Association

Daniel Ryba Zanardini de Oliveira

Association utilities

Why?

In retail the most of purchases are bought on impulse. Some analysis provides clues as to what a customer might have bought for "impulse". This information can be used to decide the location and promotion of goods within a store.

If a rule is found in some store and it is not observed in another (equal stores), it is a point of attention. It may be a store that has different customers or that the organization of the products is proposing a "more profitable" way. This type of research can generate profitable insights.

It is important to realize that there are many areas in which it can be applied: Analysis of credit card purchases. Analysis of telephone calling patterns. Identification of fraudulent medical insurance claims. Analysis of telecom service purchases and others.

Introduction

In this report I use rules of association to analize a market basket.

I will use an Apriori Algorithm to do Market Basket Analysis of Customers purchasing behaviours. The goal is to predict what the customer will buy by looking at the products he is buying.

An Eclat algorithm will be used to provide a list of the most often bought together.

Data set: Market Basket Optimization (Data Mining for Grocery Stores) - https://www.kaggle.com/roshansharma/market-basket-optimization/data

Data set

```
# directory
setwd("C:\\Users\\Daniel\\Documents\\BI - MASTER (MBA)\\DM\\AULA_05_ASSOCIACAO")

# load data
df <- read.csv('Market_Basket_Optimisation.csv', header = FALSE)
```

```
# look for data
str(df)
```

```
## 'data.frame':    7501 obs. of  20 variables:
##  $ V1 : chr  "shrimp" "burgers" "chutney" "turkey" ...
##  $ V2 : chr  "almonds" "meatballs" "" "avocado" ...
##  $ V3 : chr  "avocado" "eggs" "" "" ...
##  $ V4 : chr  "vegetables mix" "" "" "" ...
##  $ V5 : chr  "green grapes" "" "" "" ...
##  $ V6 : chr  "whole weat flour" "" "" "" ...
```

```
##  $ V7 : chr  "yams" "" "" "" ...
##  $ V8 : chr  "cottage cheese" "" "" "" ...
##  $ V9 : chr  "energy drink" "" "" "" ...
##  $ V10: chr  "tomato juice" "" "" "" ...
##  $ V11: chr  "low fat yogurt" "" "" "" ...
##  $ V12: chr  "green tea" "" "" "" ...
##  $ V13: chr  "honey" "" "" "" ...
##  $ V14: chr  "salad" "" "" "" ...
##  $ V15: chr  "mineral water" "" "" "" ...
##  $ V16: chr  "salmon" "" "" "" ...
##  $ V17: chr  "antioxydant juice" "" "" "" ...
##  $ V18: chr  "frozen smoothie" "" "" "" ...
##  $ V19: chr  "spinach" "" "" "" ...
##  $ V20: chr  "olive oil" "" "" "" ...
```

Apriori

The Apriori algorithm was proposed by Agrawal and Srikant in 1994. Apriori is designed to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation or IP addresses). Each transaction is seen as a set of items (an itemset). Given a threshold x, the Apriori algorithm identifies the item sets which are subsets of at least x transactions in the database.

In another words, frequent itemsets are sets of items that appear together at least less than $x\%$ of transactions. The number x witch needs to be provided to the algorithm is called support data.

For example, the rule of association: A $\Rightarrow$ B where A, B, C and D are items. The confidence (c) is the proportion of transactions that include A and also include B. So if the rule confidence is 60%, it can be read as 60% of the people who bought A also bought B.

$confidence(A \Rightarrow B) = \frac{suport(AB)}{suport(A)}$

Example

	coffee	no coffee	total
tea	150	50	200
no tea	650	150	800
total	800	200	1000

- the rule $tea \Rightarrow coffe$ have support(s) $s = \frac{(150)}{(1000)} = 0.15$
- the confidence of this rule is $c = \frac{(0.15)}{(0.2)} = 0.75$
- but the rule is NOT interesting because the rule $\{\} \Rightarrow coffee$ have a confidence 80% (80% of people already drink coffee). The fact that someone drinks tea decreases the probability of drinking coffee!

Lift measures how much the two sides of the rule are not independent.

$lift(A \Rightarrow B) = \frac{P(AB)}{P(A)P(B)}$

- lift = 1 indicate that A and B are independent (bad).
- lift > 1 indicate a positive correlation between A and B.

