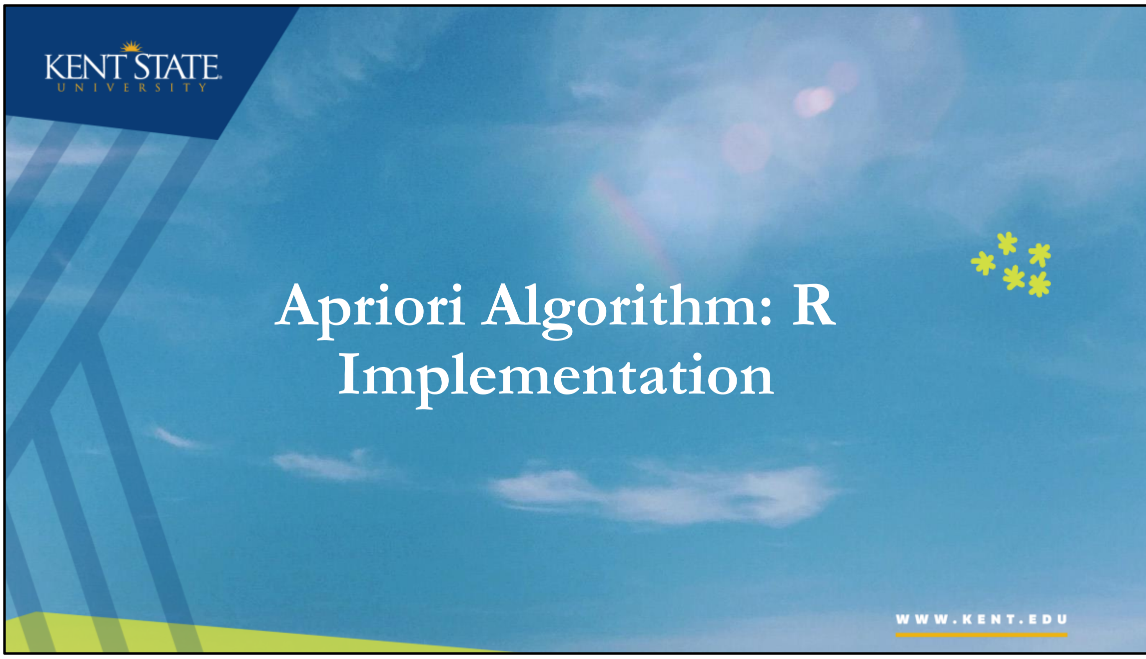


Let us now look at implementing the apriori algorithm in R

Apriori Algorithm: R Implementation

- Implementation of the Apriori algorithm is very straightforward in R
- We use the following two packages: arules and arulesViz
- You need to install these packages first using install.packages() function.

KENT STATE UNIVERSITY

WWW.KENT.EDU

We will use two packages: arules and arulesViz

Example: Explore Data I

```
library(arules)
library(arulesViz)

data("Groceries")
str(Groceries)

## Formal class 'transactions' [package "arules"] with 3 slots
##   ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
##   .. .. ..@ i       : int [1:43367] 13 60 69 78 14 29 98 24 15 29 ...
##   .. .. ..@ p       : int [1:9836] 0 4 7 8 12 16 21 22 27 28 ...
##   .. .. ..@ Dim     : int [1:2] 169 9835
##   .. .. ..@ Dimnames:List of 2
##   .. .. .. ..$ : NULL
##   .. .. .. ..$ : NULL
##   .. .. ..@ factors : list()
##   ..@ itemInfo   :'data.frame':  169 obs. of  3 variables:
##   .. ..$ labels: chr [1:169] "frankfurter" "sausage" "liver loaf" "ham" ...
##   .. ..$ level2: Factor w/ 55 levels "baby food","bags",..: 44 44 44 44 44 44 44 42
42 41 ...
##   .. ..$ level1: Factor w/ 10 levels "canned food",..: 6 6 6 6 6 6 6 6 6 6 ...
##   ..@ itemsetInfo:'data.frame':  0 obs. of  0 variables
```

The example uses the *Groceries* **dataset from the R** arules **package. The** *Groceries* **dataset is collected from 30 days of real-world point-of-sale transactions of a grocery store. The dataset contains 9,835 transactions, and the items are aggregated into 169 categories.**

The class of the dataset is transactions**, as defined by the** arules **package. The** transactions **class contains three slots:**
transactionInfo: A dataframe with vectors of the same length as the number of transactions
itemInfo: A dataframe to store item labels
Data: A binary incidence matrix that indicates which label appears in every transaction

For the *Groceries* **dataset, the** transactionInfo **is not being used.**

Example: Explore Data II

```
head(Groceries@itemInfo,n=12) #show top 12 rows
```

```
##                labels  level2           level1
## 1         frankfurter sausage meat and sausage
## 2             sausage sausage meat and sausage
## 3          liver loaf sausage meat and sausage
## 4                 ham sausage meat and sausage
## 5                meat sausage meat and sausage
## 6   finished products sausage meat and sausage
## 7     organic sausage sausage meat and sausage
## 8             chicken poultry meat and sausage
## 9              turkey poultry meat and sausage
## 10               pork    pork meat and sausage
## 11               beef    beef meat and sausage
## 12     hamburger meat    beef meat and sausage
```

