Practicum 3

Smit Patil

—Problem 1—

1. Download the data set Bank Marketing Data Set. Note that the data file does not contain header names; you may wish to add those. The description of each column can be found in the data set explanation. Use the bank-additional-full.csv data set. Select an appropriate subset for testing. Use bank-additional.csv if your computer cannot process the full data set.

```
#Importing bnak dataset
bank.data <- read.csv("bank-additional-full.csv", sep = ";", stringsAsFactors = T)

#Looking at the head and the structure of the bank data
head(bank.data)
```

```
##   age       job marital   education default housing loan   contact month
## 1  56 housemaid married    basic.4y      no      no   no telephone   may
## 2  57  services married high.school unknown      no   no telephone   may
## 3  37  services married high.school      no     yes   no telephone   may
## 4  40    admin. married    basic.6y      no      no   no telephone   may
## 5  56  services married high.school      no      no  yes telephone   may
## 6  45  services married    basic.9y unknown      no   no telephone   may
##   day_of_week duration campaign pdays previous    poutcome emp.var.rate
## 1         mon      261        1   999        0 nonexistent          1.1
## 2         mon      149        1   999        0 nonexistent          1.1
## 3         mon      226        1   999        0 nonexistent          1.1
## 4         mon      151        1   999        0 nonexistent          1.1
## 5         mon      307        1   999        0 nonexistent          1.1
## 6         mon      198        1   999        0 nonexistent          1.1
##   cons.price.idx cons.conf.idx euribor3m nr.employed  y
## 1         93.994         -36.4     4.857        5191 no
## 2         93.994         -36.4     4.857        5191 no
## 3         93.994         -36.4     4.857        5191 no
## 4         93.994         -36.4     4.857        5191 no
## 5         93.994         -36.4     4.857        5191 no
## 6         93.994         -36.4     4.857        5191 no
```

```
str(bank.data)
```

```
## 'data.frame':    41188 obs. of  21 variables:
##  $ age           : int  56 57 37 40 56 45 59 41 24 25 ...
##  $ job           : Factor w/ 12 levels "admin.","blue-collar",..: 4 8 8 1 8 8 1 2 10 8 ...
##  $ marital       : Factor w/ 4 levels "divorced","married",..: 2 2 2 2 2 2 2 2 3 3 ...
##  $ education     : Factor w/ 8 levels "basic.4y","basic.6y",..: 1 4 4 2 4 3 6 8 6 4 ...
##  $ default       : Factor w/ 3 levels "no","unknown",..: 1 2 1 1 1 2 1 2 1 1 ...
```

```
##  $ housing       : Factor w/ 3 levels "no","unknown",..: 1 1 3 1 1 1 1 1 3 3 ...
##  $ loan          : Factor w/ 3 levels "no","unknown",..: 1 1 1 1 3 1 1 1 1 1 ...
##  $ contact       : Factor w/ 2 levels "cellular","telephone": 2 2 2 2 2 2 2 2 2 2 ...
##  $ month         : Factor w/ 10 levels "apr","aug","dec",..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ day_of_week   : Factor w/ 5 levels "fri","mon","thu",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ duration      : int  261 149 226 151 307 198 139 217 380 50 ...
##  $ campaign      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays         : int  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome      : Factor w/ 3 levels "failure","nonexistent",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ emp.var.rate  : num  1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ cons.price.idx: num  94 94 94 94 94 ...
##  $ cons.conf.idx : num  -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 ...
##  $ euribor3m     : num  4.86 4.86 4.86 4.86 4.86 ...
##  $ nr.employed   : num  5191 5191 5191 5191 5191 ...
##  $ y             : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```r
# Summary of bank data
summary(bank.data)
```

```
##       age                 job            marital
##  Min.   :17.00   admin.     :10422   divorced: 4612
##  1st Qu.:32.00   blue-collar: 9254   married :24928
##  Median :38.00   technician : 6743   single  :11568
##  Mean   :40.02   services   : 3969   unknown :   80
##  3rd Qu.:47.00   management : 2924
##  Max.   :98.00   retired    : 1720
##                  (Other)    : 6156
##                education        default         housing           loan
##  university.degree  :12168   no     :32588   no     :18622   no     :33950
##  high.school        : 9515   unknown: 8597   unknown:  990   unknown:  990
##  basic.9y           : 6045   yes    :    3   yes    :21576   yes    : 6248
##  professional.course: 5243
##  basic.4y           : 4176
##  basic.6y           : 2292
##  (Other)            : 1749
##       contact          month       day_of_week    duration
##  cellular :26144   may    :13769   fri:7827    Min.   :   0.0
##  telephone:15044   jul    : 7174   mon:8514    1st Qu.: 102.0
##                    aug    : 6178   thu:8623    Median : 180.0
##                    jun    : 5318   tue:8090    Mean   : 258.3
##                    nov    : 4101   wed:8134    3rd Qu.: 319.0
##                    apr    : 2632               Max.   :4918.0
##                    (Other): 2016
##     campaign          pdays          previous            poutcome
##  Min.   : 1.000   Min.   :  0.0   Min.   :0.000   failure    : 4252
##  1st Qu.: 1.000   1st Qu.:999.0   1st Qu.:0.000   nonexistent:35563
##  Median : 2.000   Median :999.0   Median :0.000   success    : 1373
##  Mean   : 2.568   Mean   :962.5   Mean   :0.173
##  3rd Qu.: 3.000   3rd Qu.:999.0   3rd Qu.:0.000
##  Max.   :56.000   Max.   :999.0   Max.   :7.000
##
##   emp.var.rate      cons.price.idx  cons.conf.idx     euribor3m
##  Min.   :-3.40000   Min.   :92.20   Min.   :-50.8   Min.   :0.634
```

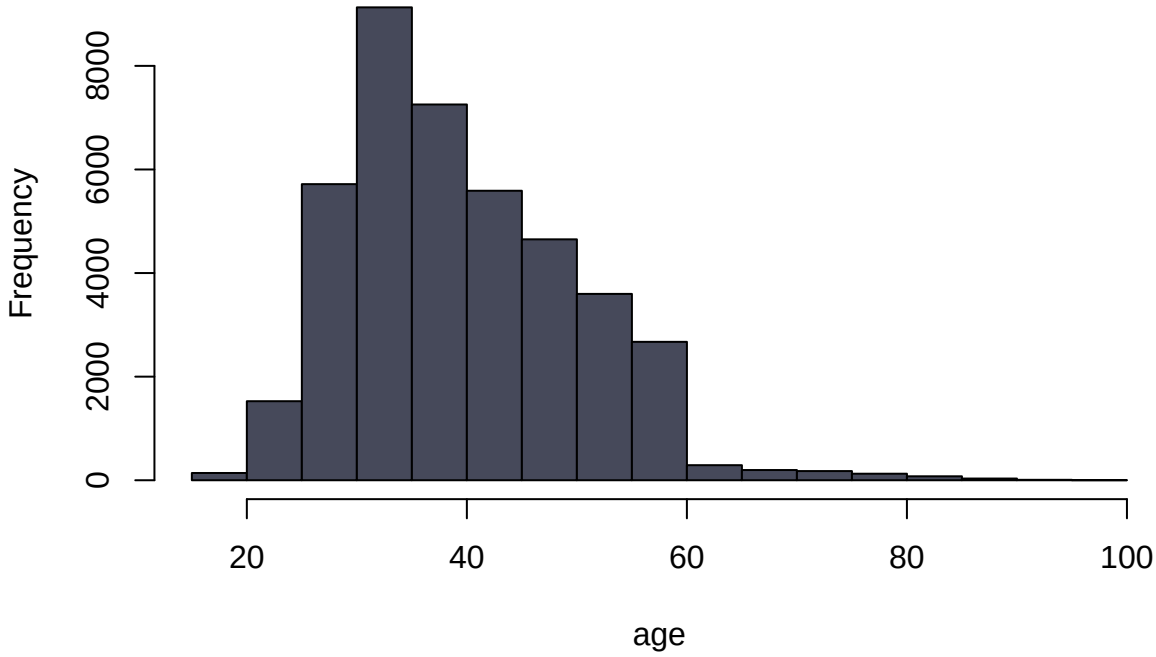
```
##  1st Qu.:-1.80000   1st Qu.:93.08   1st Qu.:-42.7   1st Qu.:1.344
##  Median : 1.10000   Median :93.75   Median :-41.8   Median :4.857
##  Mean   : 0.08189   Mean   :93.58   Mean   :-40.5   Mean   :3.621
##  3rd Qu.: 1.40000   3rd Qu.:93.99   3rd Qu.:-36.4   3rd Qu.:4.961
##  Max.   : 1.40000   Max.   :94.77   Max.   :-26.9   Max.   :5.045
##
##   nr.employed     y
##  Min.   :4964   no :36548
##  1st Qu.:5099   yes: 4640
##  Median :5191
##  Mean   :5167
##  3rd Qu.:5228
##  Max.   :5228
##
```

