COLUMBIA UNIVERSITY
School of Professional Studies

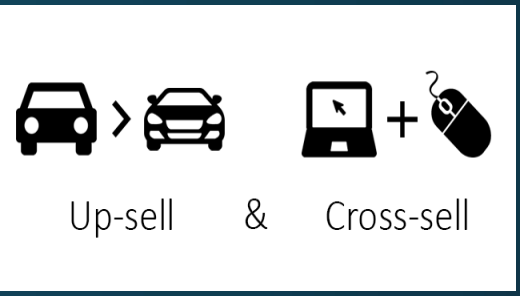
Association Rules: Market Basket Analysis
Applied Analytics: Frameworks and Methods 2

Outline

- Concept of market basket analysis
- Analytical technique: associate rule learning
  - Describe mathematical criteria for evaluating rules
- Explain the importance of domain expertise for interpreting association rules
- Illustrations
  - with a small-size data set
  - with a medium-size data set

Concept of Market Basket Analysis

Market Basket Analysis



- .... is a data mining technique that
  - help to discover which items go together
  - has the purpose of finding the optimal combination of products or services and allows marketers to exploit this knowledge to provide recommendations such as:
    - optimize product placement, or
    - develop marketing programs

  that **take advantage of cross-selling**.

- ... widely used by retailers to identify items that are purchased in the same shopping trip.

- ... not limited to retailers. May also be used by insurance companies, banks, etc.

Source: Figure from https://envato.com/blog/want-fries-art-cross-selling/

One of the most quoted illustrations of market basket analysis is a supermarket chain that found, "*male customers that bought diapers often bought beer as well*", so "*they put the diapers close to beer coolers, and their sales increased dramatically*" (Wikipedia).

Frequently bought together

Total price: $103.12

Add all three to Cart

Add all three to List

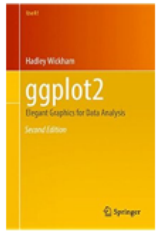
- **This item:** R for Data Science: Import, Tidy, Transform, Visualize, and Model Data by Hadley Wickham Paperback $34.01
- R Cookbook: Proven Recipes for Data Analysis, Statistics, and Graphics (O'Reilly Cookbooks) by Paul Teetor Paperback $34.73
- R Graphics Cookbook: Practical Recipes for Visualizing Data by Winston Chang Paperback $34.38

Customers who bought this item also bought

Page 1 of 13

R Cookbook: Proven Recipes for Data Analysis, Statistics, and Graphics…
› Paul Teetor
93
Paperback
$34.73 prime

R Graphics Cookbook: Practical Recipes for Visualizing Data
› Winston Chang
83
Paperback
$34.38 prime

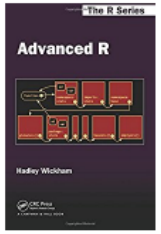


ggplot2: Elegant Graphics for Data Analysis (Use R!)
› Hadley Wickham
15
Paperback
$46.07 prime

Hands-On Programming with R: Write Your Own Functions and Simulations
› Garrett Grolemund
16
Paperback
$32.80 prime

Text Mining with R: A Tidy Approach
› Julia Silge
8
Paperback
$30.56 prime

Learning R: A Step-by-Step Function Guide to Data Analysis
Richard Cotton
16
Paperback
$25.35 prime



Advanced R (Chapman & Hall/CRC The R Series)
› Hadley Wickham
44
Paperback
$39.40

R in Action: Data Analysis and Graphics with R
› Robert Kabacoff
44
Paperback
$41.17 prime

Analytical Technique: Associate Rule Learning

Analytical Technique: Associate Rule Learning

- Unsupervised Learning
- Based on numeric thresholds, not statistical model
- Involves working with large sparse matrices

Measuring Affinity

$$Support = \frac{Number\ of\ transactions\ with\ both\ A\ and\ B}{Total\ number\ of\ transactions} = P(A \cap B)$$

$$Confidence = \frac{Number\ of\ transactions\ with\ both\ A\ and\ B}{Total\ number\ of\ transactions\ with\ A} = \frac{P(A \cap B)}{P(A)}$$

- Rule: A $\Rightarrow$ B
- Support: p(A&B)
  - Prevalence of an item set
    - Higher the support, the more popular is the set of products.
- Confidence: p(B|A)
  - Predictability of an association rule
  - It measures how often item B appears in transactions that contain item A.
    - Higher the confidence, the more likely item B appears in transactions that contain item A.

  p(B|A) = p(A&B) / p(A)
- Lift: p(B|A)/p(B)
  - Strength of association.
    - 1 indicates independence, i.e., no association.
    - > 1 indicates that the presence of A has increased the probability that product B will occur in this transaction.
    - < 1 indicates that the presence of A has decreased the probability that product B will occur in this transaction.

  p(B|A)/p(B) = p(A&B) / (p(A)*p(B))

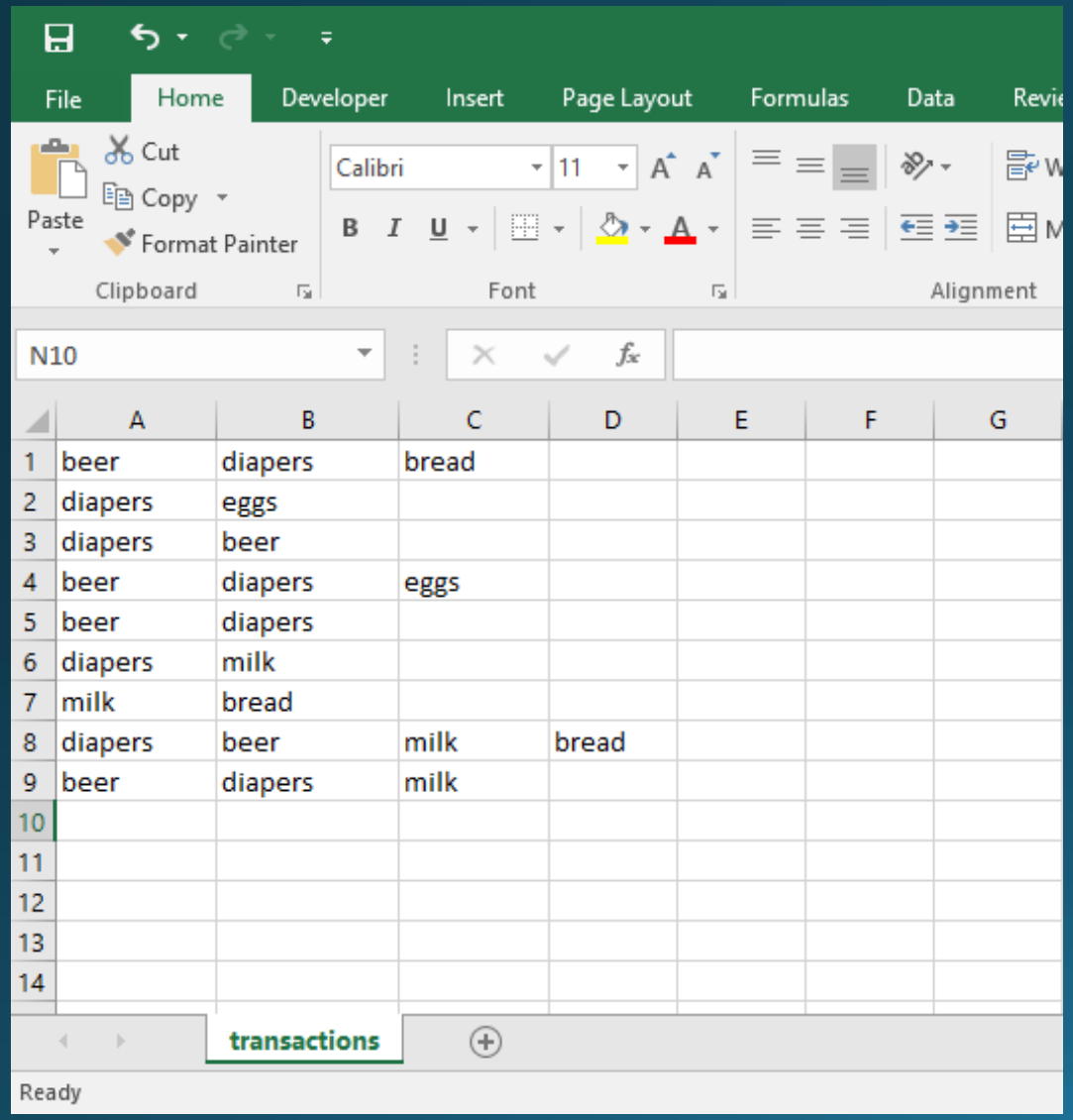
Measuring Affinity

Rule	Support	Confidence
A ⇒ D	2/5	2/3
C ⇒ A	2/5	2/4
A ⇒ C	2/5	2/3
B & C ⇒ D	1/5	1/3

Source: Found on https://themainstreamseer.blogspot.com/2012/09/numeric-measures-for-association-rules.html on Sept 2012

Illustration: Small-Size Data Set

Raw Data



	A	B	C	D	E	F	G
1	beer	diapers	bread				
2	diapers	eggs					
3	diapers	beer					
4	beer	diapers	eggs				
5	beer	diapers					
6	diapers	milk					
7	milk	bread					
8	diapers	beer	milk	bread			
9	beer	diapers	milk				
10							
11							
12							
13							
14							

- 9 transactions and 5 items
- Most frequent items:

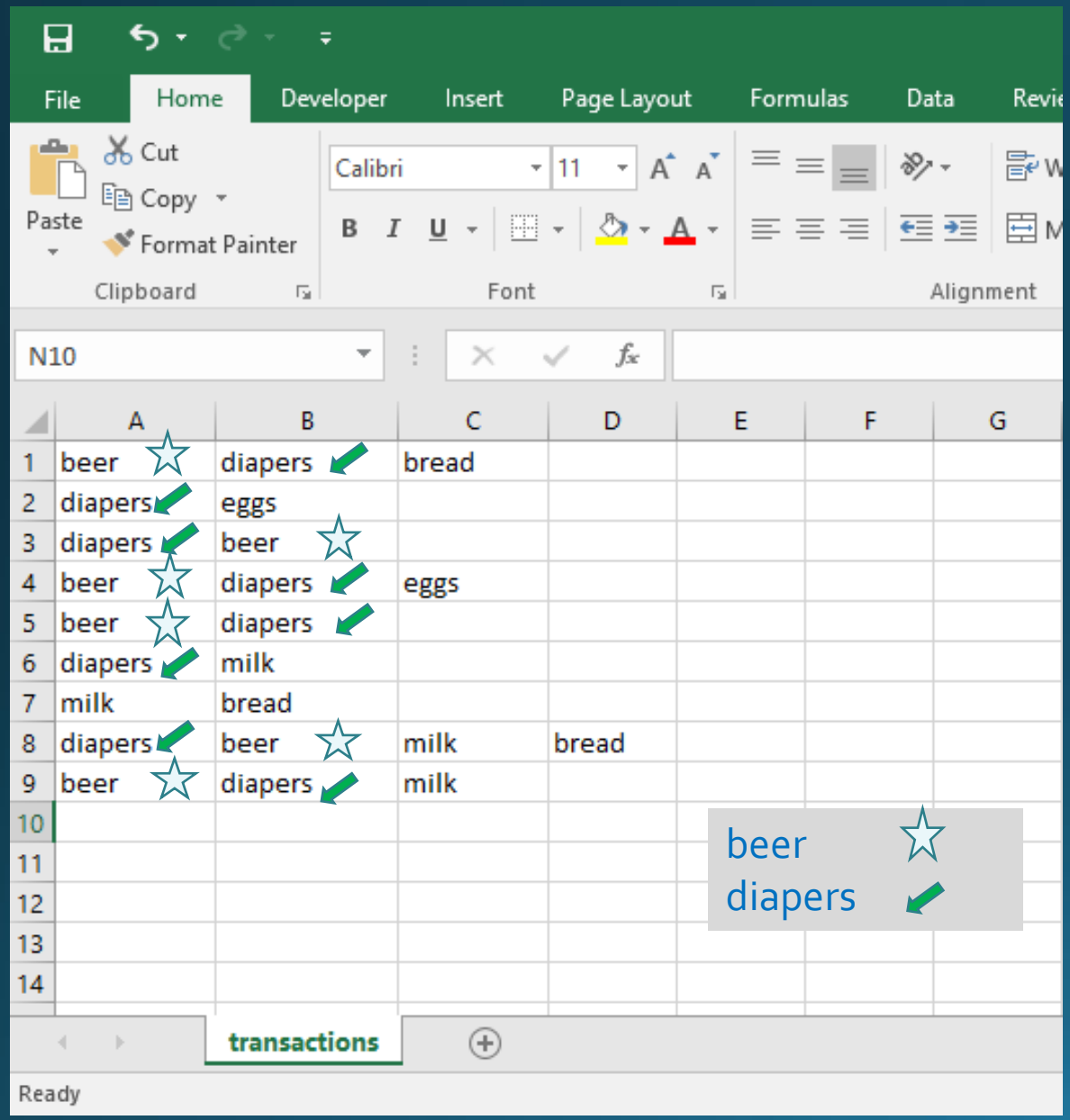
diapers beer milk bread eggs

8 6 4 3 2

Calculate Support, Confidence and Lift

Consider the Rule: Diapers ⇒ Beer

Columbia University
School of Professional Studies



	A	B	C	D
1	beer	diapers	bread	
2	diapers	eggs		
3	diapers	beer		
4	beer	diapers	eggs	
5	beer	diapers		
6	diapers	milk		
7	milk	bread		
8	diapers	beer	milk	bread
9	beer	diapers	milk	

beer ☆
diapers ✓

- Rule Frequency:
  How often do diapers and beer co-occur? 6

- Support, p(Beer & Diapers):
  What proportion of transactions contain both diapers and beer? 6/9 = 0.67

- Confidence, p(Beer|Diapers):
  Given that diapers occur in a transaction, what is the chance the transaction contains beer?
  Confidence = p(Beer & Diapers)/p(Diapers)
  = (6/9) / (8/9)
  = 6/8 = 0.75

- Lift, p(Beer|Diapers)/p(Beer)
  = p(Beer & Diapers)/p(Beer)p(Diapers)
  = (6/9) / [(6/9)*(8/9)] =1.125

Data

beer	diapers	bread	
diapers	eggs		
diapers	beer		
beer	diapers	eggs	
beer	diapers		
diapers	milk		
milk	bread		
diapers	beer	milk	bread
beer	diapers	milk	

Support and Lift matrices from data

```
> crossTable(items,measure='support',sort=T)
          diapers      beer      milk     bread      eggs
diapers 0.8888889 0.6666667 0.3333333 0.2222222 0.2222222
beer    0.6666667 0.6666667 0.2222222 0.2222222 0.1111111
milk    0.3333333 0.2222222 0.4444444 0.2222222 0.0000000
bread   0.2222222 0.2222222 0.2222222 0.3333333 0.0000000
eggs    0.2222222 0.1111111 0.0000000 0.0000000 0.2222222
```

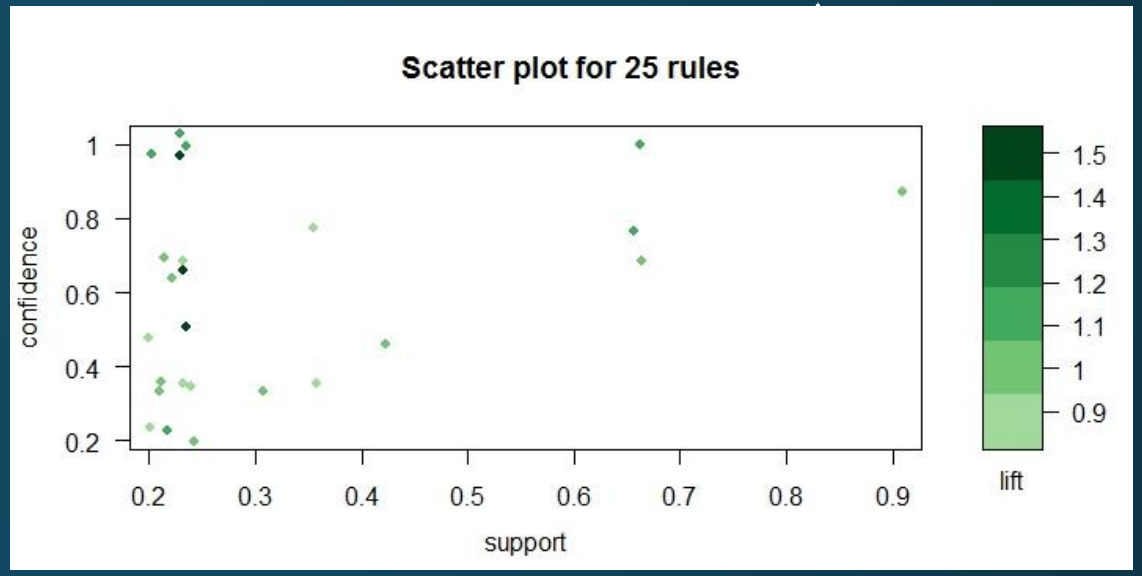
```
> crossTable(items,measure='lift',sort=T)
        diapers  beer    milk bread  eggs
diapers      NA 1.125 0.84375  0.75 1.125
beer    1.12500    NA 0.75000  1.00 0.750
milk    0.84375 0.750      NA  1.50 0.000
bread   0.75000 1.000 1.50000    NA 0.000
eggs    1.12500 0.750 0.00000  0.00    NA
```

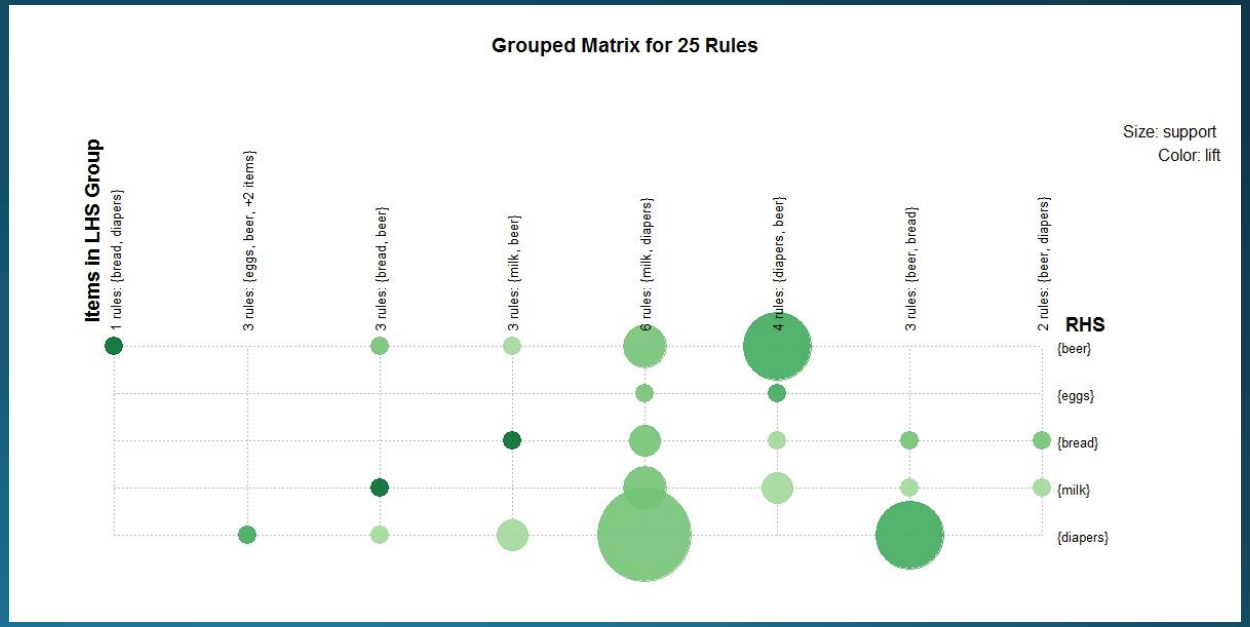
Sifting through Rules

- Set up rule filtering criteria.

