Final\_Presentation1

SAMIKSHA VENUGOPAL

12/8/2021

### Install required packages   
library(readr)  
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

library(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)  
library(plyr)

## ------------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.  
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:  
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------------

##   
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

library(ggplot2)

## Cross marketing

## The technique used to solve and infer the issue using association rules.

## Loading the file into R

## The data set contain the contains purchase data from April 2020 to November 2020 from a large home appliances and electronics online store.

## Reading the file as Kz

### Reading the file  
getwd()

## [1] "/Users/samikshachalla/Desktop"

setwd("/users/Samikshachalla/downloads")  
kz<-read.csv("kz.csv")  
head(kz)

## event\_time order\_id product\_id category\_id  
## 1 2020-04-24 11:50:39 UTC 2.294360e+18 1.515966e+18 2.268105e+18  
## 2 2020-04-24 11:50:39 UTC 2.294360e+18 1.515966e+18 2.268105e+18  
## 3 2020-04-24 14:37:43 UTC 2.294444e+18 2.273948e+18 2.268105e+18  
## 4 2020-04-24 14:37:43 UTC 2.294444e+18 2.273948e+18 2.268105e+18  
## 5 2020-04-24 19:16:21 UTC 2.294584e+18 2.273948e+18 2.268105e+18  
## 6 2020-04-26 08:45:57 UTC 2.295717e+18 1.515966e+18 2.268105e+18  
## category\_code brand price user\_id  
## 1 electronics.tablet samsung 162.01 1.515916e+18  
## 2 electronics.tablet samsung 162.01 1.515916e+18  
## 3 electronics.audio.headphone huawei 77.52 1.515916e+18  
## 4 electronics.audio.headphone huawei 77.52 1.515916e+18  
## 5 karcher 217.57 1.515916e+18  
## 6 furniture.kitchen.table maestro 39.33 1.515916e+18

` ## Data Prepartion for prediction ## set the seed for the radom number generator for predictability

## Re-viewing the dataset  
  
set.seed(111)  
str(kz)

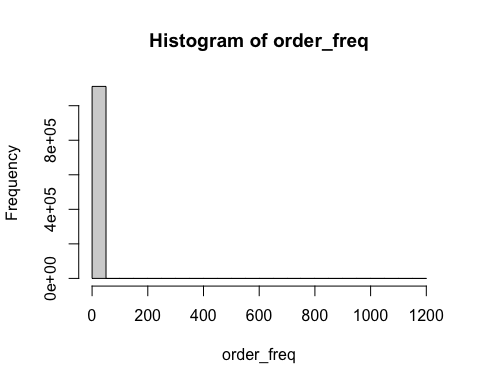
## 'data.frame': 2633521 obs. of 8 variables:  
## $ event\_time : chr "2020-04-24 11:50:39 UTC" "2020-04-24 11:50:39 UTC" "2020-04-24 14:37:43 UTC" "2020-04-24 14:37:43 UTC" ...  
## $ order\_id : num 2.29e+18 2.29e+18 2.29e+18 2.29e+18 2.29e+18 ...  
## $ product\_id : num 1.52e+18 1.52e+18 2.27e+18 2.27e+18 2.27e+18 ...  
## $ category\_id : num 2.27e+18 2.27e+18 2.27e+18 2.27e+18 2.27e+18 ...  
## $ category\_code: chr "electronics.tablet" "electronics.tablet" "electronics.audio.headphone" "electronics.audio.headphone" ...  
## $ brand : chr "samsung" "samsung" "huawei" "huawei" ...  
## $ price : num 162 162 77.5 77.5 217.6 ...  
## $ user\_id : num 1.52e+18 1.52e+18 1.52e+18 1.52e+18 1.52e+18 ...

summary(kz)