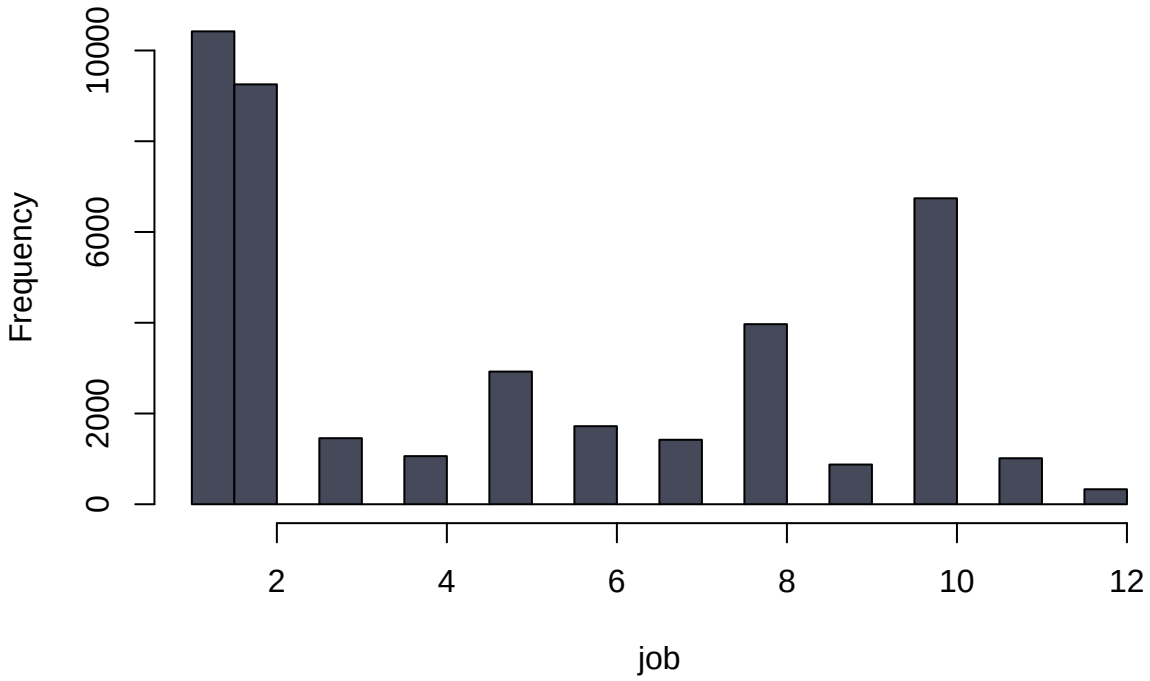
2. Explore the data set as you see fit and that allows you to get a sense of the data and get comfortable with it. Is there distributional skew in any of the features? Is there a need to apply a transform?

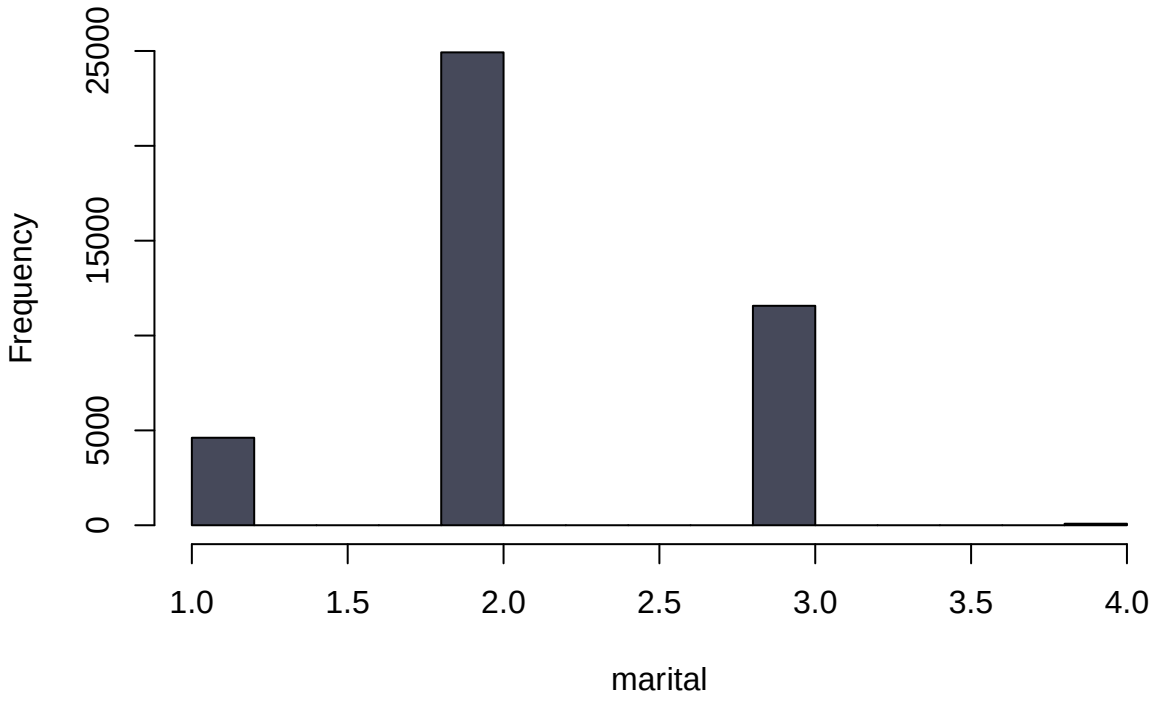
```
#Checking for any NA's in the dataset
anyNA(bank.data)
```

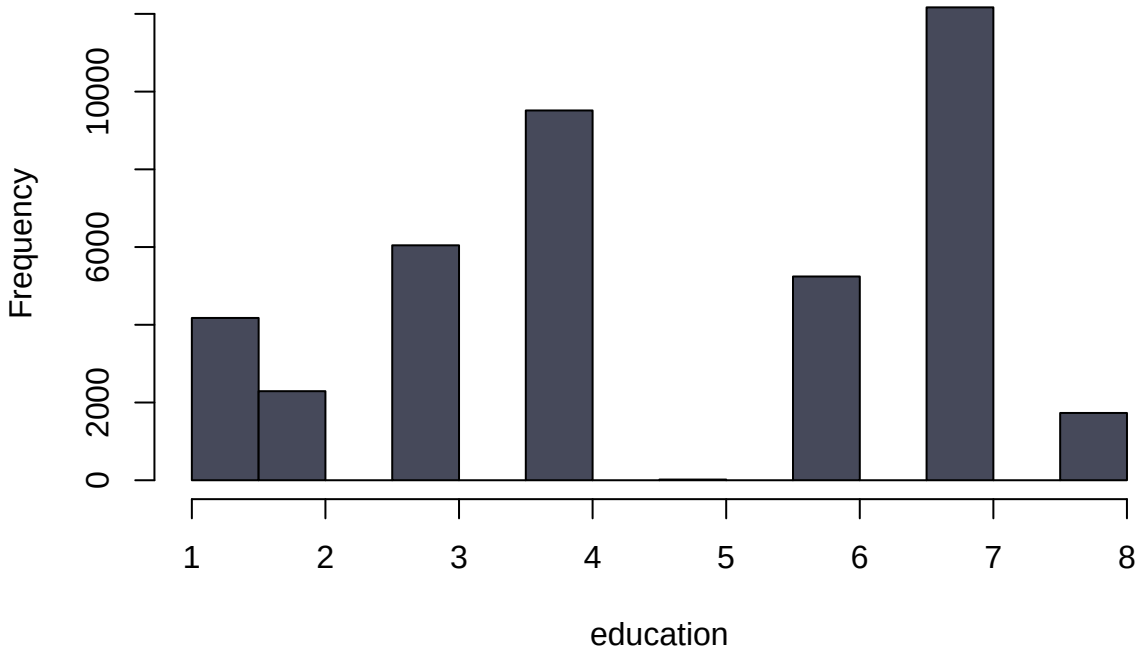
```
## [1] FALSE
```

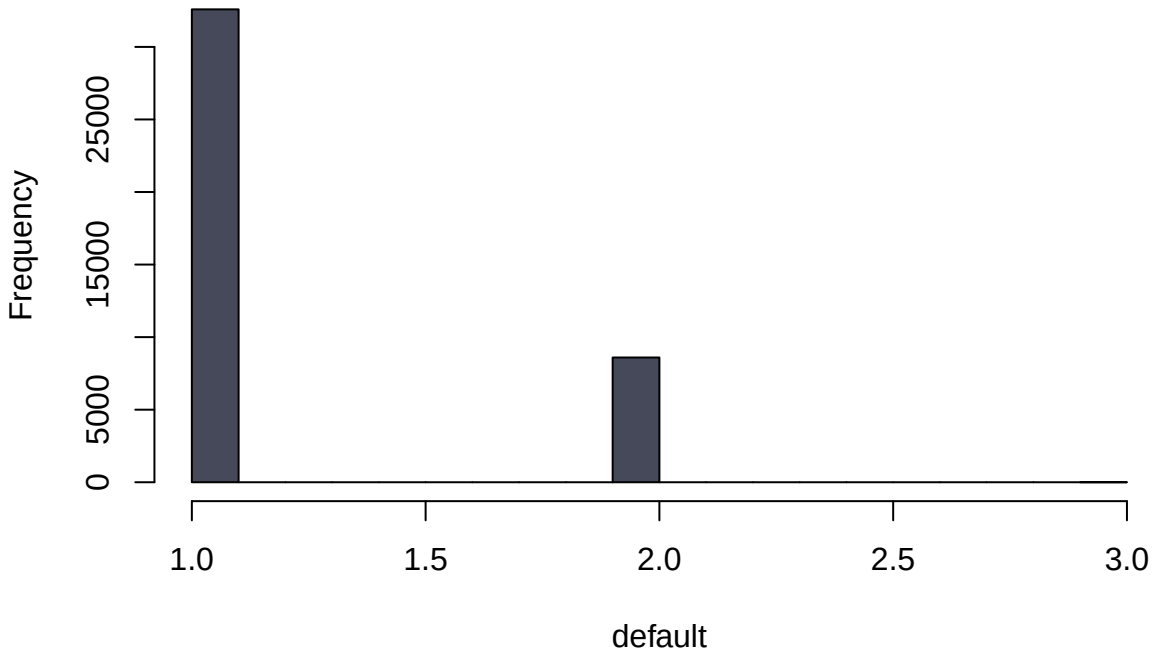
```
#Using for loop to print histograms of bank data
for(i in 1:ncol(bank.data))
  {
    sprintf("Histogram for: ", colnames(bank.data[i]))
    hist((as.numeric(bank.data[,i])), xlab = colnames(bank.data[i]), col = '#46495a', main = NULL)
 }
```

