```r
# Data Ingestion and Summary statistics
credit_data.raw<-read.csv("BOA.csv")
str(credit_data.raw)
```

```
## 'data.frame':    8950 obs. of  19 variables:
##  $ X                                 : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ CUST_ID                           : Factor w/ 8950 levels "C10001","C10002",..: 1 2 3 4 5 6 7 8 9 10
##  $ BALANCE                           : num  40.9 3202.5 2495.1 1666.7 817.7 ...
##  $ BALANCE_FREQUENCY                 : num  0.818 0.909 1 0.636 1 ...
##  $ PURCHASES                         : num  95.4 0 773.2 1499 16 ...
##  $ ONEOFF_PURCHASES                  : num  0 0 773 1499 16 ...
##  $ INSTALLMENTS_PURCHASES            : num  95.4 0 0 0 0 ...
##  $ CASH_ADVANCE                      : num  0 6443 0 206 0 ...
##  $ PURCHASES_FREQUENCY               : num  0.1667 0 1 0.0833 0.0833 ...
##  $ ONEOFF_PURCHASES_FREQUENCY        : num  0 0 1 0.0833 0.0833 ...
##  $ PURCHASES_INSTALLMENTS_FREQUENCY: num  0.0833 0 0 0 0 ...
##  $ CASH_ADVANCE_FREQUENCY            : num  0 0.25 0 0.0833 0 ...
##  $ CASH_ADVANCE_TRX                  : int  0 4 0 1 0 0 0 0 0 0 ...
##  $ PURCHASES_TRX                     : int  2 0 12 1 1 8 64 12 5 3 ...
##  $ CREDIT_LIMIT                      : num  1000 7000 7500 7500 1200 1800 13500 2300 7000 11000 ...
##  $ PAYMENTS                          : num  202 4103 622 0 678 ...
##  $ MINIMUM_PAYMENTS                  : num  140 1072 627 0 245 ...
##  $ PRC_FULL_PAYMENT                  : num  0 0.222 0 0 0 ...
##  $ TENURE                            : int  12 12 12 12 12 12 12 12 12 12 ...
```

```r
summary(credit_data.raw)
```

```
##        X           CUST_ID        BALANCE        BALANCE_FREQUENCY
##  Min.   :   1   C10001 :   1   Min.   :    0.0   Min.   :0.0000
##  1st Qu.:2238   C10002 :   1   1st Qu.:  128.3   1st Qu.:0.8889
##  Median :4476   C10003 :   1   Median :  873.4   Median :1.0000
##  Mean   :4476   C10004 :   1   Mean   : 1564.5   Mean   :0.8773
##  3rd Qu.:6713   C10005 :   1   3rd Qu.: 2054.1   3rd Qu.:1.0000
##  Max.   :8950   C10006 :   1   Max.   :19043.1   Max.   :1.0000
##                 (Other):8944
##    PURCHASES        ONEOFF_PURCHASES  INSTALLMENTS_PURCHASES  CASH_ADVANCE
##  Min.   :    0.00   Min.   :    0.0   Min.   :    0.0        Min.   :    0.0
##  1st Qu.:   39.63   1st Qu.:    0.0   1st Qu.:    0.0        1st Qu.:    0.0
##  Median :  361.28   Median :   38.0   Median :   89.0        Median :    0.0
##  Mean   : 1003.20   Mean   :  592.4   Mean   :  411.1        Mean   :  978.9
##  3rd Qu.: 1110.13   3rd Qu.:  577.4   3rd Qu.:  468.6        3rd Qu.: 1113.8
##  Max.   :49039.57   Max.   :40761.2   Max.   :22500.0        Max.   :47137.2
##
##  PURCHASES_FREQUENCY ONEOFF_PURCHASES_FREQUENCY
##  Min.   :0.00000     Min.   :0.00000
##  1st Qu.:0.08333     1st Qu.:0.00000
##  Median :0.50000     Median :0.08333
##  Mean   :0.49035     Mean   :0.20246
##  3rd Qu.:0.91667     3rd Qu.:0.30000
##  Max.   :1.00000     Max.   :1.00000
##
##  PURCHASES_INSTALLMENTS_FREQUENCY CASH_ADVANCE_FREQUENCY CASH_ADVANCE_TRX
##  Min.   :0.0000                   Min.   :0.0000         Min.   :  0.000
```

```
##  1st Qu.:0.0000                  1st Qu.:0.0000         1st Qu.:  0.000
##  Median :0.1667                  Median :0.0000         Median :  0.000
##  Mean   :0.3644                  Mean   :0.1351         Mean   :  3.249
##  3rd Qu.:0.7500                  3rd Qu.:0.2222         3rd Qu.:  4.000
##  Max.   :1.0000                  Max.   :1.5000         Max.   :123.000
##
##  PURCHASES_TRX     CREDIT_LIMIT      PAYMENTS       MINIMUM_PAYMENTS
##  Min.   :  0.00   Min.   :   50   Min.   :    0.0   Min.   :    0.0
##  1st Qu.:  1.00   1st Qu.: 1600   1st Qu.:  383.3   1st Qu.:  163.0
##  Median :  7.00   Median : 3000   Median :  856.9   Median :  289.6
##  Mean   : 14.71   Mean   : 4494   Mean   : 1733.1   Mean   :  834.0
##  3rd Qu.: 17.00   3rd Qu.: 6500   3rd Qu.: 1901.1   3rd Qu.:  788.7
##  Max.   :358.00   Max.   :30000   Max.   :50721.5   Max.   :76406.2
##
##  PRC_FULL_PAYMENT     TENURE
##  Min.   :0.0000   Min.   : 6.00
##  1st Qu.:0.0000   1st Qu.:12.00
##  Median :0.0000   Median :12.00
##  Mean   :0.1537   Mean   :11.52
##  3rd Qu.:0.1429   3rd Qu.:12.00
##  Max.   :1.0000   Max.   :12.00
##
```

```
#Data Exploration and Cleaning
library(psych)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
 #Remove identity columns - cust id and x from the table
credit_data<-credit_data.raw[c(-1,-2)]
describe(credit_data)
```

```
##                                  vars    n    mean      sd  median trimmed
## BALANCE                             1 8950 1564.47 2081.53  873.39 1128.17
## BALANCE_FREQUENCY                   2 8950    0.88    0.24    1.00    0.94
## PURCHASES                           3 8950 1003.20 2136.63  361.28  583.10
## ONEOFF_PURCHASES                    4 8950  592.44 1659.89   38.00  263.82
## INSTALLMENTS_PURCHASES              5 8950  411.07  904.34   89.00  223.03
## CASH_ADVANCE                        6 8950  978.87 2097.16    0.00  494.90
## PURCHASES_FREQUENCY                 7 8950    0.49    0.40    0.50    0.49
## ONEOFF_PURCHASES_FREQUENCY          8 8950    0.20    0.30    0.08    0.14
## PURCHASES_INSTALLMENTS_FREQUENCY    9 8950    0.36    0.40    0.17    0.33
```

```
## CASH_ADVANCE_FREQUENCY             10 8950    0.14    0.20    0.00    0.09
## CASH_ADVANCE_TRX                   11 8950    3.25    6.82    0.00    1.73
## PURCHASES_TRX                      12 8950   14.71   24.86    7.00    9.43
## CREDIT_LIMIT                       13 8950 4494.45 3638.61 3000.00 3927.21
## PAYMENTS                           14 8950 1733.14 2895.06  856.90 1152.89
## MINIMUM_PAYMENTS                   15 8950  833.98 2335.99  289.63  463.58
## PRC_FULL_PAYMENT                   16 8950    0.15    0.29    0.00    0.08
## TENURE                             17 8950   11.52    1.34   12.00   11.92
##                                      mad min      max    range  skew kurtosis
## BALANCE                          1185.88   0 19043.14 19043.14  2.39     7.67
## BALANCE_FREQUENCY                   0.00   0     1.00     1.00 -2.02     3.09
## PURCHASES                         535.63   0 49039.57 49039.57  8.14   111.30
## ONEOFF_PURCHASES                   56.34   0 40761.25 40761.25 10.04   164.06
## INSTALLMENTS_PURCHASES            131.95   0 22500.00 22500.00  7.30    96.50
## CASH_ADVANCE                        0.00   0 47137.21 47137.21  5.16    52.86
## PURCHASES_FREQUENCY                 0.62   0     1.00     1.00  0.06    -1.64
## ONEOFF_PURCHASES_FREQUENCY          0.12   0     1.00     1.00  1.54     1.16
## PURCHASES_INSTALLMENTS_FREQUENCY    0.25   0     1.00     1.00  0.51    -1.40
## CASH_ADVANCE_FREQUENCY              0.00   0     1.50     1.50  1.83     3.33
## CASH_ADVANCE_TRX                    0.00   0   123.00   123.00  5.72    61.60
## PURCHASES_TRX                      10.38   0   358.00   358.00  4.63    34.76
## CREDIT_LIMIT                     2668.68  50 30000.00 29950.00  1.52     2.83
## PAYMENTS                          861.91   0 50721.48 50721.48  5.91    54.73
## MINIMUM_PAYMENTS                  279.80   0 76406.21 76406.21 13.80   292.13
## PRC_FULL_PAYMENT                    0.00   0     1.00     1.00  1.94     2.43
## TENURE                              0.00   6    12.00     6.00 -2.94     7.69
##                                      se
## BALANCE                           22.00
## BALANCE_FREQUENCY                  0.00
## PURCHASES                         22.58
## ONEOFF_PURCHASES                  17.55
## INSTALLMENTS_PURCHASES             9.56
## CASH_ADVANCE                      22.17
## PURCHASES_FREQUENCY                0.00
## ONEOFF_PURCHASES_FREQUENCY         0.00
## PURCHASES_INSTALLMENTS_FREQUENCY   0.00
## CASH_ADVANCE_FREQUENCY             0.00
## CASH_ADVANCE_TRX                   0.07
## PURCHASES_TRX                      0.26
## CREDIT_LIMIT                      38.46
## PAYMENTS                          30.60
## MINIMUM_PAYMENTS                  24.69
## PRC_FULL_PAYMENT                   0.00
## TENURE                             0.01
```

```r
#credit_data<-credit_data[(credit_data[,3]>0),]

# To check for null value in the data
sapply(credit_data, function(x) sum(is.na(x)))
```

```
##                           BALANCE                BALANCE_FREQUENCY
##                                 0                                0
##                         PURCHASES                 ONEOFF_PURCHASES
##                                 0                                0
```

```
##           INSTALLMENTS_PURCHASES                     CASH_ADVANCE
##                                0                                0
##              PURCHASES_FREQUENCY       ONEOFF_PURCHASES_FREQUENCY
##                                0                                0
## PURCHASES_INSTALLMENTS_FREQUENCY           CASH_ADVANCE_FREQUENCY
##                                0                                0
##                 CASH_ADVANCE_TRX                    PURCHASES_TRX
##                                0                                0
##                     CREDIT_LIMIT                         PAYMENTS
##                                0                                0
##                 MINIMUM_PAYMENTS                PRC_FULL_PAYMENT
##                                0                                0
##                           TENURE
##                                0
```

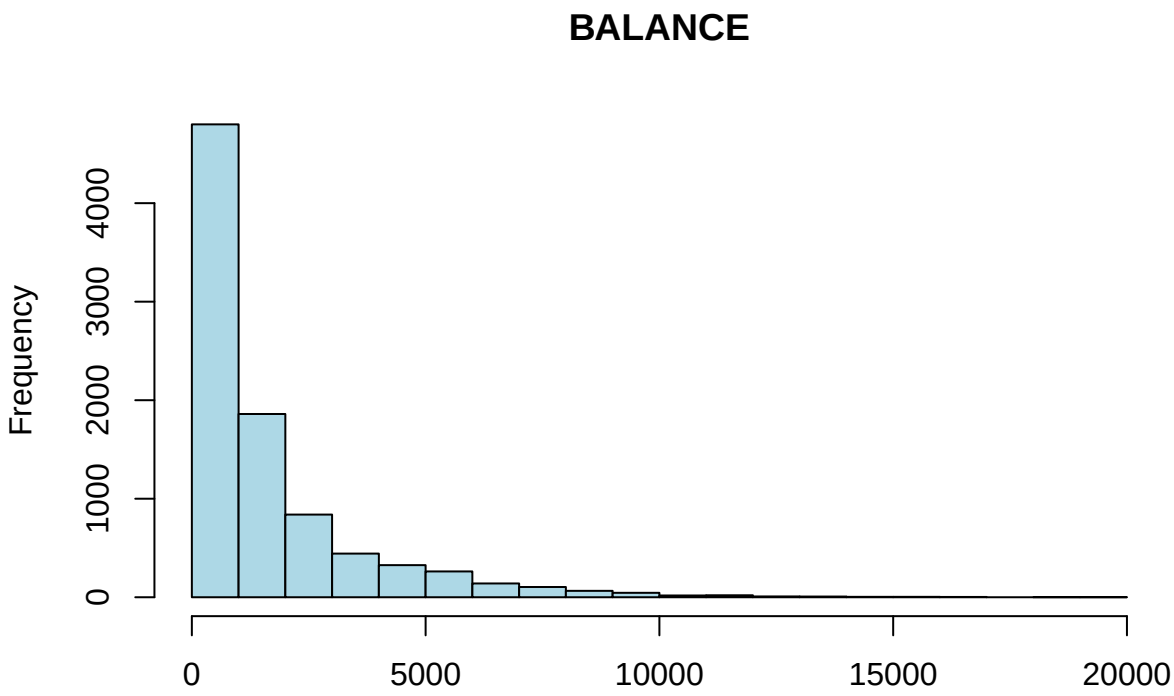
```
#Plot graph for each variable for distribution
for ( i in seq(1,length( credit_data ),1) ) hist(credit_data[,i],ylab="Frequency",xlab = "",type="l",ma
```

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete
```

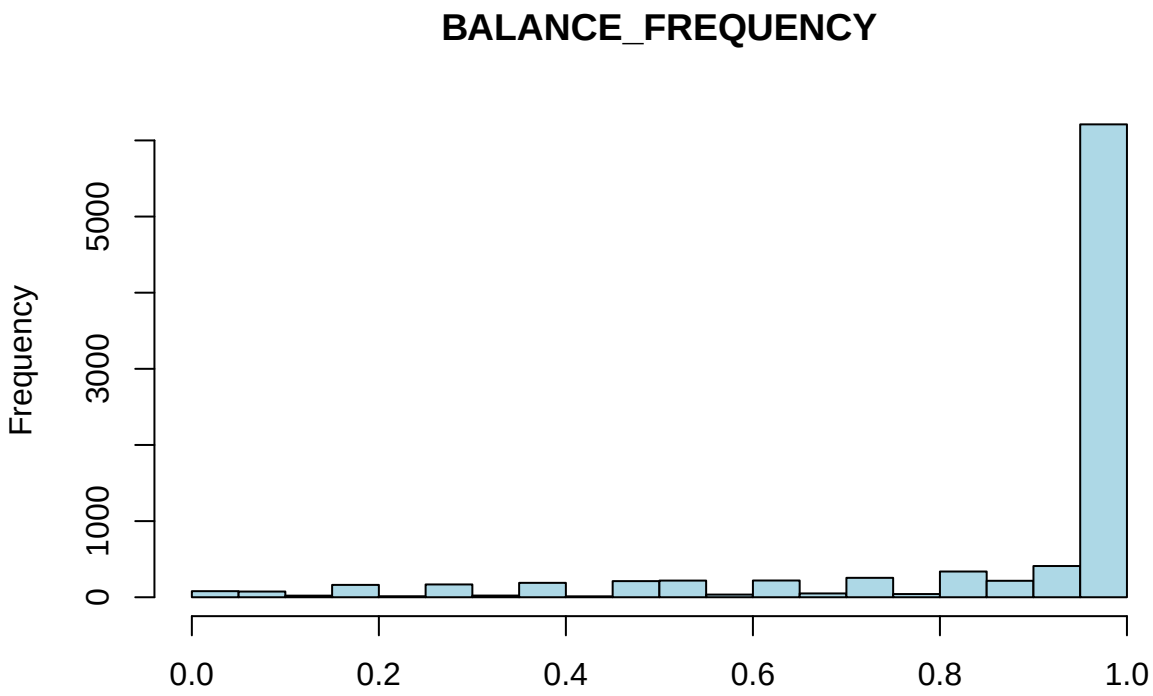
```
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete
```

