Grocery Association Rule Code ->

Soln\_>

> #Association rule for data 1

> #install.packages("arulesViz")

> #install.packages("arules")

>

> library(arules)

> groceries<-read.transactions(file.choose(),format="basket")

Warning message:

In asMethod(object) : removing duplicated items in transactions

> inspect(groceries[1:10])

items

[1] {bread,margarine,ready,citrus,fruit,semi-finished,soups}

[2] {fruit,yogurt,coffee,tropical}

[3] {milk,whole}

[4] {,meat,cheese,fruit,yogurt,cream,pip,spreads}

[5] {bakery,life,milk,condensed,milk,long,other,product,vegetables,whole}

[6] {cleaner,milk,butter,yogurt,rice,abrasive,whole}

[7] {rolls/buns}

[8] {(appetizer),beer,liquor,other,vegetables,UHT-milk,rolls/buns,bottled}

[9] {plants,pot}

[10] {milk,cereals,whole}

> class(groceries)

[1] "transactions"

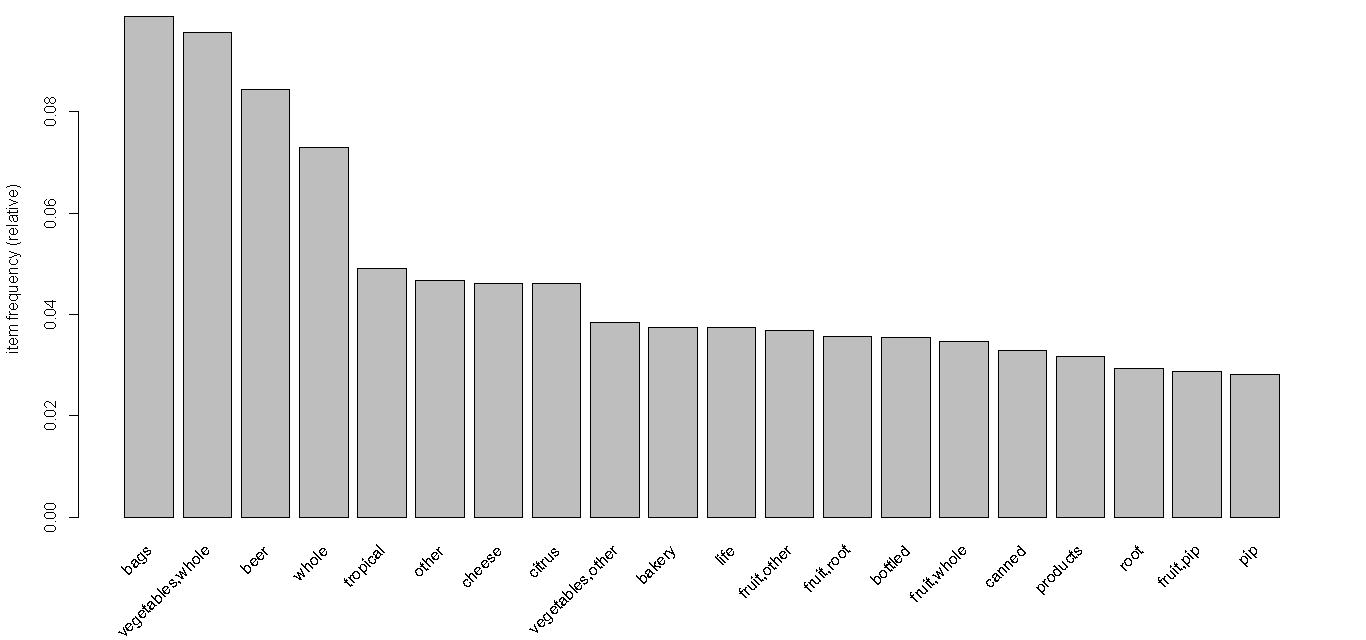
attr(,"package")

[1] "arules"

> # itemFrequencyPlot can be applicable only for transaction data

> # count of each item from all the transactions

> itemFrequencyPlot(groceries,topN=20)



> groceries\_rules<-apriori(groceries,parameter = list(support = 0.002,confidence = 0.05,minlen=3))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext

0.05 0.1 1 none FALSE TRUE 5 0.002 3 10 rules FALSE

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 ...[6928 item(s), 9835 transaction(s)] done [0.02s].

sorting and recoding items ... [257 item(s)] done [0.00s].

creating transaction tree ... done [0.00s].

checking subsets of size 1 2 3 4 done [0.00s].

writing ... [118 rule(s)] done [0.00s].

creating S4 object ... done [0.00s].

