Market Basket Analysis

Code

Xavier Vivancos García

2019-08-02

1 Introduction

Hi! In this kernel we are going to use the **Apriori algorithm** to perform a **Market Basket Analysis**. A Market what? Is a technique used by large retailers to uncover associations between items. It works by looking for combinations of items that occur together frequently in transactions, providing information to understand the purchase behavior. The outcome of this type of technique is, in simple terms, a set of **rules** that can be understood as **"if this, then that"**. For more information about these topics, please check in the following links:

- Market Basket Analysis (https://en.wikipedia.org/wiki/Affinity_analysis)
- Apriori algorithm (https://en.wikipedia.org/wiki/Apriori_algorithm)
- Association rule learning (https://en.wikipedia.org/wiki/Association_rule_learning)

First it's important to define the Apriori algorithm, including some statistical concepts (support, confidence, lift and conviction) to select interesting rules. Then we are going to use a data set containing more than 6.000 transactions from a bakery to apply the algorithm and find combinations of products that are bought together. Let's start!

2 Association rules

The Apriori algorithm generates association rules for a given data set. An association rule implies that if an item A occurs, then item B also occurs with a certain probability. Let's see an example,

Transaction	Items
t1	{T-shirt, Trousers, Belt}

Transaction	Items
t2	{T-shirt, Jacket}
t3	{Jacket, Gloves}
t4	{T-shirt, Trousers, Jacket}
t5	{T-shirt, Trousers, Sneakers, Jacket, Belt}
t6	{Trousers, Sneakers, Belt}
t7	{Trousers, Belt, Sneakers}

In the table above we can see seven transactions from a clothing store. Each transaction shows items bought in that transaction. We can represent our items as an **item set** as follows:

$$I = \{i_1, i_2, \ldots, i_k\}$$

In our case it corresponds to:

$$I = \{T\text{-}shirt, Trousers, Belt, Jacket, Gloves, Sneakers\}$$

A **transaction** is represented by the following expression:

$$T = \{t_1, t_2, \ldots, t_n\}$$

For example,

$$t_1 = \{T\text{-}shirt, Trousers, Belt\}$$

Then, an **association rule** is defined as an implication of the form:

$$X \Rightarrow Y, \text{ where } X \subset I,\ Y \subset I \text{ and } X \cap Y = 0$$

For example,

$$\{T\text{-}shirt, Trousers\} \Rightarrow \{Belt\}$$

In the following sections we are going to define four metrics to measure the precision of a rule.

2.1 Support

Support is an indication of how frequently the item set appears in the data set.

$$supp(X \Rightarrow Y) = \frac{|X \cup Y|}{n}$$

In other words, it's the number of transactions with both X and Y divided by the total number of transactions. The rules are not useful for low support values. Let's see different examples using the clothing store transactions from the previous table.

- $supp(T\text{-}shirt \Rightarrow Trousers) = \frac{3}{7} = 43\%$
- $supp(Trousers \Rightarrow Belt) = \frac{4}{7} = 57\%$

- $supp(T\text{-}shirt \Rightarrow Belt) = \frac{2}{7} = 28\%$
- $supp(\{T\text{-}shirt, Trousers\} \Rightarrow \{Belt\}) = \frac{2}{7} = 28\%$

2.2 Confidence

For a rule $X \Rightarrow Y$, confidence shows the percentage in which Y is bought with X. It's an indication of how often the rule has been found to be true.

$$conf(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$$

For example, the rule $T\text{-}shirt \Rightarrow Trousers$ has a confidence of 3/4, which means that for 75% of the transactions containing a t-shirt the rule is correct (75% of the times a customer buys a t-shirt, trousers are bought as well). Three more examples:

- $conf(Trousers \Rightarrow Belt) = \frac{4/7}{5/7} = 80\%$
- $conf(T\text{-}shirt \Rightarrow Belt) = \frac{2/7}{4/7} = 50\%$
- $conf(\{T\text{-}shirt, Trousers\} \Rightarrow \{Belt\}) = \frac{2/7}{3/7} = 66\%$

2.3 Lift

The lift of a rule is the ratio of the observed support to that expected if X and Y were independent, and is defined as

$$lift(X \Rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)supp(Y)}$$

Greater lift values indicate stronger associations. Let's see some examples:

- $lift(T\text{-}shirt \Rightarrow Trousers) = \frac{3/7}{(4/7)(5/7)} = 1.05$
- $lift(Trousers \Rightarrow Belt) = \frac{4/7}{(5/7)(4/7)} = 1.4$
- $lift(T\text{-}shirt \Rightarrow Belt) = \frac{2/7}{(4/7)(4/7)} = 0.875$
- $lift(\{T\text{-}shirt, Trousers\} \Rightarrow \{Belt\}) = \frac{2/7}{(3/7)(4/7)} = 1.17$

2.4 Conviction

The conviction of a rule is defined as

$$conv(X \Rightarrow Y) = \frac{1 - supp(Y)}{1 - conf(X \Rightarrow Y)}$$

It can be interpreted as the ratio of the expected frequency that X occurs without Y if X and Y were independent divided by the observed frequency of incorrect predictions. A high value means that the consequent depends strongly on the antecedent. Let's see some examples:

- $conv(T\text{-}shirt \Rightarrow Trousers) = \frac{1 - 5/7}{1 - 3/4} = 1.14$
- $conv(Trousers \Rightarrow Belt) = \frac{1 - 4/7}{1 - 4/5} = 2.14$
- $conv(T\text{-}shirt \Rightarrow Belt) = \frac{1 - 4/7}{1 - 1/2} = 0.86$
- $conv(\{T\text{-}shirt, Trousers\} \Rightarrow \{Belt\}) = \frac{1 - 4/7}{1 - 2/3} = 1.28$

If you want more information about these measures, please check here (https://en.wikipedia.org/wiki/Association_rule_learning).

3 **Loading Data**

First we need to load some libraries and import our data. We can use the function `read.transactions()` from the `arules` package to create a `transactions` object.

Hide

```
# Load libraries
library(tidyverse)
library(arules)
library(arulesViz)
library(knitr)
library(gridExtra)
library(lubridate)

# Read the data
trans <- read.transactions("../input/BreadBasket_DMS.csv", format="single", c
  ols=c(3,4), sep=",", rm.duplicates=TRUE)
```

Let's get an idea of what we're working with.

3.1 Transaction object 3.2 Summary 3.3 Glimpse 3.4 Structure

Hide

```
# Transaction object
trans
```

```
## transactions in sparse format with
##  6614 transactions (rows) and
##  104 items (columns)
```