```
## Warning in axis(1, ...): graphical parameter "type" is obsolete
```

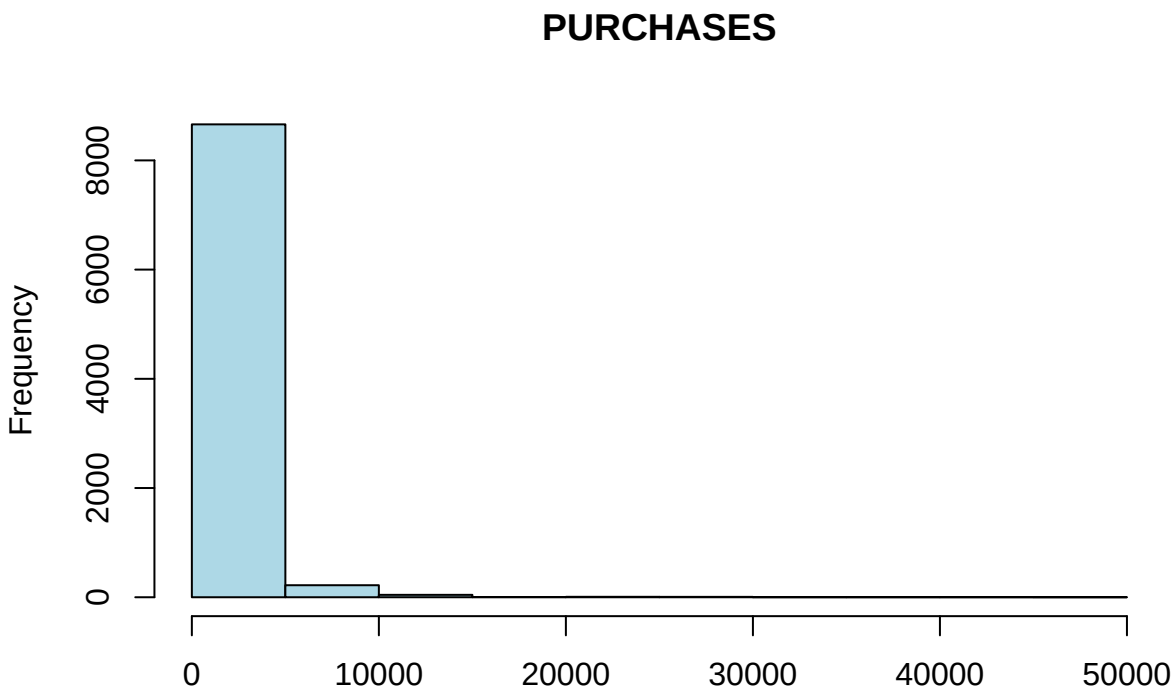
```
## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



