Association Rules

Mateusz Buczyński

Association rules are statements that help to find patterns in seemingly unrelated data or a relational database (information repository). Easy example of such would be: If I buy milk, there is 80% probability that I will also buy yogurt"

An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data. A consequent is an item that is found in combination with the antecedent. Association rules are created by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements have been found to be true. In data mining, association rules are useful for analyzing and predicting customer behavior. They play an important part in shopping basket data analysis, product clustering, catalog design and store layout.

```
data(Groceries)

transactions <-  Groceries
```

I will be using free Groceries dataset provided with *arules* package. It contains 9835 transactions (rows) and 169 items (columns). The aim of the analysis is to show the methods that allow to obtain certain rules within the dataset.

```
summary(transactions)
```

```
## 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
yogurt          (Other)

##             2513             1903             1809             1715
1372            34055
```

```
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
16   17   18   19   20   21   22   23   24   26   27   28   29   32
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55
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  level2           level1
## 1 frankfurter sausage meat and sausage
## 2     sausage sausage meat and sausage
## 3  liver loaf sausage meat and sausage
```

```
nrow(transactions)
```

```
## [1] 9835
```

As we can see the most commonly found items in the dataset are:

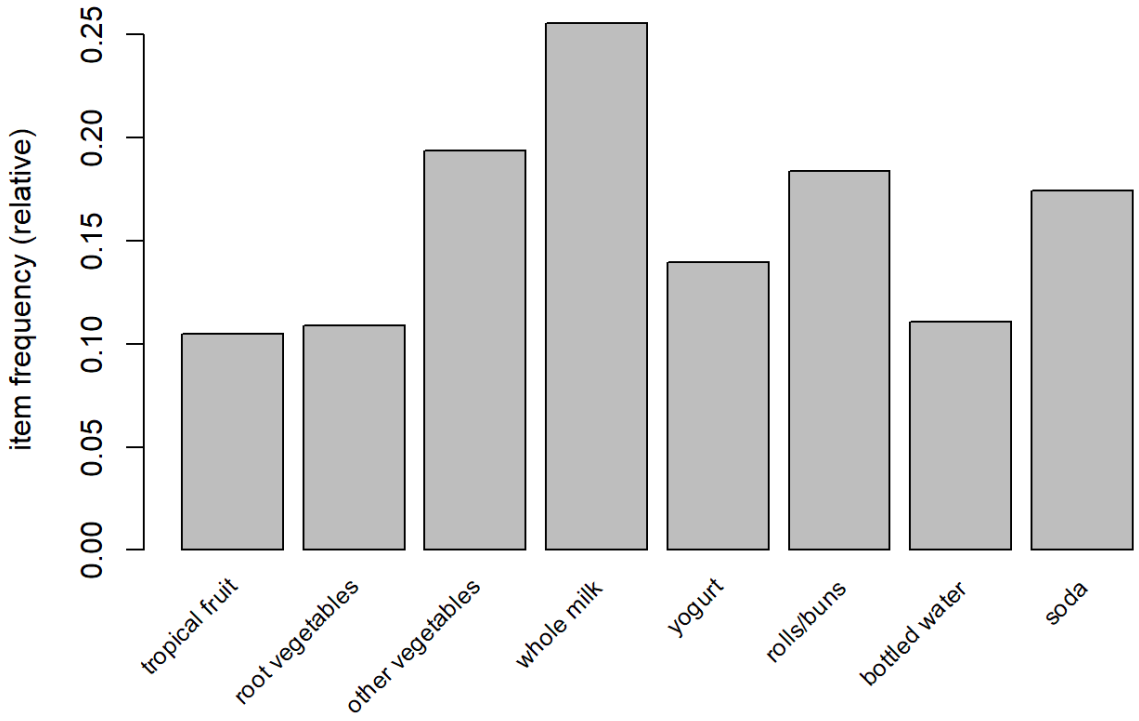
- whole milk
- other vegetables
- rolls/buns
- soda
- yogurt

Density of 0.026 means that there are 2.6% non zero cells in the matrix. Matrix has 9835 times 169 = 1662115 cells. Since 2.6% of that are non-zero cells, so 4.336710^{4} items were purchased.

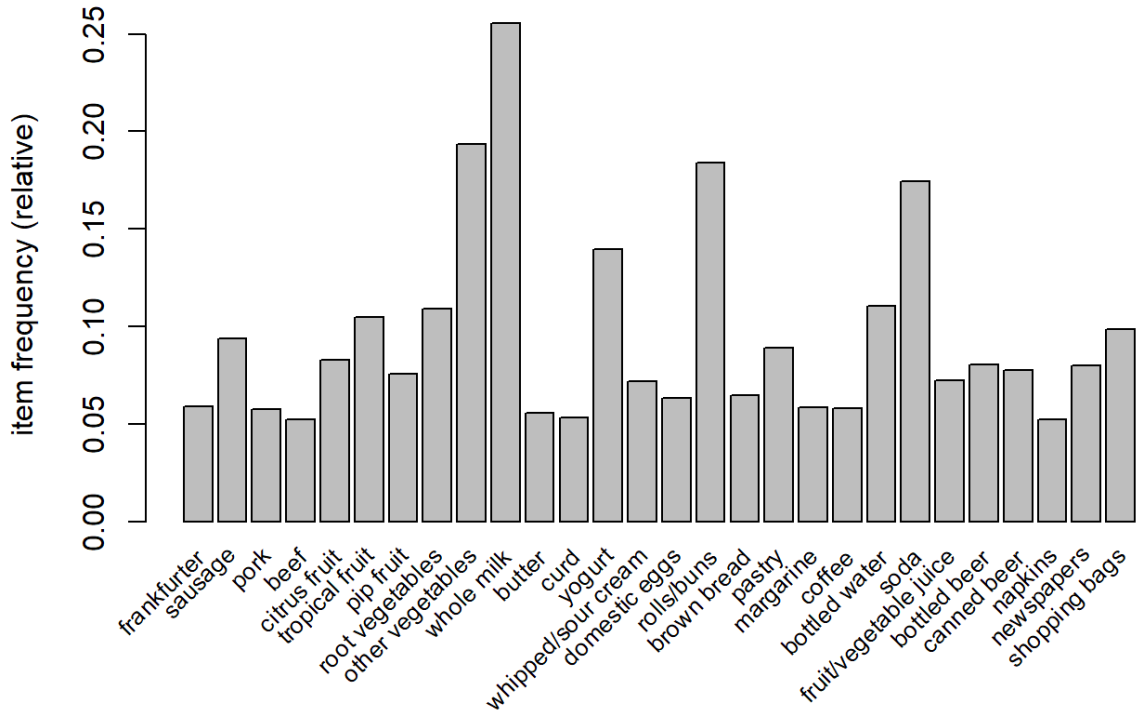
Average transaction consisted of 4.409456 items, whereas only one item have been bought in 2159 transactions. Maximum number of items bought was 32.

Let us proceed to frequency plots. The more frequent the item will be in transaction the higher its bar. Morover there are plots with different support levels. Support is the frequency of the pattern in the rule, therefore it being set to 0.1 means that the item must occur at least 10 times in 100 transactions. That is why the second plot has more items. Other way of selecting desired number of elements is to provide not support, but just the desired number. This is presented on the third graph.

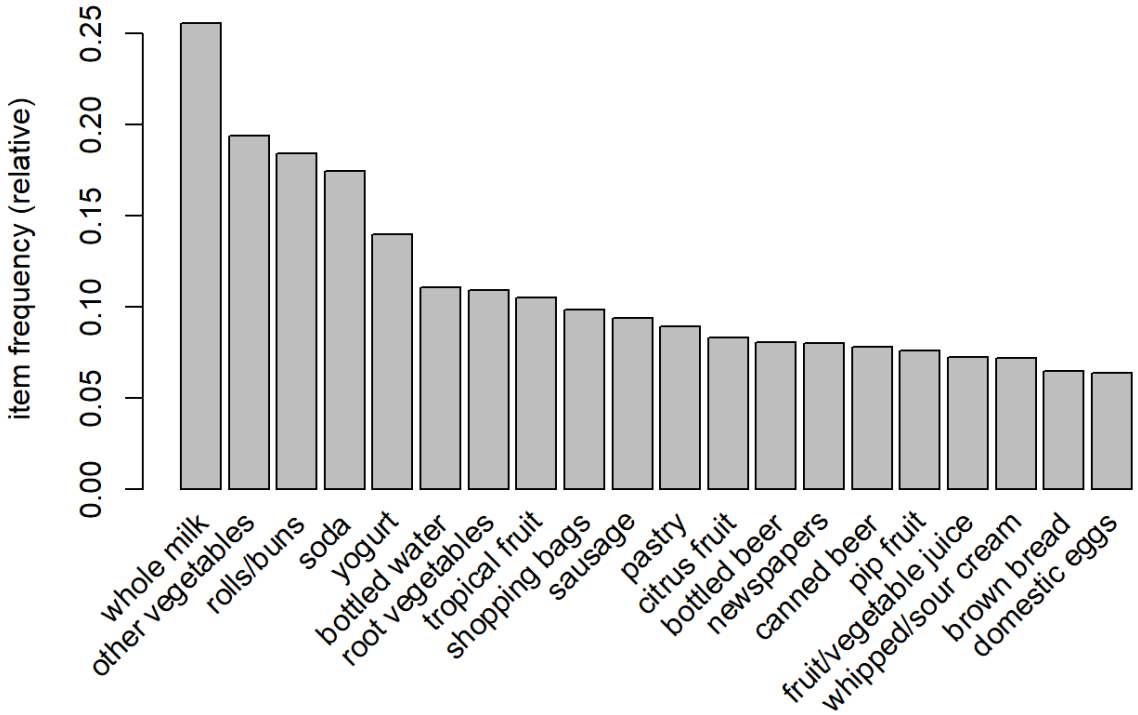
```
itemFrequencyPlot(transactions, support=0.1, cex.names=0.8)
```




```
itemFrequencyPlot(transactions, support=0.05, cex.names=0.8)
```




```
itemFrequencyPlot(transactions, topN=20)
```



On an average, each itemset or basket contains 4 to 5 items. In other words, basket having less than 5 items is more frequent as compare to baskets having more than 15 items. Buyers generally come to purchase fewer items from the shop. Support being set to .01 means that plot only includes item set having more than 1 repetition in each 100 transactions. Anything less than that is ignored for the study.