WWW.KENT.EDU

The above command **displays only the first 12 grocery labels. Each grocery label is mapped to two levels of categories—**level2 **and** level1**—where** level1 **is a superset of** level2**. For example, grocery label** sausage **belongs to the** sausage **category in** level2**, and it is part of the** meat and sausage **category in** level1**.**

Example: Apply Apriori Alogorithm

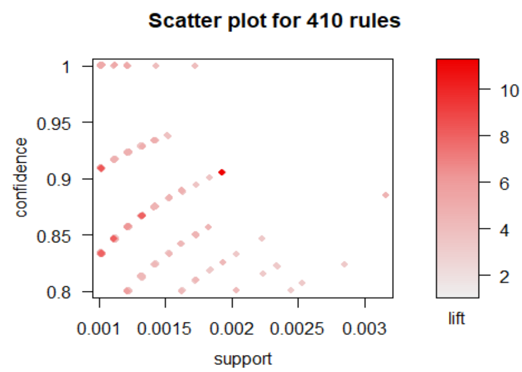
```
rules <- apriori(Groceries,parameter = list(supp = 0.001, conf = 0.80))

## Apriori
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
## Absolute minimum support count: 9
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [157 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.01s].
## writing ... [410 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

WWW.KENT.EDU

The apriori() **function from the** arule **package implements the Apriori algorithm to create frequent itemsets. Note the values of minimum support and confidence.**

Example: Show Rules’ Strength

```
plot(rules)
```



Enter plot(rules) **to display the scatterplot of the 410 rules, where the horizontal axis is the support, the vertical axis is the confidence, and the shading is the lift. The scatterplot shows that, of the 410 rules generated from the** *Groceries* **dataset, the highest lift occurs at a mid support and a mid confidence.**

Example: Show Rules

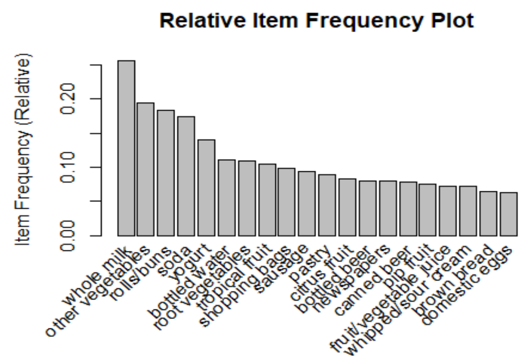
```
inspect(rules[1:10])

