Market basket analysis

## Anuradha Sahithi Padavala

# Market Basket Analysis

Market basket analysis is a technique to discover co-occurrence relationships among activities performed by subjects. For example, shampoo and conditioner co-occur in shopping. Promoting conditioner could improve shampoo revenue.

# Market Basket Analysis - Metrics

There are three major metric for evaluating a market basket rule. Support - Support is an indication of how frequently the itemset appears in the dataset.

Support(X)=frequency(X)/N Support(A=>B)=frequency(A,B)/N

Confidence - Confidence is an indication of how often the rule has been found to be true.

Confidence(A=>B) = P(A∩B)/P(A) = frequency(A,B)/frequency(A)

Lift - Lift gives the correlation between A and B in the rule A=>B. Correlation shows how one item-set A effects the item-set B.

Lift(A=>B)=Support(A,B)/(Supp(A)Supp(B))

Density indicates how frequently items appear together in a basket. It measures how sparse the transaction-item matrix is If density is high, it is more common for more items to appear in a basket If density is low, few items are purchased together

In R, the most common package for market basket analysis is arules. arules::apriori() is the workhorse function of the package arules::apriori() arguments: data, transaction class object, or data.frame or binary matrix that can be coerced into a transaction object parameter = NULL appearance = NULL control = NULL

If we are only interested in rules, we can use the appearance parameter in arules::apriori() to specify if we want the item on the lhs, rhs or both.

options(digits = 1)  
library(arules)

## Loading required package: Matrix

##   
## Attaching package: 'arules'

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

data(Groceries,package = "arules")  
class(Groceries)

## [1] "transactions"  
## attr(,"package")  
## [1] "arules"

inspect(head(Groceries))

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

summary(Groceries)

## transactions as itemMatrix in sparse format with  
## 9835 rows (elements/itemsets/transactions) and  
## 169 columns (items) and a density of 0.03   
##   
## most frequent items:  
## whole milk other vegetables rolls/buns soda   
## 2513 1903 1809 1715   
## yogurt (Other)   
## 1372 34055   
##   
## element (itemset/transaction) length distribution:  
## sizes  
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16   
## 2159 1643 1299 1005 855 645 545 438 350 246 182 117 78 77 55 46   
## 17 18 19 20 21 22 23 24 26 27 28 29 32   
## 29 14 14 9 11 4 6 1 1 1 1 3 1   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1 2 3 4 6 32   
##   
## 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

itemFrequencyPlot(Groceries,topN = 10)

A graph of food items

Description automatically generated

apriori\_Groceries <- apriori(  
 data = Groceries,   
 parameter = list(  
 target = "rules",  
 maxlen = 2,  
 confidence = 0.1,  
 support = 0.05,  
 minlen = 2  
 ),   
 appearance = list(  
 both = c(  
 "whole milk",  
 "other vegetables",  
 "rolls/buns",  
 "soda",  
 "yogurt",  
 "bottled water",  
 "root vegetables",  
 "tropical fruit"  
 )  
 )  
  
)

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.1 0.1 1 none FALSE TRUE 5 0.05 2  
## maxlen target ext  
## 2 rules FALSE  
##   
## Algorithmic control:  
## filter tree heap memopt load sort verbose  
## 0.1 TRUE TRUE FALSE TRUE 2 TRUE  
##   
## Absolute minimum support count: 491   
##   
## set item appearances ...[8 item(s)] done [0.00s].  
## set transactions ...[8 item(s), 9835 transaction(s)] done [0.00s].  
## sorting and recoding items ... [8 item(s)] done [0.00s].  
## creating transaction tree ... done [0.00s].  
## checking subsets of size 1 2

## Warning in apriori(data = Groceries, parameter = list(target = "rules", : Mining  
## stopped (maxlen reached). Only patterns up to a length of 2 returned!

## done [0.00s].  
## writing ... [6 rule(s)] done [0.00s].  
## creating S4 object ... done [0.00s].

apriori\_Groceries

## set of 6 rules

summary(apriori\_Groceries)

## set of 6 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 2   
## 6   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2 2 2 2 2 2   
##   
## summary of quality measures:  
## support confidence lift count   
## Min. :0.06 Min. :0.2 Min. :1.2 Min. :551   
## 1st Qu.:0.06 1st Qu.:0.2 1st Qu.:1.3 1st Qu.:552   
## Median :0.06 Median :0.3 Median :1.5 Median :557   
## Mean :0.06 Mean :0.3 Mean :1.4 Mean :615   
## 3rd Qu.:0.07 3rd Qu.:0.4 3rd Qu.:1.6 3rd Qu.:691   
## Max. :0.07 Max. :0.4 Max. :1.6 Max. :736   
##   
## mining info:  
## data ntransactions support confidence  
## Groceries 9835 0.05 0.1

inspect(sort(apriori\_Groceries,by = "count"))

