DESCO - Knowledge Discovery - Association Rules

1141074 - Sérgio Silva | 1970400 - Pedro Neves | 1040706 - Sérgio Castro

5/3/2018

Neste documento descreve-se a criação de modelos para identificar perfis de grupos de clientes, com o objetivo de efetuar recomendações dos produtos mais adequados para cada grupo de clientes.

1. Exploração e preparação dos dados

Para o cálculo do valor de RFM dos clientes, foi efetuado tratamento dos dados das tabelas **TRANSACTION.dat**, **TRANSACTION_ITEM.dat** e **CARD.DAT**, idêntico ao realizado para a previsão de resposta a campanhas.

Foi criada uma categoria para dividir os clientes por intervalos de idades. Os intervalos considerados foram: menor de 50 anos, maior ou igual a 50 e menor de 65 anos, maior ou igual a 65 anos.

```
df_customers$ageInterval <- cut(df_customers$age,
                      breaks = c(0, 50, 65, +Inf),
                      labels = c("< 50", "< 65", ">= 65"),
                      right = FALSE)
```

Verificação dos dados dos clientes.

```
summary(df_customers)
```

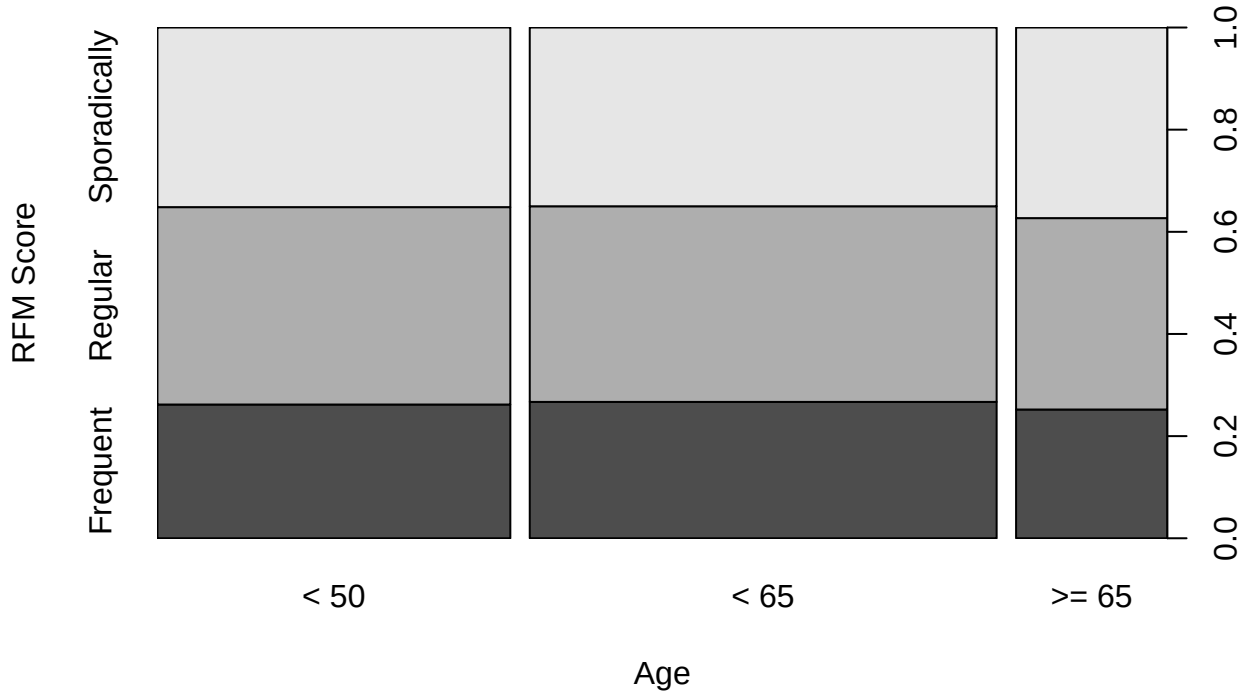
```
##     CardID                    City          Region         PostalCode
##  Length:60519       Catburg    :10302   Central:30176   A039798 : 4467
##  Class :character   Foxton     : 9987   Eastern:30343   A001761 :  538
##  Mode  :character   Kingsville :10130                   A024496 :  445
##                     Princeton  :10042                   A0104173:  286
##                     Queensbury :10004                   A049814 :  280
##                     Ravensville:10054                   A0117302:  279
##                                                         (Other) :54224
##  CardStartDate              Gender       DateOfBirth
##  Min.   :1998-01-01   Feminino :30412   Min.   :1902-02-13
##  1st Qu.:1998-11-01   Masculino:30107   1st Qu.:1954-11-12
##  Median :1999-09-02                     Median :1962-01-26
##  Mean   :1999-09-18                     Mean   :1961-06-08
##  3rd Qu.:2000-06-29                     3rd Qu.:1969-02-08
##  Max.   :2001-12-30                     Max.   :1991-12-11
##
##   MaritalStatus   HasChildren  NumChildren    YoungestChild
##  Casado  :20033   Sim:34122   Min.   :0.000   Min.   : 0.000
##  Solteiro:20196   Não:26397   1st Qu.:0.000   1st Qu.: 0.000
##  Outro   :20290               Median :1.000   Median : 0.000
##                               Mean   :1.147   Mean   : 6.344
##                               3rd Qu.:2.000   3rd Qu.:11.000
##                               Max.   :7.000   Max.   :68.000
##
##    rfm_score        age         clientYears         rfm_score_cat
##  Min.   :111   Min.   : 23.0   Min.   :13.00   Frequent    :15904
```

```
##  1st Qu.:214   1st Qu.: 46.0   1st Qu.:15.00   Regular     :23179
##  Median :324   Median : 53.0   Median :15.00   Sporadically:21436
##  Mean   :330   Mean   : 53.6   Mean   :15.28
##  3rd Qu.:453   3rd Qu.: 60.0   3rd Qu.:16.00
##  Max.   :555   Max.   :113.0   Max.   :17.00
##
##  ageInterval
##  < 50 :21992
##  < 65 :29097
##  >= 65: 9430
##
##
##
##
```

```
summary(df_customers$clientYears)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.00   15.00   15.00   15.28   16.00   17.00
```

```
# Amplitude de clientYears é pequena.

# CardID é mantido para permitir a identificação das transações depois
# da criação dos clusters; no entanto, não é utilizado para a criação dos clusters

dataCustomers <- df_customers[, c("CardID", "Region", "Gender", "MaritalStatus", "HasChildren",
                                  "rfm_score_cat", "clientYears", "age", "ageInterval")]


with(dataCustomers,
  plot( ageInterval, rfm_score_cat, xlab = "Age", ylab = "RFM Score")
  )
```



Clustering

Determine Best Number of Clusters in Customers Data Set

```
#library(NbClust)
#library(cluster)

#set.seed(123)

#gower.dist <- daisy(dataCustomers[1:1000, ], metric = "gower")

#nb <- NbClust(diss = gower.dist, distance = NULL, min.nc = 2,
#        max.nc = 10, method = "complete", index ="all")
```

Model-based Clustering

```
library(VarSelLCM)
set.seed(123)

# without ageInterval
out <- VarSelCluster(dataCustomers[1:1000, -c("CardID", "ageInterval")], gvals = 2, nbcores = 2)

#VarSelShiny(out)
```

Hierarchical Clustering

```
#clusters <- hclust(dist(data[, 5]))
#plot(clusters)
#clusterCut <- cutree(clusters, 3)

library(cluster)
set.seed(123)

# without clientYears and age
gower.dist <- daisy(dataCustomers[1:500, -c("CardID", "clientYears", "age")], metric = "gower")
summary(gower.dist)
```

```
## 124750 dissimilarities, summarized :
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.5000  0.5000  0.5647  0.6667  1.0000
## Metric :  mixed ;  Types = N, N, N, N, N, N
## Number of objects : 500
```

```
gower.mat <- as.matrix(gower.dist)


# Most similar pair
dataCustomers[
  which(gower.mat == min(gower.mat[gower.mat != min(gower.mat)]),
        arr.ind = TRUE)[1, ], ]
```

