Would You Like a Cookie With That Coffee? A Basket Analisys on Data From a Bakery

Michał Szałański

20 March 2019

Introduction

Goal of the paper

I always wondered how efficient is upselling in coffee shops and bakeries. There always seem to be some kind of promotion to incentivise the customers to buy more products. Thanks to this dataset from Kaggle (temporary 404) we can analise what product combinations are popular in one particular bakery.

Data preparation

Libraries

```
library(tibble)
library(dplyr)
library(tidyr)
library(ggplot2)
library(gridExtra)
library(knitr)
library(arules)
library(arulesViz)
library(arulesCBA)
library(arulesSequences)
library(knitr)
library(reshape)
options(scipen=999)

# Nice colors
cYellow = '#FADA5E'
cBlue = '#378CC7'
```

Manipulating data

The data out of the box isn't perfect - we have to remove some transactions, and also make a subset that will be explaned later. Then we save the data back as csv, since the read.transactions() cannot use data frames.

```
# Read the data
dataImport <- read.csv('data/teaBasket_DMS.csv')

# Remove transactions with NONE and Adjustment
dataExport <- dataImport %>% filter(dataImport$Item != "NONE" & dataImport$Item != "Adjustment")

# Remove two most frequent items
dataExportNoCoffee <- dataImport %>%
  filter(!(dataImport$Item %in% c("NONE", "Adjustment", "Coffee", "Bread", "Postcard")))
```

```r
# Write only the relevant columns
dataExport[, c(3:4)] %>% write.csv( './data/transactions.csv')
dataExportNoCoffee[, c(3:4)] %>% write.csv( './data/transactionsNoCoffee.csv')
```

Loading transactions from csv

Now, we can load the processed data using the read.transactions() function from the arules package. We make to main datasets - one containing all the transactions (coffee) and one skiping the two most popular products, coffee and bread (tea).

```r
# Read the data as transactions
coffee <- read.transactions('./data/transactions.csv',
                            format="single",
                            cols = c('Transaction', 'Item'),
                            sep = ',',
                            header = TRUE)

tea <- read.transactions('./data/transactionsNoCoffee.csv',
                         format="single",
                         cols = c('Transaction', 'Item'),
                         sep = ',',
                         header = TRUE)
```

Data exploration

First look at the data

With coffee

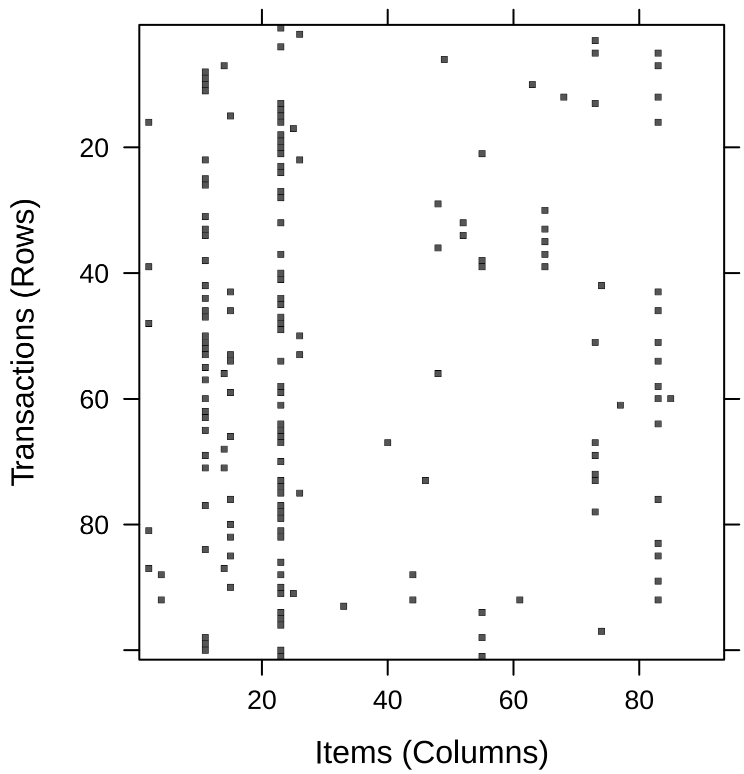
We can now take a look at a quick summary of the data.

```
coffee
```

```
    transactions in sparse format with
     9464 transactions (rows) and
     93 items (columns)
```

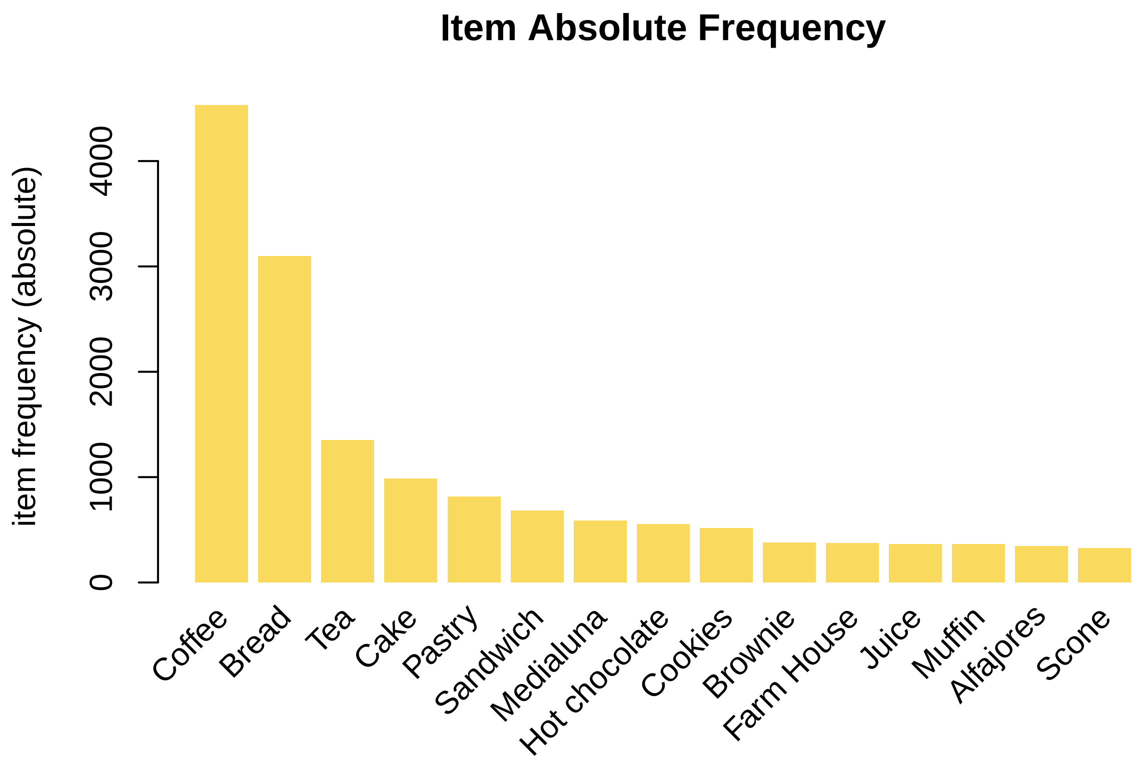
9500 transactions should give some interesting results. The dataset consists of 93 different items.

```
image(coffee[1000:1100])
```