## event\_time order\_id product\_id category\_id   
## Length:2633521 Min. :2.294e+18 Min. :1.516e+18 Min. :2.268e+18   
## Class :character 1st Qu.:2.349e+18 1st Qu.:1.516e+18 1st Qu.:2.268e+18   
## Mode :character Median :2.353e+18 Median :1.516e+18 Median :2.268e+18   
## Mean :2.362e+18 Mean :1.674e+18 Mean :2.274e+18   
## 3rd Qu.:2.383e+18 3rd Qu.:1.516e+18 3rd Qu.:2.268e+18   
## Max. :2.388e+18 Max. :2.388e+18 Max. :2.374e+18   
## NA's :431954   
## category\_code brand price user\_id   
## Length:2633521 Length:2633521 Min. : 0.0 Min. :1.516e+18   
## Class :character Class :character 1st Qu.: 14.6 1st Qu.:1.516e+18   
## Mode :character Mode :character Median : 55.5 Median :1.516e+18   
## Mean : 154.1 Mean :1.516e+18   
## 3rd Qu.: 196.7 3rd Qu.:1.516e+18   
## Max. :50925.9 Max. :1.516e+18   
## NA's :431954 NA's :2069352

## Considering order ID

## Clearning the data   
  
## Removing single items order  
  
order\_freq<-table(kz$order\_id)  
hist(order\_freq)

 ## Items brought in each order is in range of r range ( order\_fre)

order\_freq<-order\_freq[order\_freq>1]  
kz<-subset(kz,kz$order\_id%in% names(order\_freq),c("order\_id","category\_code"))

## Removing duplicate items from order list

rm(order\_freq)  
kz<-unique(kz)

## Convert the data frame into basket format

if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){  
 detach(package:dplyr, unload=TRUE)  
 }  
  
 df\_itemList <- ddply(kz,c("order\_id"),  
 function(kz)paste(kz$category\_code,  
 collapse = ","))  
  
rm(kz)  
df\_itemList$order\_id <- NULL

## Rename column header for ease of use

colnames(df\_itemList) <- c("itemList")  
write.csv(df\_itemList,"ItemList.csv",row.names = TRUE)  
rm(df\_itemList)

# Compute transactions for the product names from the order baskets

# Find the association rules  
  
txn = read.transactions(file="ItemList.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1);  
txn@itemInfo$labels <- gsub("\"","",txn@itemInfo$labels)  
basket\_rules <- apriori(txn,parameter = list(sup = 0.001, conf = 0.001,target="rules"))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.001 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: 448   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[94187 item(s), 448709 transaction(s)] done [0.18s].  
## sorting and recoding items ... [104 item(s)] done [0.01s].  
## creating transaction tree ... done [0.05s].  
## checking subsets of size 1 done [0.00s].  
## writing ... [104 rule(s)] done [0.00s].  
## creating S4 object ... done [0.05s].

basket\_rules

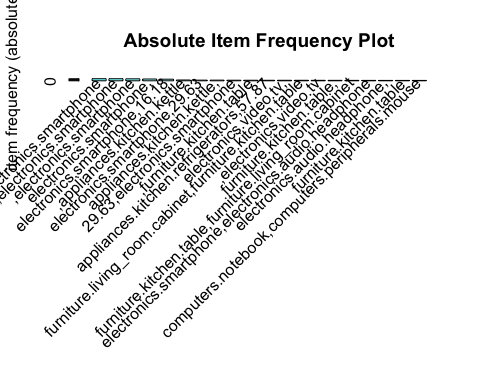
## set of 104 rules

summary(basket\_rules)