  E.g. support>0.2, confidence>0.2.

	lhs		rhs	support	confidence	lift	count
[8]	{bread}	=>	{milk}	0.2222222	0.6666667	1.50000	2
[9]	{milk}	=>	{bread}	0.2222222	0.5000000	1.50000	2
[21]	{bread,diapers}	=>	{beer}	0.2222222	1.0000000	1.50000	2
[18]	{beer}	=>	{diapers}	0.6666667	1.0000000	1.12500	6
[19]	{diapers}	=>	{beer}	0.6666667	0.7500000	1.12500	6
[6]	{eggs}	=>	{diapers}	0.2222222	1.0000000	1.12500	2
[7]	{diapers}	=>	{eggs}	0.2222222	0.2500000	1.12500	2
[20]	{beer,bread}	=>	{diapers}	0.2222222	1.0000000	1.12500	2
[23]	{beer,milk}	=>	{diapers}	0.2222222	1.0000000	1.12500	2
[5]	{}	=>	{diapers}	0.8888889	0.8888889	1.00000	8
[4]	{}	=>	{beer}	0.6666667	0.6666667	1.00000	6
[3]	{}	=>	{milk}	0.4444444	0.4444444	1.00000	4
[2]	{}	=>	{bread}	0.3333333	0.3333333	1.00000	3
[1]	{}	=>	{eggs}	0.2222222	0.2222222	1.00000	2
[10]	{bread}	=>	{beer}	0.2222222	0.6666667	1.00000	2
[11]	{beer}	=>	{bread}	0.2222222	0.3333333	1.00000	2
[22]	{beer,diapers}	=>	{bread}	0.2222222	0.3333333	1.00000	2
[24]	{diapers,milk}	=>	{beer}	0.2222222	0.6666667	1.00000	2
[16]	{milk}	=>	{diapers}	0.3333333	0.7500000	0.84375	3
[17]	{diapers}	=>	{milk}	0.3333333	0.3750000	0.84375	3
[12]	{bread}	=>	{diapers}	0.2222222	0.6666667	0.75000	2
[13]	{diapers}	=>	{bread}	0.2222222	0.2500000	0.75000	2
[14]	{milk}	=>	{beer}	0.2222222	0.5000000	0.75000	2
[15]	{beer}	=>	{milk}	0.2222222	0.3333333	0.75000	2
[25]	{beer,diapers}	=>	{milk}	0.2222222	0.3333333	0.75000	2





Sifting Through Rules

- In some cases, interest may be in a specific association. E.g., consider all transactions with diapers in LHS

```
    lhs           rhs    support   confidence lift  count
[1] {diapers} => {eggs} 0.2222222 0.25       1.125 2
[2] {diapers} => {beer} 0.6666667 0.75       1.125 6
```

Visualizing Rules


```
library(arulesViz)

plot(rules, method = 'ENTER METHOD', measure = c('support','confidence'),
                                                shading = 'lift', interactive = FALSE, control = list(reorder=T), ...)
```

Technique	Method	Rule set	Measures	Interactive	Reordering	Ease of use
Scatterplot	`"scatterplot"`	large	3	✓		++
Two-Key plot	`"scatterplot"`	large	2 + order	✓		++
Matrix-based	`"matrix"`	medium	1		✓	0
Matrix-b. (2 measures)	`"matrix"`	medium	2		✓	− −
Matrix-b. (3D bar)	`"matrix3D"`	small	1		✓	+
Grouped matrix	`"grouped"`	large	2	✓	✓	0
Graph-based	`"graph"`	small	2			++
Graph-b. (external)	`"graph"`	large	2	✓	✓	+
Parallel coordinates	`"paracoord"`	small	1		✓	−
Double decker	`"doubledecker"`	single rule	(2)			−

Source: arulesViz Vignette

Association Rules (Market Basket Analysis)

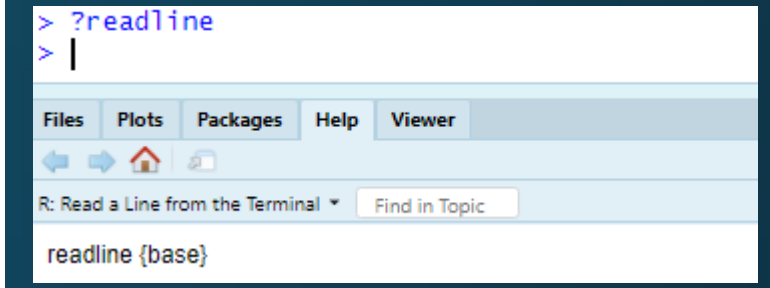
Examine Data

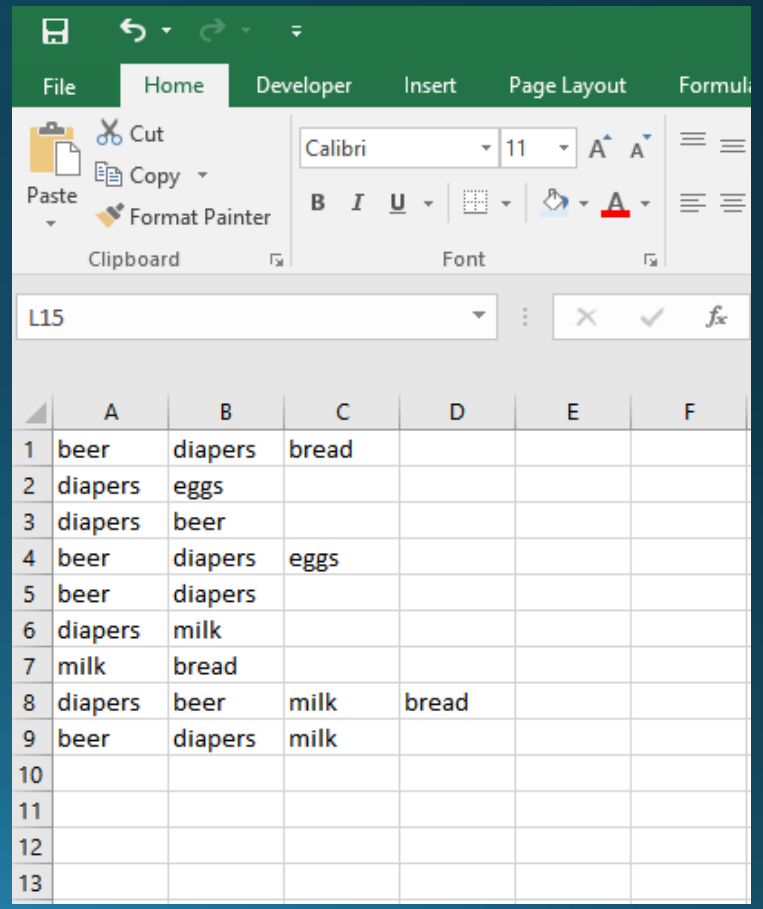
To illustrate the process of market basket analysis, let us begin with a simple dataset consisting of a set of nine transactions. Each row represents items purchased on a single visit to a supermarket. Note that specific items as well as number of items vary across visits.

```
cat(paste(readLines('transactions.csv'),'\n',collapse=''))
```

```
## beer,diapers,bread,
## diapers,eggs,,
## diapers,beer,,
## beer,diapers,eggs,
## beer,diapers,,
## diapers,milk,,
## milk,bread,,
## diapers,beer,milk,bread
## beer,diapers,milk,
```

COLUMBIA UNIVERSITY
School of Professional Studies





	A	B	C	D	E	F
1	beer	diapers	bread			
2	diapers	eggs				
3	diapers	beer				
4	beer	diapers	eggs			
5	beer	diapers				
6	diapers	milk				
7	milk	bread				
8	diapers	beer	milk	bread		
9	beer	diapers	milk			
10						
11						
12						
13						

Read Data

Load libraries

```
library(arules); library(arulesViz)
```

Traditional ways of importing data will result in an incomplete matrix. The read.transactions() function is specifically designed to yield an analysis usable complete matrix.

```
items = read.transactions('transactions.csv',format='basket',sep=',')
as(items,'matrix')
```

```
##        beer bread diapers  eggs  milk
##  [1,]  TRUE  TRUE    TRUE FALSE FALSE
##  [2,] FALSE FALSE    TRUE  TRUE FALSE
##  [3,]  TRUE FALSE    TRUE FALSE FALSE
##  [4,]  TRUE FALSE    TRUE  TRUE FALSE
##  [5,]  TRUE FALSE    TRUE FALSE FALSE
##  [6,] FALSE FALSE    TRUE FALSE  TRUE
##  [7,] FALSE  TRUE   FALSE FALSE  TRUE
##  [8,]  TRUE  TRUE    TRUE FALSE  TRUE
##  [9,]  TRUE FALSE    TRUE FALSE  TRUE
```

- 9 transactions (rows)
- 5 items (columns)

read.transactions() yields an arules object where transactions are organized along rows and items along columns. Here are the transactions and items

```
items
```

```
## transactions in sparse format with
##  9 transactions (rows) and
##  5 items (columns)
```

Number of transactions or market baskets

```
dim(items)[1]
```

```
## [1] 9
```

```
> dim(items)
[1] 9 5
```

Total number of items

```
dim(items)[2]
```

```
## [1] 5
```

A summary of all transactions

```
summary(items)
```

```
## transactions as itemMatrix in sparse format with
##  9 rows (elements/itemsets/transactions) and
##  5 columns (items) and a density of 0.5111111
##
## most frequent items:
## diapers    beer    milk   bread    eggs (Other)
##       8       6       4       3       2       0
##
## element (itemset/transaction) length distribution:
## sizes
## 2 3 4
## 5 3 1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   2.000   2.000   2.556   3.000   4.000
##
## includes extended item information - examples:
##    labels
## 1    beer
## 2   bread
## 3 diapers
```

- Top three frequently occurring items

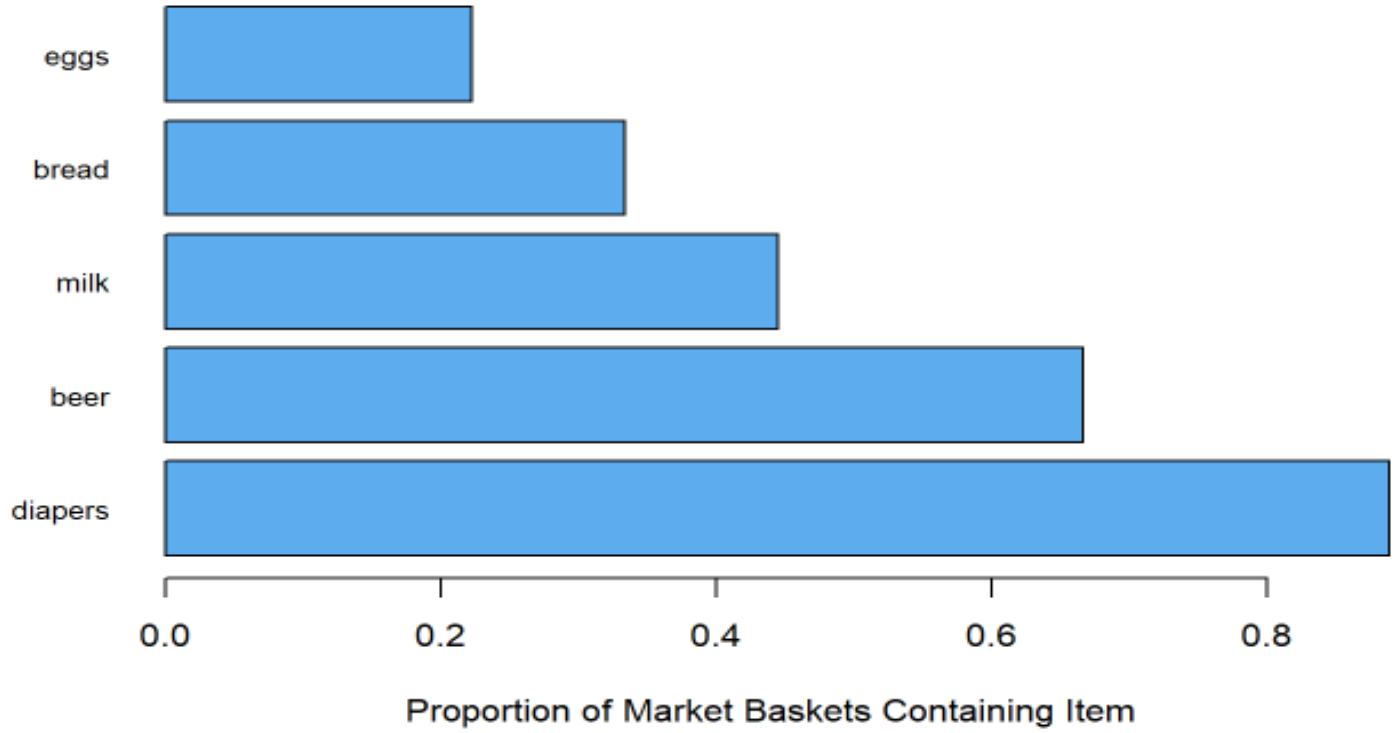
- 5 transactions has 2 items
- 3 transactions has 3 items
- 1 transaction has 4 items

- Minimum number of items in a transaction is 2
- Maximum number of items item in a transaction is 4

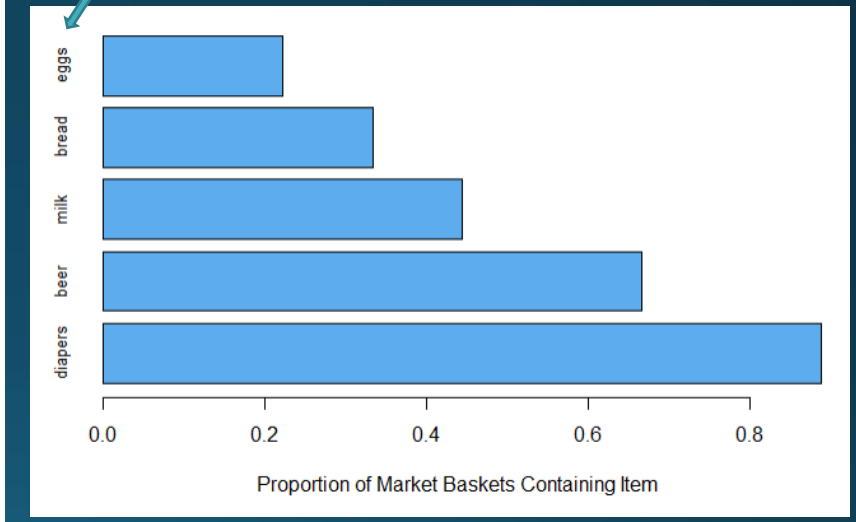
Explore Data

Examine individual items. Support is set to 0 so we can see all items.

```
itemFrequencyPlot(items, support = 0.0, cex.names=0.8,
                  type = "relative", horiz = TRUE, col = "steelblue2", las = 1, topN=5,
                  xlab = paste("Proportion of Market Baskets Containing Item"))
```




```
itemFrequencyPlot(items, support = 0.0, cex.names=0.8,
                  type = "relative", horiz = TRUE, col = "steelblue2", las = 0, topN=5,
                  xlab = paste("Proportion of Market Baskets Containing Item"))
```



Frequency or Counts

Compute the frequency of occurence of various pairs of items. Note measure='count' is the default, code below will generate the same result if measure is not specified

```
crossTable(items,measure='count',sort=T)
```

```
##         diapers beer milk bread eggs
## diapers       8    6    3     2    2
## beer          6    6    2     2    1
## milk          3    2    4     2    0
## bread         2    2    2     3    0
## eggs          2    1    0     0    2
```

```
> ?crossTable
>
```

Files | Plots | Packages | Help | Viewer

R: Cross-tabulate joint occurrences across pairs of items — Find in Topic

Arguments

x	object to be cross-tabulated (transactions or itemMatrix).
measure	measure to return. Default is co-occurence counts.
sort	sort the items by support.
...	aditional arguments.