## lhs rhs support confidence lift count  
## [1] {other vegetables} => {whole milk} 0.07 0.4 2 736   
## [2] {whole milk} => {other vegetables} 0.07 0.3 2 736   
## [3] {rolls/buns} => {whole milk} 0.06 0.3 1 557   
## [4] {whole milk} => {rolls/buns} 0.06 0.2 1 557   
## [5] {yogurt} => {whole milk} 0.06 0.4 2 551   
## [6] {whole milk} => {yogurt} 0.06 0.2 2 551

apriori\_Groceries <- apriori(  
 data = Groceries,   
 parameter = list(  
 target = "rules",  
 maxlen = 3,  
 confidence = 0.1,  
 support = 0.02,  
 minlen = 2  
 )  
)

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.1 0.1 1 none FALSE TRUE 5 0.02 2  
## maxlen target ext  
## 3 rules FALSE  
##   
## Algorithmic control:  
## filter tree heap memopt load sort verbose  
## 0.1 TRUE TRUE FALSE TRUE 2 TRUE  
##   
## Absolute minimum support count: 196   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].  
## sorting and recoding items ... [59 item(s)] done [0.00s].  
## creating transaction tree ... done [0.00s].  
## checking subsets of size 1 2 3

## Warning in apriori(data = Groceries, parameter = list(target = "rules", : Mining  
## stopped (maxlen reached). Only patterns up to a length of 3 returned!

## done [0.00s].  
## writing ... [120 rule(s)] done [0.00s].  
## creating S4 object ... done [0.00s].

#apriori\_Groceries  
#summary(apriori\_Groceries)  
#inspect(sort(apriori\_Groceries,by = "count"))  
sum(is.redundant(apriori\_Groceries))

## [1] 0

inspect(apriori\_Groceries[is.redundant(apriori\_Groceries)])  
M <- DATAFRAME(apriori\_Groceries)  
M[M$RHS == "{whole milk}" & grepl(x = M$LHS,pattern = "\\{sausage\\}|\\{other vegetables\\}|\\{yogurt\\}|\\{soda\\}"),]

## LHS RHS support confidence lift count  
## 63 {sausage} {whole milk} 0.03 0.3 1.2 294  
## 101 {soda} {whole milk} 0.04 0.2 0.9 394  
## 107 {yogurt} {whole milk} 0.06 0.4 1.6 551  
## 113 {other vegetables} {whole milk} 0.07 0.4 1.5 736

apriori\_Groceries <- apriori\_Groceries[!is.redundant(apriori\_Groceries)]

require(arulesViz)

## Loading required package: arulesViz

## Loading required package: grid

## Registered S3 method overwritten by 'seriation':  
## method from   
## reorder.hclust gclus

plot(apriori\_Groceries)

## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.

A diagram of a scatter plot

Description automatically generated

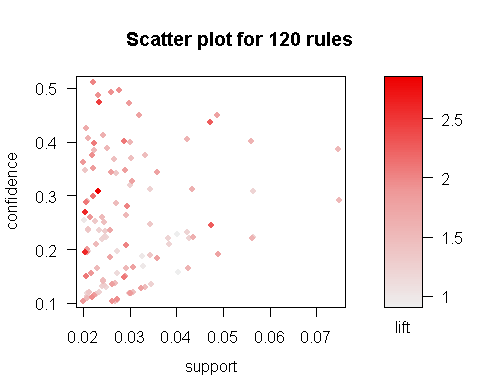
plot(  
 x = apriori\_Groceries,  
 measure = c("confidence", "lift"),  
 shading = "support"  
)

A diagram of a scatter plot

Description automatically generated

plot(  
 x = apriori\_Groceries,  
 method = "scatterplot"  
)

## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.



plot(  
 x = apriori\_Groceries,  
 method = "two-key plot"  
)

## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.