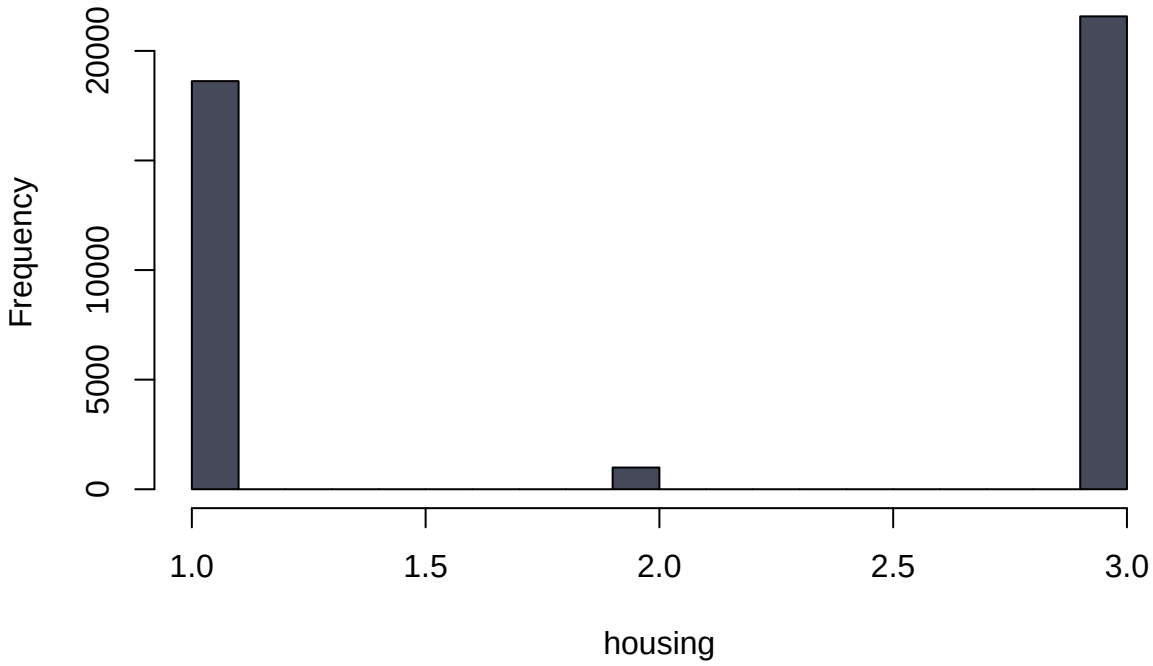

Frequency
0
2000
4000
6000
8000
20
40
60
80
100
age

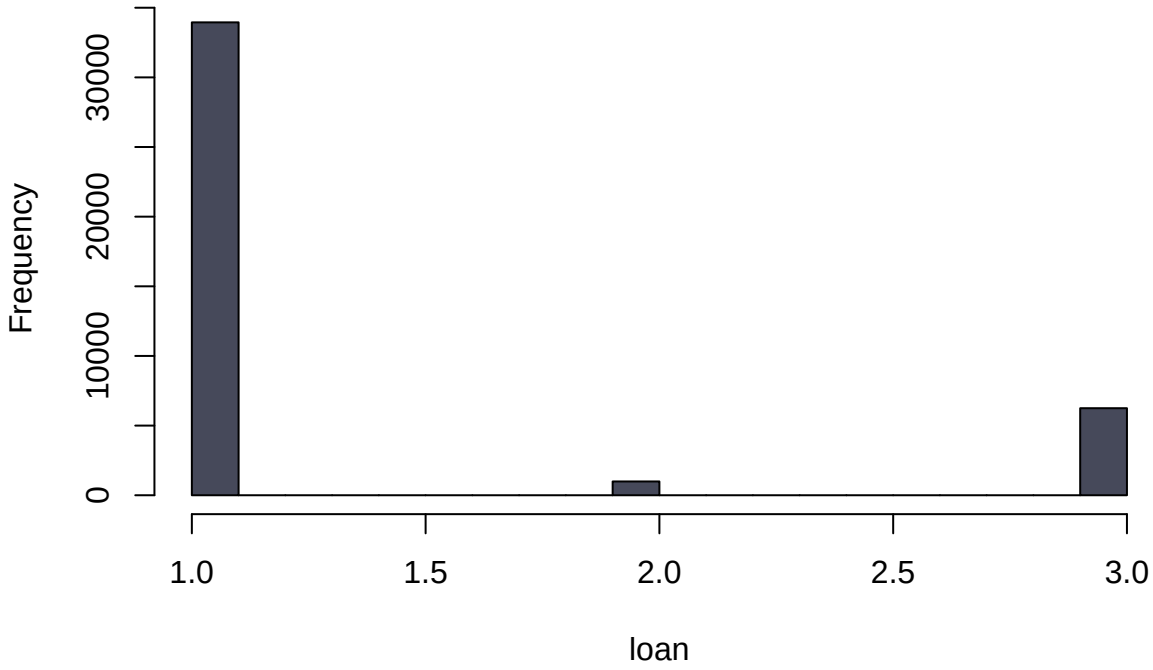

Frequency
0
2000
6000
10000
2
4
6
8
10
12
job

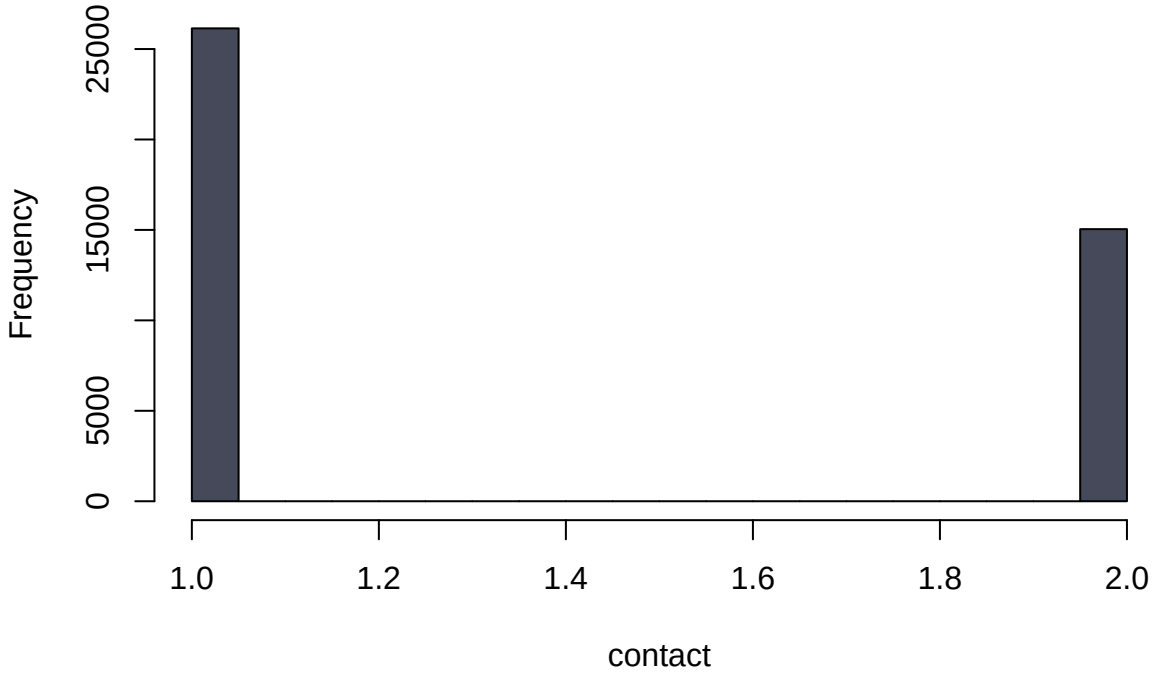

Frequency
0
5000
15000
25000
1.0
1.5
2.0
2.5
3.0
3.5
4.0
marital

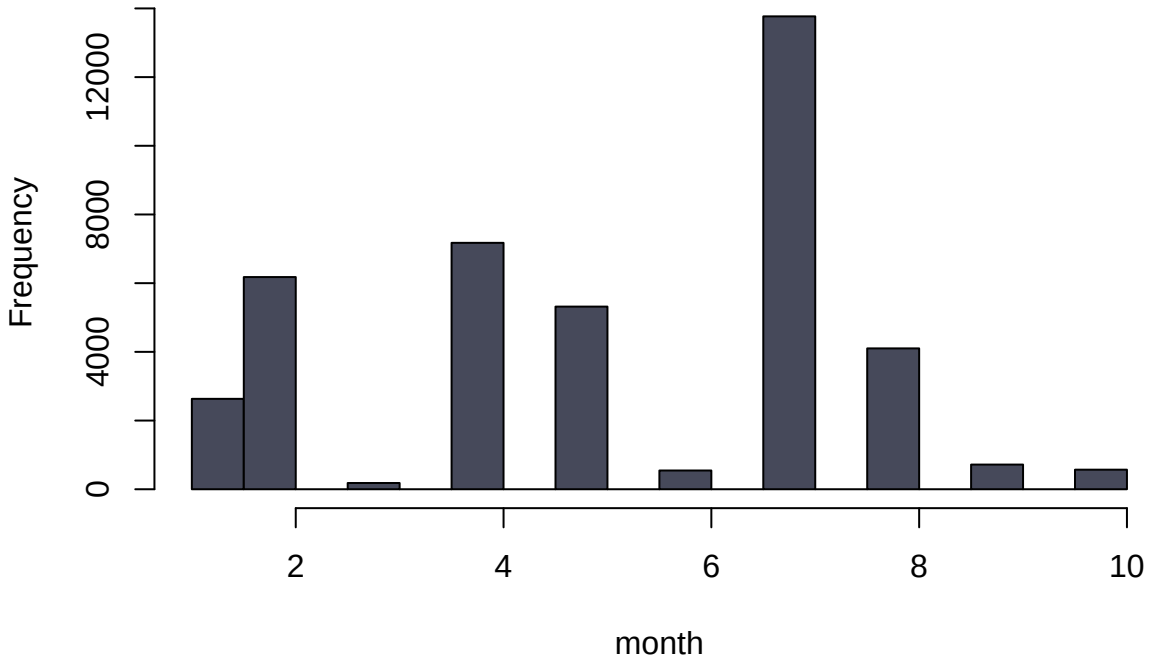

Frequency
0
2000
6000
10000
1
2
3
4
5
6
7
8
education

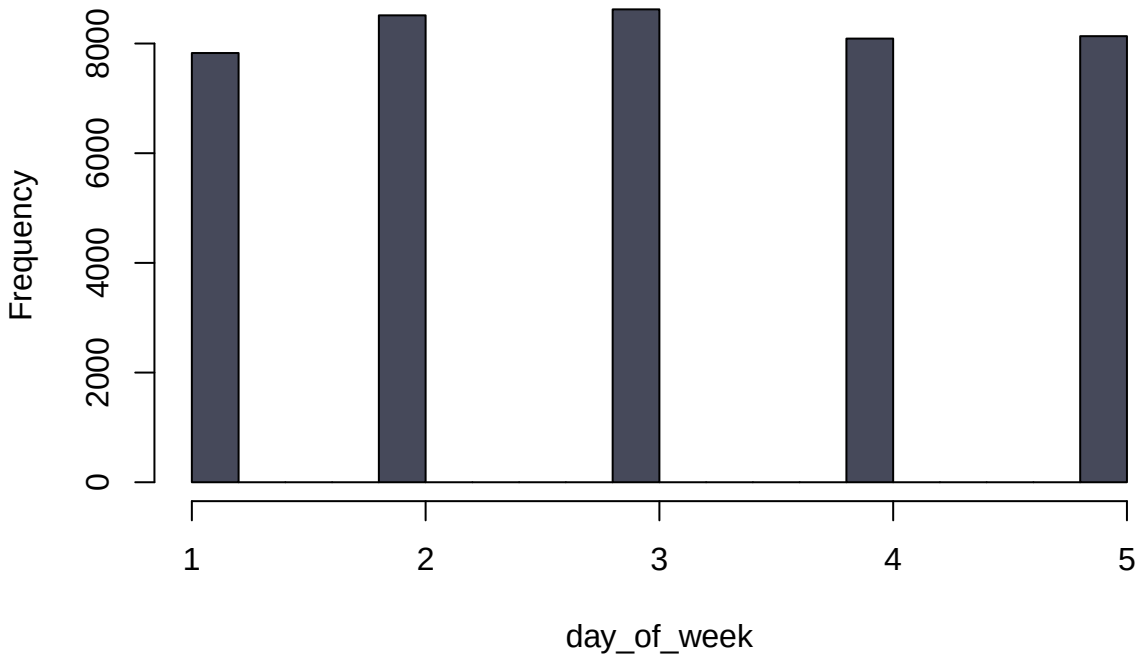

Frequency
0
5000
15000
25000
1.0
1.5
2.0
2.5
3.0
default

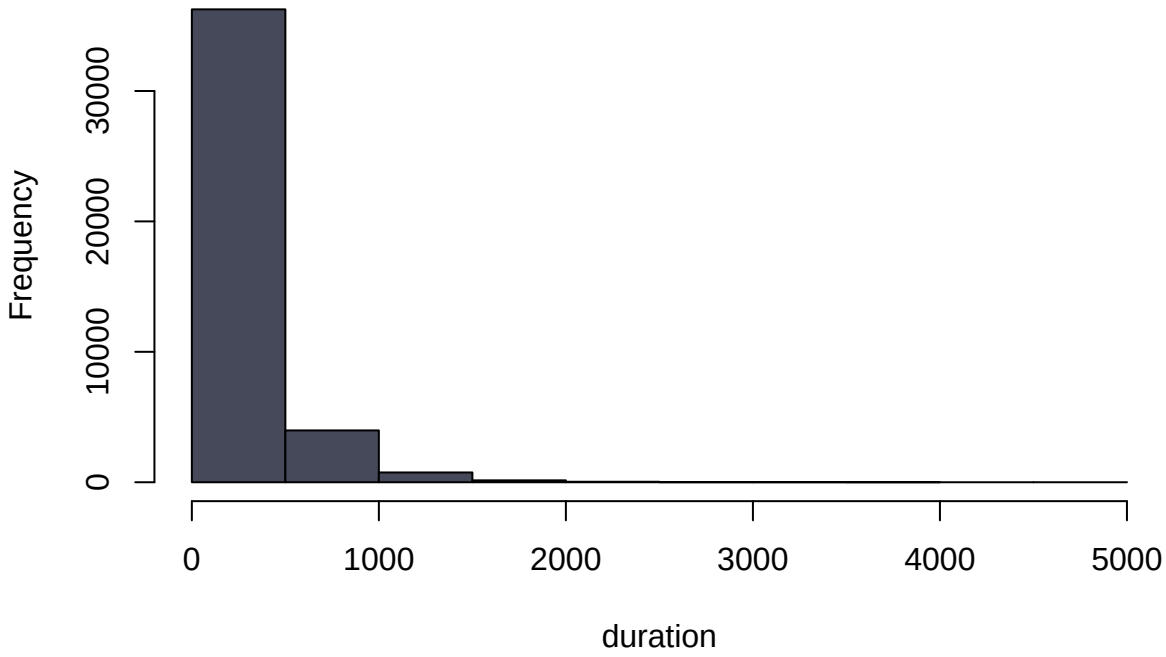

Frequency
0
5000
10000
20000
1.0
1.5
2.0
2.5
3.0
housing

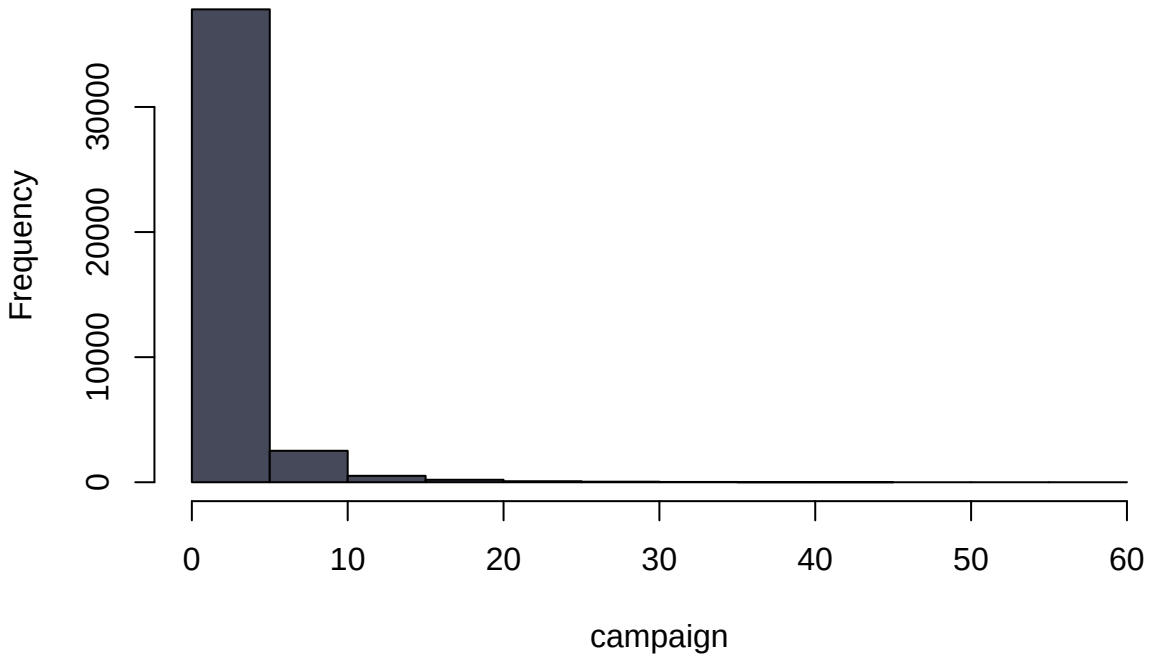

Frequency
0
10000
20000
30000
1.0
1.5
2.0
2.5
3.0
loan

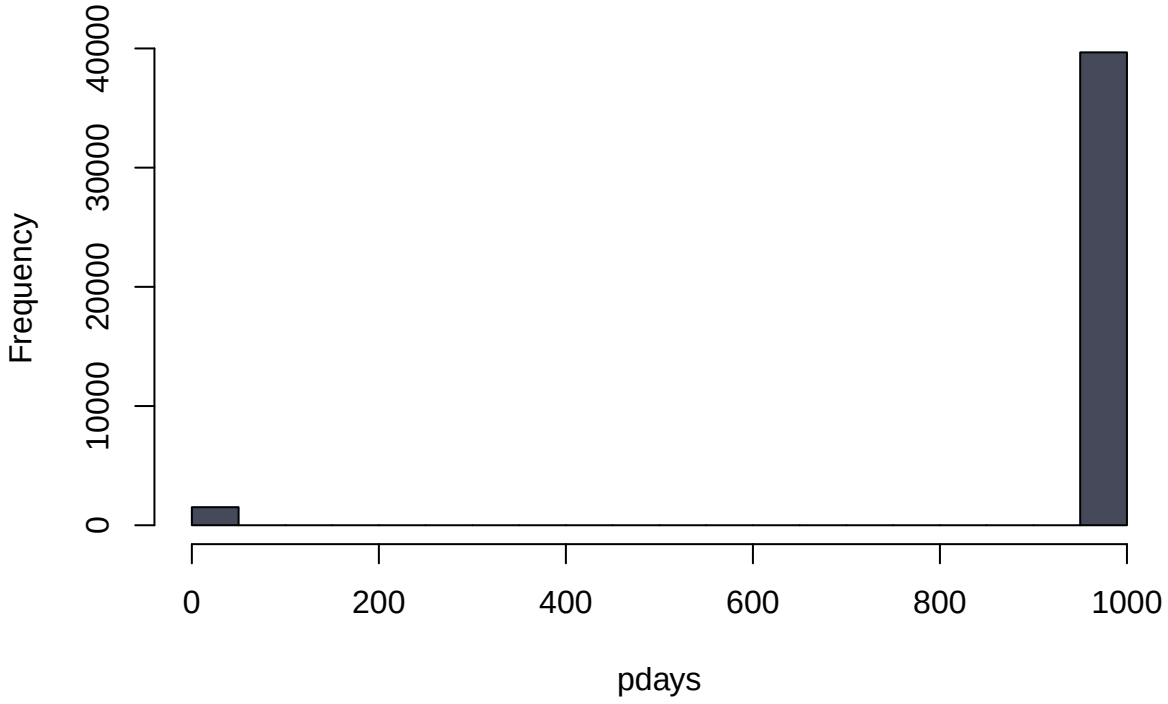

Frequency
0
5000
15000
25000
1.0
1.2
1.4
1.6
1.8
2.0
contact

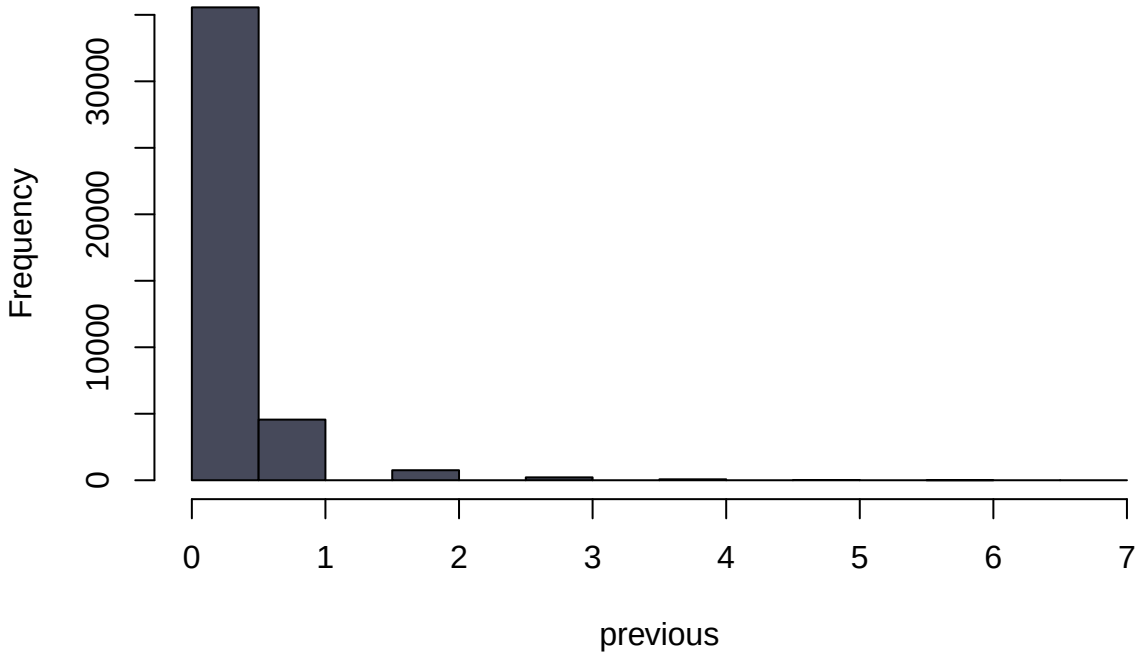

Frequency
0
4000
8000
12000
2
4
6
8
10
month

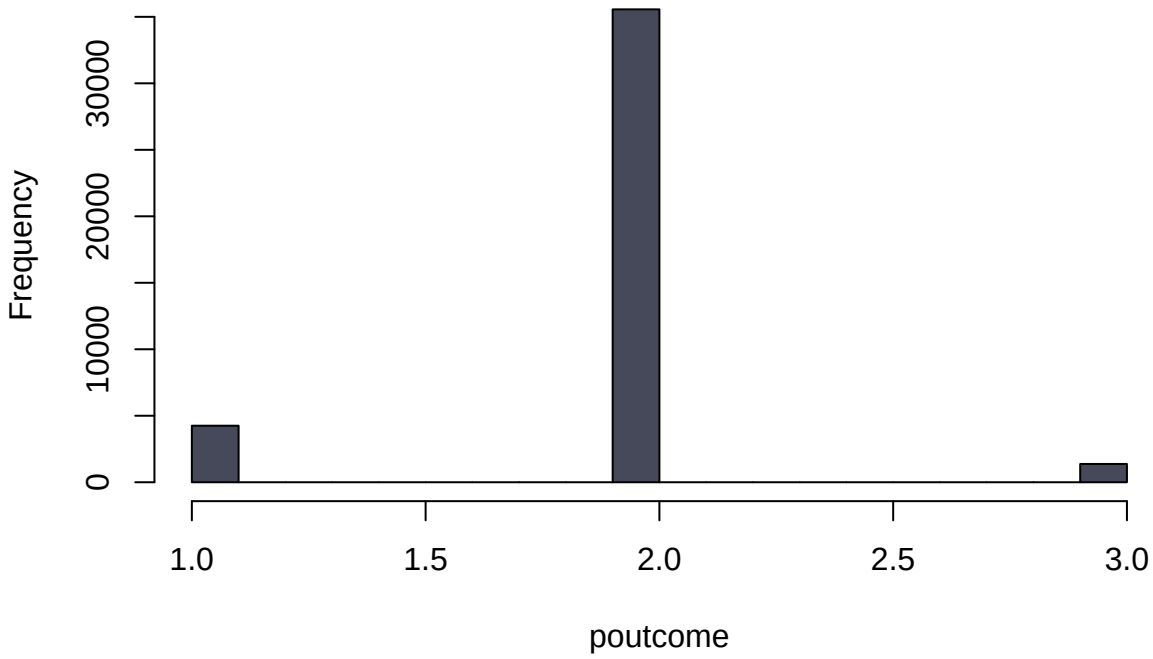

Frequency
0
2000
4000
6000
8000
1
2
3
4
5
day_of_week

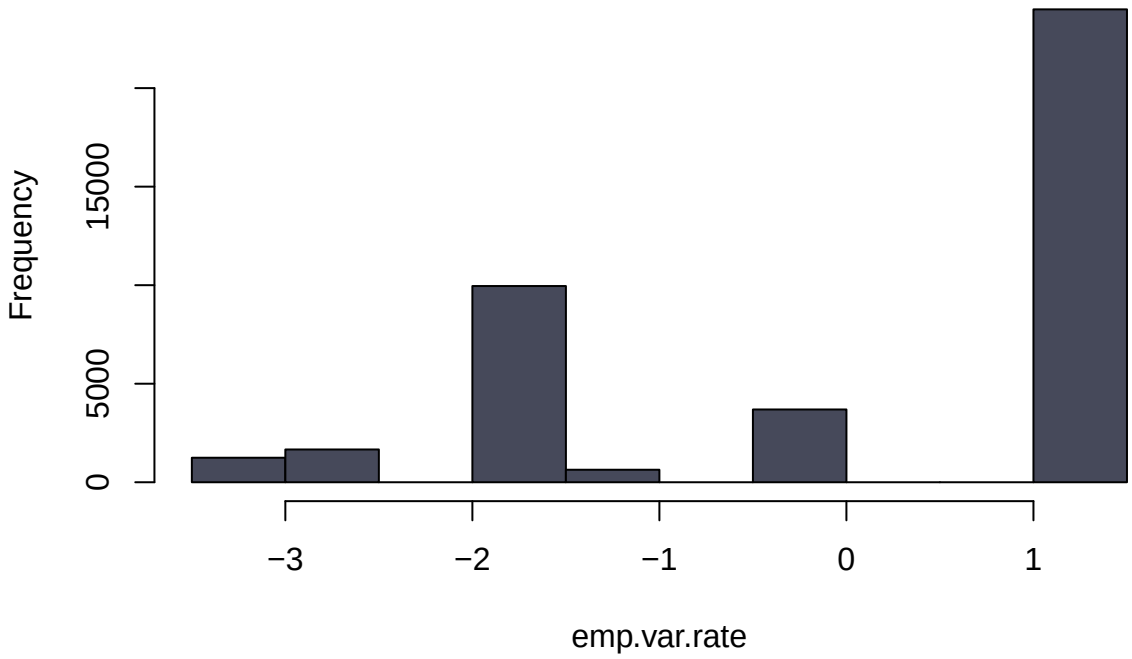