Support

Prevalence of an item set

```
crossTable(items,measure='support',sort=T)
```

```
##           diapers      beer      milk     bread      eggs
## diapers 0.8888889 0.6666667 0.3333333 0.2222222 0.2222222
## beer    0.6666667 0.6666667 0.2222222 0.2222222 0.1111111
## milk    0.3333333 0.2222222 0.4444444 0.2222222 0.0000000
## bread   0.2222222 0.2222222 0.2222222 0.3333333 0.0000000
## eggs    0.2222222 0.1111111 0.0000000 0.0000000 0.2222222
```

Lift

Strength of association. Lift of 1 indicates independence, i.e., no association

```
crossTable(items,measure='lift',sort=T)
```

```
##         diapers  beer    milk bread  eggs
## diapers      NA 1.125 0.84375  0.75 1.125
## beer    1.12500    NA 0.75000  1.00 0.750
## milk    0.84375 0.750      NA  1.50 0.000
## bread   0.75000 1.000 1.50000    NA 0.000
## eggs    1.12500 0.750 0.00000  0.00    NA
```

Analysis

Association Rules

Even a small number of products can generate a pretty large number of rules

```
rules_all = apriori(items,parameter=list(support=0,confidence=0))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##           0    0.1    1 none FALSE            TRUE       5       0      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 0
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[5 item(s), 9 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [75 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(rules_all)
```

```
## set of 75 rules
##
## rule length distribution (lhs + rhs):sizes
##  1  2  3  4
##  5 20 30 20
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   2.867   4.000   4.000
##
## summary of quality measures:
##     support         confidence        coverage            lift
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.08333   1st Qu.:0.0000   1st Qu.:0.2812
##  Median :0.1111   Median :0.50000   Median :0.2222   Median :1.0000
##  Mean   :0.1304   Mean   :0.52130   Mean   :0.3472   Mean   :0.9333
##  3rd Qu.:0.2222   3rd Qu.:1.00000   3rd Qu.:0.6667   3rd Qu.:1.1250
##  Max.   :0.8889   Max.   :1.00000   Max.   :1.0000   Max.   :3.0000
##                                     NA's   :19
##      count
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :1.000
##  Mean   :1.173
##  3rd Qu.:2.000
##  Max.   :8.000
##
##
## mining info:
##   data ntransactions support confidence
##  items             9       0          0
```

- The number of rules generated: 75
- The distribution of rules by length: most rules are 3 items long
- The summary of quality/affinity measures: interesting to see ranges of support, lift, and confidence.
- The information on the data mined: total data mined, and minimum parameters.

```
x = inspect(rules_all)
```

```
##      lhs                rhs          support   confidence coverage  lift    
## [1]  {}              => {eggs}    0.2222222 0.2222222  1.0000000 1.00000
## [2]  {}              => {bread}   0.3333333 0.3333333  1.0000000 1.00000
## [3]  {}              => {milk}    0.4444444 0.4444444  1.0000000 1.00000
## [4]  {}              => {beer}    0.6666667 0.6666667  1.0000000 1.00000
## [5]  {}              => {diapers} 0.8888889 0.8888889  1.0000000 1.00000
## [6]  {eggs}          => {bread}   0.0000000 0.0000000        NaN 0.00000
## [7]  {bread}         => {eggs}    0.0000000 0.0000000        NaN 0.00000
## [8]  {eggs}          => {milk}    0.0000000 0.0000000        NaN 0.00000
## [9]  {milk}          => {eggs}    0.0000000 0.0000000        NaN 0.00000
## [10] {eggs}          => {beer}    0.1111111 0.5000000  0.2222222 0.75000
## [11] {beer}          => {eggs}    0.1111111 0.1666667  0.6666667 0.75000
## [12] {eggs}          => {diapers} 0.2222222 1.0000000  0.2222222 1.12500
## [13] {diapers}       => {eggs}    0.2222222 0.2500000  0.8888889 1.12500
## [14] {bread}         => {milk}    0.2222222 0.6666667  0.3333333 1.50000
## [15] {milk}          => {bread}   0.2222222 0.5000000  0.4444444 1.50000
## [16] {bread}         => {beer}    0.2222222 0.6666667  0.3333333 1.00000
## [17] {beer}          => {bread}   0.2222222 0.3333333  0.6666667 1.00000
## [18] {bread}         => {diapers} 0.2222222 0.6666667  0.3333333 0.75000
## [19] {diapers}       => {bread}   0.2222222 0.2500000  0.8888889 0.75000
## [20] {milk}          => {beer}    0.2222222 0.5000000  0.4444444 0.75000
## [21] {beer}          => {milk}    0.2222222 0.3333333  0.6666667 0.75000
## [22] {milk}          => {diapers} 0.3333333 0.7500000  0.4444444 0.84375
## [23] {diapers}       => {milk}    0.3333333 0.3750000  0.8888889 0.84375
## [24] {beer}          => {diapers} 0.6666667 1.0000000  0.6666667 1.12500
## [25] {diapers}       => {beer}    0.6666667 0.7500000  0.8888889 1.12500
## [26] {bread,eggs}    => {milk}    0.0000000 1.0000000  0.0000000 2.25000
## [27] {eggs,milk}     => {bread}   0.0000000 1.0000000  0.0000000 3.00000
## [28] {bread,milk}    => {eggs}    0.0000000 0.0000000        NaN 0.00000
```

```
##      count
## [1]  2
## [2]  3
## [3]  4
## [4]  6
## [5]  8
## [6]  0
## [7]  0
## [8]  0
## [9]  0
## [10] 1
## [11] 1
## [12] 2
## [13] 2
## [14] 2
## [15] 2
## [16] 2
## [17] 2
## [18] 2
## [19] 2
## [20] 2
## [21] 2
## [22] 3
## [23] 3
## [24] 6
## [25] 6
## [26] 0
## [27] 0
## [28] 0
```

```
> ?apriori
>
```

Files | Plots | Packages | Help | Viewer

R: Mining Associations with Apriori — Find in Topic

Details

Calls the C implementation of the Apriori algorithm by Christian Borgelt for mining frequent itemsets, rules or hyperedges.

Note: Apriori only creates rules with one item in the RHS (Consequent)! The default value in `APparameter` for `minlen` is 1. This means that rules with only one item (i.e., an empty antecedent/LHS) like

{} => {beer}

will be created. These rules mean that no matter what other items are involved, the item in the RHS will appear with the probability given by the rule's confidence (which equals the support). If you want to avoid these rules then use the argument `parameter=list(minlen=2)`.

Notes on run time and memory usage: If the minimum `support` is chosen too low for the dataset, then the algorithm will try to create an extremely large set of itemsets/rules. This will result in very long run time and eventually the process will run out of memory. To prevent this, the default maximal length of itemsets/rules is restricted to 10 items (via the parameter element `maxlen=10`) and the time for checking subsets is limited to 5 seconds (via `maxtime=5`). The output will show if you hit these limits in the "checking subsets" line of the output. The time limit is only checked when the subset size increases, so it may run significantly longer than what you specify in maxtime. Setting `maxtime=0` disables the time limit.

`parameter`	object of class `APparameter` or named list. The default behavior is to mine rules with minimum support of 0.1, minimum confidence of 0.8, maximum of 10 items (maxlen), and a maximal time for subset checking of 5 seconds (maxtime).

The object resulting from inspect() is a dataframe, so it can be subset and sorted using commonly used approaches.

```
x = x[x$count!=0,]
x[order(x$lift,x$support, decreasing = T),]
```

	lhs <fctr>		rhs <fctr>	support <dbl>	confidence <dbl>	lift <dbl>	count <int>
[14]	{bread}	=>	{milk}	0.2222222	0.6666667	1.50000	2
[15]	{milk}	=>	{bread}	0.2222222	0.5000000	1.50000	2
[51]	{bread,diapers}	=>	{beer}	0.2222222	1.0000000	1.50000	2
[46]	{beer,milk}	=>	{bread}	0.1111111	0.5000000	1.50000	1
[73]	{bread,diapers,milk}	=>	{beer}	0.1111111	1.0000000	1.50000	1
[75]	{beer,diapers,milk}	=>	{bread}	0.1111111	0.5000000	1.50000	1
[24]	{beer}	=>	{diapers}	0.6666667	1.0000000	1.12500	6
[25]	{diapers}	=>	{beer}	0.6666667	0.7500000	1.12500	6
[12]	{eggs}	=>	{diapers}	0.2222222	1.0000000	1.12500	2
[13]	{diapers}	=>	{eggs}	0.2222222	0.2500000	1.12500	2

1-10 of 40 rows Previous 1 2 3 4 Next

[50]	{beer,bread}	=>	{diapers}	0.2222222	1.0000000	1.12500	2
[53]	{beer,milk}	=>	{diapers}	0.2222222	1.0000000	1.12500	2
[41]	{beer,eggs}	=>	{diapers}	0.1111111	1.0000000	1.12500	1
[45]	{beer,bread}	=>	{milk}	0.1111111	0.5000000	1.12500	1
[48]	{bread,diapers}	=>	{milk}	0.1111111	0.5000000	1.12500	1
[72]	{beer,bread,milk}	=>	{diapers}	0.1111111	1.0000000	1.12500	1
[74]	{beer,bread,diapers}	=>	{milk}	0.1111111	0.5000000	1.12500	1
[5]	{}	=>	{diapers}	0.8888889	0.8888889	1.00000	8
[4]	{}	=>	{beer}	0.6666667	0.6666667	1.00000	6
[3]	{}	=>	{milk}	0.4444444	0.4444444	1.00000	4

11-20 of 40 rows Previous 1 **2** 3 4 Next

[2]	{}	=>	{bread}	0.3333333	0.3333333	1.00000	3
[1]	{}	=>	{eggs}	0.2222222	0.2222222	1.00000	2
[16]	{bread}	=>	{beer}	0.2222222	0.6666667	1.00000	2
[17]	{beer}	=>	{bread}	0.2222222	0.3333333	1.00000	2
[52]	{beer,diapers}	=>	{bread}	0.2222222	0.3333333	1.00000	2
[54]	{diapers,milk}	=>	{beer}	0.2222222	0.6666667	1.00000	2
[49]	{diapers,milk}	=>	{bread}	0.1111111	0.3333333	1.00000	1
[22]	{milk}	=>	{diapers}	0.3333333	0.7500000	0.84375	3
[23]	{diapers}	=>	{milk}	0.3333333	0.3750000	0.84375	3
[18]	{bread}	=>	{diapers}	0.2222222	0.6666667	0.75000	2

21-30 of 40 rows Previous 1 2 **3** 4 Next

[19]	{diapers}	=>	{bread}	0.2222222	0.2500000	0.75000	2
[20]	{milk}	=>	{beer}	0.2222222	0.5000000	0.75000	2
[21]	{beer}	=>	{milk}	0.2222222	0.3333333	0.75000	2
[55]	{beer,diapers}	=>	{milk}	0.2222222	0.3333333	0.75000	2
[10]	{eggs}	=>	{beer}	0.1111111	0.5000000	0.75000	1
[11]	{beer}	=>	{eggs}	0.1111111	0.1666667	0.75000	1
[42]	{diapers,eggs}	=>	{beer}	0.1111111	0.5000000	0.75000	1
[43]	{beer,diapers}	=>	{eggs}	0.1111111	0.1666667	0.75000	1
[44]	{bread,milk}	=>	{beer}	0.1111111	0.5000000	0.75000	1
[47]	{bread,milk}	=>	{diapers}	0.1111111	0.5000000	0.56250	1

31-40 of 40 rows Previous 1 2 3 **4** Next

Sifting through Rules

- Set up rule filtering criteria

Reduce Rules

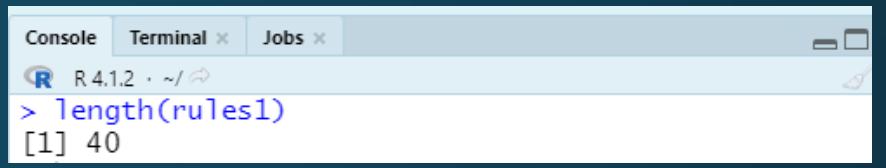
It is prudent to examine only strong rules. Here, we only retain rules with support>=0.001 and confidence>=0.05

```
rules1 = apriori(items,parameter = list(support = 0.001, confidence = 0.05))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 0
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[5 item(s), 9 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [40 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(rules1)   # too many rules?
```

```
## set of 40 rules
##
## rule length distribution (lhs + rhs):sizes
##  1  2  3  4
##  5 16 15  4
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    2.00    2.00    2.45    3.00    4.00
##
## summary of quality measures:
##     support         confidence        coverage           lift
##  Min.   :0.1111   Min.   :0.1667   Min.   :0.1111   Min.   :0.5625
##  1st Qu.:0.1111   1st Qu.:0.3333   1st Qu.:0.2222   1st Qu.:0.7500
##  Median :0.2222   Median :0.5000   Median :0.3333   Median :1.0000
##  Mean   :0.2444   Mean   :0.5774   Mean   :0.4861   Mean   :1.0281
##  3rd Qu.:0.2222   3rd Qu.:0.7500   3rd Qu.:0.6667   3rd Qu.:1.1250
##  Max.   :0.8889   Max.   :1.0000   Max.   :1.0000   Max.   :1.5000
##      count
##  Min.   :1.0
##  1st Qu.:1.0
##  Median :2.0
##  Mean   :2.2
##  3rd Qu.:2.0
##  Max.   :8.0
##
## mining info:
##   data ntransactions support confidence
##  items             9   0.001       0.05
```



We are interested in selecting rules that:

- indicate a strong association (i.e., high lift), and
- are not rare (i.e., high support)

```
inspect(rules1)
```

```
##      lhs                    rhs       support   confidence coverage  lift    count
## [1]  {}                  => {eggs}    0.2222222 0.2222222  1.0000000 1.00000 2
## [2]  {}                  => {bread}   0.3333333 0.3333333  1.0000000 1.00000 3
## [3]  {}                  => {milk}    0.4444444 0.4444444  1.0000000 1.00000 4
## [4]  {}                  => {beer}    0.6666667 0.6666667  1.0000000 1.00000 6
## [5]  {}                  => {diapers} 0.8888889 0.8888889  1.0000000 1.00000 8
## [6]  {eggs}              => {beer}    0.1111111 0.5000000  0.2222222 0.75000 1
## [7]  {beer}              => {eggs}    0.1111111 0.1666667  0.6666667 0.75000 1
## [8]  {eggs}              => {diapers} 0.2222222 1.0000000  0.2222222 1.12500 2
## [9]  {diapers}           => {eggs}    0.2222222 0.2500000  0.8888889 1.12500 2
## [10] {bread}             => {milk}    0.2222222 0.6666667  0.3333333 1.50000 2
## [11] {milk}              => {bread}   0.2222222 0.5000000  0.4444444 1.50000 2
## [12] {bread}             => {beer}    0.2222222 0.6666667  0.3333333 1.00000 2
## [13] {beer}              => {bread}   0.2222222 0.3333333  0.6666667 1.00000 2
## [14] {bread}             => {diapers} 0.2222222 0.6666667  0.3333333 0.75000 2
## [15] {diapers}           => {bread}   0.2222222 0.2500000  0.8888889 0.75000 2
## [16] {milk}              => {beer}    0.2222222 0.5000000  0.4444444 0.75000 2
## [17] {beer}              => {milk}    0.2222222 0.3333333  0.6666667 0.75000 2
## [18] {milk}              => {diapers} 0.3333333 0.7500000  0.4444444 0.84375 3
## [19] {diapers}           => {milk}    0.3333333 0.3750000  0.8888889 0.84375 3
## [20] {beer}              => {diapers} 0.6666667 1.0000000  0.6666667 1.12500 6
## [21] {diapers}           => {beer}    0.6666667 0.7500000  0.8888889 1.12500 6
## [22] {beer,eggs}         => {diapers} 0.1111111 1.0000000  0.1111111 1.12500 1
```

```
quality(rules1) # hide rule details
```

	support <dbl>	confidence <dbl>	coverage <dbl>	lift <dbl>	count <int>
1	0.2222222	0.2222222	1.0000000	1.00000	2
2	0.3333333	0.3333333	1.0000000	1.00000	3
3	0.4444444	0.4444444	1.0000000	1.00000	4
4	0.6666667	0.6666667	1.0000000	1.00000	6
5	0.8888889	0.8888889	1.0000000	1.00000	8

