Datamining_assignment_1.R

rocky

Sun Apr 16 19:13:10 2017

```
# 2017-1 Ajou univ. Data mining Assignment #1-4
# Heerak Lim, lrocky1229@gmail.com

library (MASS)

# for making Decision tree
#install.packages("party")
library(party)
```

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

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

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
# for Naive Bayes Classification
#install.packages("e1071")
library(e1071)

# for cross validation
#install.packages("caret")
library(caret)
```

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

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

```
library(class)

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



trainingData = Pima.tr
testData = Pima.te

# number of samples and features in dataset
nrow(trainingData)
```

```
## [1] 200
```

```
nrow(testData)
```

```
## [1] 332
```

```
length(trainingData)
```

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

```
length(testData)
```

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

```
# summary of statistics of each feature of the dataset

# trainingData
summary(trainingData$npreg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    1.00    2.00    3.57    6.00   14.00
```

```
summary(trainingData$glu)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    56.0   100.0   120.5   124.0   144.0   199.0
```

```
summary(trainingData$bp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   38.00   64.00   70.00   71.26   78.00  110.00
```

```
summary(trainingData$skin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    7.00   20.75   29.00   29.22   36.00   99.00
```

```
summary(trainingData$bmi)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   18.20   27.58   32.80   32.31   36.50   47.90
```

```
summary(trainingData$ped)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.0850  0.2535  0.3725  0.4608  0.6160  2.2880
```

```
summary(trainingData$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   21.00   23.00   28.00   32.11   39.25   63.00
```

```
summary(trainingData$type)
```

```
##  No Yes 
## 132  68
```

```
# testData
summary(testData$npreg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.000   1.000   2.000   3.485   5.000  17.000
```

```
summary(testData$glu)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    65.0    96.0   112.0   119.3   136.2   197.0
```

```
summary(testData$bp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   24.00   64.00   72.00   71.65   80.00  110.00
```

```
summary(testData$skin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    7.00   22.00   29.00   29.16   36.00   63.00
```

```
summary(testData$bmi)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.40   28.17   32.90   33.24   37.20   67.10
```

```
summary(testData$ped)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0850  0.2660  0.4400  0.5284  0.6792  2.4200
```

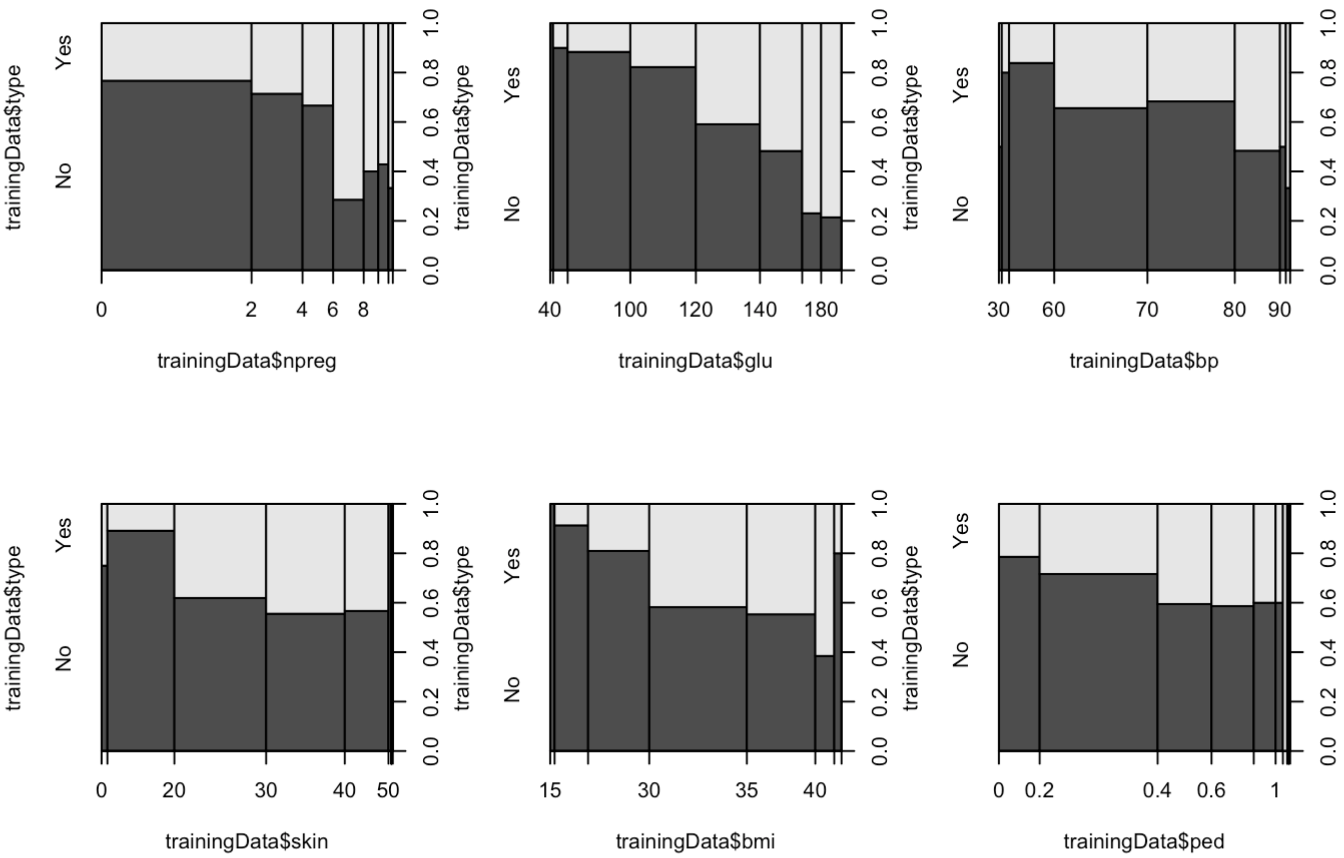
```
summary(testData$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   23.00   27.00   31.32   37.00   81.00
```

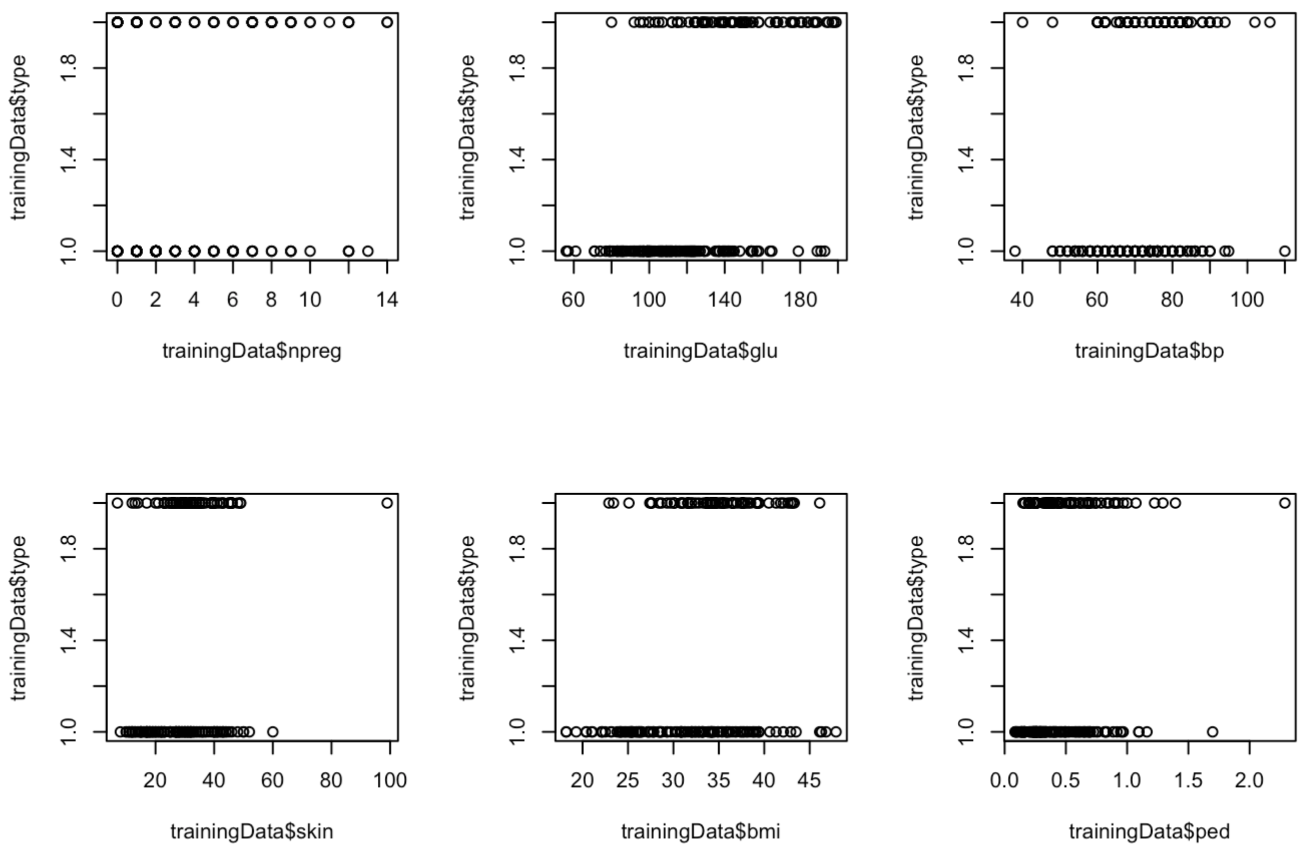
```
summary(testData$type)
```

