R Notebook

Bheeni Garg

Association Rules ————-

Identifying Frequently-Purchased Groceries —-

Step 1: Collecting Data

The market basket analysis utilizes purchase data from one month of operation at a real-world grocery store. The data contain 9,835 transactions, or about 327 transactions per day (roughly 30 transactions per hour in a 12 hour business day), suggesting that the retailer is not particularly large, nor is it particularly small.

Step 2: Exploring and preparing the data —-

```r
# load the grocery data into a sparse matrix
library(arules)
```

```
## Warning: package 'arules' was built under R version 3.2.5
```

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

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

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

```r
groceries <- read.transactions("groceries.csv", sep = ",")
summary(groceries)
```

```
## 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
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55
##   16   17   18   19   20   21   22   23   24   26   27   28   29   32
##   46   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
## 1 abrasive cleaner
## 2 artif. sweetener
## 3   baby cosmetics
```

The output shows that there are a total of 9835 transactions with the maximum number of 169 items in a single transaction.

```r
# look at the first five transactions
inspect(groceries[1:5])
```

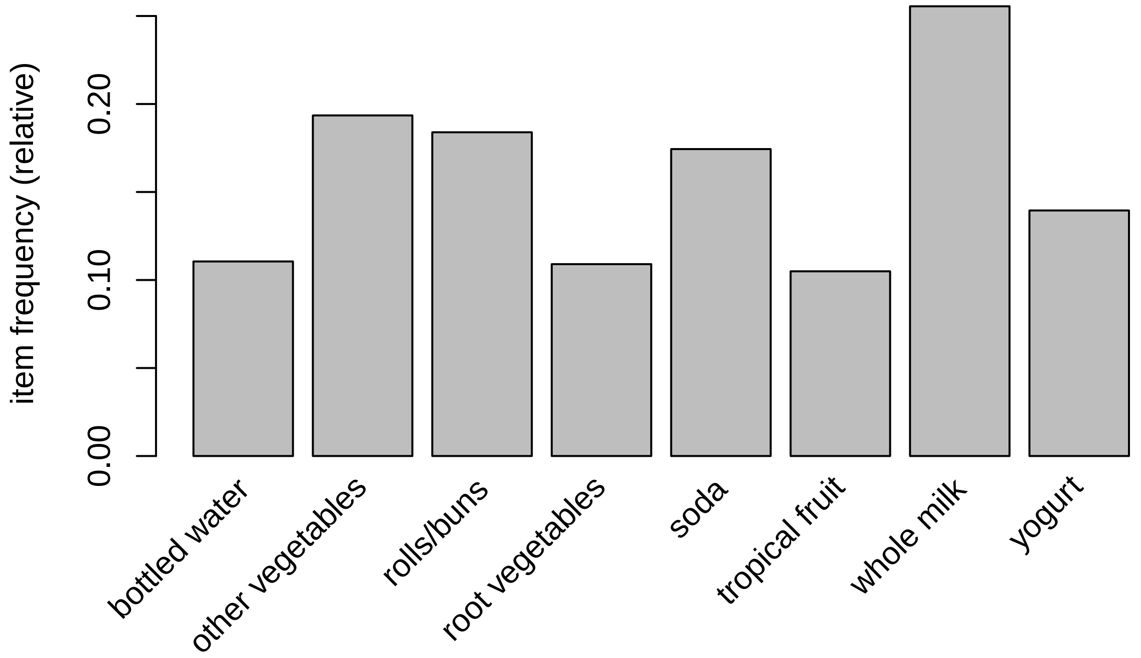
```
##   items
## 1 {citrus fruit,
##    margarine,
##    ready soups,
##    semi-finished bread}
## 2 {coffee,
##    tropical fruit,
##    yogurt}
## 3 {whole milk}
## 4 {cream cheese,
##    meat spreads,
##    pip fruit,
##    yogurt}
## 5 {condensed milk,
##    long life bakery product,
##    other vegetables,
##    whole milk}
```

```r
# examine the frequency of items
itemFrequency(groceries[, 1:3])
```