Back to our study

```
# install.packages('arules')
library(arules)
```

```
# sparse matrix
dataset = read.transactions('Market_Basket_Optimisation.csv', sep = ',',
                            rm.duplicates = TRUE) #removes duplicate entries in each transaction
```

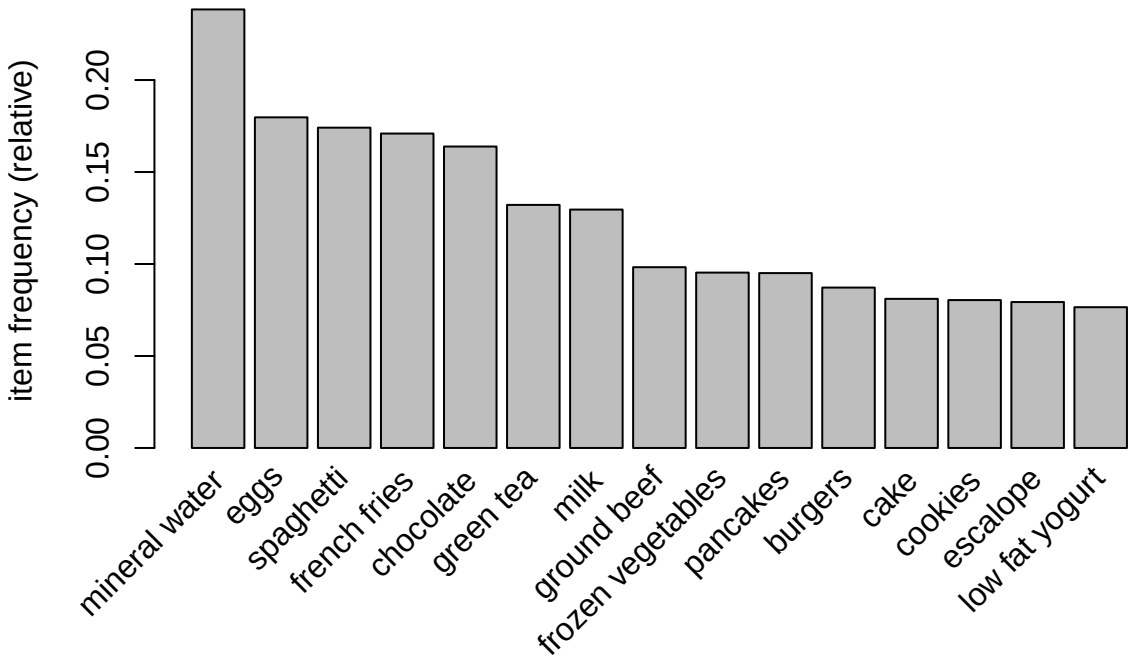
```
## distribution of transactions with duplicates:
## 1
## 5
```

```
summary(dataset) # matrix details
```

```
## transactions as itemMatrix in sparse format with
##  7501 rows (elements/itemsets/transactions) and
##  119 columns (items) and a density of 0.03288973
##
## most frequent items:
## mineral water          eggs     spaghetti  french fries     chocolate
##          1788          1348          1306          1282          1229
##       (Other)
##         22405
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 1754 1358 1044  816  667  493  391  324  259  139  102   67   40   22   17    4
##   18   19   20
##    1    2    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.914   5.000  20.000
##
## includes extended item information - examples:
##              labels
## 1           almonds
## 2 antioxydant juice
## 3         asparagus
```

Now print the 15 most frequent products in the market basket.

```
itemFrequencyPlot(dataset, topN = 15) #topN (choose your number)
```



Create the rules indicating the support and the confidence,

- Support - optimize the sale of products purchased 3 times a week $\frac{(3*7)}{(7501)} = 0.003$.
- Confidence - Low: rules that don't make sense; High: obvious rules. *ATTENTION*, a value of 0.8 means that all rules generated must be correct in 80% of transactions

I put a confidence of 0.2 but we must do several tests and understand the "business" questions.