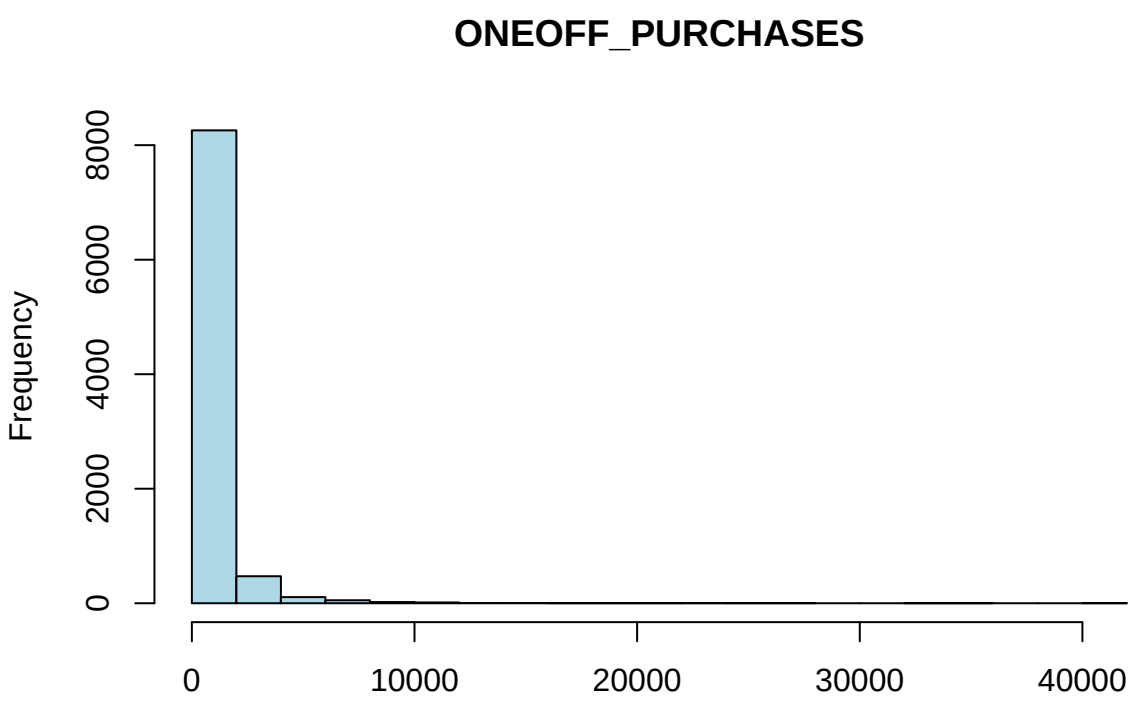

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



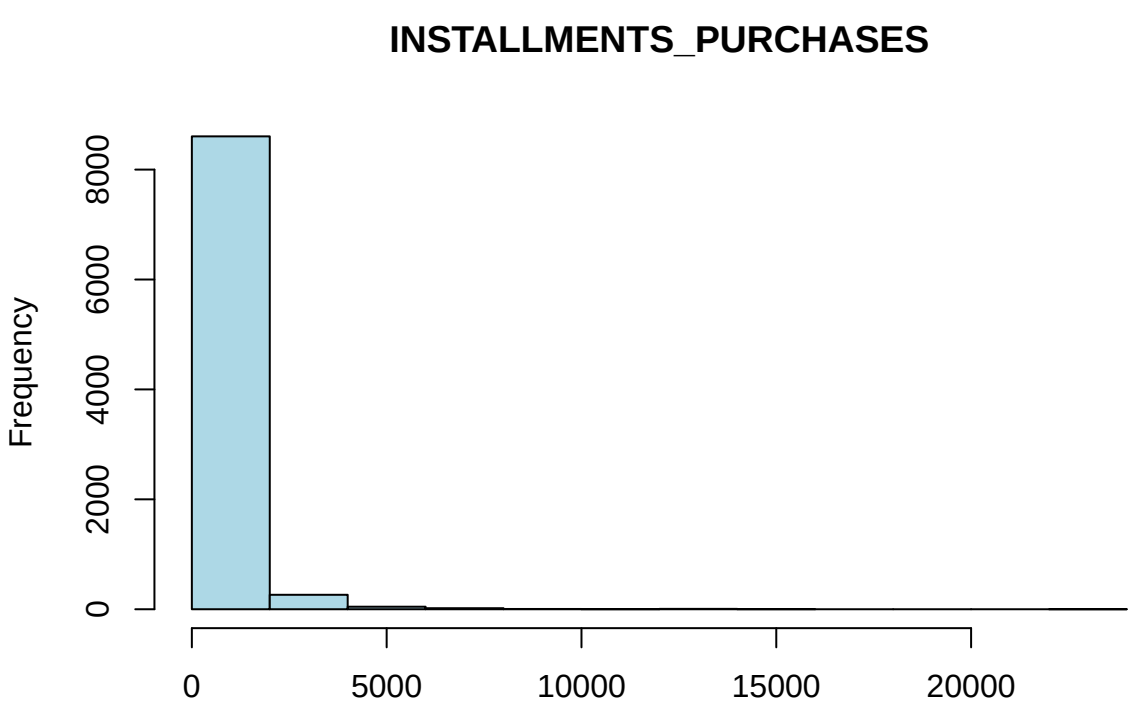

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



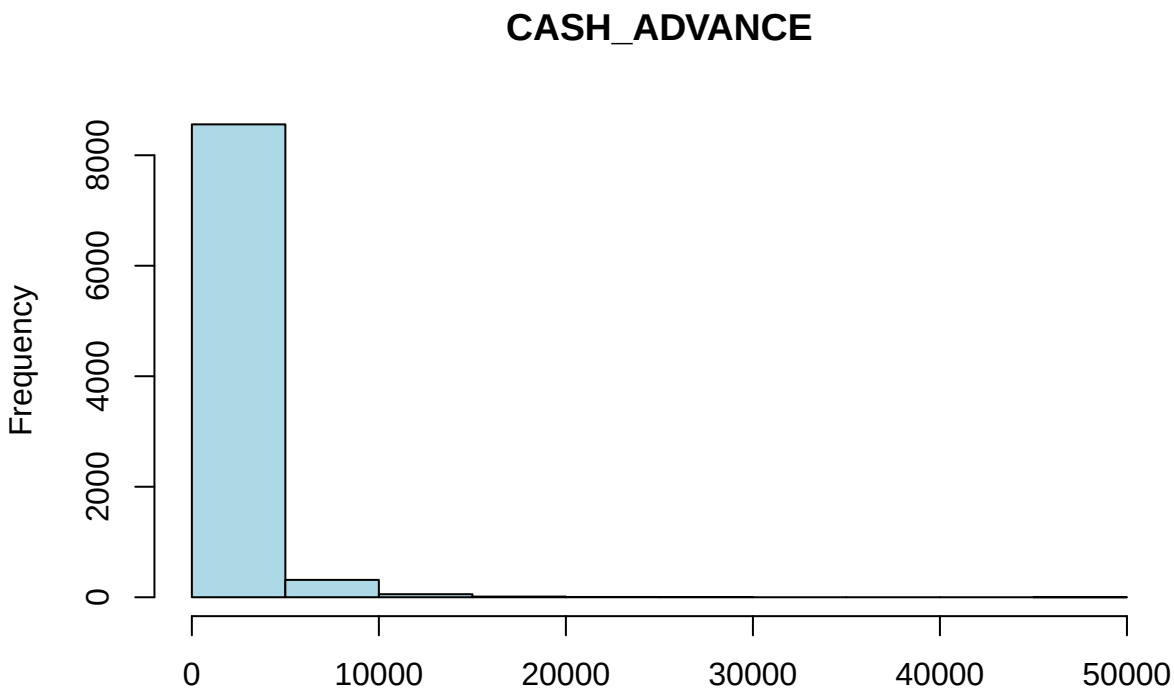

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



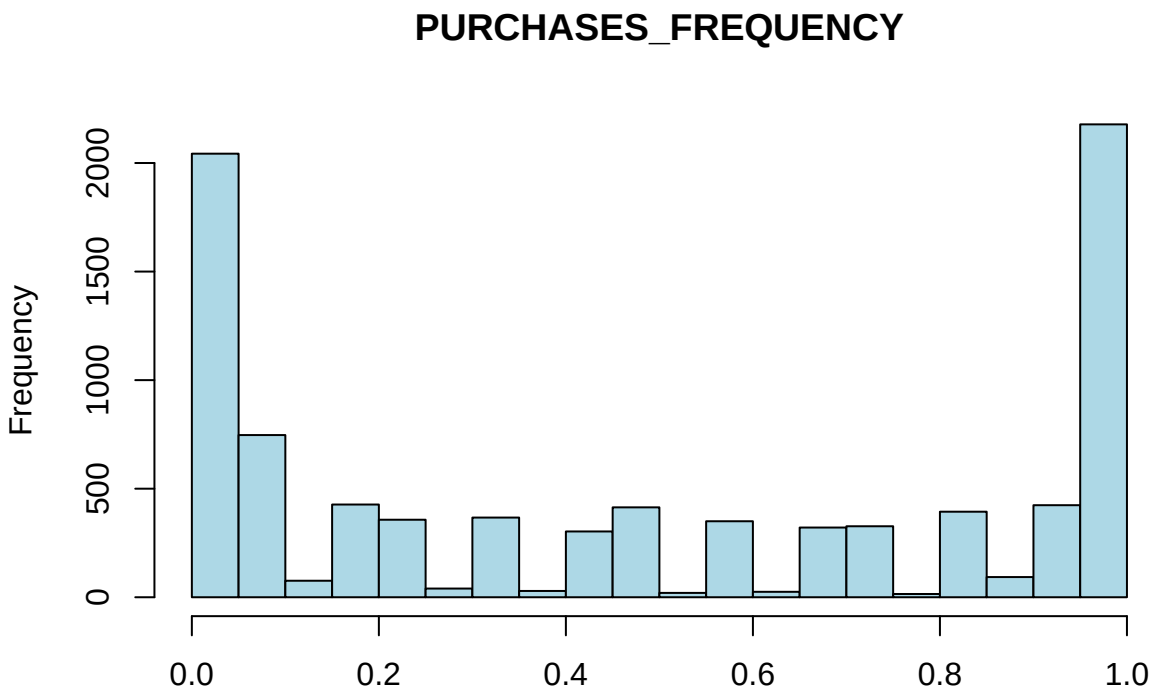

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



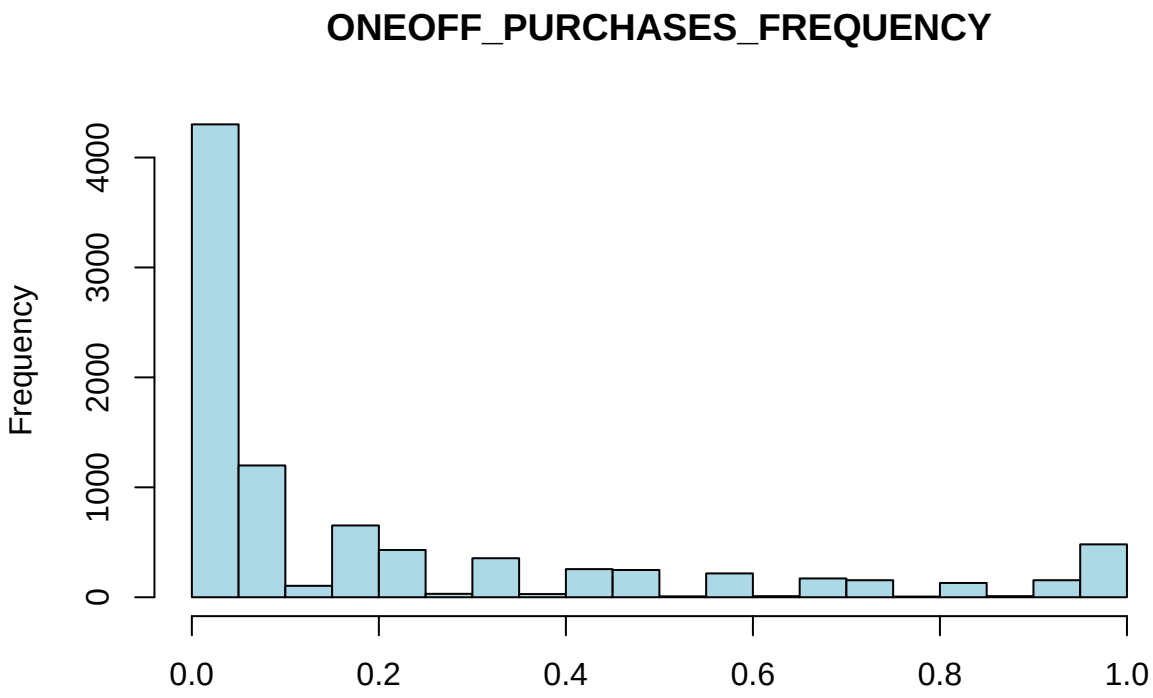

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



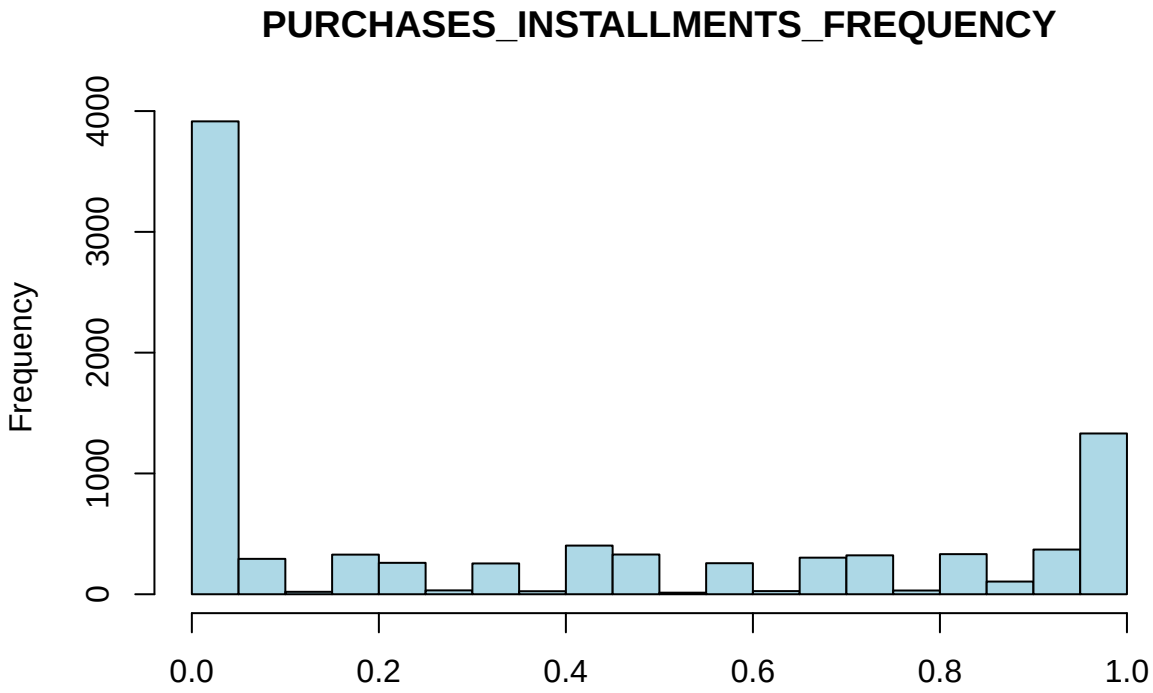

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



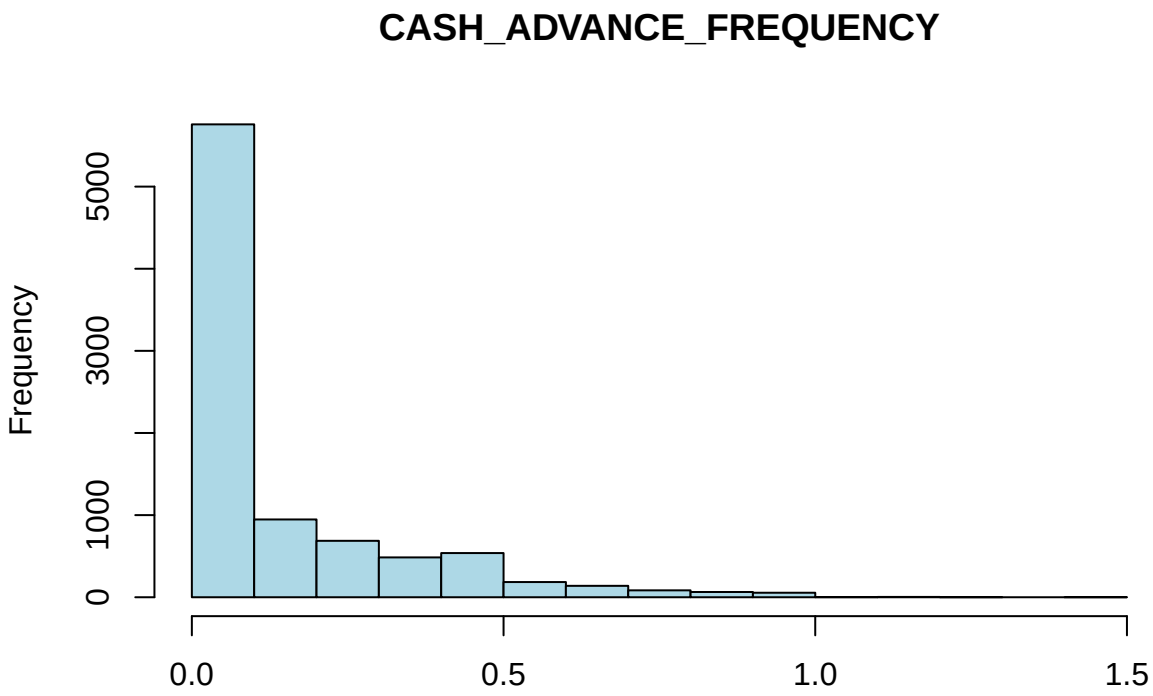

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



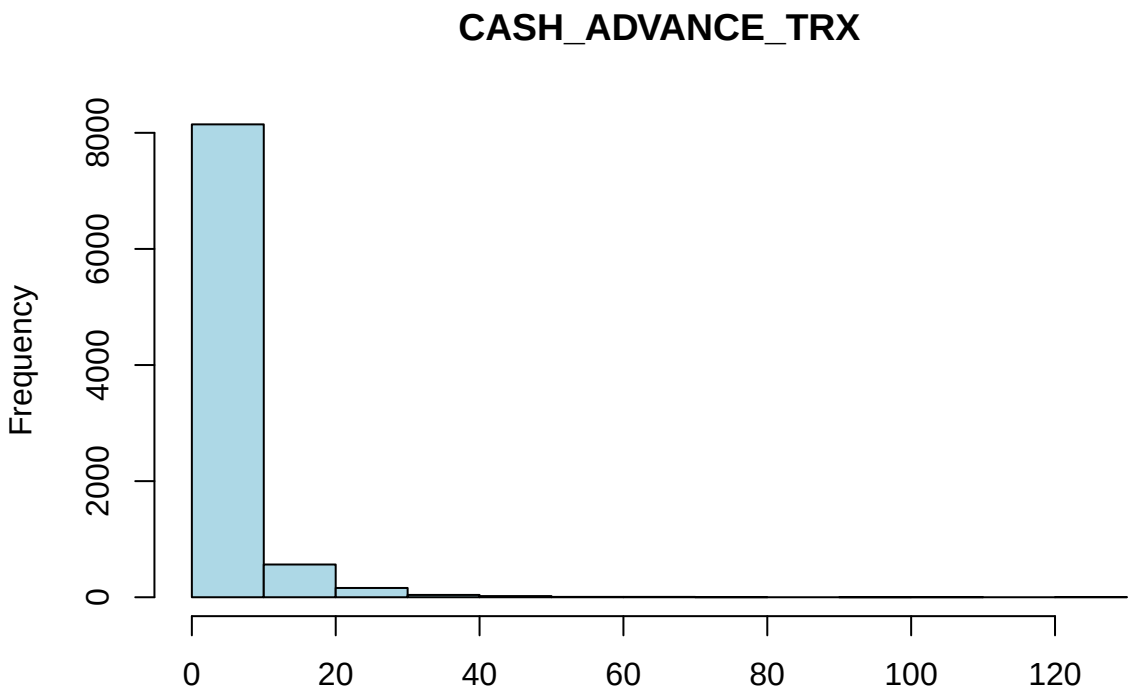

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



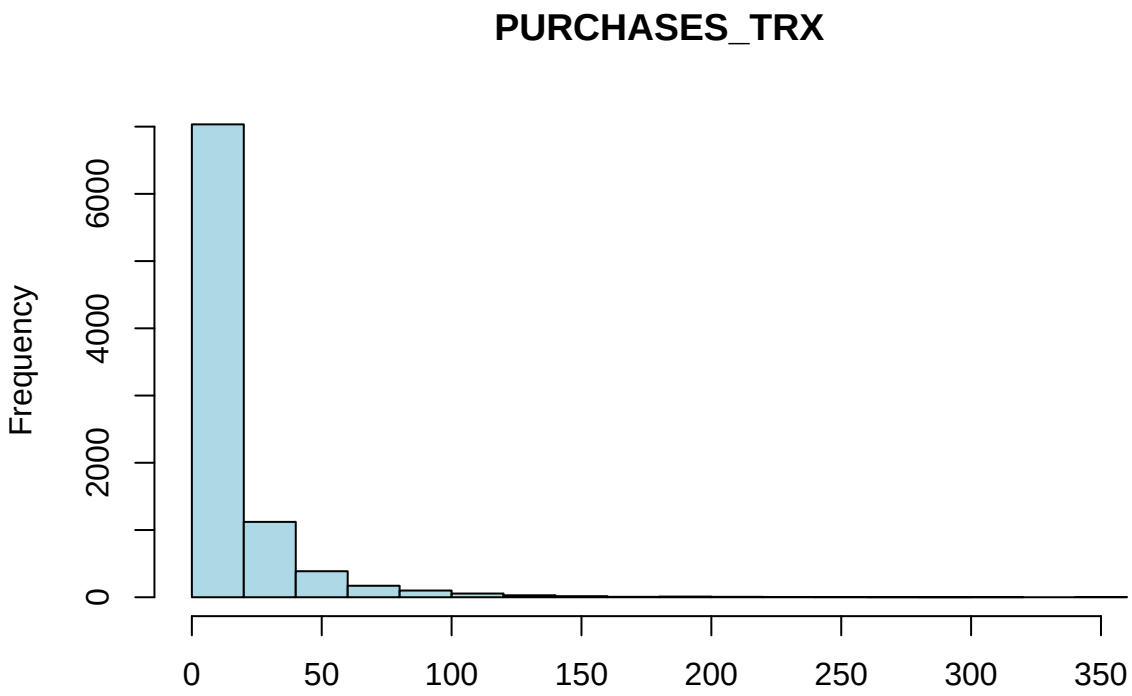

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



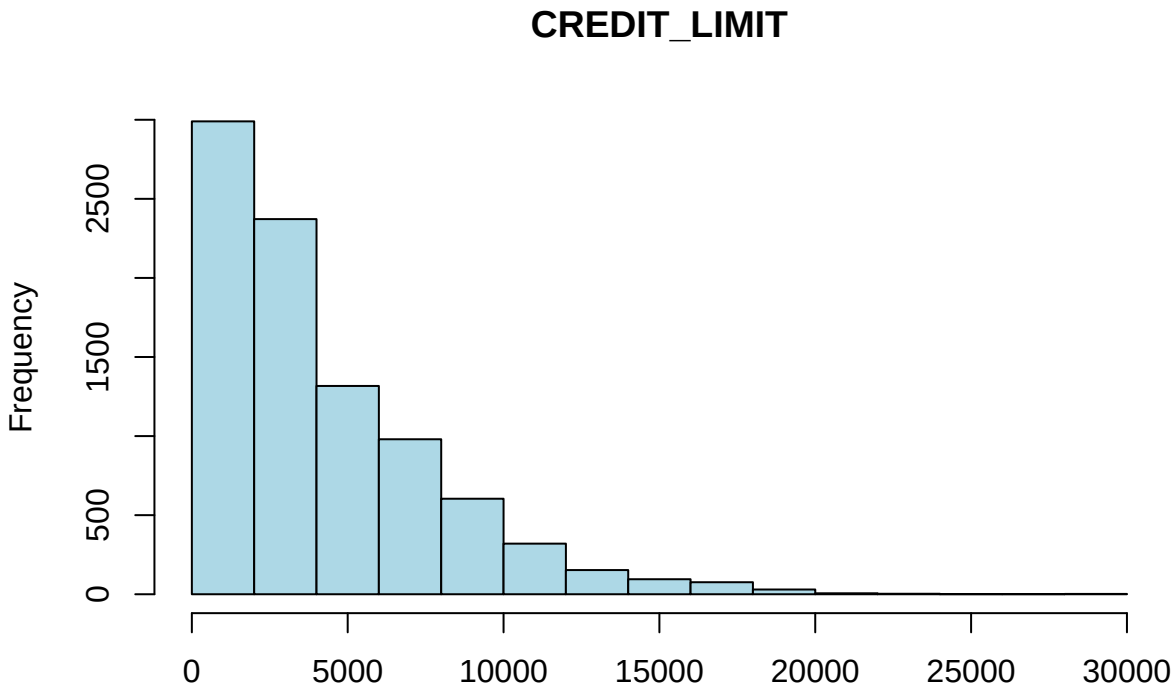

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



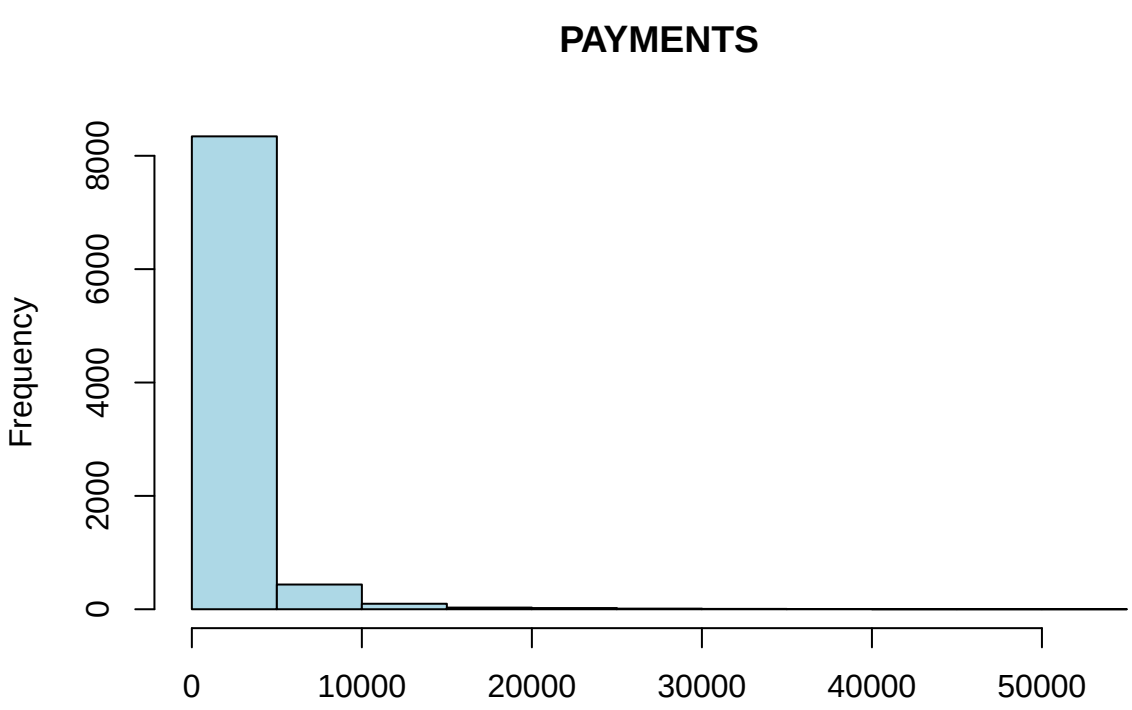

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



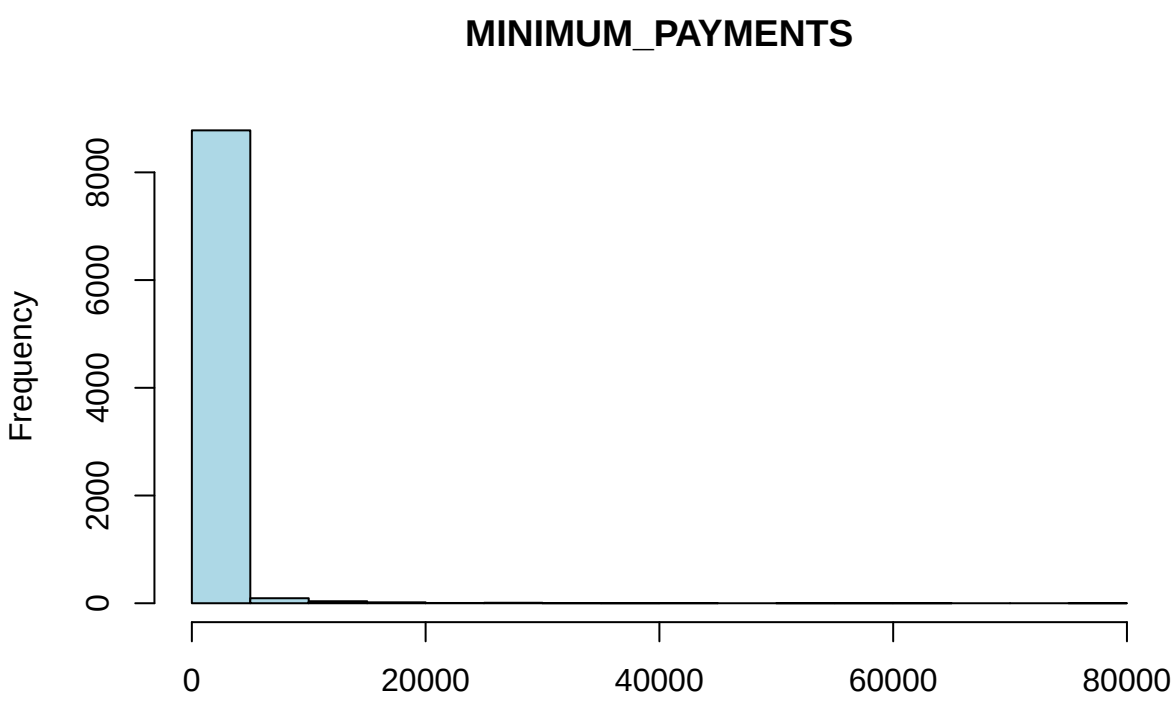

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



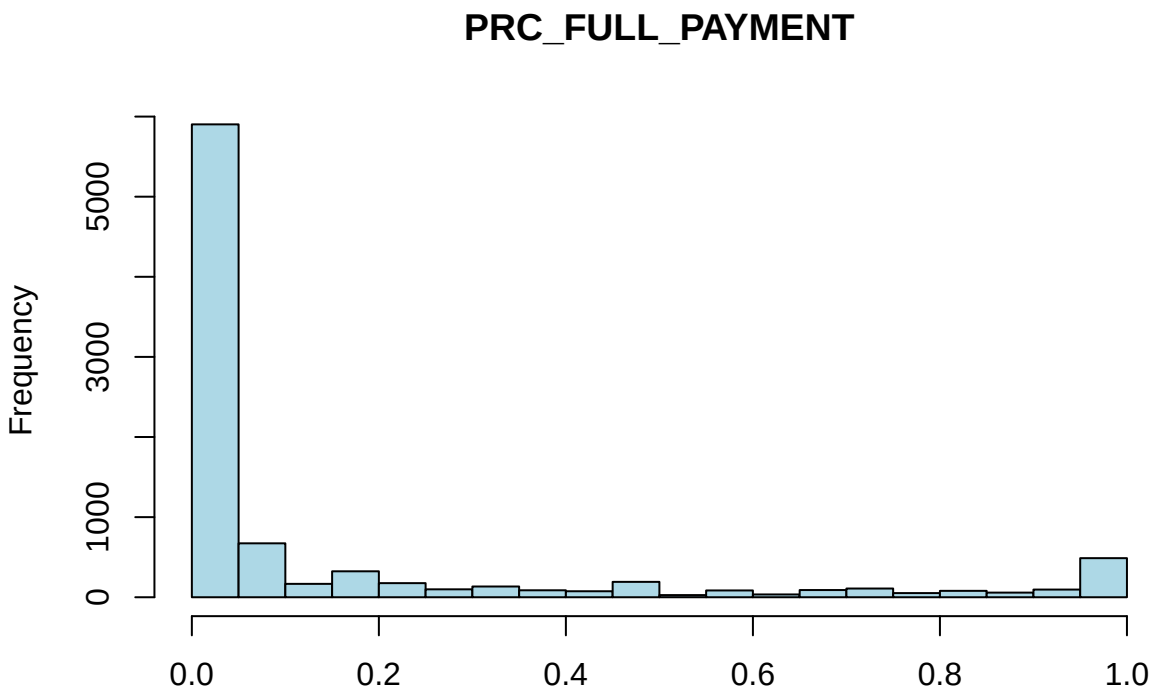

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



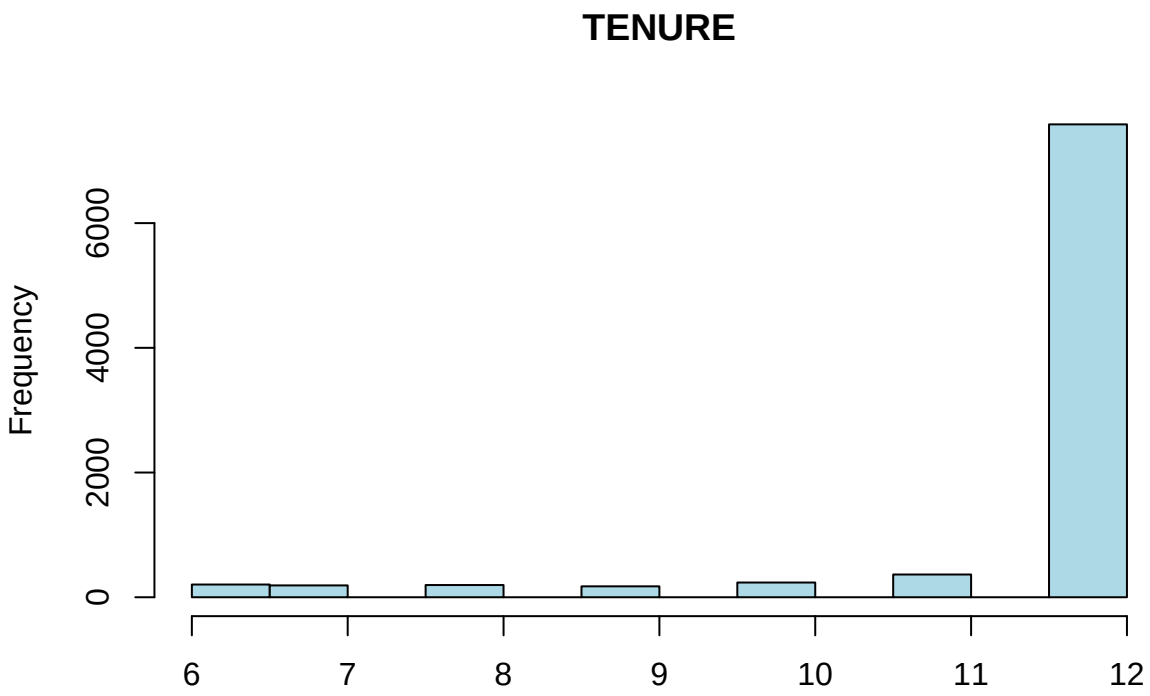

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```




```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```



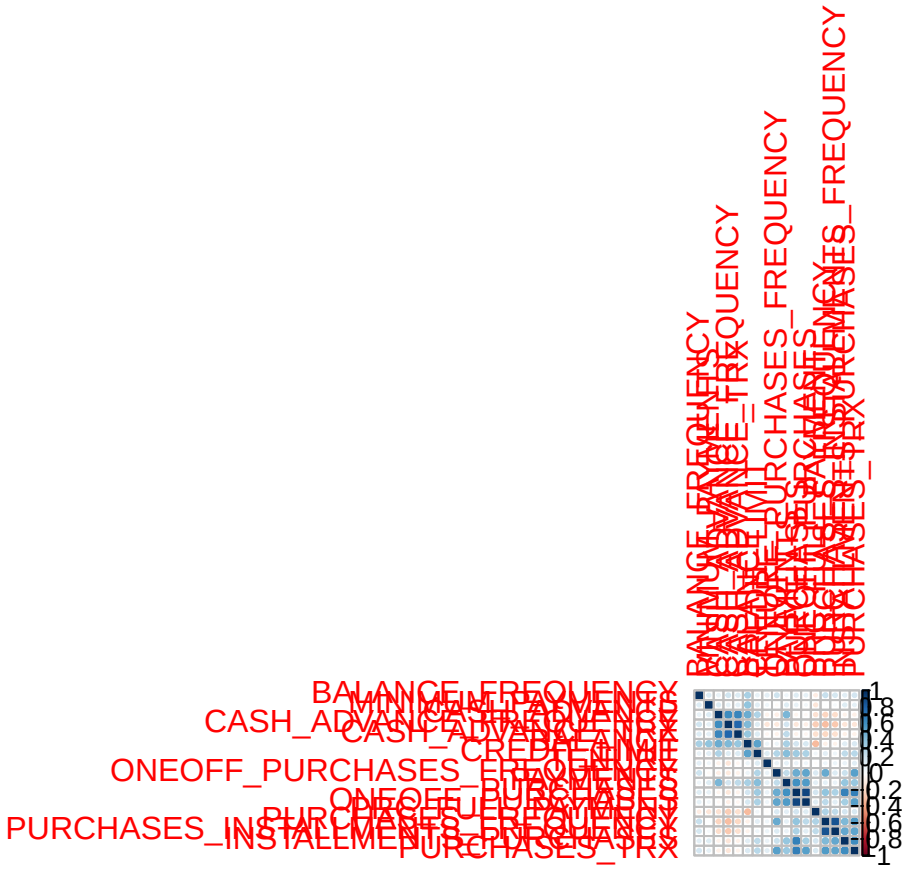

```
## Warning in plot.window(xlim, ylim, "", ...): graphical parameter "type" is
## obsolete

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## graphical parameter "type" is obsolete

## Warning in axis(1, ...): graphical parameter "type" is obsolete

## Warning in axis(2, ...): graphical parameter "type" is obsolete
```




```
#Function to calculate the mean of each variable for each group/class
data.summ <- function(data, groups) {
  aggregate(data, list(groups), function(x) mean(as.numeric(x)))
}

#rescales the data for normalization to avoid biasness in the model
credit_data_scale<-scale(credit_data)


library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
#Checking for correlation between variables
corrplot(cor(credit_data_scale [ , ]) , order="hclust")
```

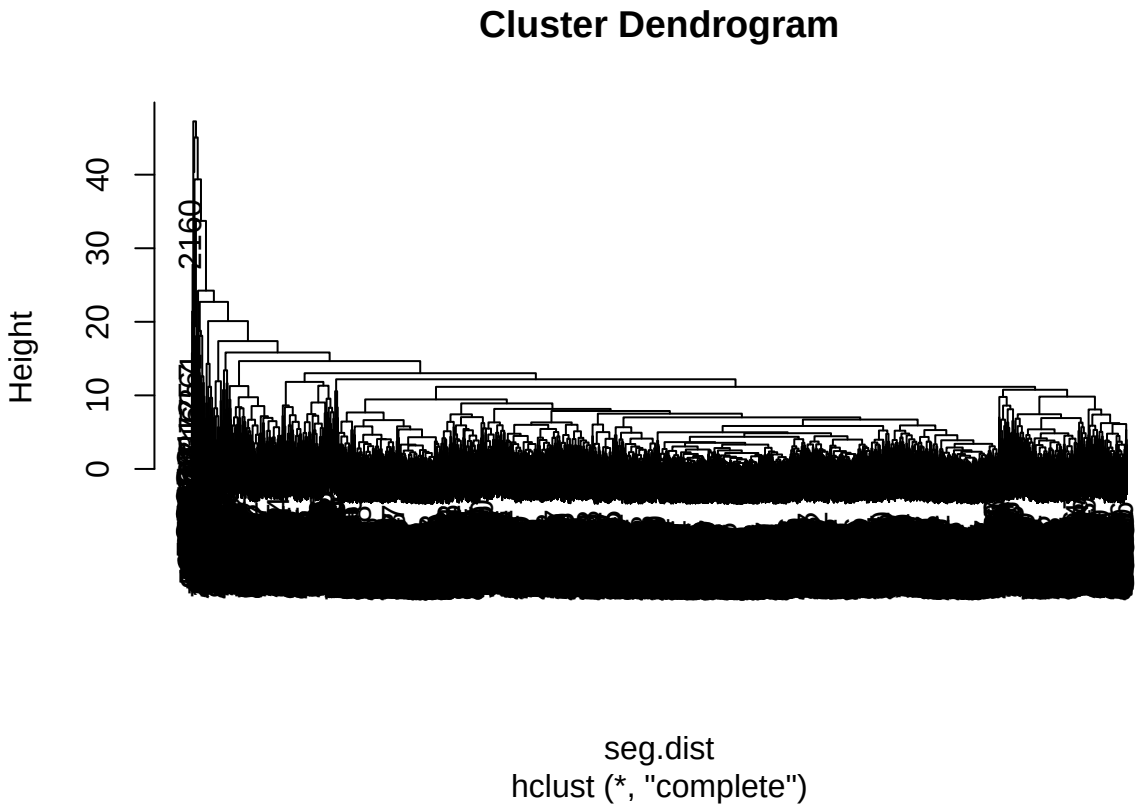



```
# creating principal components that will one of the input for the models
credit_data_pc<-prcomp(credit_data_scale)