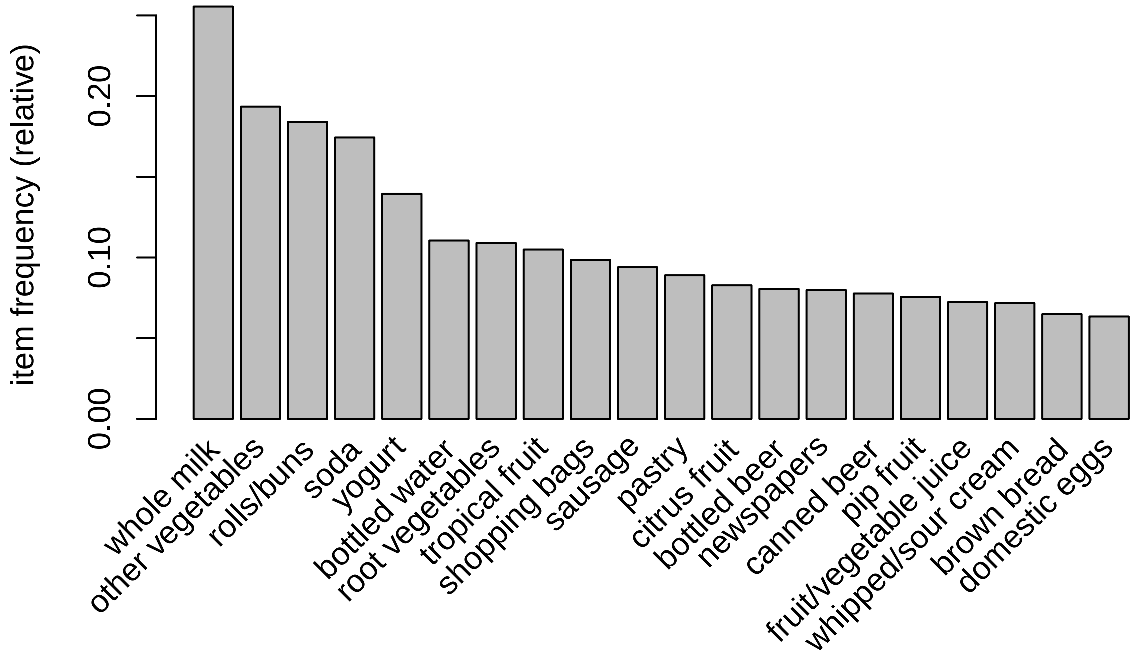
```
## abrasive cleaner artif. sweetener   baby cosmetics
##     0.0035587189     0.0032536858     0.0006100661
```

itemFrequency calculates the support of each item which is equal to count(item)/total number of transactions.

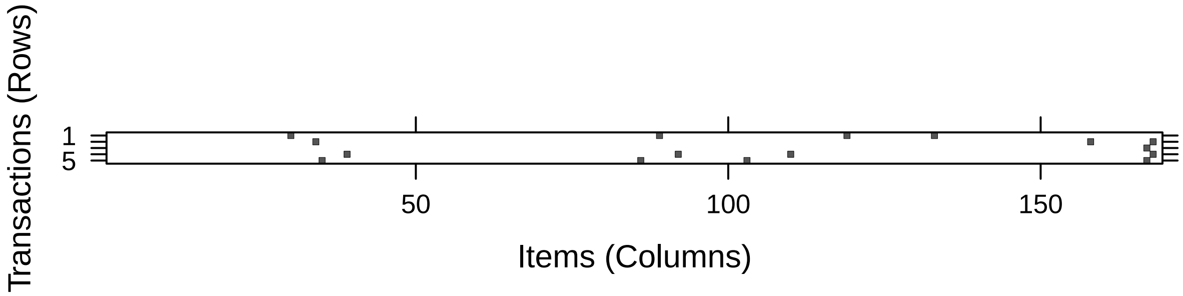
```r
# plot the frequency of items
itemFrequencyPlot(groceries, support = 0.1)
```



```r
itemFrequencyPlot(groceries, topN = 20)
```