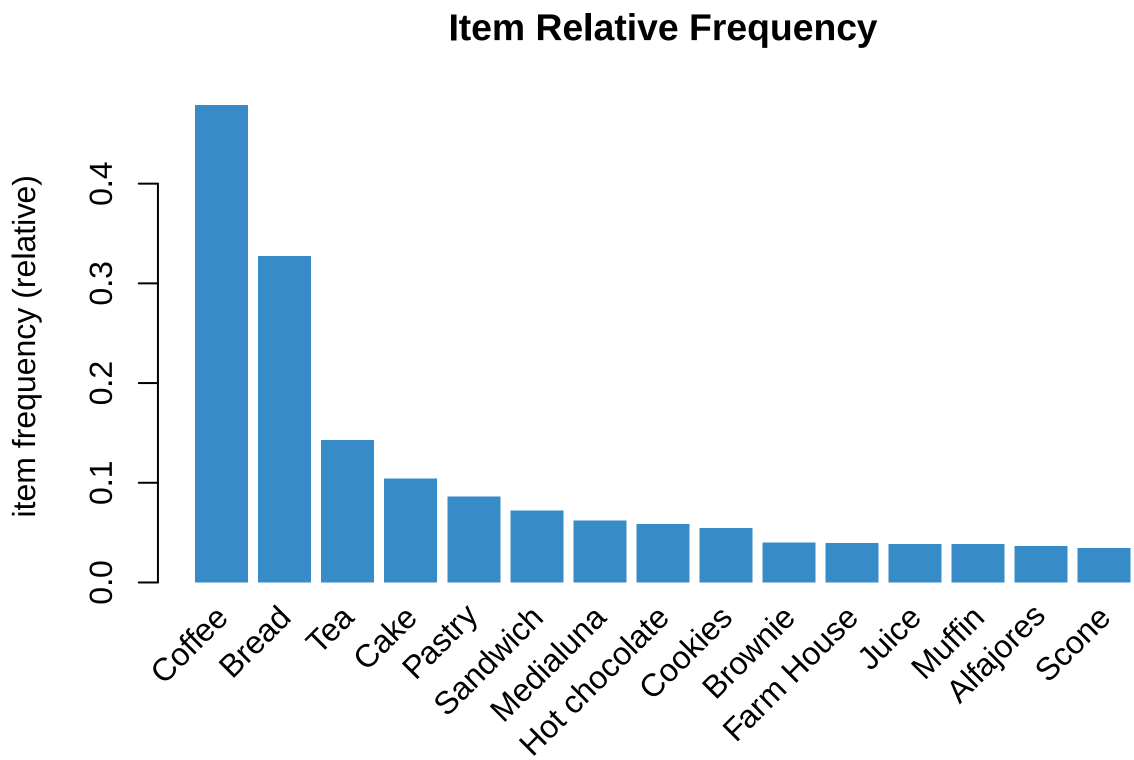


We can see that there are some strong patterns in a random sample from the dataset. We can look at them using a frequency plot.

```
coffee %>% itemFrequencyPlot(topN=15,
                             type="absolute",
                             main="Item Absolute Frequency",
                             col = cYellow,
                             border = NA)
```




```
coffee %>% itemFrequencyPlot(topN=15,
                             type="relative",
                             main="Item Relative Frequency",
                             col = cBlue,
                             border = NA)
```



Coffee and bread appear frequently in the transactions - nearly 50% and 30% respectivly. As I discovered later, this can significantly skew the results of basket analisys, so I prepader a second dataset that excluded these two products.

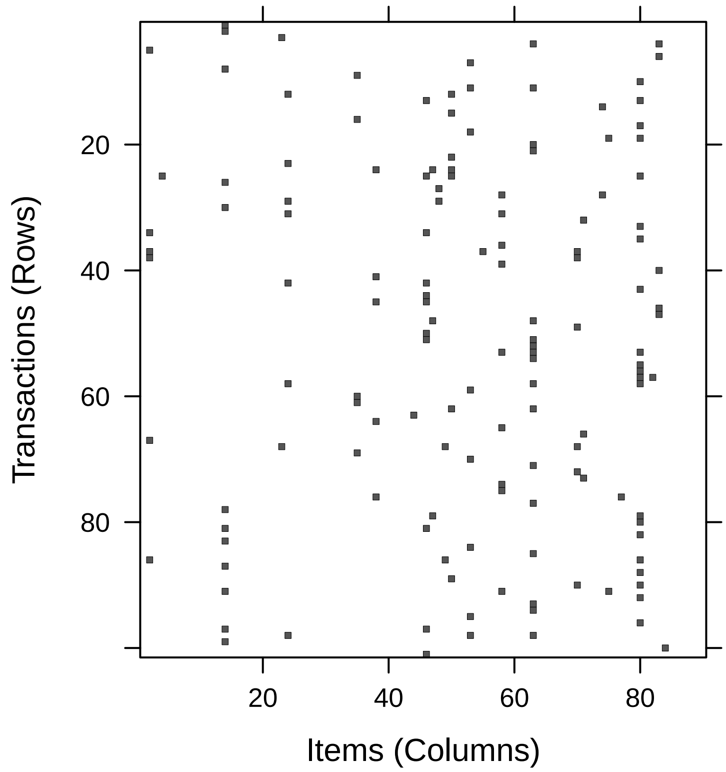
Without coffee and bread

```r
tea
```

```
## transactions in sparse format with
##  6788 transactions (rows) and
##  90 items (columns)
```

