Instacart Analysis

From: RStudio Markdown  
Purpose: assignment for *R for Data Science*Author: yours truly =]

### Recently, Instacart open sourced this data - see their blog post on 3 Million Instacart Orders.

### Instacart is challenging the Kaggle community to use this anonymized data on customer orders over time to predict which previously purchased products will be in a user’s next order.

Competition link: <https://www.kaggle.com/c/instacart-market-basket-analysis/overview>

### Here, I will try to understand what items are linked together in same orders.

### I will use the arules package to do some light association rule mining.

Load libraries

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

require(arules)

## Loading required package: arules

## Loading required package: Matrix

##   
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':  
##   
## recode

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

library(arulesViz)

## Loading required package: grid

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

library(RColorBrewer)

#### read products and merge with orders from trainset

order\_products <- read.csv("data/order\_products\_\_train.csv")  
  
products <- read.csv("data/products.csv")  
  
orderItems <- left\_join(order\_products[,c(1,2)], products[,c(1,2)], by="product\_id")  
orderItems <- orderItems[,-2]

#### convert orderItems into a transaction object.

# Using read.transactions to transform your data.frame  
# code source: http://www.learnbymarketing.com/1043/working-with-arules-transactions-and-read-transactions/  
   
# Write our data.frame to a csv  
write.csv(orderItems, "data/tall\_orders.csv", quote=F, row.names=F)  
   
# Read that csv back in  
order\_trans <- read.transactions("data/tall\_orders.csv",  
 format = "single",  
 sep = ",", quote="",  
 header=T, cols=c("order\_id","product\_name"),  
 rm.duplicates = T  
)

#### Inspect order\_trans data

inspect(order\_trans[1])

## items transactionID  
## [1] {Bag of Organic Bananas,   
## Bulgarian Yogurt,   
## Cucumber Kirby,   
## Lightly Smoked Sardines in Olive Oil,   
## Organic 4% Milk Fat Whole Milk Cottage Cheese,   
## Organic Celery Hearts,   
## Organic Hass Avocado,   
## Organic Whole String Cheese} 1

summary(order\_trans)

## transactions as itemMatrix in sparse format with  
## 131209 rows (elements/itemsets/transactions) and  
## 37980 columns (items) and a density of 0.000277778   
##   
## most frequent items:  
## Banana Bag of Organic Bananas Organic Strawberries   
## 18781 15480 10894   
## Organic Baby Spinach Large Lemon (Other)   
## 9784 8135 1321182   
##   
## element (itemset/transaction) length distribution:  
## sizes  
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16   
## 6848 7368 8032 8225 8895 8712 8544 7984 7211 6553 6042 5376 4849 4389 3831 3520   
## 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32   
## 3113 2721 2464 2102 1854 1686 1457 1294 1082 984 860 675 635 556 441 402   
## 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48   
## 344 319 279 211 191 178 140 98 90 88 76 81 63 49 49 30   
## 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64   
## 27 29 24 25 17 14 12 10 6 5 4 8 3 3 6 3   
## 65 66 67 68 70 72 74 75 76 77 80   
## 3 2 2 1 4 2 2 1 2 1 2   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.00 5.00 9.00 10.55 14.00 80.00   
##   
## includes extended item information - examples:  
## labels  
## 1 \\"Constant Comment\\" Black Tea  
## 2 \\"Constant Comment\\" Decaffeinated Black Tea Blend  
## 3 \\"Darn Good\\" Chili Mix  
##   
## includes extended transaction information - examples:  
## transactionID  
## 1 1  
## 2 100000  
## 3 1000008

#### Finding Frequent itemsets

frequentItems <- eclat (order\_trans ,parameter = list(supp = 0.07, maxlen = 15))

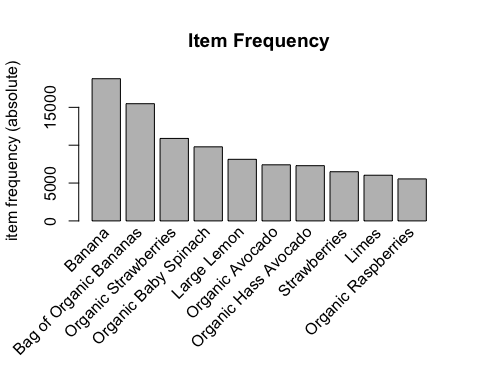
## Eclat  
##   
## parameter specification:  
## tidLists support minlen maxlen target ext  
## FALSE 0.07 1 15 frequent itemsets FALSE  
##   
## algorithmic control:  
## sparse sort verbose  
## 7 -2 TRUE  
##   
## Absolute minimum support count: 9184   
##   
## create itemset ...   
## set transactions ...[37980 item(s), 131209 transaction(s)] done [1.05s].  
## sorting and recoding items ... [4 item(s)] done [0.01s].  
## creating sparse bit matrix ... [4 row(s), 131209 column(s)] done [0.01s].  
## writing ... [4 set(s)] done [0.00s].  
## Creating S4 object ... done [0.00s].

