Clustering - K means, SOM

woosa7

Clustering - K means, SOM(Self-Organaizing Map)

이론설명 자료 링크 (https://github.com/woosa7/R_DataAnalytics/blob/master/R_DataMining/Lec/2016_2_DM_MBA_13.pdf)

다변량 자료를 각 특성의 유사성에 따라 여러 그룹(군집 또는 집락)으로 나누는 통계적 기법중의 하나
군집의 개수, 내용, 구조가 파악되지 않은 상태에서 특성을 파악하며, 군집들 간의 관계를 분석 (탐색적 분석)
고객의 세분화 또는 군집 별로 추가적인 분석을 수행하기 위해 활용

유사성, 거리
- 군집으로 묶기 위해서는 개체간에 유사한 특성을 가지고 있어야 함
- 이 유사한 특성의 정도를 나타내는 척도로 개체간의 거리를 사용하고, 거리가 상대적으로 가까운 개체들을 동일 군집으로 묶음

1. 계층적 군집분석(Hierarchical Clustering)
- 정확히 하나의 레코드로 구성된 군집들로 시작
- 최종적으로 모든 레코드들로 구성된 하나의 군집만이 남을 때까지 가장 가까운 두 군집들을 점진적으로 병합

2. 비계층적 군집분석 : K-평균 군집분석(K-Means algorithm)
- 단계0 : 사전에 군집의 수 K를 지정한다.
- 단계1 : 각 군집에 1개의 군집중심을 임의로 정한다.(보통 서로 상당히 떨어진 개체를 선택함)
- 단계2 : 모든 개체를 각각 가장 가까운 군집중심에 배속시킨다.
- 단계3 : 각 군집의 중심을 산출한다.
- 단계4 : 단계2와 단계3을 변화가 거의 없을 때까지(보통 10회 이하) 반복한다.

3. Kohonen SOM 군집분석

SOM(Self-Organizing Map)은 "자기조직화 지도"라는 하며, 관측 개체들을 스스로 조직화하여 지도의 형태로 뿌려주는 신경망 기법이다.
SOM 개념도는 2개의 층으로 이루어져 있으며, 첫 번째 층이 입력층(input layer), 두 번째 층은 출력층(output layer)으로 이루어진 2차원 격자(grid)로 되어 있다.

```
library(cluster)
library(NbClust)
library(kohonen)
library(ggplot2)
library(gridExtra)
library(scales)
```

```
# Read Data
cdata <- read.delim("data/Cluster.txt", stringsAsFactors=FALSE)
head(cdata)
```

```
##   ID  MONEY VISIT CROSS API
## 1  1 367900    15     3  14
## 2  2  64000     3     1 109
## 3  3 467400    10     8  12
## 4  4  61000     3     1  70
## 5  5 128000     4     2  45
## 6  6 353620     6     5  22
```

```
##### K-means Clustering with R

# 군집수를 4로 하는 k-means clustering
km <- kmeans(subset(cdata, select=-c(ID)), centers=4)
str(km)
```

```
## List of 9
##  $ cluster     : Named int [1:1000] 3 3 3 3 3 3 1 3 3 3 ...
##   ..- attr(*, "names")= chr [1:1000] "1" "2" "3" "4" ...
##  $ centers     : num [1:4, 1:4] 1.28e+06 3.42e+06 2.55e+05 1.13e+07 3.26e+01 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:4] "1" "2" "3" "4"
##   .. ..$ : chr [1:4] "MONEY" "VISIT" "CROSS" "API"
##  $ totss       : num 1.65e+15
##  $ withinss    : num [1:4] 2.80e+13 3.39e+13 2.77e+13 1.42e+14
##  $ tot.withinss: num 2.32e+14
##  $ betweenss   : num 1.42e+15
##  $ size        : int [1:4] 181 41 770 8
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

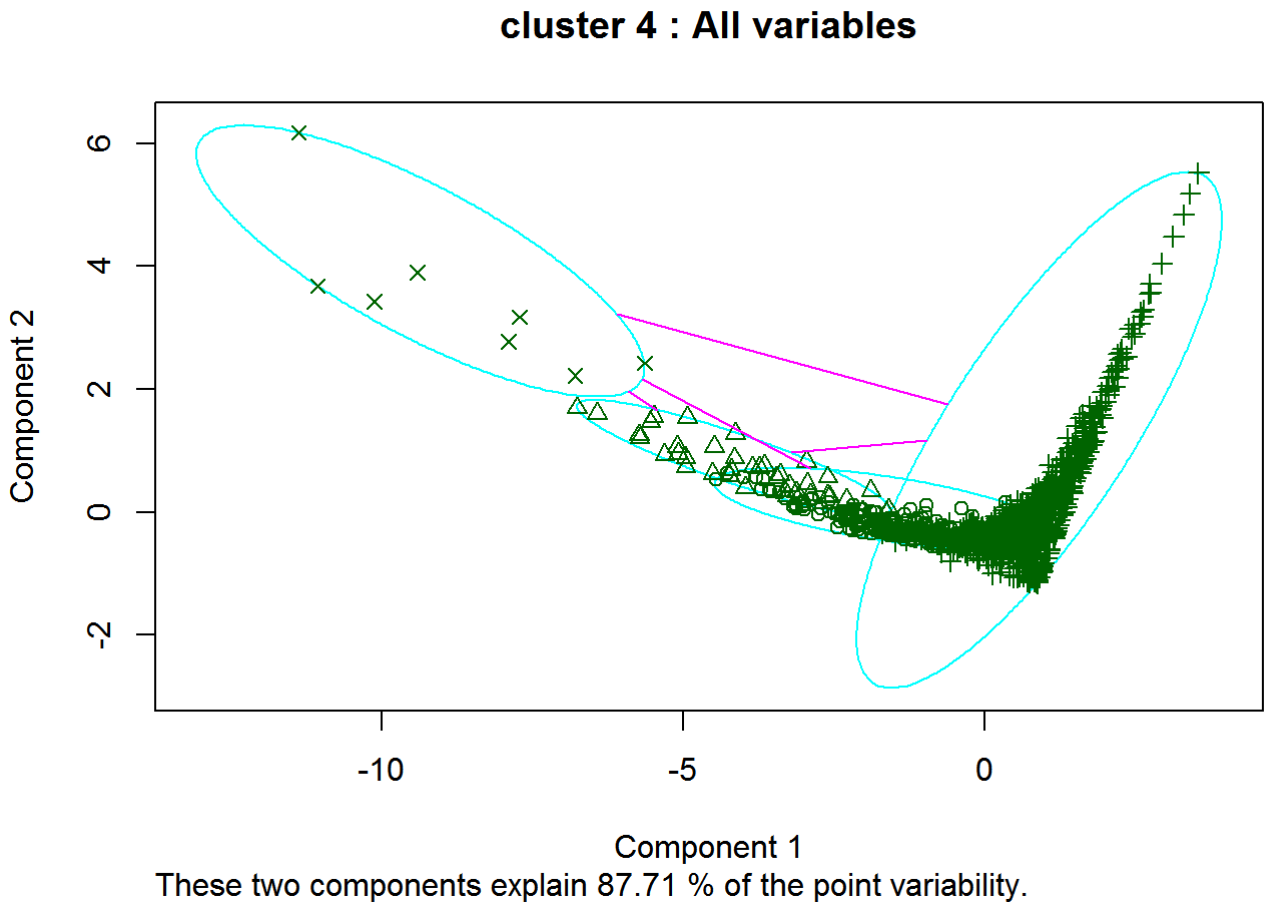
```
km
```

```
## K-means clustering with 4 clusters of sizes 181, 41, 770, 8
##
## Cluster means:
##       MONEY     VISIT    CROSS      API
## 1  1284818.8  32.607735 10.966851  7.232044
## 2  3421840.0  62.146341 15.390244  3.024390
## 3   255051.4   8.124675  3.654545 34.877922
## 4 11323243.8 101.125000 20.750000  0.750000
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##   3   3   3   3   3   3   1   3   3   3   1   3   1   3   1
##  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30
##   3   3   3   3   1   3   3   1   1   3   3   1   3   1   1
##  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45
##   3   3   4   2   1   3   3   3   3   3   3   3   3   3   3
##  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   3   3   3   1   3   3   3   1   1   1   3   3   3   3   3
   .........
## 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945
##   3   3   3   3   1   3   3   3   3   3   3   3   3   3   3
## 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960
##   3   3   2   3   1   1   3   3   3   1   3   3   3   2   1
## 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975
##   3   3   3   3   3   3   3   3   3   1   3   3   3   1   3
## 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990
##   3   3   3   3   2   3   3   3   3   3   3   2   3   3   3
## 991 992 993 994 995 996 997 998 999 1000
##   3   3   3   3   3   3   3   3   3   3
##
## Within cluster sum of squares by cluster:
## [1] 2.803588e+13 3.391002e+13 2.770304e+13 1.424076e+14
##  (between_SS / total_SS =  85.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