Reduce rules further

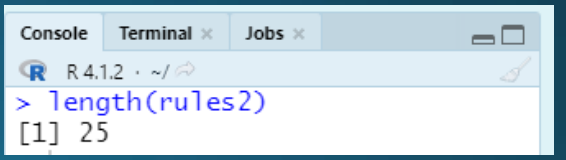
Only keeping rules with a support>=0.2 and confidence>=0.2

```
rules2 = apriori(items,parameter = list(support = 0.2, confidence = 0.2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.2    0.1    1 none FALSE            TRUE       5     0.2      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 1
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[5 item(s), 9 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [25 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
x = inspect(rules2)
```

```
##      lhs                rhs       support   confidence coverage  lift    count
## [1]  {}              => {eggs}    0.2222222 0.2222222  1.0000000 1.00000 2
## [2]  {}              => {bread}   0.3333333 0.3333333  1.0000000 1.00000 3
## [3]  {}              => {milk}    0.4444444 0.4444444  1.0000000 1.00000 4
## [4]  {}              => {beer}    0.6666667 0.6666667  1.0000000 1.00000 6
## [5]  {}              => {diapers} 0.8888889 0.8888889  1.0000000 1.00000 8
## [6]  {eggs}          => {diapers} 0.2222222 1.0000000  0.2222222 1.12500 2
## [7]  {diapers}       => {eggs}    0.2222222 0.2500000  0.8888889 1.12500 2
## [8]  {bread}         => {milk}    0.2222222 0.6666667  0.3333333 1.50000 2
## [9]  {milk}          => {bread}   0.2222222 0.5000000  0.4444444 1.50000 2
## [10] {bread}         => {beer}    0.2222222 0.6666667  0.3333333 1.00000 2
## [11] {beer}          => {bread}   0.2222222 0.3333333  0.6666667 1.00000 2
## [12] {bread}         => {diapers} 0.2222222 0.6666667  0.3333333 0.75000 2
## [13] {diapers}       => {bread}   0.2222222 0.2500000  0.8888889 0.75000 2
## [14] {milk}          => {beer}    0.2222222 0.5000000  0.4444444 0.75000 2
## [15] {beer}          => {milk}    0.2222222 0.3333333  0.6666667 0.75000 2
## [16] {milk}          => {diapers} 0.3333333 0.7500000  0.4444444 0.84375 3
## [17] {diapers}       => {milk}    0.3333333 0.3750000  0.8888889 0.84375 3
## [18] {beer}          => {diapers} 0.6666667 1.0000000  0.6666667 1.12500 6
## [19] {diapers}       => {beer}    0.6666667 0.7500000  0.8888889 1.12500 6
## [20] {beer,bread}    => {diapers} 0.2222222 1.0000000  0.2222222 1.12500 2
## [21] {bread,diapers} => {beer}    0.2222222 1.0000000  0.2222222 1.50000 2
## [22] {beer,diapers}  => {bread}   0.2222222 0.3333333  0.6666667 1.00000 2
## [23] {beer,milk}     => {diapers} 0.2222222 1.0000000  0.2222222 1.12500 2
## [24] {diapers,milk}  => {beer}    0.2222222 0.6666667  0.3333333 1.00000 2
## [25] {beer,diapers}  => {milk}    0.2222222 0.3333333  0.6666667 0.75000 2
```



Among the rules generated, what is the value of largest lift?

```
x[order(x$lift,x$support, decreasing = T),]
```

	lhs		rhs	support	confidence	coverage	lift	count
	<chr>		<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
[8]	{bread}	=>	{milk}	0.2222222	0.6666667	0.3333333	1.50000	2
[9]	{milk}	=>	{bread}	0.2222222	0.5000000	0.4444444	1.50000	2
[21]	{bread,diapers}	=>	{beer}	0.2222222	1.0000000	0.2222222	1.50000	2
[18]	{beer}	=>	{diapers}	0.6666667	1.0000000	0.6666667	1.12500	6
[19]	{diapers}	=>	{beer}	0.6666667	0.7500000	0.8888889	1.12500	6
[6]	{eggs}	=>	{diapers}	0.2222222	1.0000000	0.2222222	1.12500	2
[7]	{diapers}	=>	{eggs}	0.2222222	0.2500000	0.8888889	1.12500	2
[20]	{beer,bread}	=>	{diapers}	0.2222222	1.0000000	0.2222222	1.12500	2
[23]	{beer,milk}	=>	{diapers}	0.2222222	1.0000000	0.2222222	1.12500	2
[5]	{}	=>	{diapers}	0.8888889	0.8888889	1.0000000	1.00000	8

1-10 of 25 rows Previous 1 2 3 Next

Diapers only

Here we are going to narrow our search to just rules with diapers in LHS. Furthermore, we will subset the rules to only those with a lift greater than 1.

```
rules_diapers = apriori(items,parameter=list(support=0.001,confidence=0.05),appearance = list(lhs='diapers'))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 0
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[5 item(s), 9 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [8 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules_diapers = subset(rules_diapers, subset= lift>1)
inspect(rules_diapers)
```

```
##     lhs          rhs    support   confidence coverage  lift  count
## [1] {diapers} => {eggs} 0.2222222 0.25       0.8888889 1.125 2
## [2] {diapers} => {beer} 0.6666667 0.75       0.8888889 1.125 6
```

Diapers only with dplyr

```
library(dplyr)

apriori(data = items, parameter = list(support=0,confidence=0),appearance = list(lhs='diapers'))%>%
  inspect()%>%
  janitor::clean_names()%>%
  filter(count>0, support>0)%>%
  arrange(desc(lift),desc(support))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##           0    0.1    1 none FALSE            TRUE       5       0      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 0
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[5 item(s), 9 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [8 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
##     lhs          rhs     support   confidence coverage  lift    count
## [1] {}        => {eggs}  0.2222222 0.2222222  1.0000000 1.00000 2
## [2] {}        => {bread} 0.3333333 0.3333333  1.0000000 1.00000 3
## [3] {}        => {milk}  0.4444444 0.4444444  1.0000000 1.00000 4
## [4] {}        => {beer}  0.6666667 0.6666667  1.0000000 1.00000 6
## [5] {diapers} => {eggs}  0.2222222 0.2500000  0.8888889 1.12500 2
## [6] {diapers} => {bread} 0.2222222 0.2500000  0.8888889 0.75000 2
## [7] {diapers} => {milk}  0.3333333 0.3750000  0.8888889 0.84375 3
## [8] {diapers} => {beer}  0.6666667 0.7500000  0.8888889 1.12500 6
```

```
> ?clean_names
>
```

Files | Plots | Packages | Help | Viewer

library(| Refresh Help Topic

R: Cleans names of an object (usually a data.frame). | Find in Topic

clean_names {janitor} R Documentation

Cleans names of an object (usually a data.frame).

Description

Resulting names are unique and consist only of the _ character, numbers, and letters. Capitalization preferences can be specified using the `case` parameter.

	lhs	x	rhs	support	confidence	coverage	lift	count
	<chr>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
[8]	{diapers}	=>	{beer}	0.6666667	0.7500000	0.8888889	1.12500	6
[5]	{diapers}	=>	{eggs}	0.2222222	0.2500000	0.8888889	1.12500	2
[4]	{}	=>	{beer}	0.6666667	0.6666667	1.0000000	1.00000	6
[3]	{}	=>	{milk}	0.4444444	0.4444444	1.0000000	1.00000	4
[2]	{}	=>	{bread}	0.3333333	0.3333333	1.0000000	1.00000	3
[1]	{}	=>	{eggs}	0.2222222	0.2222222	1.0000000	1.00000	2
[7]	{diapers}	=>	{milk}	0.3333333	0.3750000	0.8888889	0.84375	3
[6]	{diapers}	=>	{bread}	0.2222222	0.2500000	0.8888889	0.75000	2

8 rows

Rules with only two items


```
rules_2items = apriori(items,parameter=list(support=0.001,confidence=0.05,minlen=2,maxlen=2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5   0.001      2
##  maxlen target  ext
##       2  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 0
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[5 item(s), 9 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2
```

Common slots defined in `ASparameter`:

`support`:

a numeric value for the minimal support of an item set (default: *0.1*)

`minlen`:

an integer value for the minimal number of items per item set (default: 1 item)

`maxlen`:

an integer value for the maximal number of items per item set (default: 10 items)

```
summary(rules_2items)
```

```
## set of 16 rules
##
## rule length distribution (lhs + rhs):sizes
##  2
## 16
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       2       2       2       2       2       2
##
## summary of quality measures:
##     support         confidence        coverage           lift
##  Min.   :0.1111   Min.   :0.1667   Min.   :0.2222   Min.   :0.7500
##  1st Qu.:0.2222   1st Qu.:0.3333   1st Qu.:0.3333   1st Qu.:0.7500
##  Median :0.2222   Median :0.5000   Median :0.5556   Median :0.9219
##  Mean   :0.2778   Mean   :0.5443   Mean   :0.5625   Mean   :0.9805
##  3rd Qu.:0.2500   3rd Qu.:0.6875   3rd Qu.:0.7222   3rd Qu.:1.1250
##  Max.   :0.6667   Max.   :1.0000   Max.   :0.8889   Max.   :1.5000
##      count
##  Min.   :1.00
##  1st Qu.:2.00
##  Median :2.00
##  Mean   :2.50
##  3rd Qu.:2.25
##  Max.   :6.00
##
## mining info:
##   data ntransactions support confidence
##  items             9   0.001       0.05
```

```
inspect(rules_2items)
```

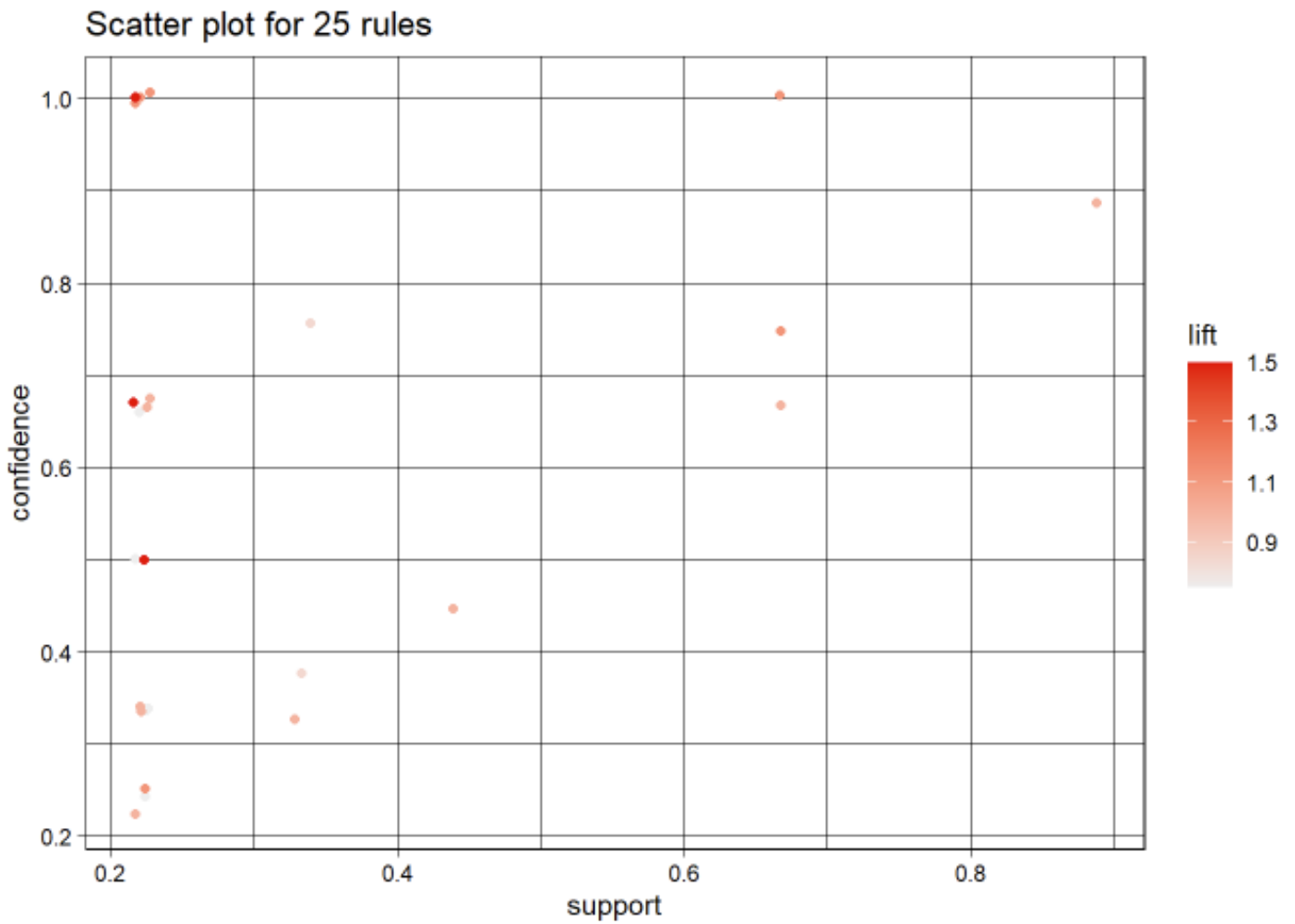
```
##      lhs          rhs       support   confidence coverage  lift    count
## [1]  {eggs}    => {beer}    0.1111111 0.5000000  0.2222222 0.75000 1
## [2]  {beer}    => {eggs}    0.1111111 0.1666667  0.6666667 0.75000 1
## [3]  {eggs}    => {diapers} 0.2222222 1.0000000  0.2222222 1.12500 2
## [4]  {diapers} => {eggs}    0.2222222 0.2500000  0.8888889 1.12500 2
## [5]  {bread}   => {milk}    0.2222222 0.6666667  0.3333333 1.50000 2
## [6]  {milk}    => {bread}   0.2222222 0.5000000  0.4444444 1.50000 2
## [7]  {bread}   => {beer}    0.2222222 0.6666667  0.3333333 1.00000 2
## [8]  {beer}    => {bread}   0.2222222 0.3333333  0.6666667 1.00000 2
## [9]  {bread}   => {diapers} 0.2222222 0.6666667  0.3333333 0.75000 2
## [10] {diapers} => {bread}   0.2222222 0.2500000  0.8888889 0.75000 2
## [11] {milk}    => {beer}    0.2222222 0.5000000  0.4444444 0.75000 2
## [12] {beer}    => {milk}    0.2222222 0.3333333  0.6666667 0.75000 2
## [13] {milk}    => {diapers} 0.3333333 0.7500000  0.4444444 0.84375 3
## [14] {diapers} => {milk}    0.3333333 0.3750000  0.8888889 0.84375 3
## [15] {beer}    => {diapers} 0.6666667 1.0000000  0.6666667 1.12500 6
## [16] {diapers} => {beer}    0.6666667 0.7500000  0.8888889 1.12500 6
```

- No association between bread and beer

Visualize

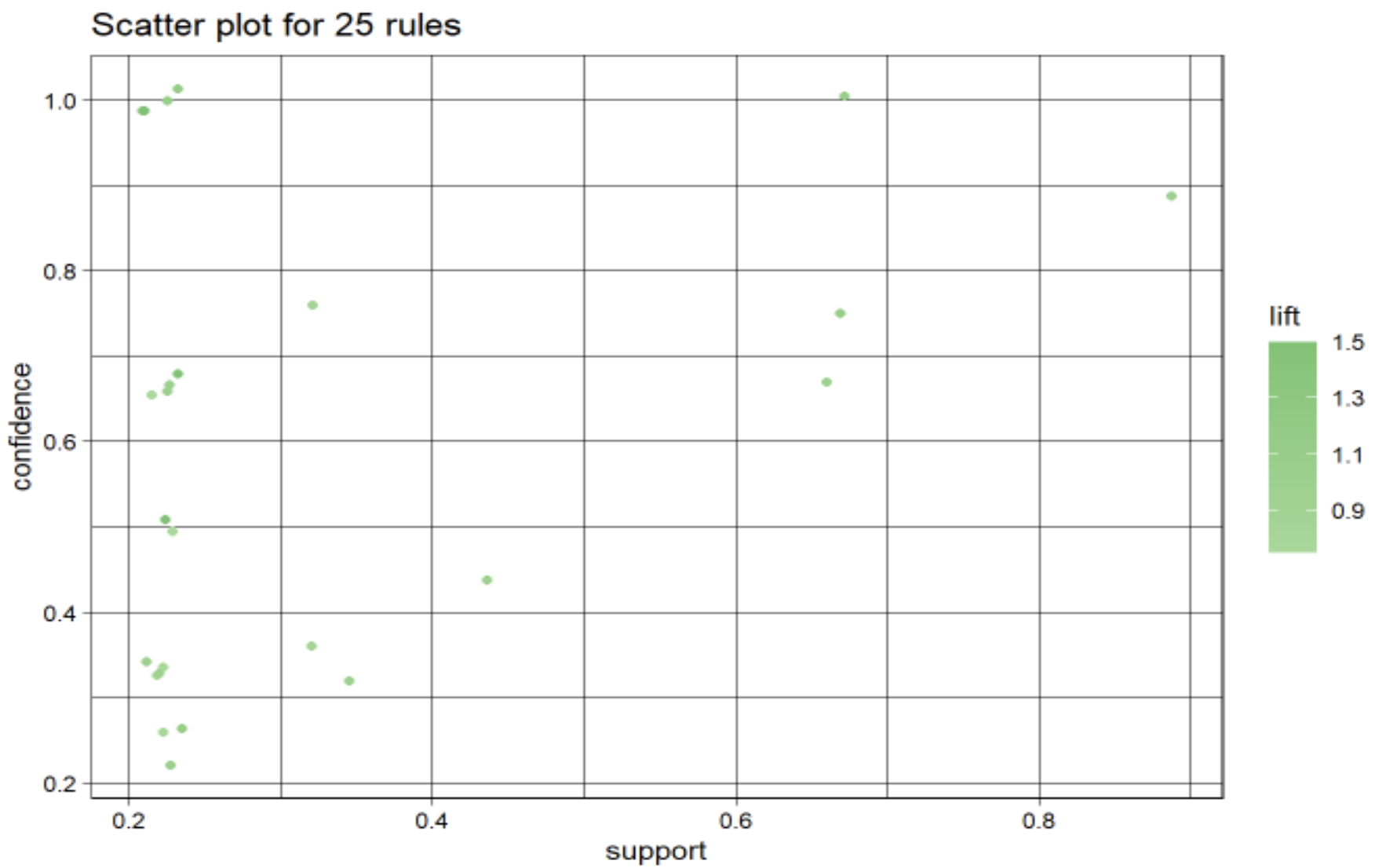
Visualize Association Rules

```
library(RColorBrewer)
plot(rules2) # default method = 'scatterplot', measure=c('support','confidence'), shading='lift'
```