Rules

Support shows the frequency of the patterns in the rule; it is the percentage of transactions that contain both A and B, i.e. Support = Probability (A and B) Support = (# of transactions involving A and B) / (total # of transactions).

Confidence is the strength of implication of a rule; it is the percentage of transactions that contain B if they contain A, i.e. Confidence = Probability (A and B) = P(A) Confidence = (# of transactions involving A and B) / (total # of transactions that have A).

Expected confidence is the percentage of transactions that contain B to all transactions, i.e Expected confidence = Probability (B)

Correlation analysis

The lift score . Lift = 1 ??? A and B are independent . Lift > 1 ??? A and B are positively correlated . Lift < 1 ??? A and B are negatively correlated

Firstly let us try the eclat algorithm - to see most frequent itemsets. Below we will see the list of the most common items together with their individual support.

```
freq.itemsets <- eclat(transactions, parameter=list(supp=0.075, maxlen=15))
```

```
## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen            target   ext
##     FALSE   0.075      1     15 frequent itemsets FALSE
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 737
##
## create itemset ...
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [16 item(s)] done [0.00s].
## creating sparse bit matrix ... [16 row(s), 9835 column(s)] done [0.00s].
## writing  ... [16 set(s)] done [0.00s].
## Creating S4 object  ... done [0.00s].
```

```
inspect(freq.itemsets)
```

```
##      items              support    count
## [1]  {whole milk}       0.25551601 2513
## [2]  {other vegetables} 0.19349263 1903
## [3]  {rolls/buns}       0.18393493 1809
## [4]  {yogurt}           0.13950178 1372
## [5]  {soda}             0.17437722 1715
## [6]  {root vegetables}  0.10899847 1072
## [7]  {tropical fruit}   0.10493137 1032
## [8]  {bottled water}    0.11052364 1087
## [9]  {sausage}          0.09395018  924
```

```
## [10] {shopping bags}    0.09852567  969
## [11] {citrus fruit}     0.08276563  814
## [12] {pastry}           0.08896797  875
## [13] {pip fruit}        0.07564820  744
## [14] {newspapers}       0.07981698  785
## [15] {bottled beer}     0.08052872  792
## [16] {canned beer}      0.07768175  764
```

Most frequent itemsets correspond to the most frequent items (as there are no more than 2 items itemsets.)

Let us create rules then. Rules are created using apriori algorithm and giving minimal support and confidence of a rule.

```
rules <- apriori(Groceries, parameter = list(support = 0.009, confidence = 0.
25, minlen = 2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
maxlen target   ext
##        0.25    0.1    1 none FALSE            TRUE       5   0.009      2
10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 88
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [93 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [224 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Summary of rules will provide us with statistical information about support, confidence, lift and count of items.

```
summary(rules)
```

```
## set of 224 rules
##
## rule length distribution (lhs + rhs):sizes
##   2   3
## 111 113
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   3.000   2.504   3.000   3.000
##
## summary of quality measures:
##     support           confidence          lift            count
##  Min.   :0.009049   Min.   :0.2513   Min.   :0.9932   Min.   : 89.0
##  1st Qu.:0.010066   1st Qu.:0.2974   1st Qu.:1.5767   1st Qu.: 99.0
##  Median :0.012303   Median :0.3603   Median :1.8592   Median :121.0
##  Mean   :0.016111   Mean   :0.3730   Mean   :1.9402   Mean   :158.5
##  3rd Qu.:0.018480   3rd Qu.:0.4349   3rd Qu.:2.2038   3rd Qu.:181.8
##  Max.   :0.074835   Max.   :0.6389   Max.   :3.7969   Max.   :736.0
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.009       0.25
```

We obtained a set of 224 rules, where mean support is equal to 16% and mean confidence is 37%. These are not bad values. It means that mean rule occurrs in 16% transactions and its implication has 37% power.

Inspect top 5 rules

```
inspect(head(sort(rules, by ="lift"),5))
```

```
##     lhs                                rhs                    support
confidence lift     count
## [1] {berries}                       => {whipped/sour cream} 0.009049314
0.2721713  3.796886  89
```

```
## [2] {tropical fruit,other vegetables} => {pip fruit}          0.009456024
0.2634561  3.482649  93

## [3] {pip fruit,other vegetables}      => {tropical fruit}     0.009456024
0.3618677  3.448613  93

## [4] {citrus fruit,other vegetables}   => {root vegetables}    0.010371124
0.3591549  3.295045 102

## [5] {tropical fruit,other vegetables} => {root vegetables}    0.012302999
0.3427762  3.144780 121
```

Above rules (sorted by lift - preference of buying B if A was bought) can be interpreted as such:

- Anyone who buys citruses/tropical fruits is more than 3 times more likely to buy root vegetables than any other client.
- Anyone who buys beef is more than 3 times more likely to buy root vegetables than any other client.
- People like to buy berries and eat them with cream.

Let us see rules that have high support and high confidence.

```
inspect(sort(sort(rules, by ="support"),by ="confidence")[1:5])
```

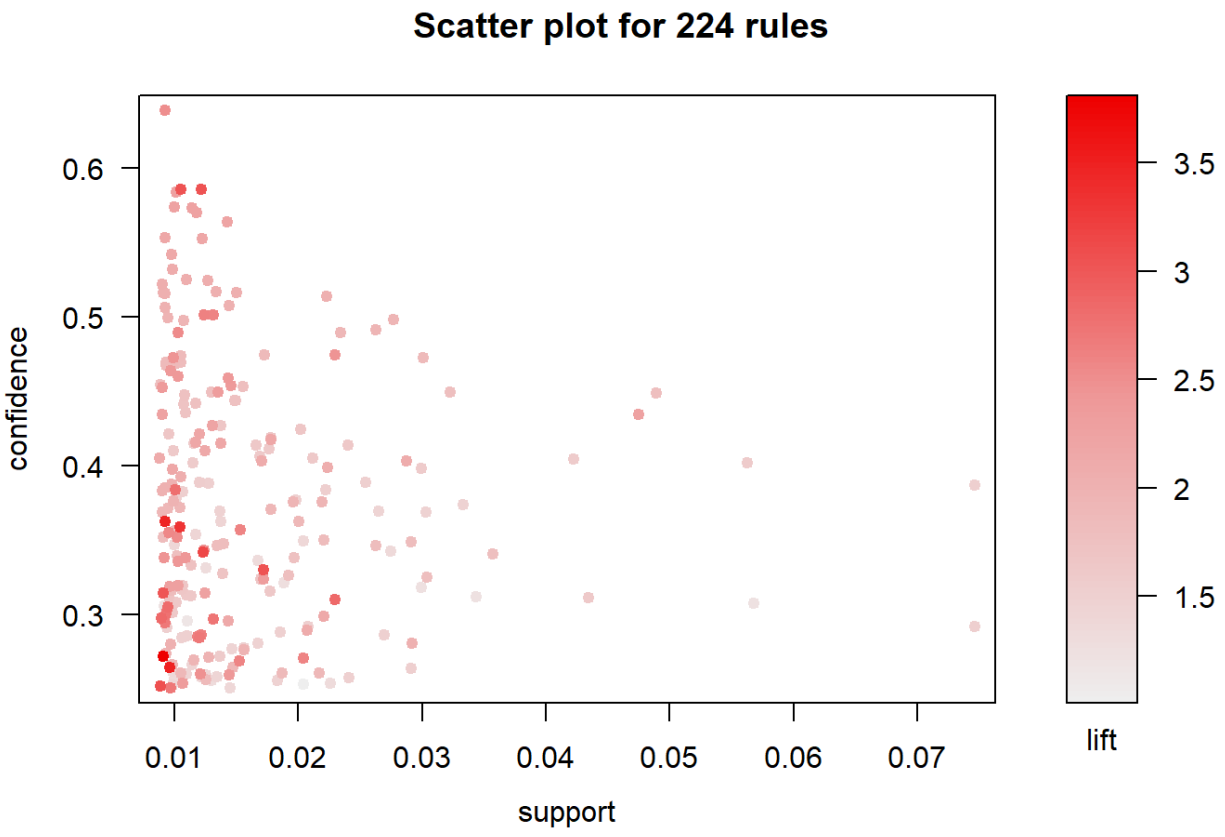
```
##     lhs                                rhs                support     con
fidence lift     count

## [1] {butter,yogurt}                 => {whole milk}       0.009354347 0.6
388889  2.500387  92

## [2] {citrus fruit,root vegetables}  => {other vegetables} 0.010371124 0.5
862069  3.029608 102

## [3] {tropical fruit,root vegetables} => {other vegetables} 0.012302999 0.5
845411  3.020999 121

## [4] {curd,yogurt}                   => {whole milk}       0.010066090 0.5
823529  2.279125  99

## [5] {other vegetables,curd}         => {whole milk}       0.009862735 0.5
739645  2.246296  97
```

There is new rule (very strong) that says that buying milk is associated with buying curd, yoghurt or butter.

Moreover we can plot the rules in support and confidence axes and colour them with lift values. Most of the rules have small values of support, but confidence varies up to 0.55. The reddier the point the more likely is the rule to happen.

We cannot find any particular patterns on a graph below.