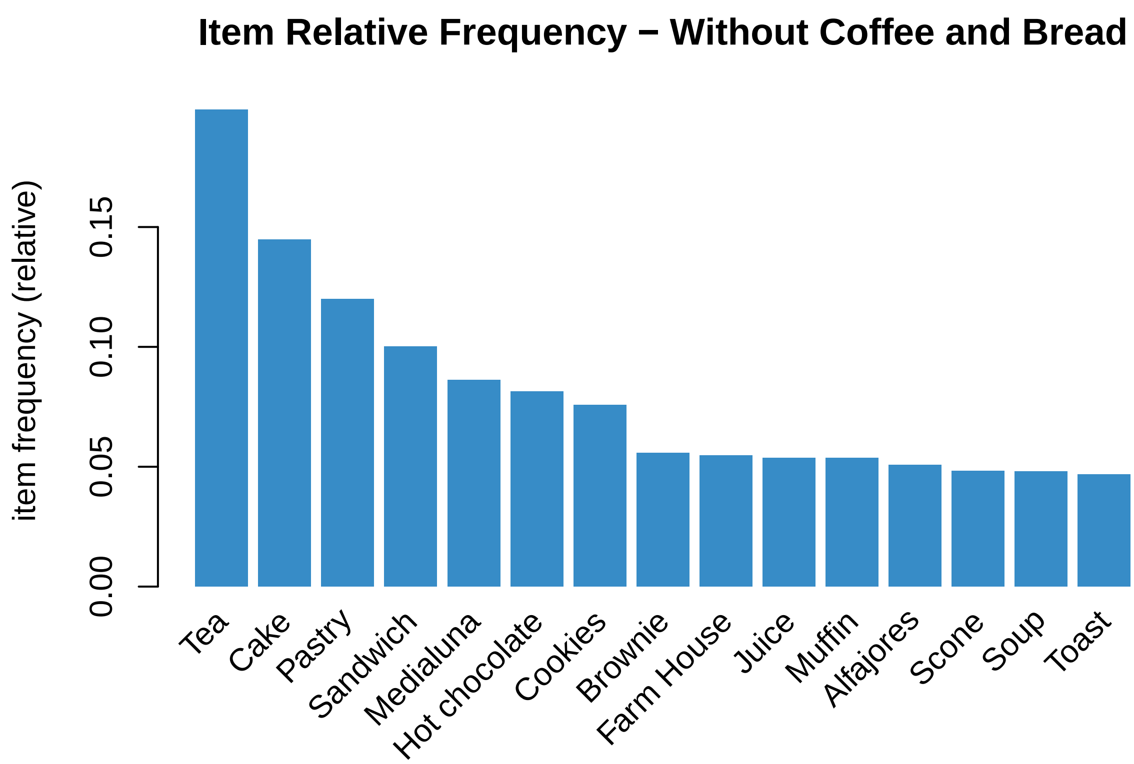
The number of transactions drops to 6 800, and products to 90.

```r
image(tea[1000:1100])
```



The patterns in the sample are much wekaer now - the analisys should show better results.

```
tea %>% itemFrequencyPlot(topN=15,
                          type="relative",
                          main="Item Relative Frequency - Without Coffee and Bread",
                          col = cBlue,
                          border = NA)
```



The most popular items are now: Tea, Cake and Pastry.

Cross tables

To better see the different pairs of items, we can calculate a cross table.

```
coffeeCount <- coffee %>% crossTable(measure="count", sort=TRUE)
teaCount <- tea %>% crossTable(measure="count", sort=TRUE)
```

```
coffeeCount[1:8,1:6] %>% kable()
```

	Coffee	Bread	Tea	Cake	Pastry	Sandwich
Coffee	4528	852	472	518	450	362
Bread	852	3097	266	221	276	161
Tea	472	266	1350	225	91	136
Cake	518	221	225	983	49	65
Pastry	450	276	91	49	815	11
Sandwich	362	161	136	65	11	680
Medialuna	333	160	77	35	87	20
Hot chocolate	280	127	76	108	54	42

```
coffeeCount[1,1]
```

```
[1] 4528
```

```r
coffeeCount[2,2]
```

```
[1] 3097
```

The data seems to be evenly distributed, there are no obvious outliers. There are a total of 4 528 transactions with coffee, and 3 097 with bread.
We can analise the support and lift for the most popular items.

```r
teaSupport <- tea %>% crossTable(measure="support", sort=TRUE)
teaLift <- tea %>% crossTable(measure="lift", sort=TRUE)
```

```r
teaSupport[1:8,1:6] %>% kable(digits = 3)
```

	Tea	Cake	Pastry	Sandwich	Medialuna	Hot chocolate
Tea	0.199	0.033	0.013	0.020	0.011	0.011
Cake	0.033	0.145	0.007	0.010	0.005	0.016
Pastry	0.013	0.007	0.120	0.002	0.013	0.008
Sandwich	0.020	0.010	0.002	0.100	0.003	0.006
Medialuna	0.011	0.005	0.013	0.003	0.086	0.007
Hot chocolate	0.011	0.016	0.008	0.006	0.007	0.081
Cookies	0.014	0.010	0.004	0.004	0.004	0.008
Brownie	0.009	0.006	0.003	0.003	0.003	0.006

```r
teaLift[1:8,1:6] %>% kable(digits = 2)
```