```

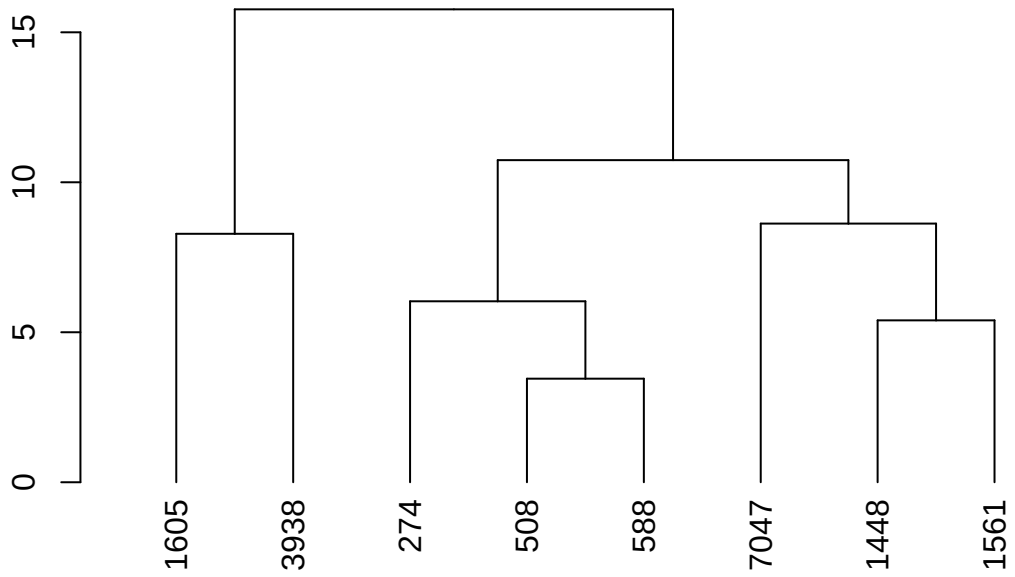
```
library(cluster)
#rescales the data to avoid over proportion and then computes distance

seg.dist<- daisy(credit_data_scale)

#Sample of distance matrix
as.matrix(seg.dist)[1:5,1:5]
```

```
##           1        2        3        4         5
## 1 0.0000000 4.393692 4.616890 2.455220 0.9694075
## 2 4.3936916 0.000000 5.630916 3.980908 4.1914567
## 3 4.6168904 5.630916 0.000000 4.242227 4.3428945
## 4 2.4552203 3.980908 4.242227 0.000000 2.6590634
## 5 0.9694075 4.191457 4.342894 2.659063 0.0000000
```

```
 #using hclust to form a hierarchical tree by using distance matrix
seg.hc <- hclust(seg.dist, method="complete")
plot(seg.hc)
```




```
plot(cut(as.dendrogram(seg.hc), h=20)$lower[[1]])       #we cut from the tree at a height of 20 for sam
```




```
credit_data.raw[c(508, 588), ]
```

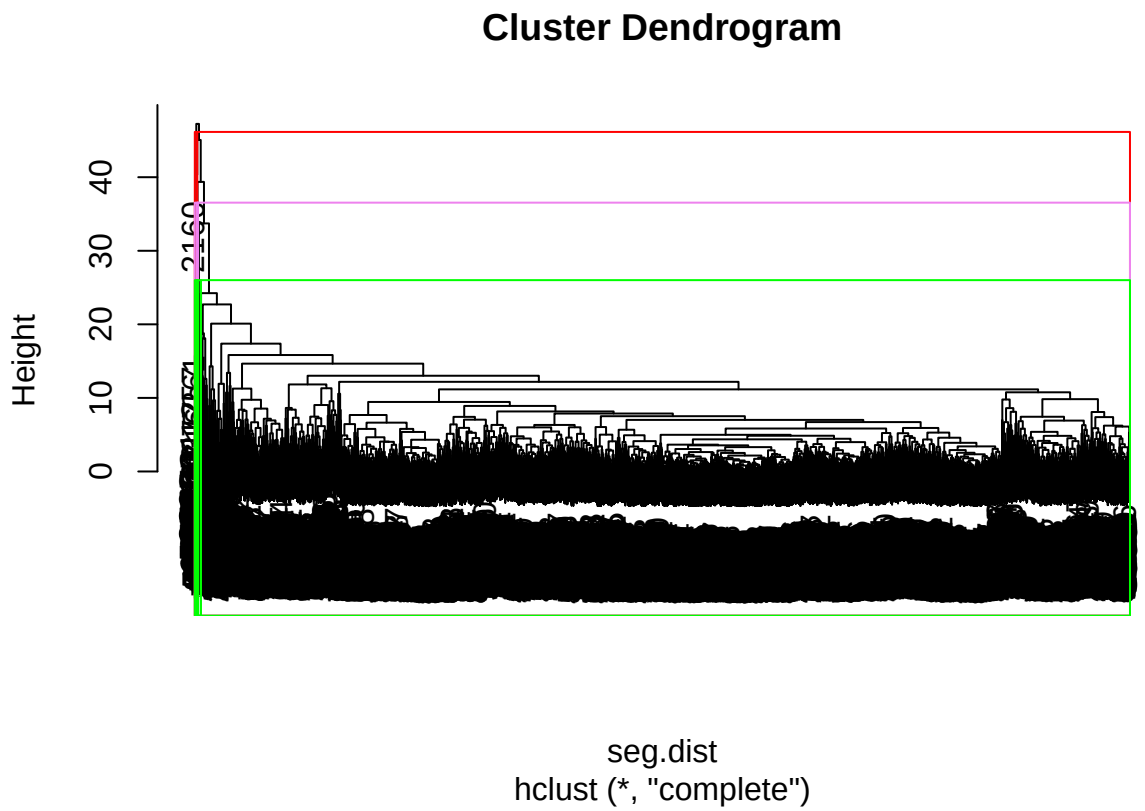
```
##       X CUST_ID  BALANCE BALANCE_FREQUENCY PURCHASES ONEOFF_PURCHASES
## 508 508  C10529 2643.343                 1  26402.39         22257.39
## 588 588  C10611 2492.726                 1  27957.68         23032.97
##     INSTALLMENTS_PURCHASES CASH_ADVANCE PURCHASES_FREQUENCY
## 508                4145.00            0                   1
## 588                4924.71            0                   1
##     ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMENTS_FREQUENCY
## 508                          1                         0.333333
## 588                          1                         0.916667
##     CASH_ADVANCE_FREQUENCY CASH_ADVANCE_TRX PURCHASES_TRX CREDIT_LIMIT PAYMENTS
## 508                      0                0           114        16500 24529.28
## 588                      0                0            70         9000 27255.01
##     MINIMUM_PAYMENTS PRC_FULL_PAYMENT TENURE
## 508         534.0323                1     12
## 588         537.3727                1     12
```

```
cor(cophenetic(seg.hc), seg.dist) #cpcc shows how well the clustering model reflects the distance matri
```

```
## [1] 0.8149523
```

```
# Plot to check on different cluster sizes
plot(seg.hc)
```

```
rect.hclust(seg.hc, k=2, border="red")
rect.hclust(seg.hc, k=4, border="violet")
rect.hclust(seg.hc, k=6, border="green")
```




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

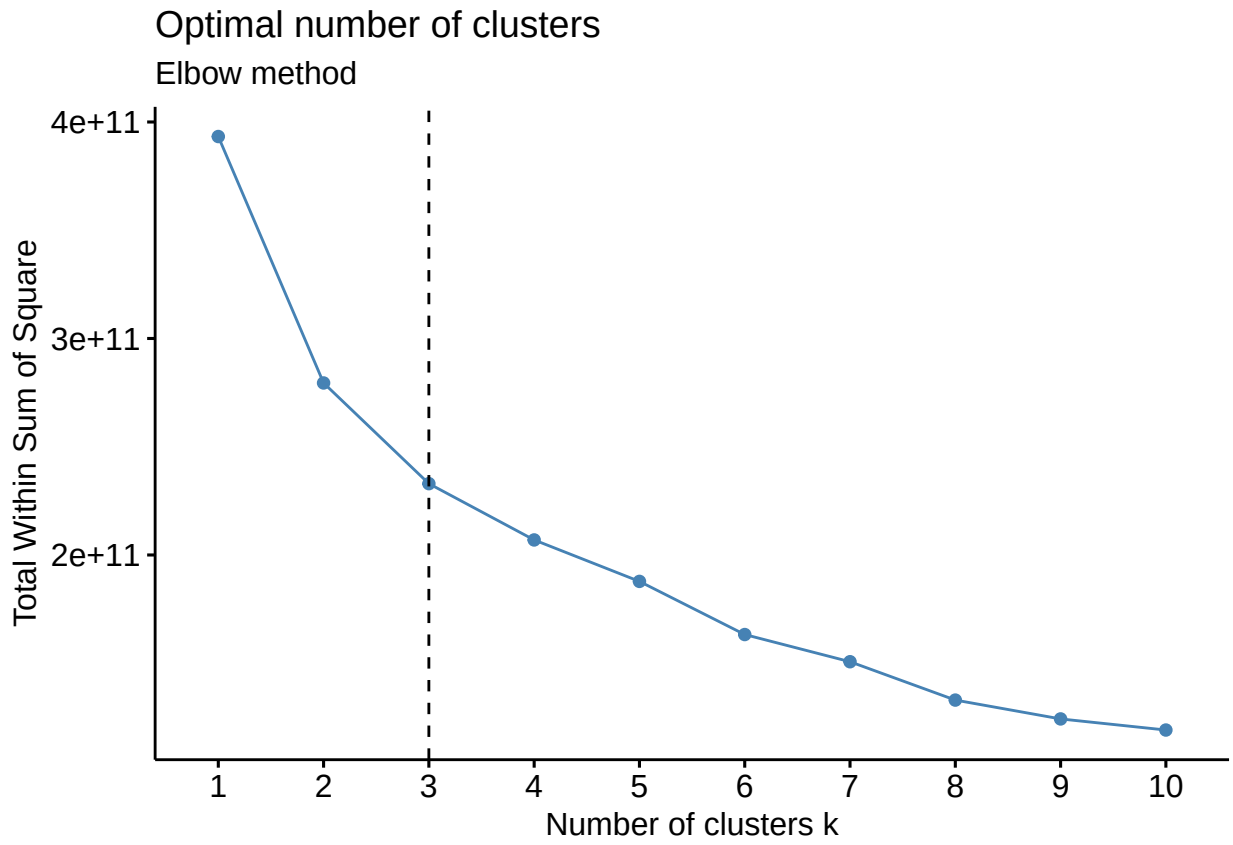
```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

