Association Rules Mining

Blake Pappas

2023-12-17

Association Rule Mining in R

Load the arules package

```r
# install.packages("arules")

library(arules)
```

In this first exercise, we use the "supermarket.csv" file.

This dataset contains 8 shopping baskets.

P1: Import this dataset as transaction data

Think about parameters including format, sep, and rm.duplicates.

```r
supermarket = read.transactions("supermarket.csv",
                                format = "basket",
                                sep = ",",
                                rm.duplicates = TRUE)
```

P2: Understand the supermarket data

Which unique items are there in all shopping baskets?

```r
itemInfo(supermarket)
```

```
##       labels
## 1      Bread
```

```
## 2    Butter
## 3    Cereal
## 4    Cheese
## 5 Ice Cream
## 6     Juice
## 7      Milk
```

P3: Understand the supermarket data.

How many transactions contain purchase of Butter?

Answer: 2 transactions contain purchase of Butter.

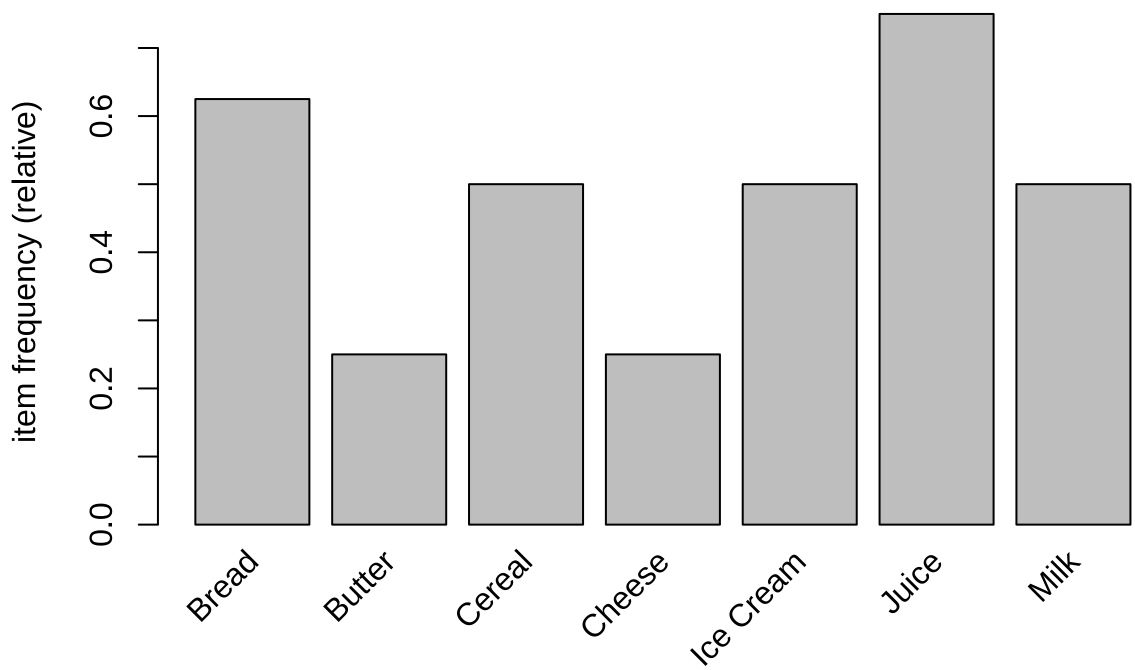
```r
itemFrequency(supermarket, type = "absolute")
```

```
##     Bread    Butter    Cereal    Cheese Ice Cream     Juice      Milk
##         5         2         4         2         4         6         4
```