4 Data Dictionary

The data set contains 15.010 observations and the following columns,

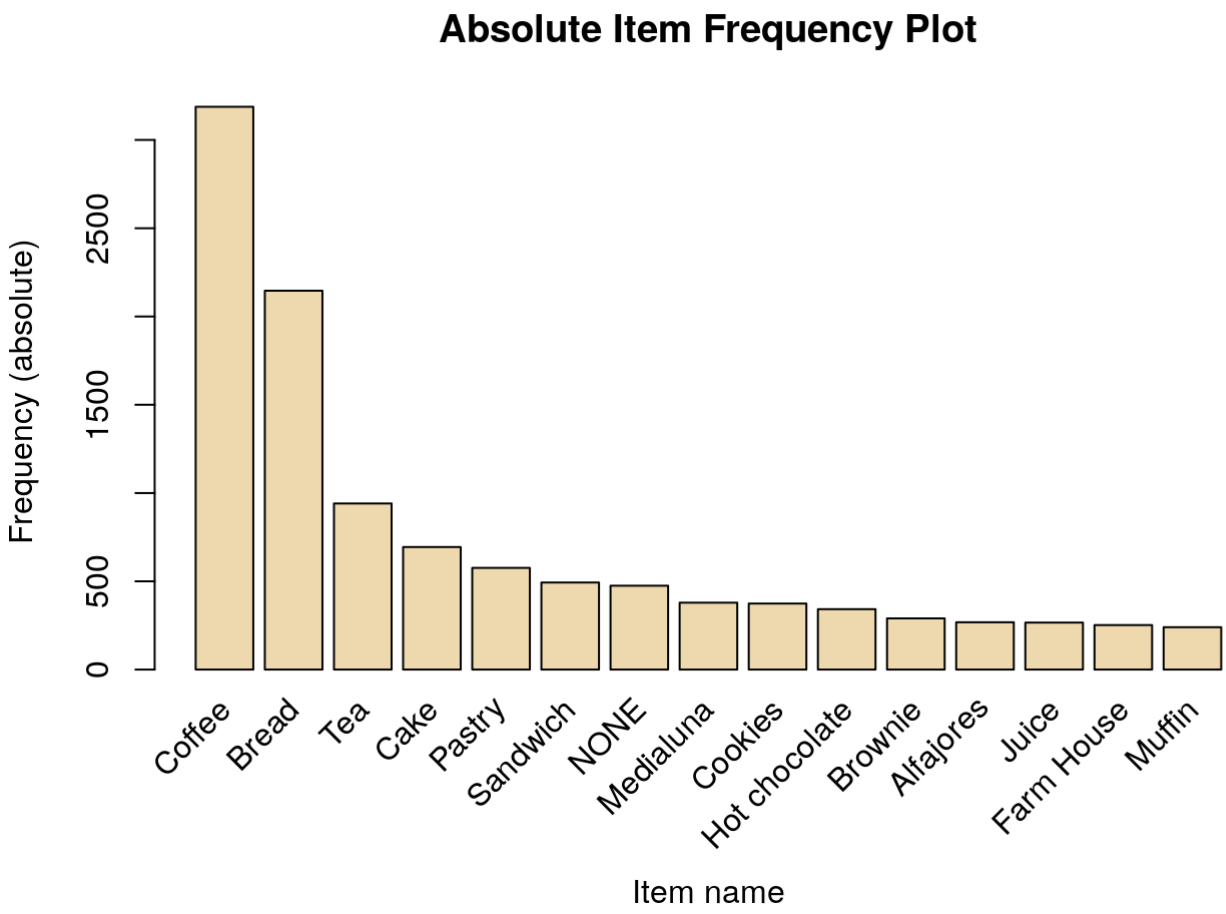
- `Date` . Categorical variable that tells us the date of the transactions (YYYY-MM-DD format). The column includes dates from 30/10/2016 to 09/04/2017.
- `Time` . Categorical variable that tells us the time of the transactions (HH:MM:SS format).
- `Transaction` . Quantitative variable that allows us to differentiate the transactions. The rows that share the same value in this field belong to the same transaction, that's why the data set has less transactions than observations.
- `Item` . Categorical variable containing the products.

5 Data Analysis

Before applying the Apriori algorithm on the data set, we are going to show some visualizations to learn more about the transactions. For example, we can generate an `itemFrequencyPlot()` to create an item Frequency Bar Plot to view the distribution of products.

Hide

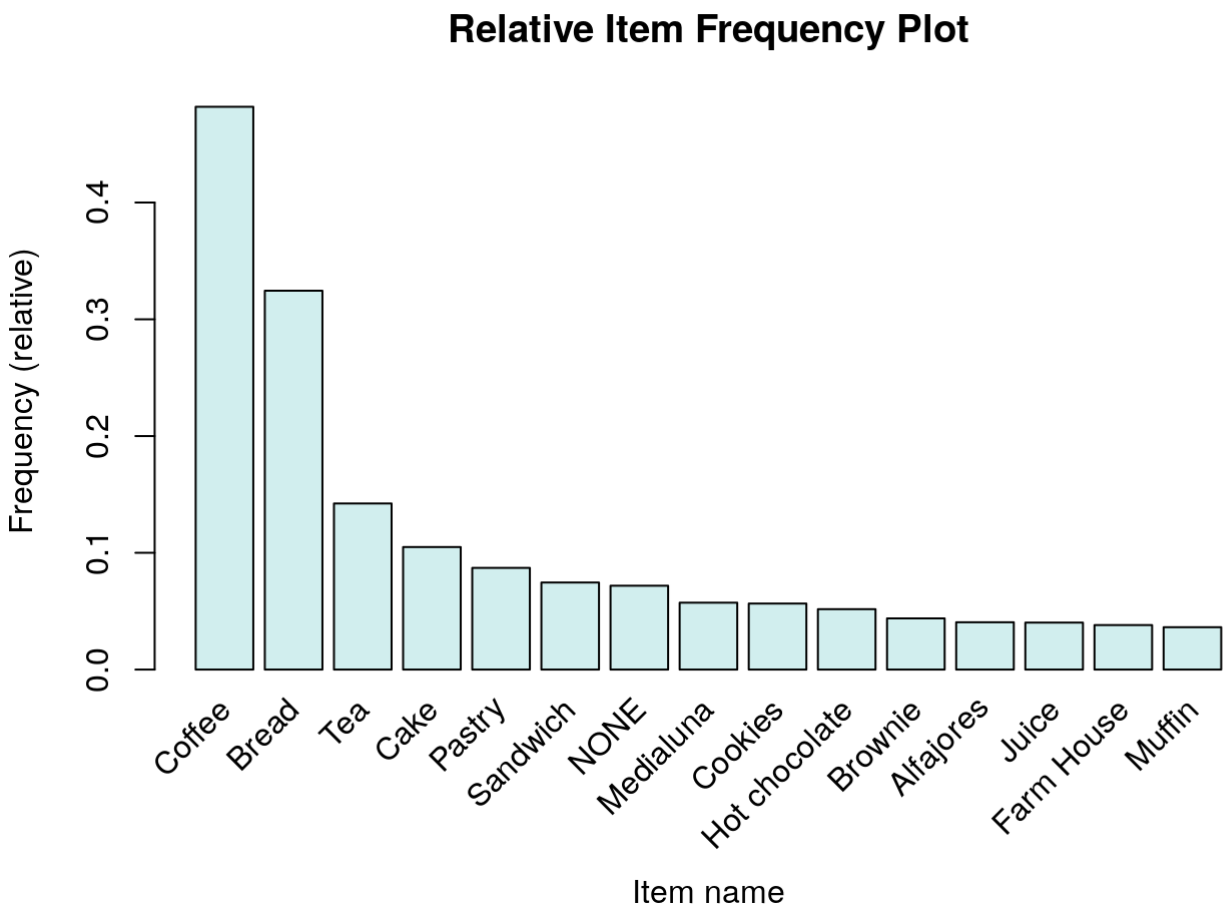
```
# Absolute Item Frequency Plot
itemFrequencyPlot(trans, topN=15, type="absolute", col="wheat2",xlab="Item na
  me",
                  ylab="Frequency (absolute)", main="Absolute Item Frequency
   Plot")
```



The `itemFrequencyPlot()` allows us to show the absolute or relative values. If absolute it will plot numeric frequencies of each item independently. If relative it will plot how many times these items have appeared as compared to others, as it's shown in the following plot.