>

> library(arulesViz)

>

> groceries\_rules<-apriori(groceries,parameter = list(support = 0.006,confidence = 0.05,minlen=3))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext

0.05 0.1 1 none FALSE TRUE 5 0.006 3 10 rules FALSE

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 59

set item appearances ...[0 item(s)] done [0.00s].

set transactions ...[6928 item(s), 9835 transaction(s)] done [0.02s].

sorting and recoding items ... [77 item(s)] done [0.00s].

creating transaction tree ... done [0.00s].

checking subsets of size 1 2 3 done [0.00s].

writing ... [9 rule(s)] done [0.00s].

creating S4 object ... done [0.00s].

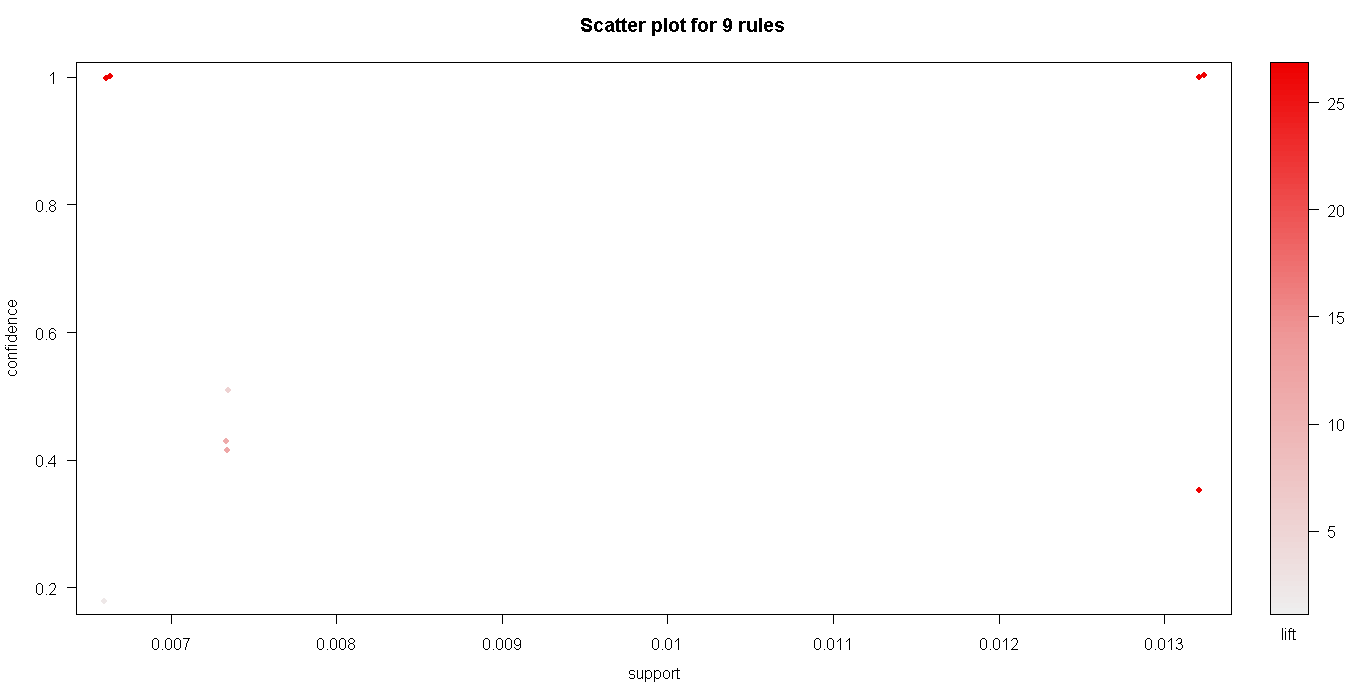
>

>

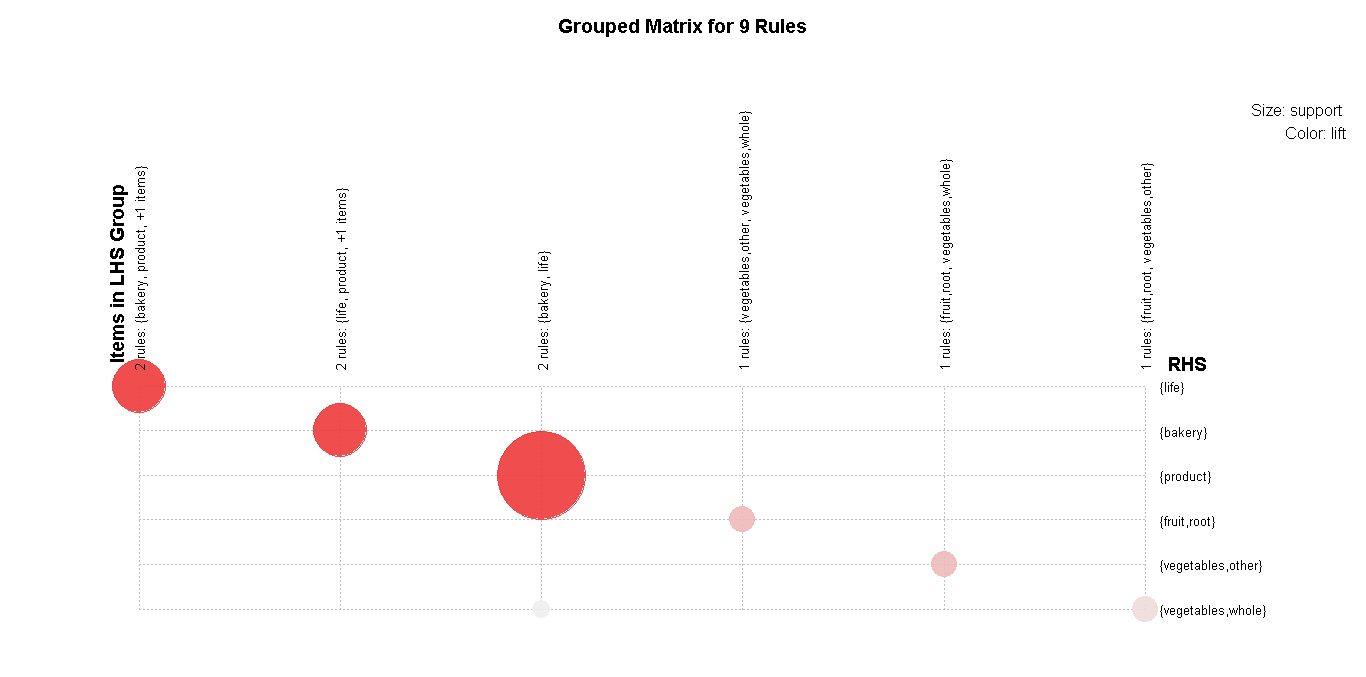
>

> plot(groceries\_rules,method = "scatterplot")

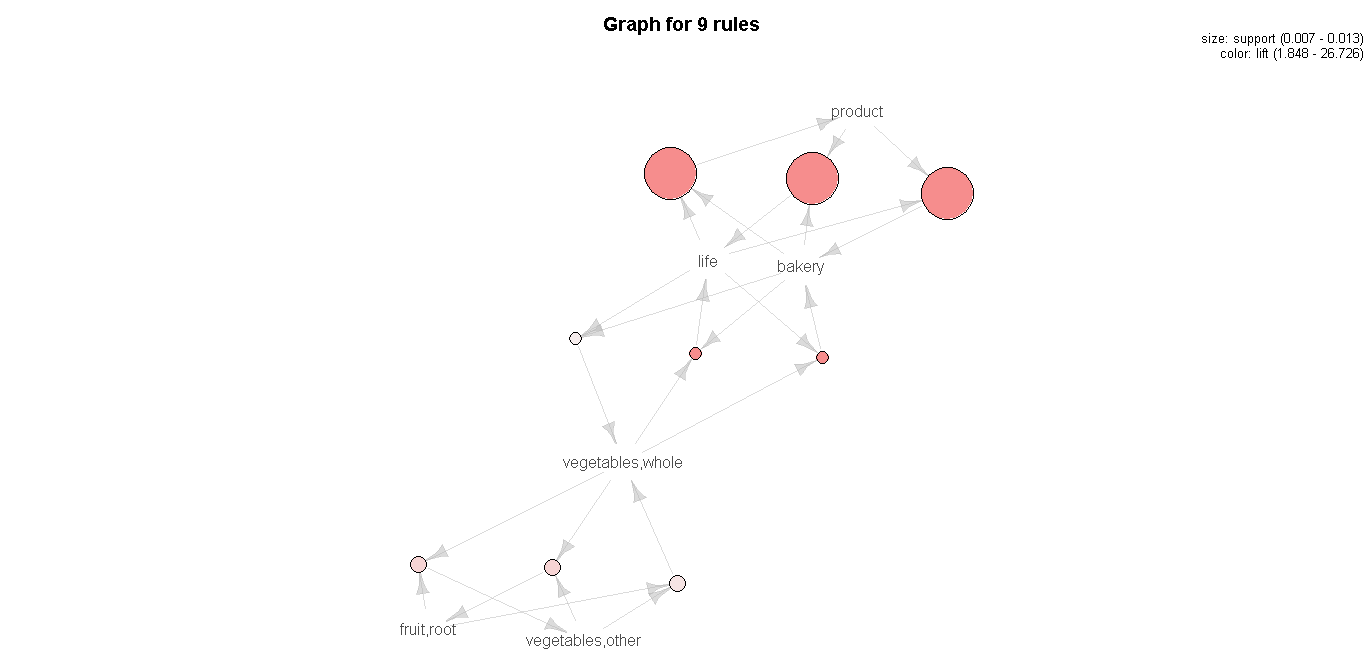
To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.