```
plot(rules, measure=c("support", "confidence"), shading="lift", interactive=F
ALSE)
```



Induction

Below analyses depend on choosing one product and checking which products it implies or by which products it is implied.

Beverages:

```
milk.rules <- sort(subset(rules, subset = rhs %in% "whole milk"), by = "confidence")
summary(milk.rules)
```

```
## set of 85 rules
##
## rule length distribution (lhs + rhs):sizes
##  2  3
## 46 39
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   2.000   2.459   3.000   3.000
```

```
##
## summary of quality measures:
##     support           confidence          lift            count
##  Min.   :0.009049   Min.   :0.2538   Min.   :0.9932   Min.   : 89.0
##  1st Qu.:0.010269   1st Qu.:0.3845   1st Qu.:1.5047   1st Qu.:101.0
##  Median :0.013523   Median :0.4344   Median :1.7002   Median :133.0
##  Mean   :0.018057   Mean   :0.4374   Mean   :1.7116   Mean   :177.6
##  3rd Qu.:0.021251   3rd Qu.:0.4976   3rd Qu.:1.9474   3rd Qu.:209.0
##  Max.   :0.074835   Max.   :0.6389   Max.   :2.5004   Max.   :736.0
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.009       0.25
```

```
inspect(milk.rules)
```

```
##      lhs                                    rhs          support
confidence lift      count
## [1]  {butter,yogurt}                        => {whole milk} 0.009354347
0.6388889  2.5003869  92
## [2]  {curd,yogurt}                          => {whole milk} 0.010066090
0.5823529  2.2791250  99
## [3]  {other vegetables,curd}                => {whole milk} 0.009862735
0.5739645  2.2462956  97
## [4]  {other vegetables,butter}              => {whole milk} 0.011489578
0.5736041  2.2448850 113
## [5]  {tropical fruit,root vegetables}       => {whole milk} 0.011997966
0.5700483  2.2309690 118
## [6]  {root vegetables,yogurt}               => {whole milk} 0.014539908
0.5629921  2.2033536 143
## [7]  {root vegetables,whipped/sour cream}   => {whole milk} 0.009456024
0.5535714  2.1664843  93
## [8]  {other vegetables,domestic eggs}       => {whole milk} 0.012302999
0.5525114  2.1623358 121
## [9]  {other vegetables,frozen vegetables}   => {whole milk} 0.009659380
0.5428571  2.1245523  95
## [10] {pip fruit,yogurt}                     => {whole milk} 0.009557702
0.5310734  2.0784351  94
## [11] {yogurt,whipped/sour cream}            => {whole milk} 0.010879512
0.5245098  2.0527473 107
```

```
## [12] {root vegetables,rolls/buns}             => {whole milk} 0.012709710
0.5230126  2.0468876 125

## [13] {baking powder}                          => {whole milk} 0.009252669
0.5229885  2.0467935  91

## [14] {pip fruit,other vegetables}             => {whole milk} 0.013523132
0.5175097  2.0253514 133

## [15] {tropical fruit,yogurt}                  => {whole milk} 0.015149975
0.5173611  2.0247698 149

## [16] {yogurt,pastry}                          => {whole milk} 0.009150991
0.5172414  2.0243012  90

## [17] {citrus fruit,root vegetables}           => {whole milk} 0.009150991
0.5172414  2.0243012  90

## [18] {other vegetables,yogurt}                => {whole milk} 0.022267412
0.5128806  2.0072345 219

## [19] {other vegetables,whipped/sour cream}    => {whole milk} 0.014641586
0.5070423  1.9843854 144

## [20] {yogurt,fruit/vegetable juice}           => {whole milk} 0.009456024
0.5054348  1.9780943  93

## [21] {other vegetables,brown bread}           => {whole milk} 0.009354347
0.5000000  1.9568245  92

## [22] {other vegetables,fruit/vegetable juice} => {whole milk} 0.010472801
0.4975845  1.9473713 103

## [23] {butter}                                 => {whole milk} 0.027554652
0.4972477  1.9460530 271

## [24] {curd}                                   => {whole milk} 0.026131164
0.4904580  1.9194805 257

## [25] {root vegetables,other vegetables}       => {whole milk} 0.023182511
0.4892704  1.9148326 228

## [26] {tropical fruit,other vegetables}        => {whole milk} 0.017081851
0.4759207  1.8625865 168

## [27] {citrus fruit,yogurt}                    => {whole milk} 0.010269446
0.4741784  1.8557678 101

## [28] {domestic eggs}                          => {whole milk} 0.029994916
0.4727564  1.8502027 295

## [29] {pork,other vegetables}                  => {whole milk} 0.010167768
0.4694836  1.8373939 100

## [30] {beef,other vegetables}                  => {whole milk} 0.009252669
0.4690722  1.8357838  91

## [31] {other vegetables,margarine}             => {whole milk} 0.009252669
0.4690722  1.8357838  91

## [32] {other vegetables,pastry}                => {whole milk} 0.010574479
0.4684685  1.8334212 104
```

```
## [33] {citrus fruit,tropical fruit}           => {whole milk} 0.009049314
0.4540816  1.7771161  89

## [34] {yogurt,rolls/buns}                     => {whole milk} 0.015556685
0.4526627  1.7715630 153

## [35] {citrus fruit,other vegetables}         => {whole milk} 0.013014743
0.4507042  1.7638982 128

## [36] {whipped/sour cream}                    => {whole milk} 0.032231825
0.4496454  1.7597542 317

## [37] {root vegetables}                       => {whole milk} 0.048906965
0.4486940  1.7560310 481

## [38] {tropical fruit,rolls/buns}             => {whole milk} 0.010981190
0.4462810  1.7465872 108

## [39] {sugar}                                 => {whole milk} 0.015048297
0.4444444  1.7393996 148

## [40] {hamburger meat}                        => {whole milk} 0.014743264
0.4434251  1.7354101 145

## [41] {ham}                                   => {whole milk} 0.011489578
0.4414062  1.7275091 113

## [42] {sliced cheese}                         => {whole milk} 0.010777834
0.4398340  1.7213560 106

## [43] {other vegetables,bottled water}        => {whole milk} 0.010777834
0.4344262  1.7001918 106

## [44] {other vegetables,soda}                 => {whole milk} 0.013929842
0.4254658  1.6651240 137

## [45] {frozen vegetables}                     => {whole milk} 0.020437214
0.4249471  1.6630940 201

## [46] {yogurt,bottled water}                  => {whole milk} 0.009659380
0.4203540  1.6451180  95

## [47] {other vegetables,rolls/buns}           => {whole milk} 0.017895272
0.4200477  1.6439194 176

## [48] {cream cheese }                         => {whole milk} 0.016471784
0.4153846  1.6256696 162

## [49] {butter milk}                           => {whole milk} 0.011591256
0.4145455  1.6223854 114

## [50] {margarine}                             => {whole milk} 0.024199288
0.4131944  1.6170980 238

## [51] {hard cheese}                           => {whole milk} 0.010066090
0.4107884  1.6076815  99

## [52] {chicken}                               => {whole milk} 0.017590239
0.4099526  1.6044106 173

## [53] {white bread}                           => {whole milk} 0.017081851
0.4057971  1.5881474 168
```

```
## [54] {beef}                                  => {whole milk} 0.021250635
0.4050388  1.5851795 209

## [55] {tropical fruit}                        => {whole milk} 0.042297916
0.4031008  1.5775950 416

## [56] {oil}                                   => {whole milk} 0.011286223
0.4021739  1.5739675 111

## [57] {yogurt}                                => {whole milk} 0.056024403
0.4016035  1.5717351 551

## [58] {pip fruit}                             => {whole milk} 0.030096594
0.3978495  1.5570432 296

## [59] {onions}                                => {whole milk} 0.012099644
0.3901639  1.5269647 119

## [60] {hygiene articles}                      => {whole milk} 0.012811388
0.3888889  1.5219746 126

## [61] {brown bread}                           => {whole milk} 0.025216065
0.3887147  1.5212930 248

## [62] {other vegetables}                      => {whole milk} 0.074834774
0.3867578  1.5136341 736

## [63] {meat}                                  => {whole milk} 0.009964413
0.3858268  1.5099906  98

## [64] {pork}                                  => {whole milk} 0.022165735
0.3844797  1.5047187 218

## [65] {yogurt,soda}                           => {whole milk} 0.010472801
0.3828996  1.4985348 103

## [66] {sausage,other vegetables}              => {whole milk} 0.010167768
0.3773585  1.4768487 100

## [67] {napkins}                               => {whole milk} 0.019725470
0.3766990  1.4742678 194

## [68] {pastry}                                => {whole milk} 0.033248602
0.3737143  1.4625865 327

## [69] {dessert}                               => {whole milk} 0.013726487
0.3698630  1.4475140 135

## [70] {citrus fruit}                          => {whole milk} 0.030503305
0.3685504  1.4423768 300

## [71] {fruit/vegetable juice}                 => {whole milk} 0.026639553
0.3684951  1.4421604 262

## [72] {long life bakery product}              => {whole milk} 0.013523132
0.3614130  1.4144438 133

## [73] {berries}                               => {whole milk} 0.011794611
0.3547401  1.3883281 116

## [74] {frankfurter}                           => {whole milk} 0.020538892
0.3482759  1.3630295 202
```

```
## [75] {frozen meals}                          => {whole milk} 0.009862735
0.3476703  1.3606593  97