## set of 104 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 1   
## 104   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1 1 1 1 1 1   
##   
## summary of quality measures:  
## support confidence coverage lift count   
## Min. :0.001001 Min. :0.001001 Min. :1 Min. :1 Min. : 449.0   
## 1st Qu.:0.001523 1st Qu.:0.001523 1st Qu.:1 1st Qu.:1 1st Qu.: 683.5   
## Median :0.002147 Median :0.002147 Median :1 Median :1 Median : 963.5   
## Mean :0.003041 Mean :0.003041 Mean :1 Mean :1 Mean :1364.5   
## 3rd Qu.:0.002966 3rd Qu.:0.002966 3rd Qu.:1 3rd Qu.:1 3rd Qu.:1331.0   
## Max. :0.018368 Max. :0.018368 Max. :1 Max. :1 Max. :8242.0   
##   
## mining info:  
## data ntransactions support confidence  
## txn 448709 0.001 0.001  
## call  
## apriori(data = txn, parameter = list(sup = 0.001, conf = 0.001, target = "rules"))

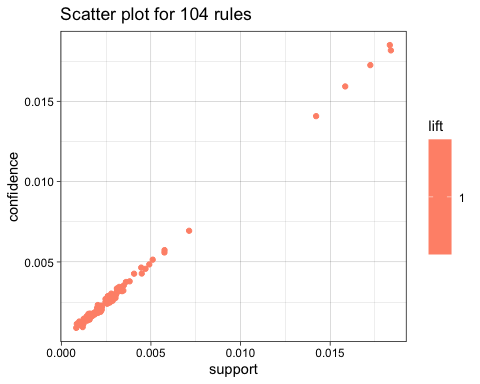
## plot

## Plot basket rules  
  
itemFrequencyPlot(txn,topN=20,type="absolute",col=cm.colors(20), main="Absolute Item Frequency Plot")



plot(basket\_rules)

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



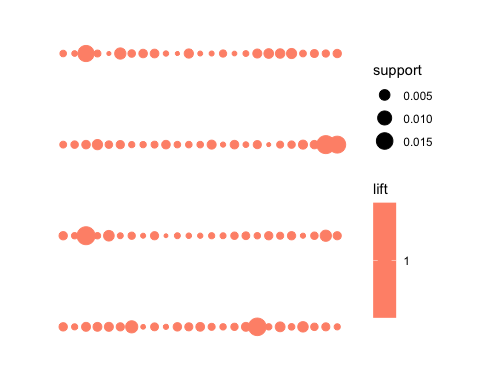
plot(basket\_rules, method="graph", control=list(type="items"))

## Warning: Unknown control parameters: type

## Available control parameters (with default values):  
## layout = stress  
## circular = FALSE  
## ggraphdots = NULL  
## edges = <environment>  
## nodes = <environment>  
## nodetext = <environment>  
## colors = c("#EE0000FF", "#EEEEEEFF")  
## engine = ggplot2  
## max = 100  
## verbose = FALSE

## Warning: Too many rules supplied. Only plotting the best 100 using  
## 'lift' (change control parameter max if needed).

## Warning: ggrepel: 100 unlabeled data points (too many overlaps). Consider  
## increasing max.overlaps

 ##Find the association rules with sup = 0.005

# Find the association rules with sup = 0.005  
txn = read.transactions(file="ItemList.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1);  
txn@itemInfo$labels <- gsub("\"","",txn@itemInfo$labels)  
basket\_rules <- apriori(txn,parameter = list(sup = 0.005, conf = 0.01,target="rules"))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.01 0.1 1 none FALSE TRUE 5 0.005 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: 2243   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[94187 item(s), 448709 transaction(s)] done [0.31s].  
## sorting and recoding items ... [9 item(s)] done [0.01s].  
## creating transaction tree ... done [0.02s].  
## checking subsets of size 1 done [0.00s].  
## writing ... [5 rule(s)] done [0.00s].  
## creating S4 object ... done [0.05s].

basket\_rules

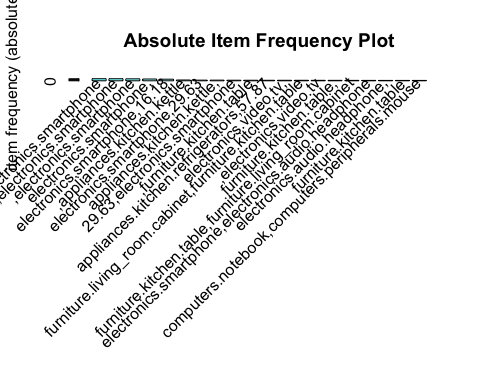
## set of 5 rules

summary(basket\_rules)