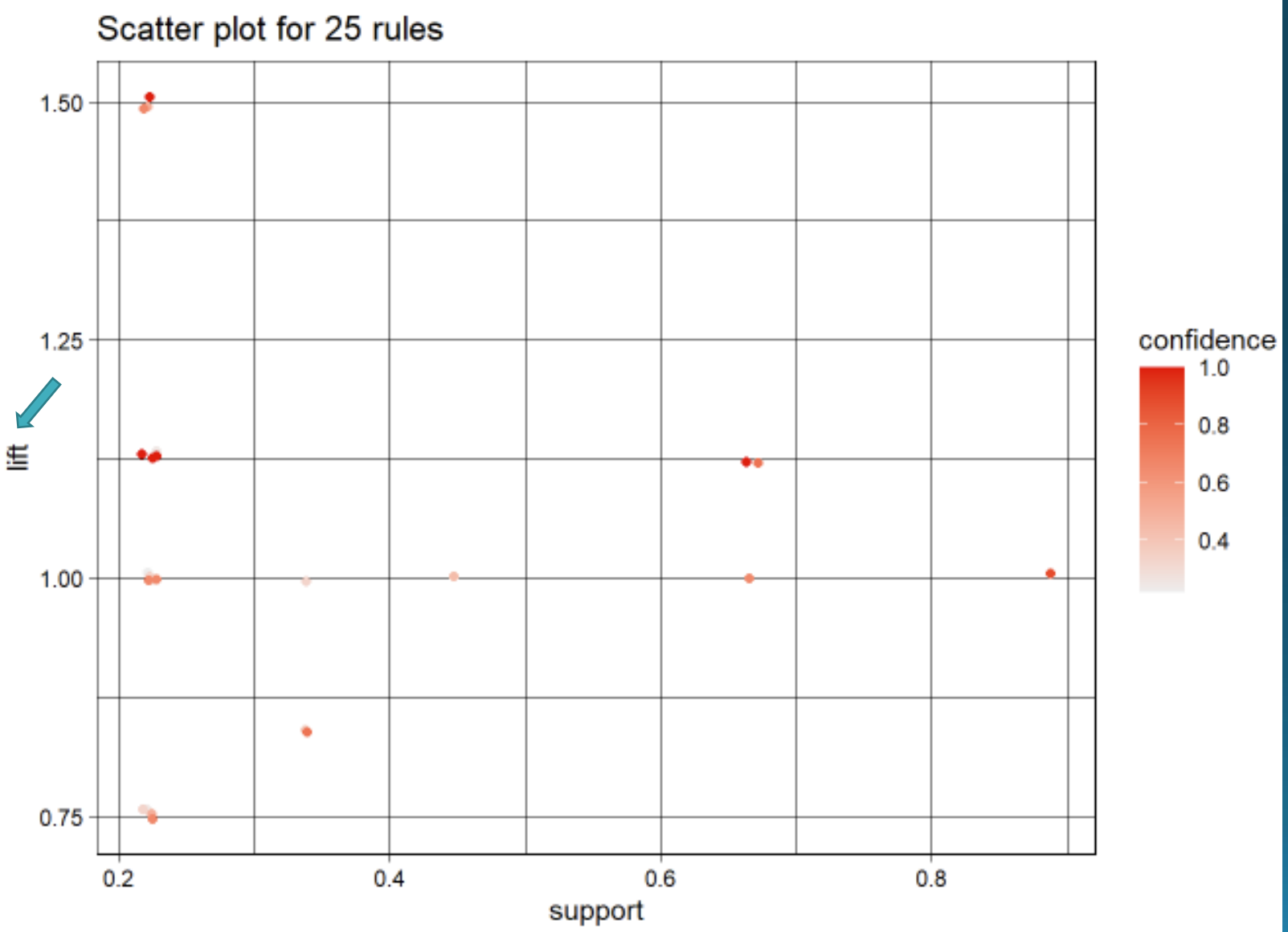
Same plot with the defaults spelt out, a bit of color and jitter. interactive does not work in RMarkdown.

```
plot(rules2,method='scatterplot',measure=c('support','confidence'),
     control=list(jitter=1, col = rev(brewer.pal(9, "Greens")[4:9])),shading = "lift")
```



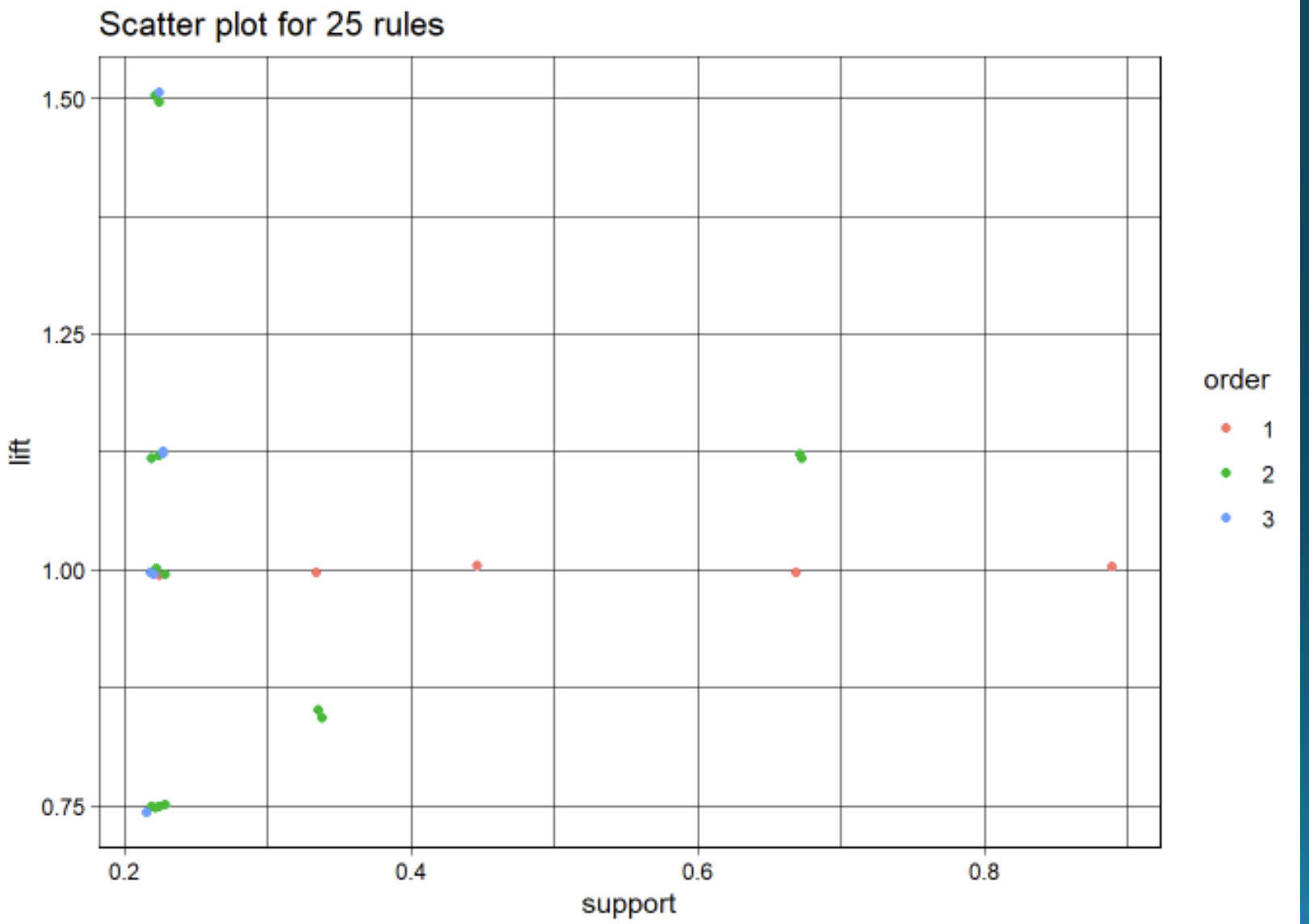
Scatterplot with Lift on the y-axis

```
plot(rules2,method='scatterplot',measure=c('support','lift'),shading='confidence')
```



If one is interested in distinguishing between rules by order

```
plot(rules2,method='scatterplot',measure=c('support','lift'),shading='order')
```



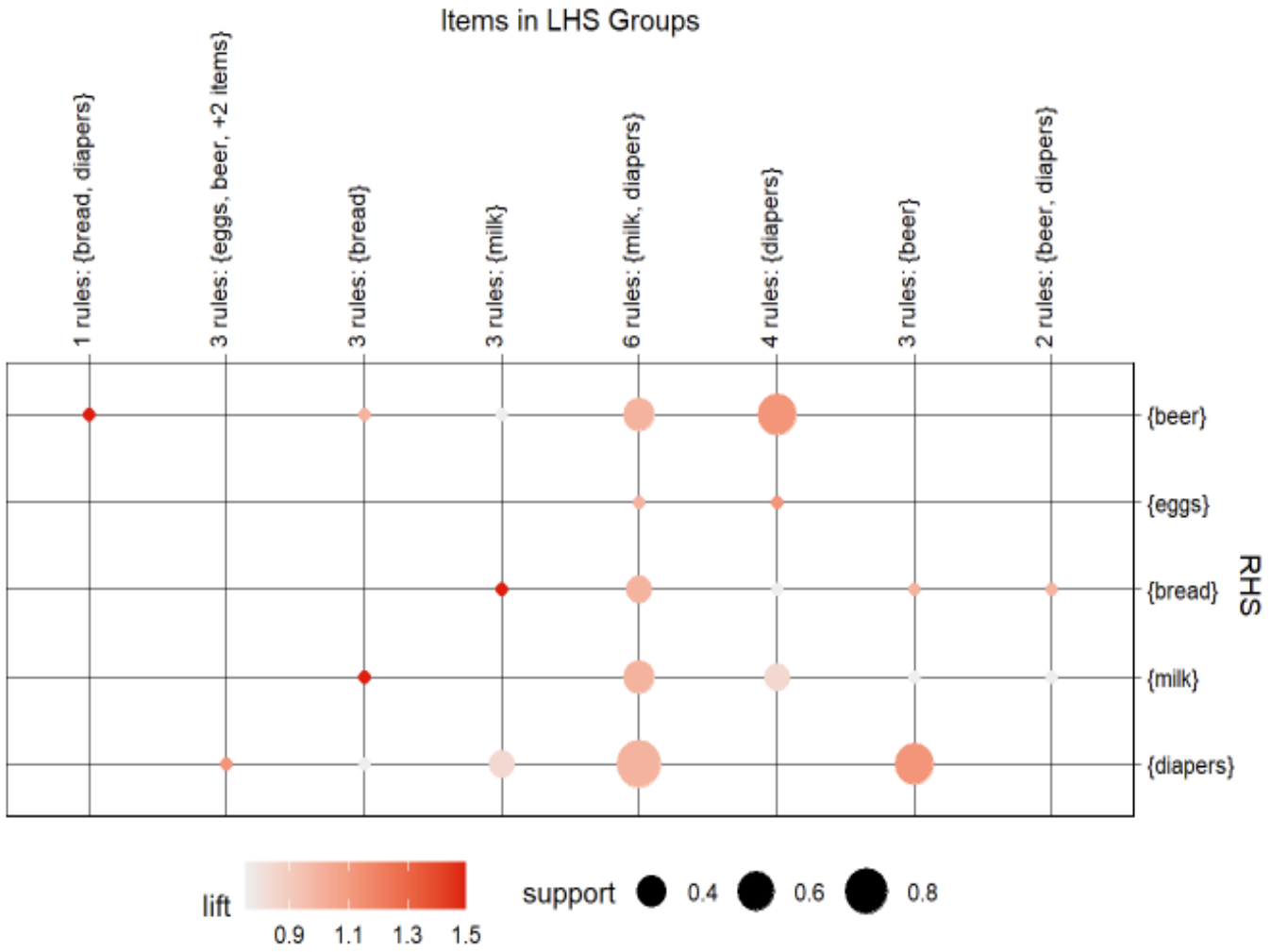

```
Console  Terminal  Jobs
R 4.1.2 · ~/
> summary(rules2)
set of 25 rules

rule length distribution (lhs + rhs):sizes
 1  2  3
 5 14  6
```

Grouped Matrix of Rules

default is measure='support',shading='lift'

```
plot(rules2, method="grouped", measure = 'support',shading='lift')
```



Console | Terminal | Jobs

R 4.1.2 · ~/

```
> inspect(rules2)
```

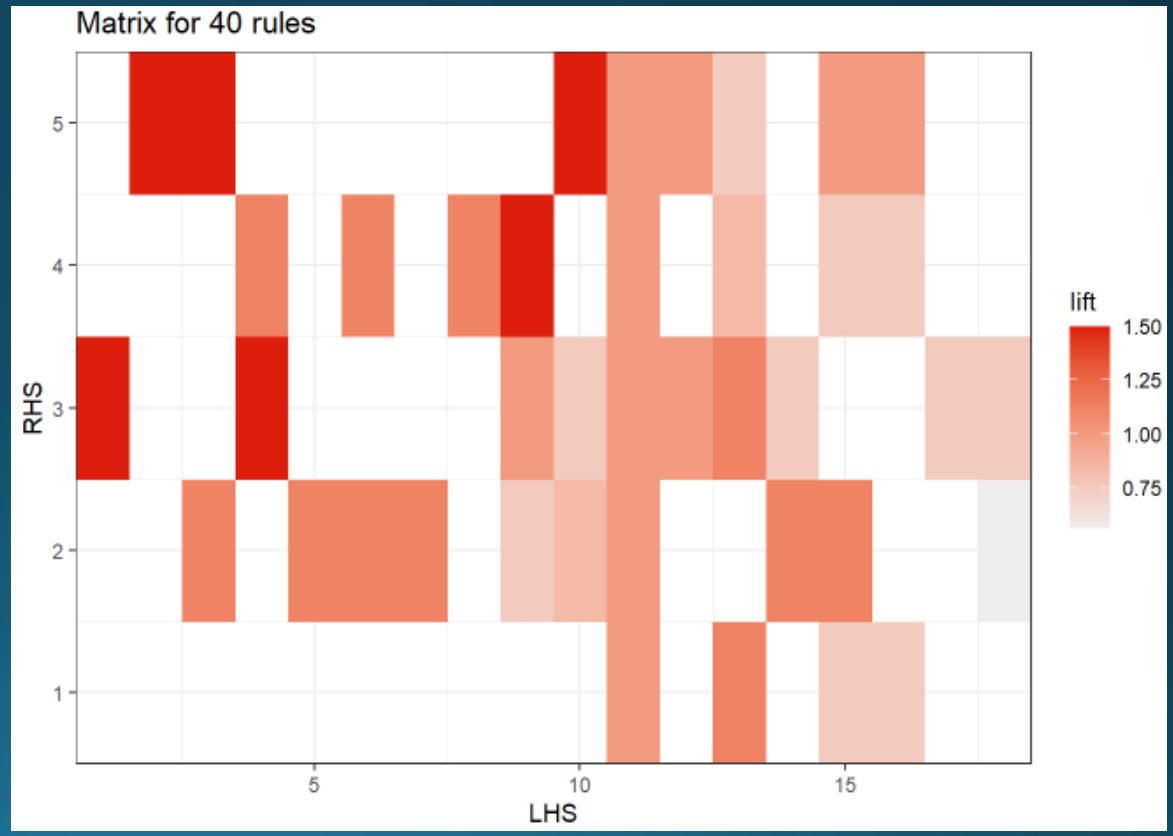
	lhs		rhs	support	confidence	coverage	lift	count
[1]	{}	=>	{eggs}	0.2222222	0.2222222	1.0000000	1.00000	2
[2]	{}	=>	{bread}	0.3333333	0.3333333	1.0000000	1.00000	3
[3]	{}	=>	{milk}	0.4444444	0.4444444	1.0000000	1.00000	4
[4]	{}	=>	{beer}	0.6666667	0.6666667	1.0000000	1.00000	6
[5]	{}	=>	{diapers}	0.8888889	0.8888889	1.0000000	1.00000	8
[6]	{eggs}	=>	{diapers}	0.2222222	1.0000000	0.2222222	1.12500	2
[7]	{diapers}	=>	{eggs}	0.2222222	0.2500000	0.8888889	1.12500	2
[8]	{bread}	=>	{milk}	0.2222222	0.6666667	0.3333333	1.50000	2
[9]	{milk}	=>	{bread}	0.2222222	0.5000000	0.4444444	1.50000	2
[10]	{bread}	=>	{beer}	0.2222222	0.6666667	0.3333333	1.00000	2
[11]	{beer}	=>	{bread}	0.2222222	0.3333333	0.6666667	1.00000	2
[12]	{bread}	=>	{diapers}	0.2222222	0.6666667	0.3333333	0.75000	2
[13]	{diapers}	=>	{bread}	0.2222222	0.2500000	0.8888889	0.75000	2
[14]	{milk}	=>	{beer}	0.2222222	0.5000000	0.4444444	0.75000	2
[15]	{beer}	=>	{milk}	0.2222222	0.3333333	0.6666667	0.75000	2
[16]	{milk}	=>	{diapers}	0.3333333	0.7500000	0.4444444	0.84375	3
[17]	{diapers}	=>	{milk}	0.3333333	0.3750000	0.8888889	0.84375	3
[18]	{beer}	=>	{diapers}	0.6666667	1.0000000	0.6666667	1.12500	6
[19]	{diapers}	=>	{beer}	0.6666667	0.7500000	0.8888889	1.12500	6
[20]	{beer, bread}	=>	{diapers}	0.2222222	1.0000000	0.2222222	1.12500	2
[21]	{bread, diapers}	=>	{beer}	0.2222222	1.0000000	0.2222222	1.50000	2
[22]	{beer, diapers}	=>	{bread}	0.2222222	0.3333333	0.6666667	1.00000	2
[23]	{beer, milk}	=>	{diapers}	0.2222222	1.0000000	0.2222222	1.12500	2
[24]	{diapers, milk}	=>	{beer}	0.2222222	0.6666667	0.3333333	1.00000	2
[25]	{beer, diapers}	=>	{milk}	0.2222222	0.3333333	0.6666667	0.75000	2