```
##  No Yes
## 223 109
```

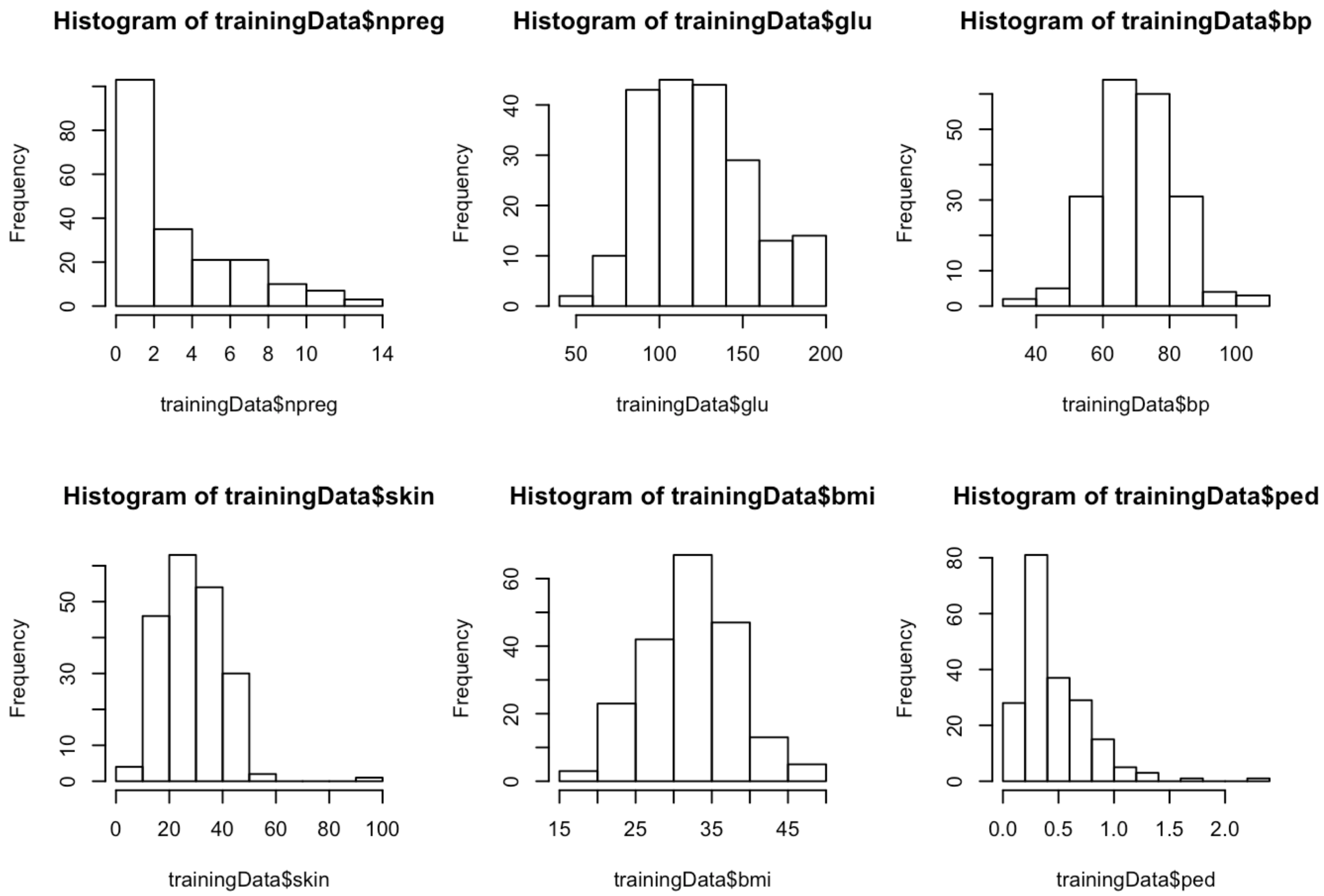
```
# Show at least one plot that shows the output variable (y)
# is correlated with some of the input variables X.

par(mfrow=c(2,3))
plot(trainingData$type~trainingData$npreg)
plot(trainingData$type~trainingData$glu)
plot(trainingData$type~trainingData$bp)
plot(trainingData$type~trainingData$skin)
plot(trainingData$type~trainingData$bmi)
plot(trainingData$type~trainingData$ped)
```




```
# or this
plot(trainingData$npreg,trainingData$type)
plot(trainingData$glu,trainingData$type)
plot(trainingData$bp,trainingData$type)
plot(trainingData$skin,trainingData$type)
plot(trainingData$bmi,trainingData$type)
plot(trainingData$ped,trainingData$type)
```



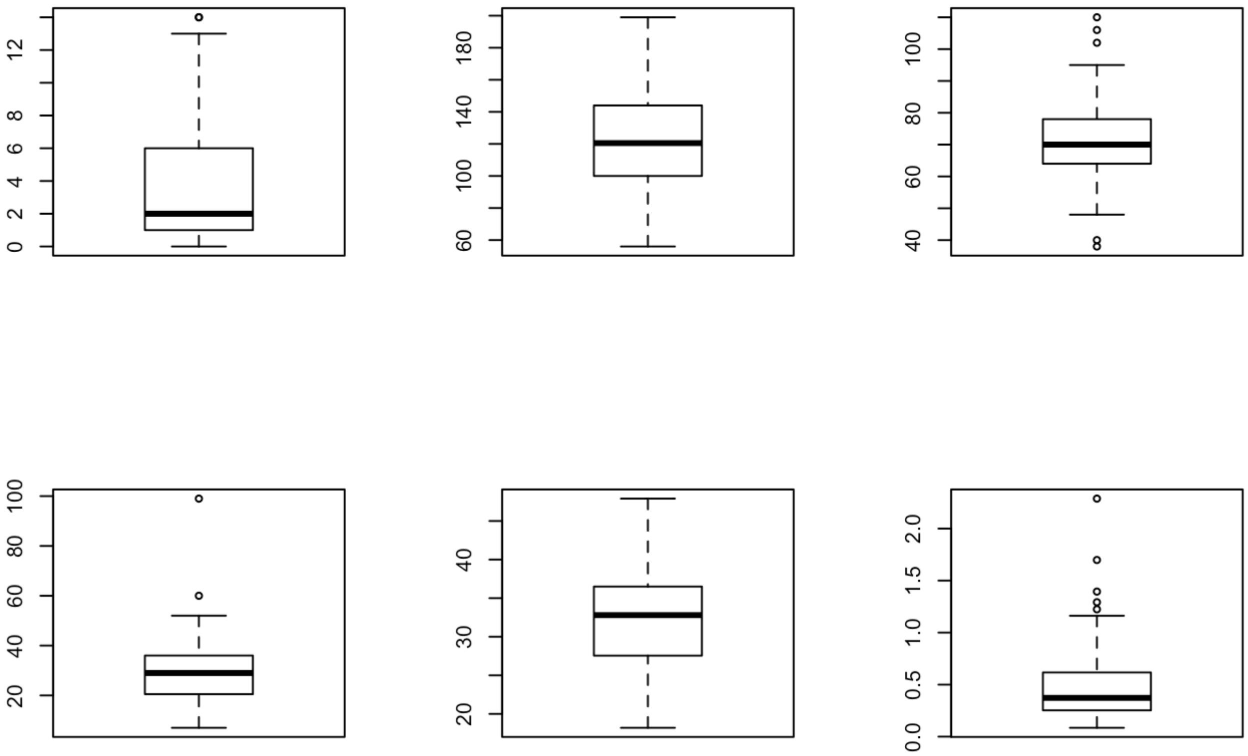

```
# histogram of 6continuous varicables
hist(trainingData$npreg)
hist(trainingData$glu)
hist(trainingData$bp)
hist(trainingData$skin)
hist(trainingData$bmi)
hist(trainingData$ped)
```



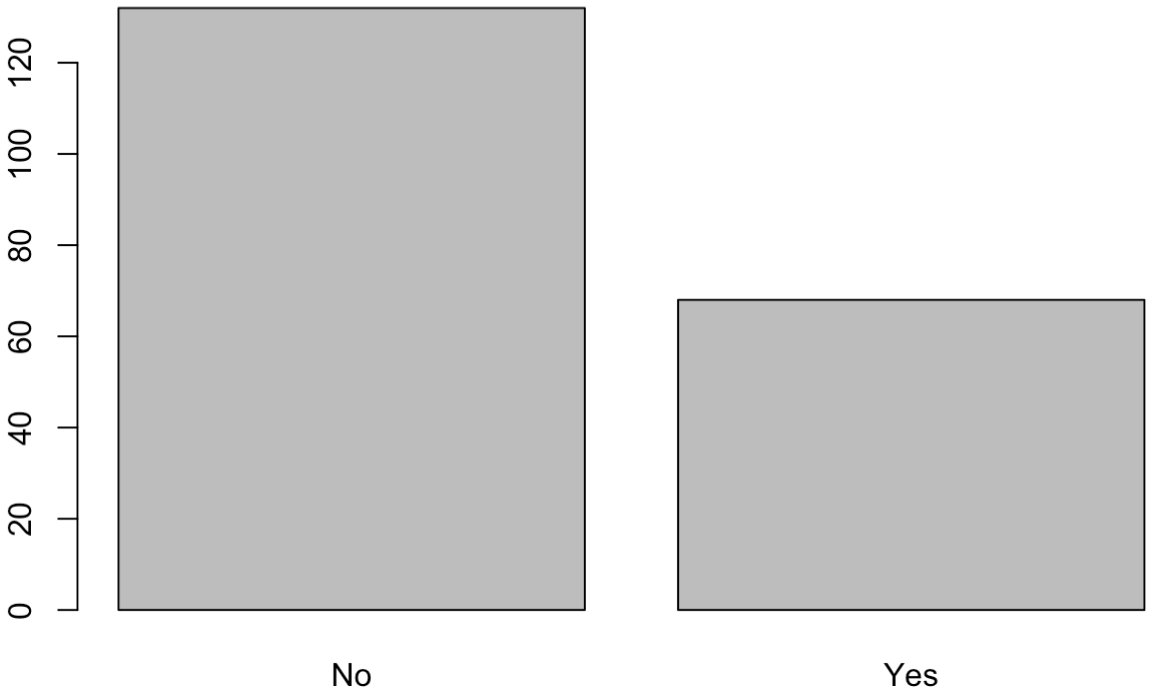