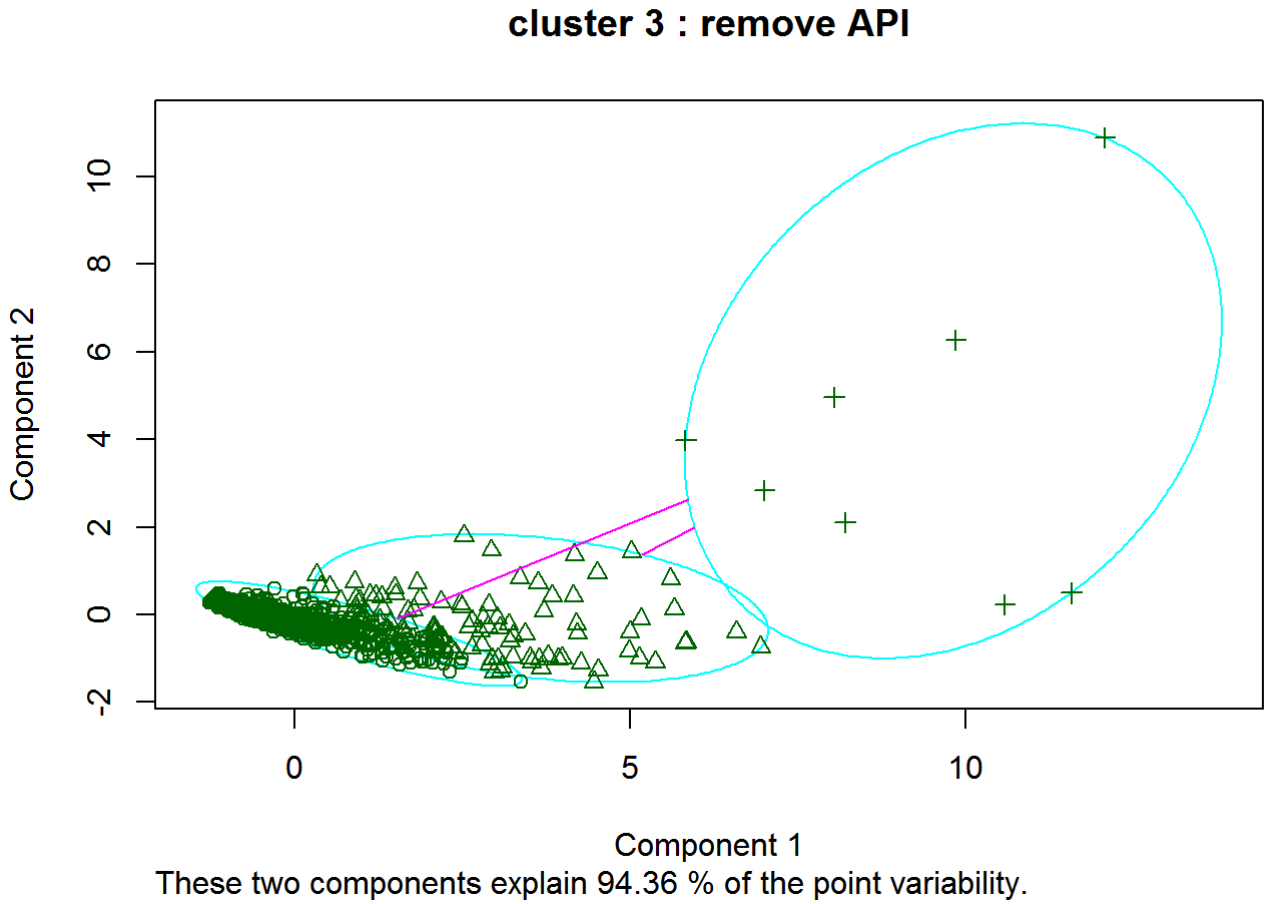
```
# API 변수를 사용하지 않고, 군집수를 3개로 설정.
km3 <- kmeans(subset(cdata, select=-c(ID, API)), centers=3)
km3
```

```
## K-means clustering with 3 clusters of sizes 887, 105, 8
##
## Cluster means:
##       MONEY    VISIT    CROSS
## 1  357964.3  10.88388  4.507328
## 2  2397360.6  48.11429 13.638095
## 3 11323243.8 101.12500 20.750000
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##   1   1   1   1   1   1   2   1   1   1   1   1   1   1   1
##  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30
##   1   1   1   1   2   1   1   1   2   1   1   1   1   2   2
##  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45
##   1   1   3   2   1   1   1   1   1   1   1   1   1   1   1
##  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   1   1   1   1   1   1   1   1   2   1   1   1   1   1   1
  .........
## 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960
##   1   1   2   1   1   2   1   1   1   1   1   1   1   2   1
## 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975
##   1   1   1   1   1   1   1   1   1   2   1   1   1   1   1
## 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990
##   1   1   1   1   2   1   1   1   1   1   1   2   1   1   1
## 991 992 993 994 995 996 997 998 999 1000
##   1   1   1   1   1   1   1   1   1   1
##
## Within cluster sum of squares by cluster:
## [1] 9.262028e+13 1.088448e+14 1.424076e+14
##  (between_SS / total_SS =  79.2 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

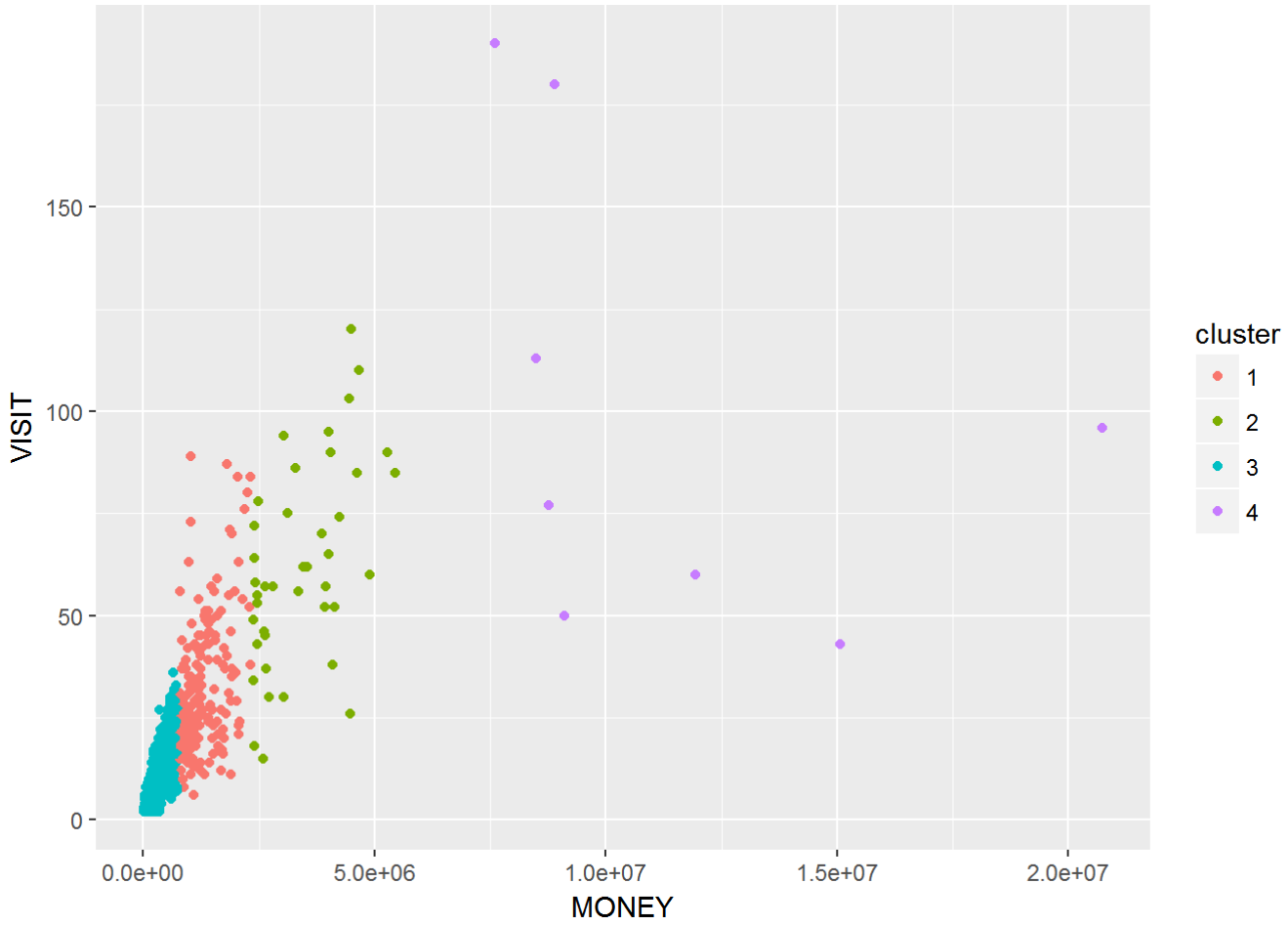
```
# 군집의 반경과 군집간 관계
clusplot(subset(cdata, select=-c(ID)), km$cluster, main="cluster 4 : All variables")
```



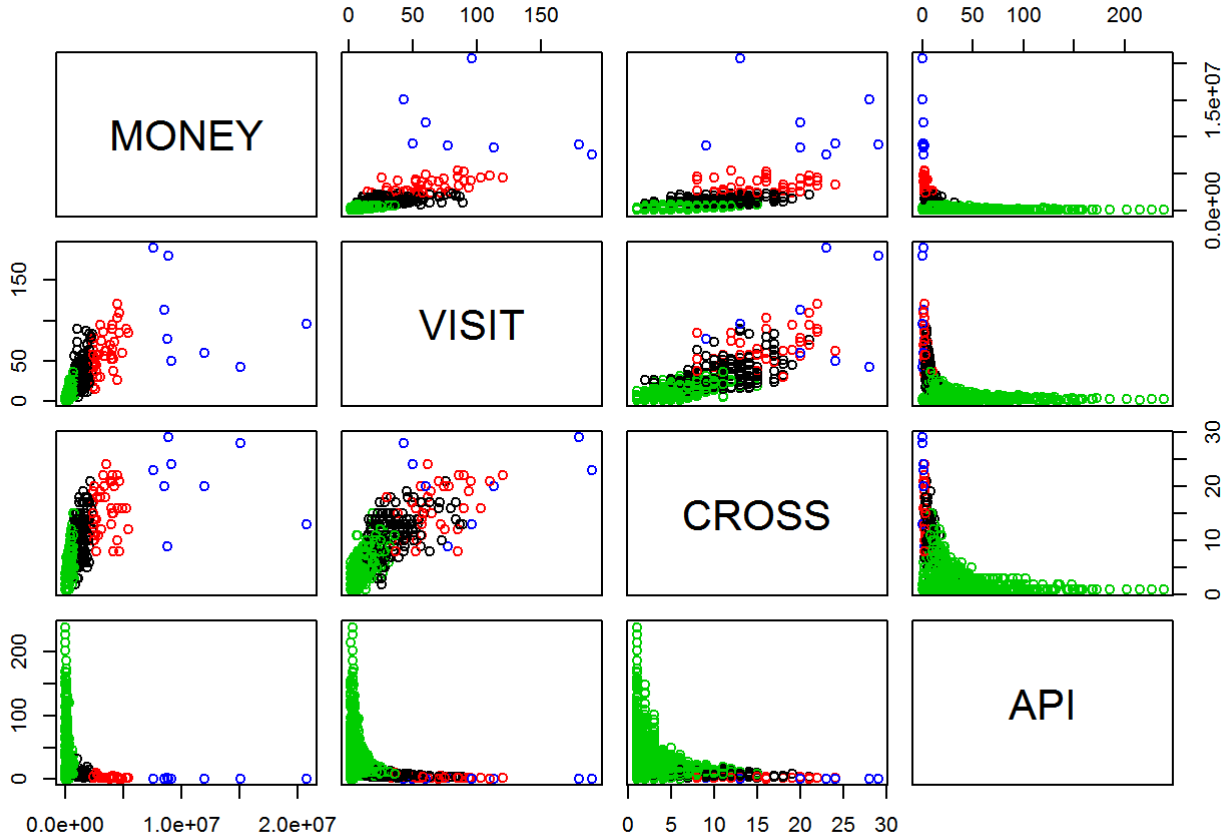