Hide

```
# Relative Item Frequency Plot
itemFrequencyPlot(trans, topN=15, type="relative", col="lightcyan2", xlab="It
  em name",
                  ylab="Frequency (relative)", main="Relative Item Frequency
   Plot")
```

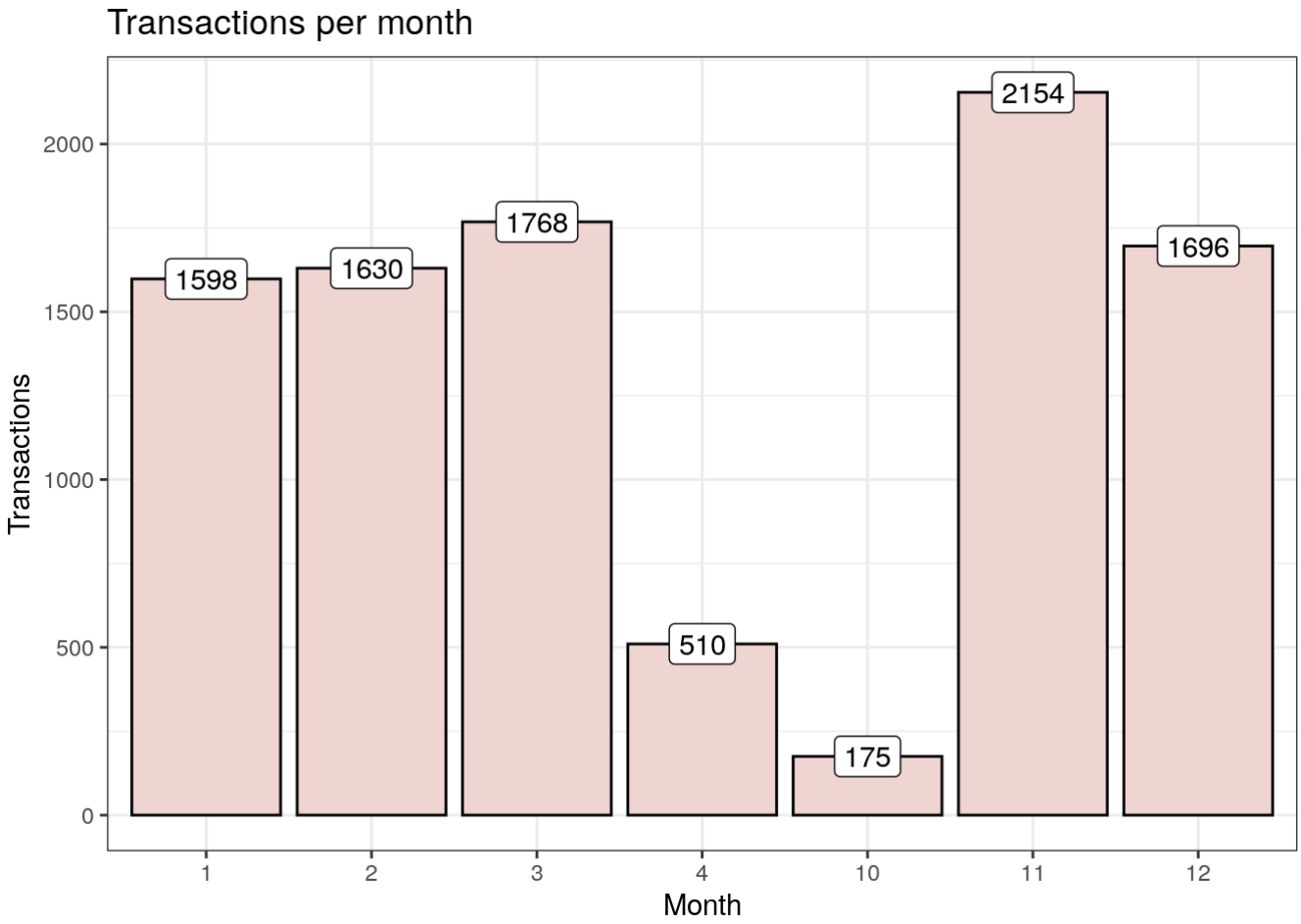


Coffee is the best-selling product by far, followed by bread and tea. Let's display some other visualizations describing the time distribution using the `ggplot()` function.

- Transactions per month
- Transactions per weekday
- Transactions per hour

Hide

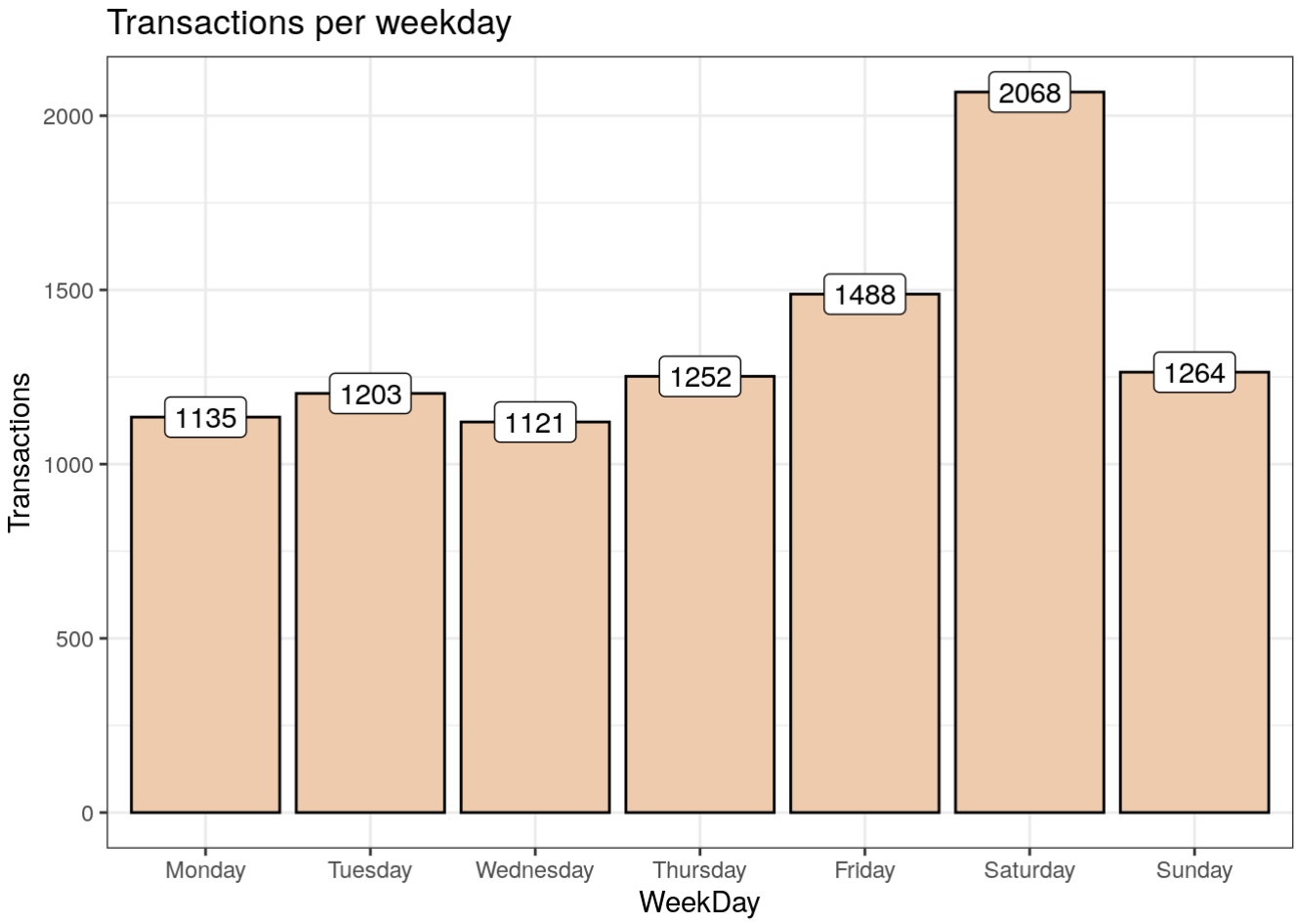
```
# Load data
trans_csv <- read.csv("../input/BreadBasket_DMS.csv")

# Transactions per month
trans_csv %>%
  mutate(Month=as.factor(month(Date))) %>%
  group_by(Month) %>%
  summarise(Transactions=n_distinct(Transaction)) %>%
  ggplot(aes(x=Month, y=Transactions)) +
  geom_bar(stat="identity", fill="mistyrose2", show.legend=FALSE, colour="bla
  ck") +
  geom_label(aes(label=Transactions)) +
  labs(title="Transactions per month") +
  theme_bw()
```



The data set includes dates from 30/10/2016 to 09/04/2017, that's why we have so few transactions in October and April.

