Credit card holders segmentation

Ran Cao

8/2/2020

```
library(readr)
creditcard <- read.csv("~/Library/Containers/com.tencent.xinWeChat/Data/Desktop/CCGENERAL.csv")
summary(creditcard)
```

```
##     CUST_ID        BALANCE        BALANCE_FREQUENCY   PURCHASES
##  C10001 :   1   Min.   :    0.0   Min.   :0.0000    Min.   :    0.00
##  C10002 :   1   1st Qu.:  128.3   1st Qu.:0.8889    1st Qu.:   39.63
##  C10003 :   1   Median :  873.4   Median :1.0000    Median :  361.28
##  C10004 :   1   Mean   : 1564.5   Mean   :0.8773    Mean   : 1003.20
##  C10005 :   1   3rd Qu.: 2054.1   3rd Qu.:1.0000    3rd Qu.: 1110.13
##  C10006 :   1   Max.   :19043.1   Max.   :1.0000    Max.   :49039.57
##  (Other):8944
##  ONEOFF_PURCHASES  INSTALLMENTS_PURCHASES  CASH_ADVANCE     PURCHASES_FREQUENCY
##  Min.   :    0.0   Min.   :    0.0        Min.   :    0.0   Min.   :0.00000
##  1st Qu.:    0.0   1st Qu.:    0.0        1st Qu.:    0.0   1st Qu.:0.08333
##  Median :   38.0   Median :   89.0        Median :    0.0   Median :0.50000
##  Mean   :  592.4   Mean   :  411.1        Mean   :  978.9   Mean   :0.49035
##  3rd Qu.:  577.4   3rd Qu.:  468.6        3rd Qu.: 1113.8   3rd Qu.:0.91667
##  Max.   :40761.2   Max.   :22500.0        Max.   :47137.2   Max.   :1.00000
##
##  ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMENTS_FREQUENCY
##  Min.   :0.00000            Min.   :0.0000
##  1st Qu.:0.00000            1st Qu.:0.0000
##  Median :0.08333            Median :0.1667
##  Mean   :0.20246            Mean   :0.3644
##  3rd Qu.:0.30000            3rd Qu.:0.7500
##  Max.   :1.00000            Max.   :1.0000
##
##  CASH_ADVANCE_FREQUENCY CASH_ADVANCE_TRX  PURCHASES_TRX     CREDIT_LIMIT
##  Min.   :0.0000         Min.   :  0.000   Min.   :  0.00   Min.   :   50
##  1st Qu.:0.0000         1st Qu.:  0.000   1st Qu.:  1.00   1st Qu.: 1600
##  Median :0.0000         Median :  0.000   Median :  7.00   Median : 3000
##  Mean   :0.1351         Mean   :  3.249   Mean   : 14.71   Mean   : 4494
##  3rd Qu.:0.2222         3rd Qu.:  4.000   3rd Qu.: 17.00   3rd Qu.: 6500
##  Max.   :1.5000         Max.   :123.000   Max.   :358.00   Max.   :30000
##                                                            NA's   :1
##     PAYMENTS       MINIMUM_PAYMENTS   PRC_FULL_PAYMENT     TENURE
##  Min.   :    0.0   Min.   :    0.02   Min.   :0.0000   Min.   : 6.00
##  1st Qu.:  383.3   1st Qu.:  169.12   1st Qu.:0.0000   1st Qu.:12.00
##  Median :  856.9   Median :  312.34   Median :0.0000   Median :12.00
##  Mean   : 1733.1   Mean   :  864.21   Mean   :0.1537   Mean   :11.52
##  3rd Qu.: 1901.1   3rd Qu.:  825.49   3rd Qu.:0.1429   3rd Qu.:12.00
##  Max.   :50721.5   Max.   :76406.21   Max.   :1.0000   Max.   :12.00
##                    NA's   :313
```

```r
sapply(creditcard, class)
```

```
##                          CUST_ID                          BALANCE
##                         "factor"                        "numeric"
##                 BALANCE_FREQUENCY                        PURCHASES
##                         "numeric"                        "numeric"
##                  ONEOFF_PURCHASES            INSTALLMENTS_PURCHASES
##                         "numeric"                        "numeric"
##                      CASH_ADVANCE               PURCHASES_FREQUENCY
##                         "numeric"                        "numeric"
##       ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMENTS_FREQUENCY
##                         "numeric"                        "numeric"
##            CASH_ADVANCE_FREQUENCY                  CASH_ADVANCE_TRX
##                         "numeric"                        "integer"
##                     PURCHASES_TRX                     CREDIT_LIMIT
##                         "integer"                        "numeric"
##                          PAYMENTS                  MINIMUM_PAYMENTS
##                         "numeric"                        "numeric"
##                  PRC_FULL_PAYMENT                           TENURE
##                         "numeric"                        "integer"
```

```r
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
```

```r
library(caret)
```

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

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

```r
library(ggplot2)
library(ggcorrplot)
library(tidyverse)  # data manipulation
```

```
## -- Attaching packages --------------------------------------------------------- tidyverse 1.3.0 --
```

```
## v tibble  2.1.3     v stringr 1.4.0
## v tidyr   1.0.0     v forcats 0.4.0
## v purrr   0.3.3
```

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

```
library(cluster)    # clustering algorithms
library(factoextra) # clustering algorithms & visualization
```

```
## Warning: package 'factoextra' was built under R version 3.6.2
```

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

```
library(mlr)
```

```
## Loading required package: ParamHelpers
```

```
## 'mlr' is in maintenance mode since July 2019. Future development
## efforts will go into its successor 'mlr3' (<https://mlr3.mlr-org.com>).
```

```
##
## Attaching package: 'mlr'
```

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

```
library(dendextend)
```

```
##
## ---------------------
## Welcome to dendextend version 1.13.4
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

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

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(ggforce)

# data cleaning
creditcardinfo = select(creditcard,-c(1))
creditcardinfo$CREDIT_LIMIT[is.na(creditcardinfo$CREDIT_LIMIT)] <- mean(creditcardinfo$CREDIT_LIMIT, na
creditcardinfo$MINIMUM_PAYMENTS[is.na(creditcardinfo$MINIMUM_PAYMENTS)] <- mean(creditcardinfo$MINIMUM_
sapply(creditcardinfo, class)
```

```
##                           BALANCE                BALANCE_FREQUENCY
##                         "numeric"                         "numeric"
##                         PURCHASES                 ONEOFF_PURCHASES
##                         "numeric"                         "numeric"
##            INSTALLMENTS_PURCHASES                     CASH_ADVANCE
##                         "numeric"                         "numeric"
##               PURCHASES_FREQUENCY       ONEOFF_PURCHASES_FREQUENCY
##                         "numeric"                         "numeric"
## PURCHASES_INSTALLMENTS_FREQUENCY           CASH_ADVANCE_FREQUENCY
##                         "numeric"                         "numeric"
##                  CASH_ADVANCE_TRX                     PURCHASES_TRX
##                         "integer"                         "integer"
##                      CREDIT_LIMIT                          PAYMENTS
##                         "numeric"                         "numeric"
##                  MINIMUM_PAYMENTS                 PRC_FULL_PAYMENT
##                         "numeric"                         "numeric"
##                            TENURE
##                         "integer"
```

```
#summary(creditcardinfo)

# data visualization
# Part1: Explore the distribution of data
require(reshape2)
melt.creditcardinfo <- melt(creditcardinfo)
```