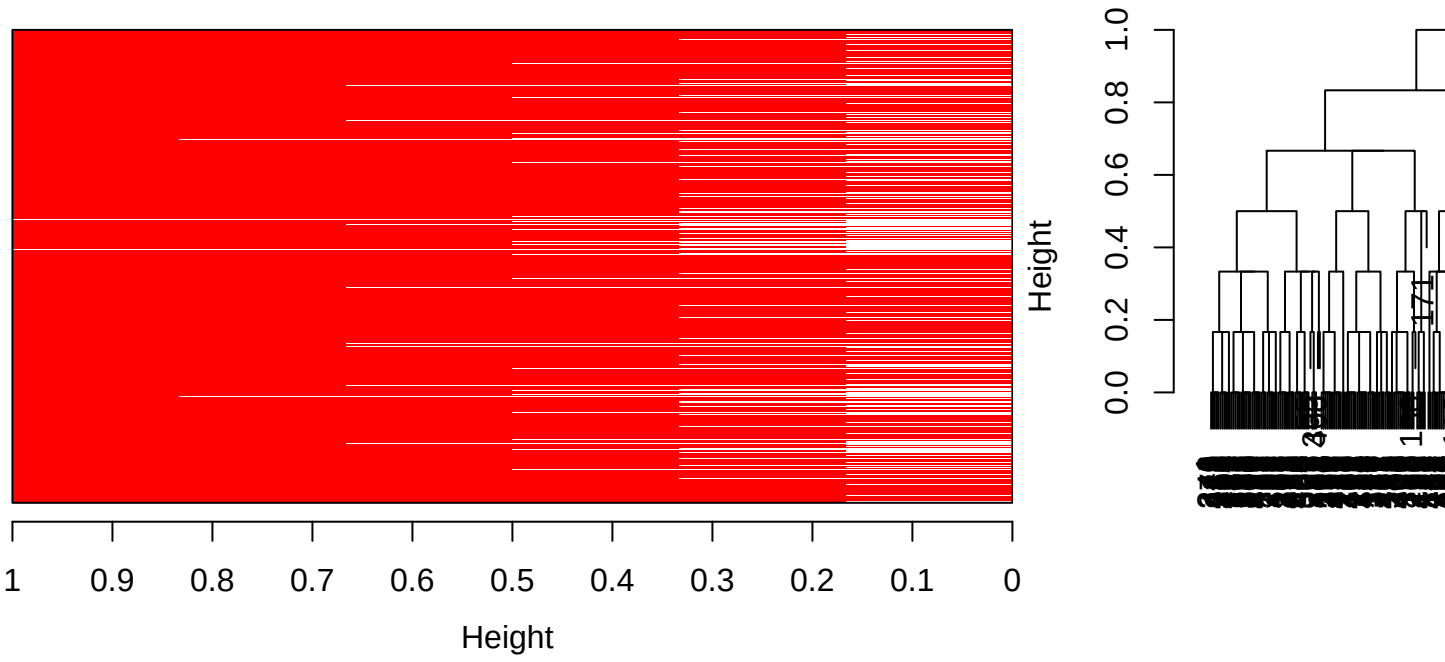
```
##         CardID  Region   Gender MaritalStatus HasChildren rfm_score_cat
## 1: C0100000726 Eastern Feminino        Casado         Sim      Frequent
## 2: C0100000111 Eastern Feminino        Casado         Sim  Sporadically
##    clientYears age ageInterval
## 1:          14  59        < 65
## 2:          14  51        < 65
```

```
# Most dissimilar pair
dataCustomers[
  which(gower.mat == max(gower.mat[gower.mat != max(gower.mat)]),
        arr.ind = TRUE)[1, ], ]
```

```
##         CardID  Region    Gender MaritalStatus HasChildren rfm_score_cat
## 1: C0100000375 Eastern Masculino      Solteiro         Não       Regular
## 2: C0100000111 Eastern  Feminino        Casado         Sim  Sporadically
##    clientYears age ageInterval
## 1:          14  47        < 50
## 2:          14  51        < 65
```

```
# Divisive (DIANA)
divisive.clust <- diana(as.matrix(gower.dist),
                  diss = TRUE, keep.diss = TRUE)

plot(divisive.clust, main = "Divisive")
```

Divisive



Divisive Coefficient = 0.98

```
# Agglomerative (AGNES)
```

Categorial clustering with k-modes algorithm

```
library(klaR)
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
set.seed(123)

# Set number of clusters
kNumberClusters <- 3

# without age and clientYears
clusters.kmodes <- kmodes(dataCustomers[, -c("CardID", "age", "clientYears")], modes = kNumberClusters,

# Place customer in its cluster
dataCustomers$cluster <- clusters.kmodes$cluster
clusters <- split(dataCustomers, dataCustomers$cluster)
```

Visualize differences between clusters

```
dataCustomers[, .N, by = .(cluster, Gender)][order(cluster, Gender)]
```

```
##    cluster    Gender     N
## 1:       1  Feminino 19734
## 2:       1 Masculino  6046
## 3:       2  Feminino  4725
## 4:       2 Masculino 12155
## 5:       3  Feminino  5953
## 6:       3 Masculino 11906
```

```
dataCustomers[, .N, by = .(cluster, Region)][order(cluster, Region)]
```

```
##    cluster  Region     N
## 1:       1 Central 10174
## 2:       1 Eastern 15606
## 3:       2 Central 13502
## 4:       2 Eastern  3378
## 5:       3 Central  6500
## 6:       3 Eastern 11359
```

```
dataCustomers[, .N, by = .(cluster, rfm_score_cat)][order(cluster, rfm_score_cat)]
```

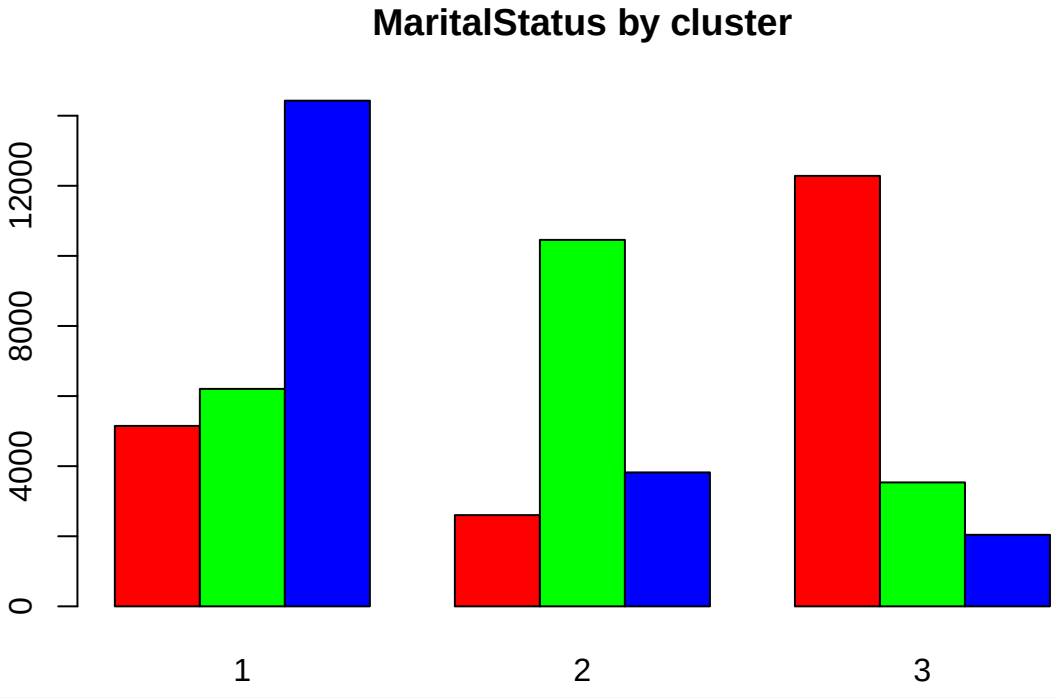
```
##    cluster rfm_score_cat     N
## 1:       1      Frequent  3818
## 2:       1       Regular  7695
## 3:       1  Sporadically 14267
## 4:       2      Frequent  8128
## 5:       2       Regular  4367
```

```
## 6:       2  Sporadically  4385
## 7:       3      Frequent  3958
## 8:       3       Regular 11117
## 9:       3  Sporadically  2784
```

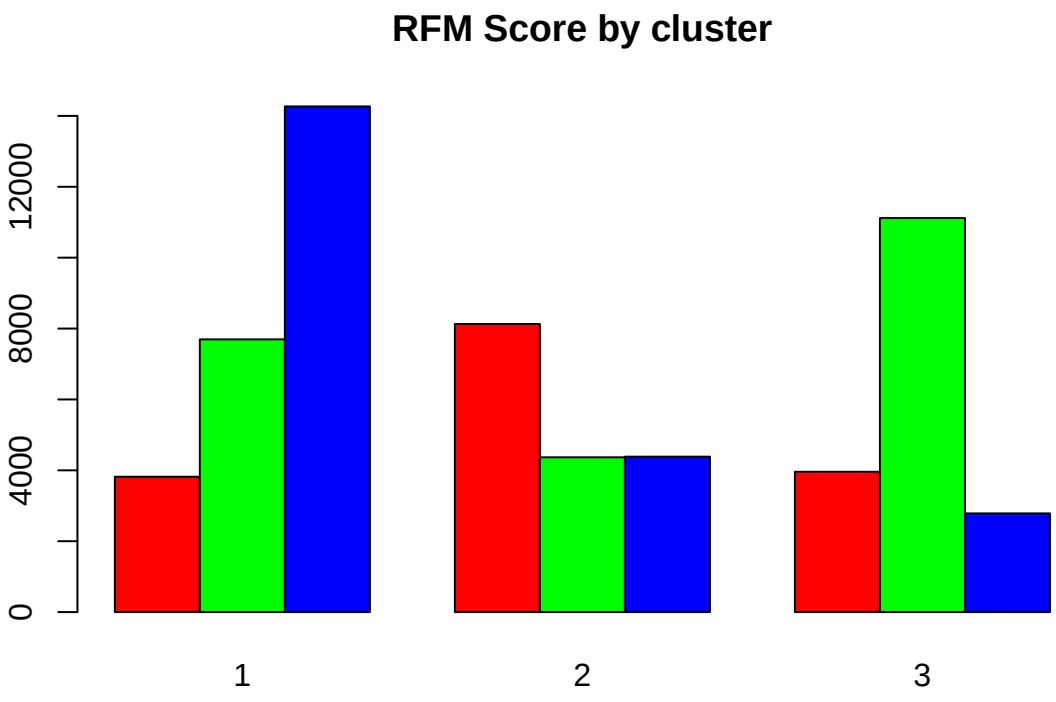
```
barplot(table(dataCustomers$Region, dataCustomers$cluster),
        beside = T, col = c("red", "green"),
        main = "Region by cluster")
```


```
barplot(table(dataCustomers$Gender, dataCustomers$cluster),
        beside = T, col = c("red", "green"),
        main = "Gender by cluster")
```


```
barplot(table(dataCustomers$MaritalStatus, dataCustomers$cluster),
        beside = T, col = c("red", "green", "blue"),
        main = "MaritalStatus by cluster")
```