Hide

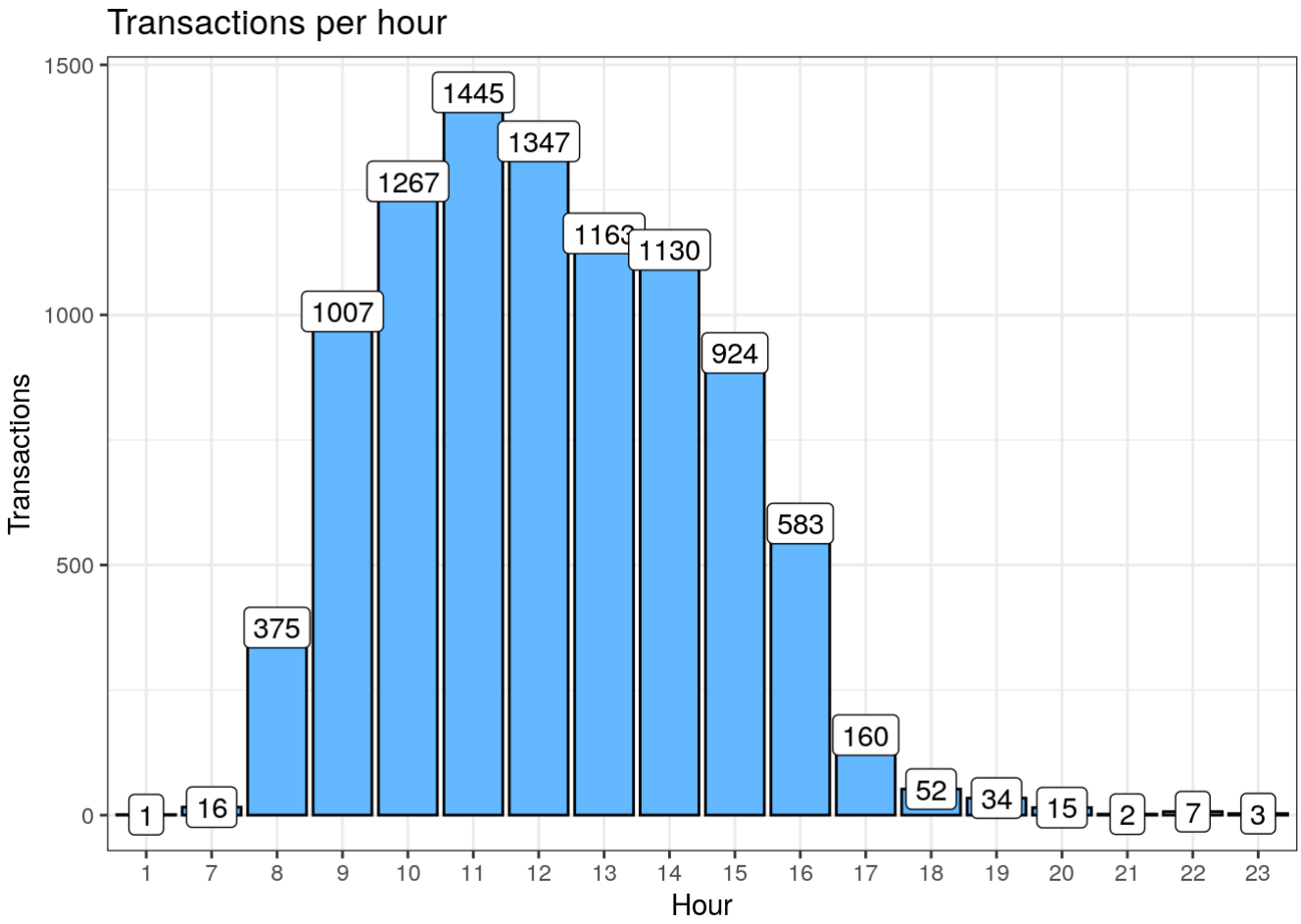
```
# Transactions per weekday
trans_csv %>%
  mutate(WeekDay=as.factor(weekdays(as.Date(Date)))) %>%
  group_by(WeekDay) %>%
  summarise(Transactions=n_distinct(Transaction)) %>%
  ggplot(aes(x=WeekDay, y=Transactions)) +
  geom_bar(stat="identity", fill="peachpuff2", show.legend=FALSE, colour="bla
  ck") +
  geom_label(aes(label=Transactions)) +
  labs(title="Transactions per weekday") +
  scale_x_discrete(limits=c("Monday", "Tuesday", "Wednesday", "Thursday",
                            "Friday", "Saturday", "Sunday")) +
  theme_bw()
```



As we can see, Saturday is the busiest day in the bakery. Conversely, Wednesday is the day with fewer transactions.

Hide

```
# Transactions per hour
trans_csv %>%
  mutate(Hour=as.factor(hour(hms(Time)))) %>%
  group_by(Hour) %>%
  summarise(Transactions=n_distinct(Transaction)) %>%
  ggplot(aes(x=Hour, y=Transactions)) +
  geom_bar(stat="identity", fill="steelblue1", show.legend=FALSE, colour="bla
  ck") +
  geom_label(aes(label=Transactions)) +
  labs(title="Transactions per hour") +
  theme_bw()
```



There’s not much to discuss with this visualization. The results are logical and expected.

6 **Apriori algorithm**

6.1 Choice of support and confidence

The first step in order to create a set of association rules is to determine the optimal thresholds for support and confidence. If we set these values too low, then the algorithm will take longer to execute and we will get a lot of rules (most of them will not be useful). Then, what values do we choose? We can try different values of support and confidence and see graphically how many rules are generated for each combination.