	Tea	Cake	Pastry	Sandwich	Medialuna	Hot chocolate
Tea	NA	1.15	0.56	1.01	0.66	0.69
Cake	1.15	NA	0.42	0.66	0.41	1.35
Pastry	0.56	0.42	NA	0.13	1.24	0.81
Sandwich	1.01	0.66	0.13	NA	0.34	0.76
Medialuna	0.66	0.41	1.24	0.34	NA	0.95
Hot chocolate	0.69	1.35	0.81	0.76	0.95	NA
Cookies	0.91	0.93	0.45	0.52	0.54	1.36
Brownie	0.85	0.77	0.51	0.58	0.58	1.27

After removing coffee and bread, the support measures are lower for the top items. One that for all items support measures when in a pair with other item are much smaller.
Lift measures seem to fall into two categories - one around 1.15, and other much lower at around 0.6.

Dissimilarity

To investigate the low support of pairs of items we can look at the dissimilarity matrix.

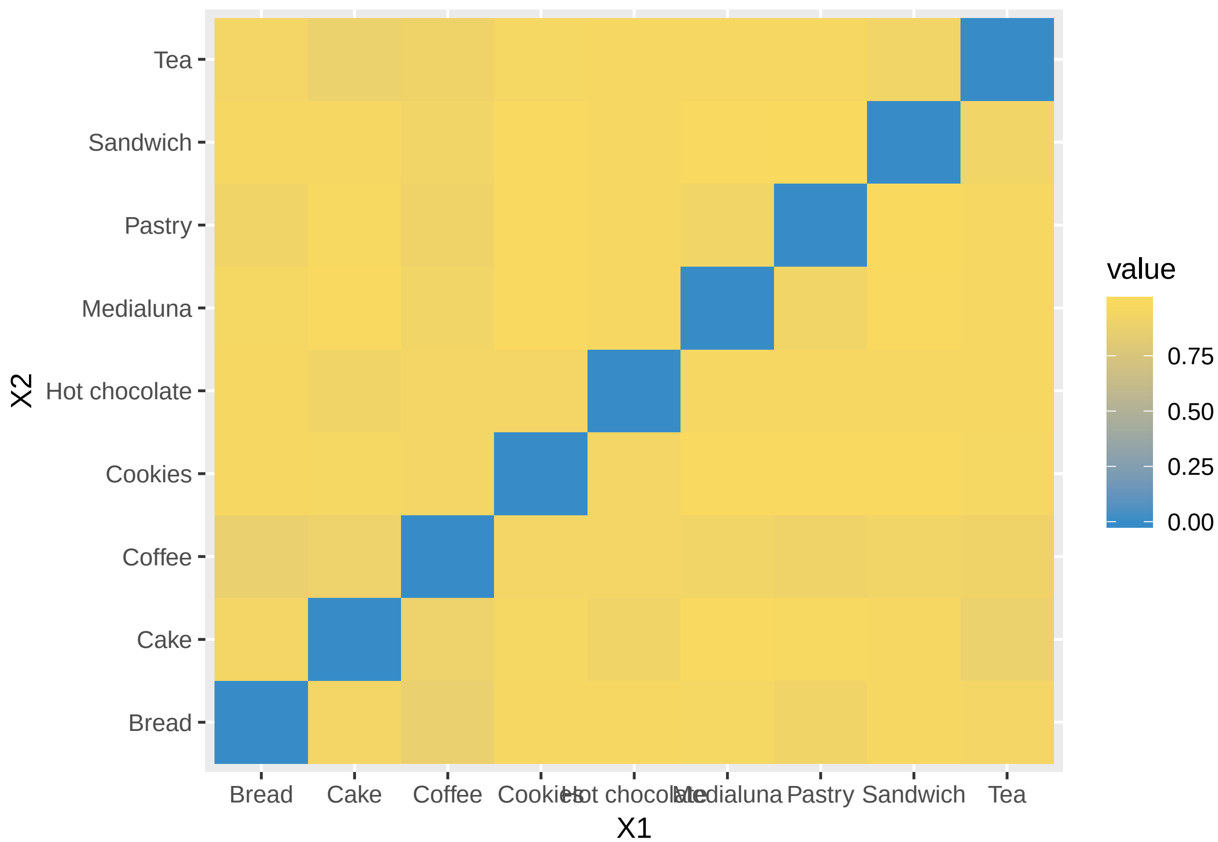
```r
coffeeDiss <- coffee[,itemFrequency(coffee)>0.05] %>%
  dissimilarity(which="items") %>%
  round(2) %>%
  as.matrix()
```

```r
kable(coffeeDiss)
```

	Bread	Cake	Coffee	Cookies	Hot chocolate	Medialuna	Pastry	Sandwich	Tea
Bread	0.00	0.94	0.87	0.96	0.96	0.95	0.92	0.96	0.94

	Bread	Cake	Coffee	Cookies	Hot chocolate	Medialuna	Pastry	Sandwich	Tea
Cake	0.94	0.00	0.90	0.95	0.92	0.98	0.97	0.96	0.89
Coffee	0.87	0.90	0.00	0.94	0.94	0.93	0.91	0.93	0.91
Cookies	0.96	0.95	0.94	0.00	0.94	0.98	0.98	0.98	0.95
Hot chocolate	0.96	0.92	0.94	0.94	0.00	0.96	0.96	0.96	0.96
Medialuna	0.95	0.98	0.93	0.98	0.96	0.00	0.93	0.98	0.96
Pastry	0.92	0.97	0.91	0.98	0.96	0.93	0.00	0.99	0.96
Sandwich	0.96	0.96	0.93	0.98	0.96	0.98	0.99	0.00	0.93
Tea	0.94	0.89	0.91	0.95	0.96	0.96	0.96	0.93	0.00

```
coffeeDiss %>%
  melt() %>%
  ggplot(aes(X1, X2, fill = value)) +
  geom_tile() +
  scale_fill_gradient(low = cBlue,  high = cYellow)
```



As was observed earlier, for most items the dissimilarity measures are very high - around 90%. This means that they only occur in one transaction around 10% of the time.