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

```
fviz_nbclust(credit_data, kmeans, method = "wss") +
geom_vline(xintercept = 3, linetype = 2)+
labs(subtitle = "Elbow method")
```




```
#Based on above we cut the tree for 3 groups
seg.hc.segment <- cutree(seg.hc, k=3)   # membership vector for 3 groups
table(seg.hc.segment)
```

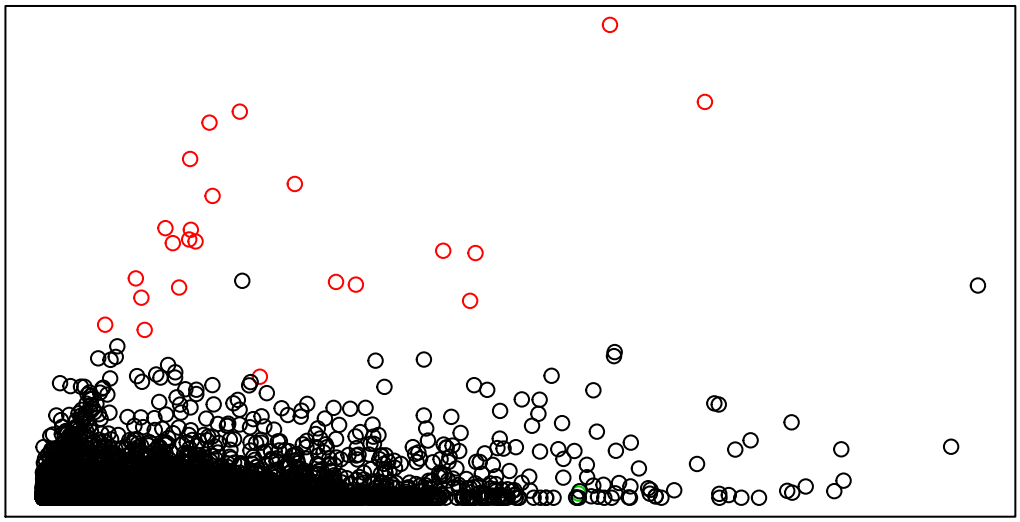
```
## seg.hc.segment
##    1    2    3
## 8926   23    1
```

```
data.summ(credit_data, seg.hc.segment)
```

```
##   Group.1   BALANCE BALANCE_FREQUENCY  PURCHASES ONEOFF_PURCHASES
## 1       1  1555.059         0.8770538   934.9797         536.2514
## 2       2  4812.383         0.9561265 27505.3396       22417.4522
## 3       3 10905.054         1.0000000   431.9300         133.5000
##   INSTALLMENTS_PURCHASES CASH_ADVANCE PURCHASES_FREQUENCY
## 1               399.0293     972.0536           0.4892715
## 2              5087.8874    1617.7861           0.9050724
## 3               298.4300   47137.2118           0.5833330
##   ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMENTS_FREQUENCY
## 1                  0.2007931                         0.3635351
## 2                  0.8463769                         0.7086956
## 3                  0.2500000                         0.5000000
##   CASH_ADVANCE_FREQUENCY CASH_ADVANCE_TRX PURCHASES_TRX CREDIT_LIMIT  PAYMENTS
## 1             0.13523683         3.237060      14.42718      4463.11  1660.922
## 2             0.06159417         2.608696     124.13043     16000.00 28138.985
```

```
## 3             1.00000000       123.000000      21.00000     19600.00 39048.598
##   MINIMUM_PAYMENTS PRC_FULL_PAYMENT   TENURE
## 1         828.9243        0.1527534 11.51624
## 2        2599.0909        0.5334321 11.91304
## 3        5394.1737        0.0000000 12.00000
```

```
plot(jitter(as.numeric(credit_data$PURCHASES)) ~ jitter(as.numeric(credit_data$BALANCE)),       col=seg
```

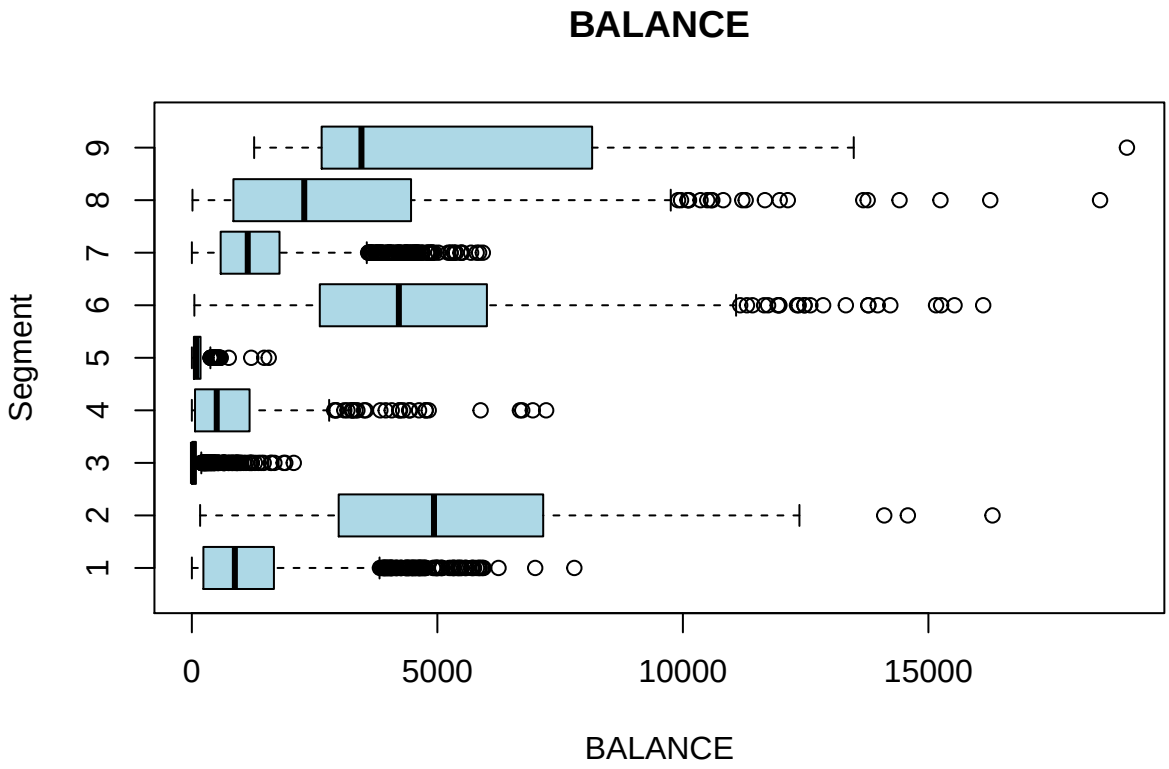


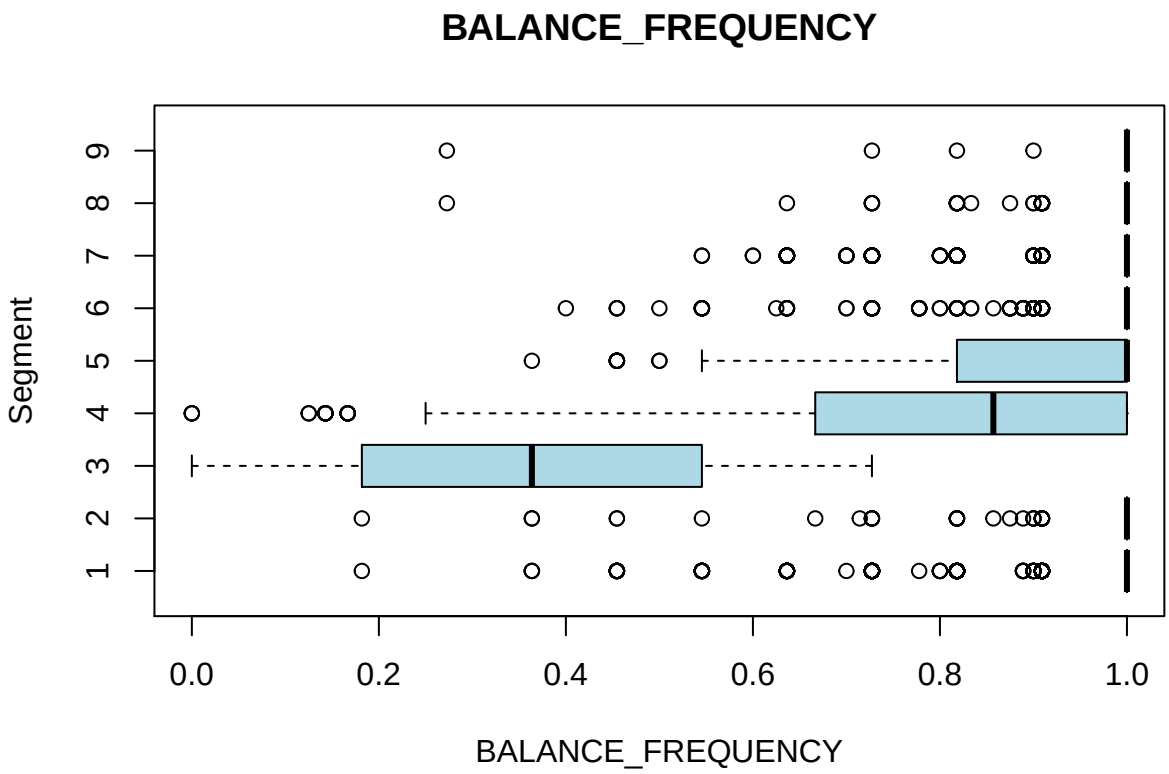
```
#k-Means method
# number of k group was selected after multiple iteration and also referring the mclust
seg.k <- kmeans(credit_data_scale, centers=9)
data.summ(credit_data, seg.k$cluster)
```

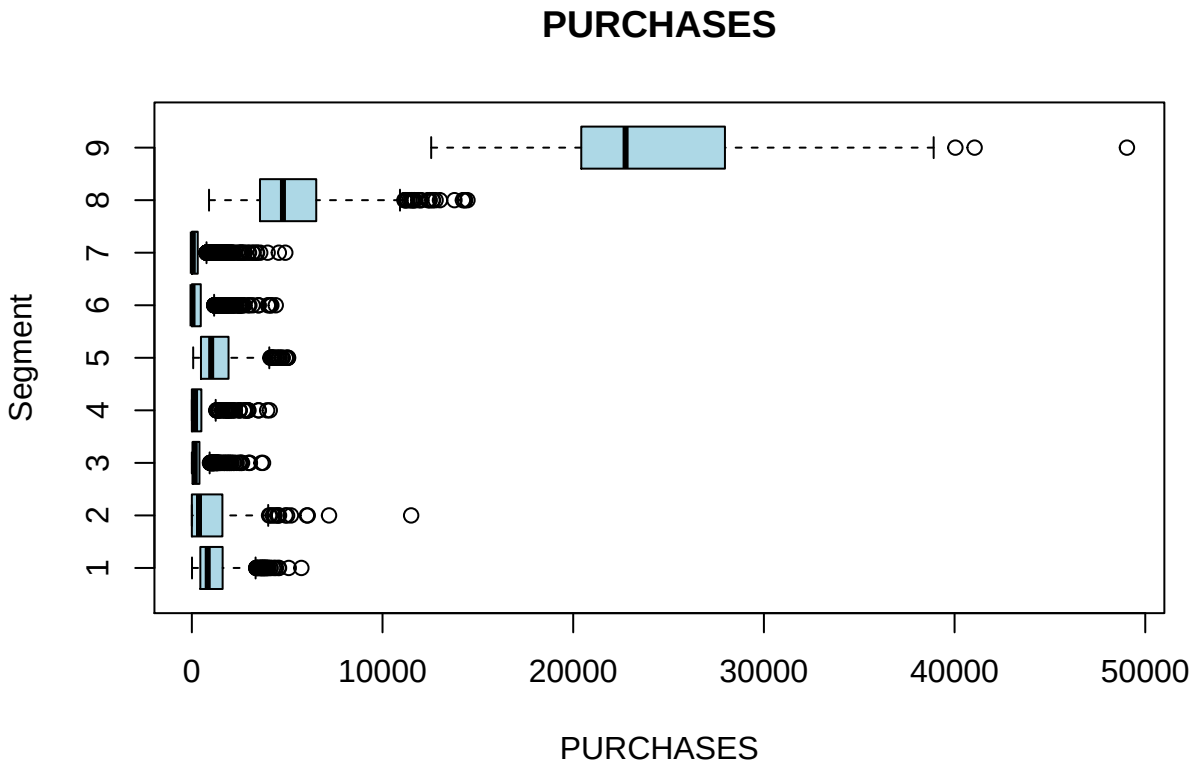
```
##   Group.1   BALANCE BALANCE_FREQUENCY  PURCHASES ONEOFF_PURCHASES
## 1       1 1201.6917         0.9707153  1105.8495         512.9371
## 2       2 5260.6708         0.9524638  1134.5458         665.1059
## 3       3  106.9111         0.3523941   315.0669         190.2356
## 4       4  851.6887         0.7942606   385.8174         247.9074
## 5       5  129.2708         0.9111572  1319.5333         616.3755
## 6       6 4553.9713         0.9758161   366.0156         240.7264
## 7       7 1340.2645         0.9708072   249.3716         205.6638
## 8       8 3127.3023         0.9893151  5399.2597        3463.3085
## 9       9 5519.2375         0.9557994 25421.9383       18707.4228
##   INSTALLMENTS_PURCHASES CASH_ADVANCE PURCHASES_FREQUENCY
## 1              593.29866    288.70767           0.8757008
```