## [76] {newspapers}                            => {whole milk} 0.027351296
0.3426752  1.3411103 269

## [77] {chocolate}                             => {whole milk} 0.016675140
0.3360656  1.3152427 164

## [78] {waffles}                               => {whole milk} 0.012709710
0.3306878  1.2941961 125

## [79] {coffee}                                => {whole milk} 0.018708693
0.3222417  1.2611408 184

## [80] {sausage}                               => {whole milk} 0.029893238
0.3181818  1.2452520 294

## [81] {bottled water}                         => {whole milk} 0.034367056
0.3109476  1.2169396 338

## [82] {rolls/buns}                            => {whole milk} 0.056634469
0.3079049  1.2050318 557

## [83] {sausage,rolls/buns}                    => {whole milk} 0.009354347
0.3056478  1.1961984  92

## [84] {salty snack}                           => {whole milk} 0.011184545
0.2956989  1.1572618 110

## [85] {bottled beer}                          => {whole milk} 0.020437214
0.2537879  0.9932367 201
```

```
is.significant(milk.rules, transactions)
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TR
UE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  T
RUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE
```

```
is.maximal(milk.rules)
```

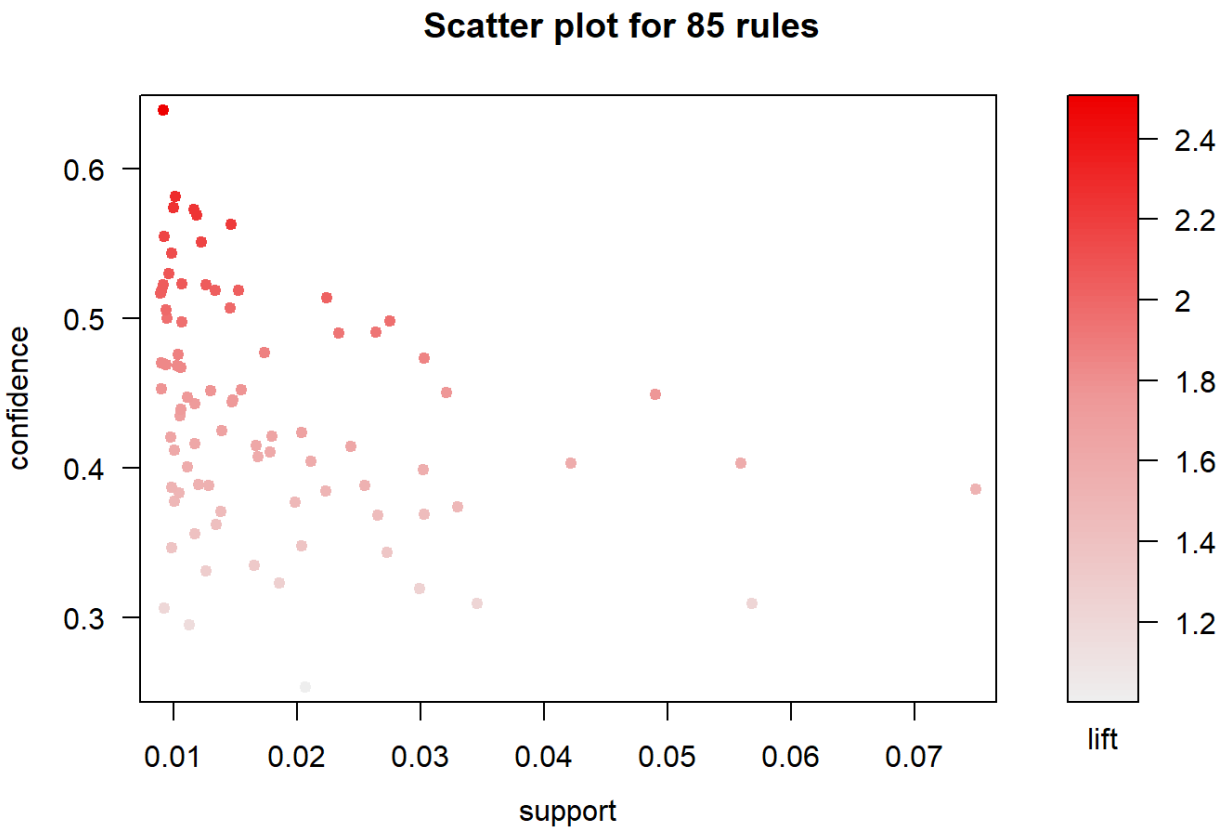
```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TR
UE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  T
RUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE
TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE
TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE
FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE
```

```
is.redundant(milk.rules)
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FAL
SE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FA
LSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE F
ALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
```

```
plot(milk.rules,  measure=c("support", "confidence"), shading="lift")
```




```
coke.rules <- sort(subset(rules, subset = rhs %in% "soda"), by = "confidence"
)
```

```
summary(coke.rules)
```

```
## set of 6 rules
##
## rule length distribution (lhs + rhs):sizes
## 2 3
## 5 1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   2.000   2.167   2.000   3.000
##
## summary of quality measures:
```

```
##     support           confidence          lift            count
##  Min.   :0.009659   Min.   :0.2546   Min.   :1.460   Min.   : 95.0
##  1st Qu.:0.010778   1st Qu.:0.2595   1st Qu.:1.488   1st Qu.:106.0
##  Median :0.015963   Median :0.2640   Median :1.514   Median :157.0
##  Mean   :0.017455   Mean   :0.2716   Mean   :1.557   Mean   :171.7
##  3rd Qu.:0.022827   3rd Qu.:0.2708   3rd Qu.:1.553   3rd Qu.:224.5
##  Max.   :0.028978   Max.   :0.3156   Max.   :1.810   Max.   :285.0
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.009       0.25
```

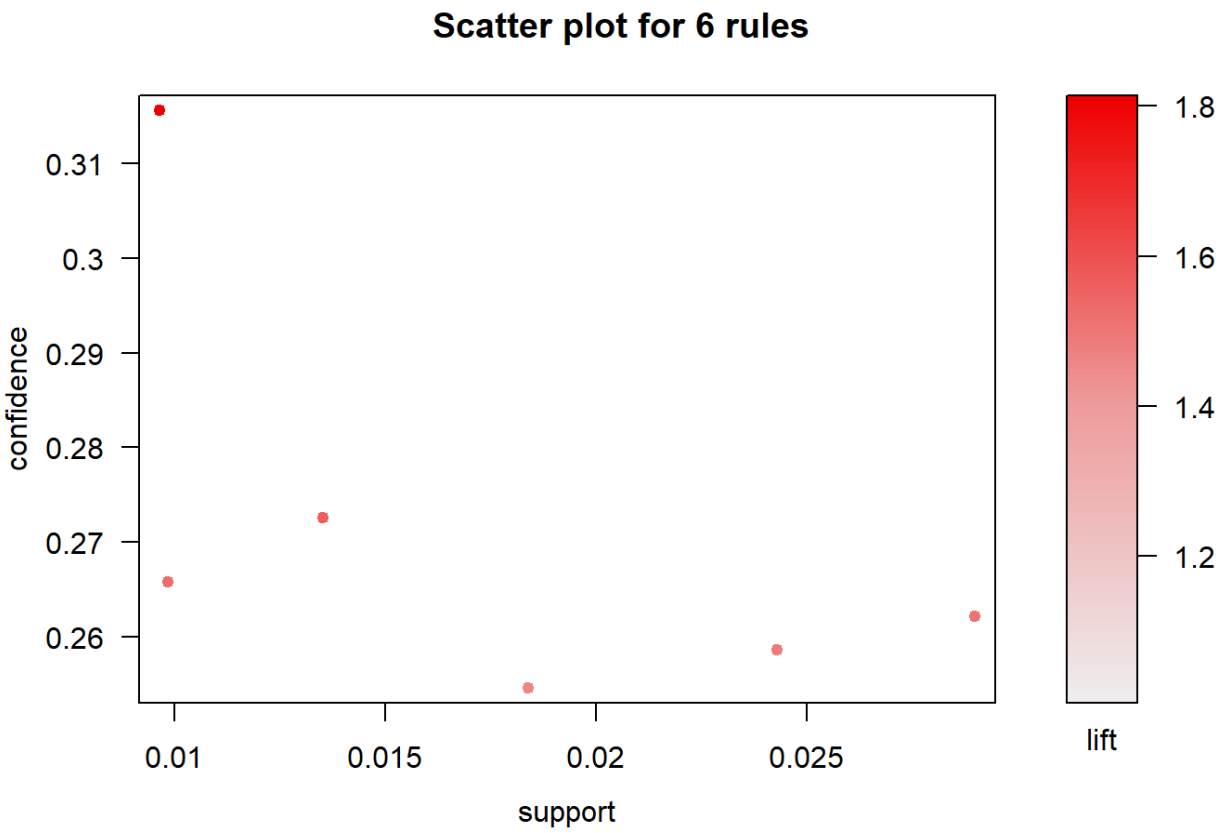
```
inspect(coke.rules)
```