Association rules

Eclat

We can start the mining frequent itemsets using the eclat() function from arules package. We have to set support to a low value because the dataset is quite large, and the items are not grouped into general categories.

```
teaFreqItems <- tea %>% eclat(list(supp=0.0003, maxlen=4))
```

```
Eclat

parameter specification:
 tidLists support minlen maxlen            target   ext
    FALSE  0.0003      1      4 frequent itemsets FALSE

algorithmic control:
 sparse sort verbose
      7   -2    TRUE

Absolute minimum support count: 2

create itemset ...
set transactions ...[90 item(s), 6788 transaction(s)] done [0.00s].
sorting and recoding items ... [77 item(s)] done [0.00s].
creating sparse bit matrix ... [77 row(s), 6788 column(s)] done [0.00s].
writing  ... [751 set(s)] done [0.00s].
Creating S4 object  ... done [0.00s].
```

```
median(teaFreqItems@quality$count)
```

```
[1] 6
```

Even thought we cet the support treshold at a low value of 0.3%, the median count of the found rules is 6.

Now, we are able to find interesting itemsets using the ruleInduction() function.

```
teaFreqRules <- teaFreqItems %>% ruleInduction(tea, confidence=0.5)
teaFreqRules
```

```
set of 33 rules
```

```
a <- teaFreqRules %>%
  head(20) %>%
  inspect(ruleSep = ">>", itemSep = " + ", setStart = "", setEnd ="") %>%
  as.data.frame()
```

```
kable(a, digits = 4)
```

	lhs		rhs	support	confidence	lift	itemset
[1]	Victorian Sponge	»	Tea	0.0006	0.5714	2.8732	1
[2]	Chocolates + Juice	»	Hot chocolate	0.0004	0.7500	9.2228	8
[3]	Chocolates + Hot chocolate	»	Juice	0.0004	0.7500	13.9479	8
[4]	Mineral water + Pick and Mix Bowls	»	Juice	0.0004	1.0000	18.5973	13
[5]	Juice + Pick and Mix Bowls	»	Mineral water	0.0004	0.7500	37.9925	13
[6]	Duck egg	»	Tea	0.0009	0.5000	2.5141	18
[7]	Duck egg	»	Spanish Brunch	0.0009	0.5000	19.7326	19
[8]	Extra Salami or Feta + Sandwich	»	Salad	0.0004	0.5000	34.2828	73
[9]	Extra Salami or Feta + Juice	»	Salad	0.0004	0.7500	51.4242	74

	lhs		rhs	support	confidence	lift	itemset
[10]	Art Tray + Hot chocolate	»	Juice	0.0004	0.5000	9.2986	85
[11]	Art Tray + Hot chocolate	»	Tea	0.0004	0.5000	2.5141	86
[12]	Cookies + Hearty & Seasonal	»	Hot chocolate	0.0004	1.0000	12.2971	144
[13]	Chicken Stew + Salad	»	Truffles	0.0004	0.7500	26.5156	177
[14]	Jammie Dodgers + Truffles	»	Cake	0.0004	0.7500	5.1790	228
[15]	Jammie Dodgers + Medialuna	»	Tea	0.0006	0.5714	2.8732	233
[16]	Tiffin + Toast	»	Cookies	0.0004	0.5000	6.5903	283
[17]	Scone + Tiffin	»	Tea	0.0007	0.7143	3.5915	284
[18]	Pastry + Tiffin	»	Tea	0.0007	0.7143	3.5915	289
[19]	Coke + Mineral water	»	Sandwich	0.0009	0.5455	5.4449	307
[20]	Scone + Truffles	»	Mineral water	0.0006	0.5000	25.3284	310

We found 33 rules. Their suppor is quite low, but the overall confidence is quite decent, at about 50% to 75%. All rules have significant lift, from 2 to even around 20.

Apriori analisys

Association rules minig can also be done using apriori analisys. Arules provides a function for this. The support is set like previously, at a low level of 0.4%.

```
b <- tea %>%
  apriori(list(supp=0.0004, conf=0.3), control=list(verbose=F)) %>%
  sort(by="lift", decreasing=TRUE) %>%
  head(10) %>%
  inspect(ruleSep = ">>", itemSep = " + ", setStart = "", setEnd ="") %>%
  as.data.frame()
```

```
kable(b, digits = 4)
```

	lhs		rhs	support	confidence	lift	count
[1]	Juice + Salad	»	Extra Salami or Feta	0.0004	0.3000	53.5895	3
[2]	Extra Salami or Feta + Juice	»	Salad	0.0004	0.7500	51.4242	3
[3]	Juice + Pick and Mix Bowls	»	Mineral water	0.0004	0.7500	37.9925	3
[4]	Extra Salami or Feta + Sandwich	»	Salad	0.0004	0.5000	34.2828	3
[5]	Cake + Extra Salami or Feta	»	Salad	0.0004	0.4286	29.3853	3
[6]	Extra Salami or Feta	»	Salad	0.0024	0.4211	28.8698	16
[7]	Chicken Stew + Salad	»	Truffles	0.0004	0.7500	26.5156	3
[8]	Extra Salami or Feta + Spanish Brunch	»	Salad	0.0004	0.3750	25.7121	3
[9]	Scone + Truffles	»	Mineral water	0.0006	0.5000	25.3284	4
[10]	Salad + Truffles	»	Chicken Stew	0.0004	0.4286	23.6516	3

Uncovered rules are simmilar to previous analisys, but there are some differences. The very high lift values of around 50 are all acheived by relations that occur in a very limited number of cases. The one excerption if *Extra Salami or Feta*, which strongly influences *Salad*.

LHS

Using apriori, we can take a look at what items result in the choice of popular products, coffee and bread.