Matrix of charts

Matrix of charts is meaningful only in interactive mode

```
plot(rules_all[quality(rules_all)$support>0,],
     method="matrix", measure="lift", control=list(reorder='measure'))
```

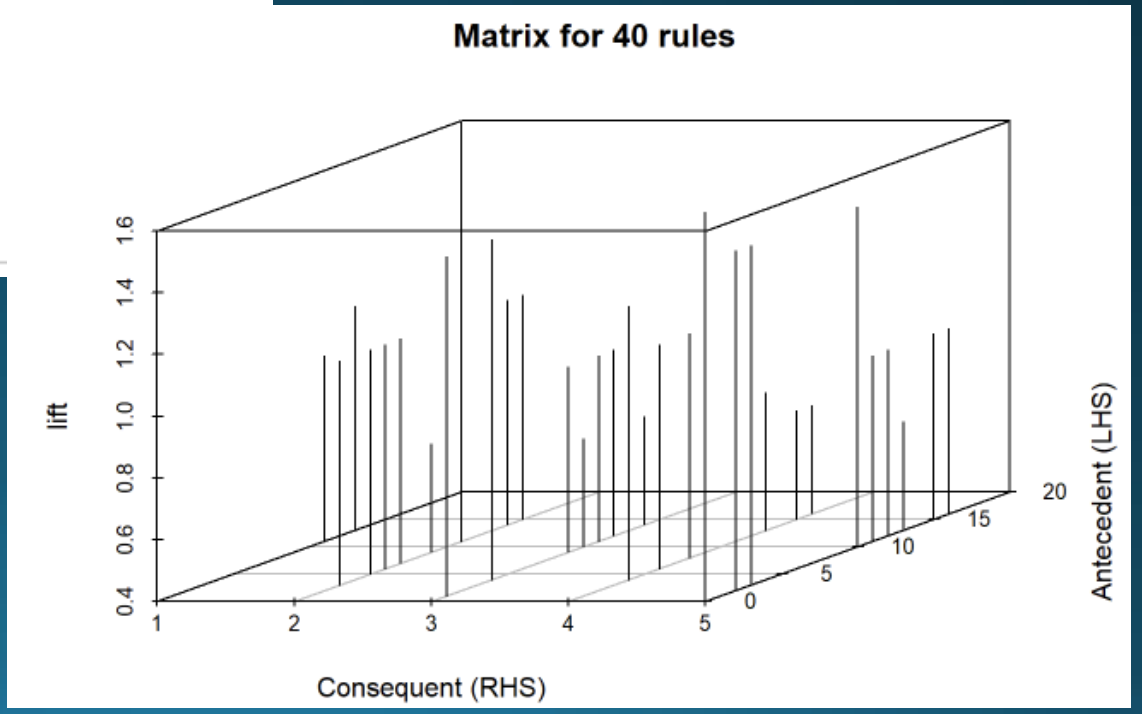
```
## Itemsets in Antecedent (LHS)
##  [1] "{bread,diapers,milk}" "{beer,diapers,milk}"  "{beer,milk}"
##  [4] "{bread,diapers}"      "{beer,eggs}"          "{beer,bread}"
##  [7] "{beer,bread,milk}"    "{beer,bread,diapers}" "{bread}"
## [10] "{milk}"               "{}"                   "{diapers,milk}"
## [13] "{diapers}"            "{eggs}"               "{beer}"
## [16] "{beer,diapers}"       "{diapers,eggs}"       "{bread,milk}"
## Itemsets in Consequent (RHS)
## [1] "{eggs}"    "{diapers}" "{beer}"    "{milk}"    "{bread}"
```



Matrix 3d

```
plot(rules_all[quality(rules_all)$support>0,],
     method="matrix", measure="lift", control=list(reorder='measure'), engine='3d')
```

```
## Itemsets in Antecedent (LHS)
##  [1] "{bread,diapers,milk}" "{beer,diapers,milk}"  "{beer,milk}"
##  [4] "{bread,diapers}"      "{beer,eggs}"          "{beer,bread}"
##  [7] "{beer,bread,milk}"    "{beer,bread,diapers}" "{bread}"
## [10] "{milk}"               "{}"                   "{diapers,milk}"
## [13] "{diapers}"            "{eggs}"               "{beer}"
## [16] "{beer,diapers}"       "{diapers,eggs}"       "{bread,milk}"
## Itemsets in Consequent (RHS)
## [1] "{eggs}"    "{diapers}" "{beer}"    "{milk}"    "{bread}"
```



Network of Rules


```
plot(rules2, method="graph", shading = "lift")
```

Identify rules with Diapers on the LHS

- In some cases, interest may be in a specific association. E.g., consider all transactions with diapers in LHS

Diaper Rules

```
rules_diapers = apriori(items,parameter=list(support=0.001,confidence=0.05),appearance = list(lhs='diapers'))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 0
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[5 item(s), 9 transaction(s)] done [0.00s].
## sorting and recoding items ... [5 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [8 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

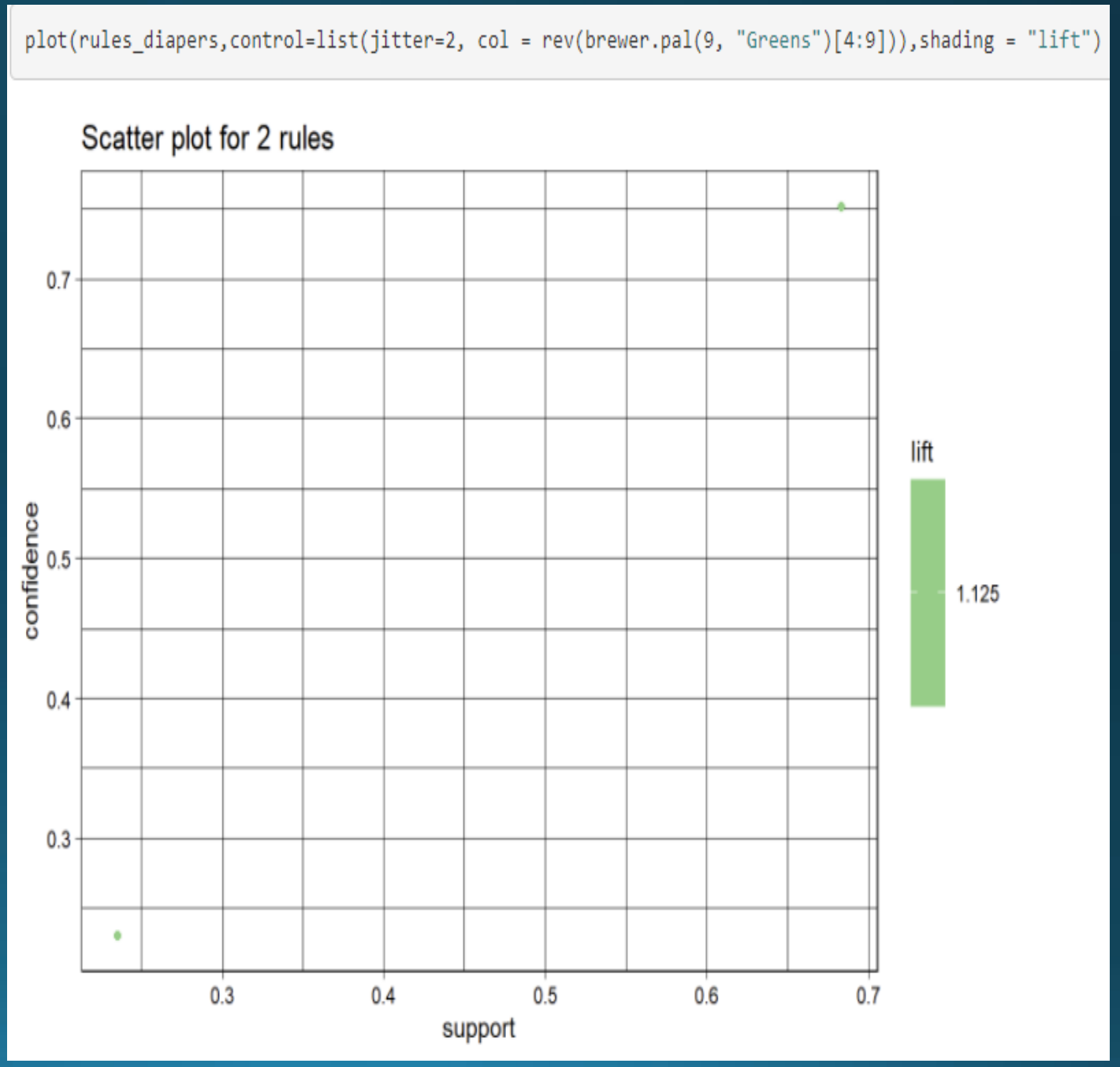
- Let's focus on rules with lift above 1
  - What is the most likely item the shopper will also buy from the grocery store after just picking up diapers? ★

```
rules_diapers = subset(rules_diapers, subset= lift>1)
inspect(rules_diapers)
```

```
##     lhs          rhs    support   confidence coverage  lift  count
## [1] {diapers} => {eggs} 0.2222222 0.25       0.8888889 1.125 2
## [2] {diapers} => {beer} 0.6666667 0.75       0.8888889 1.125 6
```

★

```
plot(rules_diapers,control=list(jitter=2, col = rev(brewer.pal(9, "Greens")[4:9])),shading = "lift")
```



```
plot(rules_diapers, method="graph", shading = "lift")
```

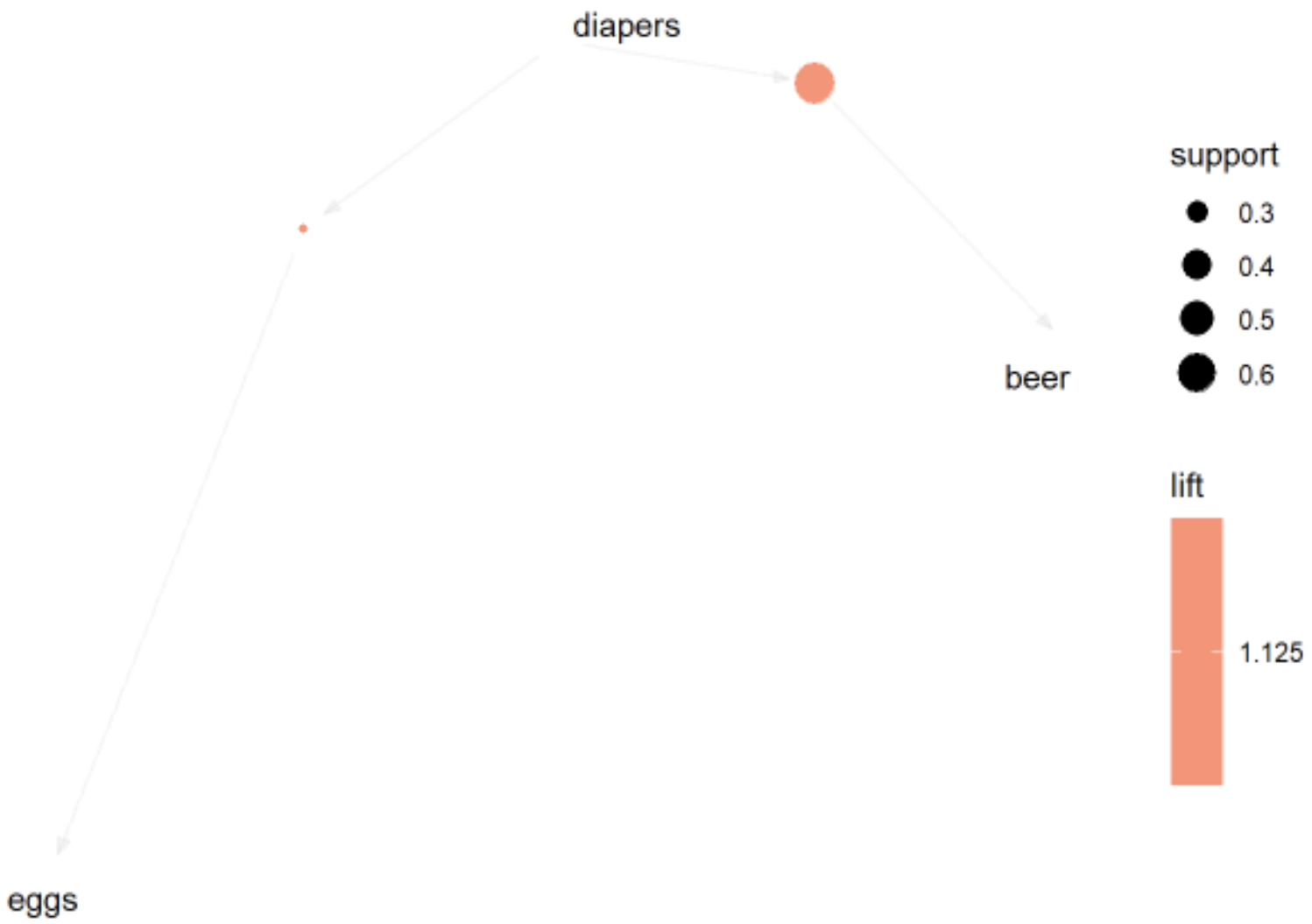


Illustration: Medium-Size Data Set

More on Market Basket Analysis

- Market Basket Analysis generates a large number of rules. It is for the domain expert to identify the rules that are relevant and meet association thresholds
- Rules can be reduced by
  - grouping items using domain knowledge. E.g., combining milk, 1% milk, 2% milk into a single category.
  - text analysis. E.g., removing mentions of colors from products. Differences in color of blue mug and red mug may not be relevant for an association rule.
  - clustering.

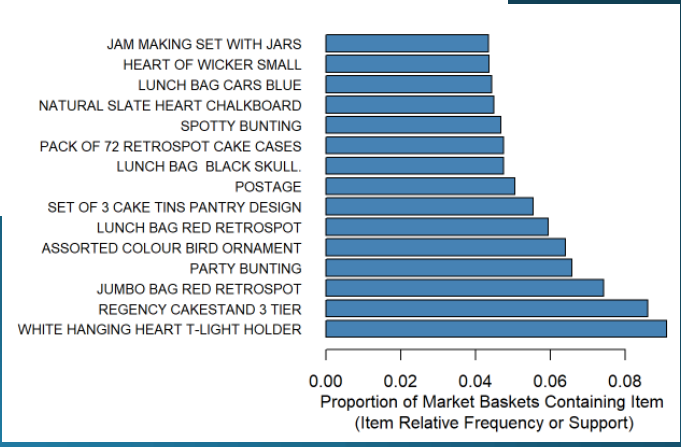
Medium Data

COLUMBIA UNIVERSITY
School of Professional Studies

- Online Retail Data (Source: UCI Machine Learning Repository)
- It contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered online retailer.
- Data contains 541909 transactions including 8 attributes

Attribute Information:

InvoiceNo: Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation.
StockCode: Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product.
Description: Product (item) name. Nominal.
Quantity: The quantities of each product (item) per transaction. Numeric.
InvoiceDate: Invice Date and time. Numeric, the day and time when each transaction was generated.
UnitPrice: Unit price. Numeric, Product price per unit in sterling.
CustomerID: Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.
Country: Country name. Nominal, the name of the country where each customer resides.



Association Rules: Market Basket Analysis

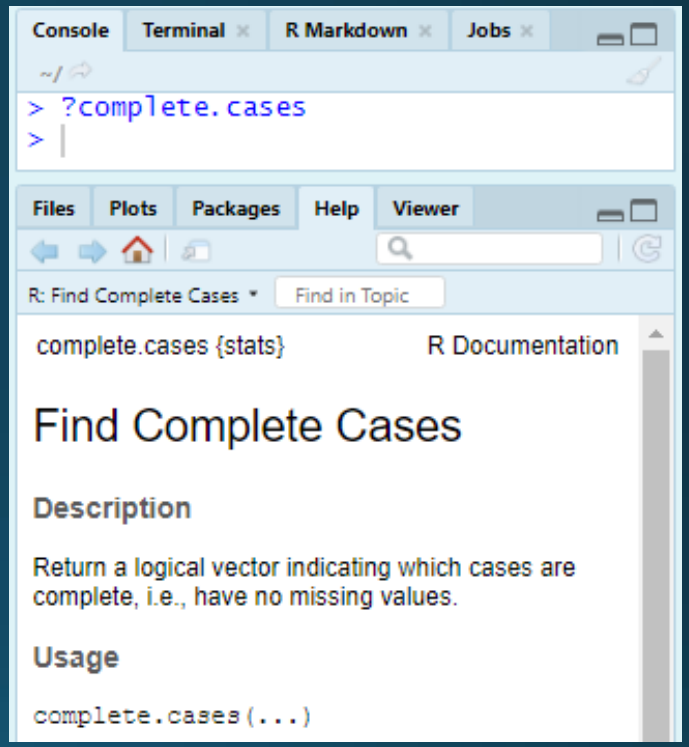
About the Data

This online retail dataset downloaded from UCI Machine Learning Repository contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retailer.

Prepare the Data