Visualization 15 rules created, ordered by lift.

```
# visualization
knitr::kable(wide, "markdown", booktabs = T) %>%
kable_styling(font_size = 4)
```

	lhs		rhs	support	confidence	coverage	lift	count
[1]	{mineral water,whole wheat pasta}	=>	{olive oil}	0.0038662	0.4027778	0.0095987	6.115863	29
[2]	{frozen vegetables,milk,mineral water}	=>	{soup}	0.0030663	0.2771084	0.0110652	5.484407	23
[3]	{fromage blanc}	=>	{honey}	0.0033329	0.2450980	0.0135982	5.164271	25
[4]	{spaghetti,tomato sauce}	=>	{ground beef}	0.0030663	0.4893617	0.0062658	4.980600	23
[5]	{light cream}	=>	{chicken}	0.0045327	0.2905983	0.0155979	4.843951	34
[6]	{pasta}	=>	{escalope}	0.0058659	0.3728814	0.0157312	4.700812	44
[7]	{french fries,herb & pepper}	=>	{ground beef}	0.0031996	0.4615385	0.0069324	4.697422	24
[8]	{cereals,spaghetti}	=>	{ground beef}	0.0030663	0.4600000	0.0066658	4.681764	23
[9]	{frozen vegetables,mineral water,soup}	=>	{milk}	0.0030663	0.6052632	0.0050660	4.670863	23

	lhs		rhs	support	confidence	coverage	lift	count
[10]	{french fries,ground beef}	=>	{herb & pepper}	0.0031996	0.2307692	0.0138648	4.665768	24
[11]	{chocolate,frozen vegetables,mineral water}	=>	{shrimp}	0.0031996	0.3287671	0.0097320	4.600900	24
[12]	{frozen vegetables,milk,mineral water}	=>	{olive oil}	0.0033329	0.3012048	0.0110652	4.573557	25
[13]	{pasta}	=>	{shrimp}	0.0050660	0.3220339	0.0157312	4.506672	38
[14]	{chocolate,herb & pepper}	=>	{ground beef}	0.0039995	0.4411765	0.0090655	4.490183	30
[15]	{chocolate,mineral water,shrimp}	=>	{frozen vegetables}	0.0031996	0.4210526	0.0075990	4.417225	24

In this table we can check the rules found. The rules are described, for example: *mineral water, whole wheat pasta* $\Rightarrow$ *olive oil* (the support, the confidence, the lift, coverage and the number of times that appears in the base).

Another way to help the analysis is utilize an algorithm that shows the frequency of which products are purchased together.

ECLAT

Eclat(alt. ECLAT, stands for Equivalence Class Transformation) is a depth-first search algorithm based on set intersection. Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node (selecting some arbitrary node as the root node in the case of a graph) and explores as far as possible along each branch before backtracking.

```
# create rules of frequency
rules2 = eclat(data = dataset, parameter = list(support = 0.003, minlen = 2)) # minimum number of produ
```

```
## Eclat
##
## parameter specification:
##  tidLists support minlen maxlen            target  ext
##     FALSE   0.003      2     10 frequent itemsets TRUE
##
## algorithmic control:
##  sparse sort verbose
##       7   -2    TRUE
##
## Absolute minimum support count: 22
##
## create itemset ...
## set transactions ...[119 item(s), 7501 transaction(s)] done [0.00s].
## sorting and recoding items ... [115 item(s)] done [0.00s].
## creating sparse bit matrix ... [115 row(s), 7501 column(s)] done [0.00s].
## writing  ... [1328 set(s)] done [0.01s].
## Creating S4 object  ... done [0.00s].
```

Visualization 15 rules created, ordered by support.

```
# visualization
knitr::kable(wide2, "markdown", booktabs = T) %>%
kable_styling(font_size = 4)
```

	items	support	transIdenticalToItemsets	count
[1]	{mineral water,spaghetti}	0.0597254	448	448
[2]	{chocolate,mineral water}	0.0526596	395	395
[3]	{eggs,mineral water}	0.0509265	382	382
[4]	{milk,mineral water}	0.0479936	360	360
[5]	{ground beef,mineral water}	0.0409279	307	307
[6]	{ground beef,spaghetti}	0.0391948	294	294
[7]	{chocolate,spaghetti}	0.0391948	294	294
[8]	{eggs,spaghetti}	0.0365285	274	274
[9]	{eggs,french fries}	0.0363951	273	273
[10]	{frozen vegetables,mineral water}	0.0357286	268	268
[11]	{milk,spaghetti}	0.0354619	266	266
[12]	{chocolate,french fries}	0.0343954	258	258
[13]	{mineral water,pancakes}	0.0337288	253	253
[14]	{french fries,mineral water}	0.0337288	253	253
[15]	{chocolate,eggs}	0.0331956	249	249

In this table we can check the associations found. We can see the most purchased products together.

Conclusion

These analyzes are quite interesting, since with them you can evaluate the operational organization, marketing, among others. Understanding the business problems we can, with the help of this information, optimize the business making some improvements.

In this example, we can see products that are purchased according to predetermined rules and products that are frequently purchased together.