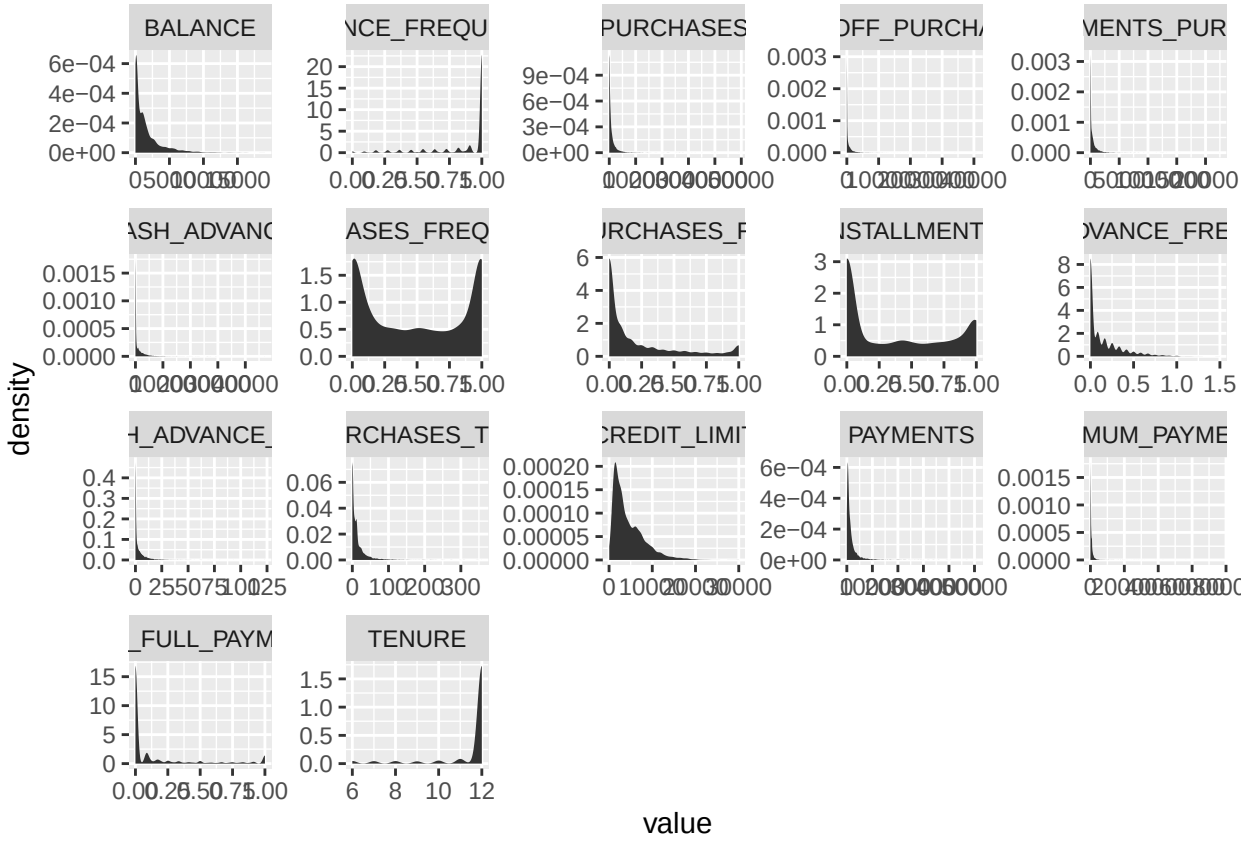
```
## No id variables; using all as measure variables
```

```
head(melt.creditcardinfo)
```

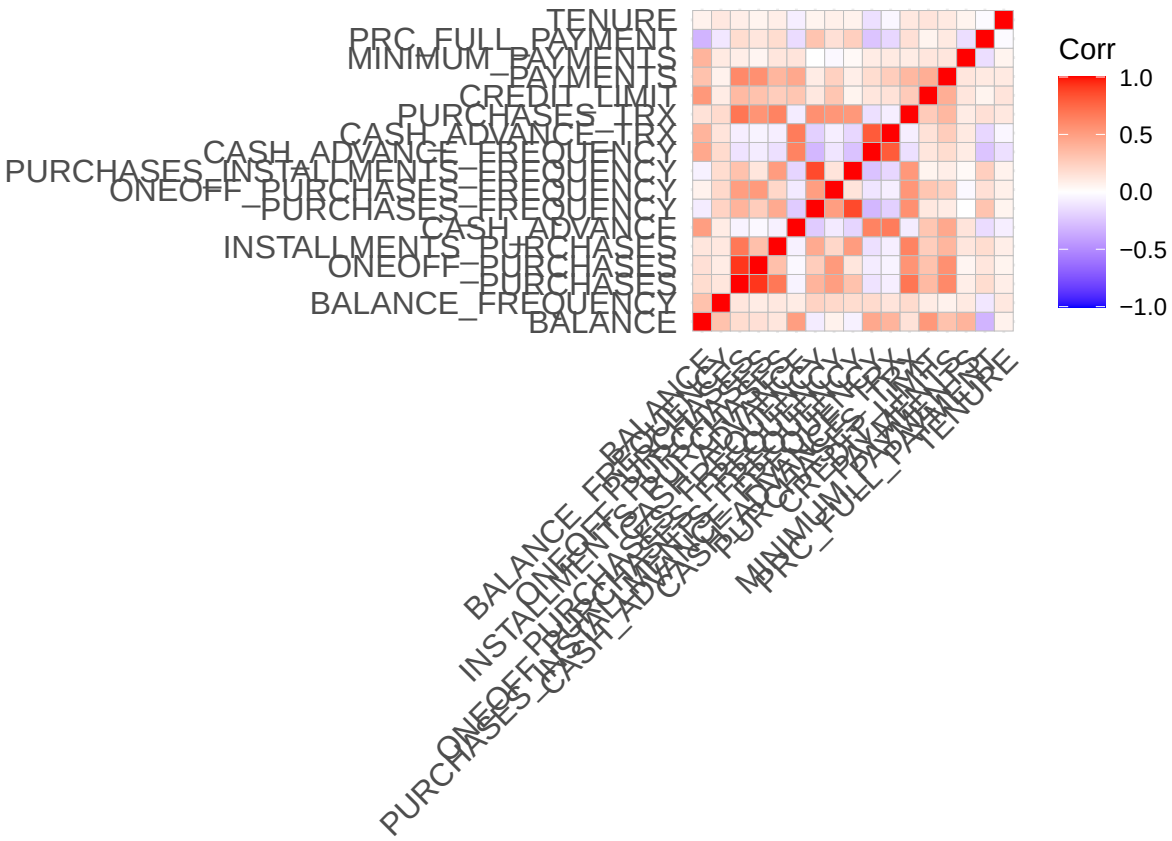
```
##   variable      value
## 1  BALANCE   40.90075
## 2  BALANCE 3202.46742
## 3  BALANCE 2495.14886
```

```
## 4  BALANCE 1666.67054
## 5  BALANCE  817.71434
## 6  BALANCE 1809.82875
```