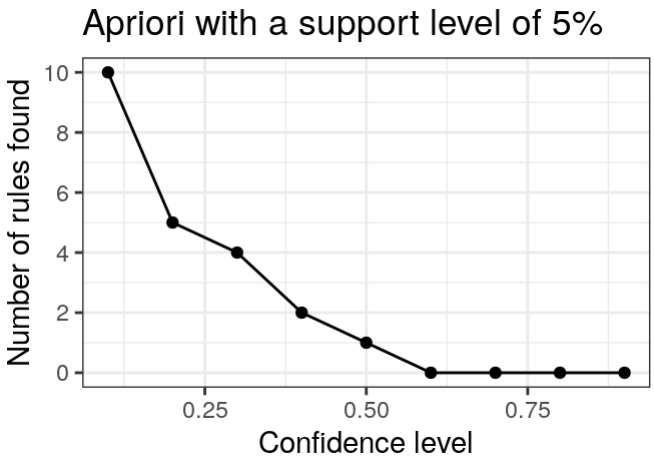
Hide

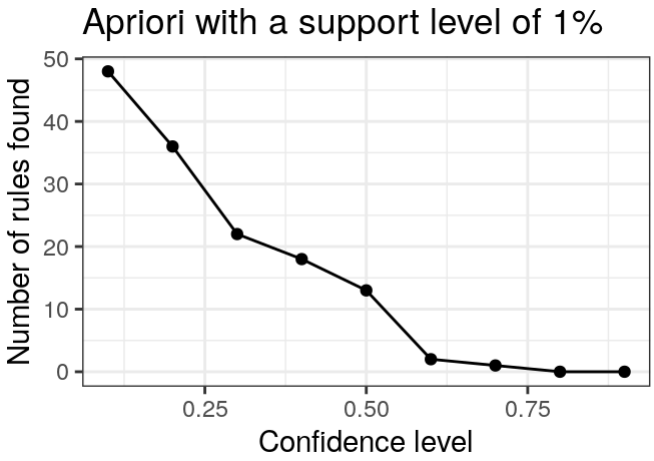
```
# Support and confidence values
supportLevels <- c(0.1, 0.05, 0.01, 0.005)
confidenceLevels <- c(0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1)

# Empty integers
rules_sup10 <- integer(length=9)
rules_sup5 <- integer(length=9)
rules_sup1 <- integer(length=9)
rules_sup0.5 <- integer(length=9)

# Apriori algorithm with a support level of 10%
for (i in 1:length(confidenceLevels)) {

  rules_sup10[i] <- length(apriori(trans, parameter=list(sup=supportLevels[1
  ],
                                   conf=confidenceLevels[i], target="rules"
  )))

}

# Apriori algorithm with a support level of 5%
for (i in 1:length(confidenceLevels)) {

  rules_sup5[i] <- length(apriori(trans, parameter=list(sup=supportLevels[2],
                                  conf=confidenceLevels[i], target="rules")))

}

# Apriori algorithm with a support level of 1%
for (i in 1:length(confidenceLevels)) {

  rules_sup1[i] <- length(apriori(trans, parameter=list(sup=supportLevels[3],
                                  conf=confidenceLevels[i], target="rules")))

}

# Apriori algorithm with a support level of 0.5%
for (i in 1:length(confidenceLevels)) {

  rules_sup0.5[i] <- length(apriori(trans, parameter=list(sup=supportLevels[4
  ],
                                    conf=confidenceLevels[i], target="rules"
  )))

}
```

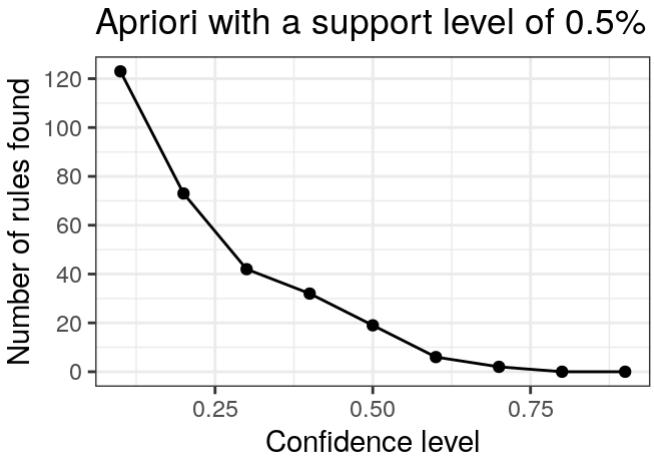
In the following graphs we can see the number of rules generated with a support level of 10%, 5%, 1% and 0.5%.

Hide

```
# Number of rules found with a support level of 10%
plot1 <- qplot(confidenceLevels, rules_sup10, geom=c("point", "line"),
               xlab="Confidence level", ylab="Number of rules found",
               main="Apriori with a support level of 10%") +
  theme_bw()

# Number of rules found with a support level of 5%
plot2 <- qplot(confidenceLevels, rules_sup5, geom=c("point", "line"),
               xlab="Confidence level", ylab="Number of rules found",
               main="Apriori with a support level of 5%") +
  scale_y_continuous(breaks=seq(0, 10, 2)) +
  theme_bw()

# Number of rules found with a support level of 1%
plot3 <- qplot(confidenceLevels, rules_sup1, geom=c("point", "line"),
               xlab="Confidence level", ylab="Number of rules found",
               main="Apriori with a support level of 1%") +
  scale_y_continuous(breaks=seq(0, 50, 10)) +
  theme_bw()

# Number of rules found with a support level of 0.5%
plot4 <- qplot(confidenceLevels, rules_sup0.5, geom=c("point", "line"),
               xlab="Confidence level", ylab="Number of rules found",
               main="Apriori with a support level of 0.5%") +
  scale_y_continuous(breaks=seq(0, 130, 20)) +
  theme_bw()

# Subplot
grid.arrange(plot1, plot2, plot3, plot4, ncol=2)
```

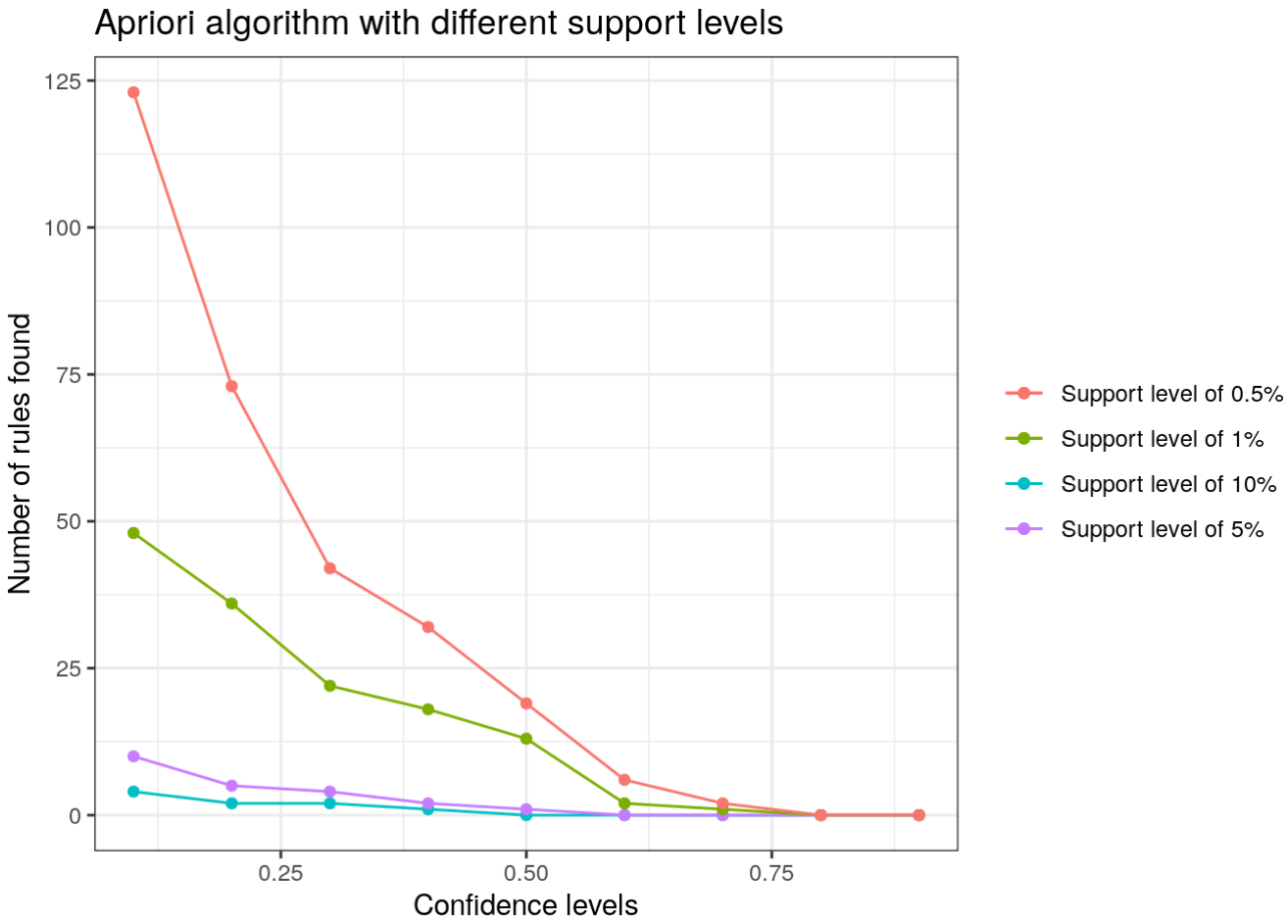






We can join the four lines to improve the visualization.