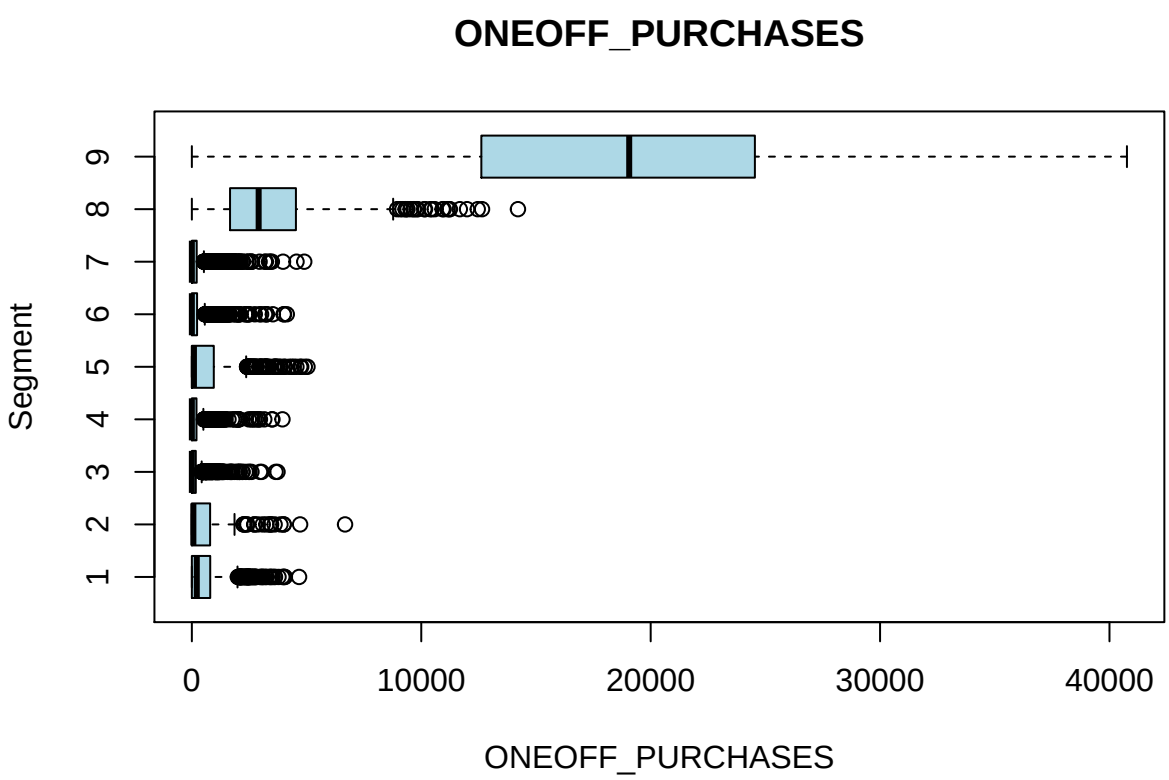
```
## 2              469.43988  10098.83863           0.4129682
## 3              125.09922    321.05182           0.2586243
## 4              138.39780   1108.57517           0.4136024
## 5              703.25420     37.76055           0.8601567
## 6              125.38726   3454.24888           0.2391794
## 7               43.91036    609.35475           0.1493029
## 8             1937.04611    565.33769           0.9550183
## 9             6714.51552   1401.09676           0.9074712
##   ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMENTS_FREQUENCY
## 1                 0.27980488                        0.71137786
## 2                 0.21602255                        0.30573573
## 3                 0.06553617                        0.18609265
## 4                 0.12202090                        0.27318235
## 5                 0.31444365                        0.65863298
## 6                 0.11463645                        0.14389397
## 7                 0.08996138                        0.05837768
## 8                 0.72087600                        0.79305663
## 9                 0.76609200                        0.74597700
##   CASH_ADVANCE_FREQUENCY CASH_ADVANCE_TRX PURCHASES_TRX CREDIT_LIMIT   PAYMENTS
## 1             0.05513745       1.0725264     21.305476     3848.964  1279.8600
## 2             0.64488215      29.4848485     16.339394     9354.848 10643.1679
## 3             0.03287452       0.7193141      4.097473     3798.795  1133.8009
## 4             0.19492899       3.2048780      5.126829     2415.471   594.2495
## 5             0.00651996       0.1213434     20.752979     5044.344  1430.4664
## 6             0.44137626      11.5699132      5.438766     7292.001  2334.1259
## 7             0.13281853       2.4686860      2.715514     3082.610   929.3490
## 8             0.07724783       1.8394161     75.447080     8708.029  4968.3909
## 9             0.05172410       2.1379310    142.482759    15624.138 25353.2578
##   MINIMUM_PAYMENTS PRC_FULL_PAYMENT    TENURE
## 1         858.3048       0.05698105 11.901057
## 2        2194.0887       0.10679539 11.575758
## 3         131.1111       0.18977085 11.784296
## 4         326.1832       0.14496709  7.222764
## 5         183.6771       0.78976752 11.771398
## 6        2122.6107       0.02097680 11.659595
## 7         663.0414       0.02311058 11.889889
## 8        1598.7898       0.21663564 11.941606
## 9        3539.1300       0.48915883 11.931034
```

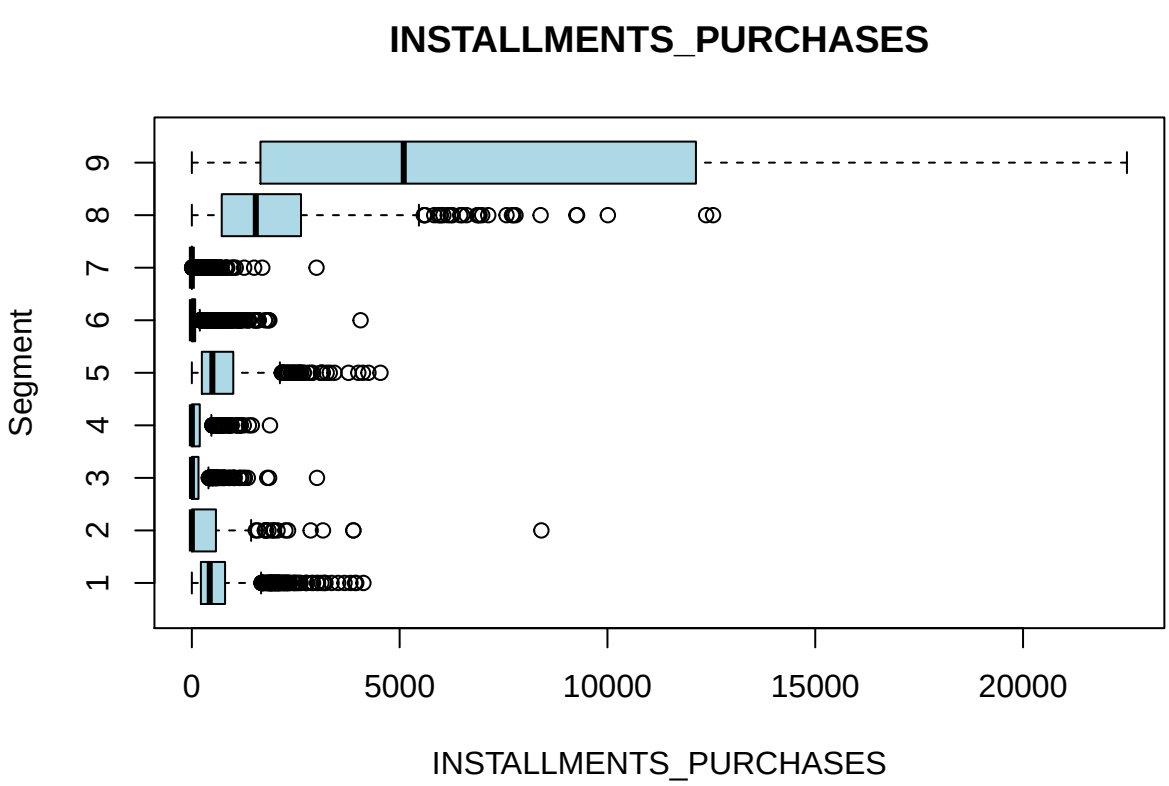
```
for ( i in seq(1,length(credit_data ),1) ) boxplot(credit_data[,i] ~ seg.k$cluster, main=names(credit_da
```