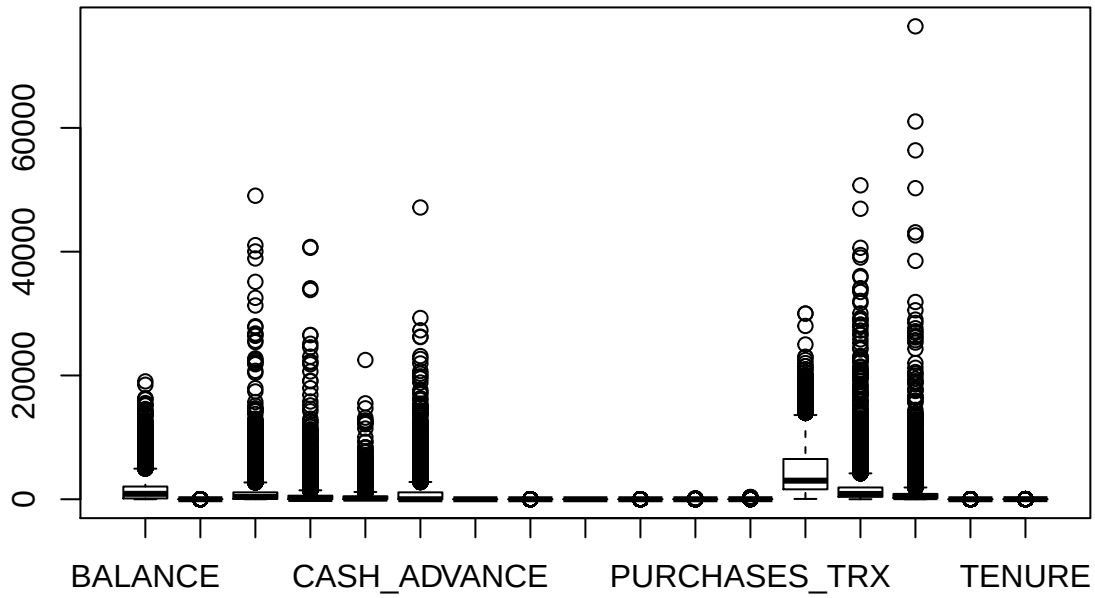
```
ggplot(data = melt.creditcardinfo, aes(x = value)) +
  stat_density() +
  facet_wrap(~variable, scales = "free")
```




```
# Part2: visualize the correlations
#cor(creditcardinfo)
ggcorrplot(cor(creditcardinfo))
```




```
# Part3: check the outliers
OutVals = boxplot(creditcardinfo)$out
```

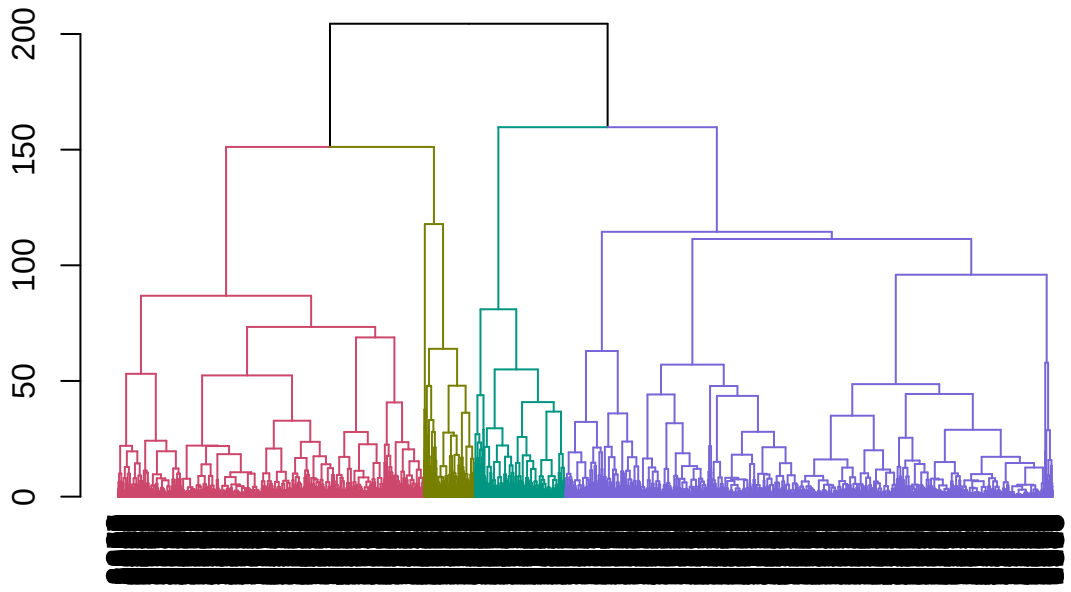



```
#which(creditcardinfo %in% OutVals)

#scale the values
credit_card_scaled <- scale(creditcardinfo)
summary(credit_card_scaled)
```

```
##     BALANCE        BALANCE_FREQUENCY    PURCHASES        ONEOFF_PURCHASES
##  Min.   :-0.7516   Min.   :-3.70306   Min.   :-0.46953   Min.   :-0.356914
##  1st Qu.:-0.6900   1st Qu.: 0.04904   1st Qu.:-0.45098   1st Qu.:-0.356914
##  Median :-0.3320   Median : 0.51806   Median :-0.30044   Median :-0.334021
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.000000
##  3rd Qu.: 0.2352   3rd Qu.: 0.51806   3rd Qu.: 0.05004   3rd Qu.:-0.009056
##  Max.   : 8.3970   Max.   : 0.51806   Max.   :22.48225   Max.   :24.199714
##  INSTALLMENTS_PURCHASES  CASH_ADVANCE      PURCHASES_FREQUENCY
##  Min.   :-0.45455       Min.   :-0.46676   Min.   :-1.22169
##  1st Qu.:-0.45455       1st Qu.:-0.46676   1st Qu.:-1.01407
##  Median :-0.35614       Median :-0.46676   Median : 0.02404
##  Mean   : 0.00000       Mean   : 0.00000   Mean   : 0.00000
##  3rd Qu.: 0.06366       3rd Qu.: 0.06435   3rd Qu.: 1.06215
##  Max.   :24.42552       Max.   :22.00989   Max.   : 1.26977
##  ONEOFF_PURCHASES_FREQUENCY PURCHASES_INSTALLMENTS_FREQUENCY
##  Min.   :-0.6786            Min.   :-0.9169
##  1st Qu.:-0.6786            1st Qu.:-0.9169
##  Median :-0.3993            Median :-0.4976
##  Mean   : 0.0000            Mean   : 0.0000
```

```
##  3rd Qu.: 0.3270            3rd Qu.: 0.9701
##  Max.   : 2.6733            Max.   : 1.5991
##  CASH_ADVANCE_FREQUENCY CASH_ADVANCE_TRX  PURCHASES_TRX       CREDIT_LIMIT
##  Min.   :-0.6753        Min.   :-0.4760   Min.   :-0.59176   Min.   :-1.2215
##  1st Qu.:-0.6753        1st Qu.:-0.4760   1st Qu.:-0.55153   1st Qu.:-0.7955
##  Median :-0.6753        Median :-0.4760   Median :-0.31016   Median :-0.4107
##  Mean   : 0.0000        Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.4351        3rd Qu.: 0.1101   3rd Qu.: 0.09213   3rd Qu.: 0.5512
##  Max.   : 6.8201        Max.   :17.5469   Max.   :13.81024   Max.   : 7.0097
##     PAYMENTS        MINIMUM_PAYMENTS  PRC_FULL_PAYMENT       TENURE
##  Min.   :-0.59866   Min.   :-0.3708   Min.   :-0.52552   Min.   :-4.1225
##  1st Qu.:-0.46626   1st Qu.:-0.2975   1st Qu.:-0.52552   1st Qu.: 0.3607
##  Median :-0.30267   Median :-0.2268   Median :-0.52552   Median : 0.3607
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.05803   3rd Qu.: 0.0000   3rd Qu.:-0.03712   3rd Qu.: 0.3607
##  Max.   :16.92133   Max.   :32.4133   Max.   : 2.89329   Max.   : 0.3607
```

```
# Hierarchical Clustering
distances = dist(credit_card_scaled,method = 'euclidean')
clusters = hclust(d = distances,method = 'ward.D2')
plot(color_branches(as.dendrogram(clusters),k = 4,groupLabels = F))
```