```
v <- readline("please enter any key to next: ")
```

```
## please enter any key to next:
```

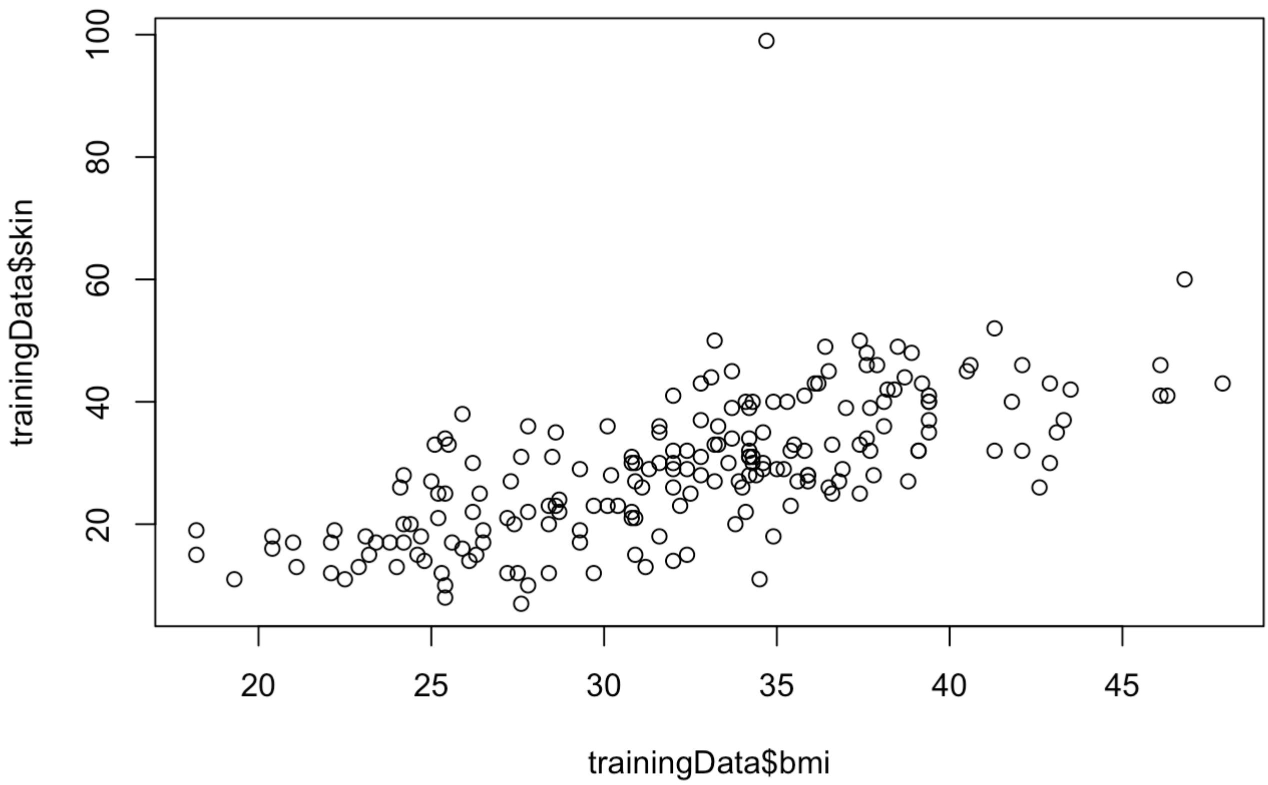
```
# boxblot of 6continuous varicables
boxplot(trainingData$npreg)
boxplot(trainingData$glu)
boxplot(trainingData$bp)
boxplot(trainingData$skin)
boxplot(trainingData$bmi)
boxplot(trainingData$ped)
```




```
# plot of categorical variable(type)
par(mfrow=c(1,1))
plot(trainingData$type)
```




```
# analyze one interesting relationship between the selected two features
# I choose the skin(triceps skin fold thickness) and bmi(body mass index)

# simple plot
plot(trainingData$skin~trainingData$bmi)
```



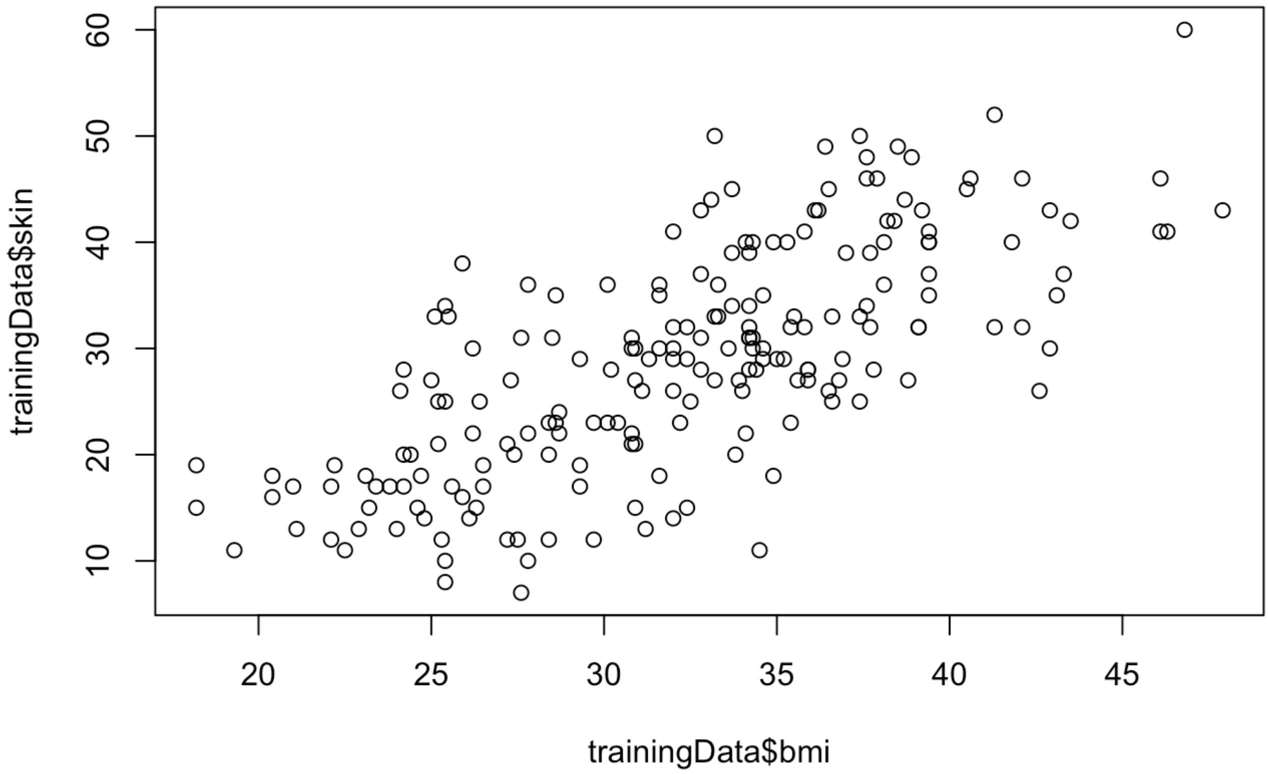

```
# linear regression
fig = lm(trainingData$skin~trainingData$bmi)
summary(fig)
```

```
##
## Call:
## lm(formula = trainingData$skin ~ trainingData$bmi)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.975  -5.633  -0.913   4.558  66.772
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -11.5107     3.3616  -3.424  0.00075 ***
## trainingData$bmi   1.2605     0.1022  12.330  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.84 on 198 degrees of freedom
## Multiple R-squared:  0.4343, Adjusted R-squared:  0.4315
## F-statistic:   152 on 1 and 198 DF,  p-value: < 2.2e-16
```