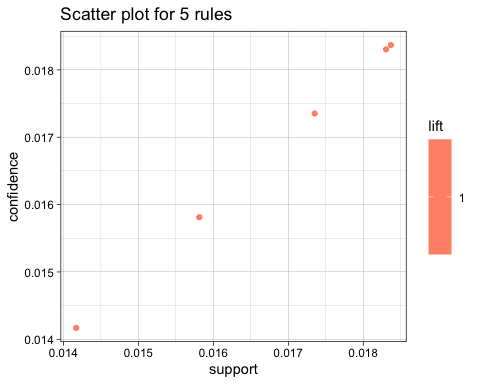
## set of 5 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 1   
## 5   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1 1 1 1 1 1   
##   
## summary of quality measures:  
## support confidence coverage lift count   
## Min. :0.01417 Min. :0.01417 Min. :1 Min. :1 Min. :6358   
## 1st Qu.:0.01581 1st Qu.:0.01581 1st Qu.:1 1st Qu.:1 1st Qu.:7095   
## Median :0.01735 Median :0.01735 Median :1 Median :1 Median :7786   
## Mean :0.01680 Mean :0.01680 Mean :1 Mean :1 Mean :7539   
## 3rd Qu.:0.01830 3rd Qu.:0.01830 3rd Qu.:1 3rd Qu.:1 3rd Qu.:8213   
## Max. :0.01837 Max. :0.01837 Max. :1 Max. :1 Max. :8242   
##   
## mining info:  
## data ntransactions support confidence  
## txn 448709 0.005 0.01  
## call  
## apriori(data = txn, parameter = list(sup = 0.005, conf = 0.01, target = "rules"))

##Plot

## Plot basket rules  
  
  
itemFrequencyPlot(txn,topN=20,type="absolute",col=cm.colors(20), main="Absolute Item Frequency Plot")



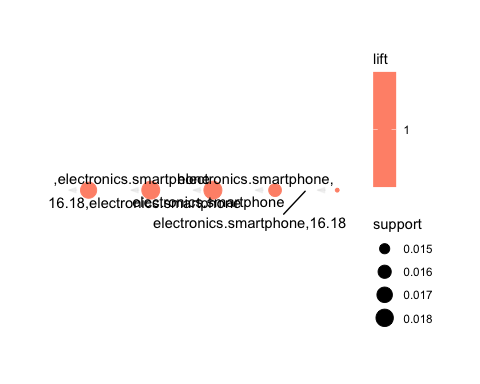
plot(basket\_rules)



plot(basket\_rules, method="graph", control=list(type="items"))

## Warning: Unknown control parameters: type

## Available control parameters (with default values):  
## layout = stress  
## circular = FALSE  
## ggraphdots = NULL  
## edges = <environment>  
## nodes = <environment>  
## nodetext = <environment>  
## colors = c("#EE0000FF", "#EEEEEEFF")  
## engine = ggplot2  
## max = 100  
## verbose = FALSE

 ## Camparing the results with between sup = 0.001 and sup = 0.005

## Find the association rule with conf= 0.001  
txn = read.transactions(file="ItemList.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1);  
txn@itemInfo$labels <- gsub("\"","",txn@itemInfo$labels)  
basket\_rules <- apriori(txn,parameter = list(sup = 0.001, conf = 0.001,target="rules"))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.001 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: 448   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[94187 item(s), 448709 transaction(s)] done [0.19s].  
## sorting and recoding items ... [104 item(s)] done [0.01s].  
## creating transaction tree ... done [0.05s].  
## checking subsets of size 1 done [0.00s].  
## writing ... [104 rule(s)] done [0.00s].  
## creating S4 object ... done [0.04s].