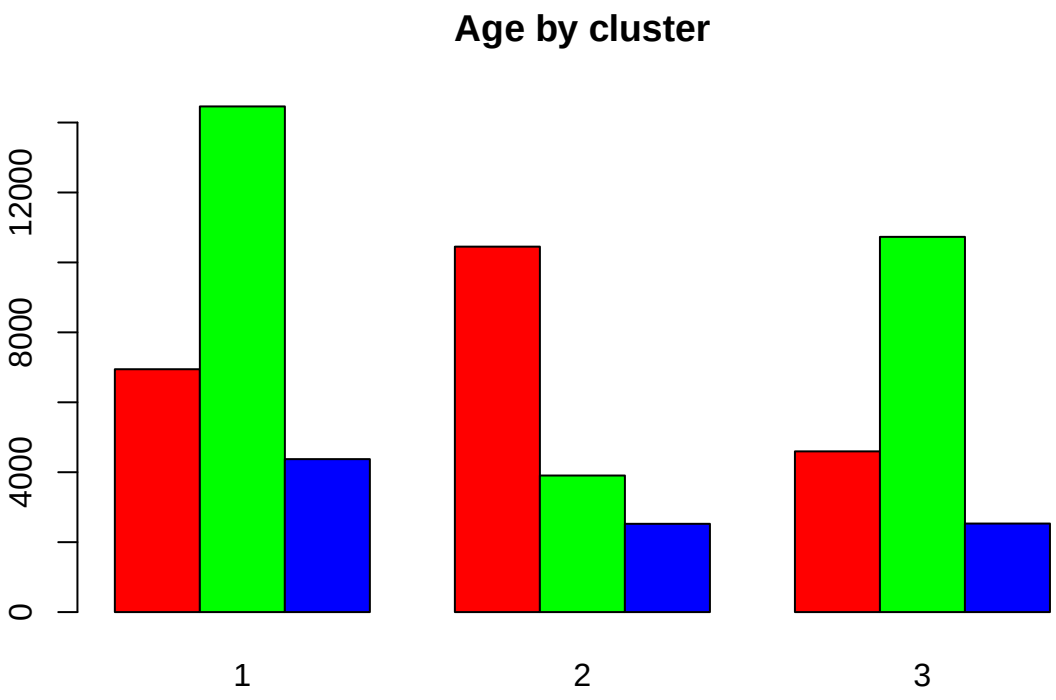



```
barplot(table(dataCustomers$rfm_score_cat, dataCustomers$cluster),
        beside = T, col = c("red", "green", "blue"),
        main = "RFM Score by cluster")
```




```
barplot(table(dataCustomers$HasChildren, dataCustomers$cluster),
        beside = T, col = c("red", "green"),
        main = "HasChildren by cluster")
```


```
barplot(table(dataCustomers$ageInterval, dataCustomers$cluster),
        beside = T, col = c("red", "green", "blue"),
        main = "Age by cluster")
```



Differences between clusters

```
dataCustomers.cl1 <- clusters[[1]]
round(prop.table(table(dataCustomers.cl1$Region))*100, digits = 2)
```

```
##
## Central Eastern
##   39.46   60.54
```

```
round(prop.table(table(dataCustomers.cl1$Gender))*100, digits = 2)
```

```
##
##  Feminino Masculino
##     76.55     23.45
```

```
round(prop.table(table(dataCustomers.cl1$MaritalStatus))*100, digits = 2)
```

```
##
##   Casado Solteiro    Outro
##    19.97    24.07    55.97
```

```
round(prop.table(table(dataCustomers.cl1$HasChildren))*100, digits = 2)
```

```
##
##   Sim   Não
## 40.91 59.09
```

```
round(prop.table(table(dataCustomers.cl1$rfm_score_cat))*100, digits = 2)
```

```
##
##     Frequent      Regular Sporadically
##        14.81        29.85        55.34
```

```
round(prop.table(table(dataCustomers.cl1$ageInterval))*100, digits = 2)
```

```
##
##  < 50  < 65 >= 65
## 26.94 56.09 16.97
```

```
dataCustomers.cl2 <- clusters[[2]]
round(prop.table(table(dataCustomers.cl2$Region))*100, digits = 2)
```

```
##
## Central Eastern
##   79.99   20.01
```

```
round(prop.table(table(dataCustomers.cl2$Gender))*100, digits = 2)
```

```
##
##  Feminino Masculino
##     27.99     72.01
```

```
round(prop.table(table(dataCustomers.cl2$MaritalStatus))*100, digits = 2)
```

```
##
##   Casado Solteiro    Outro
##    15.42    61.94    22.64
```

```
round(prop.table(table(dataCustomers.cl2$HasChildren))*100, digits = 2)
```

```
##
##   Sim   Não
## 41.37 58.63
```

```
round(prop.table(table(dataCustomers.cl2$rfm_score_cat))*100, digits = 2)
```

```
##
##     Frequent      Regular Sporadically
##        48.15        25.87        25.98
```

```
round(prop.table(table(dataCustomers.cl2$ageInterval))*100, digits = 2)
```

```
##
##  < 50  < 65 >= 65
## 61.91 23.13 14.95
```

```
dataCustomers.cl3 <- clusters[[3]]
round(prop.table(table(dataCustomers.cl3$Region))*100, digits = 2)
```

```
##
## Central Eastern
##    36.4    63.6
```

```
round(prop.table(table(dataCustomers.cl3$Gender))*100, digits = 2)
```

```
##
##  Feminino Masculino
##     33.33     66.67
```

```
round(prop.table(table(dataCustomers.cl3$MaritalStatus))*100, digits = 2)
```

```
##
##   Casado Solteiro    Outro
```

```
##    68.77    19.80    11.43
```

```
round(prop.table(table(dataCustomers.cl3$HasChildren))*100, digits = 2)
```

```
##
##   Sim   Não
## 92.91  7.09
```

```
round(prop.table(table(dataCustomers.cl3$rfm_score_cat))*100, digits = 2)
```

```
##
##     Frequent      Regular Sporadically
##        22.16        62.25        15.59
```

```
round(prop.table(table(dataCustomers.cl3$ageInterval))*100, digits = 2)
```

```
##
##  < 50  < 65 >= 65
## 25.73 60.09 14.18
```

Clustering by RFM Score

```
# Divide customers by its RFM Score
rfm.clusters <- split(dataCustomers, dataCustomers$rfm_score_cat)

dataCustomers.rfmFrequent <- rfm.clusters$Frequent
dataCustomers.rfmRegular <- rfm.clusters$Regular
dataCustomers.rfmSporadically <- rfm.clusters$Sporadically
```

Dados das compras

```
## Tabela ITEM.dat
items <- fread("DATA-CRM/ITEM.dat", quote = "'")

### Verificação dos dados da tabela item, tal como o número de colunas e linhas, bem como se os dados fo
summary(items)
```

```
##    ItemCode         ItemDescription    CategoryCode
##  Length:819         Length:819         Length:819
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##  SubCategoryCode     BrandCode         UpmarketFlag
##  Length:819         Length:819         Length:819
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

```
dim(items)
```

```
## [1] 819   6
```

```
str(items)
```

```
## Classes 'data.table' and 'data.frame':   819 obs. of  6 variables:
##  $ ItemCode       : chr  "I0000000001" "I0000000002" "I0000000003" "I0000000004" ...
##  $ ItemDescription: chr  "BXT - Listen2This1" "BXT - Listen2This2" "BXT - Listen2This3" "ENDOS - ENSI
```

```
##  $ CategoryCode   : chr  "MACC" "MACC" "MACC" "MACC" ...
##  $ SubCategoryCode: chr  "PMSPE" "PMSPE" "PMSPE" "PMSPE" ...
##  $ BrandCode      : chr  "BBXT" "BBXT" "BBXT" "BENDOS" ...
##  $ UpmarketFlag   : chr  "F" "F" "F" "F" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
#Verificar se a tabela possui dados nulos
table(is.na(items))
```