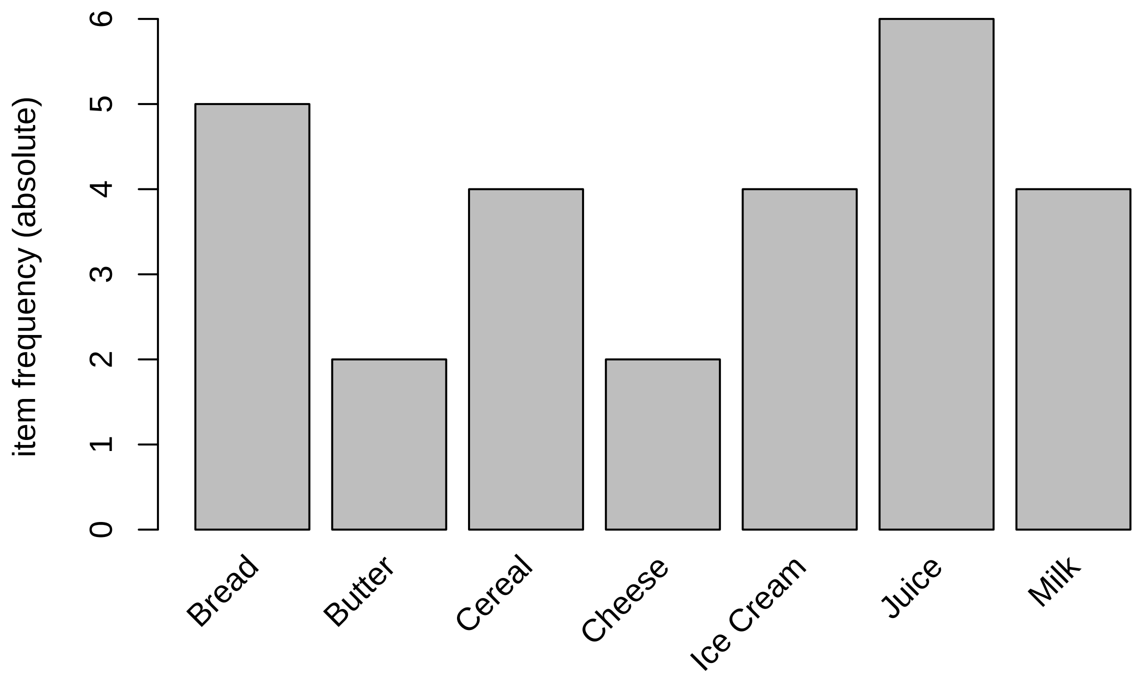
P4: Understand the supermarket data

Plot the frequency of each item

```r
itemFrequencyPlot(supermarket)
```



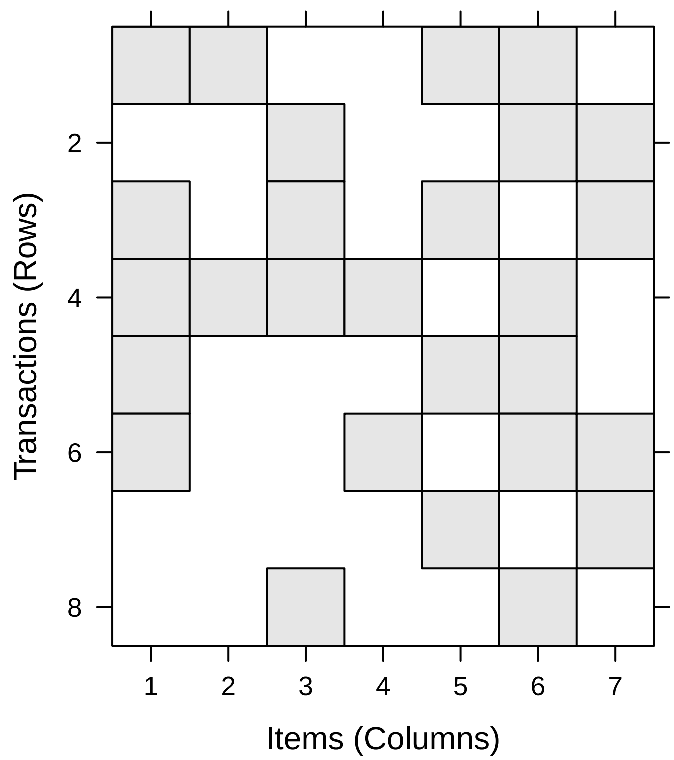

```r
itemFrequencyPlot(supermarket, type = "absolute")
```



P5: Understand the supermarket data

Visualize the entire dataset, showing which items show up in which transactions.

```
image(supermarket)
```