##      lhs                        rhs                   support confidence      lift count
## [1]  {liquor,
##       red/blush wine}        => {bottled beer}      0.001931876  0.9047619 11.235269    19
## [2]  {curd,
##       cereals}               => {whole milk}        0.001016777  0.9090909  3.557863    10
## [3]  {yogurt,
##       cereals}               => {whole milk}        0.001728521  0.8095238  3.168192    17
## [4]  {butter,
##       jam}                   => {whole milk}        0.001016777  0.8333333  3.261374    10
## [5]  {soups,
##       bottled beer}          => {whole milk}        0.001118454  0.9166667  3.587512    11
## [6]  {napkins,
##       house keeping products} => {whole milk}       0.001321810  0.8125000  3.179840    13
## [7]  {whipped/sour cream,
##       house keeping products} => {whole milk}       0.001220132  0.9230769  3.612599    12
## [8]  {pastry,
##       sweet spreads}         => {whole milk}        0.001016777  0.9090909  3.557863    10
## [9]  {turkey,
##       curd}                  => {other vegetables} 0.001220132  0.8000000  4.134524    12
## [10] {rice,
##       sugar}                 => {whole milk}        0.001220132  1.0000000  3.913649    12
```

WWW.KENT.EDU

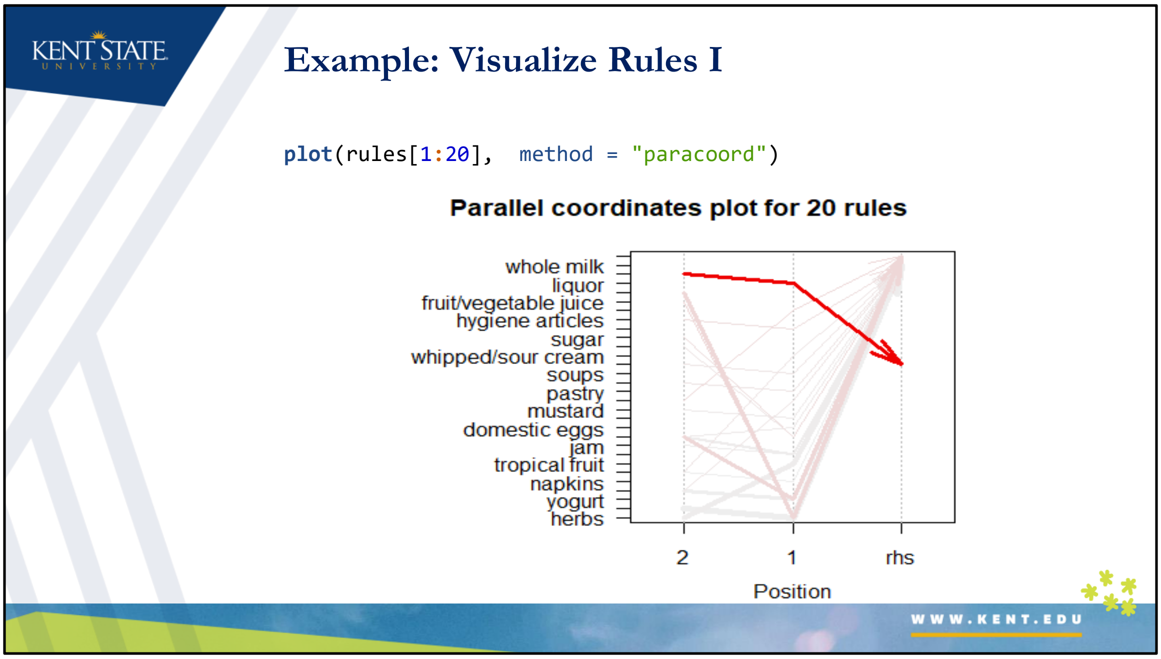
We can inspect the top ten rules that match our support and confidence level. The Lift values are also outputted. The antecedent itemsets are the LHS, while the consequent itemsets are the RHS.

Example: Show Frequent Items

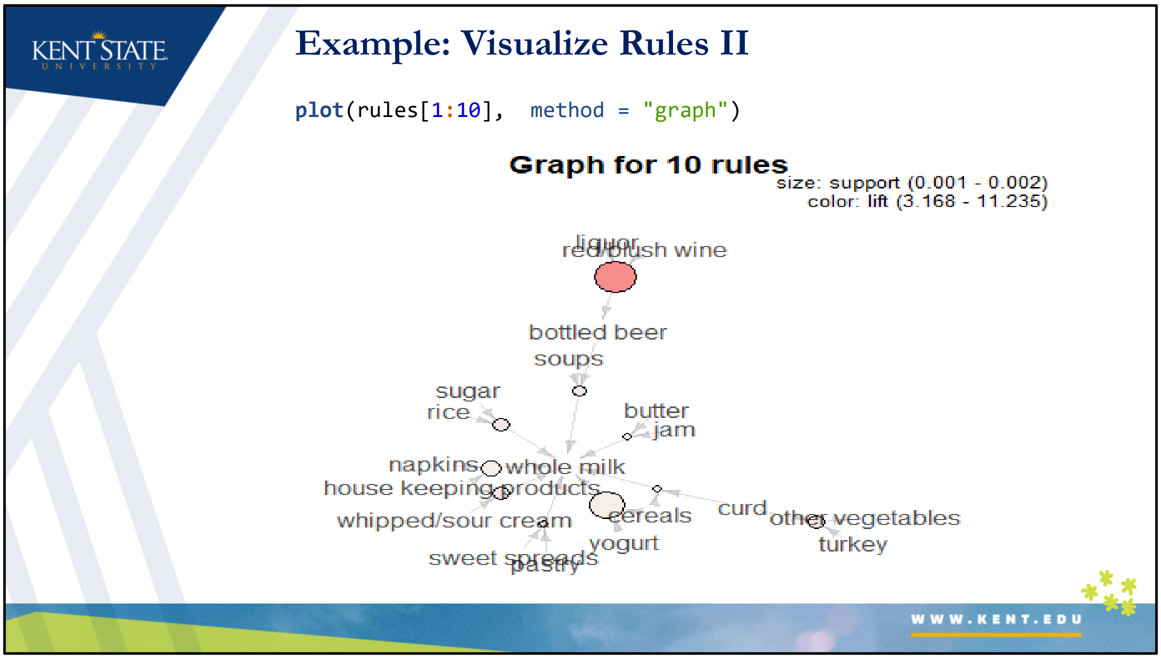
```
itemFrequencyPlot(Groceries,topN=20,main='Relative Item Frequency
Plot',type="relative",ylab="Item Frequency (Relative)")
```



A relative frequency of items can be output from the top 20 rules.



We can also use a parallel coordinate plot to visualize our rules.



The following code provides a visualization of the top 10 rules with the highest lift. The plot is shown above. In the graph, the arrow always points from an item on the LHS to an item on the RHS. For example, the arrows that connect liquor and red blush wine to bottled bear suggest rule {liquor, red blush wine}→{bottled bear}**. The legend on the top right of the graph shows that the size of a circle indicates the support of the rules ranging from 0.001 to 0.002. The color (or shade) represents the lift, which ranges from 3.168 to 11.235. The rule with the highest lift is the one we just identified.**

This concludes our module on Association Rules.