```
clusplot(subset(cdata, select=-c(ID, API)), km3$cluster, main="cluster 3 : remove API")
```



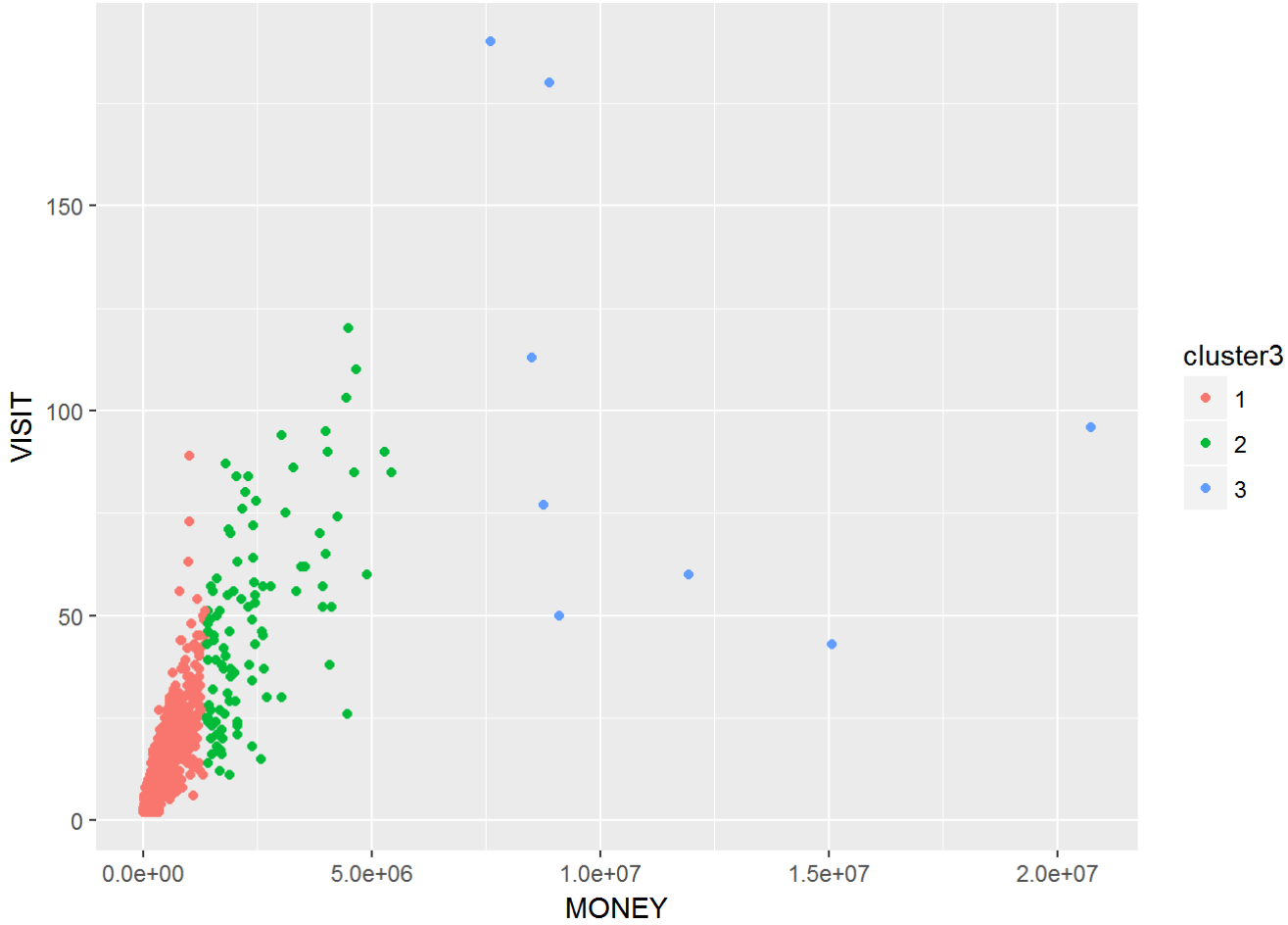
```
# 군집의 분포
cdata$cluster <- as.factor(km$cluster)
qplot(MONEY, VISIT, colour=cluster, data=cdata)
```



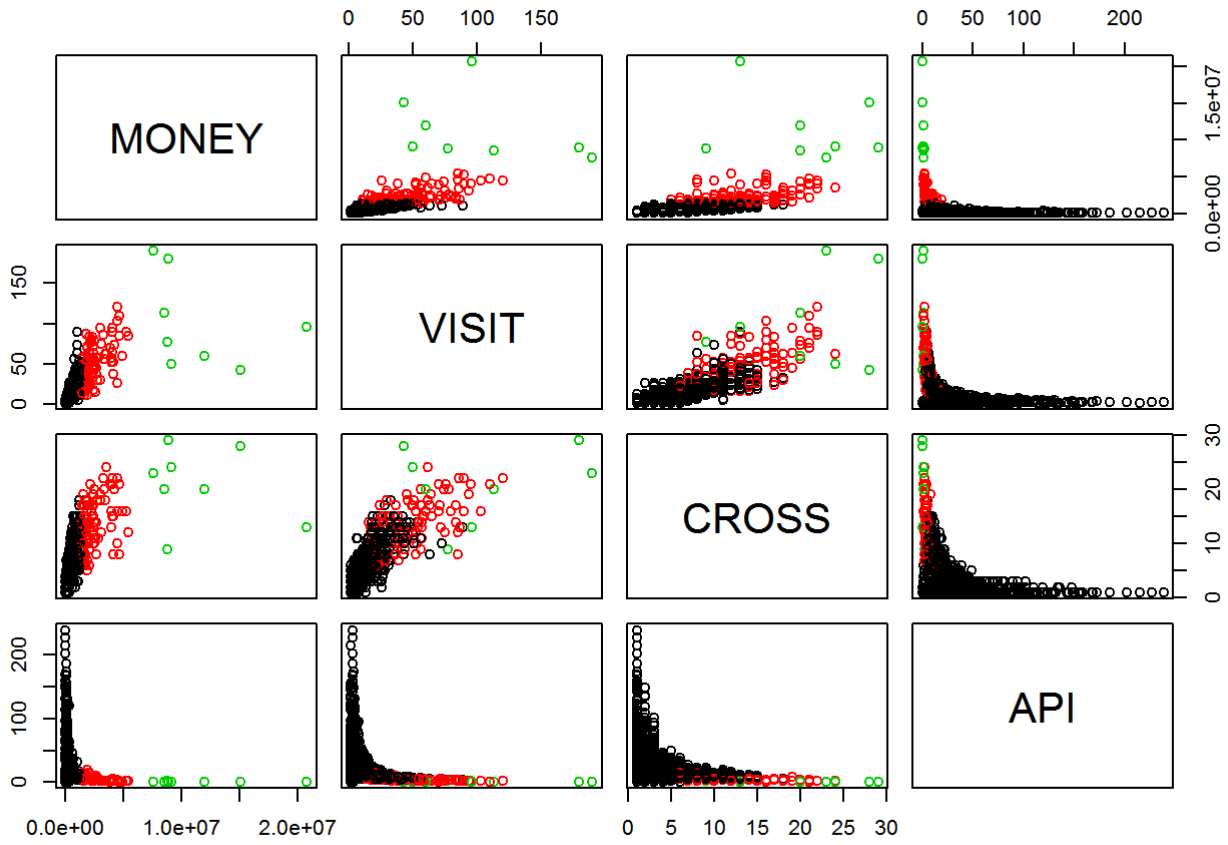