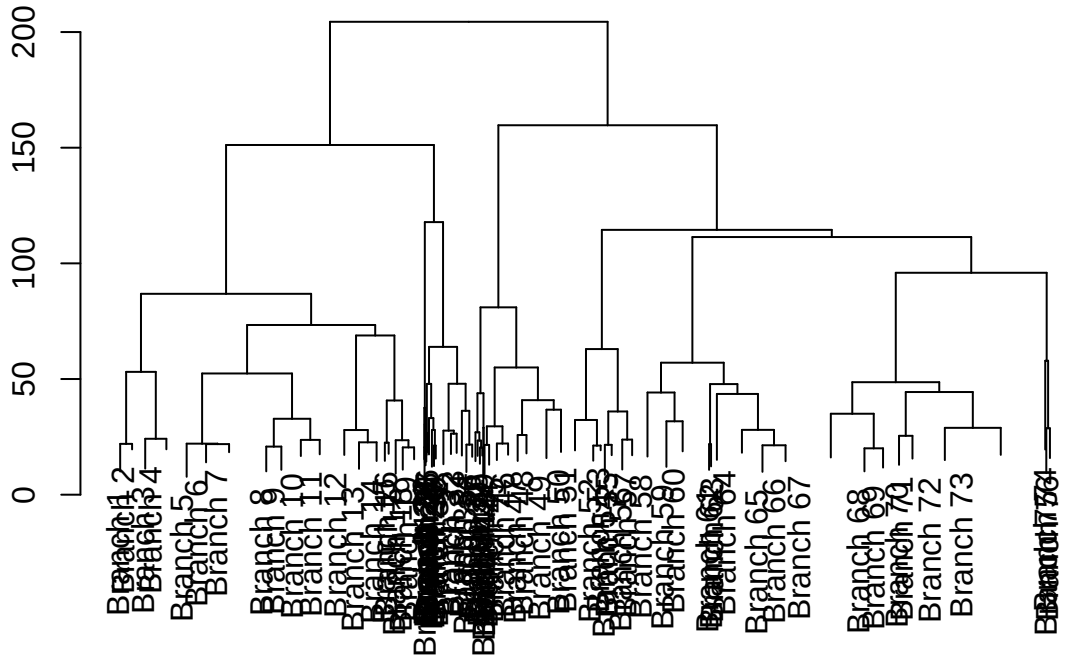



```
clusterGroups = cutree(clusters,k=4)
table(clusterGroups)
```

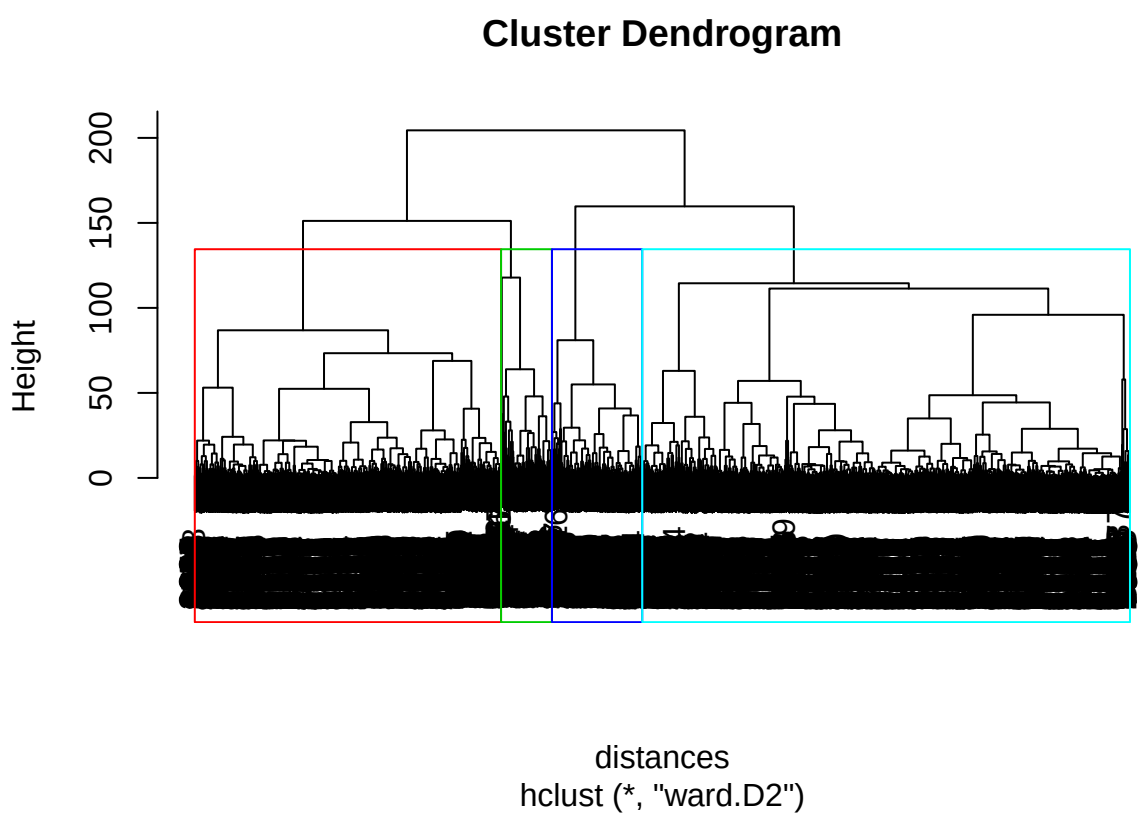
```
## clusterGroups
```

```
##    1    2    3    4
## 4668  864 2931  487
```

```
# plot only the upper part
plot(cut(as.dendrogram(clusters),h=20)$upper)
```