```
##
## FALSE
##  4914
```

```
## Tabelas CATEGORY.dat e SUBCATEGORY.dat
categories <- fread("DATA-CRM/CATEGORY.dat", quote = "'")
subcategories <- fread("DATA-CRM/SUBCATEGORY.dat", quote = "'")
```

Join com a tabela de transações + cardID

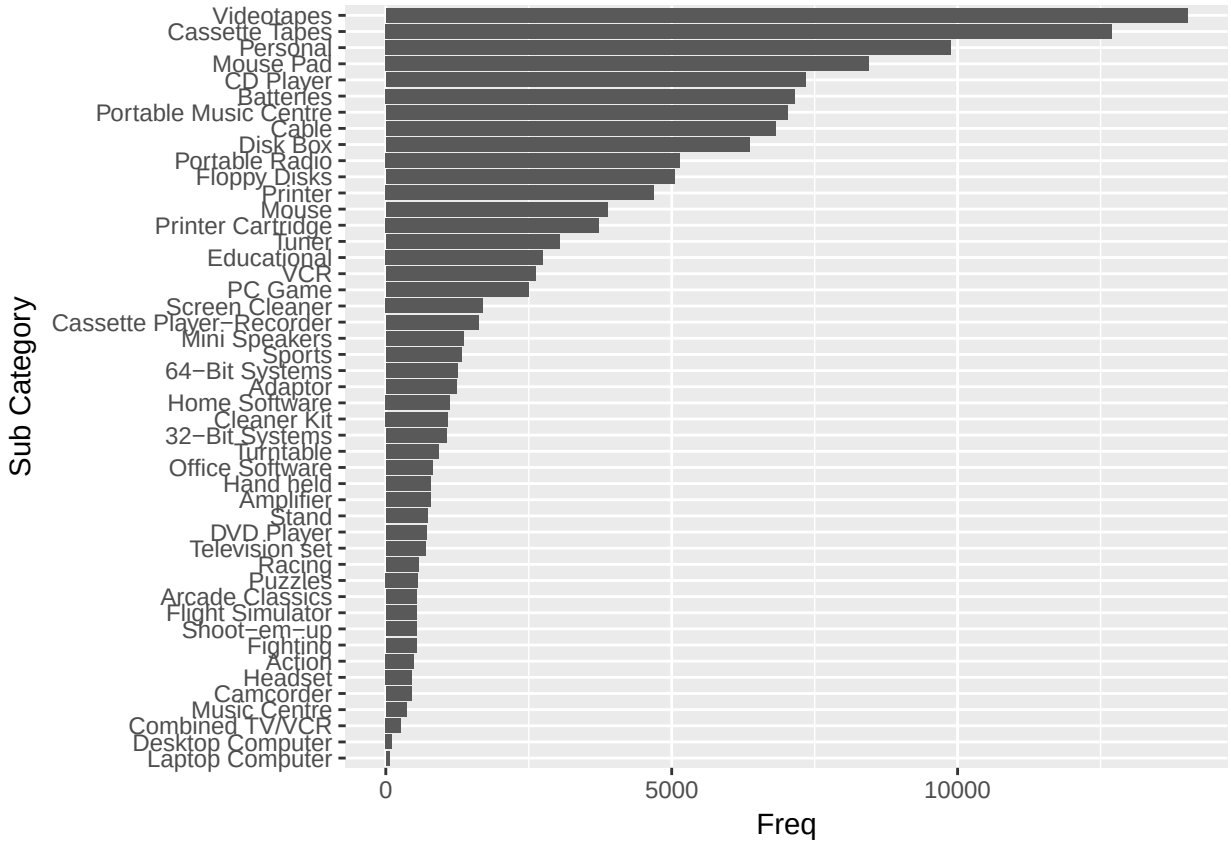
```
result.aux <- merge(items, categories, all.x = TRUE, by = 'CategoryCode')
result.aux <- merge(result.aux, subcategories, all.x = TRUE, by = 'SubCategoryCode')

result.purchases <- merge(result.transactions, result.aux[ ,c(3:5, 7:8)], all.x = TRUE, by = 'ItemCode')

# Se retirados 'ItemNumber' e 'TransactionID' passam a existir observações repetidas
dataPurchases <- result.purchases[, c("CardID", "Date", "PaymentMethod", "Amount", "ItemDescription", "C
```

```
dataPurchases$PaymentMethod <- as.factor(dataPurchases$PaymentMethod)
dataPurchases$ItemDescription <- as.factor(dataPurchases$ItemDescription)
dataPurchases$CategoryDescription <- as.factor(dataPurchases$CategoryDescription)
dataPurchases$SubCategoryDescription <- as.factor(dataPurchases$SubCategoryDescription)
dataPurchases$BrandCode <- as.factor(dataPurchases$BrandCode)


# Split dataPurchases by clusters
dataPurchases.cl1 <- merge(dataPurchases, dataCustomers.cl1[, c("CardID")], by = "CardID")
dataPurchases.cl2 <- merge(dataPurchases, dataCustomers.cl2[, c("CardID")], by = "CardID")
dataPurchases.cl3 <- merge(dataPurchases, dataCustomers.cl3[, c("CardID")], by = "CardID")

# Split dataPurchases by rfm clusters
dataPurchases.rmfFrequent <- merge(dataPurchases, dataCustomers.rfmFrequent[, c("CardID")], by = "CardI
dataPurchases.rfmRegular <- merge(dataPurchases, dataCustomers.rfmRegular[, c("CardID")], by = "CardID"
dataPurchases.rfmSporadically <- merge(dataPurchases, dataCustomers.rfmSporadically[, c("CardID")], by =
```

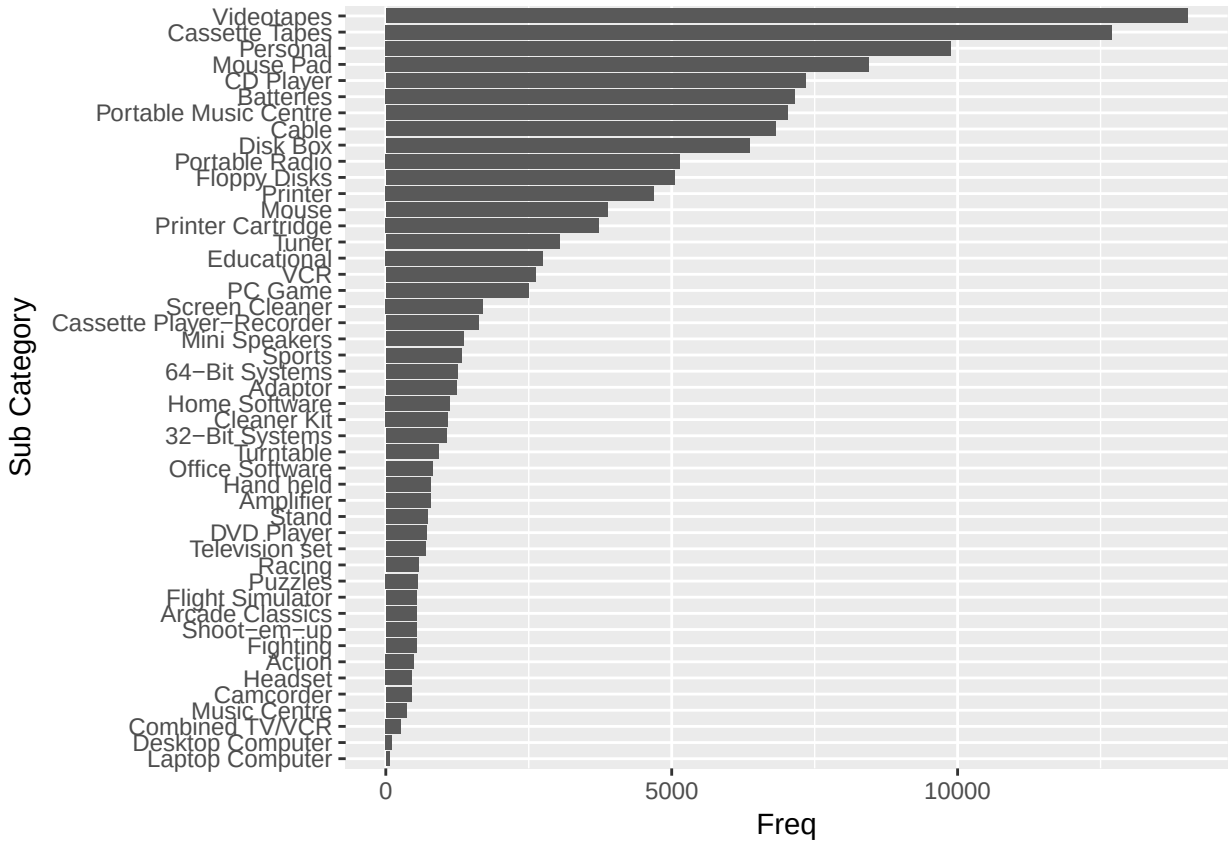
Vendas por subcategorias de produtos