```
plot(subset(cdata, select=-c(ID,cluster)), col=km$cluster)
```



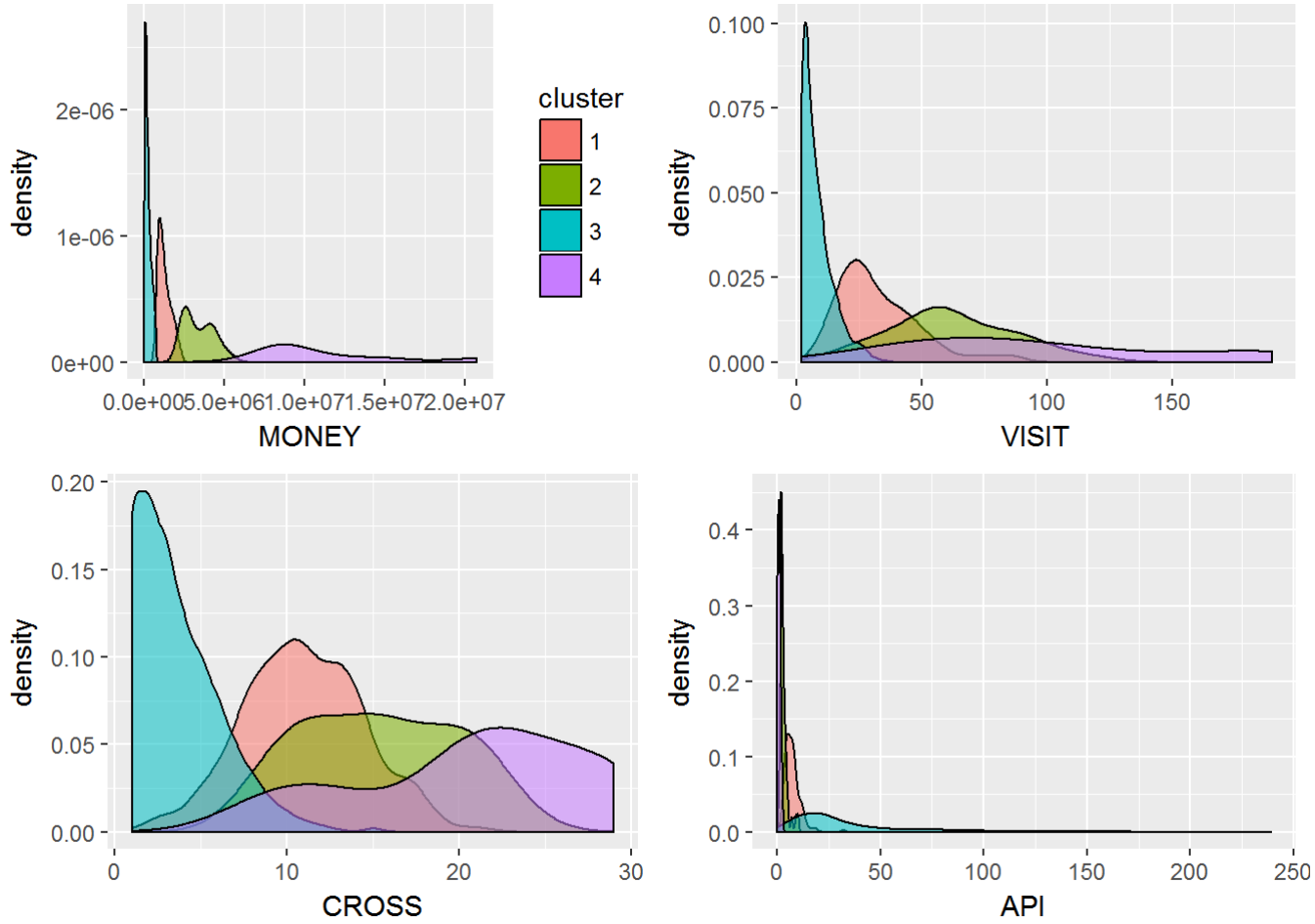

```
cdata$cluster3 <- as.factor(km3$cluster)
qplot(MONEY, VISIT, colour=cluster3, data=cdata)
```



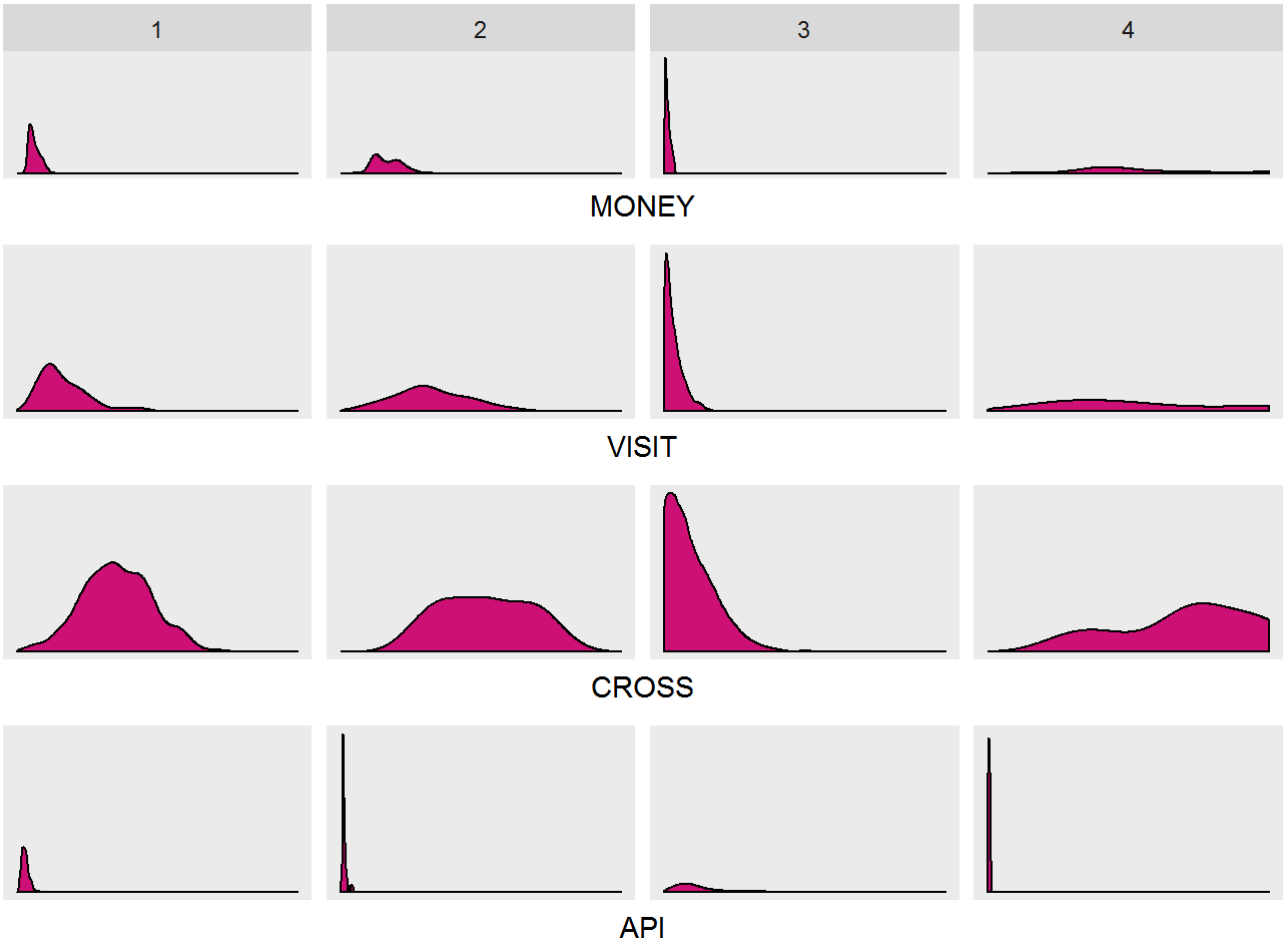
```
plot(subset(cdata, select=-c(ID,cluster,cluster3)), col=km3$cluster)
```



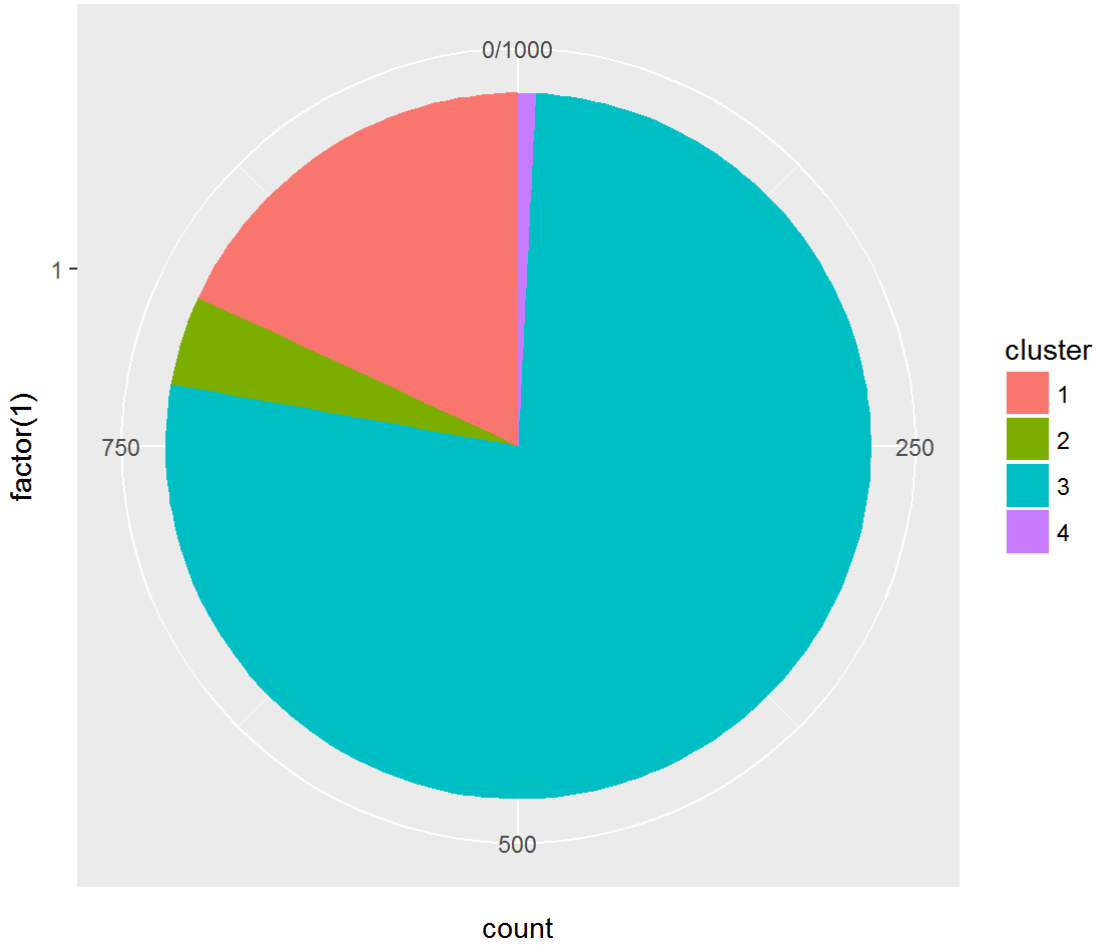