inspect(frequentItems)

## items support count  
## [1] {Banana} 0.14313805 18781  
## [2] {Bag of Organic Bananas} 0.11797971 15480  
## [3] {Organic Strawberries} 0.08302784 10894  
## [4] {Organic Baby Spinach} 0.07456806 9784

#### Output the itemFrequencyPlot

itemFrequencyPlot(order\_trans, topN=10,  
type="absolute", main="Item Frequency")



#### Generating association rules using apriori package

rules <- apriori (order\_trans, parameter=list(supp=0.001, conf=0.25, minlen=2))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.25 0.1 1 none FALSE TRUE 5 0.001 2  
## 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: 131   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[37980 item(s), 131209 transaction(s)] done [1.11s].  
## sorting and recoding items ... [1873 item(s)] done [0.02s].  
## creating transaction tree ... done [0.11s].  
## checking subsets of size 1 2 3 4 done [0.09s].  
## writing ... [682 rule(s)] done [0.01s].  
## creating S4 object ... done [0.07s].

summary(rules)

## set of 682 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 2 3 4   
## 207 457 18   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2.000 2.000 3.000 2.723 3.000 4.000   
##   
## summary of quality measures:  
## support confidence lift count   
## Min. :0.001006 Min. :0.2500 Min. : 1.749 Min. : 132.0   
## 1st Qu.:0.001143 1st Qu.:0.2706 1st Qu.: 2.402 1st Qu.: 150.0   
## Median :0.001379 Median :0.3017 Median : 3.298 Median : 181.0   
## Mean :0.001950 Mean :0.3206 Mean : 5.532 Mean : 255.9   
## 3rd Qu.:0.001789 3rd Qu.:0.3525 3rd Qu.: 4.471 3rd Qu.: 234.8   
## Max. :0.023428 Max. :0.5984 Max. :80.298 Max. :3074.0   
##   
## mining info:  
## data ntransactions support confidence  
## order\_trans 131209 0.001 0.25

#### find high-confidence rules.

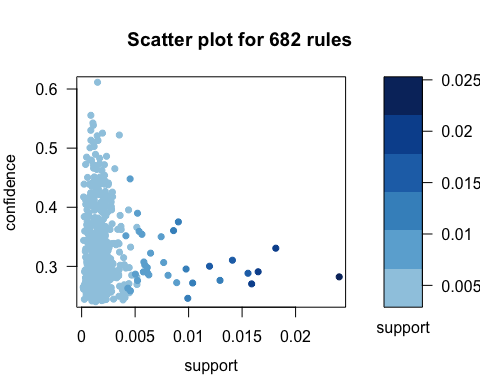
# show the support, lift and confidence for all rules  
rules\_conf <- sort (rules, by="confidence", decreasing=TRUE)

inspect(head(rules\_conf))

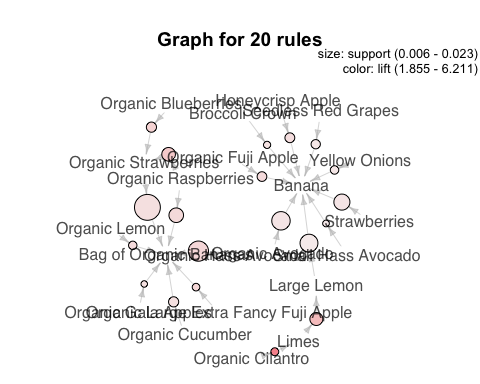
## lhs rhs support confidence lift count  
## [1] {Organic Hass Avocado,   
## Organic Raspberries,   
## Organic Strawberries} => {Bag of Organic Bananas} 0.001737686 0.5984252 5.072272 228  
## [2] {Organic Cucumber,   
## Organic Hass Avocado,   
## Organic Strawberries} => {Bag of Organic Bananas} 0.001067000 0.5468750 4.635331 140  
## [3] {Organic Hass Avocado,   
## Organic Kiwi} => {Bag of Organic Bananas} 0.001448071 0.5459770 4.627719 190  
## [4] {Organic Navel Orange,   
## Organic Raspberries} => {Bag of Organic Bananas} 0.001150836 0.5412186 4.587387 151  
## [5] {Strawberries,   
## Yellow Onions} => {Banana} 0.001143214 0.5357143 3.742641 150  
## [6] {Organic Hass Avocado,   
## Organic Whole String Cheese} => {Bag of Organic Bananas} 0.001158457 0.5314685 4.504745 152