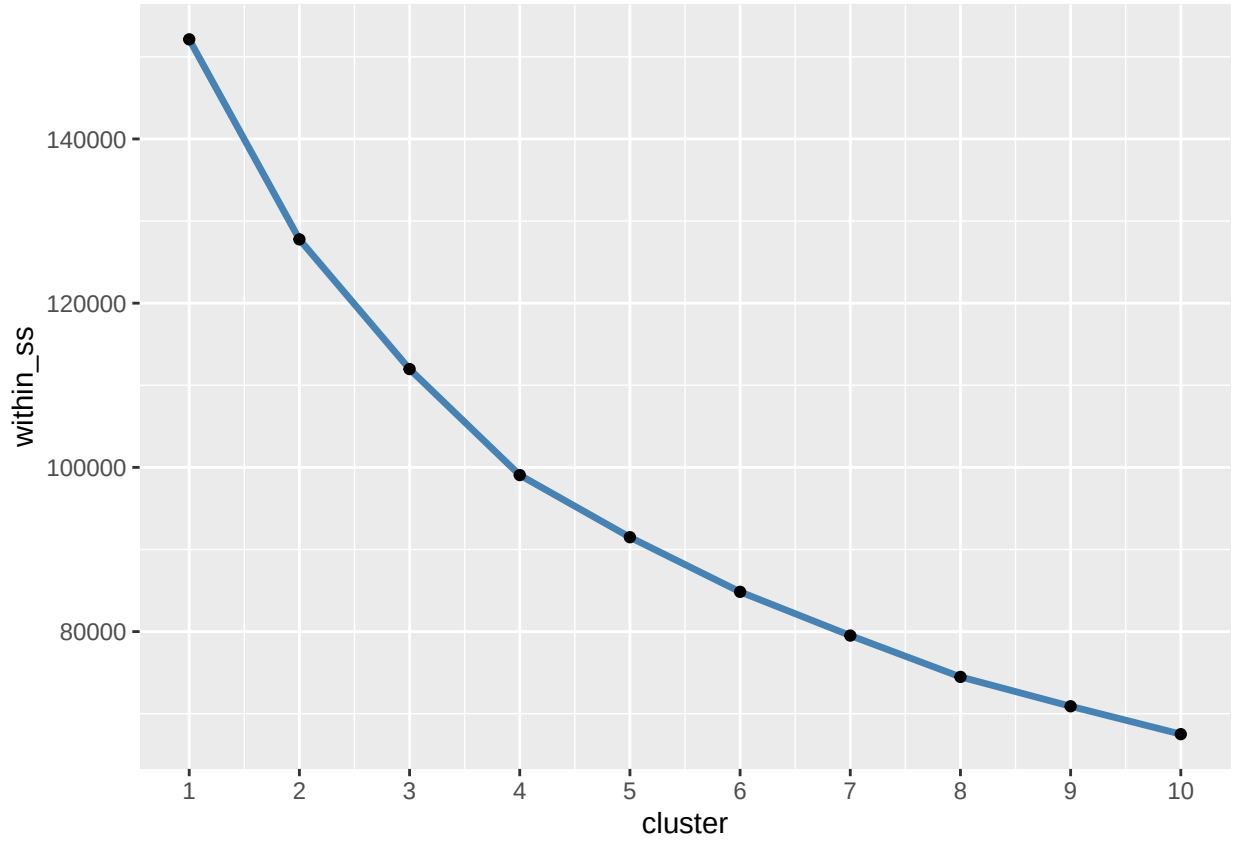

```
# label with color
plot(clusters)
rect.hclust(clusters, k = 4, border = 2:5)
```




```
# kmeans
# firstly calculate the turning point
# within error plot
within_ss = sapply(1:10,FUN = function(x){
  set.seed(617)
  kmeans(x = credit_card_scaled,centers = x,iter.max = 50,nstart = 25)$tot.withinss})
```

```
## Warning: Quick-TRANSfer stage steps exceeded maximum (= 447500)
```

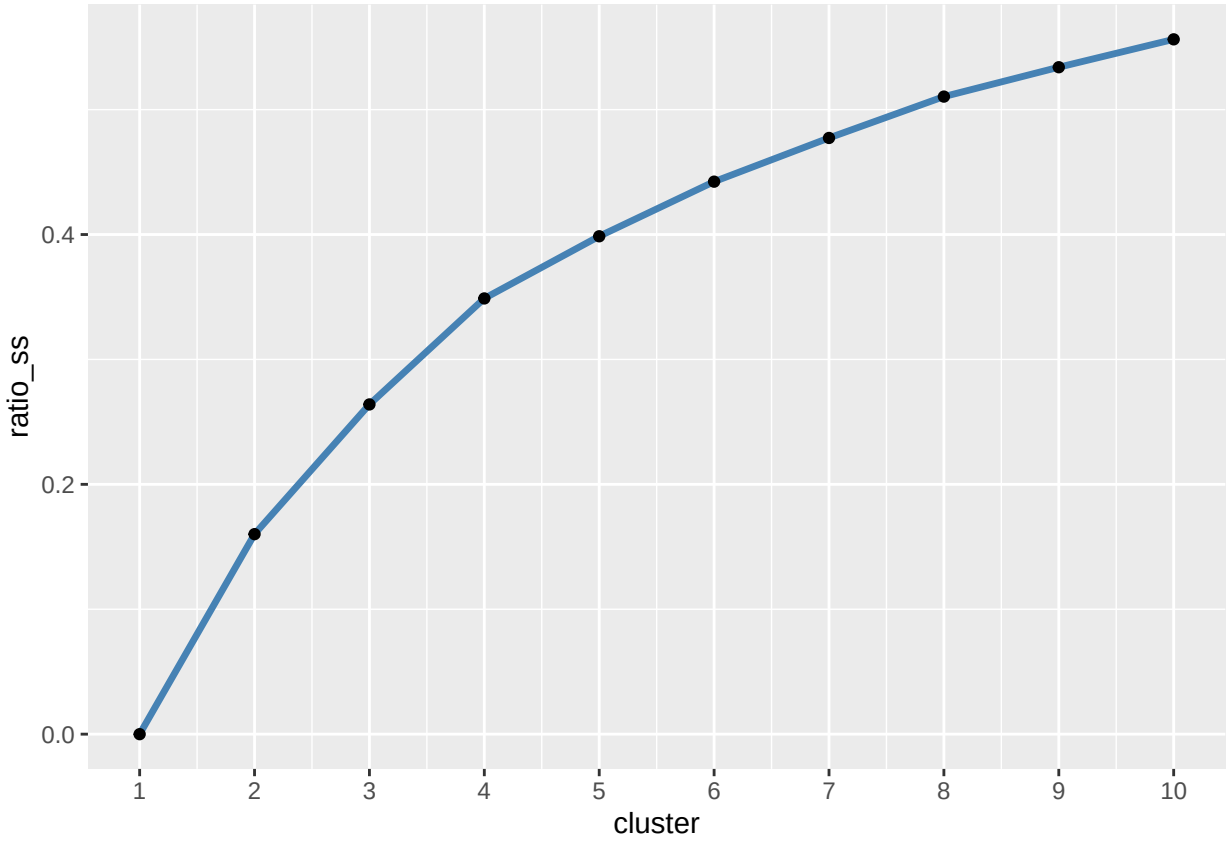
```
ggplot(data=data.frame(cluster = 1:10,within_ss),aes(x=cluster,y=within_ss))+
  geom_line(col='steelblue',size=1.2)+
  geom_point()+
  scale_x_continuous(breaks=seq(1,10,1))
```




```
# ratio plot
ratio_ss = sapply(1:10,FUN = function(x) {
  set.seed(617)
  km = kmeans(x = credit_card_scaled,centers = x,iter.max = 10000,nstart = 25)
  km$betweenss/km$totss} )
```

```
## Warning: Quick-TRANSfer stage steps exceeded maximum (= 447500)
```