Frequency
0
10000
20000
30000
0
1000
2000
3000
4000
5000
duration

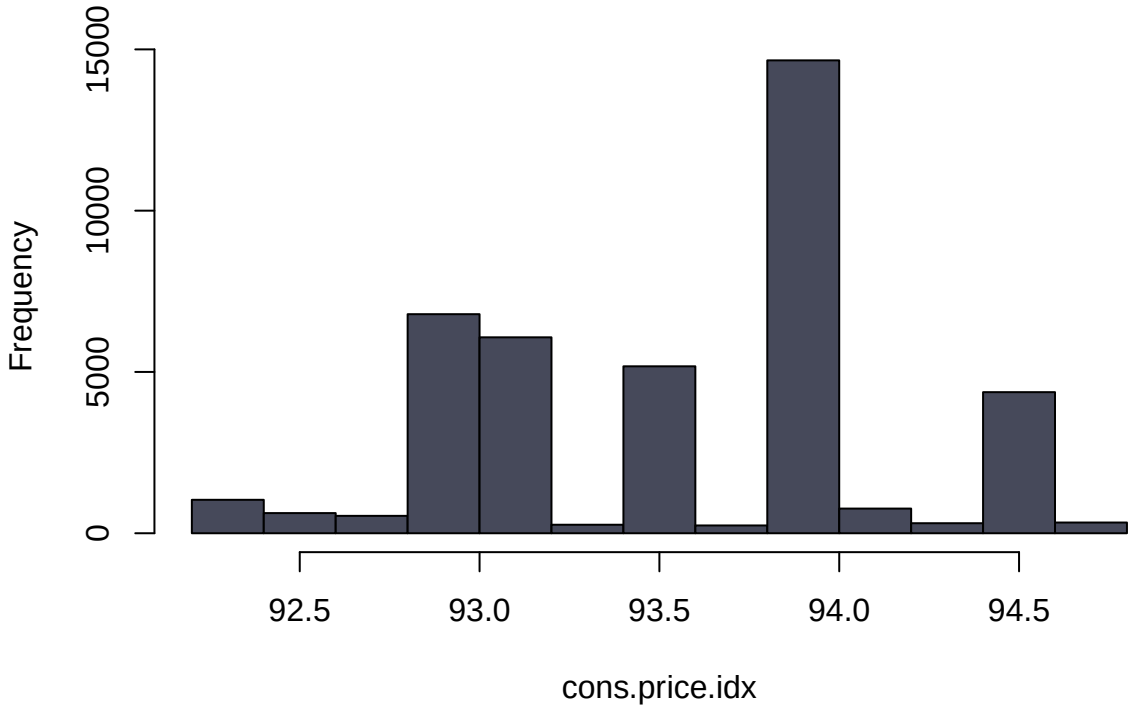

Frequency
0
10000
20000
30000
0
10
20
30
40
50
60
campaign

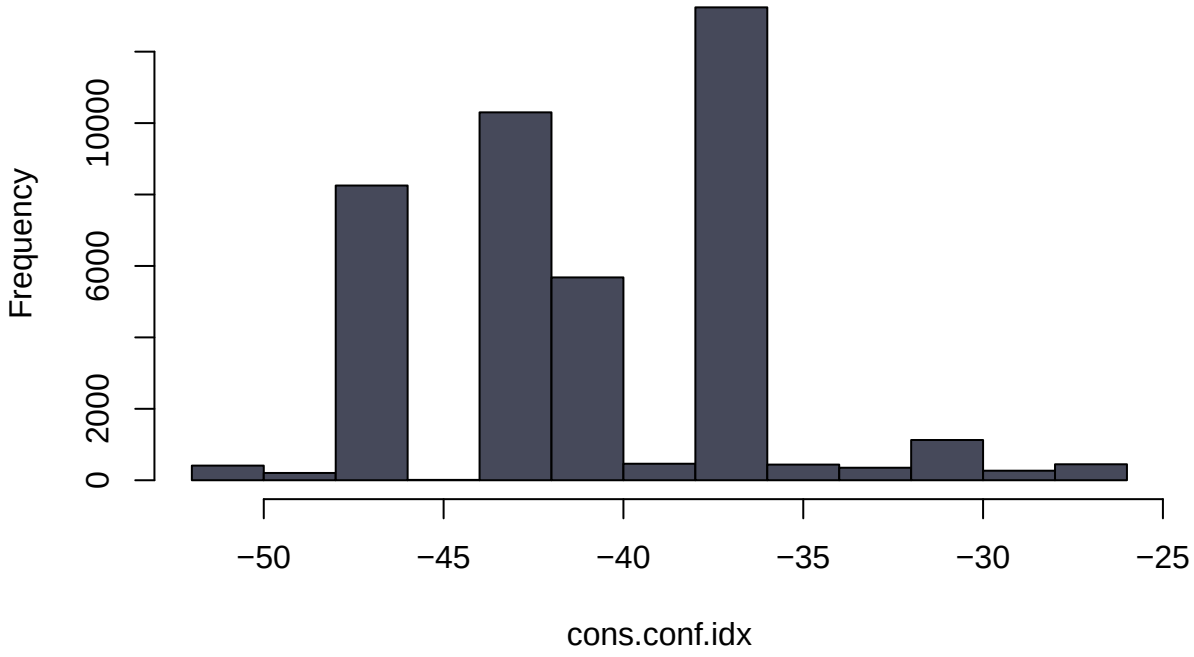

Frequency
0
10000
20000
30000
40000
0
200
400
600
800
1000
pdays

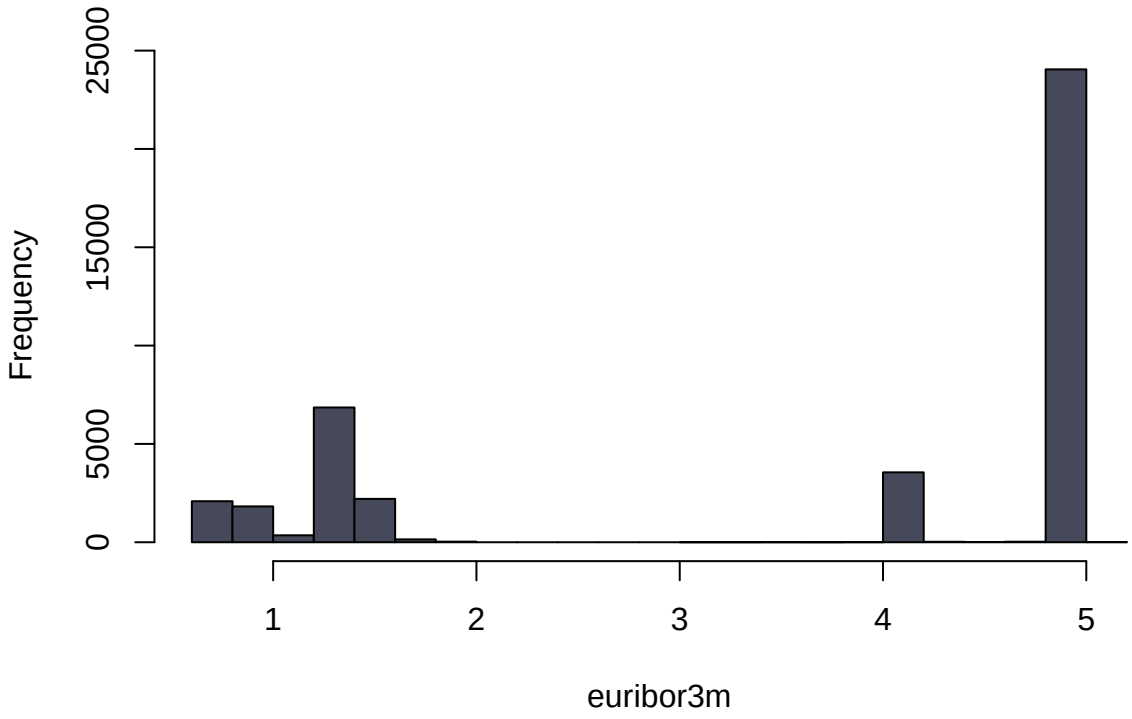

Frequency
0
10000
20000
30000
0
1
2
3
4
5
6
7
previous

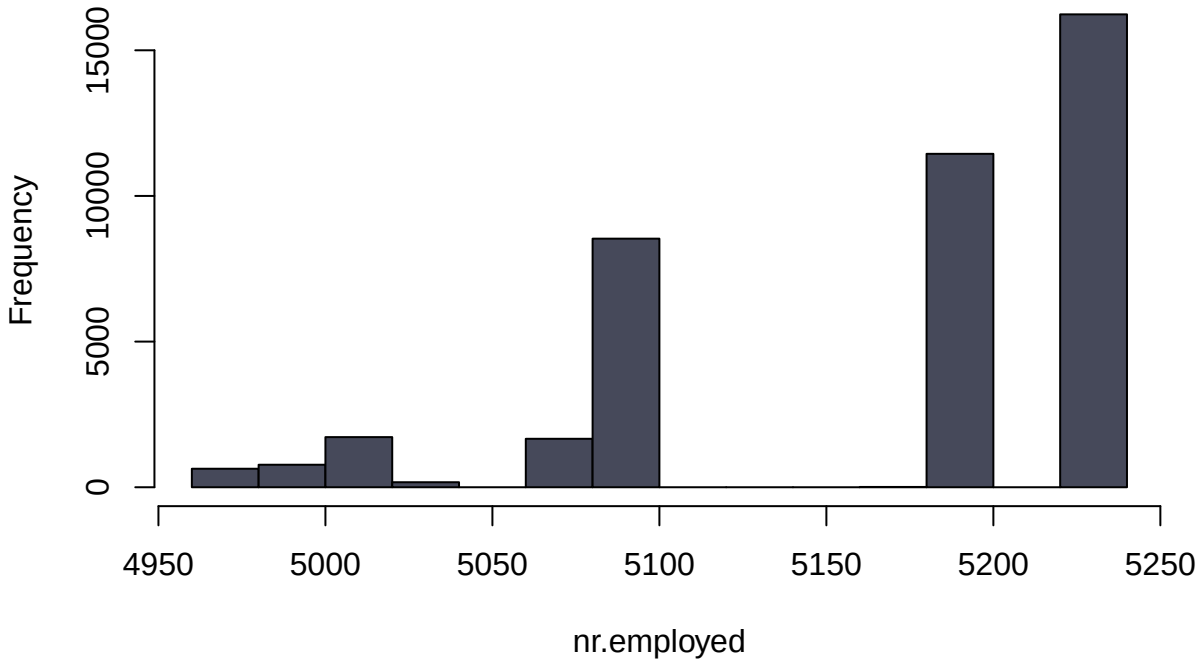

Frequency
0
10000
20000
30000
1.0
1.5
2.0
2.5
3.0
poutcome

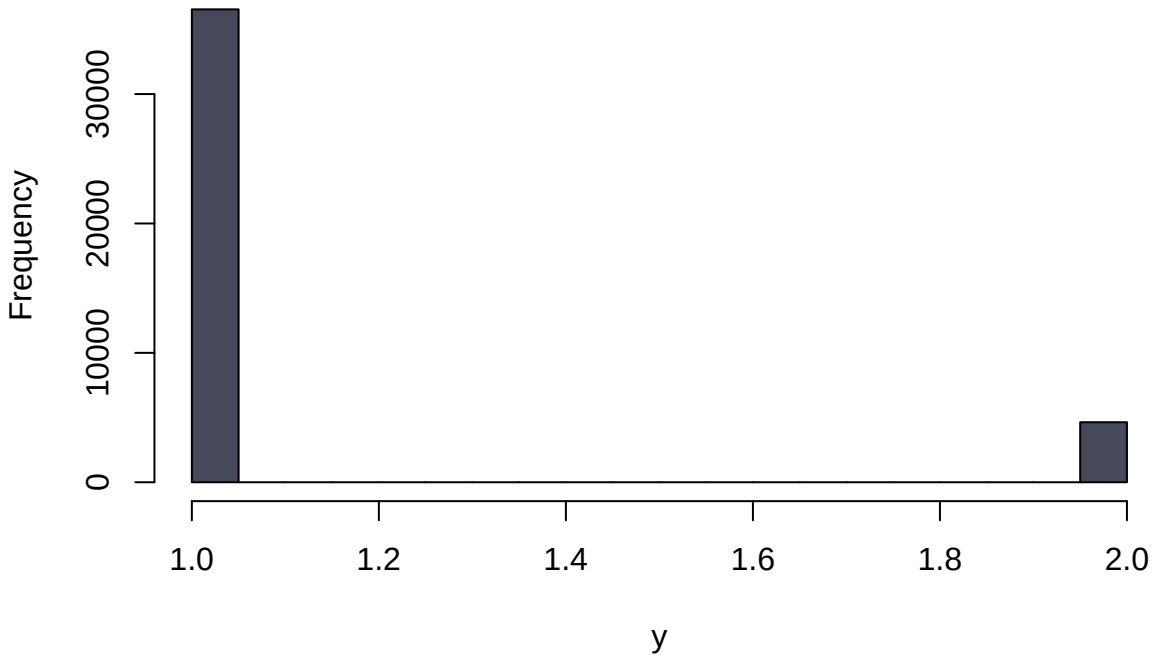

Frequency
0
5000
15000
−3
−2
−1
0
1
emp.var.rate


Frequency
0
5000
10000
15000
92.5
93.0
93.5
94.0
94.5
cons.price.idx


Frequency
0
2000
6000
10000
−50
−45
−40
−35
−30
−25
cons.conf.idx


Frequency
0
5000
15000
25000
1
2
3
4
5
euribor3m


Frequency
0
5000
10000
15000
4950
5000
5050
5100
5150
5200
5250
nr.employed




```
#Converting factor coloumns to numeric
for (i in 1:ncol(bank.data)-1)
  {
    if(is.factor(bank.data[,i]))
      {
        bank.data[,i] <- as.numeric(bank.data[,i])
