Ezra-_lab_9.R

Student

2021-08-02

```
library(arules)
```

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

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

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

```
#library(arulesViz) need to tag this for it to knit
data (Groceries) # Load data into memory
myGroc <- Groceries # Make a copy for safety
summary(myGroc) # What is the structure?
```

```
## transactions as itemMatrix in sparse format with
##  9835 rows (elements/itemsets/transactions) and
##  169 columns (items) and a density of 0.02609146
##
## most frequent items:
##       whole milk other vegetables       rolls/buns             soda
##             2513             1903             1809             1715
##           yogurt          (Other)
##             1372            34055
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46
##   17   18   19   20   21   22   23   24   26   27   28   29   32
##   29   14   14    9   11    4    6    1    1    1    1    3    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   4.409   6.000  32.000
##
## includes extended item information - examples:
##       labels  level2           level1
## 1 frankfurter sausage meat and sausage
## 2     sausage sausage meat and sausage
## 3  liver loaf sausage meat and sausage
```

```
#task1
#It is a sparse Matrix full of different items that people can buy at a grocery store, it has frequently bought i
tems and how many times they pop up in the data set And it also gives the information about the minimum maximum f
irst and third quartiles and the median for how many items each person bought
#task2
itemFrequency-itemFrequency(myGroc)
str(itemFrequency)
```

```
##  Named num [1:169] 0.05897 0.09395 0.00508 0.02603 0.02583 ...
##  - attr(*, "names")= chr [1:169] "frankfurter" "sausage" "liver loaf" "ham" ...
```