#### visualize rules

plot(rules,control=list(jitter=2,col=rev(brewer.pal(9, "Blues")[4:9])),shading = 'support')



#Interactive graph as a html widget (using igraph layout)  
top.vegie.rules <- sort(rules,  
 by=c('support','lift'))[1:20]  
plot(top.vegie.rules, measure="support", method="graph", shading = "lift")



#### read departments, product, and merge with orders from trainset

departments <- read.csv("data/departments.csv")  
productDepartment <- left\_join(products[,c(1,4)], departments,   
 by = "department\_id")  
productDepartment <- productDepartment[,-2]

orderItemsDeprtmnt <- left\_join(order\_products[,c(1,2)],   
 productDepartment, by = "product\_id")  
orderItemsDeprtmnt <- orderItemsDeprtmnt[,-2]

# clean empty rows  
sum(is.na(orderItemsDeprtmnt))

## [1] 0

ItemDepClean<-na.omit(orderItemsDeprtmnt)

#### convert orderItemsDeprtmnt into a transaction object.

# Write our data.frame to a csv  
write.csv(ItemDepClean, "data/tall\_ordersDeprtmnt.csv", quote=F, row.names=F)  
   
# Read that csv back in  
order\_department <- read.transactions(  
 file = "data/tall\_ordersDeprtmnt.csv",  
 format = "single",  
 sep = ",", quote="",  
 header=T, cols=c("order\_id","department"),  
 rm.duplicates = T  
)

#### Inspect order\_department data

inspect(order\_department[1])

## items transactionID  
## [1] {canned goods,dairy eggs,produce} 1

summary(order\_department)

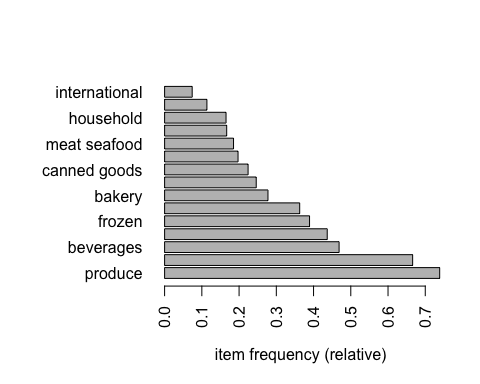
## transactions as itemMatrix in sparse format with  
## 131209 rows (elements/itemsets/transactions) and  
## 21 columns (items) and a density of 0.232414   
##   
## most frequent items:  
## produce dairy eggs beverages snacks frozen (Other)   
## 96927 87400 61482 57302 51071 286209   
##   
## element (itemset/transaction) length distribution:  
## sizes  
## 1 2 3 4 5 6 7 8 9 10 11 12 13   
## 11965 15208 17855 18930 17924 15409 11878 8640 5976 3649 2105 1022 437   
## 14 15 16   
## 163 39 9   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.000 3.000 5.000 4.881 7.000 16.000   
##   
## includes extended item information - examples:  
## labels  
## 1 alcohol  
## 2 babies  
## 3 bakery  
##   
## includes extended transaction information - examples:  
## transactionID  
## 1 1  
## 2 100000  
## 3 1000008

itemFreq <- itemFrequency(order\_department)  
head(itemFreq[order(-itemFreq)])

## produce dairy eggs beverages snacks frozen pantry   
## 0.7387222 0.6661128 0.4685807 0.4367231 0.3892340 0.3627724

#### Output the itemFrequencyPlot per Department

itemFrequencyPlot(order\_department, topN=15, horiz=T, )



## find association rules for departments in orders dataset

Drules <- apriori (order\_department, parameter = list(supp = 0.002, conf = 0.25,minlen=2))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.25 0.1 1 none FALSE TRUE 5 0.002 2  
## 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: 262   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[21 item(s), 131209 transaction(s)] done [0.07s].  
## sorting and recoding items ... [21 item(s)] done [0.01s].  
## creating transaction tree ... done [0.18s].  
## checking subsets of size 1 2 3 4 5 6 7 8 9 10

## Warning in apriori(order\_department, parameter = list(supp = 0.002, conf =  
## 0.25, : Mining stopped (maxlen reached). Only patterns up to a length of 10  
## returned!