```
# remove outlier(index 157 data)
trainingData <- trainingData[-c(157),]
nrow(trainingData)
```

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

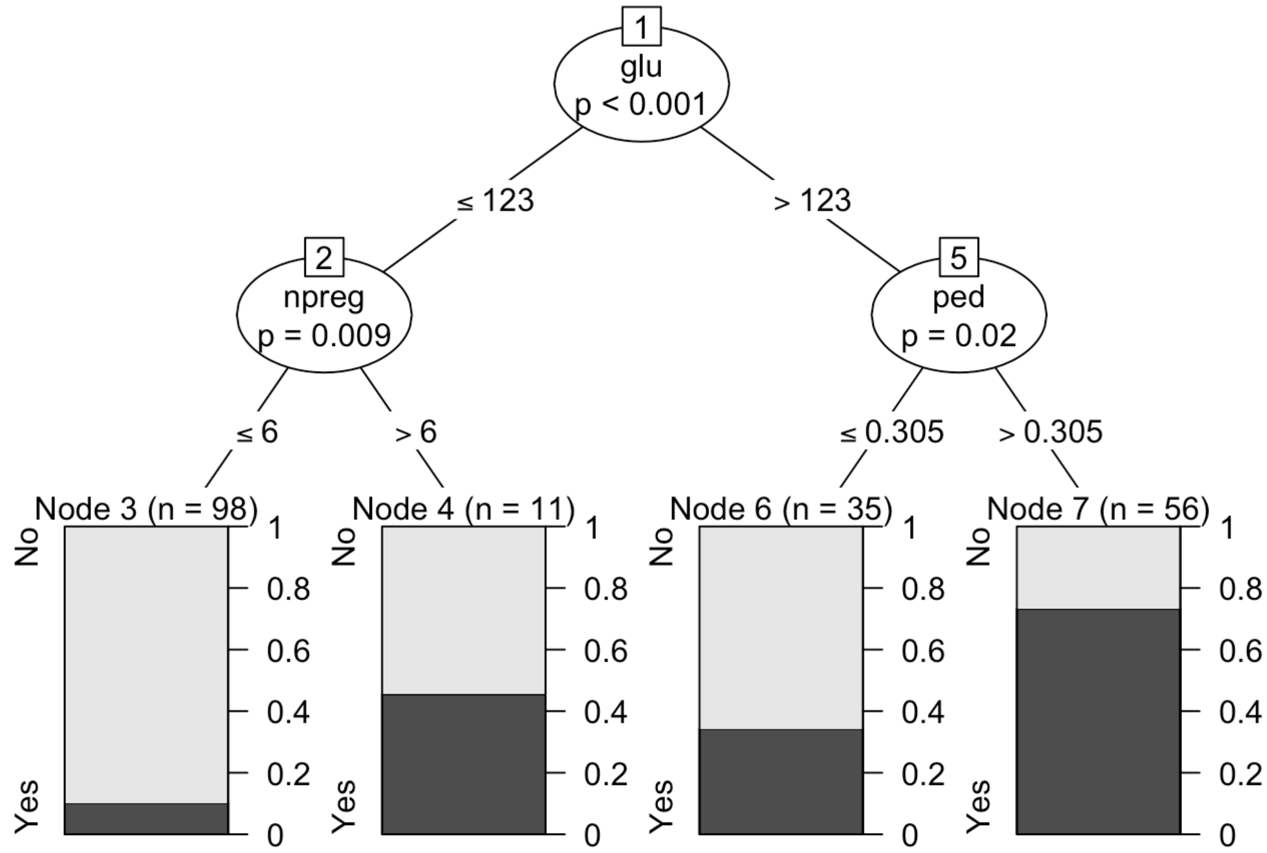
```
# plot and linear regression on the new dataset
plot(trainingData$skin~trainingData$bmi)
```




```
fig = lm(trainingData$skin~trainingData$bmi)
summary(fig)
```

```
##
## Call:
## lm(formula = trainingData$skin ~ trainingData$bmi)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.5926  -5.1018  -0.6373   4.8062  20.0181
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -11.1530     2.8404  -3.927 0.000119 ***
## trainingData$bmi   1.2390     0.0864  14.340  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.469 on 197 degrees of freedom
## Multiple R-squared:  0.5107, Adjusted R-squared:  0.5082
## F-statistic: 205.6 on 1 and 197 DF,  p-value: < 2.2e-16
```

```
# restore the original data
trainingData = Pima.tr

# make Decision Tree
training_data <- ctree(type ~ .,data=trainingData)
plot(training_data)
```




```
# Calculate training error by DT
pre_training = predict(training_data, trainingData)
trainingError = mean(pre_training!=trainingData$type)
print(trainingError)
```

```
## [1] 0.21
```

```
# Calculate test error by DT
pre_test = predict(training_data, testData)
testError = mean(pre_test!=testData$type)
print(testError)
```

```
## [1] 0.2710843
```

```
# Naive Bayes Classification
training_data <- naiveBayes(type~.,data=trainingData)

# Calculate training error by NBC
pre_training = predict(training_data, trainingData)
trainingError = mean(pre_training!=trainingData$type)
print(trainingError)
```

```
## [1] 0.22
```

```
#confusion matrix of training data
confusionMatrix <- table(pre_training, trainingData$type)
print(confusionMatrix)
```

```
##
## pre_training  No Yes
##          No  110  22
##          Yes  22  46
```

```
# Calculate test error by NBC
pre_test = predict(training_data, testData)
testError = mean(pre_test!=testData$type)
print(testError)
```

```
## [1] 0.2439759
```

```
#confusion matrix of training data
confusionMatrix <- table(pre_test, testData$type)
print(confusionMatrix)
```

```
##
## pre_test  No Yes
##      No  185  43
##      Yes  38  66
```

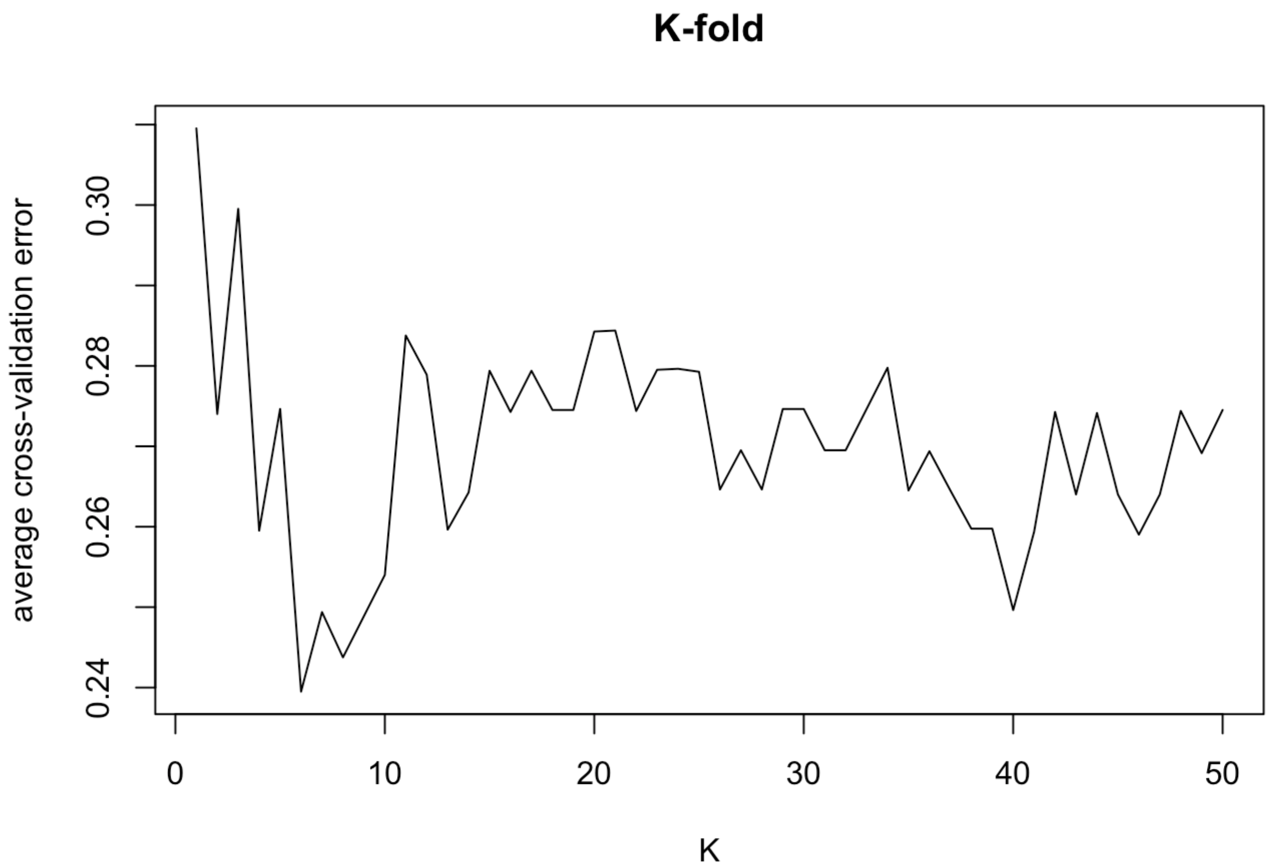
```r
# for cross validation


trainingAsNumeric <- trainingData
trainingAsNumeric$npreg <- as.numeric(as.character(trainingData$npreg))
trainingAsNumeric$glu <- as.numeric(as.character(trainingData$glu))
trainingAsNumeric$bp <- as.numeric(as.character(trainingData$bp))
trainingAsNumeric$skin <- as.numeric(as.character(trainingData$skin))
trainingAsNumeric$bmi <- as.numeric(as.character(trainingData$bmi))
trainingAsNumeric$ped <- as.numeric(as.character(trainingData$ped))
trainingAsNumeric$age <- as.numeric(as.character(trainingData$age))
trainingAsNumeric$type <- as.numeric(as.factor(trainingData$type))

normalize<-function(x){
  return((x-min(x))/(max(x)-min(x)))
}

nomalizedTrainingData <- as.data.frame(lapply(trainingData[1:7],normalize))
nomalizedTestData <- as.data.frame(lapply(testData[1:7],normalize))

set.seed(1)
# 5-fold cross validation
idx <- createFolds(trainingData$type, k=5)
print(sapply(idx, length))
```