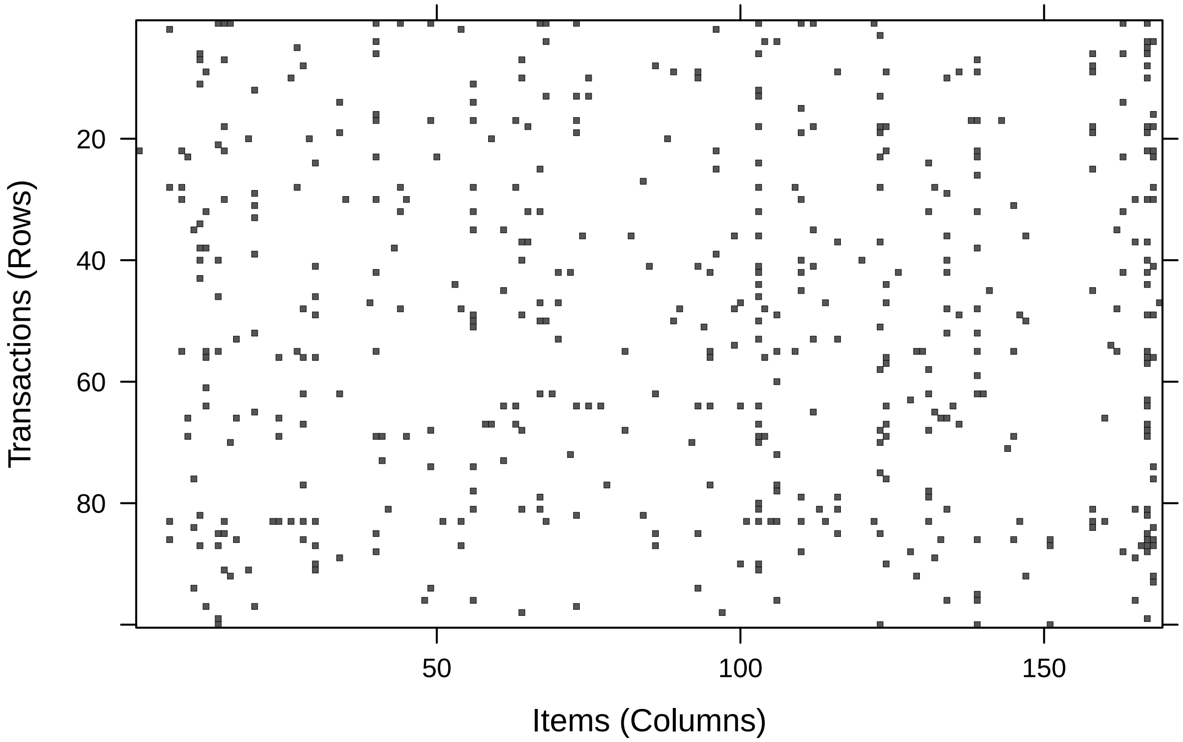


```r
# a visualization of the sparse matrix for the first five transactions
image(groceries[1:5])
```



The output to the image plot shows that the item matrix is sparse.

```
# visualization of a random sample of 100 transactions
image(sample(groceries, 100))
```



Step 3: Training a model on the data —-

```
library(arules)

# default settings result in zero rules learned
apriori(groceries)
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.8    0.1    1 none FALSE            TRUE     0.1      1     10
##  target   ext
##   rules FALSE
##
```

```
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 983
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [8 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [0 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
## set of 0 rules
```

```
# set better support and confidence levels to learn more rules
groceryrules <- apriori(groceries, parameter = list(support =
                                                      0.006, confidence = 0.25, minlen = 2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##        0.25    0.1    1 none FALSE            TRUE   0.006      2     10
##  target   ext
##   rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 59
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [109 item(s)] done [0.00s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [463 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
groceryrules
```

```
## set of 463 rules
```

We now inspect the 463 rules generated with minimum support = 0.006 and minimum confidence = 0.25.