```
library(readxl)
library(ggplot2)
library(dplyr)
retail = read_excel('Online_retail.xlsx')
retail = retail[complete.cases(retail), ]
str(retail)
```

```
## tibble [406,829 x 8] (S3: tbl_df/tbl/data.frame)
##  $ InvoiceNo  : chr [1:406829] "536365" "536365" "536365" "536365" ...
##  $ StockCode  : chr [1:406829] "85123A" "71053" "84406B" "84029G" ...
##  $ Description: chr [1:406829] "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANTERN" "CREAM CUPID
HEARTS COAT HANGER" "KNITTED UNION FLAG HOT WATER BOTTLE" ...
##  $ Quantity   : num [1:406829] 6 6 8 6 6 2 6 6 6 32 ...
##  $ InvoiceDate: POSIXct[1:406829], format: "2010-12-01 08:26:00" "2010-12-01 08:26:00" ...
##  $ UnitPrice  : num [1:406829] 2.55 3.39 2.75 3.39 3.39 7.65 4.25 1.85 1.85 1.69 ...
##  $ CustomerID : num [1:406829] 17850 17850 17850 17850 17850 ...
##  $ Country    : chr [1:406829] "United Kingdom" "United Kingdom" "United Kingdom" "United Kingdom" ...
```



```
library(stringr)
retail$Date = as.Date(retail$InvoiceDate)
retail$Description = as.factor(retail$Description)
baskets = retail %>%
  arrange(desc(CustomerID))%>%
  mutate(Description = str_to_title(Description))%>%
  group_by(CustomerID,Date)%>%
  summarise(items = paste(Description,collapse=','))
baskets$CustomerID=NULL
baskets$Date=NULL
write.csv(baskets,'baskets.csv',quote = F,row.names = F)
```

basketAnalysis.Rmd | retail

Filter

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	Date
1	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850	United Kingdom	2010-12-01
2	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850	United Kingdom	2010-12-01
3	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850	United Kingdom	2010-12-01
4	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850	United Kingdom	2010-12-01
5	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850	United Kingdom	2010-12-01
6	536365	22752	SET 7 BABUSHKA NESTING BOXES	2	2010-12-01 08:26:00	7.65	17850	United Kingdom	2010-12-01
7	536365	21730	GLASS STAR FROSTED T-LIGHT HOLDER	6	2010-12-01 08:26:00	4.25	17850	United Kingdom	2010-12-01
8	536366	22633	HAND WARMER UNION JACK	6	2010-12-01 08:28:00	1.85	17850	United Kingdom	2010-12-01
9	536366	22632	HAND WARMER RED POLKA DOT	6	2010-12-01 08:28:00	1.85	17850	United Kingdom	2010-12-01
10	536367	84879	ASSORTED COLOUR BIRD ORNAMENT	32	2010-12-01 08:34:00	1.69	13047	United Kingdom	2010-12-01
11	536367	22745	POPPY'S PLAYHOUSE BEDROOM	6	2010-12-01 08:34:00	2.10	13047	United Kingdom	2010-12-01
12	536367	22748	POPPY'S PLAYHOUSE KITCHEN	6	2010-12-01 08:34:00	2.10	13047	United Kingdom	2010-12-01
13	536367	22749	FELTCRAFT PRINCESS CHARLOTTE DOLL	8	2010-12-01 08:34:00	3.75	13047	United Kingdom	2010-12-01
14	536367	22310	IVORY KNITTED MUG COSY	6	2010-12-01 08:34:00	1.65	13047	United Kingdom	2010-12-01
15	536367	84969	BOX OF 6 ASSORTED COLOUR TEASPOONS	6	2010-12-01 08:34:00	4.25	13047	United Kingdom	2010-12-01

Showing 1 to 16 of 406,829 entries, 9 total columns

baskets

Filter

	items
1	Medium Ceramic Top Storage Jar,Medium Ceramic Top Stor...
2	Black Candelabra T-Light Holder,Airline Bag Vintage Jet Set ...
3	Pink New Baroquecandlestick Candle,Blue New Baroque Ca...
4	Airline Bag Vintage Jet Set White,Airline Bag Vintage Jet Set ...
5	Rabbit Night Light,Regency Tea Strainer,Regency Tea Plate G...
6	Set Of 60 Vintage Leaf Cake Cases,Set 40 Heart Shape Petit ...
7	Mini Lights Woodland Mushrooms,Pink Goose Feather Tree ...
8	Classic Chrome Bicycle Bell,Bicycle Puncture Repair Kit,Boom...
9	72 Sweetheart Fairy Cake Cases,60 Cake Cases Dolly Girl Des...
10	Pack Of 12 Red Retrospot Tissues,Pack Of 12 Hearts Design ...
11	Doughnut Lip Gloss,Ice Cream Pen Lip Gloss,Ice Cream Sund...
12	Doughnut Lip Gloss,Ice Cream Pen Lip Gloss,Postage
13	Parisienne Curio Cabinet,Sweetheart Wall Tidy,Pink Heart S...
14	Chocolate This Way Metal Sign,Metal Sign Neighbourhood ...
15	Wooden Happy Birthday Garland,Pink Doughnut Trinket Pot,...
16	Postage,Deluxe Sewing Kit,Pink Heart Shape Egg Frying Pan,...
17	Ceramic Heart Fairy Cake Money Bank,Ceramic Cake Design ...
18	Pink Heart Shape Egg Frying Pan,Ceramic Cake Design Spott...
19	Antique Glass Pedestal Bowl,Pantry Magnetic Shopping List,...
20	Open Closed Metal Sign,Set Of 6 Spice Tins Pantry Design,S...
21	Petit Tray Chic,Pantry Rolling Pin,Pantry Pastry Brush,Woodla...
22	Ceramic Cake Stand + Hanging Cakes,Mini Cake Stand With...

Showing 1 to 22 of 19,296 entries, 1 total columns

Environment | History | Connections | Tutorial

Import Dataset | 486 MiB | List

R | Global Environment

Data

baskets	19296 obs. of 1 variable
retail	406829 obs. of 9 variables

Read Clean Data

Load Libraries

```
library(arules); library(arulesViz)
```

Use read.transactions to construct an arules object

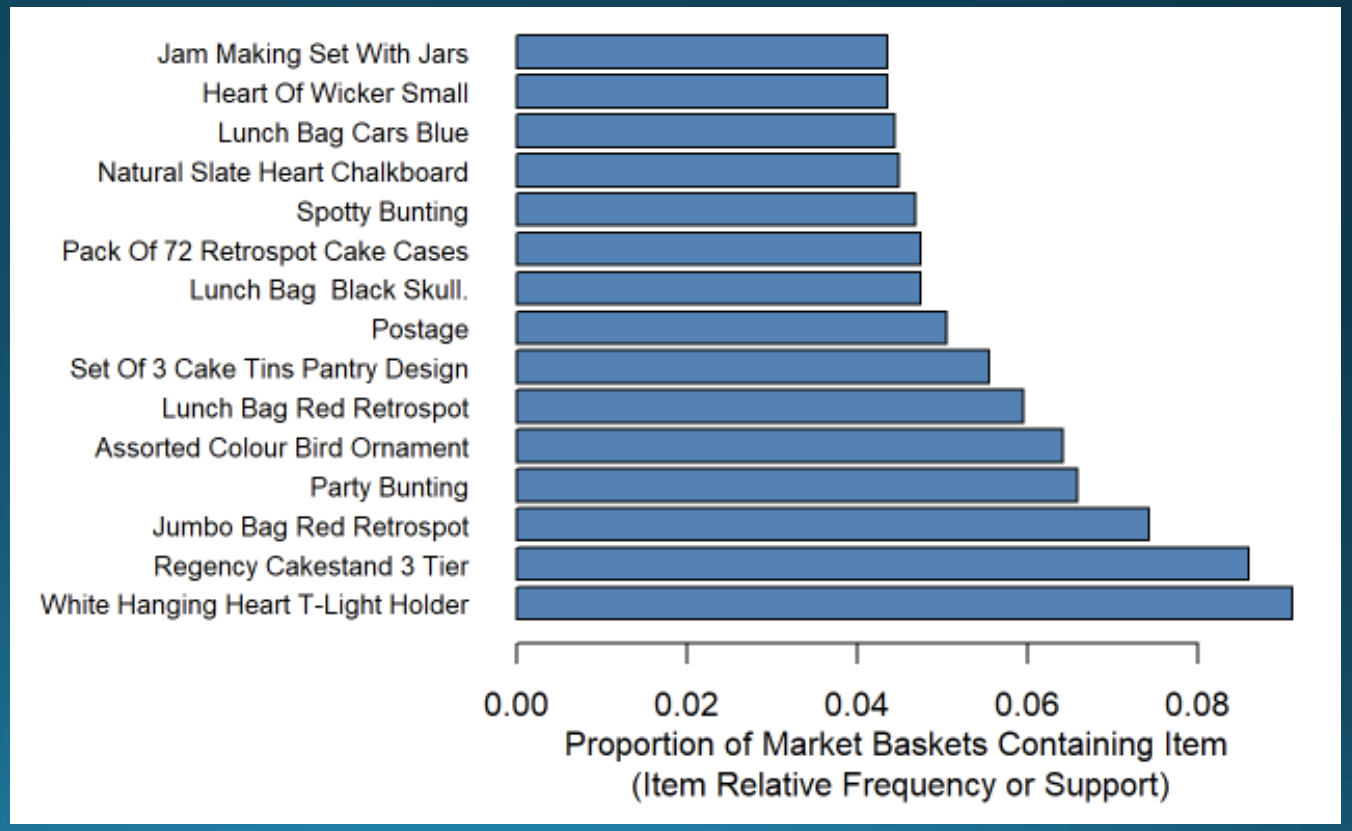
```
items = read.transactions('baskets.csv', format = 'basket', sep=',',skip=1)
items
```

```
## transactions in sparse format with
##  19296 transactions (rows) and
##  7868 items (columns)
```

Explore Data

Examine top 15 items with support>=0.04

```
itemFrequencyPlot(items, support = 0.04, cex.names=0.8,
                  type = "relative", horiz = TRUE, col = "steelblue", las = 1,topN=15,
                  xlab = paste("Proportion of Market Baskets Containing Item",
                               "\n(Item Relative Frequency or Support)"))
```



Explore relationships among items using 'count'. View relationship among first four items

```
crosstab = crossTable(items, measure="count", sort=TRUE)
crosstab[1:4,1:4]
```

```
##                                   White Hanging Heart T-Light Holder
## White Hanging Heart T-Light Holder                               1758
## Regency Cakestand 3 Tier                                          210
## Jumbo Bag Red Retrospot                                           202
## Party Bunting                                                     251
##                                   Regency Cakestand 3 Tier
## White Hanging Heart T-Light Holder                      210
## Regency Cakestand 3 Tier                               1660
## Jumbo Bag Red Retrospot                                 128
## Party Bunting                                           212
##                                   Jumbo Bag Red Retrospot Party Bunting
## White Hanging Heart T-Light Holder                     202           251
## Regency Cakestand 3 Tier                               128           212
## Jumbo Bag Red Retrospot                               1434           158
## Party Bunting                                          158          1271
```

Note small values of support.

```
crosstab = crossTable(items, measure="support", sort=TRUE)
crosstab[1:4,1:4]
```

```
##                                   White Hanging Heart T-Light Holder
## White Hanging Heart T-Light Holder                         0.09110697
## Regency Cakestand 3 Tier                                   0.01088308
## Jumbo Bag Red Retrospot                                    0.01046849
## Party Bunting                                              0.01300788
##                                   Regency Cakestand 3 Tier
## White Hanging Heart T-Light Holder               0.010883085
## Regency Cakestand 3 Tier                         0.086028192
## Jumbo Bag Red Retrospot                          0.006633499
## Party Bunting                                    0.010986733
##                                   Jumbo Bag Red Retrospot Party Bunting
## White Hanging Heart T-Light Holder              0.010468491   0.013007877
## Regency Cakestand 3 Tier                        0.006633499   0.010986733
## Jumbo Bag Red Retrospot                         0.074315920   0.008188226
## Party Bunting                                   0.008188226   0.065868574
```

Analyze Data

Identify list association of rules. Are there too many rules?

```
rules1 = apriori(items,parameter = list(support = 0.001, confidence = 0.05))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 19
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[7868 item(s), 19296 transaction(s)] done [0.14s].
## sorting and recoding items ... [2407 item(s)] done [0.01s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.29s].
## writing ... [383035 rule(s)] done [0.07s].
## creating S4 object  ... done [0.10s].
```

```
summary(rules1)   # too many rules?
```

```
## set of 383035 rules
##
## rule length distribution (lhs + rhs):sizes
##     1     2     3     4     5     6     7     8     9    10
##     8 41869 71975 92432 95895 57414 18900  3656   756   130
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   4.000   4.342   5.000  10.000
##
## summary of quality measures:
##     support           confidence        coverage             lift
##  Min.   :0.001036   Min.   :0.0500   Min.   :0.001037   Min.   :  0.6111
##  1st Qu.:0.001088   1st Qu.:0.3514   1st Qu.:0.001607   1st Qu.: 10.8839
##  Median :0.001244   Median :0.6136   Median :0.002332   Median : 16.7348
##  Mean   :0.001514   Mean   :0.5628   Mean   :0.004833   Mean   : 25.4628
##  3rd Qu.:0.001607   3rd Qu.:0.7778   3rd Qu.:0.004457   3rd Qu.: 24.8437
##  Max.   :0.091107   Max.   :1.0000   Max.   :1.000000   Max.   :622.4516
##      count
##  Min.   :  20.00
##  1st Qu.:  21.00
##  Median :  24.00
##  Mean   :  29.21
##  3rd Qu.:  31.00
##  Max.   :1758.00
##
## mining info:
##   data ntransactions support confidence
##  items         19296   0.001       0.05
##                                                                    call
##  apriori(data = items, parameter = list(support = 0.001, confidence = 0.05))
```

Reduce Rules by placing support and confidence thresholds

```
rules2 = apriori(items,parameter = list(support = 0.02, confidence = 0.05))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.05    0.1    1 none FALSE            TRUE       5    0.02      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 385
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[7868 item(s), 19296 transaction(s)] done [0.13s].
## sorting and recoding items ... [143 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.01s].
## writing ... [32 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

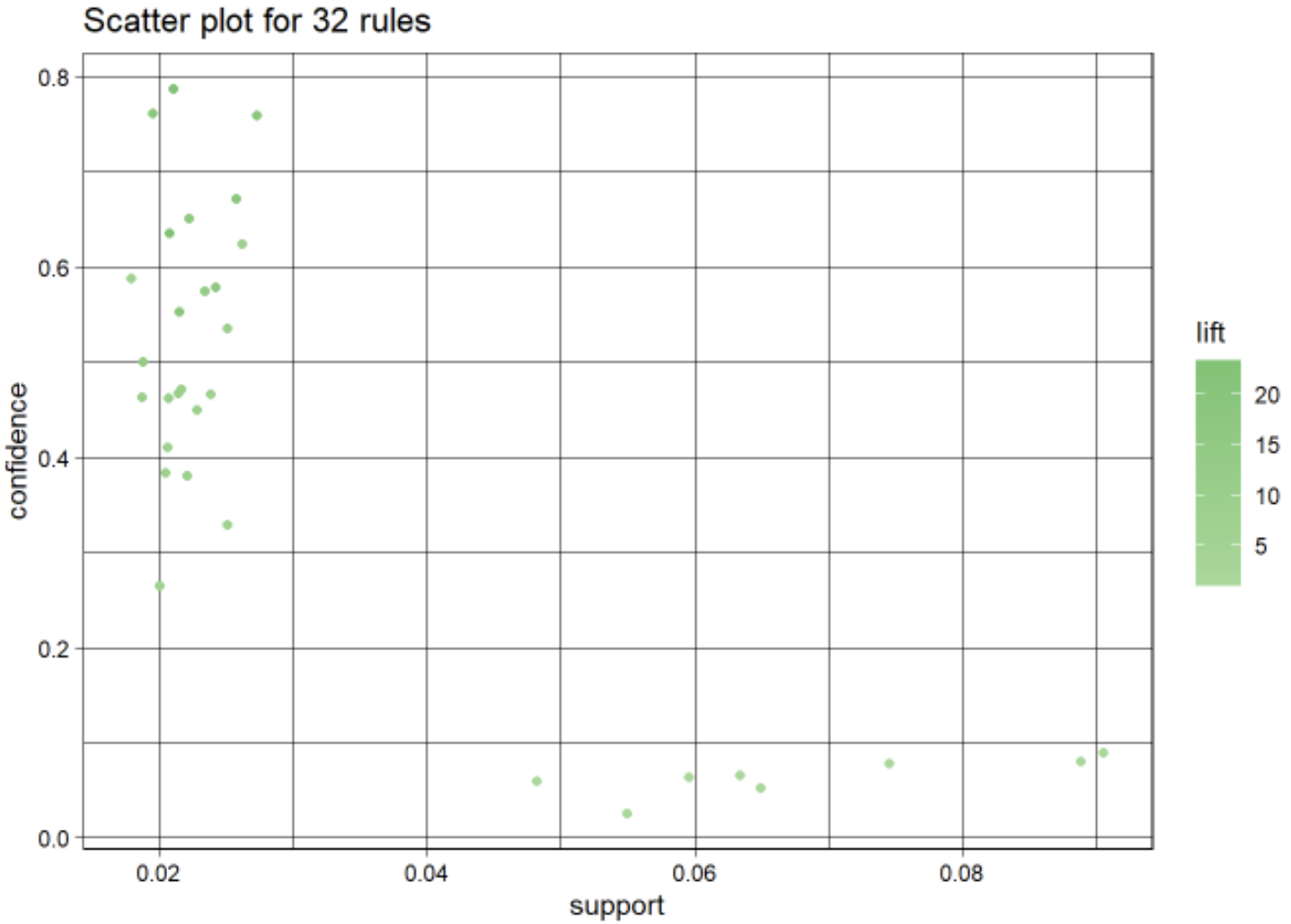
```
summary(rules2)
```

```
## set of 32 rules
##
## rule length distribution (lhs + rhs):sizes
##  1  2
##  8 24
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    1.75    2.00    1.75    2.00    2.00
##
## summary of quality measures:
##     support          confidence         coverage            lift
##  Min.   :0.02006   Min.   :0.05053   Min.   :0.02731   Min.   : 1.000
##  1st Qu.:0.02094   1st Qu.:0.22518   1st Qu.:0.03882   1st Qu.: 6.019
##  Median :0.02319   Median :0.47151   Median :0.04436   Median : 8.829
##  Mean   :0.03376   Mean   :0.41644   Mean   :0.28338   Mean   : 9.957
##  3rd Qu.:0.03187   3rd Qu.:0.57407   3rd Qu.:0.30574   3rd Qu.:14.672
##  Max.   :0.09111   Max.   :0.79696   Max.   :1.00000   Max.   :23.442
##      count
##  Min.   : 387.0
##  1st Qu.: 404.0
##  Median : 447.5
##  Mean   : 651.3
##  3rd Qu.: 615.0
##  Max.   :1758.0
##
## mining info:
##   data ntransactions support confidence
##  items         19296    0.02       0.05
##                                                                      call
##  apriori(data = items, parameter = list(support = 0.02, confidence = 0.05))
```