```
## Fold1 Fold2 Fold3 Fold4 Fold5
##    39    41    40    39    41
```

```r
set.seed(1)
ks <- 1:50
res <- sapply(ks, function(k) {
  res.k <- sapply(seq_along(idx), function(i) {
    pred <- knn(train = nomalizedTrainingData[ -idx[[i]] ,] , test = nomalizedTrainin
gData[ idx[[i]], ], cl=trainingData$type[-idx[[i]] ], k=k)
    mean(trainingData$type[idx[[i]]]!=pred)
  })
  mean(res.k)
})

print(res)
```

```
##  [1] 0.3095341 0.2740150 0.2995278 0.2595091 0.2746435 0.2394966 0.2493871
##  [8] 0.2437523 0.2488806 0.2540088 0.2837774 0.2788993 0.2596373 0.2642652
## [15] 0.2793934 0.2742652 0.2793934 0.2745153 0.2745153 0.2842714 0.2843934
## [22] 0.2743871 0.2795153 0.2796373 0.2792652 0.2646310 0.2695091 0.2646310
## [29] 0.2746373 0.2746373 0.2695091 0.2695091 0.2746373 0.2797655 0.2645091
## [36] 0.2693871 0.2645091 0.2597592 0.2597592 0.2496310 0.2593871 0.2742714
## [43] 0.2640150 0.2741432 0.2640150 0.2590150 0.2640150 0.2743996 0.2691432
## [50] 0.2745216
```

```r
plot(ks, res, type="l",ylab="average cross-validation error", xlab="K", main="K-
fold")
```




```
# What is the Best Parameter K in experiments?
bestK<-which.min(res)
print(bestK)
```

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

```
bestPredict <- knn(train=nomalizedTrainingData, test=nomalizedTestData, cl=trainingDa
ta$type, k=bestK)
#final cross-validation error
min(res)
```

```
## [1] 0.2394966
```

```
#final classification error
mean(testData$type != bestPredict)
```

```
## [1] 0.2349398
```

```
#
#
#
# following code is same code for data which has no outlier
#
# model improvement by removing outlier data

trainingData = Pima.tr
testData = Pima.te


# removing outlier
trainingData <- trainingData[-c(8, 11, 48, 104, 111, 132, 135, 157, 193),]




# number of samples and features in dataset
nrow(trainingData)
```

```
## [1] 191
```

```
nrow(testData)
```

```
## [1] 332
```

```
length(trainingData)
```

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

```
length(testData)
```

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

```
# summary of statistics of each feature of the dataset

# trainingData
summary(trainingData$npreg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   2.000   3.618   6.000  14.000
```

```
summary(trainingData$glu)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    56.0   100.0   119.0   122.6   142.5   199.0
```

```
summary(trainingData$bp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   38.00   64.00   70.00   71.27   78.00  106.00
```

```
summary(trainingData$skin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##     7.0    21.0    29.0    28.9    36.0    60.0
```

```
summary(trainingData$bmi)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   18.20   27.55   32.80   32.28   36.50   47.90
```

```
summary(trainingData$ped)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.0850  0.2495  0.3640  0.4392  0.6015  1.3940
```

```
summary(trainingData$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    21.0    23.0    28.0    31.9    39.5    63.0
```

```
summary(trainingData$type)
```

```
##  No Yes 
## 127  64
```

```
# testData
summary(testData$npreg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.000   1.000   2.000   3.485   5.000  17.000
```

```
summary(testData$glu)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    65.0    96.0   112.0   119.3   136.2   197.0
```

```
summary(testData$bp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   24.00   64.00   72.00   71.65   80.00  110.00
```

```
summary(testData$skin)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    7.00   22.00   29.00   29.16   36.00   63.00
```

```
summary(testData$bmi)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.40   28.17   32.90   33.24   37.20   67.10
```

```
summary(testData$ped)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0850  0.2660  0.4400  0.5284  0.6792  2.4200
```

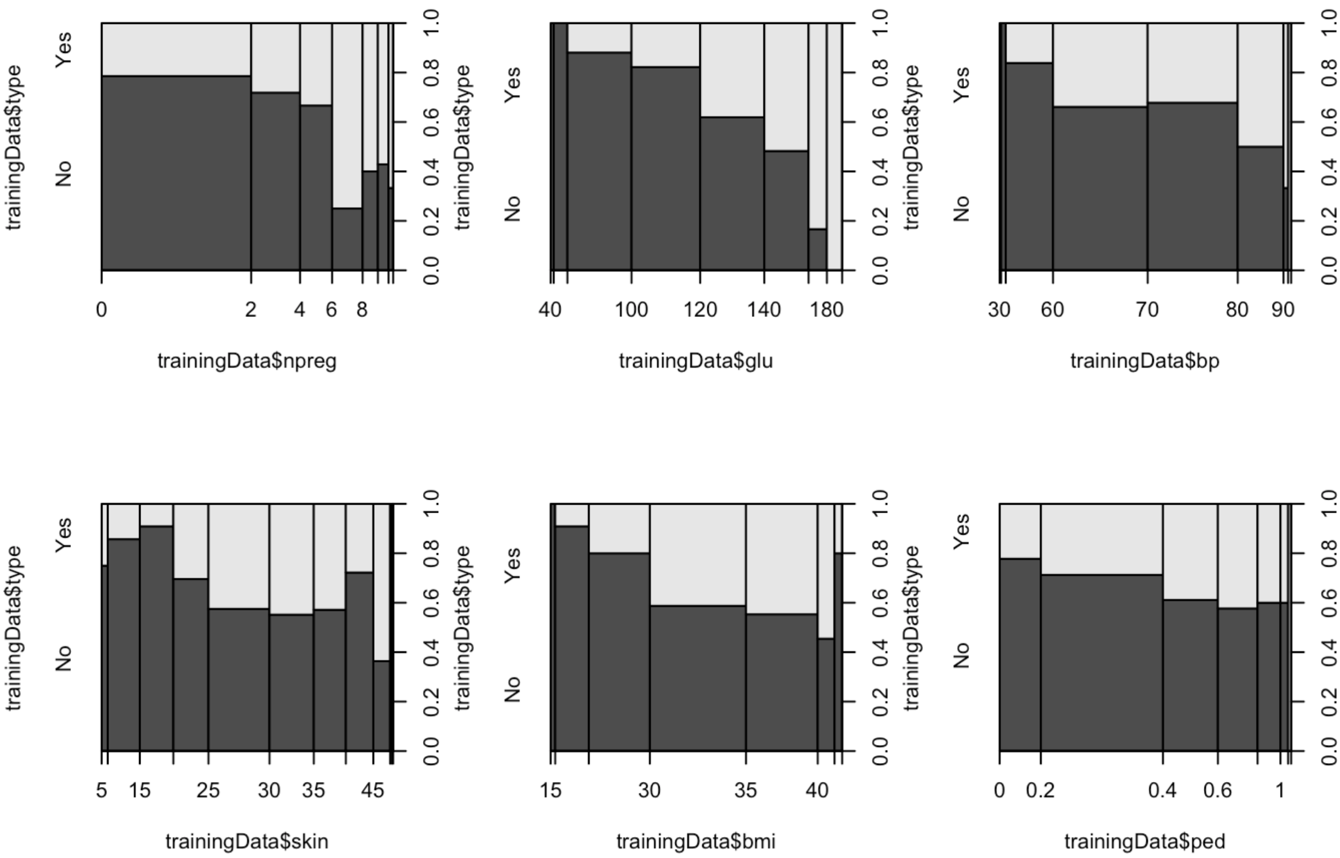
```
summary(testData$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   23.00   27.00   31.32   37.00   81.00
```

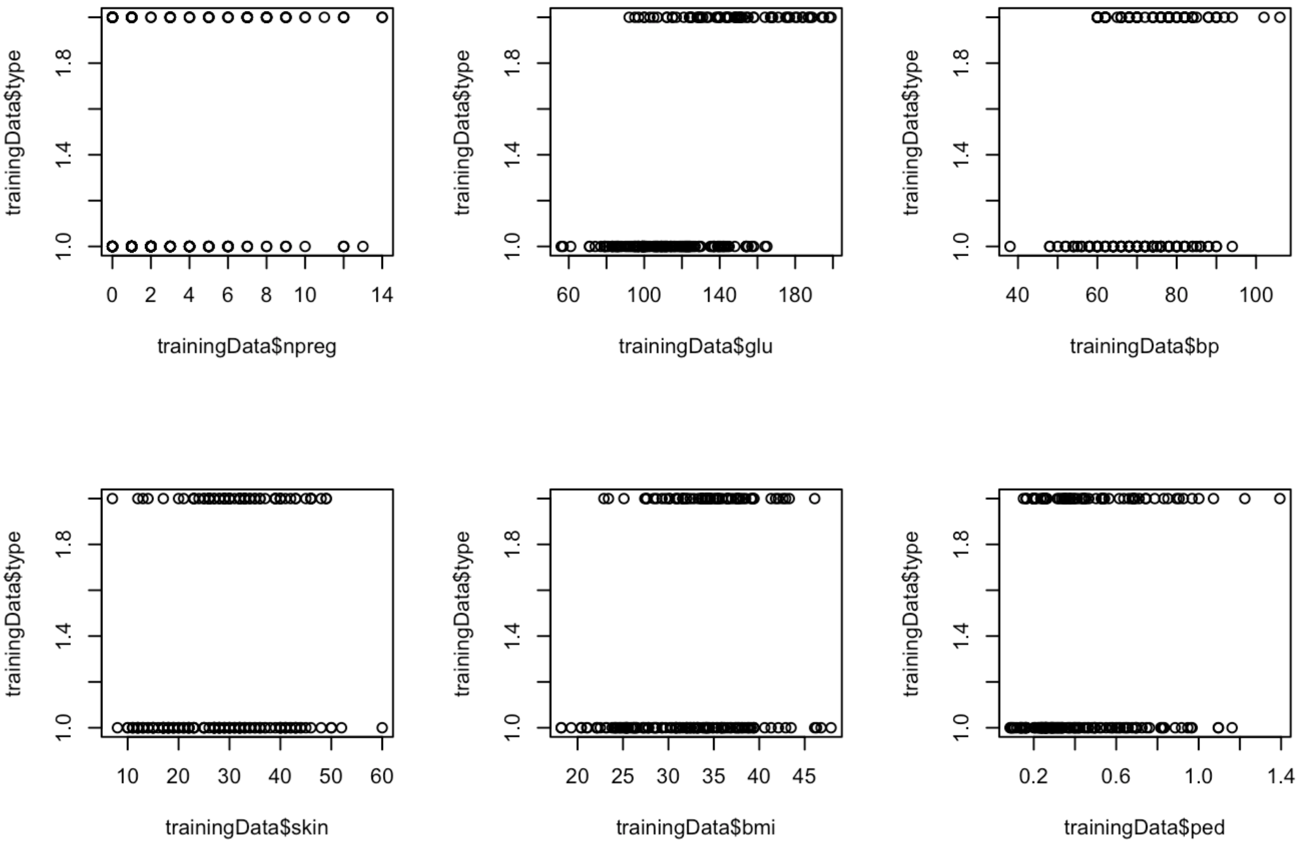
```
summary(testData$type)
```