> plot(groceries\_rules,method = "grouped")



> plot(groceries\_rules,method = "graph")



Prob 2\_> Association rule on Movies

Soln->

> my\_mov<-read.csv(file.choose())

> mov<-as.matrix(my\_mov)

> library(arules)

> # the below function CONVERTS DUMMY VARIABLE TABLE INTO A TRANSACTION ...THANK GOD

> mov\_tran <- as(mov, "transactions")

> inspect(mov\_tran)

items

[1] {Sixth.Sense,LOTR1,Harry.Potter1,LOTR2,Green.Mile}

[2] {Gladiator,Patriot,Braveheart}

[3] {LOTR1,LOTR2}

[4] {Sixth.Sense,Gladiator,Patriot}

[5] {Sixth.Sense,Gladiator,Patriot}

[6] {Sixth.Sense,Gladiator,Patriot}

[7] {Harry.Potter1,Harry.Potter2}

[8] {Gladiator,Patriot}

[9] {Sixth.Sense,Gladiator,Patriot}

[10] {Sixth.Sense,Gladiator,LOTR,Green.Mile}

> class(mov\_tran)

[1] "transactions"

attr(,"package")

[1] "arules"

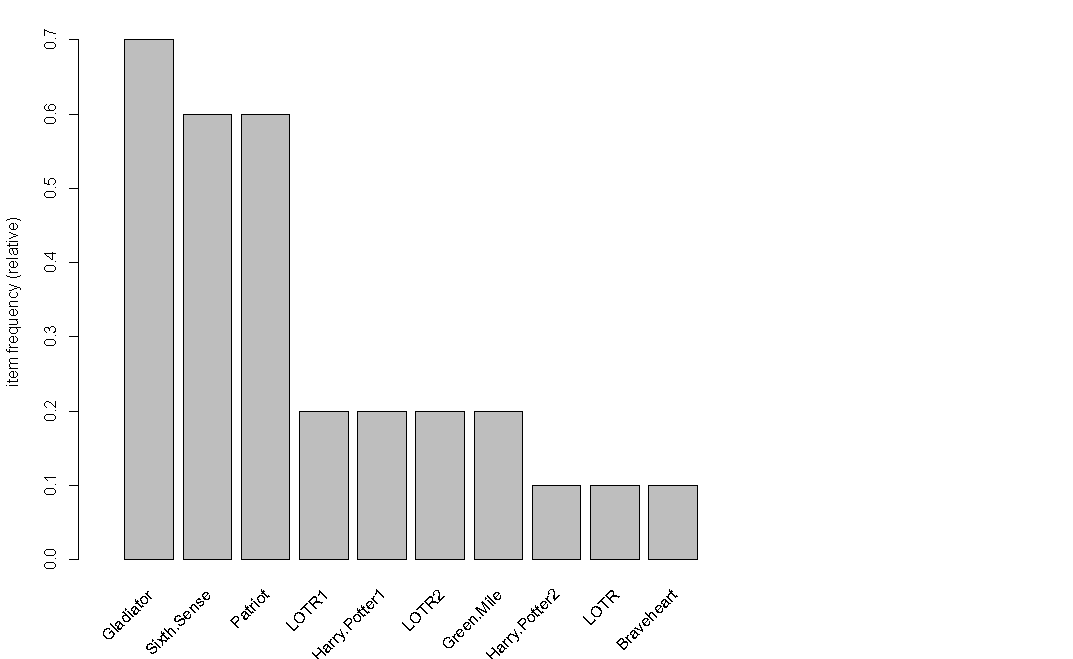
>

>

> # itemFrequencyPlot can be applicable only for transaction data

> # count of each item from all the transactions

> itemFrequencyPlot(mov\_tran,topN=20)



