting levels: control = X0, case = X1
```

```
## Setting direction: controls < cases
```

```
colAUC(pred_prob_glm$X1, data_test$Outcome, plotROC = TRUE)
```




```
##           [,1]
## X0 vs. X1 0.814
```

We can see the result of this model:

The accuracy is the best. The auc has a value of 0.81 The F1 score is 0.65 The recall (Sensitivity) is okay:- 0.6375

Based on order accuracy and recall value, Logistic Regression model is the best classification model, with accuracy level of 78% and recall level 63%