```
##  No Yes
## 223 109
```

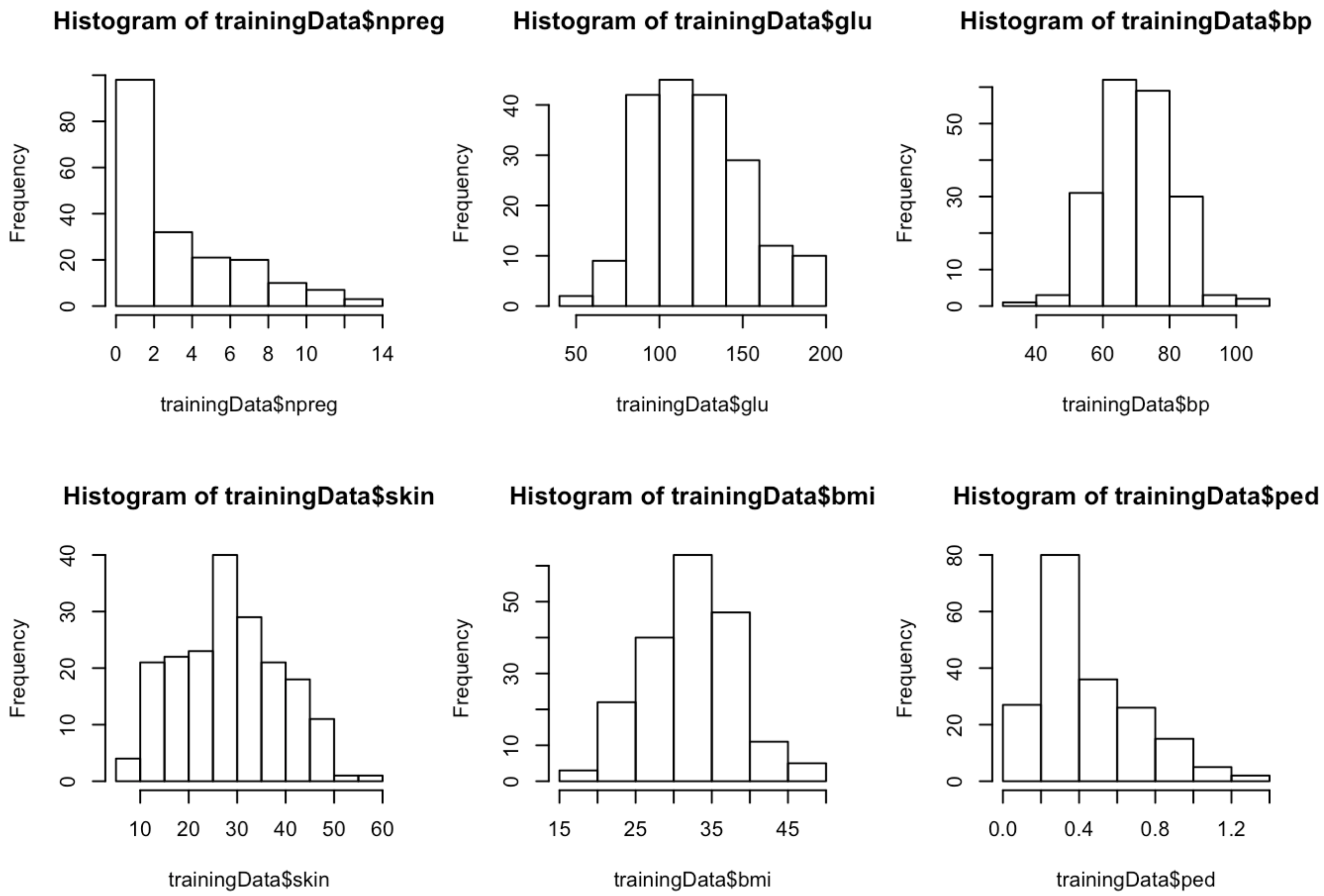
```
# Show at least one plot that shows the output variable (y)
# is correlated with some of the input variables X.

par(mfrow=c(2,3))
plot(trainingData$type~trainingData$npreg)
plot(trainingData$type~trainingData$glu)
plot(trainingData$type~trainingData$bp)
plot(trainingData$type~trainingData$skin)
plot(trainingData$type~trainingData$bmi)
plot(trainingData$type~trainingData$ped)
```




```
# or this
plot(trainingData$npreg,trainingData$type)
plot(trainingData$glu,trainingData$type)
plot(trainingData$bp,trainingData$type)
plot(trainingData$skin,trainingData$type)
plot(trainingData$bmi,trainingData$type)
plot(trainingData$ped,trainingData$type)
```



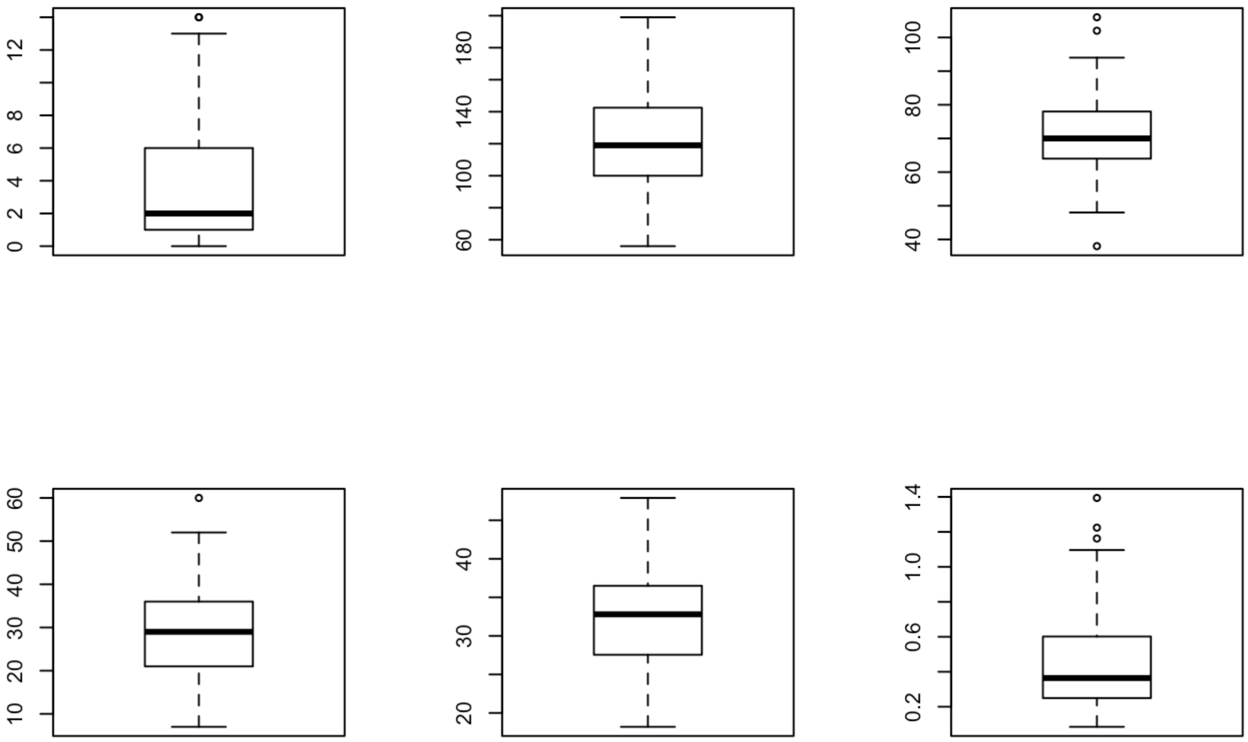