> mov\_tran\_rules<-apriori(mov\_tran,parameter = list(support = 0.002,confidence = 0.05,minlen=3))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext

0.05 0.1 1 none FALSE TRUE 5 0.002 3 10 rules FALSE

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 ...[10 item(s), 10 transaction(s)] done [0.00s].

sorting and recoding items ... [10 item(s)] done [0.00s].

creating transaction tree ... done [0.00s].

checking subsets of size 1 2 3 4 5 done [0.00s].

writing ... [77 rule(s)] done [0.00s].

creating S4 object ... done [0.00s].

>

> library(arulesViz)

>

>

> mov\_tran\_rules<-apriori(mov\_tran,parameter = list(support = 0.09,confidence = 0.05,minlen=4))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext

0.05 0.1 1 none FALSE TRUE 5 0.09 4 10 rules FALSE

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 ...[10 item(s), 10 transaction(s)] done [0.00s].

sorting and recoding items ... [10 item(s)] done [0.00s].

creating transaction tree ... done [0.00s].

checking subsets of size 1 2 3 4 5 done [0.00s].

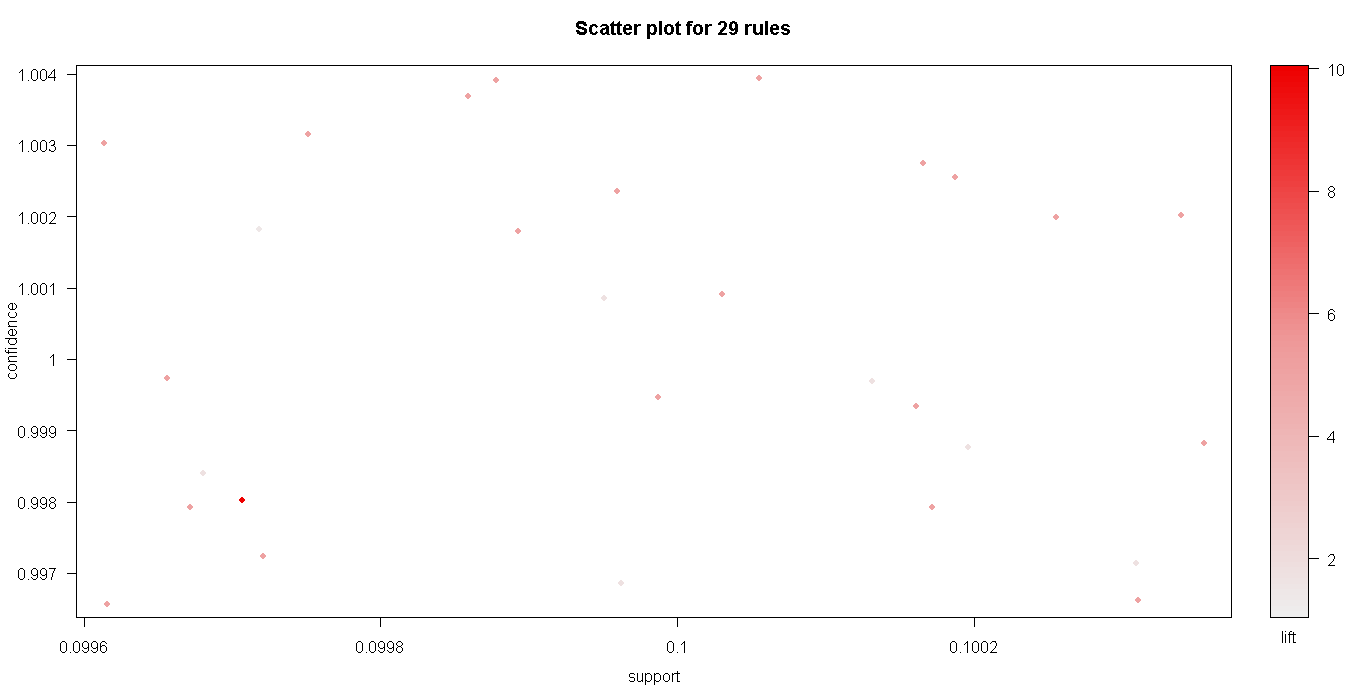
writing ... [29 rule(s)] done [0.00s].

creating S4 object ... done [0.04s].