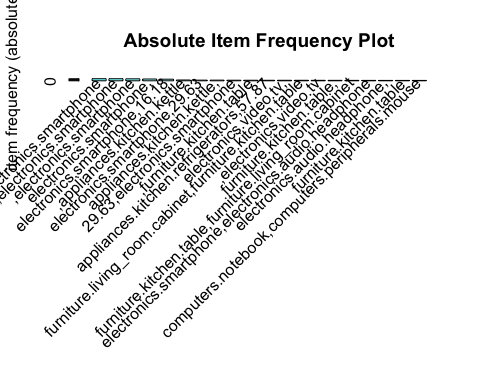
basket\_rules

## set of 104 rules

summary(basket\_rules)

## set of 104 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 1   
## 104   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1 1 1 1 1 1   
##   
## summary of quality measures:  
## support confidence coverage lift count   
## Min. :0.001001 Min. :0.001001 Min. :1 Min. :1 Min. : 449.0   
## 1st Qu.:0.001523 1st Qu.:0.001523 1st Qu.:1 1st Qu.:1 1st Qu.: 683.5   
## Median :0.002147 Median :0.002147 Median :1 Median :1 Median : 963.5   
## Mean :0.003041 Mean :0.003041 Mean :1 Mean :1 Mean :1364.5   
## 3rd Qu.:0.002966 3rd Qu.:0.002966 3rd Qu.:1 3rd Qu.:1 3rd Qu.:1331.0   
## Max. :0.018368 Max. :0.018368 Max. :1 Max. :1 Max. :8242.0   
##   
## mining info:  
## data ntransactions support confidence  
## txn 448709 0.001 0.001  
## call  
## apriori(data = txn, parameter = list(sup = 0.001, conf = 0.001, target = "rules"))

itemFrequencyPlot(txn,topN=20,type="absolute",col=cm.colors(20), main="Absolute Item Frequency Plot")



plot(basket\_rules)

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



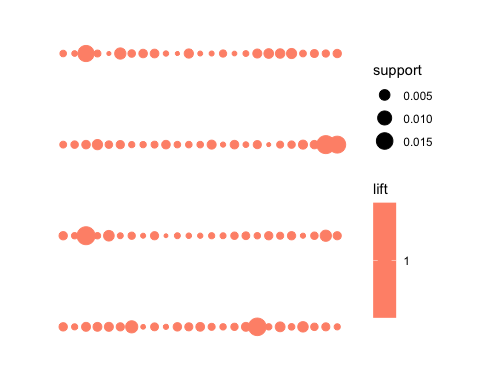
plot(basket\_rules, method="graph", control=list(type="items"))

## Warning: Unknown control parameters: type

## Available control parameters (with default values):  
## layout = stress  
## circular = FALSE  
## ggraphdots = NULL  
## edges = <environment>  
## nodes = <environment>  
## nodetext = <environment>  
## colors = c("#EE0000FF", "#EEEEEEFF")  
## engine = ggplot2  
## max = 100  
## verbose = FALSE

## Warning: Too many rules supplied. Only plotting the best 100 using  
## 'lift' (change control parameter max if needed).

## Warning: ggrepel: 100 unlabeled data points (too many overlaps). Consider  
## increasing max.overlaps

 ## Inspect

## Association rules for departments in orders dataset  
basket\_rules <- apriori(txn, parameter = list(supp = 0.001, conf = 0.001))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.001 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: 448   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[94187 item(s), 448709 transaction(s)] done [0.19s].  
## sorting and recoding items ... [104 item(s)] done [0.01s].  
## creating transaction tree ... done [0.05s].  
## checking subsets of size 1 done [0.00s].  
## writing ... [104 rule(s)] done [0.00s].  
## creating S4 object ... done [0.05s].

summary(basket\_rules)