```
##     lhs                          rhs    support     confidence lift     coun
t
## [1] {sausage,rolls/buns}     => {soda} 0.009659380 0.3156146  1.809953  95
## [2] {chocolate}              => {soda} 0.013523132 0.2725410  1.562939 133
## [3] {dessert}                => {soda} 0.009862735 0.2657534  1.524015  97
## [4] {bottled water}          => {soda} 0.028978139 0.2621895  1.503577 285
## [5] {sausage}                => {soda} 0.024300966 0.2586580  1.483324 239
## [6] {fruit/vegetable juice} => {soda} 0.018403660 0.2545710  1.459887 181
```

```
is.significant(coke.rules, transactions)
```

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

```
plot(coke.rules,  measure=c("support", "confidence"), shading="lift")
```



Analysis:

Analysis was aimed to see what makes people buy milk (what products to be exact). To do se we should choose subset of rules that has whole milk (or soda) in right hand side of a rule.

It turns out that most popular baskets are curd, yoghurt or fruits and vegetables. Seems like the most popular one-week ahead groceries we do.

On the other hand it seems that soda is mostly bought with either sweets (chocolate) or with beverages/meat. Looks like a party ahead!

Most of the rules are significant (Fisher's exact test) apart from some of the least confident rules of milk buying.

We can also see on the scatter plot of rules for milk that the higher the confidence the higher lift, which was not observed before. It also occurs on Coke rules plot, but is not that visible.

Moreover I tested supersets and subsets of milk rules. (I disabled the output here, because it really didn't show much in Markdown. One can simply recreate steps here)

```
is.superset(milk.rules)

is.subset(milk.rules)
```

It doesn't seem that the rules are supersets or subsets to each other.

Meat rules:

```
meat.rules <- sort(subset(rules, subset = lhs %in%  "beef"|lhs %in%  "sausage
" |lhs %in%  "chicken"), by = "confidence")
summary(meat.rules)
```

```
## set of 19 rules
##
## rule length distribution (lhs + rhs):sizes
##  2  3
## 11  8
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   2.000   2.421   3.000   3.000
##
## summary of quality measures:
##     support           confidence          lift            count
##  Min.   :0.009253   Min.   :0.2536   Min.   :1.196   Min.   : 91.0
##  1st Qu.:0.009659   1st Qu.:0.3093   1st Qu.:1.483   1st Qu.: 95.0
##  Median :0.013625   Median :0.3314   Median :1.758   Median :134.0
##  Mean   :0.016156   Mean   :0.3471   Mean   :1.802   Mean   :158.9
##  3rd Qu.:0.020488   3rd Qu.:0.4013   3rd Qu.:2.049   3rd Qu.:201.5
##  Max.   :0.030605   Max.   :0.4691   Max.   :3.040   Max.   :301.0
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.009       0.25
```

```
inspect(meat.rules)
```

```
##      lhs                          rhs                support     confiden
ce lift     count
## [1]  {beef,other vegetables}    => {whole milk}       0.009252669 0.469072
2  1.835784  91
## [2]  {beef,whole milk}          => {other vegetables} 0.009252669 0.435406
7  2.250250  91
## [3]  {chicken}                  => {other vegetables} 0.017895272 0.417061
6  2.155439 176
## [4]  {chicken}                  => {whole milk}       0.017590239 0.409952
6  1.604411 173
## [5]  {beef}                     => {whole milk}       0.021250635 0.405038
8  1.585180 209
```

```
## [6]  {sausage,soda}             => {rolls/buns}       0.009659380 0.397489
5  2.161034  95

## [7]  {sausage,other vegetables} => {whole milk}       0.010167768 0.377358
5  1.476849 100

## [8]  {beef}                     => {other vegetables} 0.019725470 0.375969
0  1.943066 194

## [9]  {sausage,whole milk}       => {other vegetables} 0.010167768 0.340136
1  1.757876 100

## [10] {beef}                     => {root vegetables}  0.017386884 0.331395
3  3.040367 171

## [11] {sausage}                  => {rolls/buns}       0.030604982 0.325757
6  1.771048 301

## [12] {sausage}                  => {whole milk}       0.029893238 0.318181
8  1.245252 294

## [13] {sausage,rolls/buns}       => {soda}             0.009659380 0.315614
6  1.809953  95

## [14] {sausage,whole milk}       => {rolls/buns}       0.009354347 0.312925
2  1.701282  92

## [15] {sausage,rolls/buns}       => {whole milk}       0.009354347 0.305647
8  1.196198  92

## [16] {sausage}                  => {other vegetables} 0.026944586 0.286796
5  1.482209 265

## [17] {beef}                     => {rolls/buns}       0.013624809 0.259689
9  1.411858 134

## [18] {sausage}                  => {soda}             0.024300966 0.258658
0  1.483324 239

## [19] {chicken}                  => {root vegetables}  0.010879512 0.253554
5  2.326221 107
```

```
is.significant(meat.rules, transactions)
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TR
UE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
```

In case of meat, we search whether meats like: beef, chicken (poultry) or sausage show up in the left hand sides of rules.

Let's see what people buy after they have put meat (sausage or beef) to the basket. It turns out that the most popular option associated with meat is milk! It is a little bit confusing, because only in lift column we see how popular option is. The real winner here are root vegetables that are 3 times more likely to be put into the basket than other products. Rest of the products are just regular grocery stuff.

Yogurt rules:

```
yog.rules <- sort(subset(rules, subset = lhs %in%  "yogurt"), by = "confidenc
e")
```

```
summary(yog.rules)
```

```
## set of 26 rules
##
## rule length distribution (lhs + rhs):sizes
##  2  3
##  2 24
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   3.000   3.000   2.923   3.000   3.000
##
## summary of quality measures:
##     support           confidence          lift           count
##  Min.   :0.009151   Min.   :0.2595   Min.   :1.439   Min.   : 90.0
##  1st Qu.:0.010193   1st Qu.:0.3170   1st Qu.:1.739   1st Qu.:100.2
##  Median :0.012303   Median :0.4365   Median :2.039   Median :121.0
##  Mean   :0.015651   Mean   :0.4281   Mean   :2.058   Mean   :153.9
##  3rd Qu.:0.015150   3rd Qu.:0.5162   3rd Qu.:2.356   3rd Qu.:149.0
##  Max.   :0.056024   Max.   :0.6389   Max.   :2.729   Max.   :551.0
##
## mining info:
##       data ntransactions support confidence
##  Groceries          9835   0.009       0.25
```

```
inspect(yog.rules)
```

```
##      lhs                               rhs                support     conf
idence lift     count
## [1]  {butter,yogurt}                => {whole milk}       0.009354347 0.63
88889  2.500387  92
## [2]  {curd,yogurt}                  => {whole milk}       0.010066090 0.58
23529  2.279125  99
## [3]  {root vegetables,yogurt}       => {whole milk}       0.014539908 0.56
29921  2.203354 143
## [4]  {pip fruit,yogurt}             => {whole milk}       0.009557702 0.53
10734  2.078435  94
## [5]  {yogurt,whipped/sour cream}    => {whole milk}       0.010879512 0.52
45098  2.052747 107
## [6]  {tropical fruit,yogurt}        => {whole milk}       0.015149975 0.51
73611  2.024770 149
```

```
## [7]  {yogurt,pastry}                => {whole milk}       0.009150991 0.51
72414  2.024301  90