Coffee

```
c <- coffee %>%
  apriori(list(supp=0.001,conf = 0.10),
          appearance=list(default="lhs", rhs="Coffee"),
          control=list(verbose=F)) %>%
  sort(by="confidence", decreasing=TRUE) %>%
  head(10) %>%
  inspect(ruleSep = ">>", itemSep = " + ", setStart = "", setEnd ="") %>%
  as.data.frame()
```

```
kable(c, digits = 4)
```

	lhs		rhs	support	confidence	lift	count
[1]	Extra Salami or Feta + Salad	»	Coffee	0.0015	0.8750	1.8288	14
[2]	Pastry + Toast	»	Coffee	0.0014	0.8667	1.8114	13
[3]	Hearty & Seasonal + Sandwich	»	Coffee	0.0013	0.8571	1.7915	12
[4]	Cake + Vegan mincepie	»	Coffee	0.0011	0.8333	1.7418	10
[5]	Salad + Sandwich	»	Coffee	0.0016	0.8333	1.7418	15
[6]	Extra Salami or Feta	»	Coffee	0.0033	0.8158	1.7051	31
[7]	Keeping It Local	»	Coffee	0.0054	0.8095	1.6920	51
[8]	Cookies + Scone	»	Coffee	0.0016	0.7895	1.6501	15
[9]	Juice + Pastry	»	Coffee	0.0018	0.7727	1.6151	17
[10]	Cake + Salad	»	Coffee	0.0011	0.7692	1.6078	10

We uncovered some rules with high confidence - most of them consist of typical breakfest items, such as salads, toast or sandwiches. This may be some indication that the cross-selling works well.

Bread

```
d <- coffee %>%
  apriori(list(supp=0.001,conf = 0.10),
          appearance=list(default="lhs", rhs="Bread"),
          control=list(verbose=F)) %>%
  sort(by="confidence", decreasing=TRUE) %>%
  head(10) %>%
  inspect(ruleSep = ">>", itemSep = " + ", setStart = "", setEnd ="") %>%
  as.data.frame()
```

```
kable(d)
```

	lhs		rhs	support	confidence	lift	count
[1]	Cake + Jammie Dodgers	»	Bread	0.0015850	0.5172414	1.580618	15
[2]	Eggs	»	Bread	0.0014793	0.5000000	1.527930	14
[3]	Jammie Dodgers + Tea	»	Bread	0.0010566	0.4166667	1.273275	10
[4]	Hot chocolate + Scone	»	Bread	0.0011623	0.3928571	1.200517	11
[5]	Alfajores + Brownie	»	Bread	0.0010566	0.3703704	1.131800	10
[6]	Alfajores + Medialuna	»	Bread	0.0011623	0.3666667	1.120482	11
[7]	Tea + Tiffin	»	Bread	0.0012680	0.3636364	1.111222	12
[8]	Jammie Dodgers	»	Bread	0.0046492	0.3520000	1.075663	44
[9]	Focaccia	»	Bread	0.0020076	0.3518519	1.075210	19
[10]	Pastry	»	Bread	0.0291631	0.3386503	1.034868	276

The results for bread are different - the overall confidence is much lower, and the product are mostly from the take-away category.

RHS

To complete the analisys, we can take a look at what items the coffee and bread bring to the typical transaction.

Coffee

```
e <- coffee %>%
  apriori(list(supp=0.001,conf = 0.05),
          appearance=list(default="rhs", lhs="Coffee"),
          control=list(verbose=F)) %>%
  sort(by="confidence", decreasing=TRUE) %>%
  head(10) %>%
  inspect(ruleSep = ">>", itemSep = " + ", setStart = "", setEnd ="") %>%
  as.data.frame()
```

```
kable(e)
```

	lhs		rhs	support	confidence	lift	count
[1]		»	Bread	0.3272401	0.3272401	1.0000000	3097
[2]	Coffee	»	Bread	0.0900254	0.1881625	0.5749985	852
[3]		»	Tea	0.1426458	0.1426458	1.0000000	1350
[4]	Coffee	»	Cake	0.0547337	0.1143993	1.1013987	518
[5]	Coffee	»	Tea	0.0498732	0.1042403	0.7307630	472
[6]		»	Cake	0.1038673	0.1038673	1.0000000	983
[7]	Coffee	»	Pastry	0.0475486	0.0993816	1.1540463	450
[8]		»	Pastry	0.0861158	0.0861158	1.0000000	815
[9]	Coffee	»	Sandwich	0.0382502	0.0799470	1.1126741	362
[10]	Coffee	»	Medialuna	0.0351860	0.0735424	1.1897527	333

Confidence is low - judging by lift, it seems that buying coffee signicantly decreases the chance of buing bread and tea. It does increase the chance for cakes, pastry and sandwiches.

Bread

```
f <- coffee %>%
  apriori(list(supp=0.001,conf = 0.05),
          appearance=list(default="rhs", lhs="Bread"),
          control=list(verbose=F)) %>%
  sort(by="confidence", decreasing=TRUE) %>%
  head(10) %>%
  inspect(ruleSep = ">>", itemSep = " + ", setStart = "", setEnd ="") %>%
  as.data.frame()
```

```
kable(f)
```