## set of 104 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 1   
## 104   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1 1 1 1 1 1   
##   
## summary of quality measures:  
## support confidence coverage lift count   
## Min. :0.001001 Min. :0.001001 Min. :1 Min. :1 Min. : 449.0   
## 1st Qu.:0.001523 1st Qu.:0.001523 1st Qu.:1 1st Qu.:1 1st Qu.: 683.5   
## Median :0.002147 Median :0.002147 Median :1 Median :1 Median : 963.5   
## Mean :0.003041 Mean :0.003041 Mean :1 Mean :1 Mean :1364.5   
## 3rd Qu.:0.002966 3rd Qu.:0.002966 3rd Qu.:1 3rd Qu.:1 3rd Qu.:1331.0   
## Max. :0.018368 Max. :0.018368 Max. :1 Max. :1 Max. :8242.0   
##   
## mining info:  
## data ntransactions support confidence  
## txn 448709 0.001 0.001  
## call  
## apriori(data = txn, parameter = list(supp = 0.001, conf = 0.001))

rules1d\_order\_conf <- sort(basket\_rules, by="conf", decreasing = TRUE)  
inspect(head(rules1d\_order\_conf, 10))

## lhs rhs support confidence coverage  
## [1] {} => {electronics.smartphone} 0.018368252 0.018368252 1   
## [2] {} => {16.18,electronics.smartphone} 0.018303622 0.018303622 1   
## [3] {} => {,electronics.smartphone} 0.017352003 0.017352003 1   
## [4] {} => {electronics.smartphone,} 0.015812030 0.015812030 1   
## [5] {} => {electronics.smartphone,16.18} 0.014169540 0.014169540 1   
## [6] {} => {,appliances.kitchen.kettle} 0.007006768 0.007006768 1   
## [7] {} => {electronics.smartphone,29.63} 0.005754286 0.005754286 1   
## [8] {} => {appliances.kitchen.kettle,} 0.005725314 0.005725314 1   
## [9] {} => {29.63,electronics.smartphone} 0.005003243 0.005003243 1   
## [10] {} => {,furniture.kitchen.table} 0.004824953 0.004824953 1   
## lift count  
## [1] 1 8242   
## [2] 1 8213   
## [3] 1 7786   
## [4] 1 7095   
## [5] 1 6358   
## [6] 1 3144   
## [7] 1 2582   
## [8] 1 2569   
## [9] 1 2245   
## [10] 1 2165

rules1d\_order\_lift <- sort(basket\_rules, by="lift", decreasing = TRUE)  
inspect(head(rules1d\_order\_lift, 10))

## lhs rhs support confidence coverage lift count  
## [1] {} => {electronics.video.tv,electronics.smartphone} 0.001065278 0.001065278 1 1 478  
## [2] {} => {electronics.smartphone,electronics.video.tv} 0.001092022 0.001092022 1 1 490  
## [3] {} => {57.87,} 0.002028040 0.002028040 1 1 910  
## [4] {} => {,construction.tools.screw} 0.002032498 0.002032498 1 1 912  
## [5] {} => {electronics.audio.headphone} 0.002106042 0.002106042 1 1 945  
## [6] {} => {furniture.kitchen.table,} 0.003937964 0.003937964 1 1 1767  
## [7] {} => {appliances.kitchen.refrigerators,57.87} 0.004544148 0.004544148 1 1 2039  
## [8] {} => {,electronics.smartphone} 0.017352003 0.017352003 1 1 7786  
## [9] {} => {57.87,appliances.kitchen.washer} 0.001000649 0.001000649 1 1 449  
## [10] {} => {computers.peripherals.mouse,computers.peripherals.keyboard} 0.001002877 0.001002877 1 1 450

basket\_rules2 <- apriori(txn, parameter = list(supp = 0.005, conf = 0.01))