```
# 군집별 군집화변수의 밀도 : 방법1
p1 <- qplot(MONEY, fill=cluster, alpha=.5, data=cdata, geom="density") + scale_alpha(guide="none")
p2 <- qplot(VISIT, fill=cluster, alpha=.5, data=cdata, geom="density") + theme(legend.position="none")
p3 <- qplot(CROSS, fill=cluster, alpha=.5, data=cdata, geom="density") + theme(legend.position="none")
p4 <- qplot(API, fill=cluster, alpha=.5, data=cdata, geom="density") + theme(legend.position="none")
grid.arrange(p1, p2, p3, p4, ncol=2, nrow=2)
```



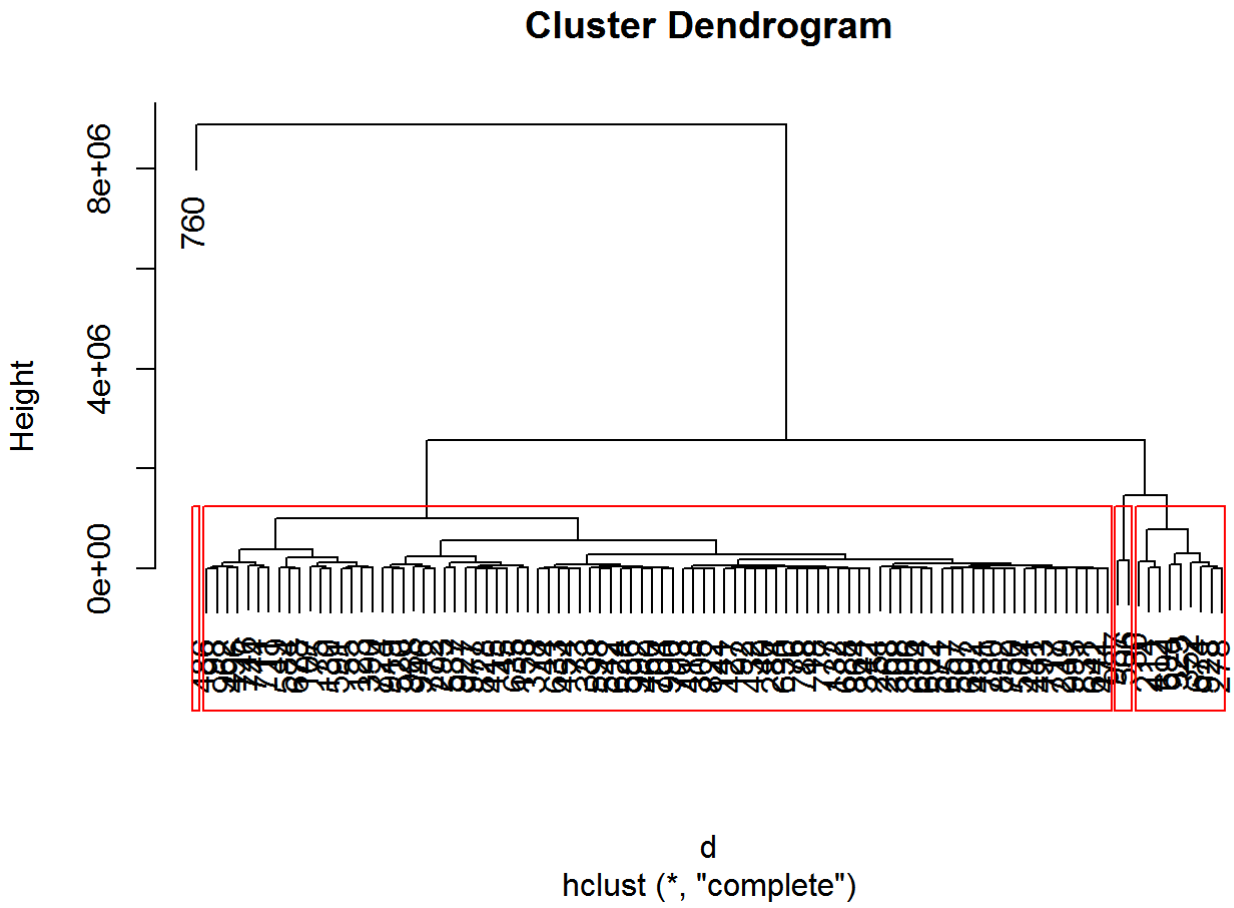

```
# 군집별 군집화변수의 밀도 : 방법2
p1 <- ggplot(cdata, aes(MONEY)) + geom_density(fill='deeppink3', adjust=1) + facet_grid(. ~ cluster) + scale_x_continuous(breaks=NULL) + scale_y_continuous("", breaks=NULL)
p2 <- ggplot(cdata, aes(VISIT)) + geom_density(fill='deeppink3', adjust=1) + facet_grid(. ~ cluster) + scale_x_continuous(breaks=NULL) + scale_y_continuous("", breaks=NULL) + theme(strip.text.x=element_blank())
p3 <- ggplot(cdata, aes(CROSS)) + geom_density(fill='deeppink3', adjust=1) + facet_grid(. ~ cluster) + scale_x_continuous(breaks=NULL) + scale_y_continuous("", breaks=NULL) + theme(strip.text.x=element_blank())
p4 <- ggplot(cdata, aes(API)) + geom_density(fill='deeppink3', adjust=1) + facet_grid(. ~ cluster) + scale_x_continuous(breaks=NULL) + scale_y_continuous("", breaks=NULL) + theme(strip.text.x=element_blank())
grid.arrange(p1, p2, p3, p4, ncol=1, nrow=4)
```



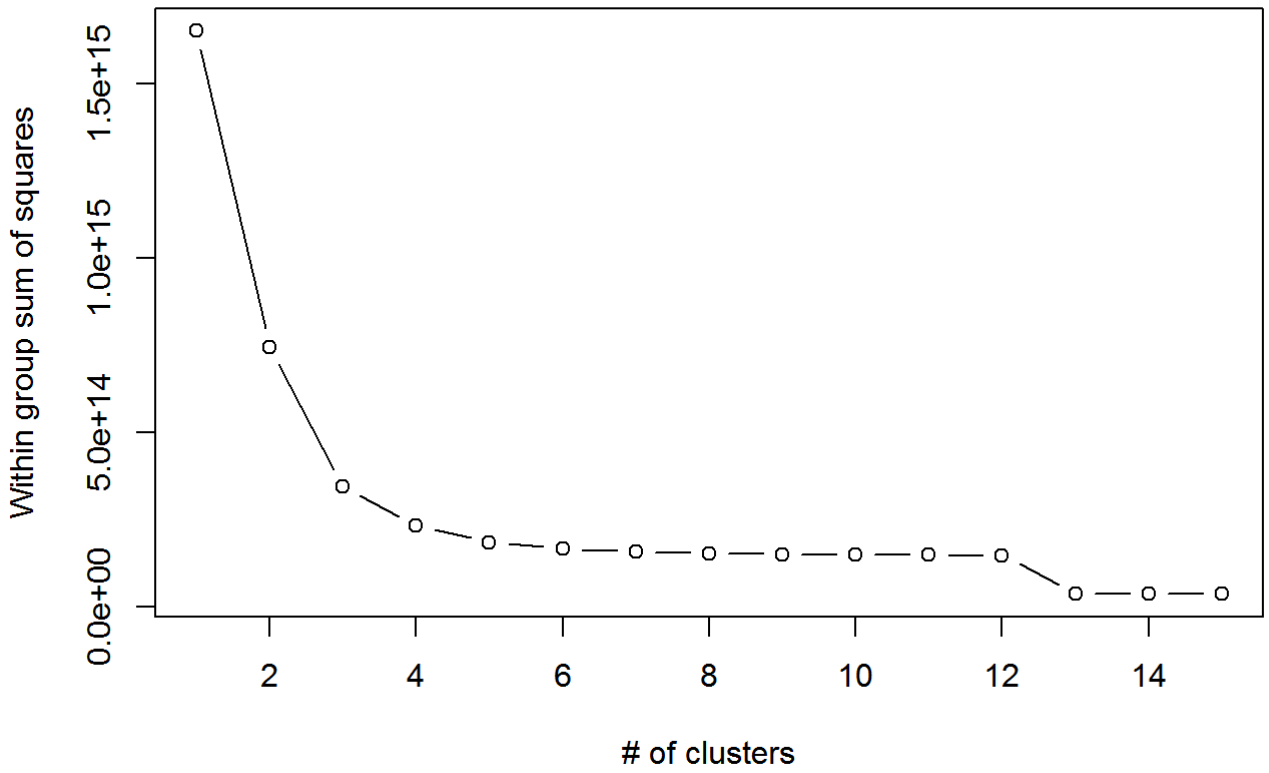