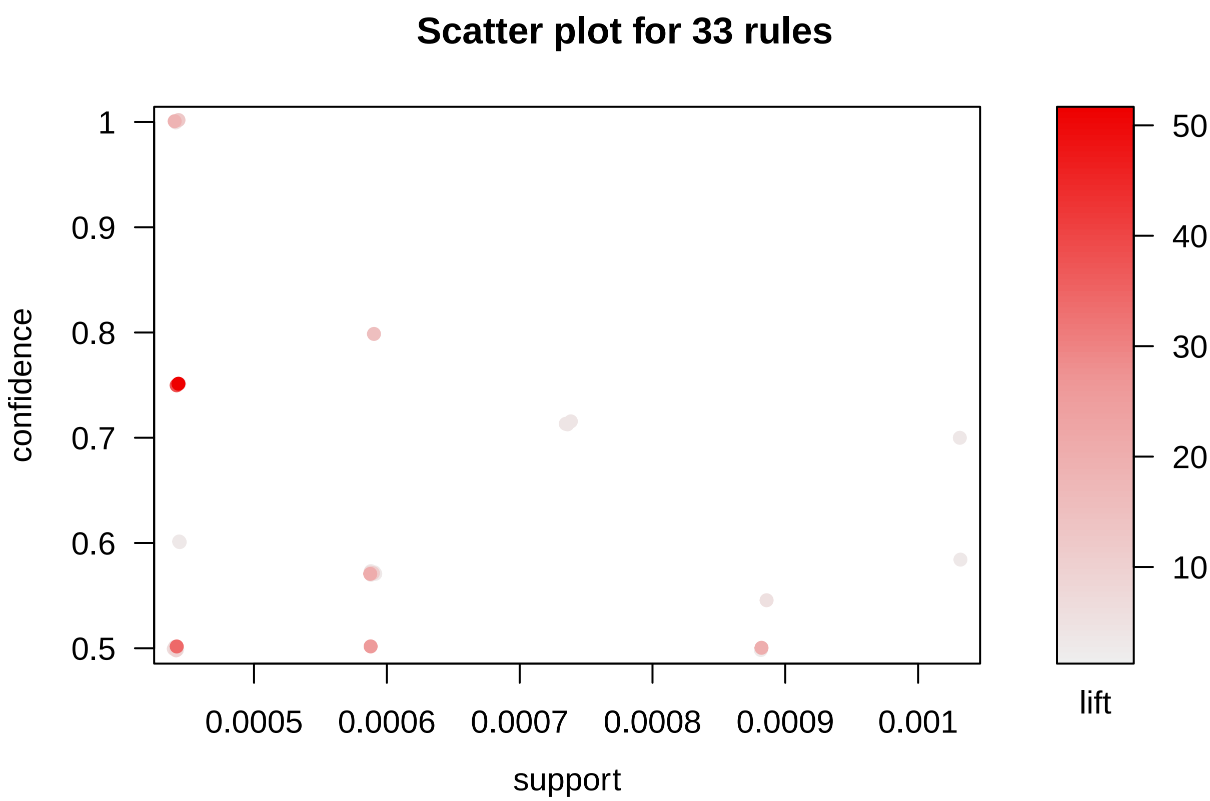
	lhs		rhs	support	confidence	lift	count
[1]		»	Coffee	0.4784446	0.4784446	1.0000000	4528
[2]	Bread	»	Coffee	0.0900254	0.2751049	0.5749985	852
[3]		»	Tea	0.1426458	0.1426458	1.0000000	1350

	lhs		rhs	support	confidence	lift	count
[4]		»	Cake	0.1038673	0.1038673	1.0000000	983
[5]	Bread	»	Pastry	0.0291631	0.0891185	1.0348681	276
[6]		»	Pastry	0.0861158	0.0861158	1.0000000	815
[7]	Bread	»	Tea	0.0281065	0.0858896	0.6021177	266
[8]		»	Sandwich	0.0718512	0.0718512	1.0000000	680
[9]	Bread	»	Cake	0.0233516	0.0713594	0.6870246	221
[10]		»	Medialuna	0.0618132	0.0618132	1.0000000	585

Confidence here is also low. Buying bread seems to decrease the chance of buing coffee, tea and cakes. It does increase the chance slightly for pastry.

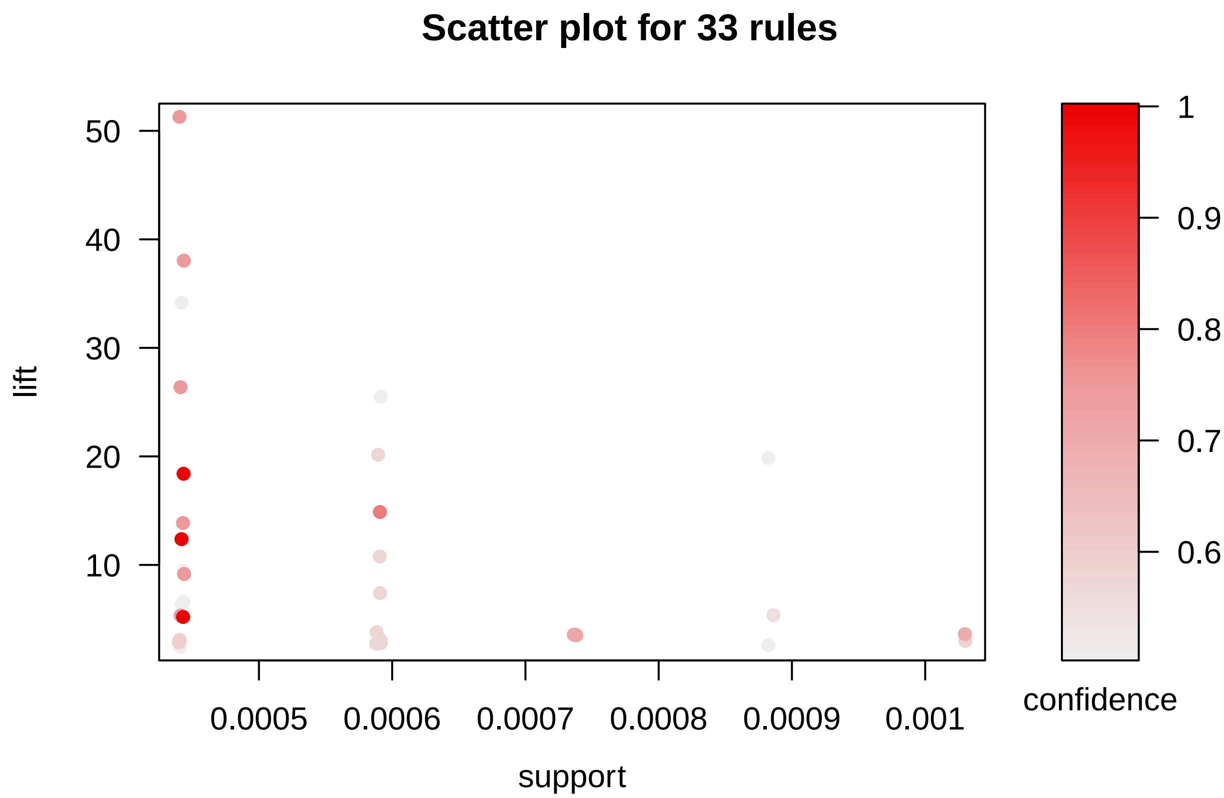
Visulalization

```
plot(teaFreqRules)
```