```
# histogram of 6continuous varicables
hist(trainingData$npreg)
hist(trainingData$glu)
hist(trainingData$bp)
hist(trainingData$skin)
hist(trainingData$bmi)
hist(trainingData$ped)
```



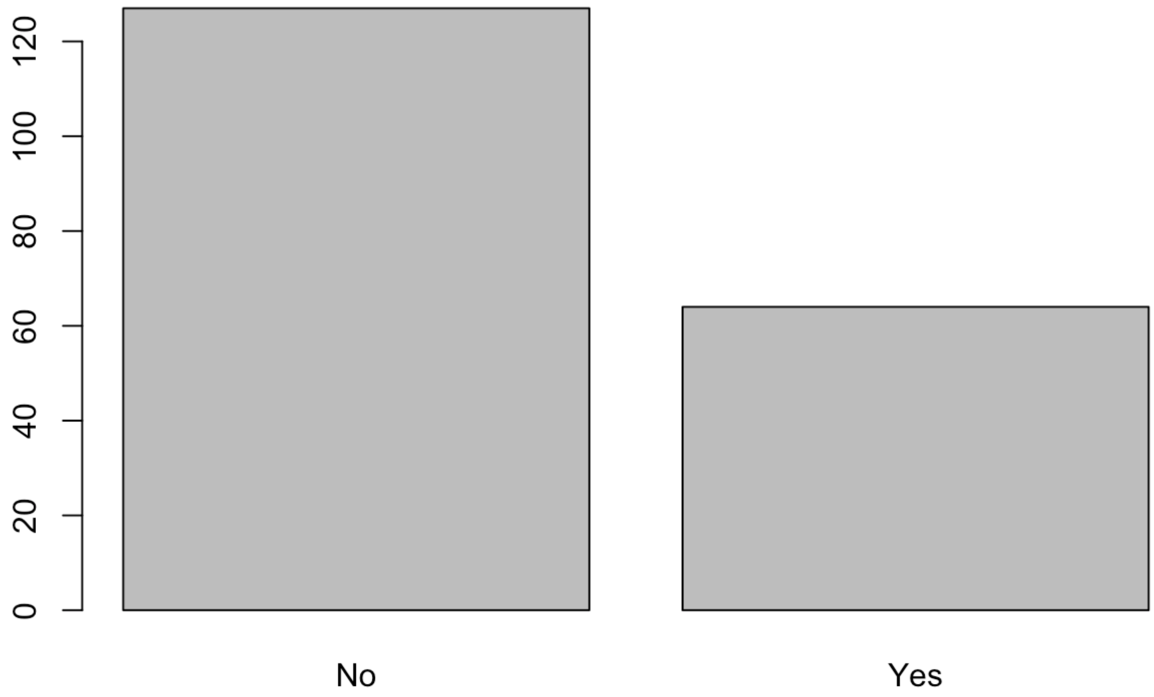

```
v <- readline("please enter any key to next: ")
```

```
## please enter any key to next:
```

```
# boxblot of 6continuous varicables
boxplot(trainingData$npreg)
boxplot(trainingData$glu)
boxplot(trainingData$bp)
boxplot(trainingData$skin)
boxplot(trainingData$bmi)
boxplot(trainingData$ped)
```



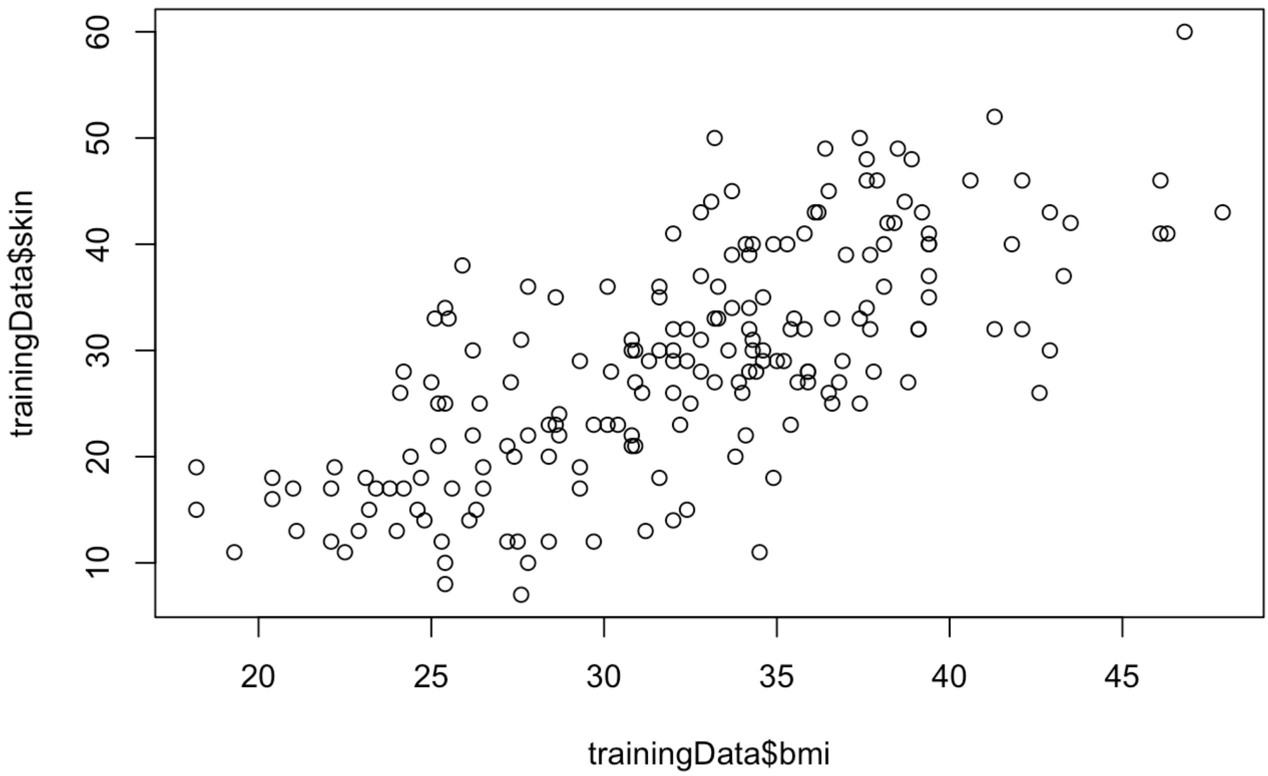

```
# plot of categorical variable(type)
par(mfrow=c(1,1))
plot(trainingData$type)
```




```
# analyze one interesting relationship between the selected two features
# I choose the skin(triceps skin fold thickness) and bmi(body mass index)

# simple plot
plot(trainingData$skin~trainingData$bmi)

# linear regression
fig = lm(trainingData$skin~trainingData$bmi)
summary(fig)
```

```
##
## Call:
## lm(formula = trainingData$skin ~ trainingData$bmi)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.6544  -5.1752  -0.7373   4.7666  19.9595
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -11.17421    2.91944  -3.828 0.000176 ***
## trainingData$bmi   1.24141    0.08885  13.972  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.518 on 189 degrees of freedom
## Multiple R-squared:  0.5081, Adjusted R-squared:  0.5055
## F-statistic: 195.2 on 1 and 189 DF,  p-value: < 2.2e-16
```

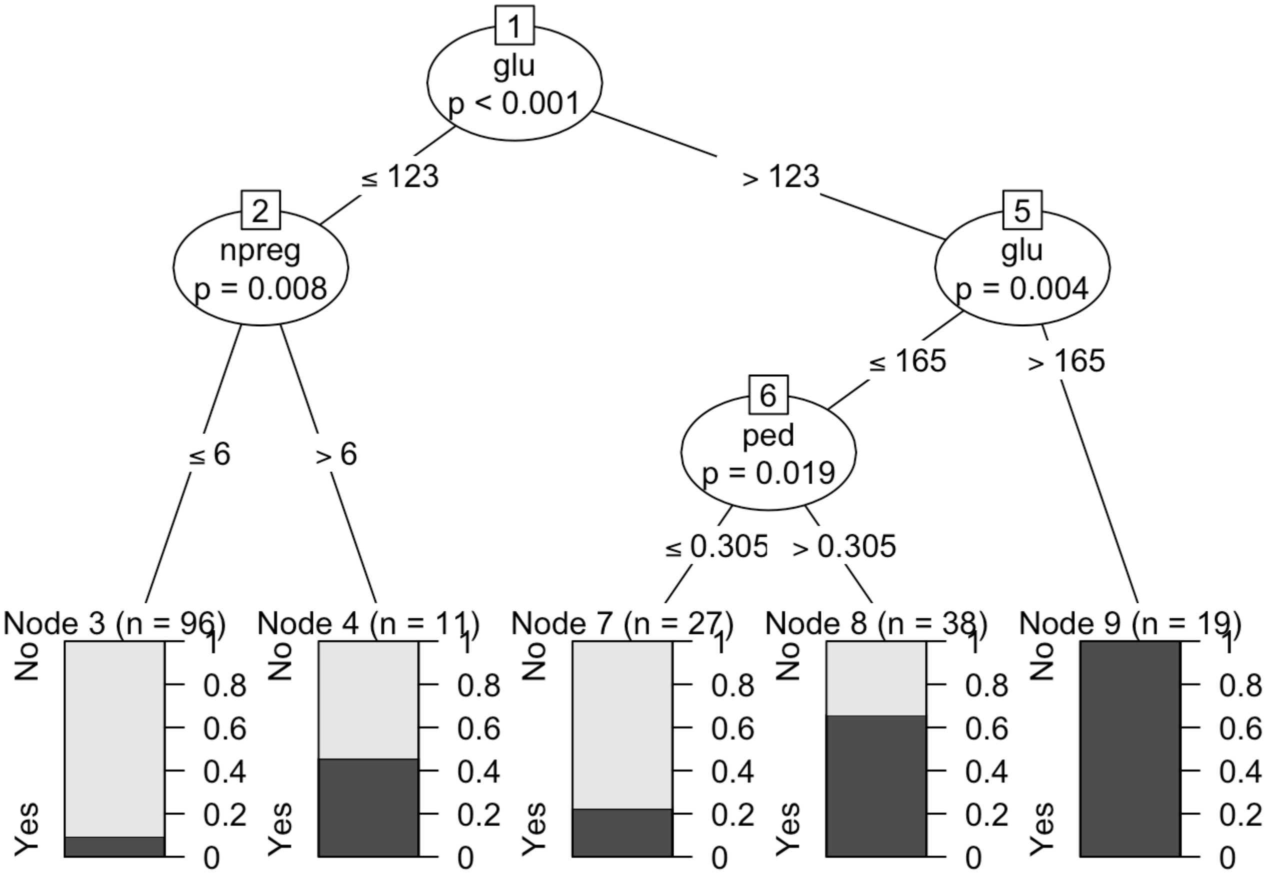
```
# remove outlier(index 157 data)
#trainingData <- trainingData[-c(157),]
#nrow(trainingData)