```
# 군집의 크기
x <- ggplot(cdata, aes(x=factor(1), fill=cluster))
x + geom_bar(width=1) + coord_polar(theta="y")
```




```
# 최적의 군집 수 찾기 : 방법1
sd <- cdata[sample(1:nrow(cdata),100),-1]
d <- dist(sd, method = "euclidean")
fit <- hclust(d, method="complete")
plot(fit)
rect.hclust(fit, k=4, border = "red")
```




```
# 최적의 군집 수 찾기 : 방법2
wss <- 0; set.seed(1)
for(i in 1:15) wss[i] <- kmeans(subset(cdata, select=-c(ID,cluster,cluster3)), centers=i)$tot.withinss
plot(1:15, wss, type="b", xlab="# of clusters", ylab="Within group sum of squares")
```



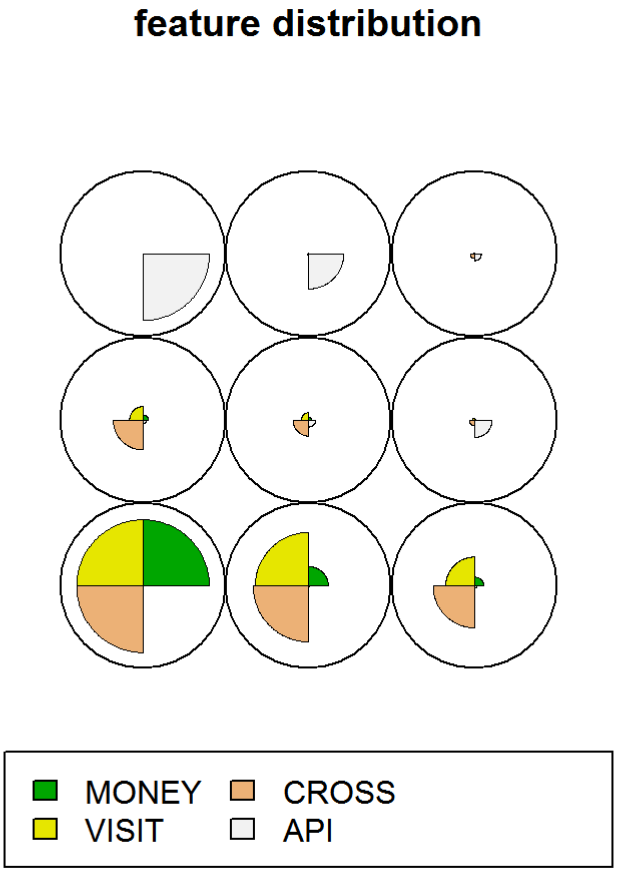

```
# 최적의 군집 수 찾기: 방법3. 많은 시간 소요됨.
# nc = NbClust(subset(cdata, select=-c(ID,cluster,cluster3)), min.nc=2, max.nc=15, method='kmeans')
# barplot(table(nc$Best.nc[1,]), xlab="# of clusters", ylab="# of criteria", main="Number of clusters chosen by 26 criteria")
```

```
##########################################
SOM Clustering with R
##########################################
```

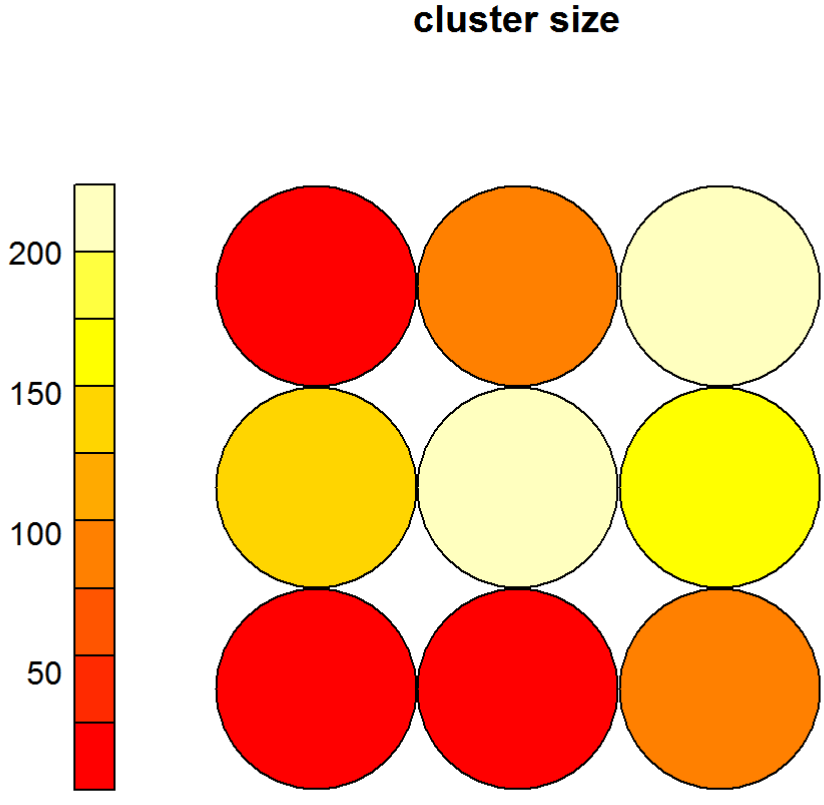
```
cdata <- read.delim("data/Cluster.txt", stringsAsFactors=FALSE)

# 데이터 정규화
cdata.n <- scale(subset(cdata, select=-c(ID)))

# SOM clustering 을 3 x 3 으로 설정.
set.seed(1)
som_model <- som(data = cdata.n, grid = somgrid(3, 3, "rectangular"))
str(som_model)    # codes : 각 노드의 weight
```