Visualize

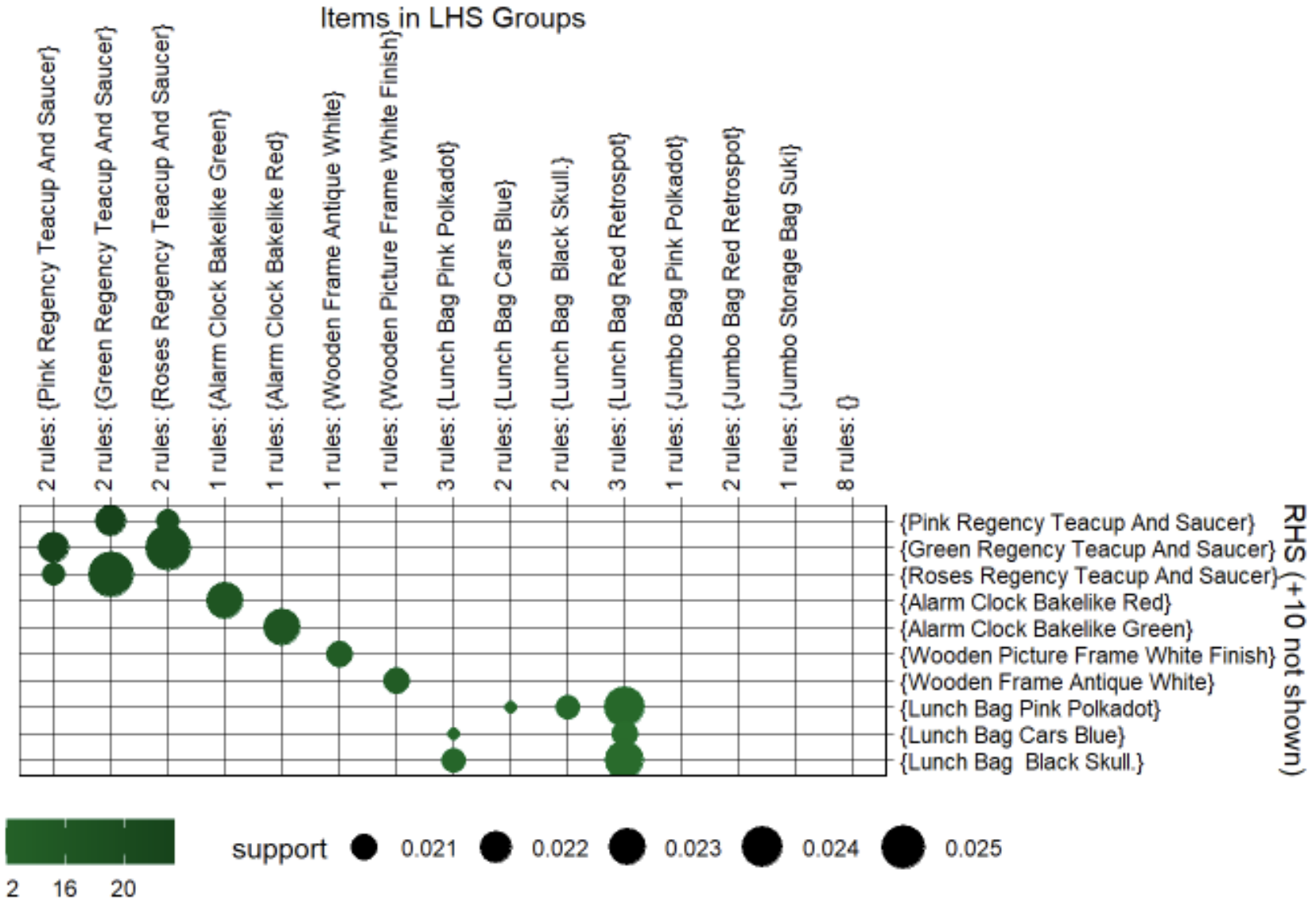
Visualize association rules

```
library(RColorBrewer)
plot(rules2,control=list(jitter=2, col = rev(brewer.pal(9, "Greens")[4:9])),shading = "lift")
```



Grouped matrix of rules

```
plot(rules2, method="grouped",control=list(col = rev(brewer.pal(9, "Greens")[4:9])))
```



Teacup Only

Select Rules with Teacup in the LHS

```
teacup_left = subset(rules2,subset=rhs %pin% 'Teacup')
inspect(teacup_left)
```

```
##     lhs                                rhs                                support confidence   coverage
lift count
## [1] {Pink Regency Teacup And Saucer}  => {Green Regency Teacup And Saucer} 0.02176617  0.7969639 0.02731136
23.44240   420
## [2] {Green Regency Teacup And Saucer} => {Pink Regency Teacup And Saucer}  0.02176617  0.6402439 0.03399668
23.44240   420
## [3] {Pink Regency Teacup And Saucer}  => {Roses Regency Teacup And Saucer} 0.02067786  0.7571157 0.02731136
19.50508   399
## [4] {Roses Regency Teacup And Saucer} => {Pink Regency Teacup And Saucer}  0.02067786  0.5327103 0.03881633
19.50508   399
## [5] {Green Regency Teacup And Saucer} => {Roses Regency Teacup And Saucer} 0.02565299  0.7545732 0.03399668
19.43958   495
## [6] {Roses Regency Teacup And Saucer} => {Green Regency Teacup And Saucer} 0.02565299  0.6608812 0.03881633
19.43958   495
```

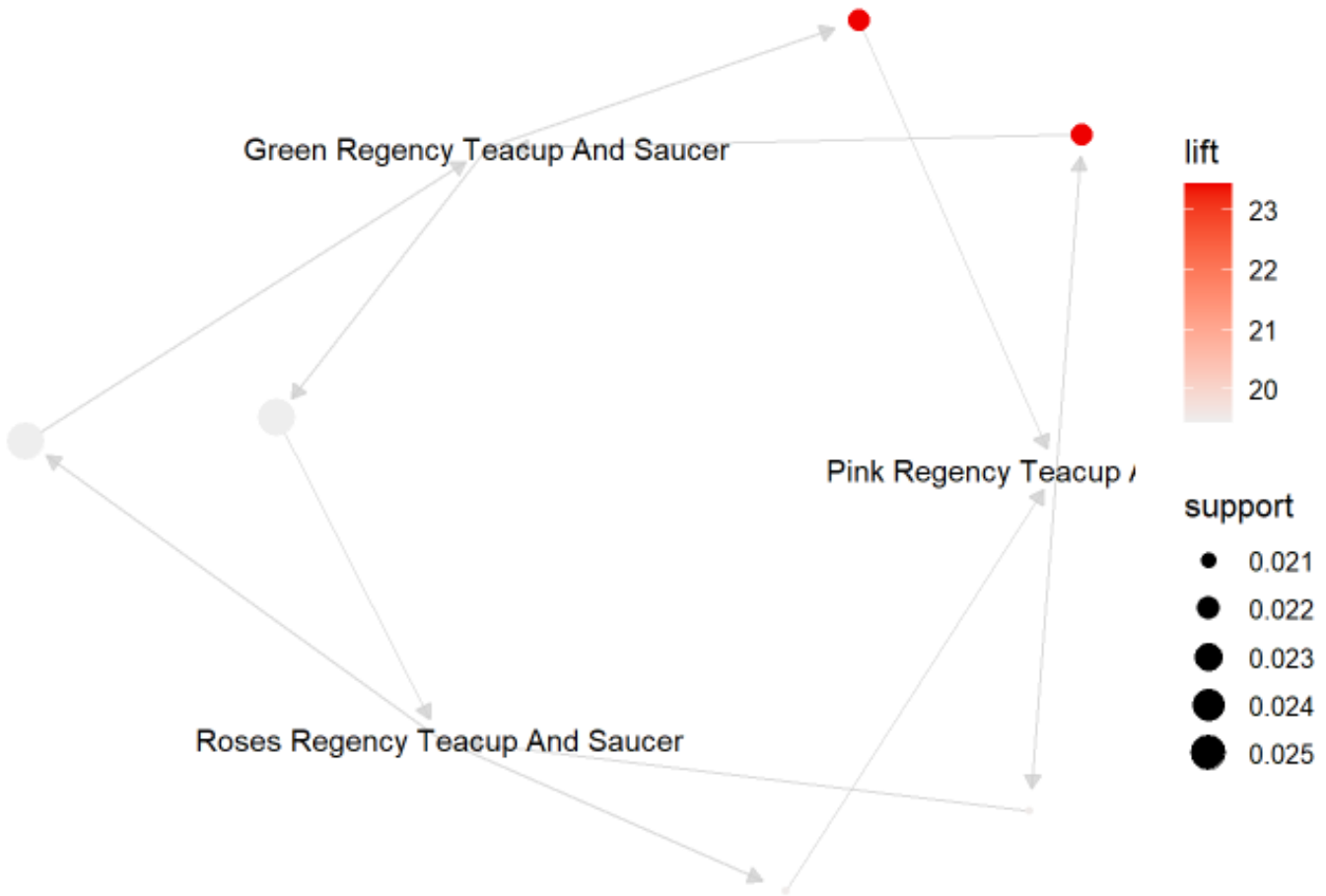
Sort rules by lift and limit to 10

```
top_teacup_left = head(sort(teacup_left,decreasing=T,by='lift'),10)
inspect(top_teacup_left)
```


```
##     lhs                                rhs                                support confidence   coverage
lift count
## [1] {Pink Regency Teacup And Saucer}  => {Green Regency Teacup And Saucer} 0.02176617  0.7969639 0.02731136
23.44240   420
## [2] {Green Regency Teacup And Saucer} => {Pink Regency Teacup And Saucer}  0.02176617  0.6402439 0.03399668
23.44240   420
## [3] {Pink Regency Teacup And Saucer}  => {Roses Regency Teacup And Saucer} 0.02067786  0.7571157 0.02731136
19.50508   399
## [4] {Roses Regency Teacup And Saucer} => {Pink Regency Teacup And Saucer}  0.02067786  0.5327103 0.03881633
19.50508   399
## [5] {Green Regency Teacup And Saucer} => {Roses Regency Teacup And Saucer} 0.02565299  0.7545732 0.03399668
19.43958   495
## [6] {Roses Regency Teacup And Saucer} => {Green Regency Teacup And Saucer} 0.02565299  0.6608812 0.03881633
19.43958   495
```

Network graphs of teacup rules

```
plot(top_teacup_left, method="graph", shading = "lift")
```

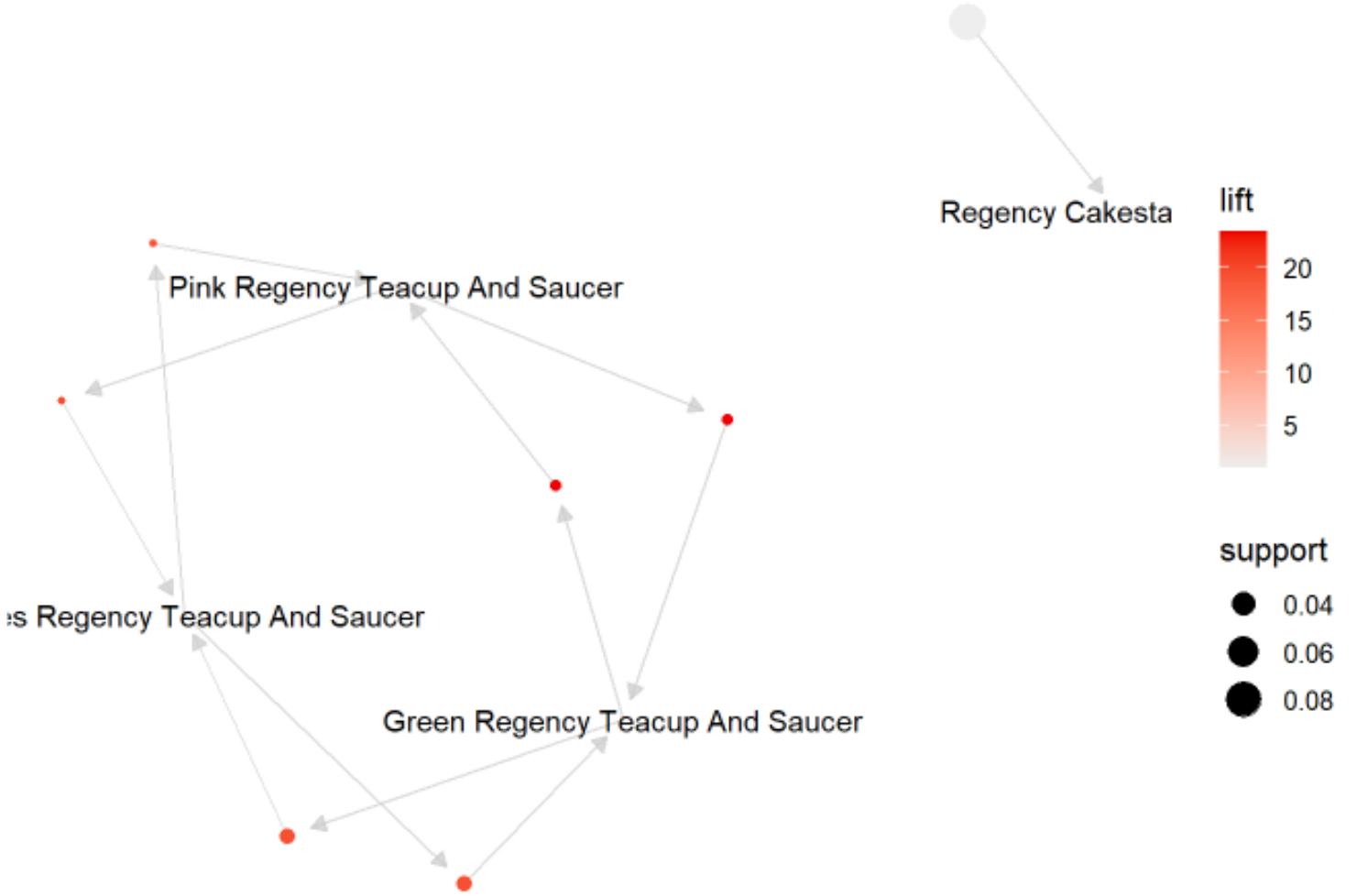


Similarly, select rules with Regency in the RHS, sort and show top 10 by lift

```
regency_right = subset(rules2,subset=rhs %pin% 'Regency')
top_regency_right = head(sort(regency_right,decreasing=T,by='lift'),10)
inspect(top_regency_right)
```

```
##     lhs                                  rhs                                    support confidence   coverage
lift count
## [1] {Pink Regency Teacup And Saucer}  => {Green Regency Teacup And Saucer} 0.02176617 0.79696395 0.02731136 2
3.44240   420
## [2] {Green Regency Teacup And Saucer} => {Pink Regency Teacup And Saucer}  0.02176617 0.64024390 0.03399668 2
3.44240   420
## [3] {Pink Regency Teacup And Saucer}  => {Roses Regency Teacup And Saucer} 0.02067786 0.75711575 0.02731136 1
9.50508   399
## [4] {Roses Regency Teacup And Saucer} => {Pink Regency Teacup And Saucer}  0.02067786 0.53271028 0.03881633 1
9.50508   399
## [5] {Green Regency Teacup And Saucer} => {Roses Regency Teacup And Saucer} 0.02565299 0.75457317 0.03399668 1
9.43958   495
## [6] {Roses Regency Teacup And Saucer} => {Green Regency Teacup And Saucer} 0.02565299 0.66088117 0.03881633 1
9.43958   495
## [7] {}                                => {Regency Cakestand 3 Tier}        0.08602819 0.08602819 1.00000000
1.00000  1660
```

```
plot(top_regency_right, method="graph", shading = "lift")
```



Summary

- In this module we,
  - discussed applications of association rules
  - described mathematical criteria for evaluating rules
  - conducted market basket analysis
  - explained the importance of domain expertise for interpreting association rules