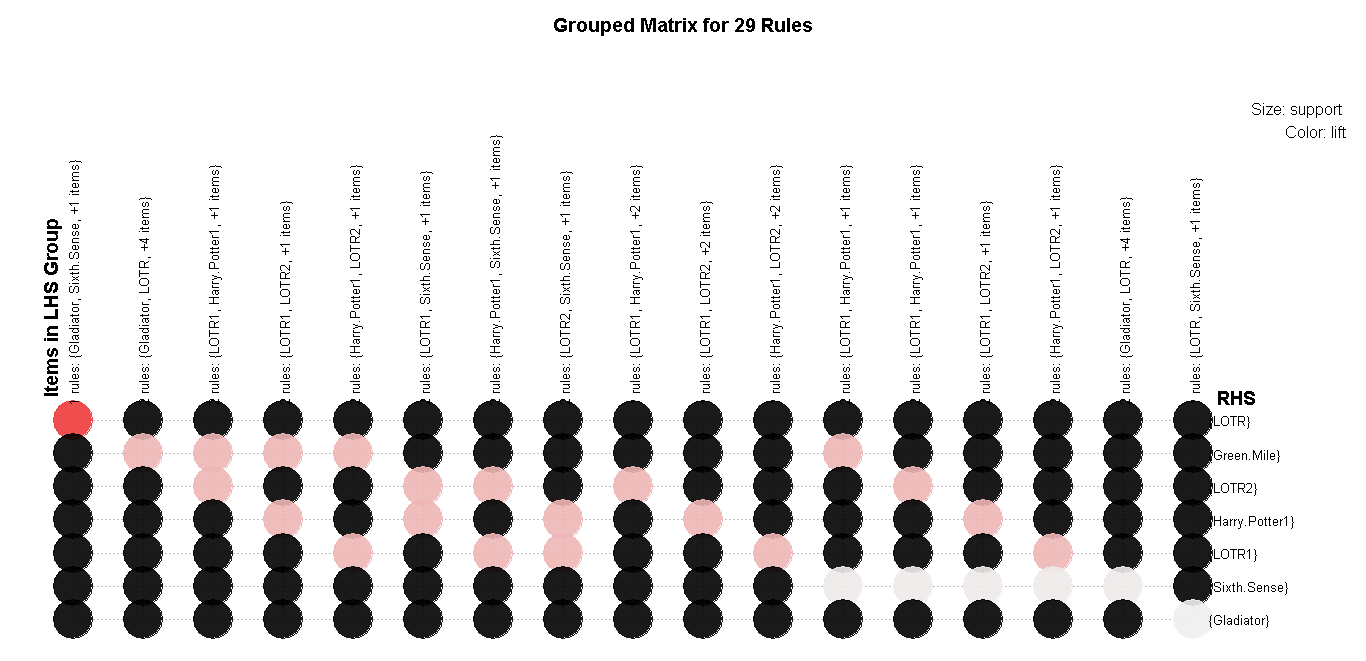
>

> plot(mov\_tran\_rules,method = "scatterplot")