## [8]  {other vegetables,yogurt}      => {whole milk}       0.022267412 0.51
28806  2.007235 219

## [9]  {yogurt,fruit/vegetable juice} => {whole milk}       0.009456024 0.50
54348  1.978094  93

## [10] {root vegetables,yogurt}       => {other vegetables} 0.012913066 0.50
00000  2.584078 127

## [11] {yogurt,whipped/sour cream}    => {other vegetables} 0.010167768 0.49
01961  2.533410 100

## [12] {citrus fruit,yogurt}          => {whole milk}       0.010269446 0.47
41784  1.855768 101

## [13] {yogurt,rolls/buns}            => {whole milk}       0.015556685 0.45
26627  1.771563 153

## [14] {yogurt,bottled water}         => {whole milk}       0.009659380 0.42
03540  1.645118  95

## [15] {tropical fruit,yogurt}        => {other vegetables} 0.012302999 0.42
01389  2.171343 121

## [16] {yogurt}                       => {whole milk}       0.056024403 0.40
16035  1.571735 551

## [17] {whole milk,yogurt}            => {other vegetables} 0.022267412 0.39
74592  2.054131 219

## [18] {yogurt,soda}                  => {whole milk}       0.010472801 0.38
28996  1.498535 103

## [19] {yogurt,rolls/buns}            => {other vegetables} 0.011489578 0.33
43195  1.727815 113

## [20] {yogurt}                       => {other vegetables} 0.043416370 0.31
12245  1.608457 427

## [21] {other vegetables,yogurt}      => {root vegetables}  0.012913066 0.29
74239  2.728698 127

## [22] {other vegetables,yogurt}      => {tropical fruit}   0.012302999 0.28
33724  2.700550 121

## [23] {whole milk,yogurt}            => {rolls/buns}       0.015556685 0.27
76770  1.509648 153

## [24] {whole milk,yogurt}            => {tropical fruit}   0.015149975 0.27
04174  2.577089 149

## [25] {other vegetables,yogurt}      => {rolls/buns}       0.011489578 0.26
46370  1.438753 113

## [26] {whole milk,yogurt}            => {root vegetables}  0.014539908 0.25
95281  2.381025 143
```

```
is.significant(yog.rules, transactions)
```

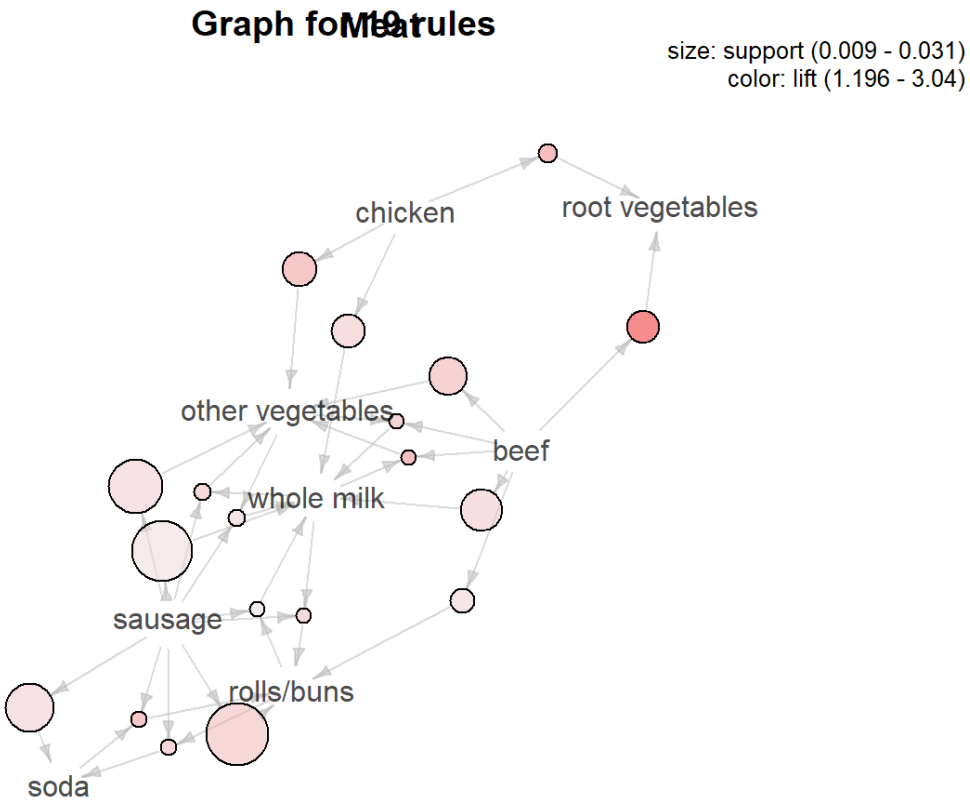
```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Same as above we subset only these rules that have yogurt in left hand side of a rule.

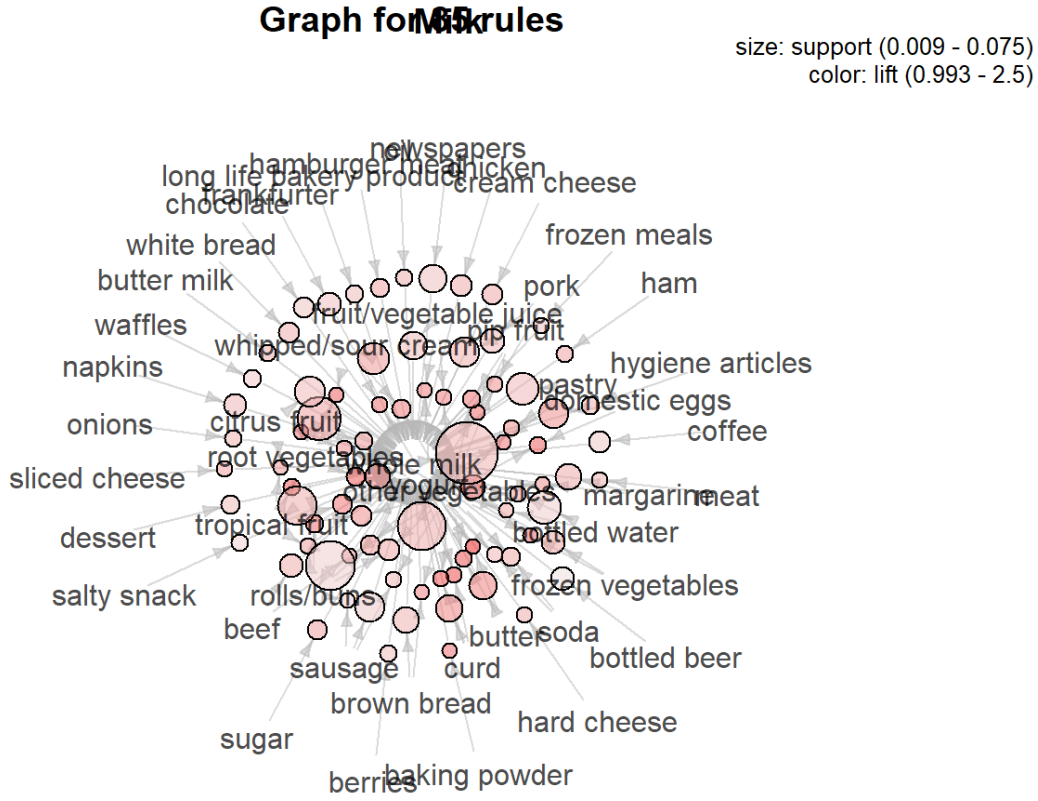
Most of the times someone buys yogurt he will also put milk or vegetables into his basket - with greater correlation to 'other vegetables'. There is not much variation, nothing changes with the lowering confidence.

Some Visualization for above subrules:

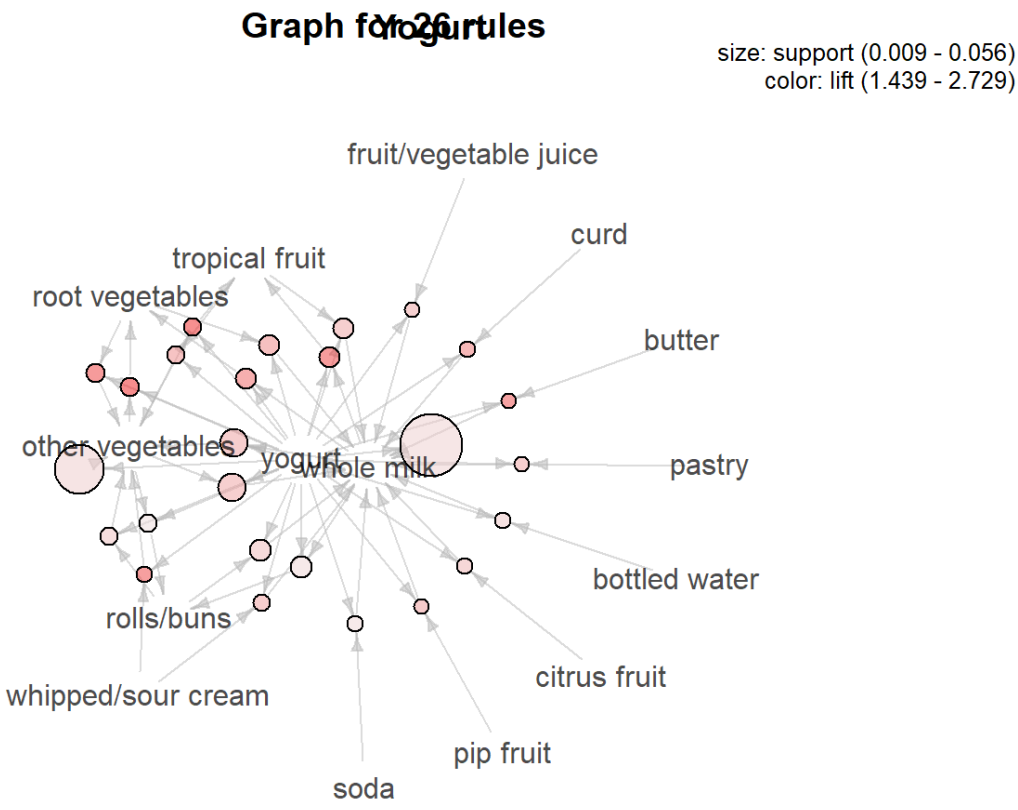
```
# plot for subrules
plot(meat.rules,method="graph",interactive=FALSE,shading="lift")
title(main = "Meat")
```