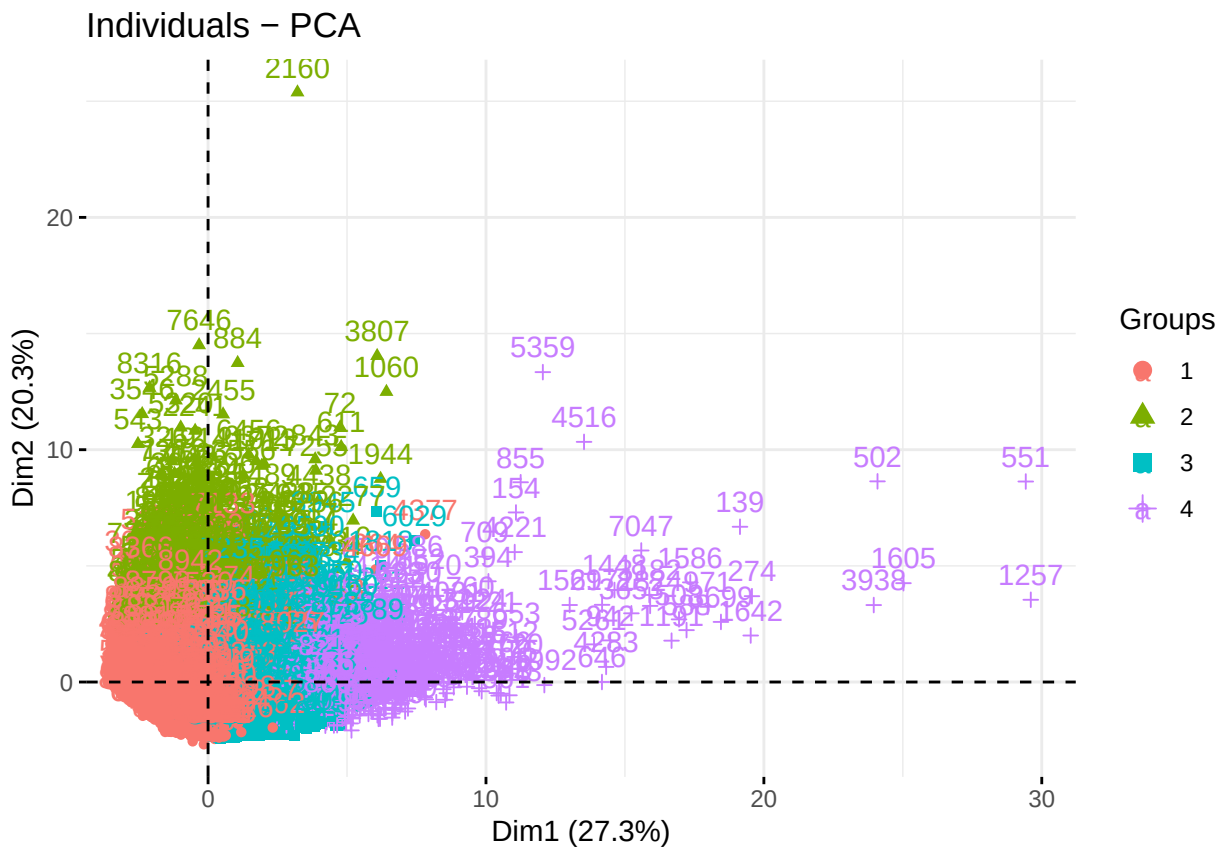
```
ggplot(data=data.frame(cluster = 1:10,ratio_ss),aes(x=cluster,y=ratio_ss))+
  geom_line(col='steelblue',size=1.2)+
  geom_point()+
  scale_x_continuous(breaks=seq(1,10,1))
```



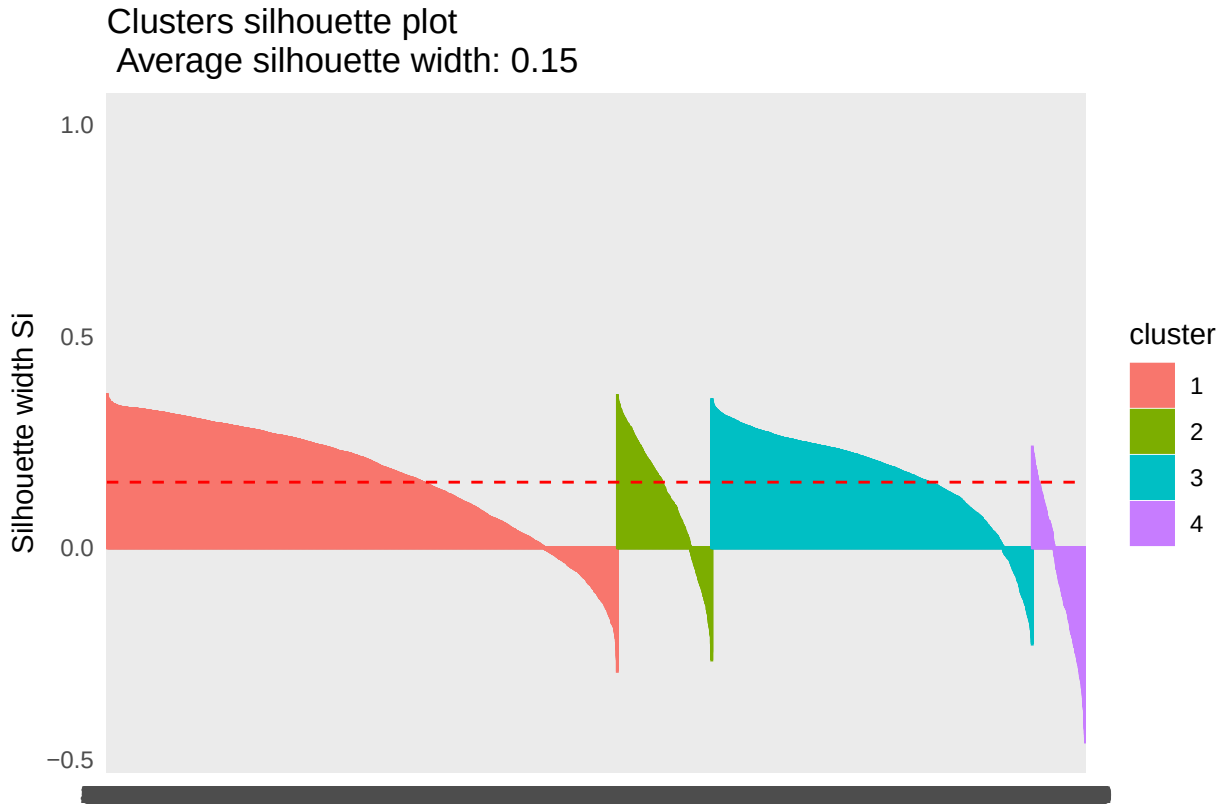

```
# silhoette
#silhoette_width = sapply(2:10,FUN = function(x) pam(x = credit_card_scaled,k = x)$silinfo$avg.width)
#ggplot(data=data.frame(cluster = 2:10,silhoette_width),aes(x=cluster,y=silhoette_width))+geom_line(col

#fviz_nbclust(credit_card_scaled, kmeans, method="wss")
#fviz_nbclust(credit_card_scaled, kmeans, method="silhouette")

km = kmeans(x = credit_card_scaled,centers = 4,iter.max=10000,nstart=25)
k_segments = table(km$cluster)
k_segments
```

```
##
##    1    2    3    4
## 1198 3364 3987  401
```

```
# plot the final graph
# 1) The PCA plots the data in two-dimensional space
# hierarchical clustering
hc_pc = prcomp(credit_card_scaled)
fviz_pca_ind(hc_pc, habillage = clusterGroups)
```