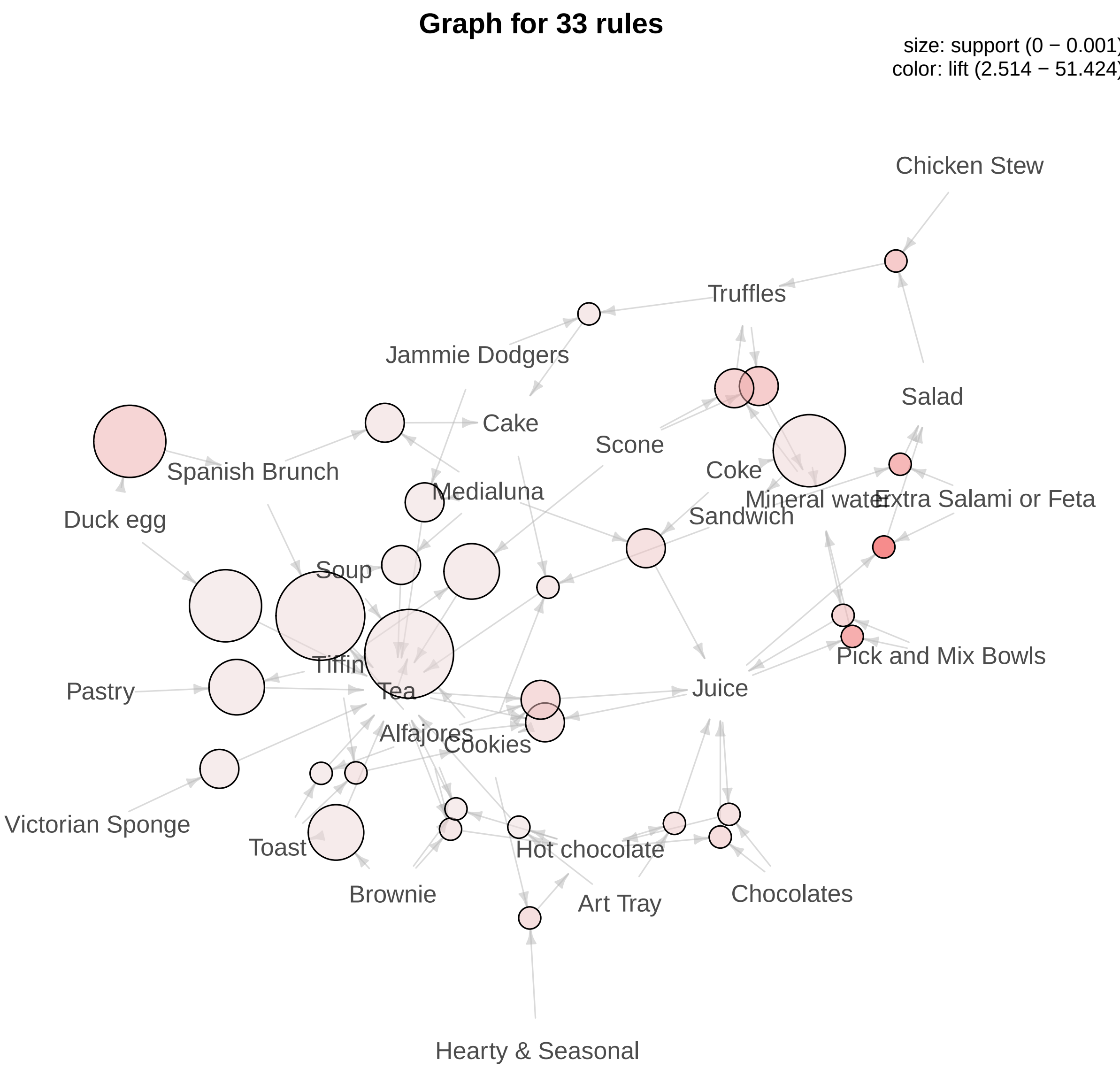
Plotting support vs confidence shows a weak negative correlation. In the rules that were discovered, there are no that have both high support and high confidence.

```r
plot(teaFreqRules, measure=c("support","lift"), shading="confidence")
```



The plot of support vs lift confirms a previous observation - the very high values of lift occur only for small values of support.

```r
plot(teaFreqRules, method="graph")
```



Graphing the results gives some new insights - Tea and Juice have by far the most connections. Tea is understandable, since, after removing coffee and bread, it is the most popular item. Juice on the other hand is interesting, because it occurs much less frequently. The high number of connections may be due to the fact that it is a drink.

Of note is a connction between *Duck Egg* and *Spanish Brunch*, that hase both quite high lift, as well as support.

Conslusions

The results of the analisys are inconclusive. There seems to be some weak evidence that up-selling coffee to customers buing other products works well, while upselling other products to customers buying coffee only decreases the probability.

Customers like to buy bread in addition to other products, but buing bread seems to decrease the probability of buing other popular items.

After removing coffee and bread from the dataset, we were able to find some interesting relations that could be used to set up some custom upselling promotions, e.g. *buy Victorian Sponge and get tea 50% off*.

Because the dataset consist of many individual items (> 90), the average support is very low. This could be imporved by manually assigning all items to 5-10 broad categories, and redoing the analisys. The process of categorisation should be ideally consulted with the owners of the bakery, to make sure it align with their business goals.