      }
  }

#Printing the structure of bank data
str(bank.data)
```

```
## 'data.frame':    41188 obs. of  21 variables:
##  $ age           : int  56 57 37 40 56 45 59 41 24 25 ...
##  $ job           : num  4 8 8 1 8 8 1 2 10 8 ...
##  $ marital       : num  2 2 2 2 2 2 2 2 3 3 ...
##  $ education     : num  1 4 4 2 4 3 6 8 6 4 ...
##  $ default       : num  1 2 1 1 1 2 1 2 1 1 ...
##  $ housing       : num  1 1 3 1 1 1 1 1 3 3 ...
##  $ loan          : num  1 1 1 1 3 1 1 1 1 1 ...
##  $ contact       : num  2 2 2 2 2 2 2 2 2 2 ...
##  $ month         : num  7 7 7 7 7 7 7 7 7 7 ...
##  $ day_of_week   : num  2 2 2 2 2 2 2 2 2 2 ...
##  $ duration      : int  261 149 226 151 307 198 139 217 380 50 ...
##  $ campaign      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays         : int  999 999 999 999 999 999 999 999 999 999 ...
```

```
##  $ previous      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome      : num  2 2 2 2 2 2 2 2 2 2 ...
##  $ emp.var.rate  : num  1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ cons.price.idx: num  94 94 94 94 94 ...
##  $ cons.conf.idx : num  -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 ...
##  $ euribor3m     : num  4.86 4.86 4.86 4.86 4.86 ...
##  $ nr.employed   : num  5191 5191 5191 5191 5191 ...
##  $ y             : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
#Creating a function for normalization
normalize <- function(x)
  {
    return ((x-min(x))/(max(x)-min(x)))
  }

#normalizing the bank data
bank.data <- data.frame(lapply(bank.data[,-21], normalize), bank.data$y)
```

3. Build a classification model using a support vector machine that predicts if a bank customer will open a term deposit account.

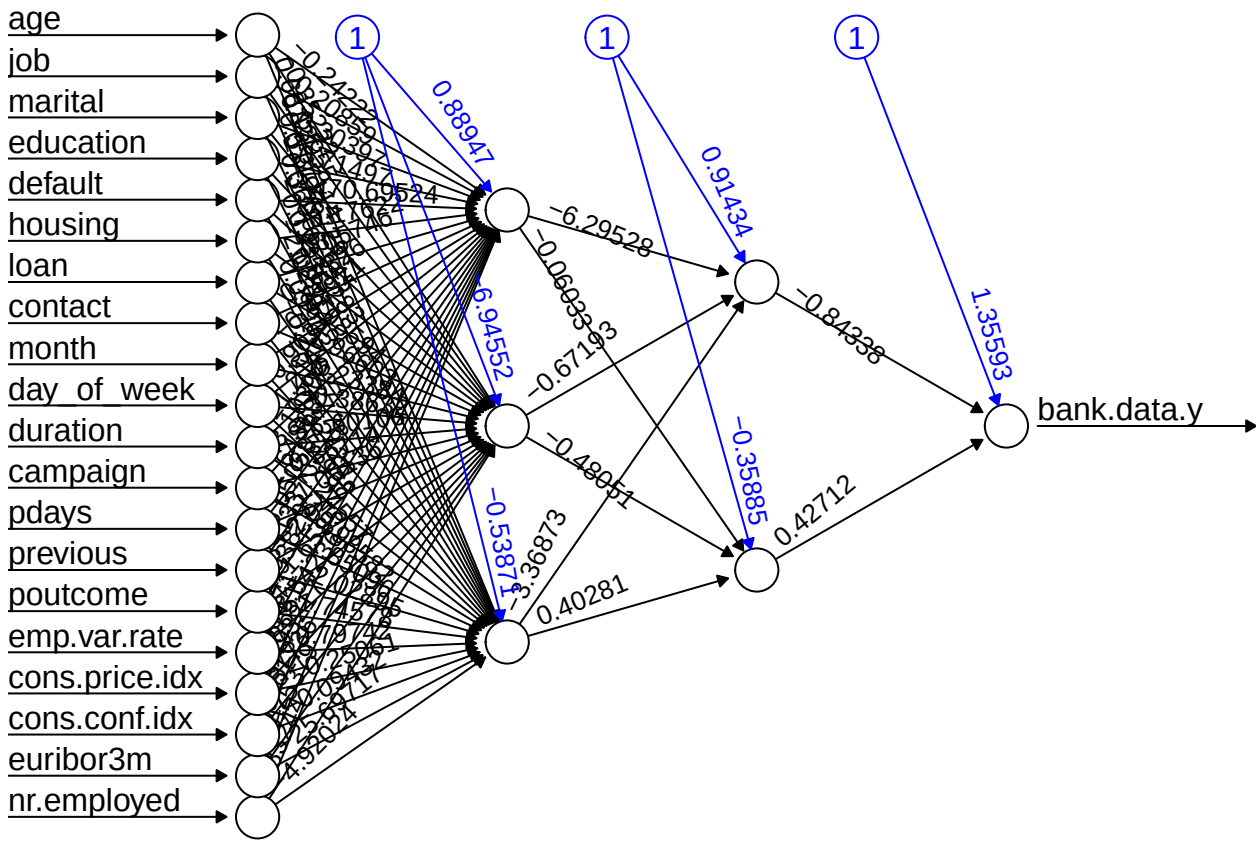
```
#Importing libraries
library(caret)
library(e1071)

#Creating a random sample using createDataPartition function
set.seed(1)
sample <- createDataPartition(bank.data$bank.data.y, p=0.75, list = FALSE)

#Creating training and testing datasets SVM
bank.train.SVM <- bank.data[sample,]
bank.test.SVM <- bank.data[-sample,]

#Creating SVM model
bank.SVM <- svm(bank.data.y ~ ., data = bank.train.SVM, probability = TRUE)

#Predicting the values for the test data
SVM.predict <- predict(bank.SVM, bank.test.SVM, probability = TRUE)

#Printing the confusionMatrix for the SVM model
confusionMatrix(SVM.predict, bank.test.SVM$bank.data.y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  8927  755
##        yes  210  405
##
##                Accuracy : 0.9063
##                  95% CI : (0.9005, 0.9118)
##     No Information Rate : 0.8873
##     P-Value [Acc > NIR] : 2.517e-10
```

```
##
##                   Kappa : 0.4103
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9770
##             Specificity : 0.3491
##          Pos Pred Value : 0.9220
##          Neg Pred Value : 0.6585
##              Prevalence : 0.8873
##          Detection Rate : 0.8670
##    Detection Prevalence : 0.9403
##       Balanced Accuracy : 0.6631
##
##        'Positive' Class : no
##
```