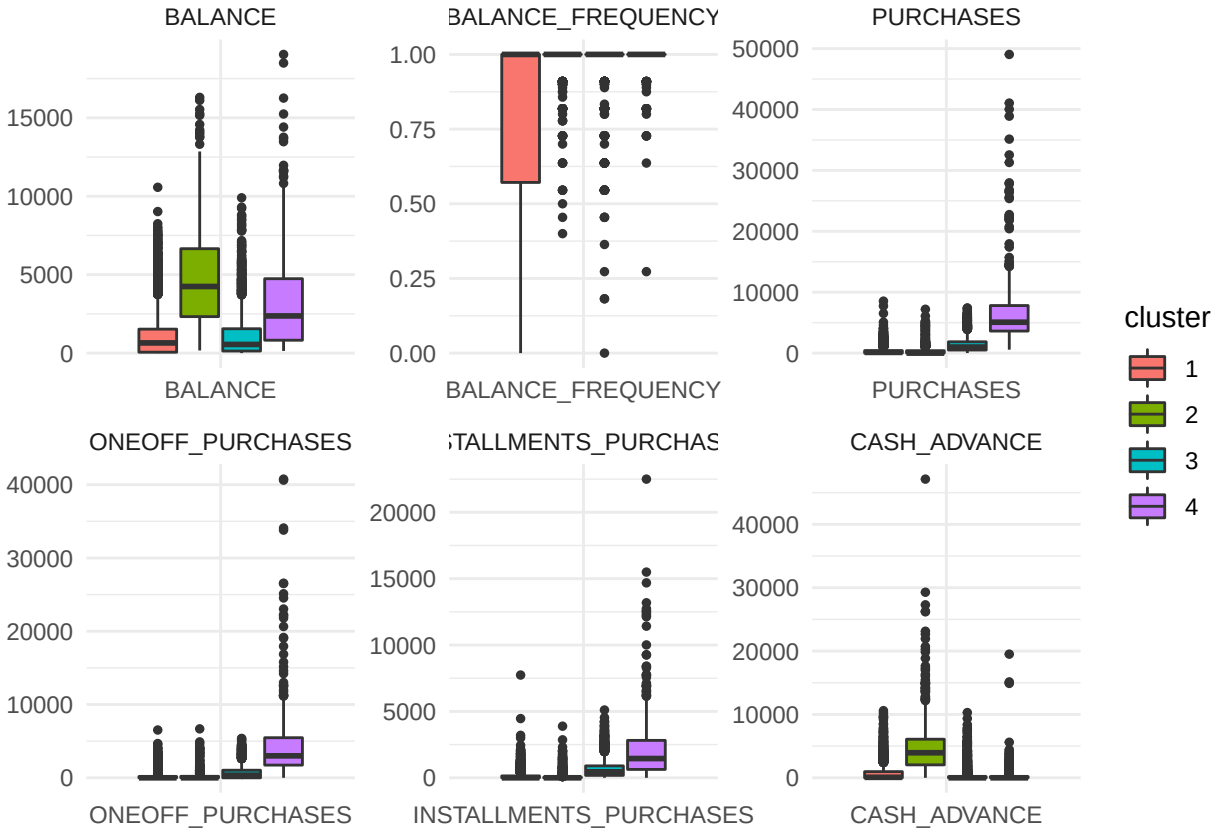



```
#k-means
#hc_pc = prcomp(credit_card_scaled)
#fviz_pca_ind(hc_pc, habillage = k_segments)

# 2) silhouette plot
# hierarchical clustering
hc_sil1 = silhouette(clusterGroups, dist(credit_card_scaled, method = "euclidean"), lable = FALSE)
fviz_silhouette(hc_sil1, print.summary = FALSE) + theme_minimal()
```