## Apriori  
##   
## Parameter specification:  
## confidence minval smax arem aval originalSupport maxtime support minlen  
## 0.01 0.1 1 none FALSE TRUE 5 0.005 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: 2243   
##   
## set item appearances ...[0 item(s)] done [0.00s].  
## set transactions ...[94187 item(s), 448709 transaction(s)] done [0.21s].  
## sorting and recoding items ... [9 item(s)] done [0.01s].  
## creating transaction tree ... done [0.02s].  
## checking subsets of size 1 done [0.00s].  
## writing ... [5 rule(s)] done [0.00s].  
## creating S4 object ... done [0.05s].

summary(basket\_rules2)

## set of 5 rules  
##   
## rule length distribution (lhs + rhs):sizes  
## 1   
## 5   
##   
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1 1 1 1 1 1   
##   
## summary of quality measures:  
## support confidence coverage lift count   
## Min. :0.01417 Min. :0.01417 Min. :1 Min. :1 Min. :6358   
## 1st Qu.:0.01581 1st Qu.:0.01581 1st Qu.:1 1st Qu.:1 1st Qu.:7095   
## Median :0.01735 Median :0.01735 Median :1 Median :1 Median :7786   
## Mean :0.01680 Mean :0.01680 Mean :1 Mean :1 Mean :7539   
## 3rd Qu.:0.01830 3rd Qu.:0.01830 3rd Qu.:1 3rd Qu.:1 3rd Qu.:8213   
## Max. :0.01837 Max. :0.01837 Max. :1 Max. :1 Max. :8242   
##   
## mining info:  
## data ntransactions support confidence  
## txn 448709 0.005 0.01  
## call  
## apriori(data = txn, parameter = list(supp = 0.005, conf = 0.01))

rules2d\_order\_conf <- sort(basket\_rules2, by="conf", decreasing = TRUE)  
inspect(head(rules2d\_order\_conf, 10))

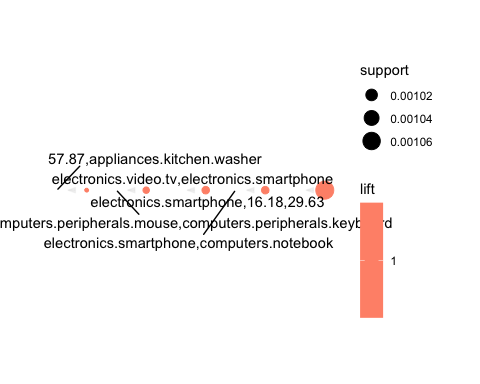
## lhs rhs support confidence coverage lift  
## [1] {} => {electronics.smartphone} 0.01836825 0.01836825 1 1   
## [2] {} => {16.18,electronics.smartphone} 0.01830362 0.01830362 1 1   
## [3] {} => {,electronics.smartphone} 0.01735200 0.01735200 1 1   
## [4] {} => {electronics.smartphone,} 0.01581203 0.01581203 1 1   
## [5] {} => {electronics.smartphone,16.18} 0.01416954 0.01416954 1 1   
## count  
## [1] 8242   
## [2] 8213   
## [3] 7786   
## [4] 7095   
## [5] 6358

rules2d\_order\_lift <- sort(basket\_rules2, by="lift", decreasing = TRUE)  
inspect(head(rules2d\_order\_lift, 10))

## lhs rhs support confidence coverage lift  
## [1] {} => {,electronics.smartphone} 0.01735200 0.01735200 1 1   
## [2] {} => {electronics.smartphone,16.18} 0.01416954 0.01416954 1 1   
## [3] {} => {electronics.smartphone,} 0.01581203 0.01581203 1 1   
## [4] {} => {16.18,electronics.smartphone} 0.01830362 0.01830362 1 1   
## [5] {} => {electronics.smartphone} 0.01836825 0.01836825 1 1   
## count  
## [1] 7786   
## [2] 6358   
## [3] 7095   
## [4] 8213   
## [5] 8242