```
# Frequência das subcategorias de produtos no cluster 1
# ordenado por ordem descrescente
sub_ord <- factor(dataPurchases.cl1$SubCategoryDescription,
                  levels = rev(levels(fct_infreq(dataPurchases.cl1$SubCategoryDescription))))
```

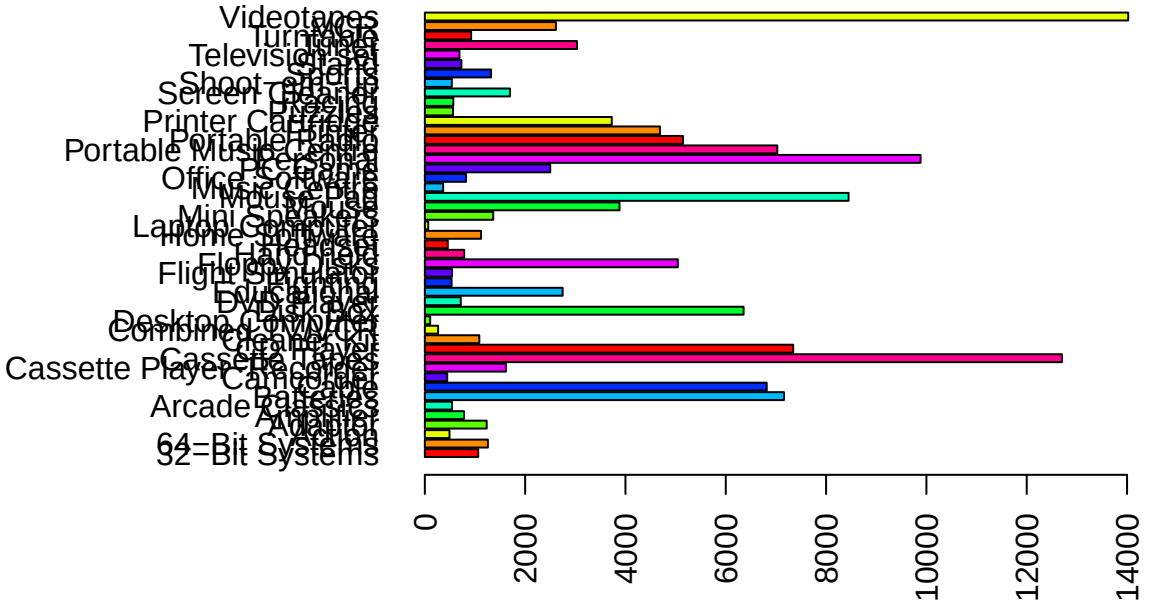
```r
ggplot(as.data.frame(dataPurchases.cl1$SubCategoryDescription), aes(x = sub_ord)) +
  geom_bar() + labs(x = "Sub Category", y = "Freq") + coord_flip()
```



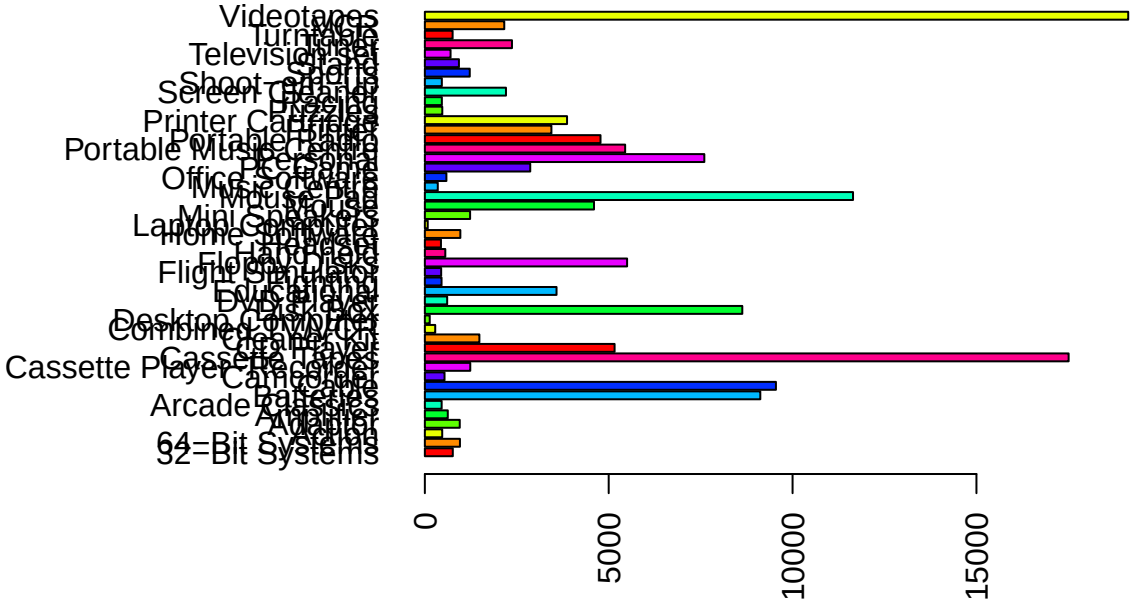

```r
# Alternativamente
x <- as.data.frame(sort(
  table(dataPurchases.cl1$SubCategoryDescription, dnn = c("SubCategory")), decreasing = F))

ggplot(x, aes(x = reorder(SubCategory, Freq), y = Freq)) +
  geom_bar(stat = 'identity') + labs(x = "Sub Category", y = "Freq") + coord_flip()
```



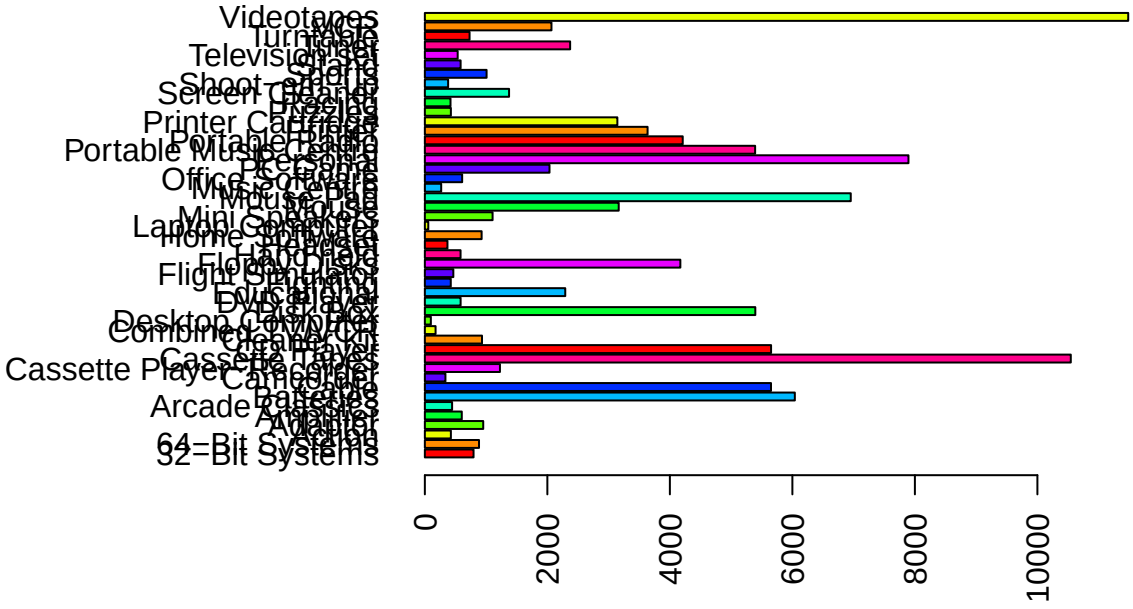

```
par(las = 2)
par(mar = c(5, 12, 5, 2))

plot(dataPurchases.cl1$SubCategoryDescription, col=rainbow(11), horiz = TRUE)
```



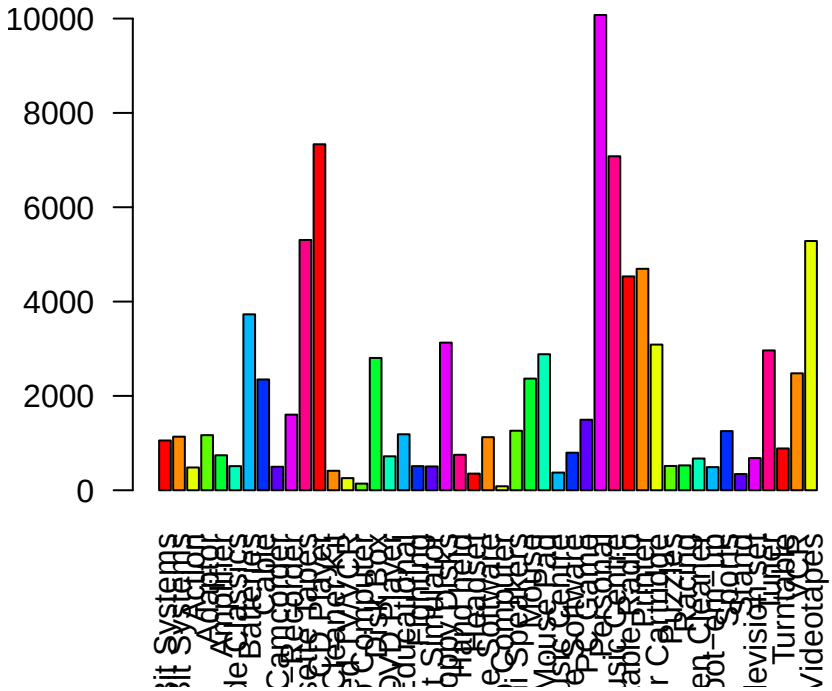