4. Build another classification model using a neural network that also predicts if a bank customer will open a term deposit account.

```r
#Importing libraries for Neural Netwrok
library(neuralnet)

#Creating training and testing datasets for Neural Netwrok(NN)
bank.train.NN <- bank.data[sample,]
bank.test.NN <- bank.data[-sample,]

#Setting y as an integer
bank.train.NN$bank.data.y <- as.integer(bank.train.NN$bank.data.y)
bank.test.NN$bank.data.y <- as.integer(bank.test.NN$bank.data.y)

#Creating function softplus for smoothening
softplus <- function(x) log(1+exp(x))

#Creating Neural Network model
bank.NN <- neuralnet(bank.data.y ~ ., bank.train.NN, hidden = c(3, 2), threshold = 0.5, rep = 1, linear

#plotting the neural network graph
plot(bank.NN, rep="best")
```

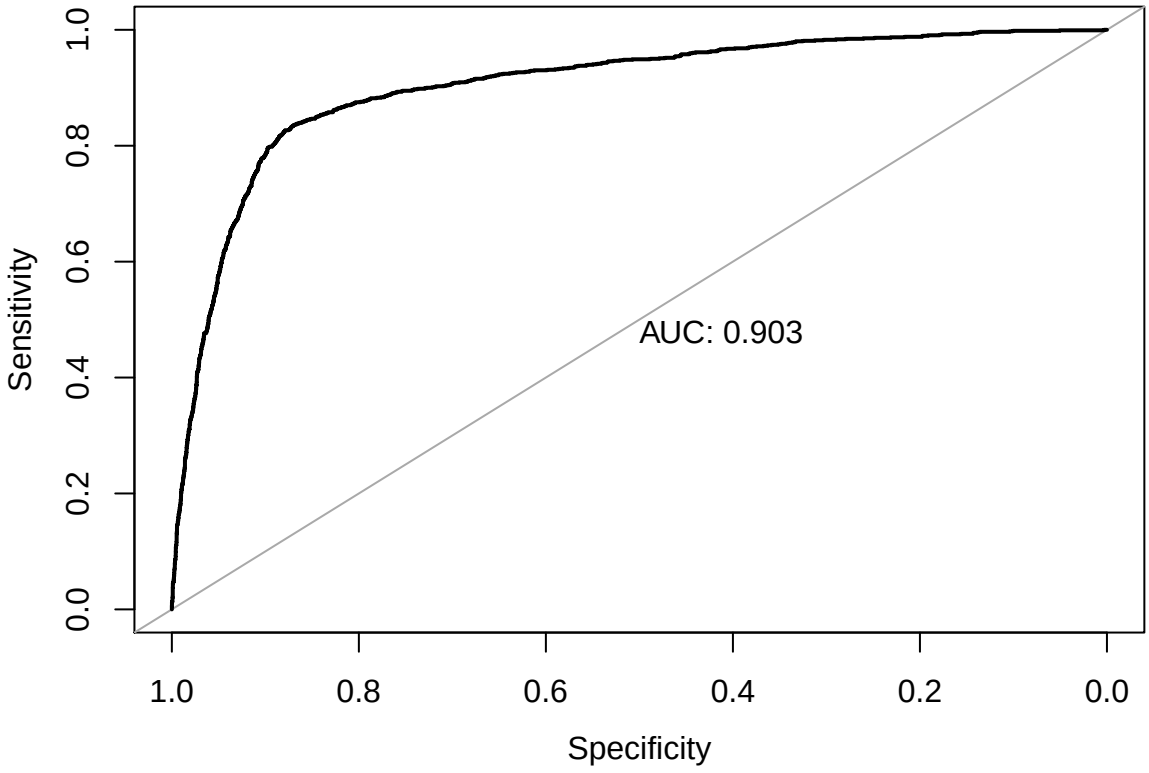



```
#Computing the values for the test data
NN.predict <- compute(bank.NN, bank.test.NN[,-21])

#Calculating correlation between the actual and predicted values
cor.NN <- cor(NN.predict$net.result, bank.test.NN$bank.data.y)
sprintf("The correlation between actual values and predicted values by Neural Network is: %s", cor.NN)
```

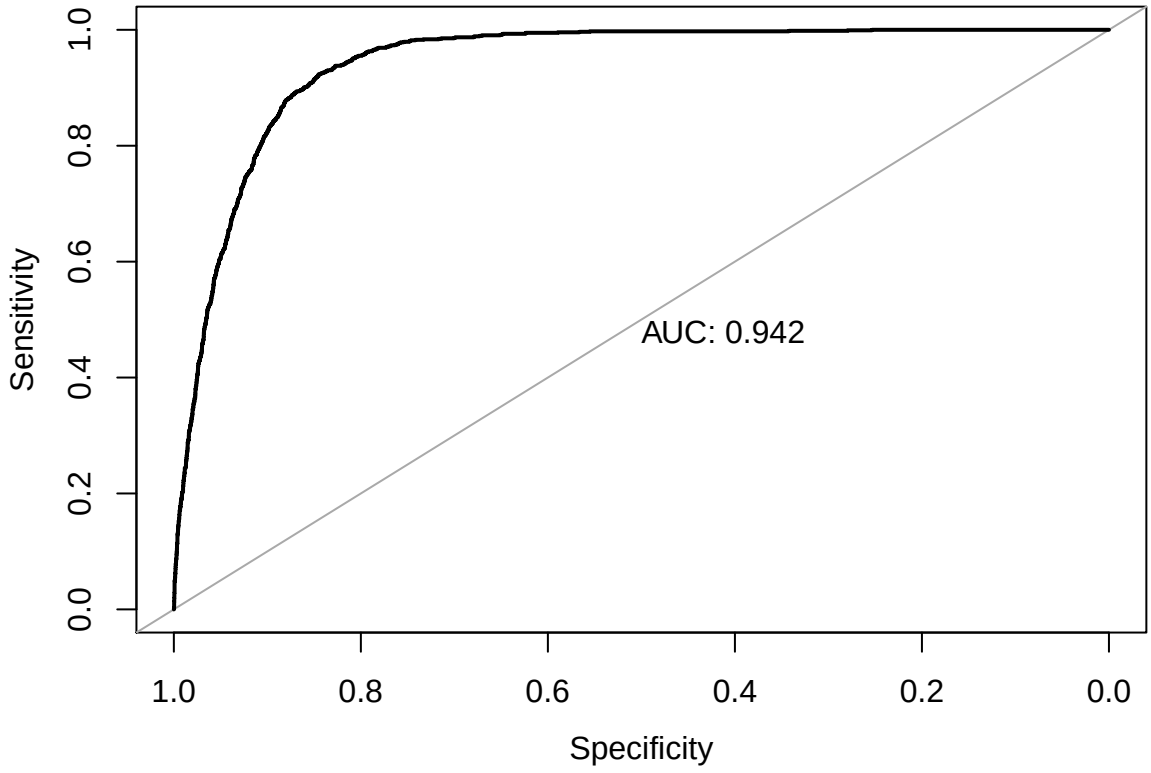
```
## [1] "The correlation between actual values and predicted values by Neural Network is: 0.641745185455
```

5. Compare the accuracy of the two models based on AUC.

```
#Importing libraries to calculate ROC and AUC
library(pROC)

#Retiving probability values for SVM
df <- as.data.frame(attr(SVM.predict, "probabilities"))
colnames(df) <- c("one", "two")

#Plotting AUC graph for SVM
plot.roc(roc(as.numeric(bank.test.SVM$bank.data.y), as.numeric(df$one)), axis = TRUE, legacy.axes = FALS
```




```
#Plotting AUC graph for Neural Network
plot.roc(roc(as.numeric(bank.test.NN$bank.data.y), as.numeric(NN.predict$net.result)), axis = TRUE, lega
```



6. Calculate precision and recall for both models. See this article to understand how to calculate these metrics.

```
#Calculating precision for SVM
SVM.precision <- posPredValue(SVM.predict, bank.test.SVM$bank.data.y, positive = "no")
sprintf("The precision for SVM is: %s", SVM.precision)
```

```
## [1] "The precision for SVM is: 0.922020243751291"
```

```
#Calculating recall for SVM
SVM.recall <- sensitivity(SVM.predict, bank.test.SVM$bank.data.y, positive = "no")
sprintf("The recall for SVM is: %s", SVM.recall)
```

```
## [1] "The recall for SVM is: 0.977016526212105"
```

```
#Creating data frame prediction strength for Neural Netwrok based on the original factor values
NN.predict.strength <- ifelse(NN.predict$net.result>1.5, 2, 1)

#Calculating precision for Neural Network
NN.precision <- posPredValue(as.factor(NN.predict.strength), as.factor(bank.test.NN$bank.data.y), positi
sprintf("The precision for Neural Network is: %s", NN.precision)
```

```
## [1] "The precision for Neural Network is: 0.940478734772387"
```

```r
#Calculating reacll for Neural Network
NN.recall <- sensitivity(as.factor(NN.predict.strength), as.factor(bank.test.NN$bank.data.y), positive =
sprintf("The recall for Neural Network is: %s", NN.recall)
```

```
## [1] "The recall for Neural Network is: 0.963226441939367"
```