## done [0.17s].  
## writing ... [93102 rule(s)] done [0.02s].  
## creating S4 object ... done [0.07s].

summary(Drules)

## set of 93102 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 2 3 4 5 6 7 8 9 10   
## 200 1820 7151 16715 25119 23680 13510 4307 600   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 2.000 5.000 6.000 6.309 7.000 10.000   
##   
## summary of quality measures:  
## support confidence lift count   
## Min. :0.002004 Min. :0.2500 Min. :0.705 Min. : 263.0   
## 1st Qu.:0.002706 1st Qu.:0.4832 1st Qu.:1.533 1st Qu.: 355.0   
## Median :0.004024 Median :0.6454 Median :1.907 Median : 528.0   
## Mean :0.007453 Mean :0.6436 Mean :1.924 Mean : 977.9   
## 3rd Qu.:0.007339 3rd Qu.:0.7857 3rd Qu.:2.219 3rd Qu.: 963.0   
## Max. :0.543835 Max. :1.0000 Max. :3.576 Max. :71356.0   
##   
## mining info:  
## data ntransactions support confidence  
## order\_department 131209 0.002 0.25

## What factors influenced purchase things from “produce” department

DepRules1 <- apriori (data=order\_department, parameter=list (supp=0.001,conf = 0.08),  
appearance = list (default="lhs",rhs="produce"),control = list(verbose=F))  
  
rules\_conf2 <- sort (DepRules1, by="confidence", decreasing=TRUE)  
inspect(head(rules\_conf2))

## lhs rhs support confidence lift count  
## [1] {bulk,   
## meat seafood,   
## snacks} => {produce} 0.001249914 1.0000000 1.353689 164  
## [2] {bulk,   
## dairy eggs,   
## meat seafood,   
## snacks} => {produce} 0.001135593 1.0000000 1.353689 149  
## [3] {beverages,   
## canned goods,   
## deli,   
## dry goods pasta,   
## international,   
## meat seafood,   
## pantry,   
## snacks} => {produce} 0.001059379 1.0000000 1.353689 139  
## [4] {beverages,   
## canned goods,   
## dairy eggs,   
## deli,   
## dry goods pasta,   
## international,   
## meat seafood,   
## pantry,   
## snacks} => {produce} 0.001051757 1.0000000 1.353689 138  
## [5] {beverages,   
## deli,   
## dry goods pasta,   
## international,   
## meat seafood,   
## pantry,   
## snacks} => {produce} 0.001577636 0.9951923 1.347181 207  
## [6] {beverages,   
## dairy eggs,   
## deli,   
## dry goods pasta,   
## international,   
## meat seafood,   
## pantry,   
## snacks} => {produce} 0.001547150 0.9950980 1.347053 203

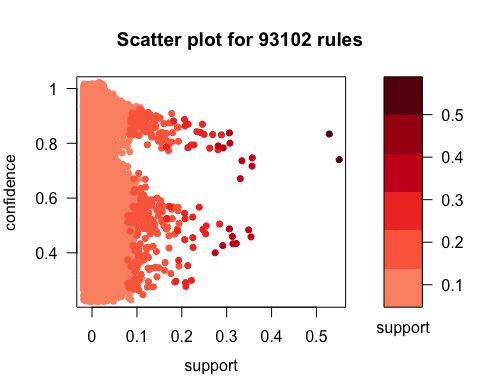
## What departments were purchased along with department “produce”

DepRules2 <- apriori (data=order\_department, parameter=list (supp=0.001,conf = 0.08, minlen=2), appearance = list (default="rhs",lhs="produce"), control = list (verbose=F))  
# 'high-confidence' rules.  
rules\_conf3 <- sort (DepRules2,by="confidence", decreasing=TRUE)  
inspect(head(rules\_conf3))

## lhs rhs support confidence lift count  
## [1] {produce} => {dairy eggs} 0.5438346 0.7361829 1.1051925 71356  
## [2] {produce} => {beverages} 0.3421488 0.4631630 0.9884381 44893  
## [3] {produce} => {snacks} 0.3383076 0.4579632 1.0486352 44389  
## [4] {produce} => {frozen} 0.3144906 0.4257225 1.0937443 41264  
## [5] {produce} => {pantry} 0.2978073 0.4031384 1.1112711 39075  
## [6] {produce} => {bakery} 0.2323240 0.3144944 1.1328931 30483

## visualization

plot(Drules,control=list(jitter=2,col=rev(brewer.pal(9, "Reds")[4:9])),shading = 'support')



top.vegie.rules <- sort(Drules,  
 by=c('support','lift'))[1:21]  
plot(top.vegie.rules, measure="support", method="graph", shading = "lift")