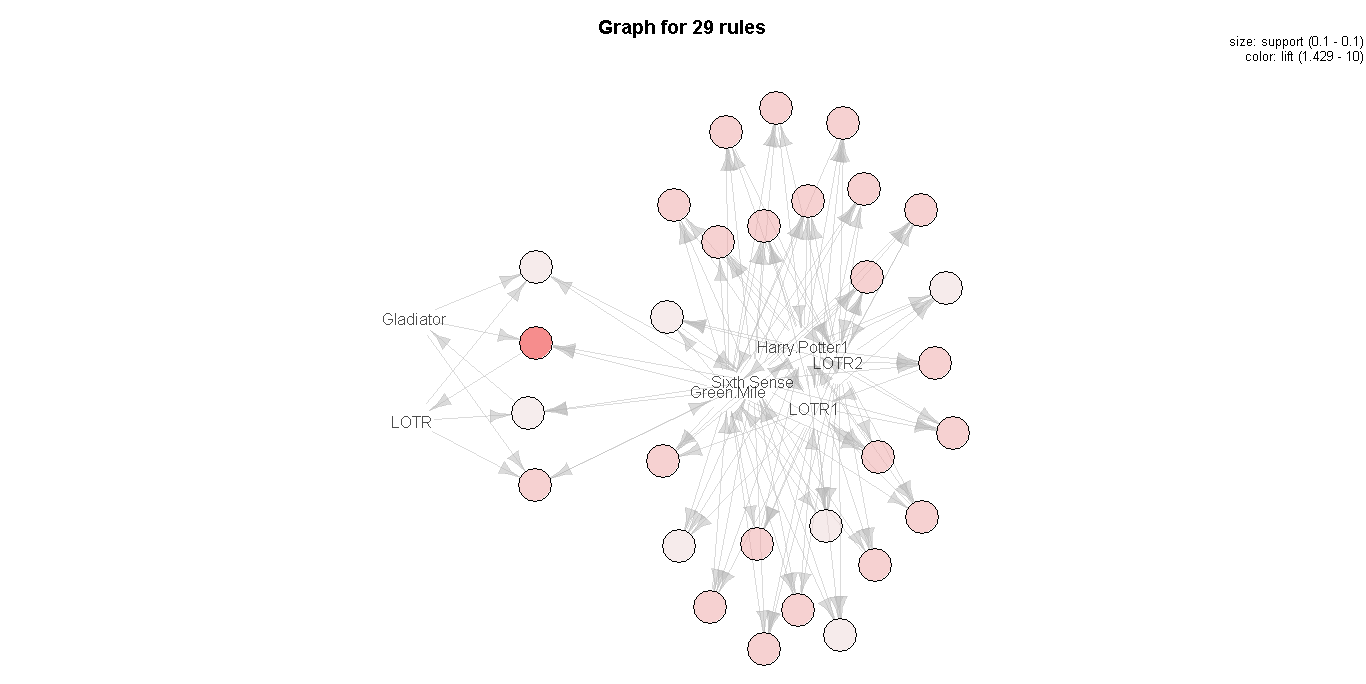
To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.



> plot(mov\_tran\_rules,method = "grouped")



> plot(mov\_tran\_rules,method = "graph")



> class(bk\_tran)

[1] "transactions"

attr(,"package")

[1] "arules"

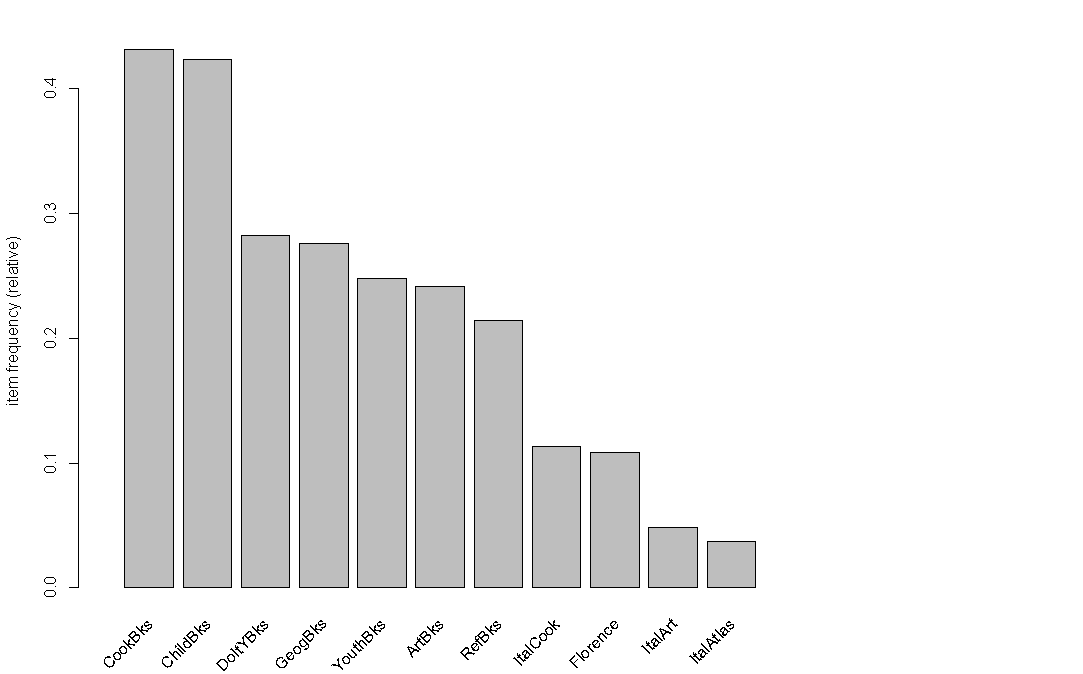
>

>

> # itemFrequencyPlot can be applicable only for transaction data

> # count of each item from all the transactions

> itemFrequencyPlot(bk\_tran,topN=20)



> bk\_tran\_rules<-apriori(bk\_tran,parameter = list(support = 0.002,confidence = 0.05,minlen=3))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext

0.05 0.1 1 none FALSE TRUE 5 0.002 3 10 rules FALSE

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 4

set item appearances ...[0 item(s)] done [0.00s].

set transactions ...[11 item(s), 2000 transaction(s)] done [0.00s].

sorting and recoding items ... [11 item(s)] done [0.00s].

creating transaction tree ... done [0.00s].

checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.00s].

writing ... [8434 rule(s)] done [0.01s].

creating S4 object ... done [0.00s].