```
plot(dataPurchases.cl2$SubCategoryDescription, col=rainbow(11), horiz = TRUE)
```



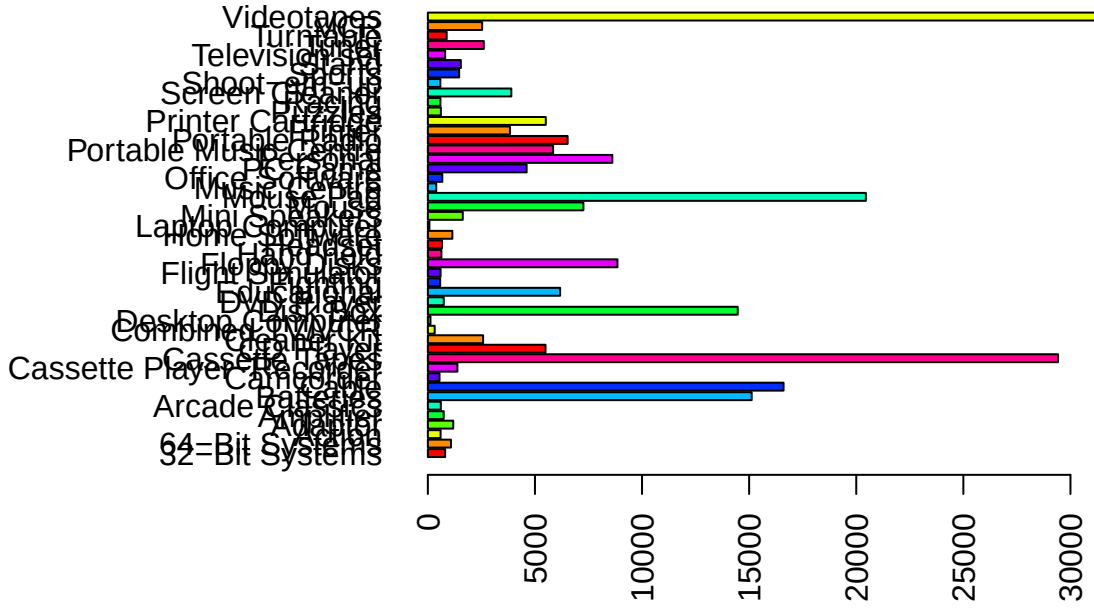

```
plot(dataPurchases.cl3$SubCategoryDescription, col=rainbow(11), horiz = TRUE)
```



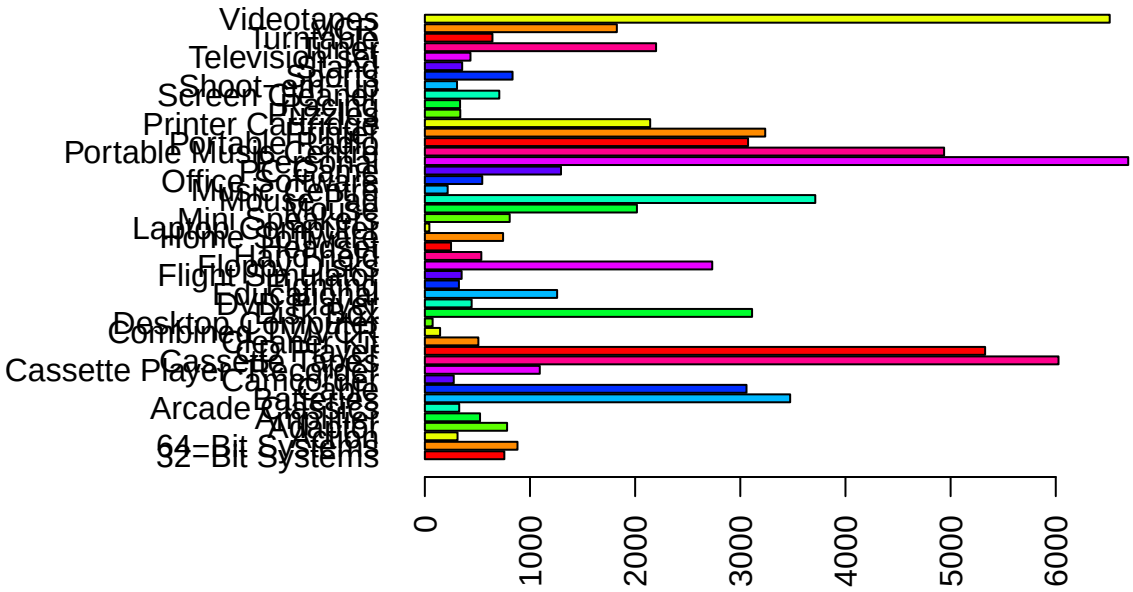

```
plot(dataPurchases.rfmRegular$SubCategoryDescription, col=rainbow(11))
```




```
plot(dataPurchases.rmfFrequent$SubCategoryDescription, col=rainbow(11), horiz = TRUE)
```




```
plot(dataPurchases.rfmSporadically$SubCategoryDescription, col=rainbow(11), horiz = TRUE)
```




```
# Todas as subcategorias de produtos
levels(dataPurchases$SubCategoryDescription)
```

```
##  [1] "32-Bit Systems"           "64-Bit Systems"
##  [3] "Action"                   "Adaptor"
##  [5] "Amplifier"                "Arcade Classics"
##  [7] "Batteries"                "Cable"
##  [9] "Camcorder"                "Cassette Player-Recorder"
## [11] "Cassette Tapes"           "CD Player"
## [13] "Cleaner Kit"              "Combined TV/VCR"
## [15] "Desktop Computer"         "Disk Box"
## [17] "DVD Player"               "Educational"
## [19] "Fighting"                 "Flight Simulator"
## [21] "Floppy Disks"             "Hand held"
## [23] "Headset"                  "Home Software"
## [25] "Laptop Computer"          "Mini Speakers"
## [27] "Mouse"                    "Mouse Pad"
## [29] "Music Centre"             "Office Software"
## [31] "PC Game"                  "Personal"
## [33] "Portable Music Centre"    "Portable Radio"
## [35] "Printer"                  "Printer Cartridge"
## [37] "Puzzles"                  "Racing"
## [39] "Screen Cleaner"           "Shoot-em-up"
## [41] "Sports"                   "Stand"
## [43] "Television set"           "Tuner"
## [45] "Turntable"                "VCR"
## [47] "Videotapes"
```

```
# Número de vendas por subcategorias
baskets.subcat <- count(dataPurchases, c("dataPurchases$SubCategoryDescription"))
baskets.subcat <- baskets.subcat[order(-baskets.subcat$freq), ]
```

```
## Warning: Unknown or uninitialised column: 'freq'.
```

```
## Error in -baskets.subcat$freq: invalid argument to unary operator
```

```r
colnames(baskets.subcat) <- c("subcategory", "freq")

length(unique(baskets.subcat$subcategory))       # 47 subcategorias
```

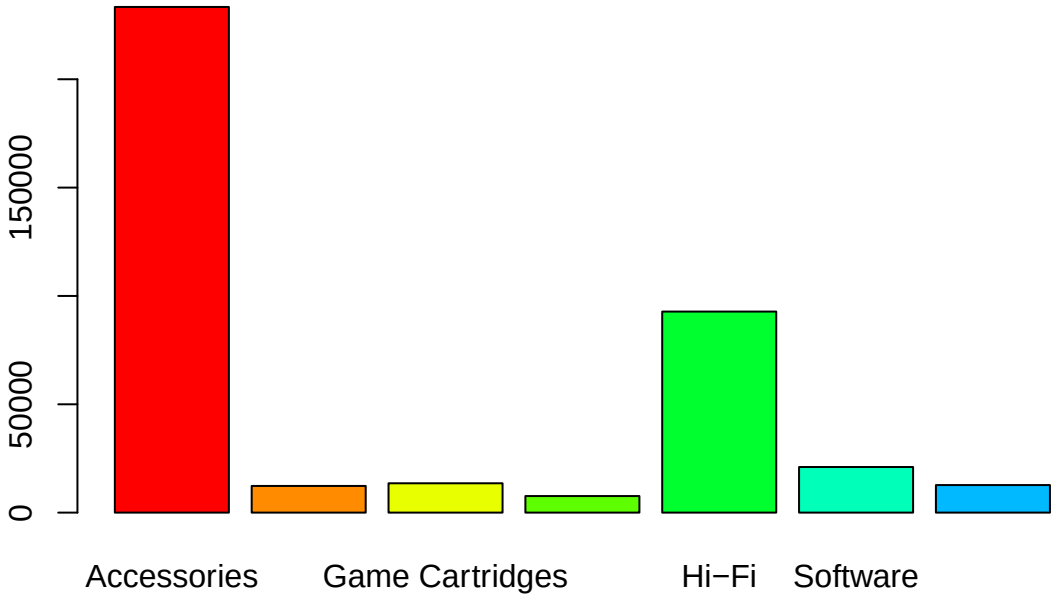
```
## [1] 1
```