```
## List of 10
##  $ data       : num [1:1000, 1:4] -0.227 -0.463 -0.15 -0.466 -0.414 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:1000] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:4] "MONEY" "VISIT" "CROSS" "API"
##   ..- attr(*, "scaled:center")= Named num [1:4] 659823.1 15.5 5.6 28.3
##   .. ..- attr(*, "names")= chr [1:4] "MONEY" "VISIT" "CROSS" "API"
##   ..- attr(*, "scaled:scale")= Named num [1:4] 1.29e+06 1.92e+01 4.67 3.22e+01
##   .. ..- attr(*, "names")= chr [1:4] "MONEY" "VISIT" "CROSS" "API"
##  $ grid       :List of 5
##   ..$ pts   : int [1:9, 1:2] 1 2 3 1 2 3 1 2 3 1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : NULL
##   .. .. ..$ : chr [1:2] "x" "y"
##   ..$ xdim  : num 3
##   ..$ ydim  : num 3
##   ..$ topo  : chr "rectangular"
##   ..$ n.hood: chr "square"
##   ..- attr(*, "class")= chr "somgrid"
##  $ codes      : num [1:9, 1:4] 8.546 2.183 0.775 0.269 -0.111 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:4] "MONEY" "VISIT" "CROSS" "API"
##  $ changes    : num [1:100, 1] 0.0202 0.021 0.0212 0.0224 0.0221 ...
##  $ alpha      : num [1:2] 0.05 0.01
##  $ radius     : num [1:2] 2 -2
##  $ toroidal   : logi FALSE
##  $ unit.classif: int [1:1000] 9 8 5 8 6 5 4 5 9 4 ...
##  $ distances  : num [1:1000] 0.309 0.752 0.273 0.146 0.055 ...
##  $ method     : chr "som"
##  - attr(*, "class")= chr "kohonen"
```

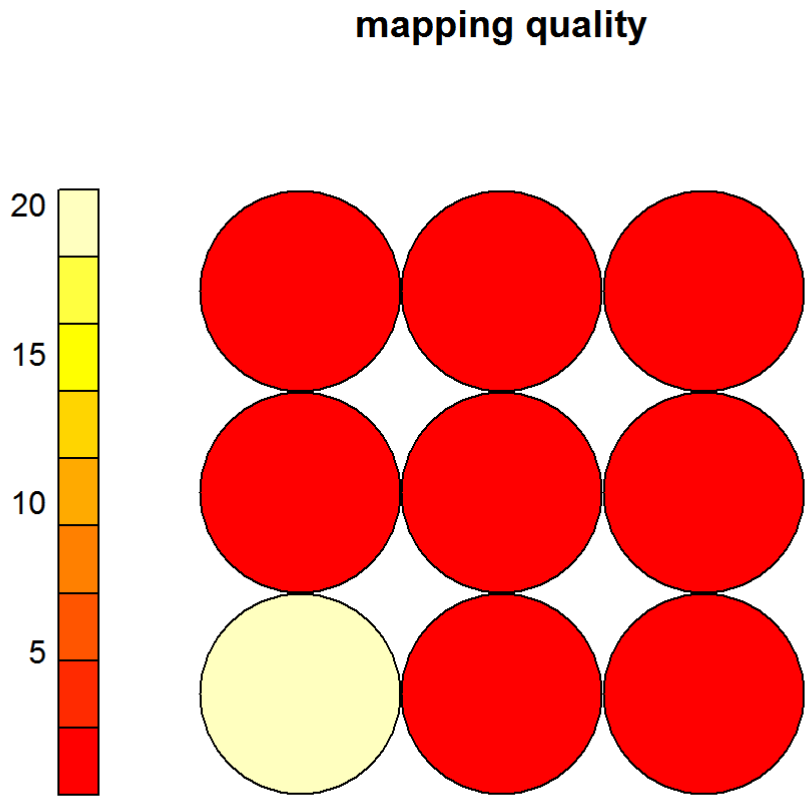
```
# SOM 시각화
plot(som_model, main = "feature distribution")
```




```
plot(som_model, type="counts", main = "cluster size")
```

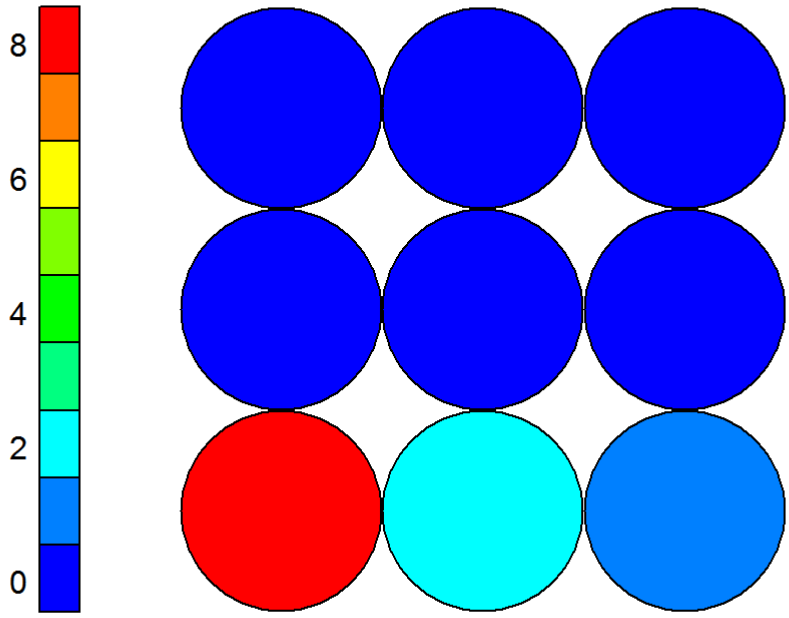


```
plot(som_model, type="quality", main = "mapping quality")  # 할당된 element 들의 유사도
```

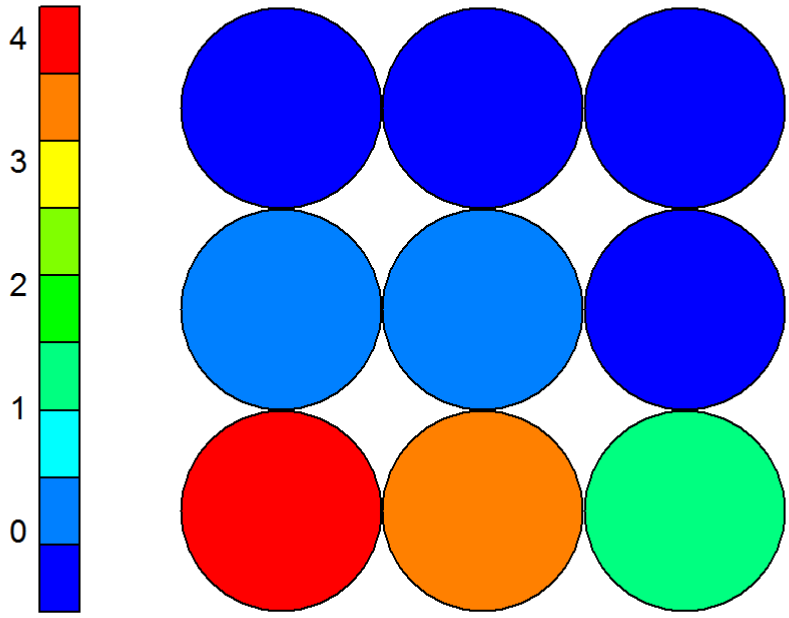



```
# 군집별 변수의 영향도
coolBlueHotRed <- function(n, alpha = 1) {
  rainbow(n, end=4/6, alpha=alpha)[n:1]
}