Code



Let's analyze the results,

- **Support level of 10%**. We only identify a few rules with very low confidence levels. This means that there are no relatively frequent associations in our data set. We can't choose this value, the resulting rules are unrepresentative.
- **Support level of 5%**. We only identify a rule with a confidence of at least 50%. It seems that we have to look for support levels below 5% to obtain a greater number of rules with a reasonable confidence.
- **Support level of 1%**. We started to get dozens of rules, of which 13 have a confidence of at least 50%.
- **Support level of 0.5%**. Too many rules to analyze!

To sum up, we are going to use a support level of 1% and a confidence level of 50%.

6.2 Execution

Let's execute the Apriori algorithm with the values obtained in the previous section.

Hide

```
# Apriori algorithm execution with a support level of 1% and a confidence level
  of 50%
rules_sup1_conf50 <- apriori(trans, parameter=list(sup=supportLevels[3],
                             conf=confidenceLevels[5], target="rules"))
```

The generated association rules are the following,

Hide

```
# Association rules
inspect(rules_sup1_conf50)
```

```
##      lhs                rhs      support    confidence lift     count
## [1]  {Tiffin}         => {Coffee} 0.01058361 0.5468750  1.134577  70
## [2]  {Spanish Brunch} => {Coffee} 0.01406108 0.6326531  1.312537  93
## [3]  {Scone}          => {Coffee} 0.01844572 0.5422222  1.124924 122
## [4]  {Toast}          => {Coffee} 0.02570305 0.7296137  1.513697 170
## [5]  {Alfajores}      => {Coffee} 0.02237678 0.5522388  1.145705 148
## [6]  {Juice}          => {Coffee} 0.02131842 0.5300752  1.099723 141
## [7]  {Hot chocolate}  => {Coffee} 0.02721500 0.5263158  1.091924 180
## [8]  {Medialuna}      => {Coffee} 0.03296039 0.5751979  1.193337 218
## [9]  {Cookies}        => {Coffee} 0.02978530 0.5267380  1.092800 197
## [10] {NONE}           => {Coffee} 0.04172966 0.5810526  1.205484 276
## [11] {Sandwich}       => {Coffee} 0.04233444 0.5679513  1.178303 280
## [12] {Pastry}         => {Coffee} 0.04868461 0.5590278  1.159790 322
## [13] {Cake}           => {Coffee} 0.05654672 0.5389049  1.118042 374
```

We can also create an HTML table widget using the `inspectDT()` function from the `aruslesViz` package. Rules can be interactively filtered and sorted.

How do we interpret these rules?

- 52% of the customers who bought a hot chocolate algo bought a coffee.
- 63% of the customers who bought a spanish brunch also bought a coffee.
- 73% of the customers who bought a toast also bought a coffee.

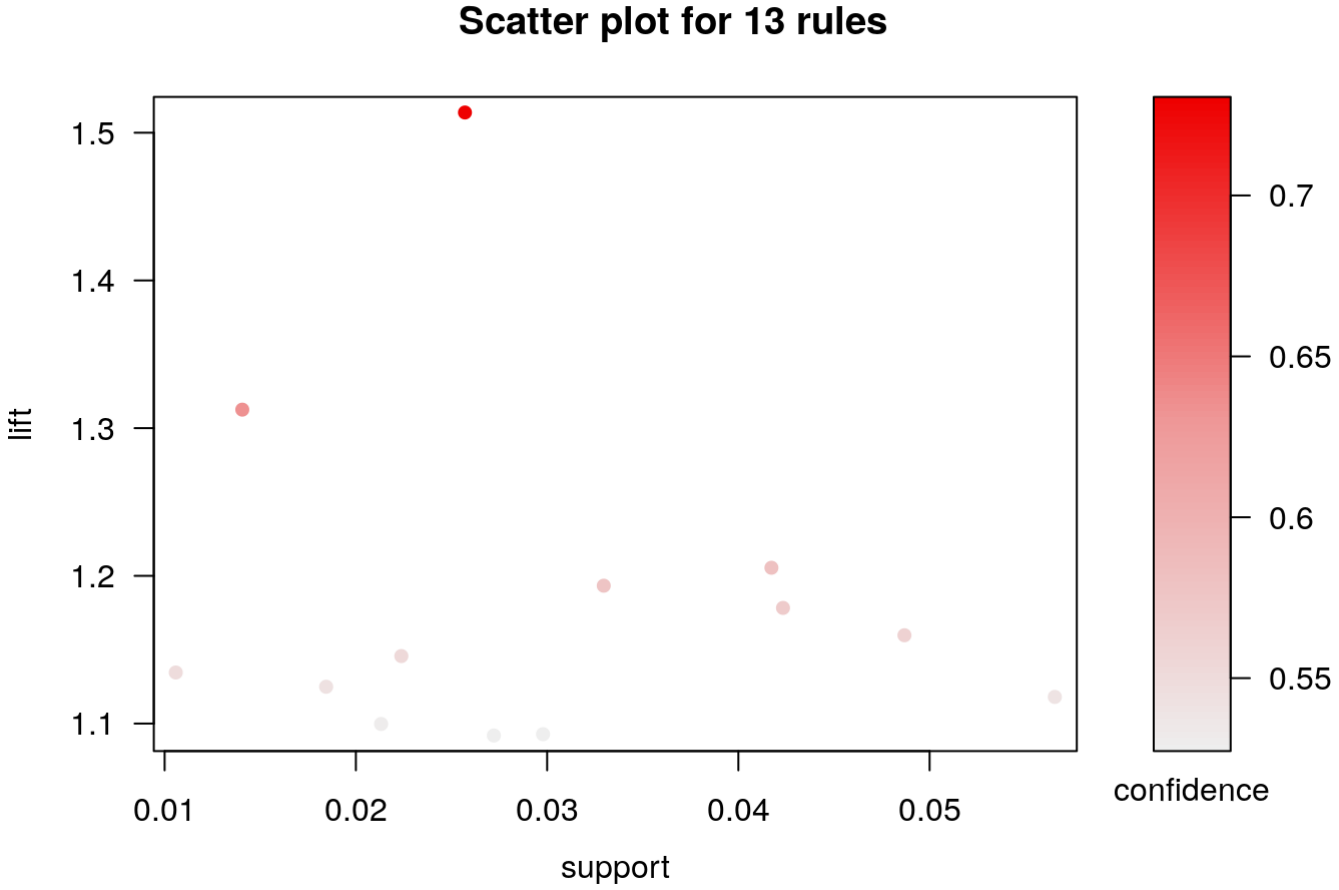
And so on. It seems that in this bakery there are many coffee lovers.

6.3 Visualize association rules

We are going to use the `arulesViz` package to create the visualizations. Let's begin with a simple scatter plot with different measures of interestingness on the axes (lift and support) and a third measure (confidence) represented by the color of the points.