```r
# Número médio de itens por basket (subcategoria)
summary(baskets.subcat$freq)
```

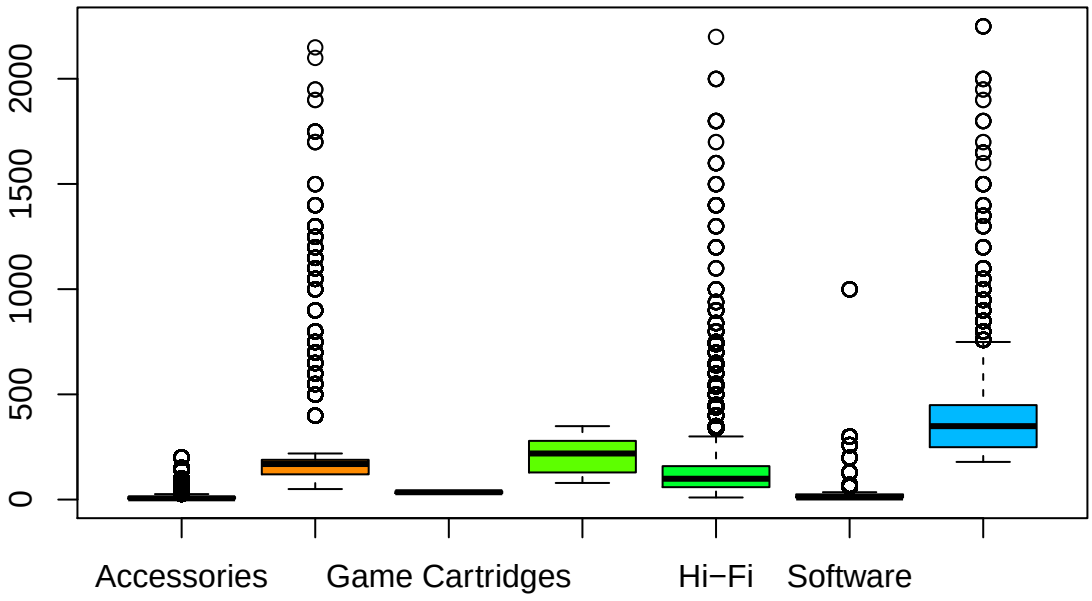
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  393381  393381  393381  393381  393381  393381
```

Vendas por categorias de produtos

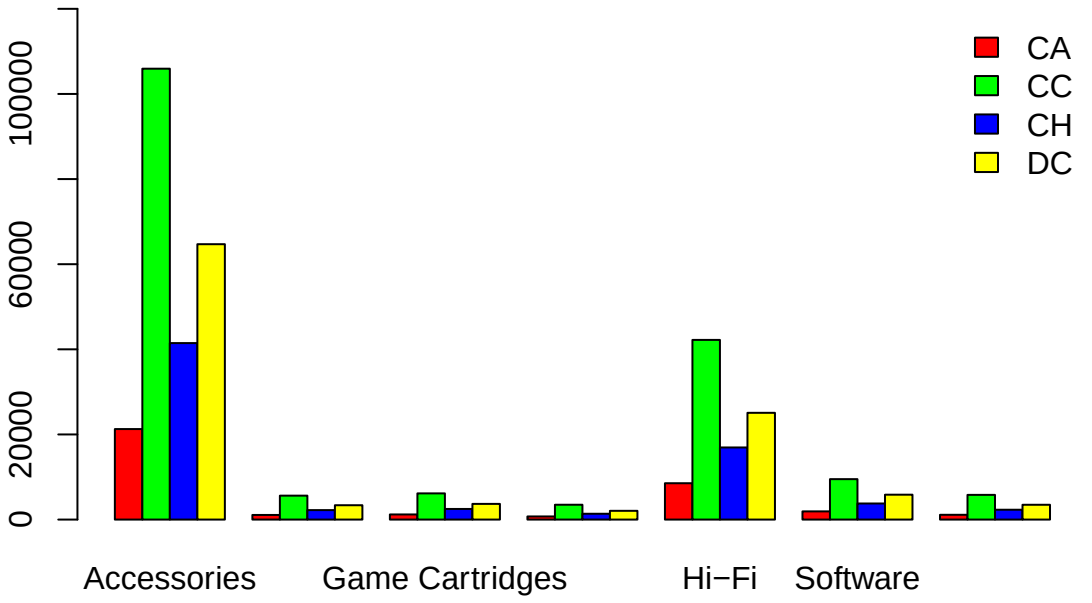
```r
plot(dataPurchases$CategoryDescription, col=rainbow(11))
```




```r
boxplot( Amount ~ CategoryDescription, data = dataPurchases, main = "Valor das Vendas por Categoria", c
```




```
barplot(table(dataPurchases$PaymentMethod, dataPurchases$CategoryDescription),
        beside = T, col = c("red", "green", "blue", "yellow"),
        main = "Métodos de Pagamento por Categoria", ylim = c(0, 120000))

legend("topright", levels(dataPurchases$PaymentMethod), bty = "n", fill=c("red", "green", "blue", "yell
```




```
# Todas as categorias de produtos
levels(dataPurchases$CategoryDescription)
```

```
## [1] "Accessories"     "Computers"       "Game Cartridges" "Game Consoles"
```

```
## [5] "Hi-Fi"           "Software"        "TV & Video"
```

```
# Número de vendas por categorias
baskets.cat <- count(dataPurchases, c("dataPurchases$CategoryDescription"))
baskets.cat <- baskets.cat[order(-baskets.cat$freq), ]
```

```
## Warning: Unknown or uninitialised column: 'freq'.
```

```
## Error in -baskets.cat$freq: invalid argument to unary operator
```

```
colnames(baskets.cat) <- c("category", "freq")

length(unique(baskets.cat$category))       # 7 categorias
```

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

```
# Número médio de itens por basket
summary(baskets.cat$freq)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  393381  393381  393381  393381  393381  393381
```

```
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
basket <- as(split(as.vector(dataPurchases$SubCategoryDescription), as.vector(dataPurchases$CardID)), "
```

```
## Warning in asMethod(object): removing duplicated items in transactions
```

```
class(basket)
```

```
## [1] "transactions"
## attr(,"package")
## [1] "arules"
```

```
summary(basket)
```

```
## transactions as itemMatrix in sparse format with
##  60519 rows (elements/itemsets/transactions) and
##  47 columns (items) and a density of 0.08919232
##
## most frequent items:
##              Personal             CD Player Portable Music Centre
##                 20731                 16191                 15661
##        Cassette Tapes            Videotapes               (Other)
##                 13358                 12819                174938
##
## element (itemset/transaction) length distribution:
## sizes
```

```
##     1     2     3     4     5     6     7     8     9    10    11    12
##  8837 14902 14455  9122  3124   938   537   566   558   772   903  1090
##    13    14    15    16    17    18    19    20    21    22
##  1173  1139   960   692   428   208    86    19     7     3
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   4.192   4.000  22.000
##
## includes extended item information - examples:
##           labels
## 1 32-Bit Systems
## 2 64-Bit Systems
## 3         Action
##
## includes extended transaction information - examples:
##   transactionID
## 1   C0100000111
## 2   C0100000199
## 3   C0100000343
```

```
dim(basket)
```

```
## [1] 60519    47
```

```
basket@itemInfo   # gives all the items of the basket
```

```
##                      labels
## 1            32-Bit Systems
## 2            64-Bit Systems
## 3                    Action
## 4                   Adaptor
## 5                 Amplifier
## 6            Arcade Classics
## 7                 Batteries
## 8                     Cable
## 9                 Camcorder
## 10 Cassette Player-Recorder
## 11           Cassette Tapes
## 12                CD Player
## 13              Cleaner Kit
## 14          Combined TV/VCR
## 15         Desktop Computer
## 16                 Disk Box
## 17               DVD Player
## 18              Educational
## 19                 Fighting
## 20         Flight Simulator
## 21             Floppy Disks
## 22                Hand held
## 23                  Headset
## 24            Home Software
## 25          Laptop Computer
## 26            Mini Speakers
## 27                    Mouse
## 28                Mouse Pad
## 29             Music Centre
```

```
## 30          Office Software
## 31                  PC Game
## 32                 Personal
## 33    Portable Music Centre
## 34           Portable Radio
## 35                  Printer
## 36        Printer Cartridge
## 37                  Puzzles
## 38                   Racing
## 39           Screen Cleaner
## 40              Shoot-em-up
## 41                   Sports
## 42                    Stand
## 43           Television set
## 44                    Tuner
## 45                Turntable
## 46                      VCR
## 47               Videotapes
```

```
#View the first five transactions
inspect(basket[1:5])
```

```
##     items                   transactionID
## [1] {Disk Box,
##      PC Game,
##      Personal,
##      Printer,
##      Tuner,
##      VCR}                    C0100000111
## [2] {Portable Music Centre,
##      VCR}                    C0100000199
## [3] {Personal,
##      Portable Music Centre,
##      Printer,
##      Shoot-em-up,
##      Turntable}              C0100000343
## [4] {Cable,
##      Home Software,
##      Mouse,
##      Portable Radio}         C0100000375
## [5] {Action,
##      Batteries,
##      Cable,
##      Cassette Tapes,
##      Cleaner Kit,
##      DVD Player,
##      Educational,
##      Mouse Pad,
##      Personal,
##      Portable Radio,
##      Sports,
##      Videotapes}             C0100000392
```

```
# Occurrences of each item - Support
itemFreq <- itemFrequency(basket)
```

```r
sort(itemFreq, decreasing = T)[1:3]
```

```
##              Personal             CD Player Portable Music Centre
##             0.3425536             0.2675358             0.2587782
```