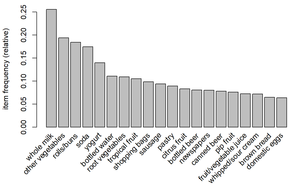
```
#2 rows, first is a list of numbers all between 0 and 1 and the second is the item that each number corresponds t
o
soreditems<-sort(itemFrequency)
head(soreditems)
```

```
##            baby food  sound storage medium preservation products
##         0.0001016777         0.0001016777         0.0002033554
##       kitchen utensil             bags        frozen chicken
##         0.0004067107         0.0004067107         0.0006100661
```

```
tail(soreditems)
```

```
##    bottled water           yogurt             soda       rolls/buns
##        0.1105236        0.1395018        0.1743772        0.1839349
## other vegetables       whole milk
##        0.1934926        0.2555100
```

```
#The most frequently purchased item is whole milk
#task2
itemFrequencyPlot(myGroc, topN=20)
```




```
#The plot confirms that whole milk is the most frequently purchased item, and the y-axis represents the percent o
f people who bought the item, with that number being the number from zero to one that was the first row provided
 when we made mygroc
#task4
ct <- crossTable(myGroc, sort=TRUE)
ct[1:2, 1:3]
```

```
##                   whole milk other vegetables rolls/buns
## whole milk              2513              736      557
## other vegetables         736             1903      419
```

```
#The first diagonal going from top left to bottom middle chose the total number of people who bought each item si
nce whole milk corresponds to whole milk and other vegetables corresponds to other vegetables, the other diagonal
from bottom left to top middle shows the amount of people who bought both items, the right most column isn't on a
diagonal and it just shows how many people who bought whole milk or other vegetables also bought bread/rolls, if
 we included a third row with buns/rolls then it would be on a diagonal
#task5
rules1 <- apriori(myGroc,
                  parameter=list(supp=0.0008, conf=0.55),
                  control=list(verbose=F),
                  appearance=list(default="lhs",rhs=("bottled beer")))
#task6
inspect(rules1)
```

```
##     lhs                            rhs            support     confidence
## [1] {liquor,red/blush wine}     => {bottled beer} 0.0019318760 0.9047619
## [2] {soda,liquor}               => {bottled beer} 0.0012201322 0.5714286
## [3] {red/blush wine,napkins}    => {bottled beer} 0.0008134215 0.5714286
## [4] {soda,liquor,red/blush wine} => {bottled beer} 0.0008134215 1.0000000
##     coverage    lift     count
## [1] 0.0021352313 11.23527 19
## [2] 0.0021352313  7.09596 12
## [3] 0.0014234875  7.09596  8
## [4] 0.0008134215 12.41793  8
```

```
#My interpretation of these rules is that people often buy alcohol for a party or event and the reason it made th
ese 4 rules it's because of other drinks you might want to get for an event and napkins which is a common item yo
u would I want if you are hosting an event
#task7
rules2 <- apriori(myGroc,
                  parameter=list(supp=0.0005, conf=0.55),
                  control=list(verbose=F),
                  appearance=list(default="lhs",rhs=("bottled beer")))
inspect(rules2)
```

```
##     lhs                             rhs                support confidence      coverage     lift count
## [1] {liquor (appetizer),
##      dishes}                     => {bottled beer} 0.0006100661  0.8571429 0.0007117438 10.643939     6
## [2] {liquor,
##      red/blush wine}             => {bottled beer} 0.0019318760  0.9047619 0.0021352313 11.235269    19
## [3] {soda,
##      liquor}                     => {bottled beer} 0.0012201322  0.5714286 0.0021352313  7.095960    12
## [4] {red/blush wine,
##      napkins}                    => {bottled beer} 0.0008134215  0.5714286 0.0014234875  7.095960     8
## [5] {soda,
##      liquor,
##      red/blush wine}             => {bottled beer} 0.0008134215  1.0000000 0.0008134215 12.417929     8
## [6] {whole milk,
##      soups,
##      bottled water}              => {bottled beer} 0.0005083884  0.8333333 0.0006100661 10.348274     5
## [7] {yogurt,
##      pastry,
##      flower (seeds)}             => {bottled beer} 0.0005083884  0.8333333 0.0006100661 10.348274     5
## [8] {whole milk,
##      yogurt,
##      flower (seeds)}             => {bottled beer} 0.0005083884  0.7142857 0.0007117438  8.869949     5
## [9] {other vegetables,
##      salt,
##      margarine}                  => {bottled beer} 0.0005083884  0.7142857 0.0007117438  8.869949     5
## [10] {soda,
##       red/blush wine,
##       napkins}                   => {bottled beer} 0.0005083884  0.8333333 0.0006100661 10.348274     5
## [11] {citrus fruit,
##       oil,
##       bottled water}             => {bottled beer} 0.0005083884  0.5555556 0.0009150991  6.898850     5
## [12] {root vegetables,
##       herbs,
##       other vegetables,
##       bottled water}             => {bottled beer} 0.0006100661  0.6000000 0.0010167768  7.450758     6
## [13] {whole milk,
##       butter,
##       rolls/buns,
##       napkins}                   => {bottled beer} 0.0005083884  0.5555556 0.0009150991  6.898850     5
## [14] {pork,
##       whole milk,
##       domestic eggs,
##       rolls/buns}                => {bottled beer} 0.0005083884  0.5555556 0.0009150991  6.898850     5
```

```
#I would say that mostly it still supports the idea that there is an event that they are buying this for, most of
 the rules seem like they're also getting ingredients to make dishes for an event, although a couple Of the new ru
les seem to indicate that some people are buying it just as an ordinary drink as they're buying it with stuff lik
e sunflower seeds or whole milk something that you wouldn't really see at an event
#task8
cars<-mtcars
goodorbad<-(7)
for(i in 1:nrow(cars)){
  if (cars$mpg[i]>25) goodorbad<-c(goodorbad,1)
  else goodorbad<-c(goodorbad,0)
}
cars$goodorbad<-goodorbad[-match(7,goodorbad)]
cars$mpg<-as.factor(cars$mpg)
cars$cyl<-as.factor(cars$cyl)
cars$disp<-as.factor(cars$disp)
cars$hp<-as.factor(cars$hp)
cars$drat<-as.factor(cars$drat)
cars$wt<-as.factor(cars$wt)
cars$qsec<-as.factor(cars$qsec)
cars$vs<-as.factor(cars$vs)
cars$am<-as.factor(cars$am)
cars$gear<-as.factor(cars$gear)
cars$carb<-as.factor(cars$carb)
cars$goodorbad<-as.factor(cars$goodorbad)
cars_t<-as(cars,"transactions")
summary(cars_t)
```

```
## transactions as itemMatrix in sparse format with
##  32 rows (elements/itemsets/transactions) and
##  173 columns (items) and a density of 0.06936416
##
## most frequent items:
## goodorbad=0         am=0        vs=0       gear=3       cyl=8     (Other)
##          26           19           18           15           14          202
##
## element (itemset/transaction) length distribution:
## sizes
## 12
## 32
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      12      12      12      12      12      12
##
## includes extended item information - examples:
##     labels variables levels
## 1 mpg=10.4      mpg  10.4
## 2 mpg=13.3      mpg  13.3
## 3 mpg=14.3      mpg  14.3
##
## includes extended transaction information - examples:
##   transactionID
## 1     Mazda RX4
## 2 Mazda RX4 Wag
## 3    Datsun 710
```

```
rules <- apriori(cars_t, parameter = list(supp=0.4, conf=0.55),control=list(verbose=F))
subrules <- subset(rules, subset = rhs %in% c("goodorbad=0"))
inspect(subrules)
```

```
##     lhs                rhs                  support confidence coverage lift     count
## [1] {}              => {goodorbad=0} 0.81250 0.8125000  1.00000  1.000000 26
## [2] {cyl=8}         => {goodorbad=0} 0.43750 1.0000000  0.43750  1.230769 14
## [3] {gear=3}        => {goodorbad=0} 0.46875 1.0000000  0.46875  1.230769 15
## [4] {vs=0}          => {goodorbad=0} 0.53125 0.9444444  0.56250  1.162393 17
## [5] {am=0}          => {goodorbad=0} 0.59375 1.0000000  0.59375  1.230769 19
## [6] {cyl=8,vs=0}    => {goodorbad=0} 0.43750 1.0000000  0.43750  1.230769 14
## [7] {am=0,gear=3}   => {goodorbad=0} 0.46875 1.0000000  0.46875  1.230769 15
```

```
#https://stackoverflow.com/questions/18131792/creating-specific-rules-with-arules-in-r
#I was having a lot of trouble with this part, so I used to Google and this was the link that eventually gave me
 the information I needed to get this part done
```