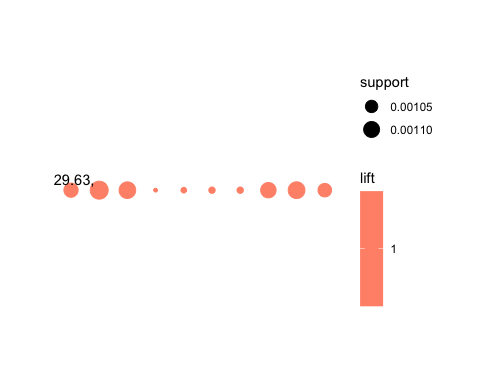
## Top 5

library(arulesViz)  
 top5rules <- (basket\_rules[1:5])  
 plot(top5rules, method = "graph")

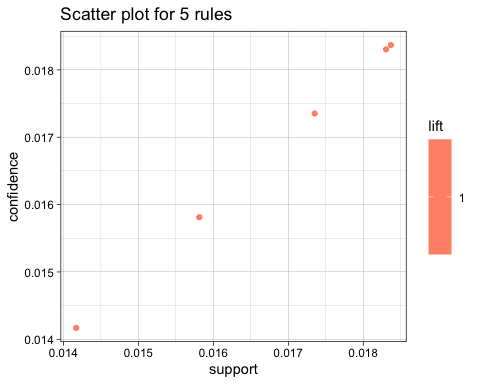
 ## Top 10

library(arulesViz)  
 top5rules <- (basket\_rules[1:10])  
 plot(top5rules, method = "graph")

## Warning: ggrepel: 9 unlabeled data points (too many overlaps). Consider  
## increasing max.overlaps



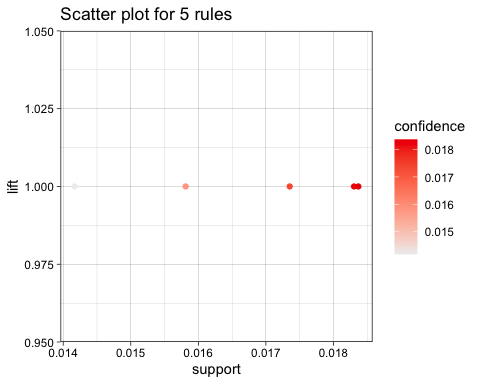
plot(basket\_rules2)



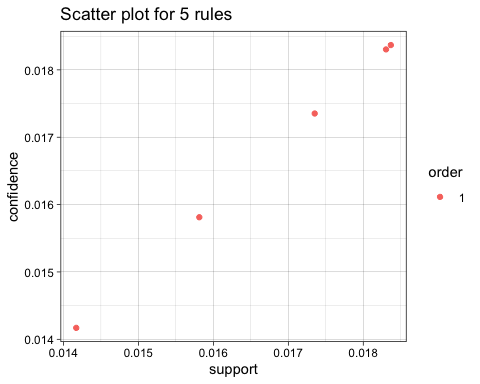
head(quality(basket\_rules2))

## support confidence coverage lift count  
## 1 0.01416954 0.01416954 1 1 6358  
## 2 0.01581203 0.01581203 1 1 7095  
## 3 0.01735200 0.01735200 1 1 7786  
## 4 0.01830362 0.01830362 1 1 8213  
## 5 0.01836825 0.01836825 1 1 8242

plot(basket\_rules2,measure = c("support","lift"),shading = "confidence")



plot(basket\_rules2,method = "two-key plot")

 ##Matrix based visulization

subrules <- basket\_rules2[quality(basket\_rules2)$confidence > 0.5]  
subrules

## set of 0 rules

## Double Decker plot

oneRule <- sample(basket\_rules,1)  
inspect(oneRule)

## lhs rhs support confidence coverage lift count  
## [1] {} => {furniture.kitchen.table,furniture.living\_room.cabinet} 0.003750761 0.003750761 1 1 1683

plot(oneRule,data = kz)