for (i in 1:ncol(som_model$data))
  plot(som_model, type="property", property=som_model$codes[,i], main=dimnames(som_model
$data)[[2]][i], palette.name=coolBlueHotRed)
```

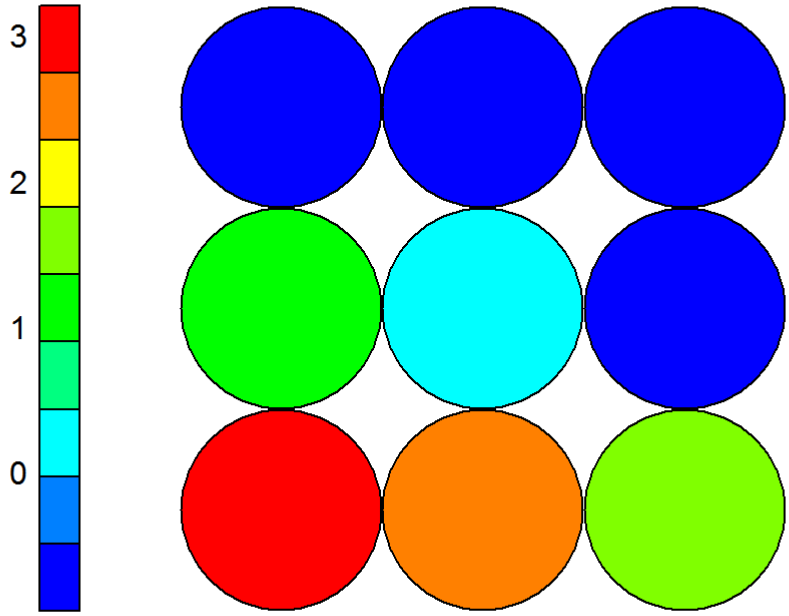
MONEY



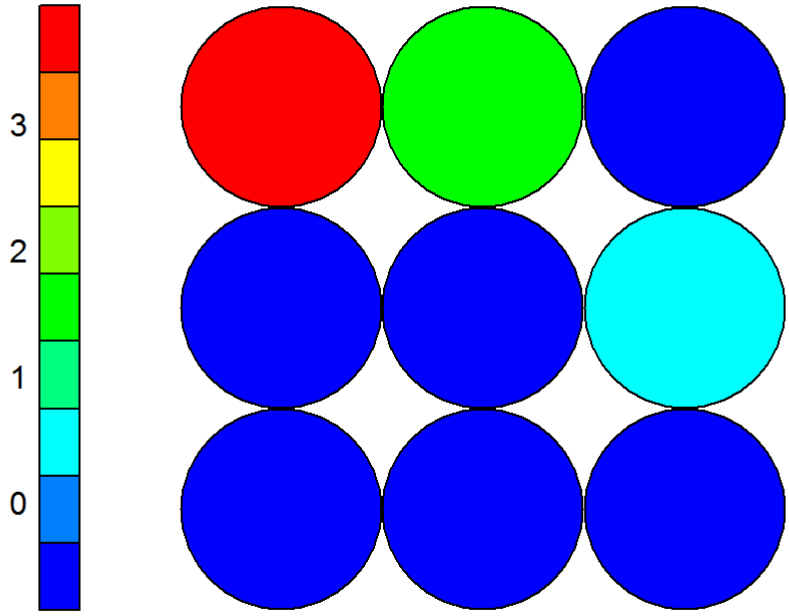
VISIT



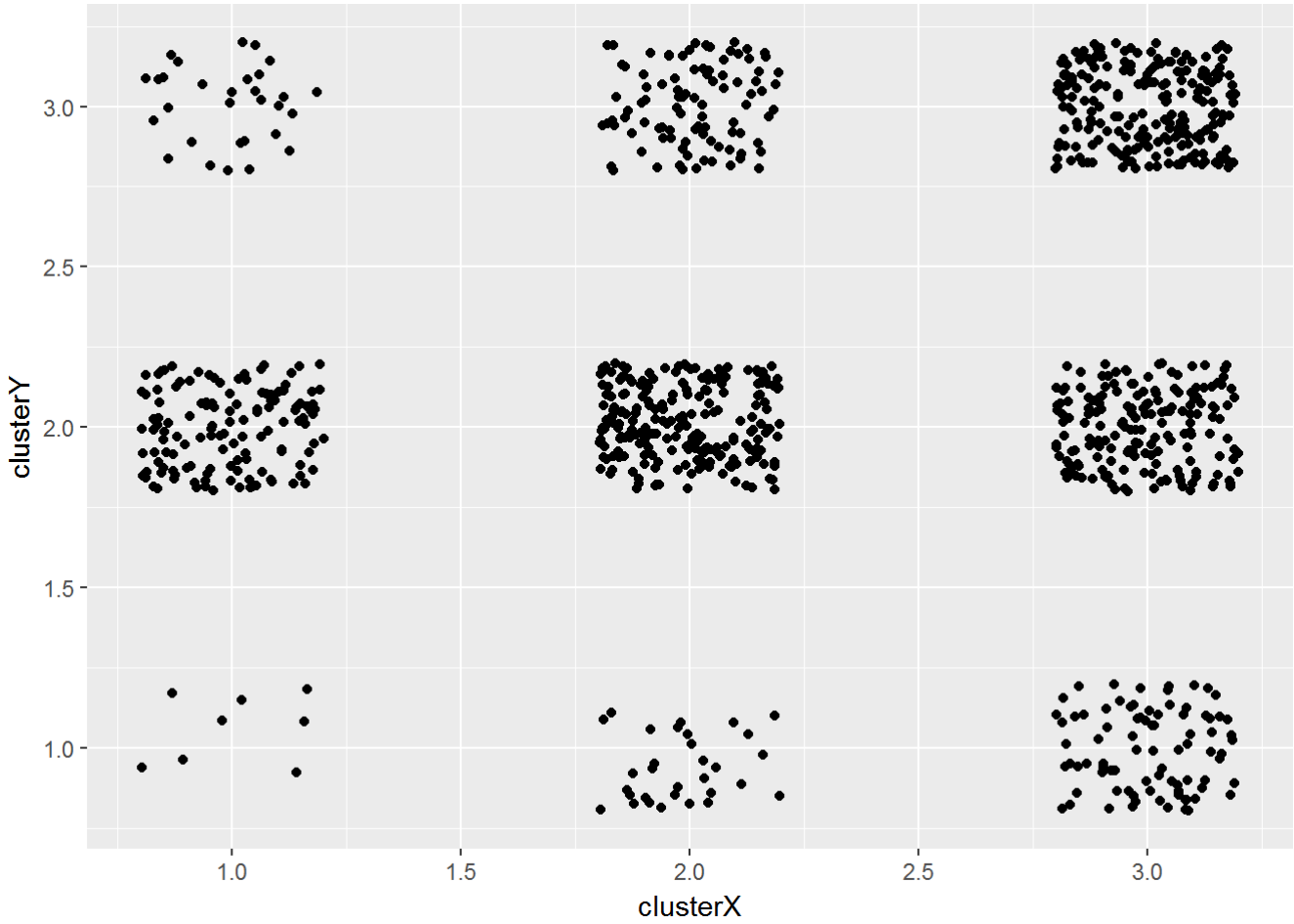
CROSS



API




```
# ggplot2 패키지를 이용하여 SPSS Modeler와 유사한 Grid 시각화
cdata$clusterX <- som_model$grid$pts[som_model$unit.classif,"x"]
cdata$clusterY <- som_model$grid$pts[som_model$unit.classif,"y"]
p <- ggplot(cdata, aes(clusterX, clusterY))
p + geom_jitter(position = position_jitter(width=.2, height=.2))
```



*# K-Means(k=6)*을 사용하여 *SOM neuron(10x10)*들 간의 유사성을 시각화

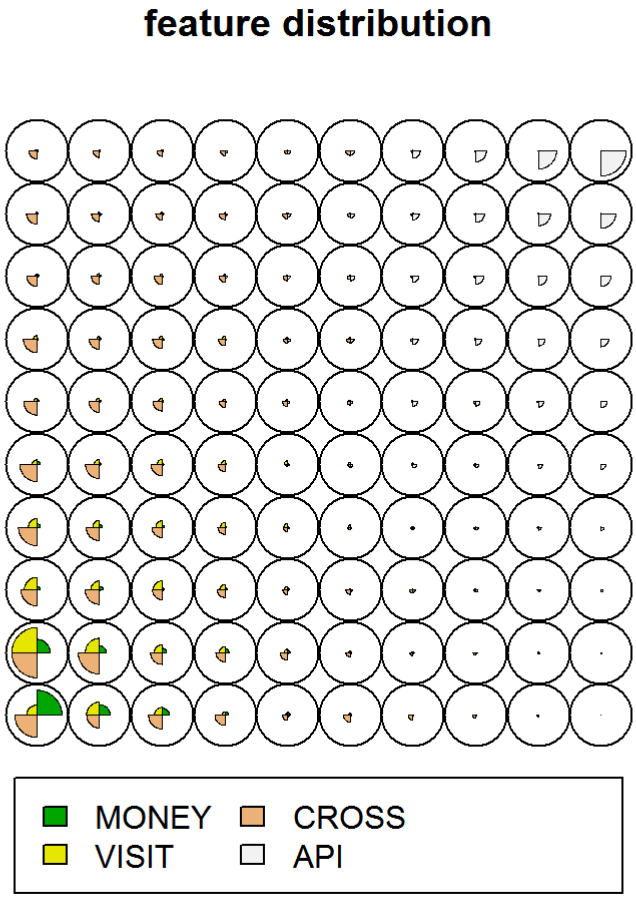
참조 슬라이드 링크 ()

```
library(kohonen)
library(RColorBrewer)
library(fields)
```

```
Hexagon <- function (x, y, unitcell = 1, col = col) {
  polygon(c(x, x, x + unitcell/2, x + unitcell, x + unitcell,
        x + unitcell/2), c(y + unitcell * 0.125,
                           y + unitcell * 0.875,
                           y + unitcell * 1.125,
                           y + unitcell * 0.875,
                           y + unitcell * 0.125,
                           y - unitcell * 0.125),
        col = col, border=NA)
}

cdata <- read.delim("data/Cluster.txt", stringsAsFactors=FALSE)
cdata.n <- scale(subset(cdata, select=-c(ID)))

som_model2 <- som(data = cdata.n, grid = somgrid(10, 10, "rectangular"))
k = 6
somClusters <- kmeans(som_model2$codes, centers = k)
somClusters
```

```
## K-means clustering with 6 clusters of sizes 2, 46, 26, 3, 7, 16
##
## Cluster means:
##       MONEY      VISIT      CROSS       API
## 1  8.2237320  4.8942046  3.6120212 -0.8606384
## 2 -0.2699493 -0.3223770 -0.3357094 -0.2496793
## 3  0.3566740  0.5408914  0.9180064 -0.6252556
## 4 -0.4596952 -0.6586599 -0.9624127  4.2828612
## 5  2.0625665  2.6264913  2.0452184 -0.7848548
## 6 -0.4199132 -0.5746073 -0.7785007  1.1355674
##
## Clustering vector:
##   [1] 1 5 5 3 3 3 2 2 2 2 1 5 5 3 3 2 2 2 2 2 5 5 3 3 3 2 2 2 2 2 5 3 3 3 2
##  [36] 2 2 2 2 2 3 3 3 3 2 2 2 2 2 6 3 3 3 2 2 2 2 2 6 6 3 3 3 2 2 2 6 6 6 6
##  [71] 3 3 3 2 2 2 6 6 6 6 3 2 2 2 2 2 2 6 6 6 4 3 2 2 2 2 2 2 6 6 4 4
##
## Within cluster sum of squares by cluster:
## [1] 25.401693 15.783441 19.868128  3.951426 16.116291  5.408053
##  (between_SS / total_SS =  84.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
# plotting 1
plot(som_model2, main = "feature distribution")
```




```
# plotting 2
plot(0, 0, type = "n", axes = FALSE, xlim = c(0, som_model2$grid$xdim), ylim = c(0, som_model2
$grid$ydim), xlab = "", ylab = "", asp = 1)
ColRamp <- rev(designer.colors(n=k, col=brewer.pal(k,"Set1")))
ColorCode <- rep("#FFFFFF", length(somClusters$cluster))

for (i in 1:length(somClusters$cluster))
  ColorCode[i] <- ColRamp[somClusters$cluster[i]]

offset <- 0.5
for (row in 1:som_model2$grid$ydim) {
  for (column in 0:(som_model2$grid$xdim-1))
    Hexagon(column + offset, row - 1, col = ColorCode[row + som_model2$grid$ydim * column])
  offset <- ifelse(offset, 0, 0.5)
}
```