Step 4: Evaluating model performance —-

```
# summary of grocery association rules
summary(groceryrules)
```

```
## set of 463 rules
##
## rule length distribution (lhs + rhs):sizes
##   2   3   4
## 150 297  16
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   3.000   2.711   3.000   4.000
##
## summary of quality measures:
##     support           confidence          lift
##  Min.   :0.006101   Min.   :0.2500   Min.   :0.9932
##  1st Qu.:0.007117   1st Qu.:0.2971   1st Qu.:1.6229
##  Median :0.008744   Median :0.3554   Median :1.9332
##  Mean   :0.011539   Mean   :0.3786   Mean   :2.0351
##  3rd Qu.:0.012303   3rd Qu.:0.4495   3rd Qu.:2.3565
##  Max.   :0.074835   Max.   :0.6600   Max.   :3.9565
##
## mining info:
##       data ntransactions support confidence
##  groceries          9835   0.006       0.25
```

We see that there are 150 rules generated for 2 items, 297 rules for 3 items and 16 rules for 4 items.

```
# look at the first three rules
inspect(groceryrules[1:3])
```

```
##   lhs                rhs               support     confidence lift
## 1 {potted plants} => {whole milk}      0.006914082 0.4000000  1.565460
## 2 {pasta}         => {whole milk}      0.006100661 0.4054054  1.586614
## 3 {herbs}         => {root vegetables} 0.007015760 0.4312500  3.956477
```

Step 5: Improving model performance —-

```
# sorting grocery rules by lift
inspect(sort(groceryrules, by = "lift")[1:5])
```

```
##   lhs                   rhs                      support confidence     lift
## 1 {herbs}            => {root vegetables}    0.007015760  0.4312500 3.956477
## 2 {berries}          => {whipped/sour cream} 0.009049314  0.2721713 3.796886
## 3 {other vegetables,
##    tropical fruit,
##    whole milk}       => {root vegetables}    0.007015760  0.4107143 3.768074
## 4 {beef,
##    other vegetables} => {root vegetables}    0.007930859  0.4020619 3.688692
## 5 {other vegetables,
##    tropical fruit}   => {pip fruit}          0.009456024  0.2634561 3.482649
```

```
# finding subsets of rules containing any berry items
berryrules <- subset(groceryrules, items %in% "berries")
inspect(berryrules)
```

```
##    lhs          rhs                  support     confidence lift
## 57 {berries} => {whipped/sour cream} 0.009049314 0.2721713  3.796886
## 58 {berries} => {yogurt}             0.010574479 0.3180428  2.279848
## 59 {berries} => {other vegetables}   0.010269446 0.3088685  1.596280
## 60 {berries} => {whole milk}         0.011794611 0.3547401  1.388328
```

```r
# writing the rules to a CSV file
write(groceryrules, file = "groceryrules.csv",
      sep = ",", quote = TRUE, row.names = FALSE)

# converting the rule set to a data frame
groceryrules_df <- as(groceryrules, "data.frame")
str(groceryrules_df)
```

```
## 'data.frame':    463 obs. of  4 variables:
##  $ rules     : Factor w/ 463 levels "{baking powder} => {other vegetables}",..: 340 302 207 206 208 3
##  $ support   : num  0.00691 0.0061 0.00702 0.00773 0.00773 ...
##  $ confidence: num  0.4 0.405 0.431 0.475 0.475 ...
##  $ lift      : num  1.57 1.59 3.96 2.45 1.86 ...
```