A diagram of a number of numbers

Description automatically generated with medium confidence

plot(  
 x = apriori\_Groceries,  
 method = "matrix"  
)

## Itemsets in Antecedent (LHS)  
## [1] "{other vegetables,whole milk}" "{root vegetables,whole milk}"   
## [3] "{whole milk,yogurt}" "{other vegetables,yogurt}"   
## [5] "{pip fruit}" "{whipped/sour cream}"   
## [7] "{curd}" "{root vegetables,other vegetables}"  
## [9] "{butter}" "{domestic eggs}"   
## [11] "{root vegetables}" "{tropical fruit}"   
## [13] "{pork}" "{citrus fruit}"   
## [15] "{frozen vegetables}" "{yogurt}"   
## [17] "{margarine}" "{other vegetables}"   
## [19] "{beef}" "{brown bread}"   
## [21] "{whole milk}" "{sausage}"   
## [23] "{fruit/vegetable juice}" "{frankfurter}"   
## [25] "{pastry}" "{newspapers}"   
## [27] "{bottled water}" "{rolls/buns}"   
## [29] "{soda}" "{shopping bags}"   
## [31] "{bottled beer}"   
## Itemsets in Consequent (RHS)  
## [1] "{soda}" "{rolls/buns}"   
## [3] "{bottled water}" "{shopping bags}"   
## [5] "{newspapers}" "{pastry}"   
## [7] "{fruit/vegetable juice}" "{sausage}"   
## [9] "{whole milk}" "{other vegetables}"   
## [11] "{yogurt}" "{citrus fruit}"   
## [13] "{tropical fruit}" "{domestic eggs}"   
## [15] "{butter}" "{curd}"   
## [17] "{root vegetables}" "{pork}"   
## [19] "{whipped/sour cream}" "{pip fruit}"

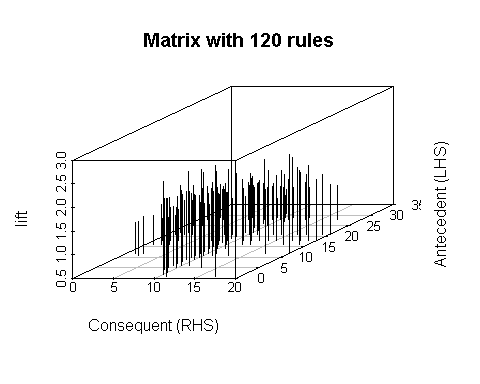
A graph with red lines

Description automatically generated

plot(  
 x = apriori\_Groceries,  
 method = "matrix3D"  
)

## Warning in plot.rules(x = apriori\_Groceries, method = "matrix3D"): method  
## 'matrix3D' is deprecated use method 'matrix' with engine '3d'

## Itemsets in Antecedent (LHS)  
## [1] "{other vegetables,whole milk}" "{root vegetables,whole milk}"   
## [3] "{whole milk,yogurt}" "{other vegetables,yogurt}"   
## [5] "{pip fruit}" "{whipped/sour cream}"   
## [7] "{curd}" "{root vegetables,other vegetables}"  
## [9] "{butter}" "{domestic eggs}"   
## [11] "{root vegetables}" "{tropical fruit}"   
## [13] "{pork}" "{citrus fruit}"   
## [15] "{frozen vegetables}" "{yogurt}"   
## [17] "{margarine}" "{other vegetables}"   
## [19] "{beef}" "{brown bread}"   
## [21] "{whole milk}" "{sausage}"   
## [23] "{fruit/vegetable juice}" "{frankfurter}"   
## [25] "{pastry}" "{newspapers}"   
## [27] "{bottled water}" "{rolls/buns}"   
## [29] "{soda}" "{shopping bags}"   
## [31] "{bottled beer}"   
## Itemsets in Consequent (RHS)  
## [1] "{soda}" "{rolls/buns}"   
## [3] "{bottled water}" "{shopping bags}"   
## [5] "{newspapers}" "{pastry}"   
## [7] "{fruit/vegetable juice}" "{sausage}"   
## [9] "{whole milk}" "{other vegetables}"   
## [11] "{yogurt}" "{citrus fruit}"   
## [13] "{tropical fruit}" "{domestic eggs}"   
## [15] "{butter}" "{curd}"   
## [17] "{root vegetables}" "{pork}"   
## [19] "{whipped/sour cream}" "{pip fruit}"



#plot(  
# x = apriori\_Groceries,  
# method = "graph",  
# engine = "htmlwidget"  
#)

plot(  
 x = apriori\_Groceries,  
 method = "paracoord"  
)

A graph of food and drinks

Description automatically generated with medium confidence

plot(  
 x = apriori\_Groceries,  
 method = "grouped"  
)

A close-up of a list of rules

Description automatically generated