P6: Mine association rules

Find all association rules with minsupp = 0.375 and minconf = 0.65.

```r
rules = apriori(supermarket,
                parameter = list(supp = 0.375, conf = 0.65))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.65    0.1    1 none FALSE            TRUE       5   0.375      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 3
```

```
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[7 item(s), 8 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [5 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
inspect(rules)
```

```
##     lhs            rhs     support confidence coverage lift     count
## [1] {}          => {Juice} 0.750   0.7500000  1.000    1.000000 6
## [2] {Ice Cream} => {Bread} 0.375   0.7500000  0.500    1.200000 3
## [3] {Cereal}    => {Juice} 0.375   0.7500000  0.500    1.000000 3
## [4] {Bread}     => {Juice} 0.500   0.8000000  0.625    1.066667 4
## [5] {Juice}     => {Bread} 0.500   0.6666667  0.750    1.066667 4
```

P7: Mine association rules

Inspect the found rules, in the order of decreasing lift ratio.

```r
inspect(sort(rules, by = "lift"))
```

```
##     lhs            rhs     support confidence coverage lift     count
## [1] {Ice Cream} => {Bread} 0.375   0.7500000  0.500    1.200000 3
## [2] {Bread}     => {Juice} 0.500   0.8000000  0.625    1.066667 4
## [3] {Juice}     => {Bread} 0.500   0.6666667  0.750    1.066667 4
## [4] {}          => {Juice} 0.750   0.7500000  1.000    1.000000 6
## [5] {Cereal}    => {Juice} 0.375   0.7500000  0.500    1.000000 3
```

In the second exercise, we use the “book.csv” file.

This dataset contains 2000 book purchases in binary matrix format.

P1: Import this dataset as transaction data

Think about the three steps of importing.

```r
book_data_frame = read.csv("book.csv")
book_matrix = as.matrix(book_data_frame)
book = as(book_matrix, "transactions")
```

P2: Understand the book data

Plot the frequency of each book category, in absolute sales.

Which book category sells best?

Answer: The CookBks category sells the best.

```
itemFrequency(book, type = "absolute")
```

```
##   ChildBks  YouthBks   CookBks  DoItYBks    RefBks    ArtBks   GeogBks  ItalCook
##        846       495       862       564       429       482       552       227
## ItalAtlas   ItalArt  Florence
##         74        97       217
```

P3: Mine association rules

Find all association rules with minsupp = 0.1 and minconf = 0.8.

```
rules = apriori(book,
                parameter = list(supp = 0.1, conf = 0.8))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5     0.1      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 200
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[11 item(s), 2000 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [7 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
inspect(sort(rules, by = "lift"))
```

```
##     lhs                    rhs        support confidence coverage lift
## [1] {ItalCook}          => {CookBks}  0.1135  1.0000000  0.1135   2.320186
## [2] {DoItYBks, ArtBks}  => {CookBks}  0.1015  0.8218623  0.1235   1.906873
## [3] {DoItYBks, GeogBks} => {CookBks}  0.1085  0.8188679  0.1325   1.899926
## [4] {CookBks, RefBks}   => {ChildBks} 0.1225  0.8032787  0.1525   1.899004
## [5] {ArtBks, GeogBks}   => {ChildBks} 0.1020  0.8000000  0.1275   1.891253
## [6] {ArtBks, GeogBks}   => {CookBks}  0.1035  0.8117647  0.1275   1.883445
## [7] {ChildBks, RefBks}  => {CookBks}  0.1225  0.8085809  0.1515   1.876058
##     count
## [1] 227
## [2] 203
## [3] 217
## [4] 245
## [5] 204
## [6] 207
## [7] 245
```

P4: Understand the found rules

Inspect the rules, and answer the following questions:

Which rule has the highest lift? What does it tell us?

Answer: The {ItalCook} -> {CookBks} rule has the highest lift. This tells us that customers who buy ItalCook are 1.320186x (132.0186%) more likely to buy CookBks than customers in general.

What can be done with this rule, if you were the bookstore manager?

Answer: If I were the bookstore manager, I could use this rule to situate ItalCook closer to CookBks in my store.