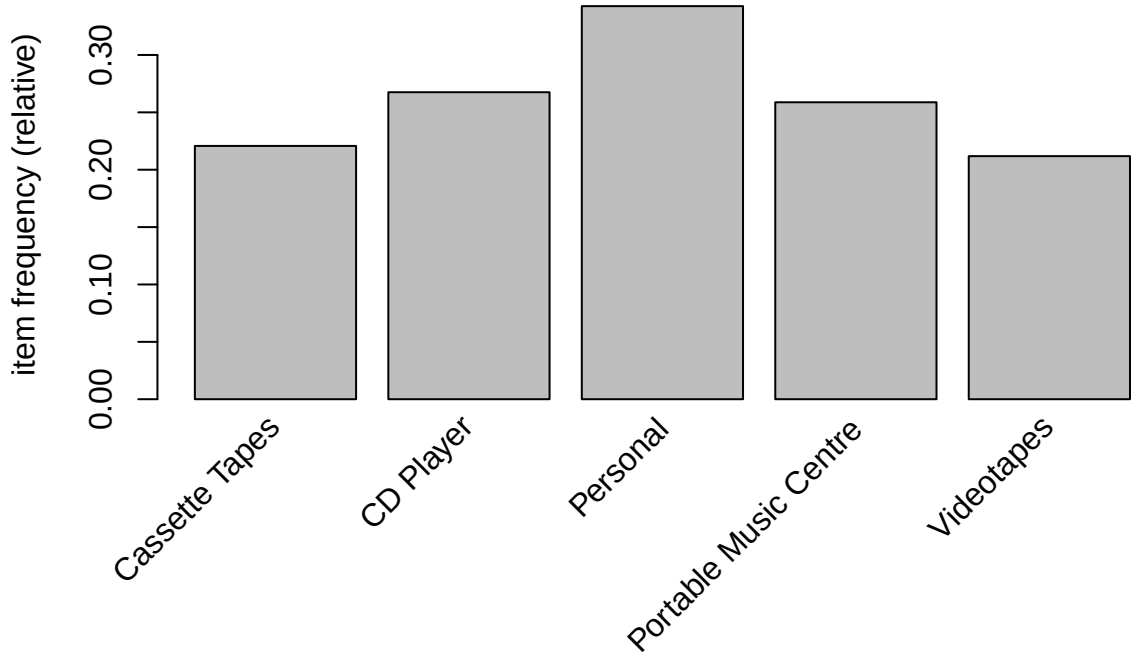
```r
summary(itemFreq)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.003239 0.023786 0.050083 0.089192 0.148805 0.342554
```

```r
#View the frequency of the first three items
itemFrequency(basket[, 1:3])
```

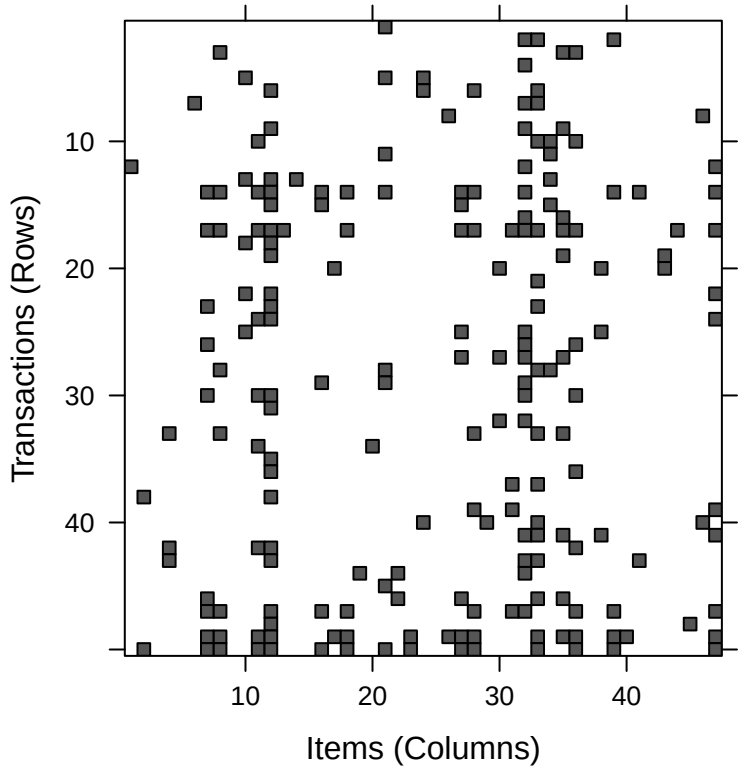
```
## 32-Bit Systems 64-Bit Systems         Action
##     0.04256514     0.05008344     0.02260447
```

```r
#Shows in a histogram plot items with at least s support
with(s <- 0.20,
  itemFrequencyPlot(basket, support = s)
)
```



Visualização da matriz de produtos comprados e respetiva dispersão.

```r
#image(basket[1:50])
image(sample(basket, 50)) # 50 linhas
```



Algoritmo Apriori para extração de Regras de Associação

Sup min = 5% e Conf min = 80%

```
sup.min = 0.05
conf.min = 0.80

basketRules <- apriori(basket, parameter = list(support = sup.min, confidence = conf.min, minlen = 2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5    0.05      2
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 3025
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[47 item(s), 60519 transaction(s)] done [0.02s].
## sorting and recoding items ... [24 item(s)] done [0.00s].
```

```
## creating transaction tree ... done [0.03s].
## checking subsets of size 1 2 3 4 5 6 7 done [0.03s].
## writing ... [919 rule(s)] done [0.00s].
## creating S4 object  ... done [0.01s].
```

```
summary(basketRules)
```

```
## set of 919 rules
##
## rule length distribution (lhs + rhs):sizes
##   2   3   4   5   6   7
##   6 214 360 241  86  12
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   4.000   4.000   4.243   5.000   7.000
##
## summary of quality measures:
##     support          confidence          lift           count
##  Min.   :0.05002   Min.   :0.8006   Min.   :3.631   Min.   :3027
##  1st Qu.:0.05479   1st Qu.:0.8717   1st Qu.:4.443   1st Qu.:3316
##  Median :0.06094   Median :0.9083   Median :4.647   Median :3688
##  Mean   :0.06586   Mean   :0.9111   Mean   :4.988   Mean   :3986
##  3rd Qu.:0.07072   3rd Qu.:0.9630   3rd Qu.:5.634   3rd Qu.:4280
##  Max.   :0.12221   Max.   :0.9885   Max.   :6.213   Max.   :7396
##
## mining info:
##    data ntransactions support confidence
##  basket         60519    0.05        0.8
```

```
measures <- interestMeasure(basketRules, measure = c("coverage", "leverage", "conviction"), transactions

summary(measures)
```

```
##     coverage          leverage         conviction
##  Min.   :0.05076   Min.   :0.03859   Min.   : 3.925
##  1st Qu.:0.05985   1st Qu.:0.04346   1st Qu.: 6.371
##  Median :0.06682   Median :0.04867   Median : 9.174
##  Mean   :0.07266   Mean   :0.05241   Mean   :15.584
##  3rd Qu.:0.07909   3rd Qu.:0.05664   3rd Qu.:21.132
##  Max.   :0.14843   Max.   :0.09633   Max.   :68.572
```

```
# Top rules by lift
inspect(head(basketRules, n = 5, by = "lift"))
```

```
##     lhs                 rhs        support confidence     lift count
## [1] {Batteries,
##      Cassette Tapes,
##      Disk Box,
##      Floppy Disks,
##      Mouse Pad,
##      Videotapes}     => {Cable} 0.05684165  0.9222520 6.213266  3440
## [2] {Batteries,
##      Disk Box,
##      Floppy Disks,
##      Mouse Pad,
##      Videotapes}     => {Cable} 0.05784960  0.9196217 6.195546  3501
```

```
## [3] {Batteries,
##      Cassette Tapes,
##      Disk Box,
##      Mouse,
##      Mouse Pad,
##      Videotapes}     => {Cable} 0.05109139  0.9194172 6.194168  3092
## [4] {Batteries,
##      Cassette Tapes,
##      Disk Box,
##      Floppy Disks,
##      Mouse Pad}      => {Cable} 0.05750260  0.9182058 6.186007  3480
## [5] {Batteries,
##      Cassette Tapes,
##      Disk Box,
##      Mouse,
##      Mouse Pad}      => {Cable} 0.05176887  0.9163498 6.173503  3133
```

```
library(arulesViz)
```

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

```
basketRules2 <- apriori(basket, parameter = list(support = 0.01, confidence = 0.05, minlen = 2, maxlen =
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5    0.01      2
##  maxlen target   ext
##      20  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 605
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[47 item(s), 60519 transaction(s)] done [0.02s].
## sorting and recoding items ... [45 item(s)] done [0.00s].
## creating transaction tree ... done [0.03s].
## checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.21s].
## writing ... [84749 rule(s)] done [0.01s].
## creating S4 object  ... done [0.04s].
```

```
summary(basketRules2)
```

```
## set of 84749 rules
##
## rule length distribution (lhs + rhs):sizes
##     2     3     4     5     6     7     8     9    10
##   576  3174 10104 19445 23712 17815  7872  1881   170
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   5.000   6.000   5.863   7.000  10.000
##
```

```
## summary of quality measures:
##     support           confidence          lift            count
##  Min.   :0.01001   Min.   :0.05021   Min.   :0.7267   Min.   : 606
##  1st Qu.:0.01148   1st Qu.:0.51873   1st Qu.:4.3112   1st Qu.: 695
##  Median :0.01413   Median :0.83846   Median :4.6837   Median : 855
##  Mean   :0.01734   Mean   :0.73614   Mean   :4.6556   Mean   :1049
##  3rd Qu.:0.01928   3rd Qu.:0.94124   3rd Qu.:5.8402   3rd Qu.:1167
##  Max.   :0.14377   Max.   :1.00000   Max.   :8.0311   Max.   :8701
##
## mining info:
##    data ntransactions support confidence
##  basket         60519    0.01       0.05
```