Hide

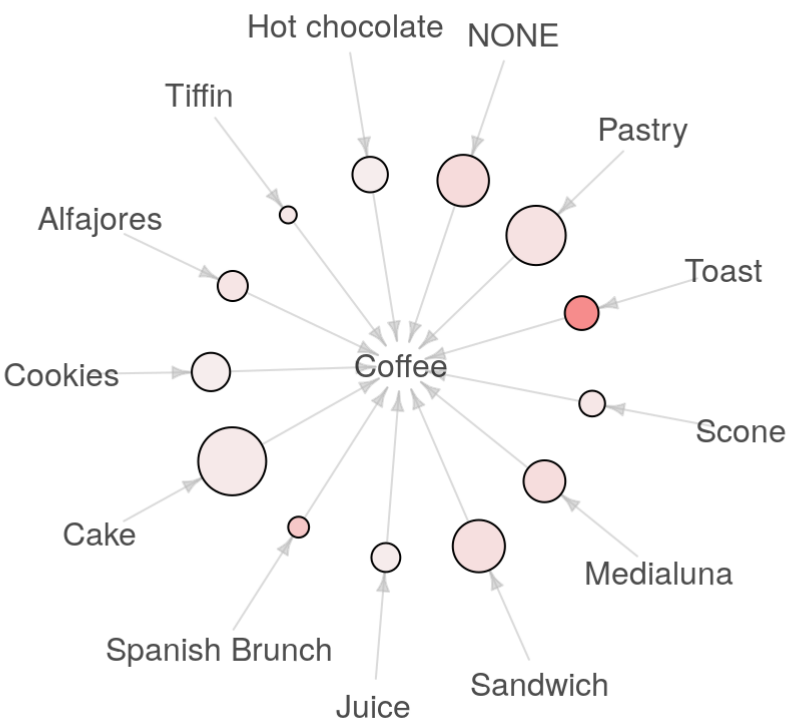
```
# Scatter plot
plot(rules_sup1_conf50, measure=c("support","lift"), shading="confidence")
```



The following visualization represents the rules as a graph with items as labeled vertices, and rules represented as vertices connected to items using arrows.

Hide

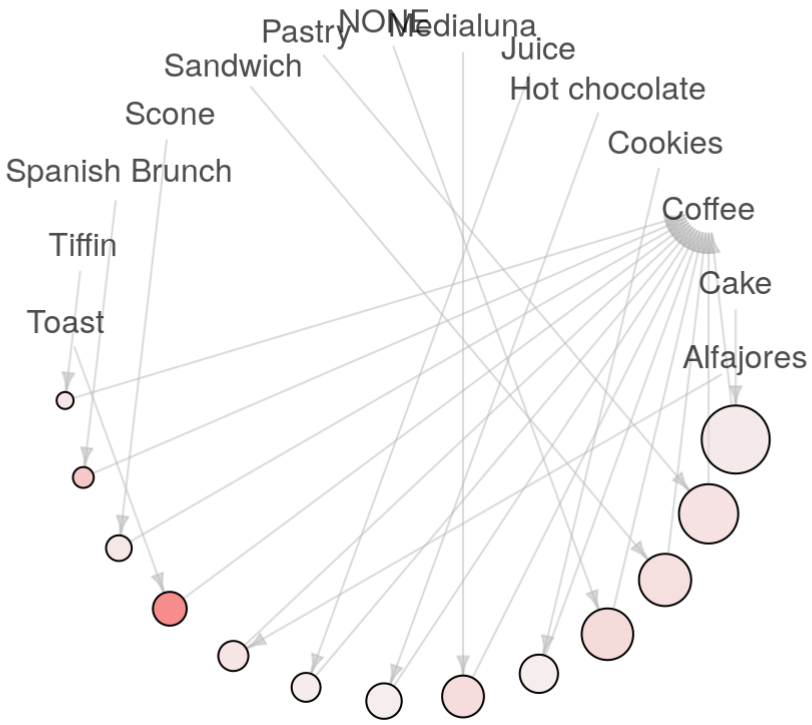
```
# Graph
plot(rules_sup1_conf50, method="graph")
```



We can also change the graph layout.

Hide

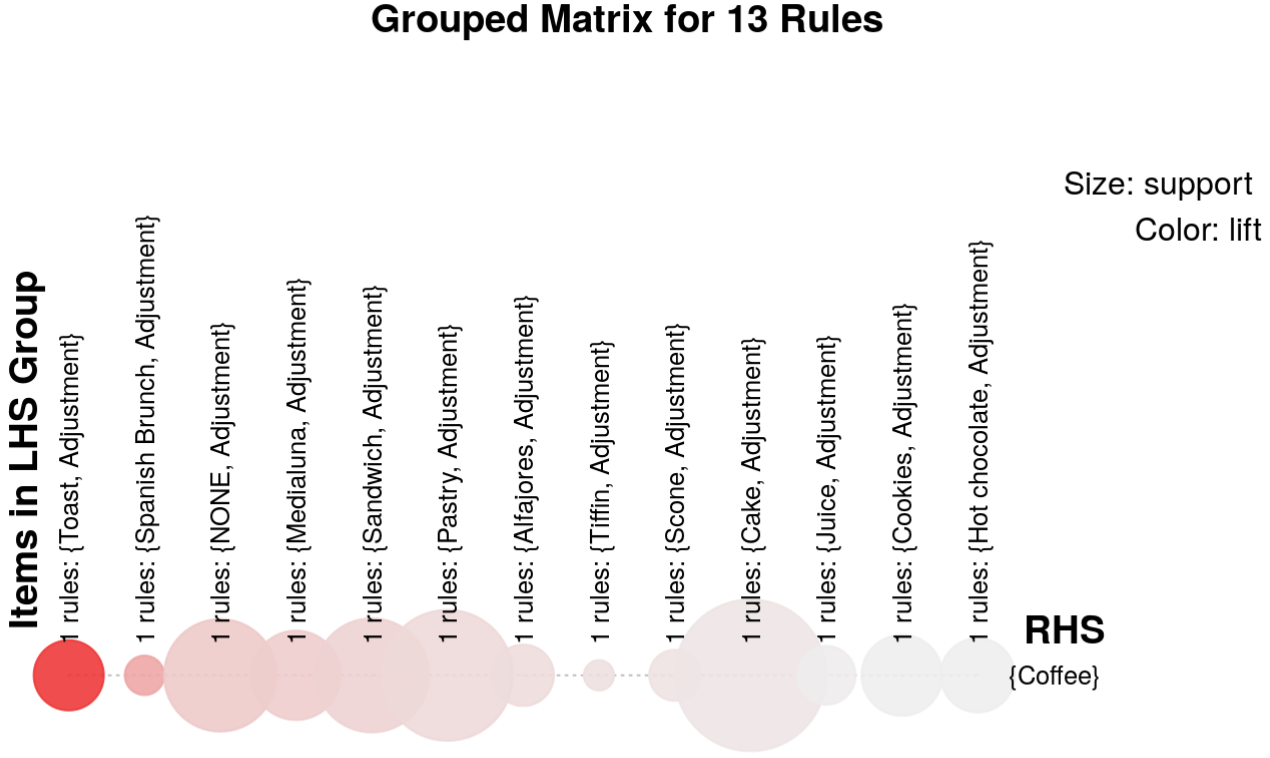
```
# Graph
plot(rules_sup1_conf50, method="graph", control=list(layout=igraph::in_circle
  ()))
```



What else can we do? We can represent the rules as a grouped matrix-based visualization. The support and lift measures are represented by the size and color of the ballons, respectively. In this case it's not a very useful visualization, since we only have coffe on the right-hand-side of the rules.

Hide

```
# Grouped matrix plot
plot(rules_sup1_conf50, method="grouped")
```



There's an awesome function called `ruleExplorer()` that explores association rules using interactive manipulations and visualization using shiny. Unfortunately, R Markdown still doesn't support shiny app objects.

6.4 Another execution

We have executed the Apriori algorithm with the appropriate support and confidence values. What happens if we execute it with low values? How do the visualizations change? Let's try with a support level of 0.5% and a confidence level of 10%.