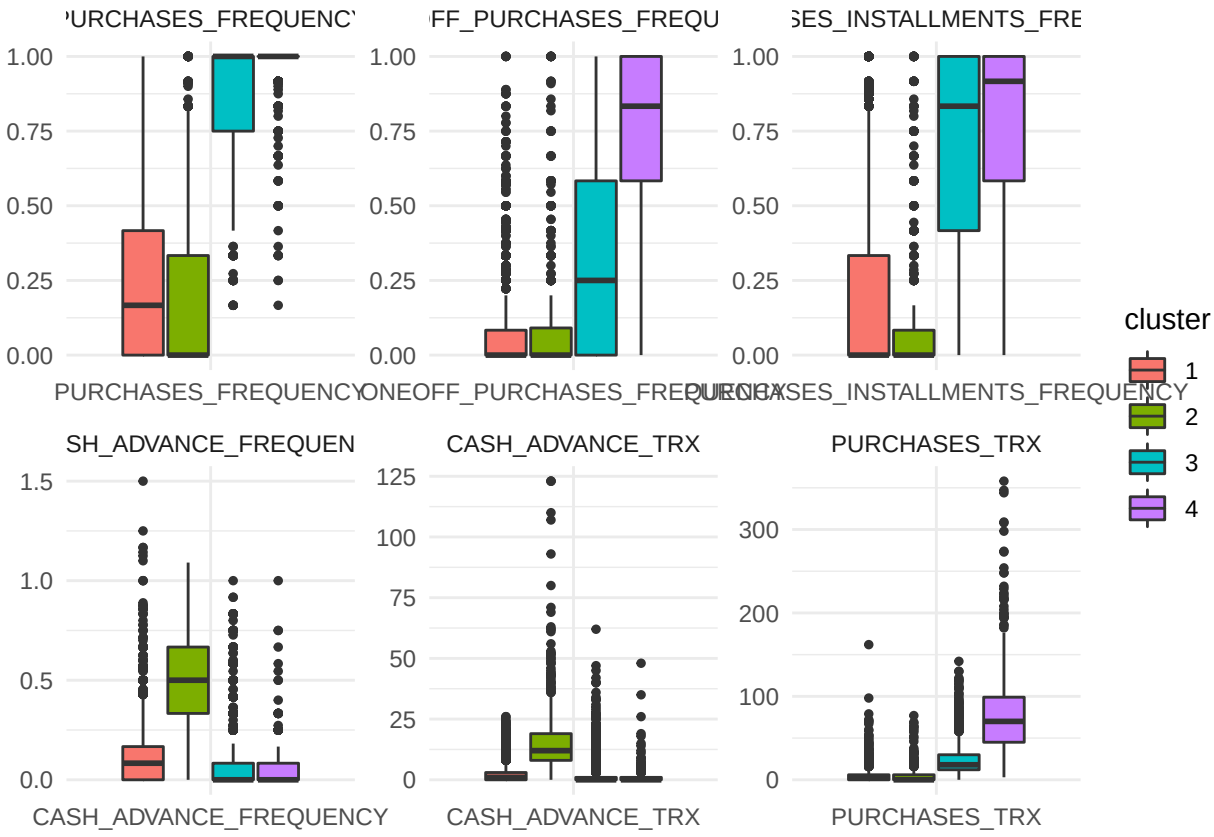



```
#k-means
#hc_sil = silhouette(k_segments, dist(credit_card_scaled, method = "euclidean"), lable = FALSE)
#fviz_silhouette(hc_sil, print.summary = FALSE) + theme_minimal()
```

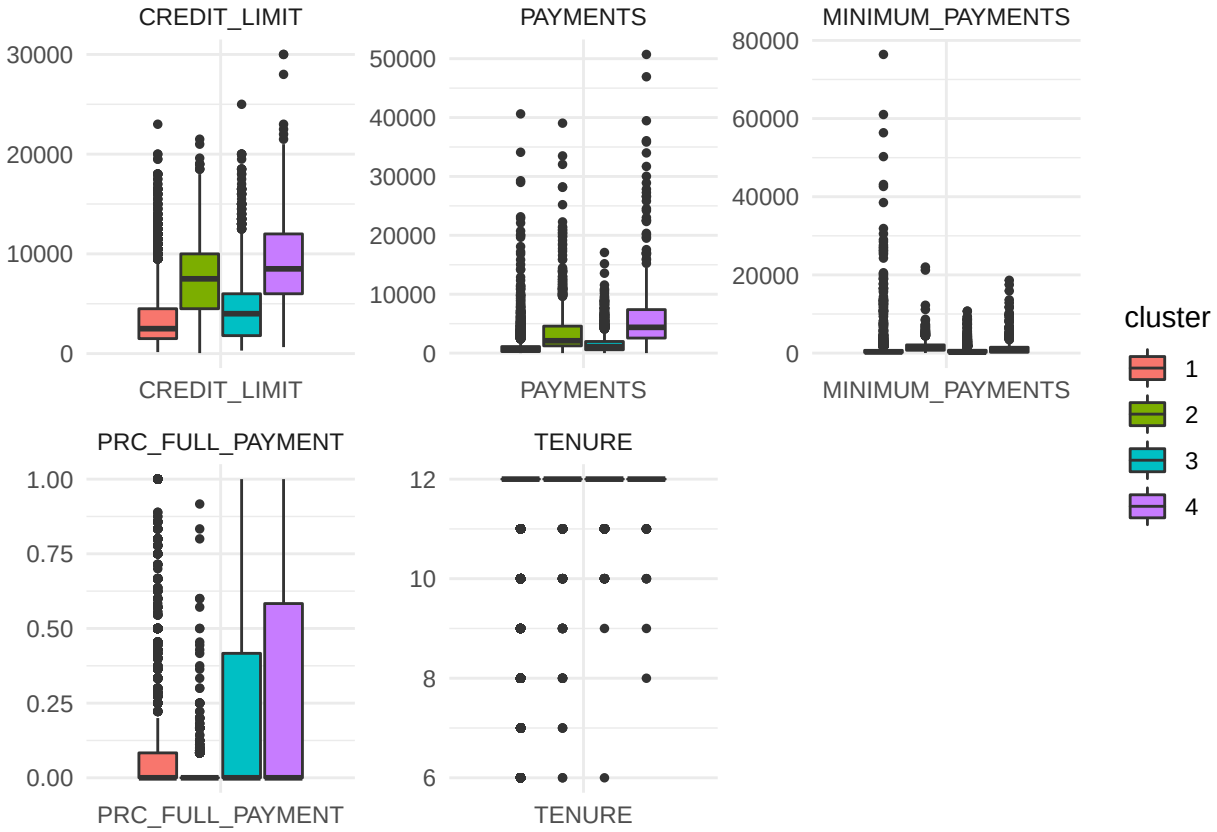
```
# customer profile
# find insights & analyze box plot
c = creditcardinfo
c$cluster = clusterGroups
c_plots = melt(c, id.var = "cluster")
c_plots$cluster = as.factor(c$cluster)

c_plots %>%
  ggplot(aes(x = variable, y = value)) +
  geom_boxplot(aes(fill = cluster), outlier.size = 1)  +
  facet_wrap_paginate( ~ variable, scales = "free", ncol = 3, nrow = 2, page = 1) +
  labs(x = NULL, y = NULL) +
  theme_minimal()
```




```
c_plots %>%
  ggplot(aes(x = variable, y = value)) +
  geom_boxplot(aes(fill = cluster), outlier.size = 1) +
  facet_wrap_paginate( ~ variable, scales = "free", ncol = 3, nrow = 2, page = 2) +
  labs(x = NULL, y = NULL) +
  theme_minimal()
```




```
c_plots %>%
  ggplot(aes(x = variable, y = value)) +
  geom_boxplot(aes(fill = cluster), outlier.size = 1) +
  facet_wrap_paginate( ~ variable, scales = "free", ncol = 3, nrow = 2, page = 3) +
  labs(x = NULL, y = NULL) +
  theme_minimal()
```


CREDIT_LIMIT
30000
20000
10000
0
CREDIT_LIMIT
PAYMENTS
50000
40000
30000
20000
10000
0
PAYMENTS
MINIMUM_PAYMENTS
80000
60000
40000
20000
0
MINIMUM_PAYMENTS
PRC_FULL_PAYMENT
1.00
0.75
0.50
0.25
0.00
PRC_FULL_PAYMENT
TENURE
12
10
8
6
TENURE
cluster
1
2
3
4