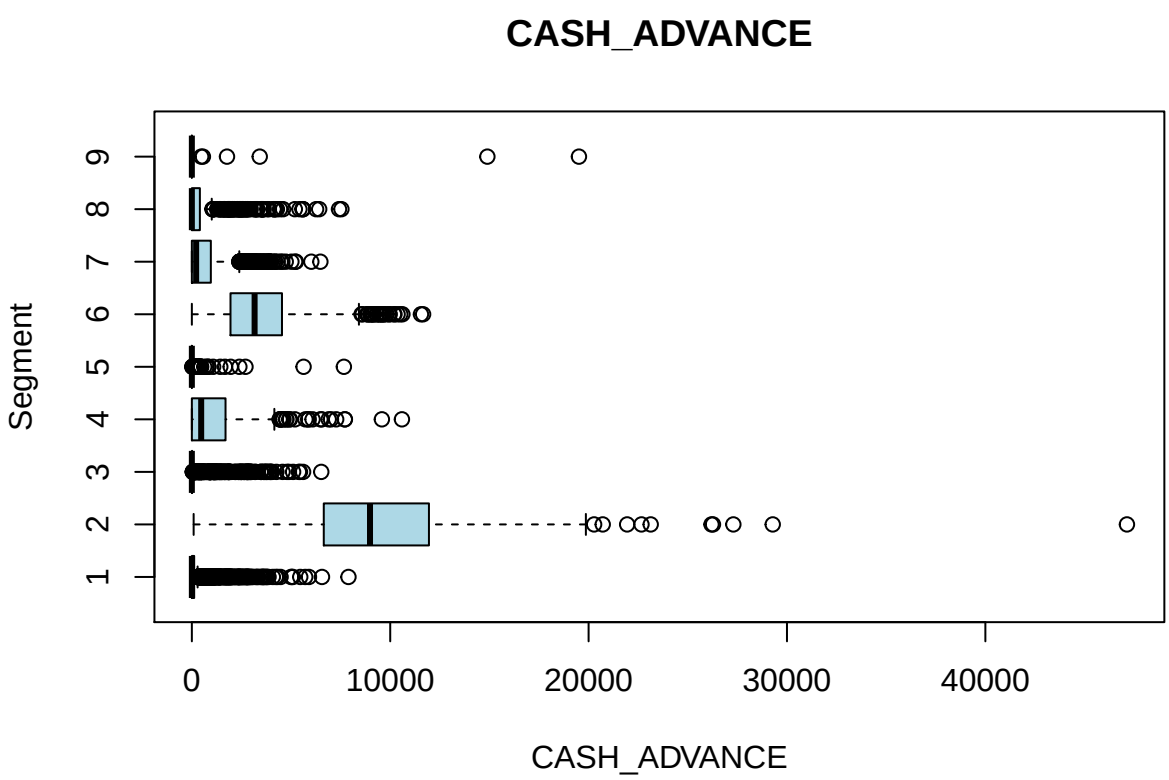

BALANCE
Segment
9
8
7
6
5
4
3
2
1
0
5000
10000
15000
BALANCE

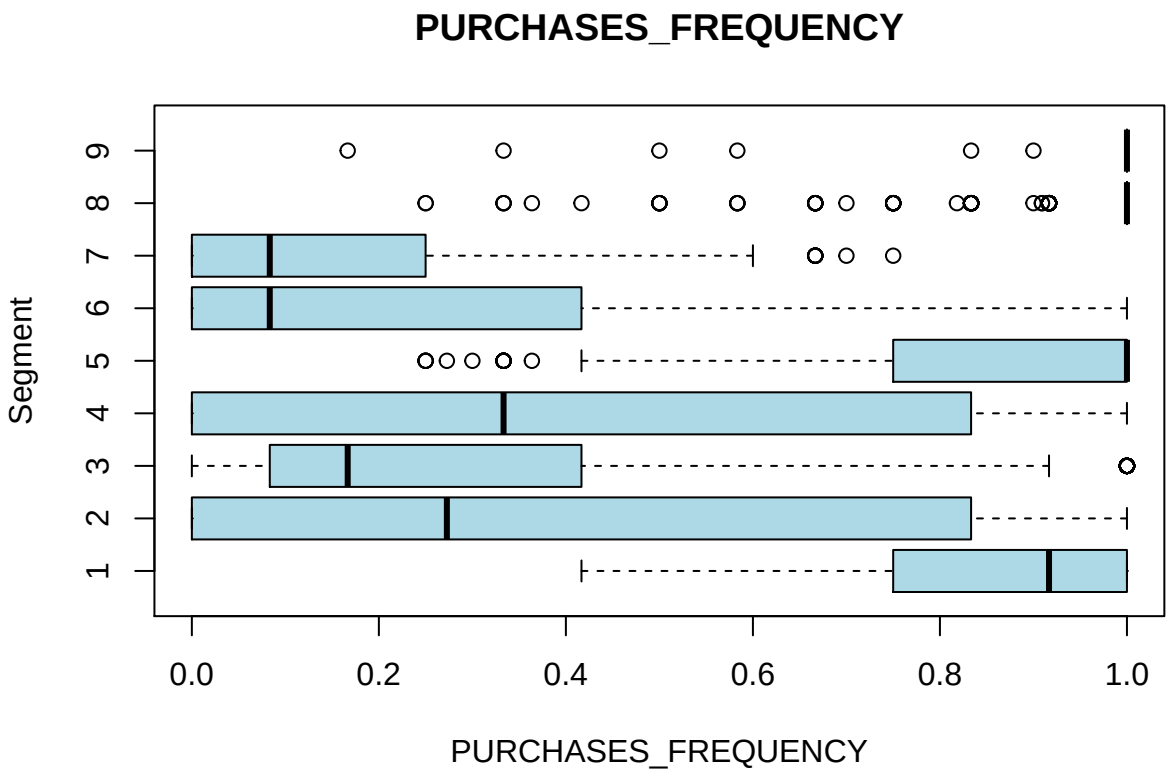

BALANCE_FREQUENCY
Segment
1
2
3
4
5
6
7
8
9
0.0
0.2
0.4
0.6
0.8
1.0
BALANCE_FREQUENCY

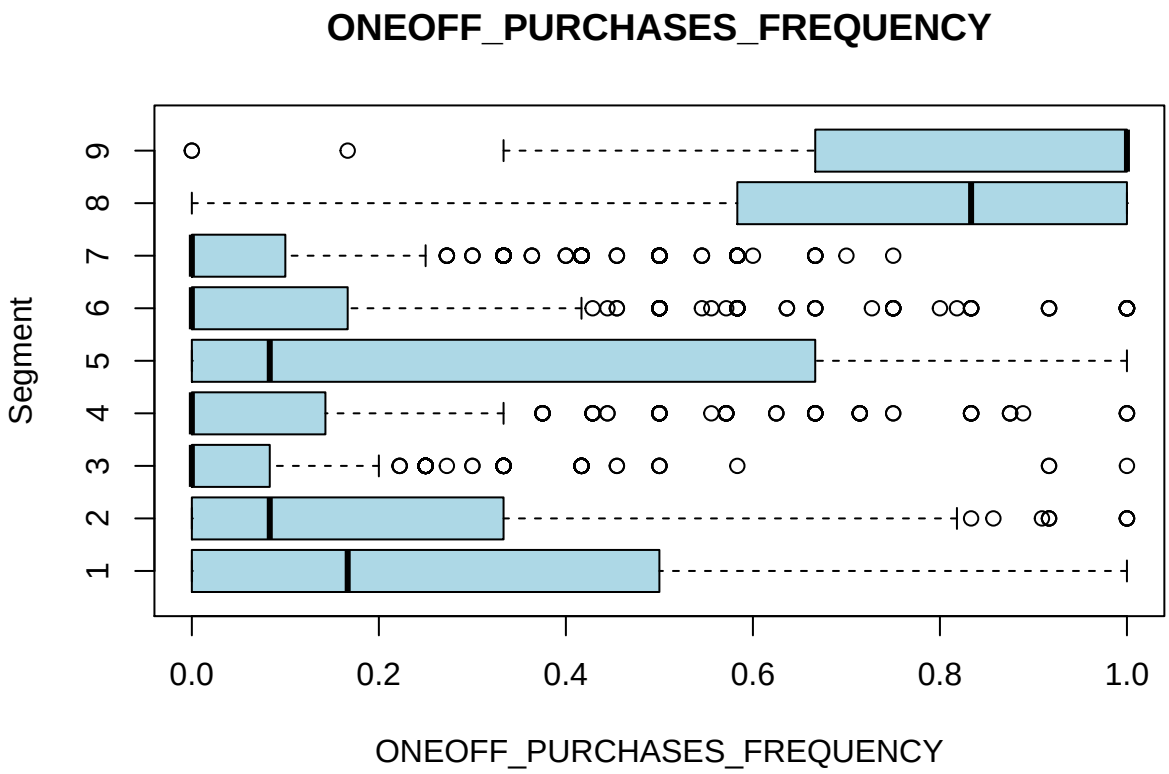

PURCHASES
Segment
9
8
7
6
5
4
3
2
1
0
10000
20000
30000
40000
50000
PURCHASES

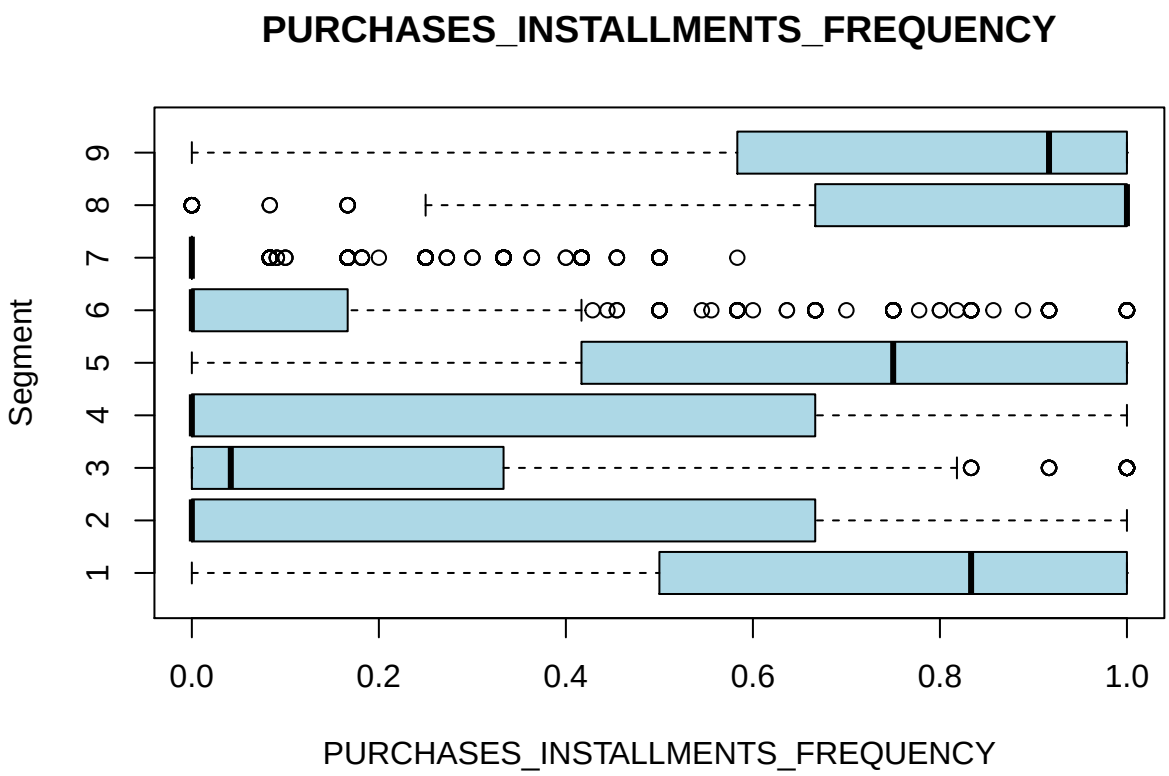

ONEOFF_PURCHASES
Segment
9
8
7
6
5
4
3
2
1
0
10000
20000
30000
40000
ONEOFF_PURCHASES

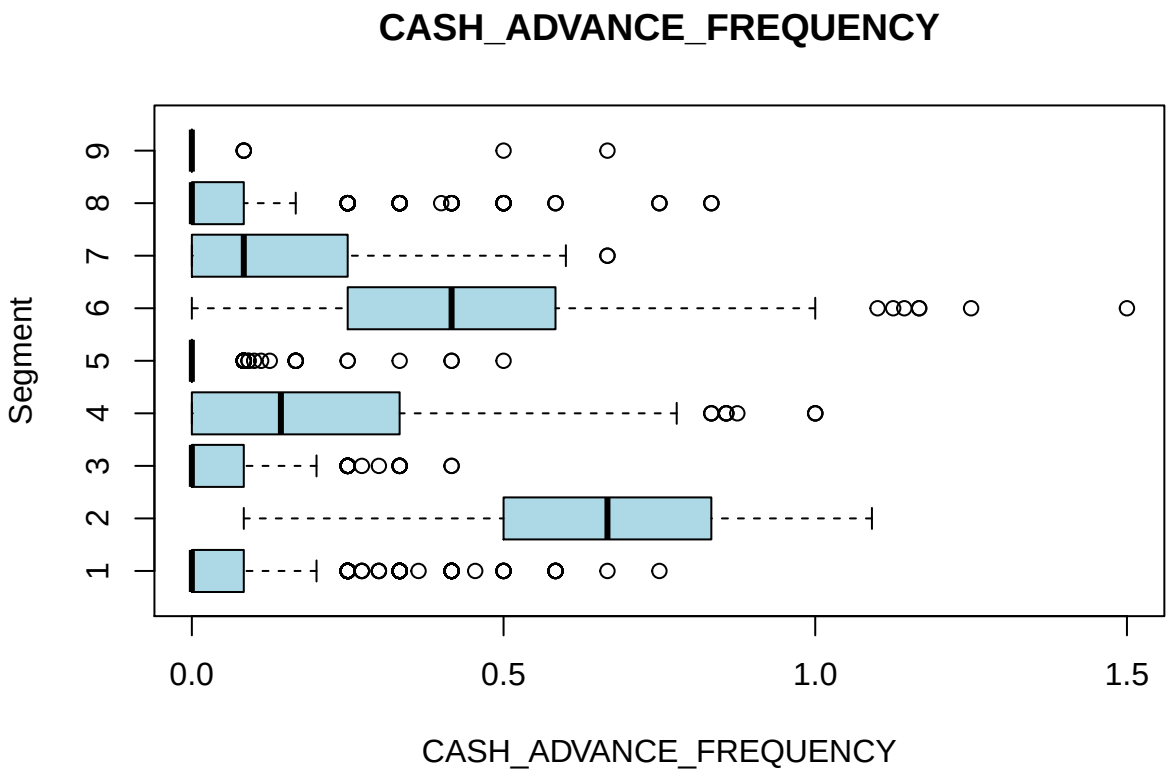

INSTALLMENTS_PURCHASES
Segment
9
8
7
6
5
4
3
2
1
0
5000
10000
15000
20000
INSTALLMENTS_PURCHASES

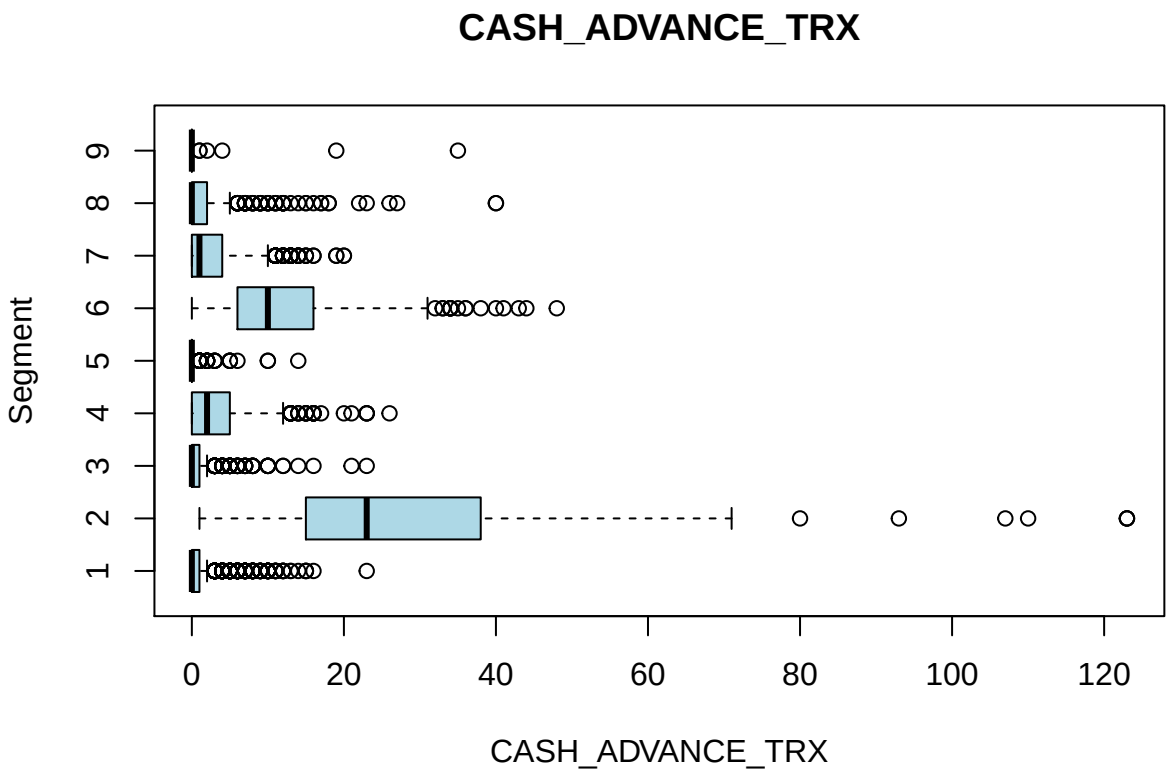

CASH_ADVANCE
Segment
9
8
7
6
5
4
3
2
1
0
10000
20000
30000
40000
CASH_ADVANCE

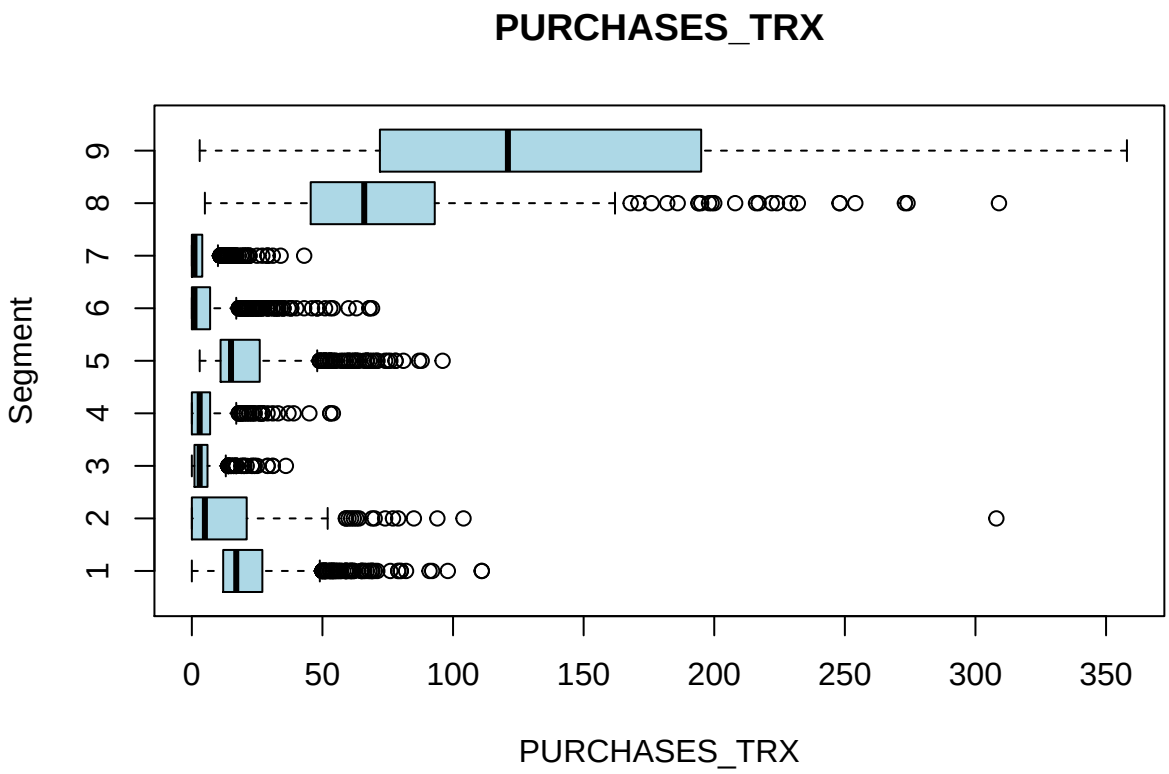

PURCHASES_FREQUENCY
Segment
9
8
7
6
5
4
3
2
1
0.0
0.2
0.4
0.6
0.8
1.0
PURCHASES_FREQUENCY

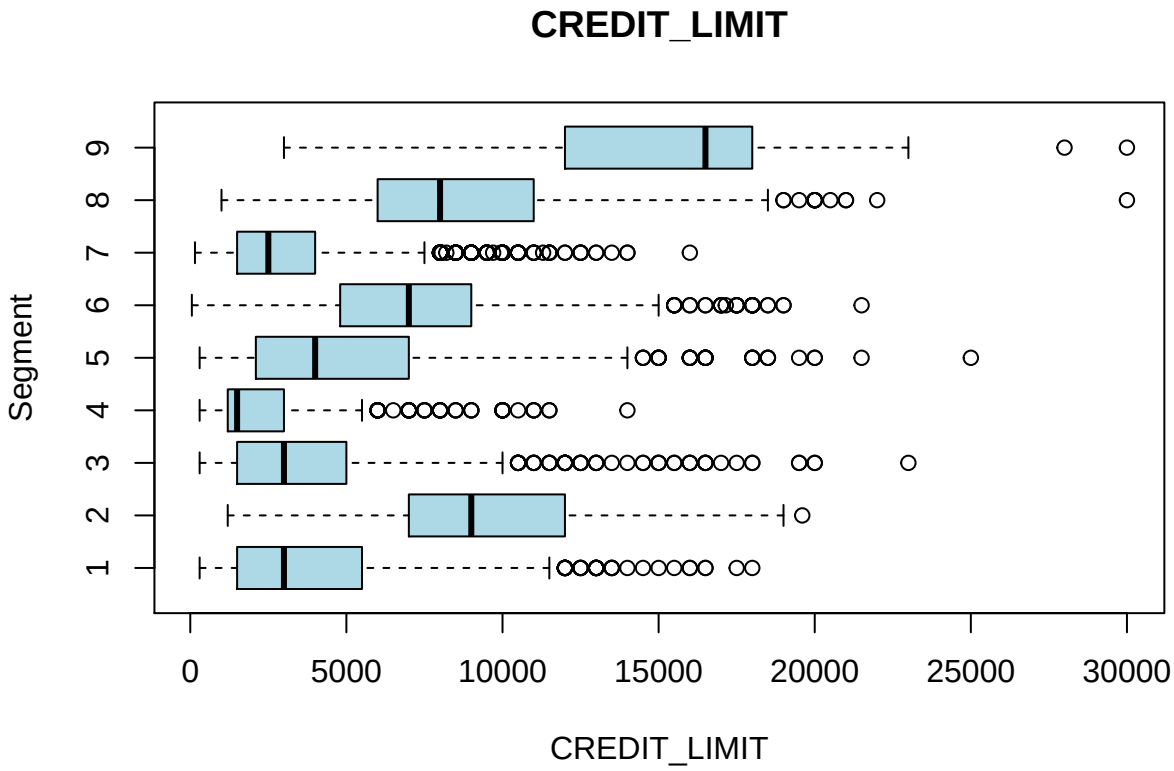

ONEOFF_PURCHASES_FREQUENCY
Segment
9
8
7
6
5
4
3
2
1
0.0
0.2
0.4
0.6
0.8
1.0
ONEOFF_PURCHASES_FREQUENCY

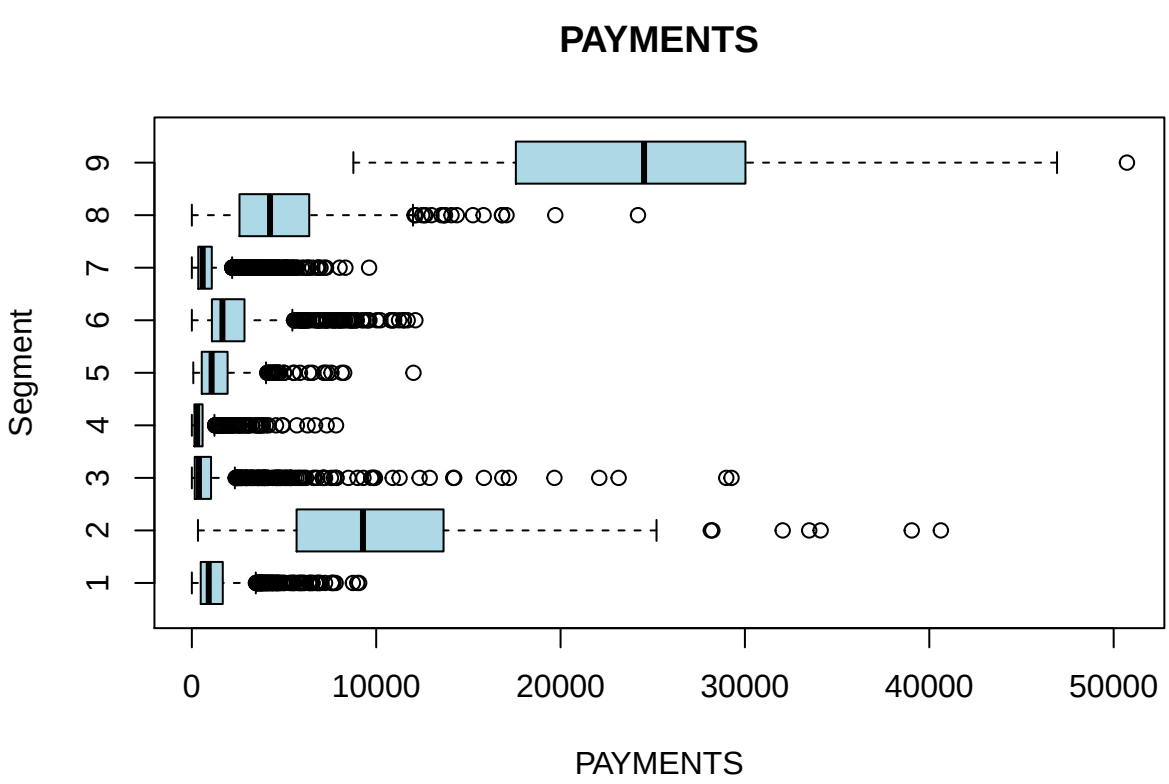

PURCHASES_INSTALLMENTS_FREQUENCY
Segment
9
8
7
6
5
4
3
2
1
0.0
0.2
0.4
0.6
0.8
1.0
PURCHASES_INSTALLMENTS_FREQUENCY

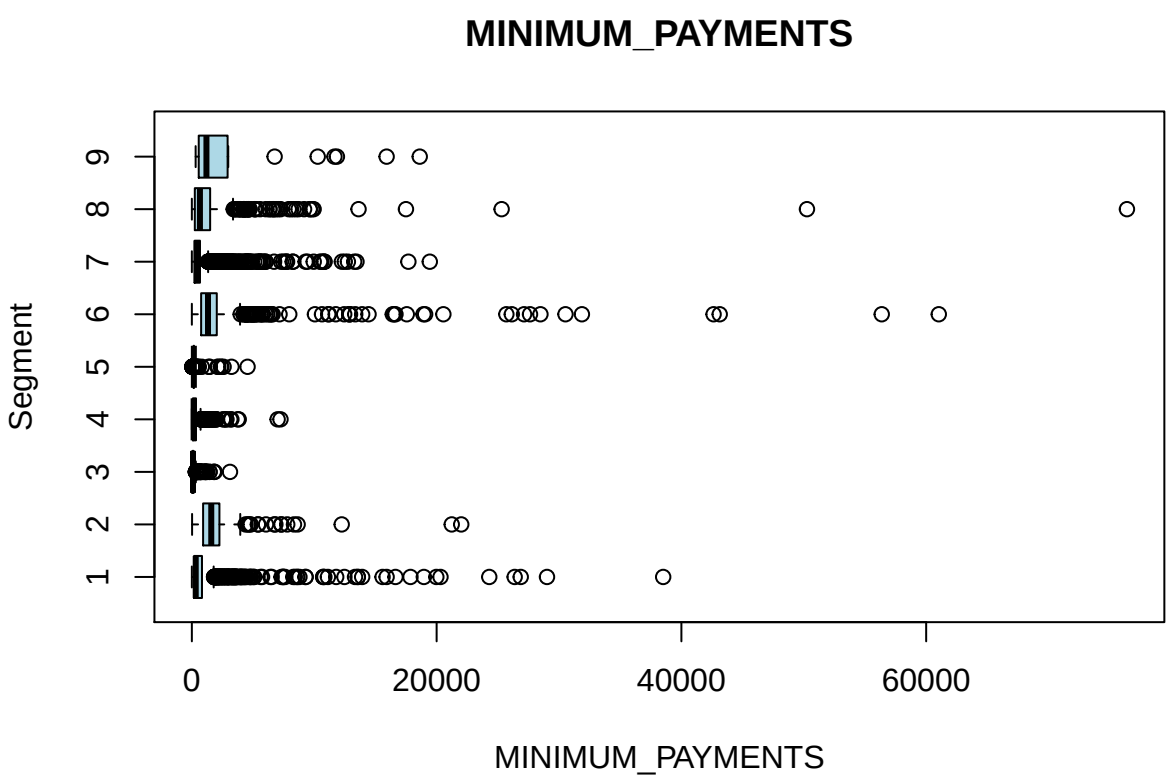

CASH_ADVANCE_FREQUENCY
Segment
9
8
7
6
5
4
3
2
1
0.0
0.5
1.0
1.5
CASH_ADVANCE_FREQUENCY

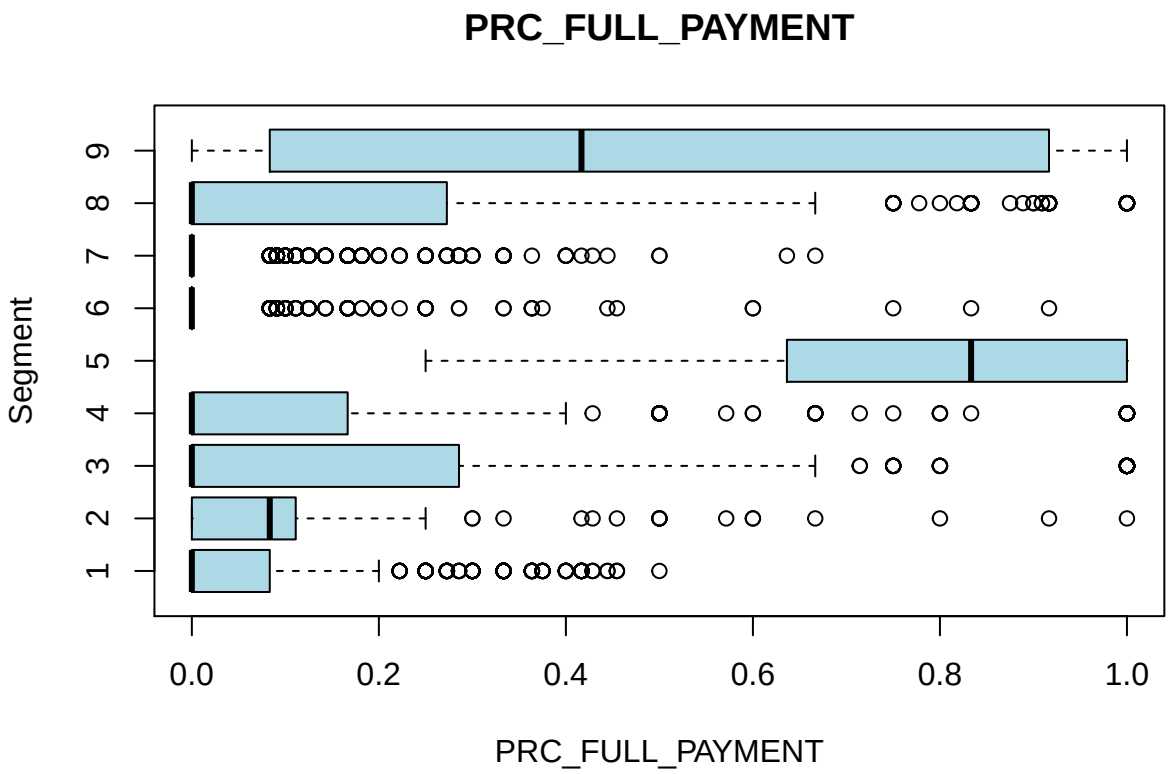

CASH_ADVANCE_TRX
Segment
9
8
7
6
5
4
3
2
1
0
20
40
60
80
100
120
CASH_ADVANCE_TRX

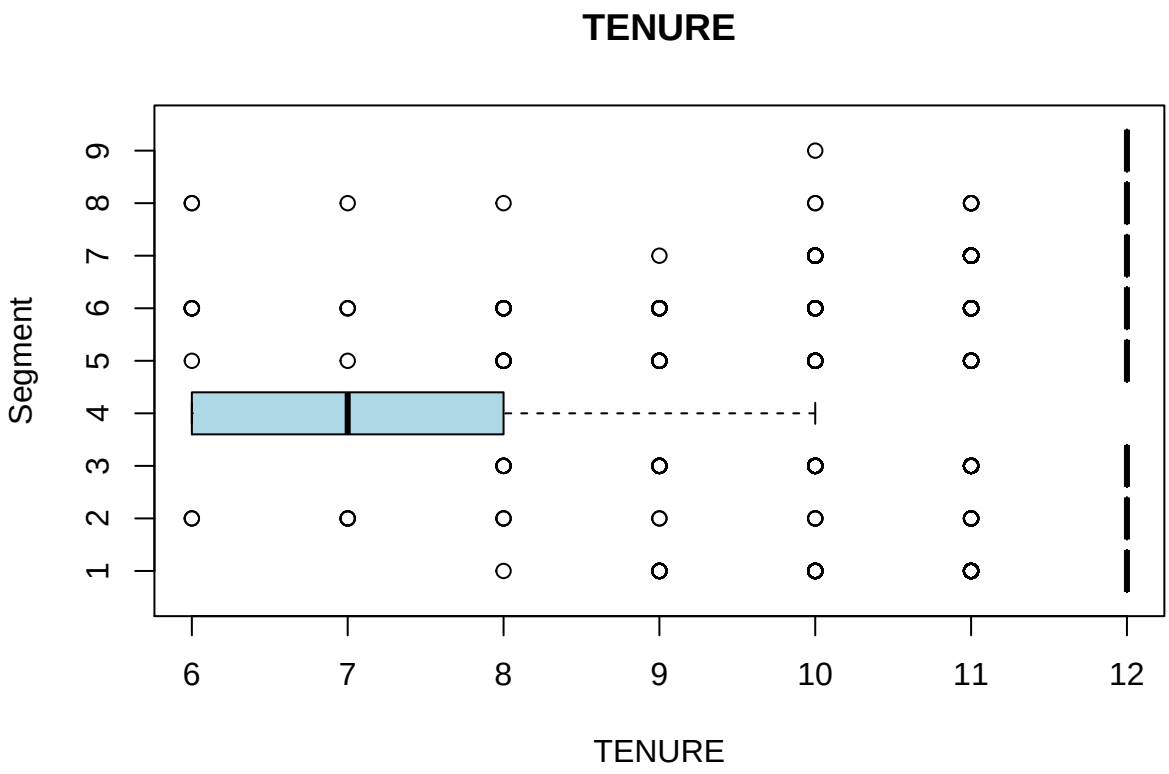

PURCHASES_TRX
Segment
9
8
7
6
5
4
3
2
1
0
50
100
150
200
250
300
350
PURCHASES_TRX


CREDIT_LIMIT
Segment
9
8
7
6
5
4
3
2
1
0
5000
10000
15000
20000
25000
30000
CREDIT_LIMIT

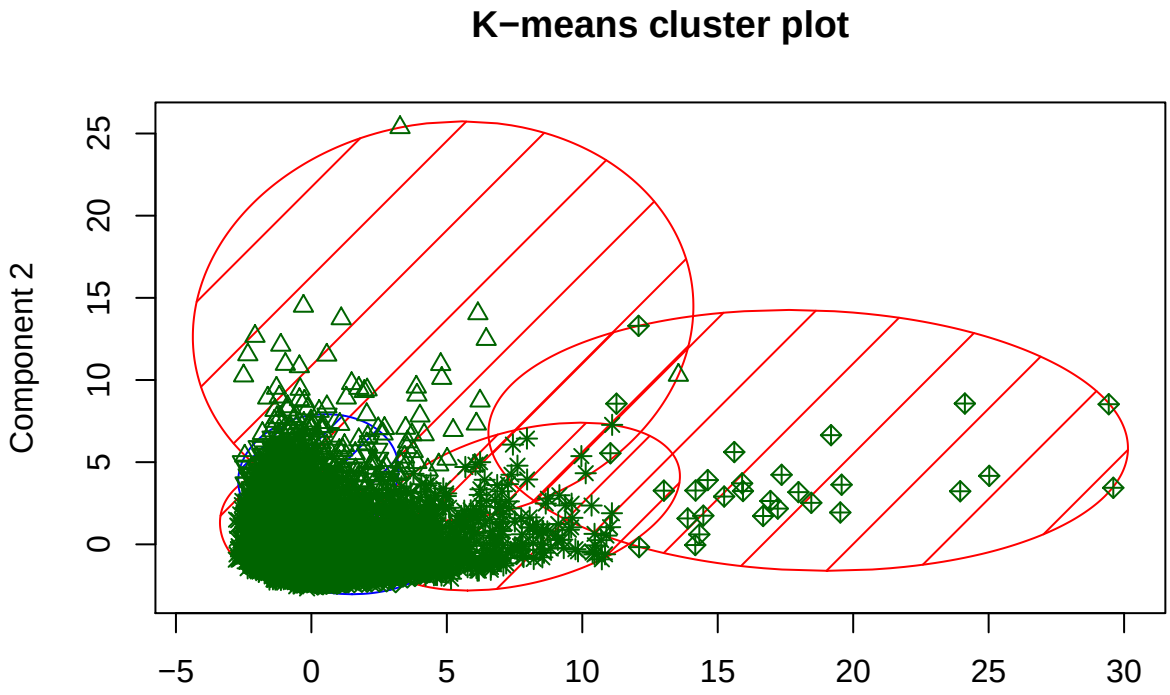

PAYMENTS
Segment
9
8
7
6
5
4
3
2
1
0
10000
20000
30000
40000
50000
PAYMENTS


MINIMUM_PAYMENTS
Segment
9
8
7
6
5
4
3
2
1
0
20000
40000
60000
MINIMUM_PAYMENTS


PRC_FULL_PAYMENT
Segment
9
8
7
6
5
4
3
2
1
0.0
0.2
0.4
0.6
0.8
1.0
PRC_FULL_PAYMENT




```
# cluster plot to see the observations in the group
clusplot(credit_data, seg.k$cluster, color=TRUE, shade=TRUE, labels=9, lines=0, main="K-means cluster pl
```




```
summary(seg.k)
```

```
##              Length Class  Mode
## cluster      8950   -none- numeric
## centers       153   -none- numeric
## totss           1   -none- numeric
## withinss        9   -none- numeric
## tot.withinss    1   -none- numeric
## betweenss       1   -none- numeric
## size            9   -none- numeric
## iter            1   -none- numeric
## ifault          1   -none- numeric
```

```
#Mclust method

# setting up random seed to remove the biasness of model to select same observations in the sample
set.seed(96743)
#install.packages("mclust")
library(mclust)
```

```
## Package 'mclust' version 5.4.6
## Type 'citation("mclust")' for citing this R package in publications.

##
## Attaching package: 'mclust'
```

```
## The following object is masked from 'package:psych':
##
##     sim
```

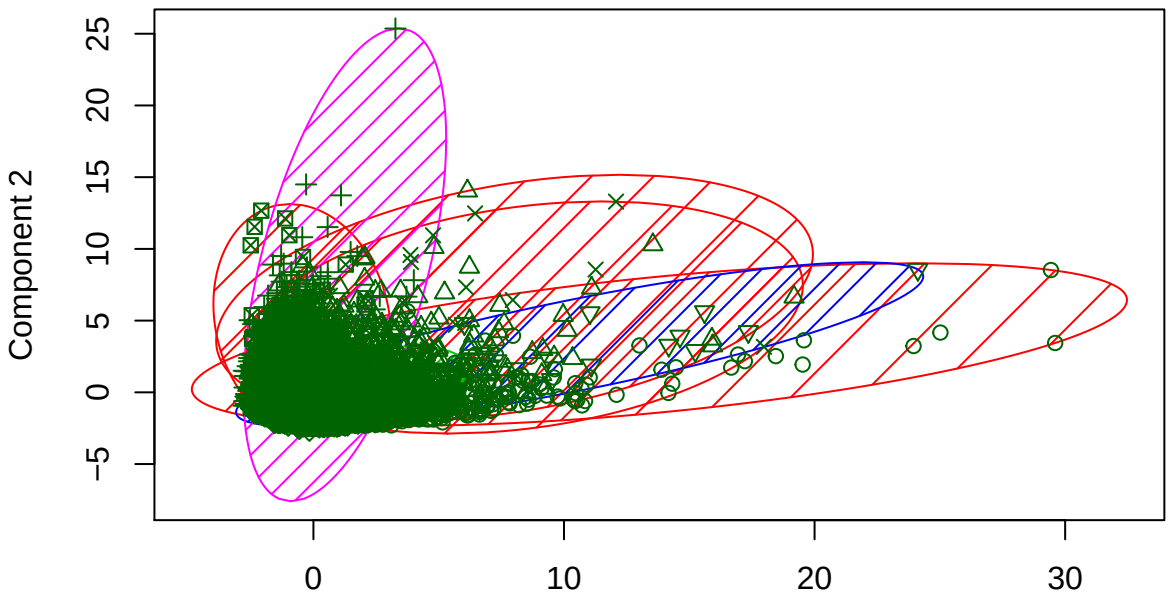
```
#Running mclust on data before scaling
seg.mc_data <- Mclust(credit_data)
#Running mclust on data after scaling
seg.mc_scale <- Mclust(credit_data_scale)
#Running mclust with rincipal component analysis
seg.mc_pc<- Mclust(credit_data_pc$x)
#Running mclust with overriden group=4
seg.mc_scale4 <- Mclust(credit_data_scale, G=4)

#comparing above models for statistical significance
BIC(seg.mc_data,seg.mc_scale,seg.mc_pc,seg.mc_scale4)
```

```