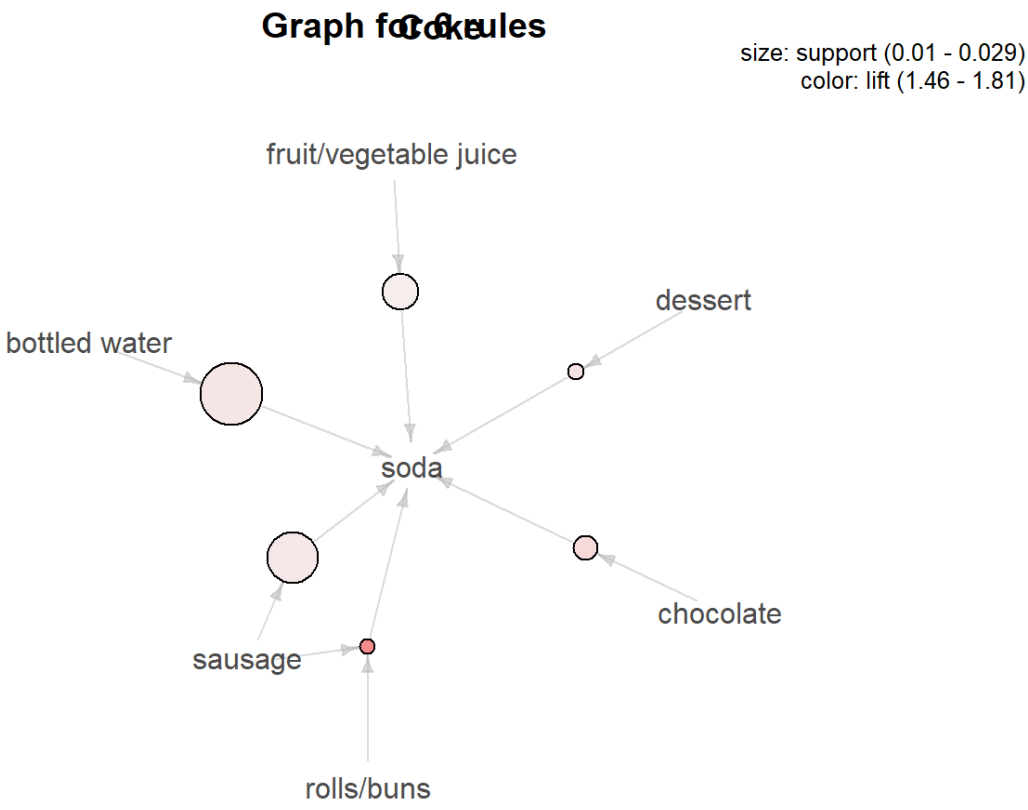

```
plot(milk.rules,method="graph",interactive=FALSE,shading="lift")
title(main = "Milk")
```




```
plot(yog.rules,method="graph",interactive=FALSE,shading="lift")
title(main = "Yogurt")
```




```
plot(coke.rules,method="graph",interactive=FALSE,shading="lift")
title(main = "Coke")
```



Above we can see graphs for the previously interpreted rules that have the same conclusions as previously. The reddier the circle the more probable is the client to buy two of those items than any other items and the bigger the circle the more probable is to buy two of those items. Moreover the arrow points to the direction of a possible basket rule. Therefore in case of Coke, we can notice bottled water and soda as the rule with highest support.

More complicated conclusions can be drawn from the meat rules plot. We can see that the sausage is the mostly supported additional product for milk.

Jaccard Index

For the set of milk rules let's calculate the Jaccard Index. It is the representation of how much likely are two items to be bought together.

```
trans.sel<-transactions[,itemFrequency(transactions)>0.1] # selected transact
ions


dissimilarity(trans.sel, which="items")
```

```
##                  tropical fruit root vegetables other vegetables whole mil
k    yogurt rolls/buns bottled water
```

```
## root vegetables       0.8908803
```

```
## other vegetables      0.8632843       0.8142686
## whole milk            0.8670502       0.8450387        0.8000000
## yogurt                0.8638941       0.8840183        0.8500702  0.834733
1
## rolls/buns            0.9068873       0.9095382        0.8727604  0.852058
4 0.8811115
## bottled water         0.9060403       0.9231920        0.9111435  0.896382
6 0.8987909  0.9104590
## soda                  0.9193548       0.9297235        0.9023058  0.897235
3 0.9045422  0.8802034     0.8867700
```

Because I have picked such high minimal frequency we have not much items, but moreover Jaccard Index seems to have high values telling us that most of those products do not overlap. Such an array as presented above tells that the higher the values of Jaccard Index the more likely are two products to be in the same transaction. Highest percentage is between root vegetables and tropical fruits.

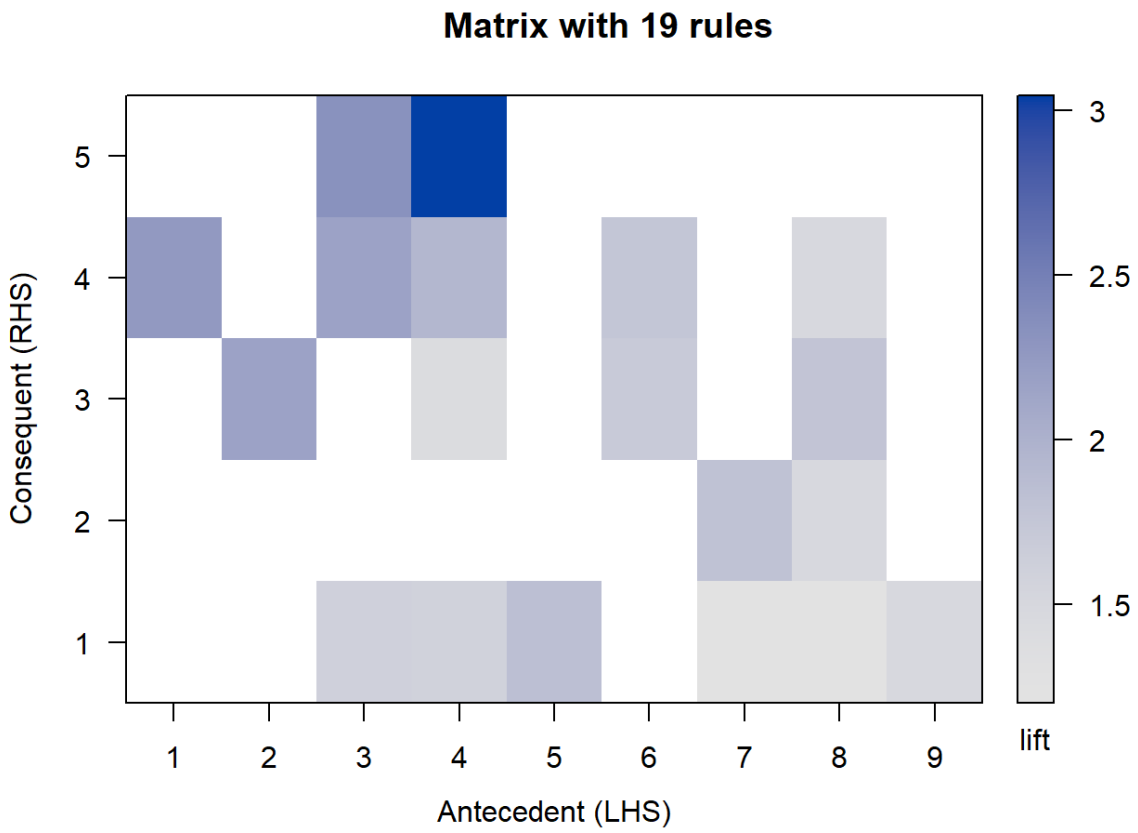
Advcanced graphics

Apart from the analytical study of the created rulesets and research of the rules for particular items, we can present more advanced graphics to more thoroughly analyze ruleset.

Let's present the ruleset for meat but in a matrix form. Each of the matrix cells can have different blue shade depending on the lift value. Numbers on the axes are corresponding to the items listed before the matrix. For example the most blue cell corresponds to the rule {beef} -> {root vegetables}, hence (as previously mentioned) root vegetables are most likely to be bought with beef. On the second place is the chicken and for the rest of antecedent items there is no significant lift at all (it is too small to be presented on the graph). Such a graph is only confirmation fo the conclusions drawn before, but in a simplier form.