Hide

```
# Apriori algorithm execution with a support level of 0.5% and a confidence lev
  el of 10%
rules_sup0.5_conf10 <- apriori(trans, parameter=list(sup=supportLevels[4], co
  nf=confidenceLevels[9], target="rules"))
```

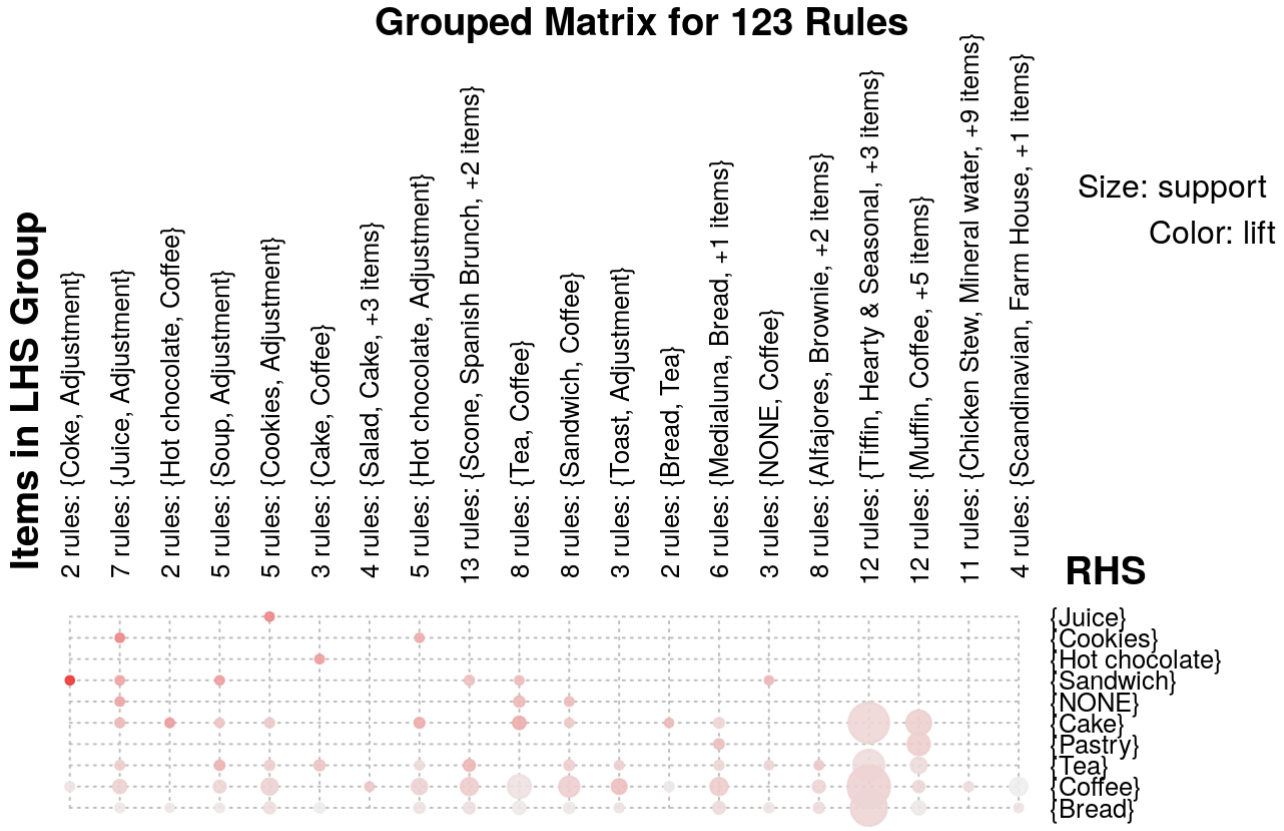
It's impossible to analyze these visualizations! For larger rule sets visual analysis becomes difficult. Furthermore, most of the rules are useless. That's why we have to carefully select the right values of support and confidence.

6.4.1 Graph 6.4.2 Parallel coordinates plot 6.4.3 Grouped matrix plot

6.4.4 Scatter plot

Hide

```
# Grouped matrix plot
plot(rules_sup0.5_conf10, method="grouped")
```



7 Exercises

In this section you can test the concepts learned during this kernel by answering the following questionnaire.

- Give an example where you can apply the Apriori algorithm.
- Calculate the support of the rule $Trousers \Rightarrow Jacket$ using the clothing store transactions. Interpret the result.
- Calculate the confidence of the rule $T\text{-}shirt \Rightarrow Jacket$ using the clothing store transactions. Interpret the result.
- Calculate the lift of the rule $Trousers \Rightarrow Sneakers$ using the clothing store transactions. Interpret the result.
- Calculate the conviction of the rule $T\text{-}shirt \Rightarrow Belt$ using the clothing store transactions. Interpret the result.
- Calculate the four metrics (support, confidence, lift and conviction) of the rule $\{T\text{-}shirt, Trousers\} \Rightarrow \{Jacket\}$ using the clothing store transactions. Interpret the results.

- What happens when we decrease the support level? Why?
- What happens when we increase the conficence level? Why?
- How many rules are generated with a support level of 0.5% and a confidence level of 20%? (you can use the previous visualizations)
- Using the previous data set, execute the Apriori algorithm with a support level of 5% and a confidence level of 10%. Are the rules interesting? Why?
- Prove the functions `ruleExplorer()` and `inspectDT()` from the package `arulesViz` on your RStudio environment.
- What recommendations would you give to the owner of the bakery?

8 **Summary**

In this kernel we have learned about the Apriori algorithm, one of the most frequently used algorithms in data mining. We have reviewed some statistical concepts (support, confidence, lift and conviction) to select interesting rules, we have chosen the appropriate values to execute the algorithm and finally we have visualized the resulting association rules.

And that's it! It has been a pleasure to make this kernel, I have learned a lot! Thank you for reading and if you like it, please upvote it.

9 **Citations for used packages**

Hadley Wickham (2017). tidyverse: Easily Install and Load the 'Tidyverse'. R package version 1.2.1. https://CRAN.R-project.org/package=tidyverse (https://CRAN.R-project.org/package=tidyverse)

Michael Hahsler, Christian Buchta, Bettina Gruen and Kurt Hornik (2018). arules: Mining Association Rules and Frequent Itemsets. R package version 1.6-1. https://CRAN.R-project.org/package=arules (https://CRAN.R-project.org/package=arules)

Michael Hahsler, Bettina Gruen and Kurt Hornik (2005), arules - A Computational Environment for Mining Association Rules and Frequent Item Sets. Journal of Statistical Software 14/15. URL: http://dx.doi.org/10.18637/jss.v014.i15 (http://dx.doi.org/10.18637/jss.v014.i15).

Michael Hahsler, Sudheer Chelluboina, Kurt Hornik, and Christian Buchta (2011), The arules R-package ecosystem: Analyzing interesting patterns from large transaction datasets. Journal of Machine Learning Research, 12:1977–1981. URL: http://jmlr.csail.mit.edu/papers/v12/hahsler11a.html (http://jmlr.csail.mit.edu/papers/v12/hahsler11a.html).

Michael Hahsler (2018). arulesViz: Visualizing Association Rules and Frequent Itemsets. R package version 1.3-1. https://CRAN.R-project.org/package=arulesViz (https://CRAN.R-project.org/package=arulesViz)

Yihui Xie (2018). knitr: A General-Purpose Package for Dynamic Report Generation in R. R package version 1.20.

Yihui Xie (2014) knitr: A Comprehensive Tool for Reproducible Research in R. In Victoria Stodden, Friedrich Leisch and Roger D. Peng, editors, Implementing Reproducible Computational Research. Chapman and Hall/CRC. ISBN 978-1466561595

Baptiste Auguie (2017). gridExtra: Miscellaneous Functions for “Grid” Graphics. R package version 2.3. https://CRAN.R-project.org/package=gridExtra (https://CRAN.R-project.org/package=gridExtra)