# plot and linear regression on the new dataset
plot(trainingData$skin~trainingData$bmi)
```




```
fig = lm(trainingData$skin~trainingData$bmi)
summary(fig)
```

```
##
## Call:
## lm(formula = trainingData$skin ~ trainingData$bmi)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.6544  -5.1752  -0.7373   4.7666  19.9595
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -11.17421    2.91944  -3.828 0.000176 ***
## trainingData$bmi   1.24141    0.08885  13.972  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.518 on 189 degrees of freedom
## Multiple R-squared:  0.5081, Adjusted R-squared:  0.5055
## F-statistic: 195.2 on 1 and 189 DF,  p-value: < 2.2e-16
```

```
# restore the original data
#trainingData = Pima.tr

# make Decision Tree
training_data <- ctree(type ~ .,data=trainingData)
plot(training_data)
```




```
# Calculate training error by DT
pre_training = predict(training_data, trainingData)
trainingError = mean(pre_training!=trainingData$type)
print(trainingError)
```

```
## [1] 0.1727749
```

```
# Calculate test error by DT
pre_test = predict(training_data, testData)
testError = mean(pre_test!=testData$type)
print(testError)
```

```
## [1] 0.2590361
```

```
# Naive Bayes Classification
training_data <- naiveBayes(type~.,data=trainingData)

# Calculate training error by NBC
pre_training = predict(training_data, trainingData)
trainingError = mean(pre_training!=trainingData$type)
print(trainingError)
```

```
## [1] 0.2041885
```

```
#confusion matrix of training data
confusionMatrix <- table(pre_training, trainingData$type)
print(confusionMatrix)
```

```
##
## pre_training  No Yes
##          No  107  19
##          Yes  20  45
```

```
# Calculate test error by NBC
pre_test = predict(training_data, testData)
testError = mean(pre_test!=testData$type)
print(testError)
```

```
## [1] 0.2379518
```

```
#confusion matrix of training data
confusionMatrix <- table(pre_test, testData$type)
print(confusionMatrix)
```

```
##
## pre_test  No Yes
##      No  182  38
##      Yes  41  71
```

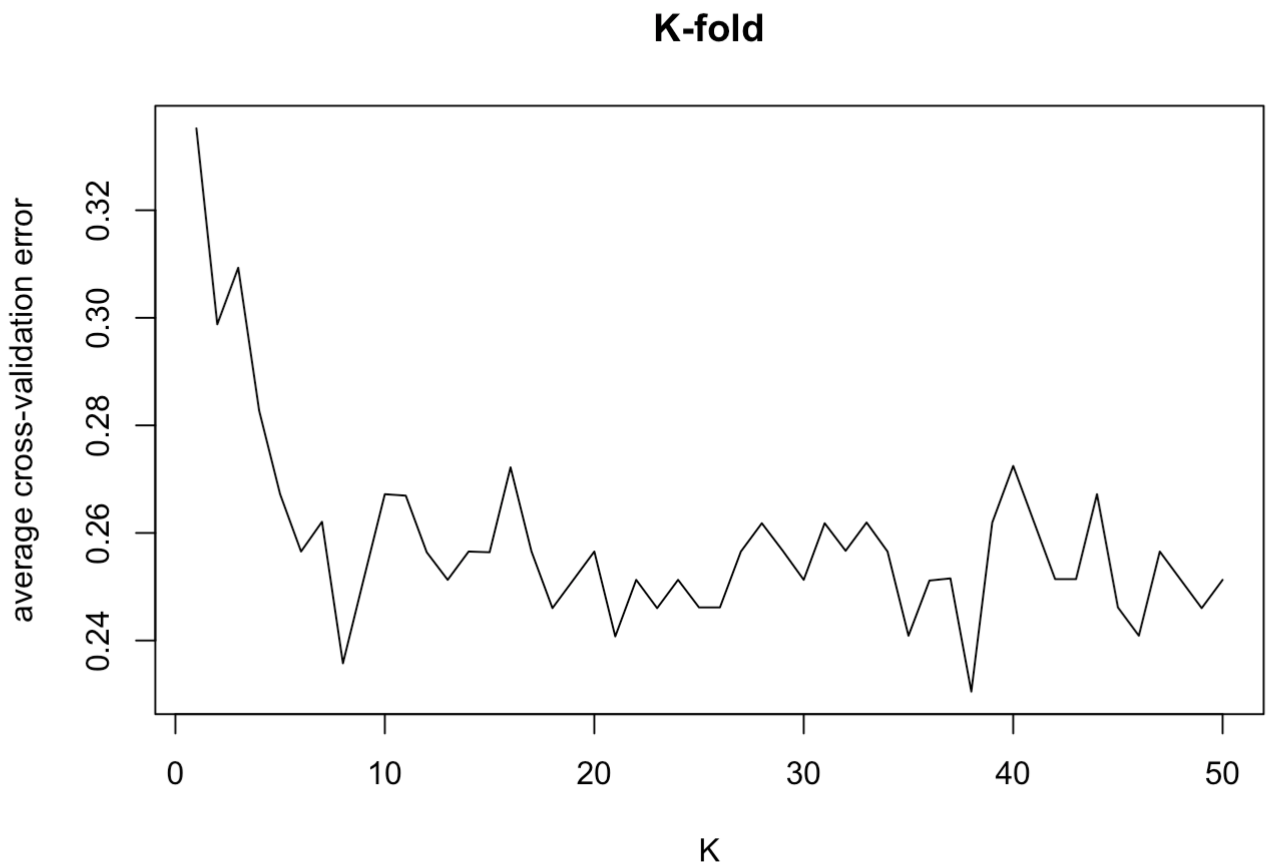
```
# for cross validation


trainingAsNumeric <- trainingData
trainingAsNumeric$npreg <- as.numeric(as.character(trainingData$npreg))
trainingAsNumeric$glu <- as.numeric(as.character(trainingData$glu))
trainingAsNumeric$bp <- as.numeric(as.character(trainingData$bp))
trainingAsNumeric$skin <- as.numeric(as.character(trainingData$skin))
trainingAsNumeric$bmi <- as.numeric(as.character(trainingData$bmi))
trainingAsNumeric$ped <- as.numeric(as.character(trainingData$ped))
trainingAsNumeric$age <- as.numeric(as.character(trainingData$age))
trainingAsNumeric$type <- as.numeric(as.factor(trainingData$type))

normalize<-function(x){
  return((x-min(x))/(max(x)-min(x)))
}

nomalizedTrainingData <- as.data.frame(lapply(trainingData[1:7],normalize))
nomalizedTestData <- as.data.frame(lapply(testData[1:7],normalize))

set.seed(1)
# 5-fold cross validation
idx <- createFolds(trainingData$type, k=5)
print(sapply(idx, length))
```

```
## Fold1 Fold2 Fold3 Fold4 Fold5
##    38    38    38    38    39
```

```
set.seed(1)
ks <- 1:50
res <- sapply(ks, function(k) {
  res.k <- sapply(seq_along(idx), function(i) {
    pred <- knn(train = nomalizedTrainingData[ -idx[[i]] ,] , test = nomalizedTrainin
gData[ idx[[i]], ], cl=trainingData$type[-idx[[i]] ], k=k)
    mean(trainingData$type[idx[[i]]]!=pred)
  })
  mean(res.k)
})

print(res)
```

```
##  [1] 0.3352227 0.2987854 0.3093117 0.2827260 0.2672065 0.2565452 0.2620783
##  [8] 0.2357625 0.2515520 0.2672065 0.2669366 0.2564103 0.2512821 0.2565452
## [15] 0.2564103 0.2721997 0.2565452 0.2460189 0.2512821 0.2565452 0.2407557
## [22] 0.2512821 0.2460189 0.2512821 0.2461538 0.2461538 0.2565452 0.2618084
## [29] 0.2566802 0.2512821 0.2618084 0.2566802 0.2619433 0.2565452 0.2408907
## [36] 0.2511471 0.2515520 0.2304993 0.2619433 0.2724696 0.2619433 0.2514170
## [43] 0.2514170 0.2672065 0.2461538 0.2408907 0.2565452 0.2512821 0.2460189
## [50] 0.2512821
```

```
plot(ks, res, type="l",ylab="average cross-validation error", xlab="K", main="K-
fold")
```




```
# What is the Best Parameter K in experiments?
bestK<-which.min(res)
print(bestK)
```

```
## [1] 38
```

```
bestPredict <- knn(train=nomalizedTrainingData, test=nomalizedTestData, cl=trainingDa
ta$type, k=bestK)
#final cross-validation error
min(res)
```

```
## [1] 0.2304993
```

```
#final classification error
mean(testData$type != bestPredict)
```

```
## [1] 0.2771084
```