##                 df       BIC
## seg.mc_data   1402 943225.80
## seg.mc_scale   480  69925.18
## seg.mc_pc     1094  20488.99
## seg.mc_scale4  632 101748.41
```

```
#Summary of the model which has the lowest BIC value
summary(seg.mc_pc)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EEV (ellipsoidal, equal volume and shape) model with 7 components:
##
##  log-likelihood    n   df       BIC       ICL
##       -5267.116 8950 1094 -20488.99 -21333.51
##
## Clustering table:
##    1    2    3    4    5    6    7
##  700 1098 2873  111 2233 1682  253
```

```
# Plot the groups
clusplot(credit_data, seg.mc_pc$class, color=TRUE, shade=TRUE, labels=7, lines=0, main="Model-based clus
```



These two components explain 47.62 % of the point variability.

```
# Attributes summary for each clusters
data.summ(credit_data,seg.mc_pc$class)
```

```
##   Group.1   BALANCE BALANCE_FREQUENCY PURCHASES ONEOFF_PURCHASES
## 1       1  917.5157         0.9754546 3945.5567        2588.1762
## 2       2 3475.3192         0.9996688 1354.5822         627.6138
## 3       3 2151.9635         0.9071967  142.0709         140.8687
## 4       4 3188.0582         0.9377920 1854.9192         921.8739
## 5       5  112.2660         0.6646972  667.0962         277.8202
## 6       6 1433.2396         0.9889201 1484.4968         962.3229
## 7       7 1367.6820         0.8419556  509.2766         219.0825
##   INSTALLMENTS_PURCHASES CASH_ADVANCE PURCHASES_FREQUENCY
## 1            1359.739643   370.415471          0.95000002
## 2             727.575200  1744.441652          0.68237705
## 3               1.202235  1932.077764          0.07597625
## 4             933.045315  3365.067779          0.69763367
## 5             389.289436    18.711782          0.60647614
## 6             522.187063     1.928592          0.70650921
## 7             291.447036  2437.942488          0.53778882
##   ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMENTS_FREQUENCY
## 1                 0.76107145                       0.665595266
## 2                 0.21964174                       0.588342467
## 3                 0.07533894                       0.001397257
## 4                 0.12992859                       0.613424477
## 5                 0.11029555                       0.500269339
## 6                 0.31157441                       0.508152290
```

```
## 7                0.14565472                    0.418495941
##   CASH_ADVANCE_FREQUENCY CASH_ADVANCE_TRX PURCHASES_TRX CREDIT_LIMIT  PAYMENTS
## 1           0.0592856871      1.098571429     52.777143     6951.110 3869.6294
## 2           0.2563751940      6.462659381     26.101093     5290.665 2340.1520
## 3           0.2623738138      5.939436129      1.414201     4109.281 1665.3710
## 4           0.2622550631      8.117117117     21.216216     4537.838 3386.2320
## 5           0.0064548012      0.075235110     10.670846     3815.964  807.4388
## 6           0.0006080404      0.008323424     19.876932     4591.577 1599.6310
## 7           0.3485931818     12.114624506      9.371542     3939.328 2289.8698
##   MINIMUM_PAYMENTS PRC_FULL_PAYMENT   TENURE
## 1         326.9777       0.48639476 12.00000
## 2        2372.1770       0.01090226 12.00000
## 3         717.5676       0.02924398 11.32022
## 4        9344.5296       0.03260013 10.00901
## 5         129.1959       0.37370189 11.26019
## 6         673.1520       0.03206320 11.99287
## 7         438.9982       0.18678225 10.09486
```

‘‘‘