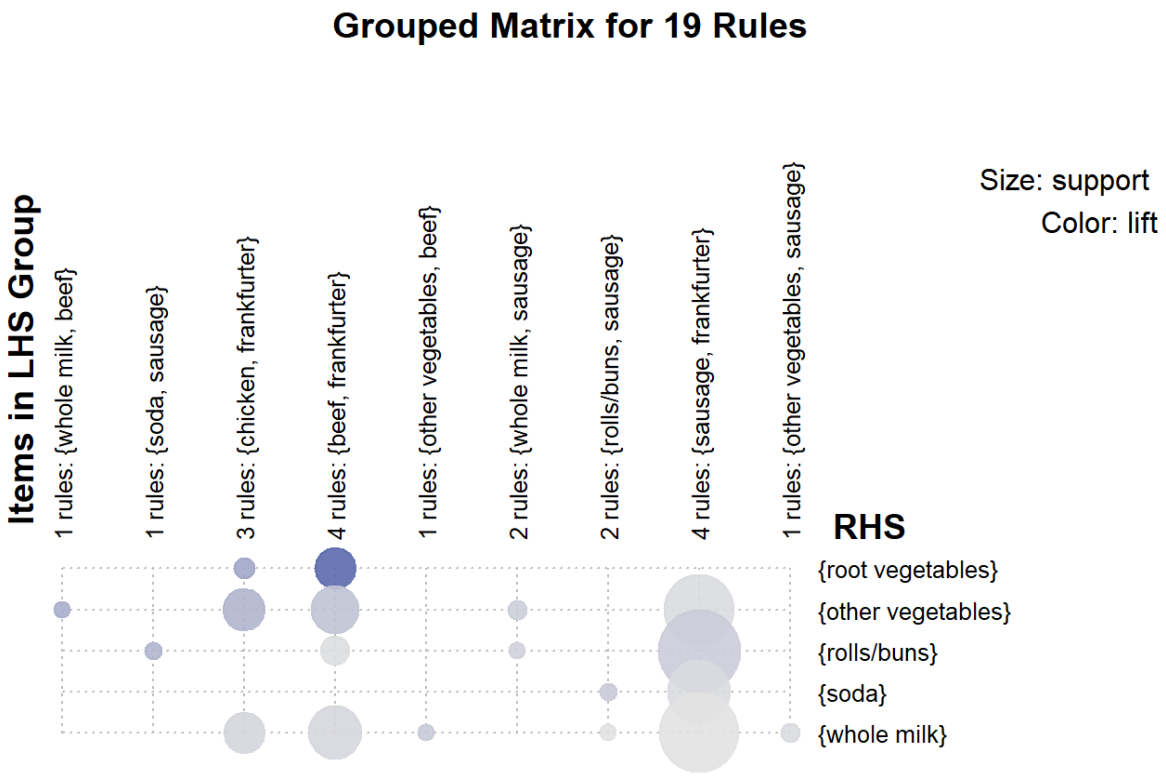
```
plot(meat.rules, method="matrix", measure=c("support","confidence"), control=
list(reorder=TRUE, col=sequential_hcl(200)))
```

```
## Itemsets in Antecedent (LHS)
## [1] "{beef,whole milk}"          "{sausage,soda}"             "{chicken}"
"{beef}"                    "{beef,other vegetables}"    "{sausage,whole mil
k}"       "{sausage,rolls/buns}"       "{sausage}"                  "{sausage
,other vegetables}"
## Itemsets in Consequent (RHS)
## [1] "{whole milk}"       "{soda}"             "{rolls/buns}"       "{other
vegetables}" "{root vegetables}"
```



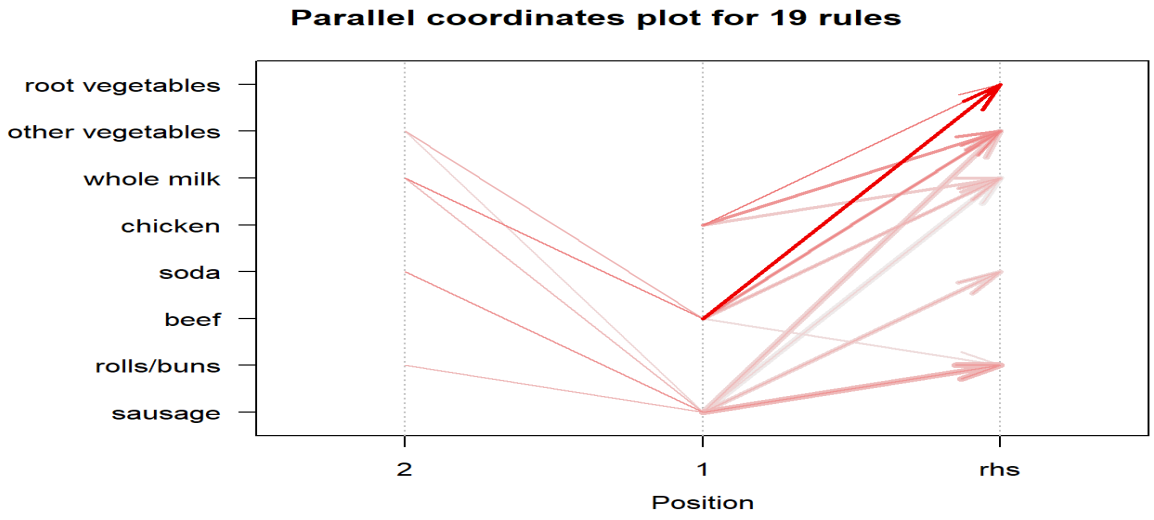
A better way to present this is a Grouped matrix plot. It has the same data on the axes as before, but moreover it shows the support of the rules. It can be noted that rules connected with sausages have the biggest support (among listed). The previously concluded biggest lift for {beef} -> {root vegetables} is also noticeable.

```
plot(meat.rules, method="grouped", measure="support", control=list(col=sequen
tial_hcl(100)))
```



We can even show dependencies with parallel coordinates plot. We can see that the mostly red arrow (each of them represents one rule) connects beef and root vegetables. Moreover most of the arrows connect sausage on the first position, as previously stated.

```
plot(meat.rules, method="paracoord", control=list(reorder=TRUE))
```



All these plots can be used to have first conclusions on the dataset to proceed with some hypotheses for analytical testing, or to ensure our analytical conclusions with nice graphics.

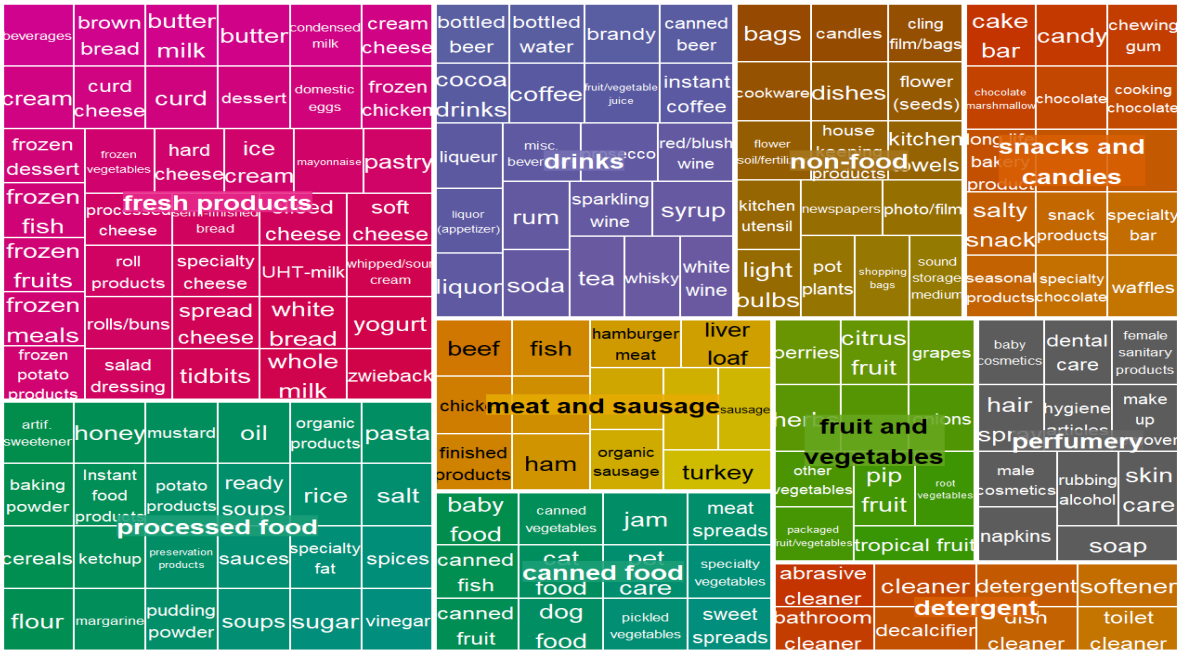
Treemap

If we want to look into data deeper, we can create interesting plots that show us how many products of each type are available to buy in the grocery store. Here I made two treemaps, including one with deeper segmentation that present which products are on the lists of the shop. Moreover it can explain why there are so many connections with milk and fresh products, whereas just a little with coke.

```
occur1 <- transactions@itemInfo %>% group_by(level1) %>% summarize(n=n())

occur2 <- transactions@itemInfo %>% group_by(level1, level2) %>% summarize(n=
n())

occur3 <- transactions@itemInfo %>% group_by(level1, level2, labels) %>% summ
arize(n=n())


treemap(occur1,index=c("level1"),vSize="n",title="",palette="Dark2",border.co
l="#FFFFFF")
```

```
treemap(occur2,index=c("level1", "level2"),vSize="n",title="",palette="Dark2"
,border.col="#FFFFFF")
```


```
treemap(occur3,index=c("level1", "labels"),vSize="n",title="",palette="Dark2"
,border.col="#FFFFFF")
```



Each of these charts have different level of deepth. First only shows the bigger group names (like aisles in the shop). Second shows deeper segmentation intro product types (for example within aisle). The last chart presents each of the products available - it does give us less information that the previous one.

Less than likely? - Lift < 1

Interesting part of the study would be checking for items that are less than likely to be bought together. These would be described by lift < 1.

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

```
##     lhs                      rhs           support     confidence lift     
count
## [1] {sausage}             => {whole milk} 0.029893238 0.3181818  1.2452520
294
## [2] {bottled water}       => {whole milk} 0.034367056 0.3109476  1.2169396
338
## [3] {rolls/buns}          => {whole milk} 0.056634469 0.3079049  1.2050318
557
## [4] {sausage,rolls/buns} => {whole milk} 0.009354347 0.3056478  1.1961984
92
## [5] {salty snack}         => {whole milk} 0.011184545 0.2956989  1.1572618
110
## [6] {bottled beer}        => {whole milk} 0.020437214 0.2537879  0.9932367
201
```

There is only one item in our rules set, that has lift less than 1. It is a connection between whole milk and bottled beer. It means that we are less likely to buy milk than any other product in dataset, while already having beer in basket. Maybe that's a hint that beeroholics don't drink milk? :)