Warning message:

In apriori(bk\_tran, parameter = list(support = 0.002, confidence = 0.05, :

Mining stopped (maxlen reached). Only patterns up to a length of 10 returned!

>

> library(arulesViz)

>

>

> #increase the support ; decrease the confidence ; and increase the minlen to reduse th

> bk\_tran\_rules<-apriori(bk\_tran,parameter = list(support = 0.0835,confidence = 0.05,minlen=4))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext

0.05 0.1 1 none FALSE TRUE 5 0.0835 4 10 rules FALSE

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 167

set item appearances ...[0 item(s)] done [0.00s].

set transactions ...[11 item(s), 2000 transaction(s)] done [0.00s].

sorting and recoding items ... [9 item(s)] done [0.00s].

creating transaction tree ... done [0.00s].

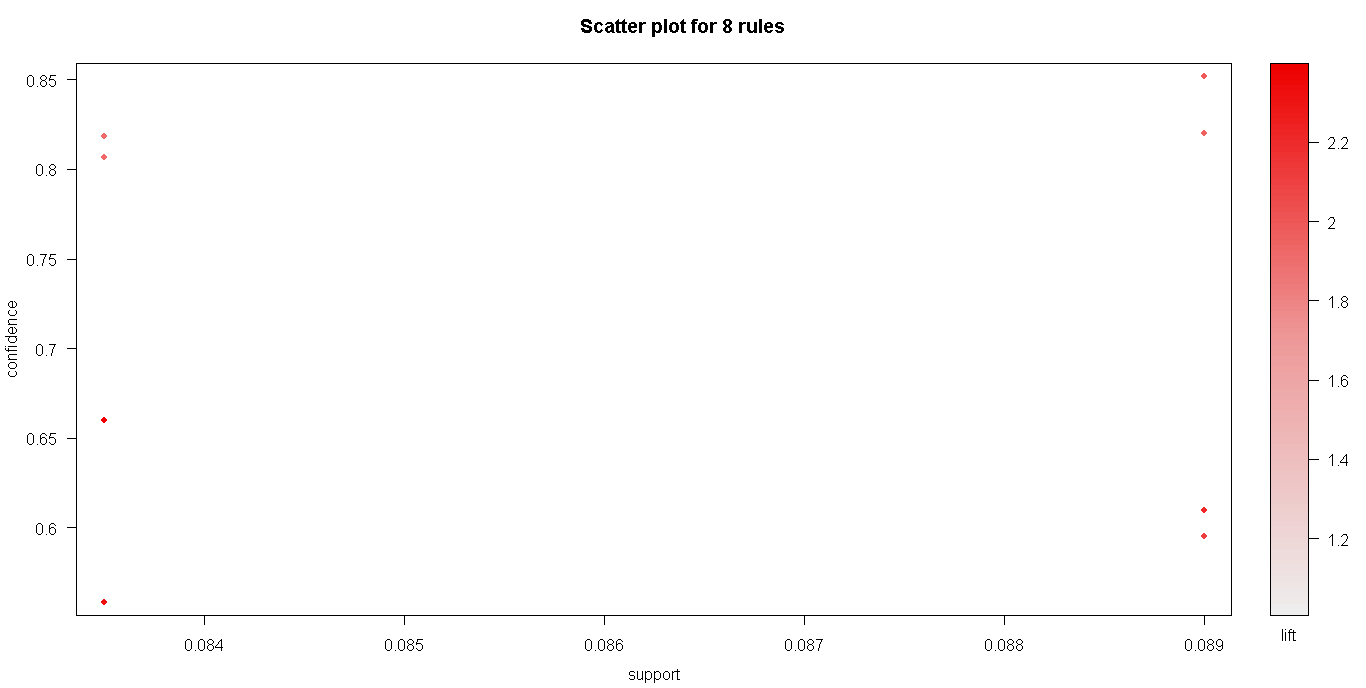
checking subsets of size 1 2 3 4 done [0.00s].

writing ... [8 rule(s)] done [0.00s].

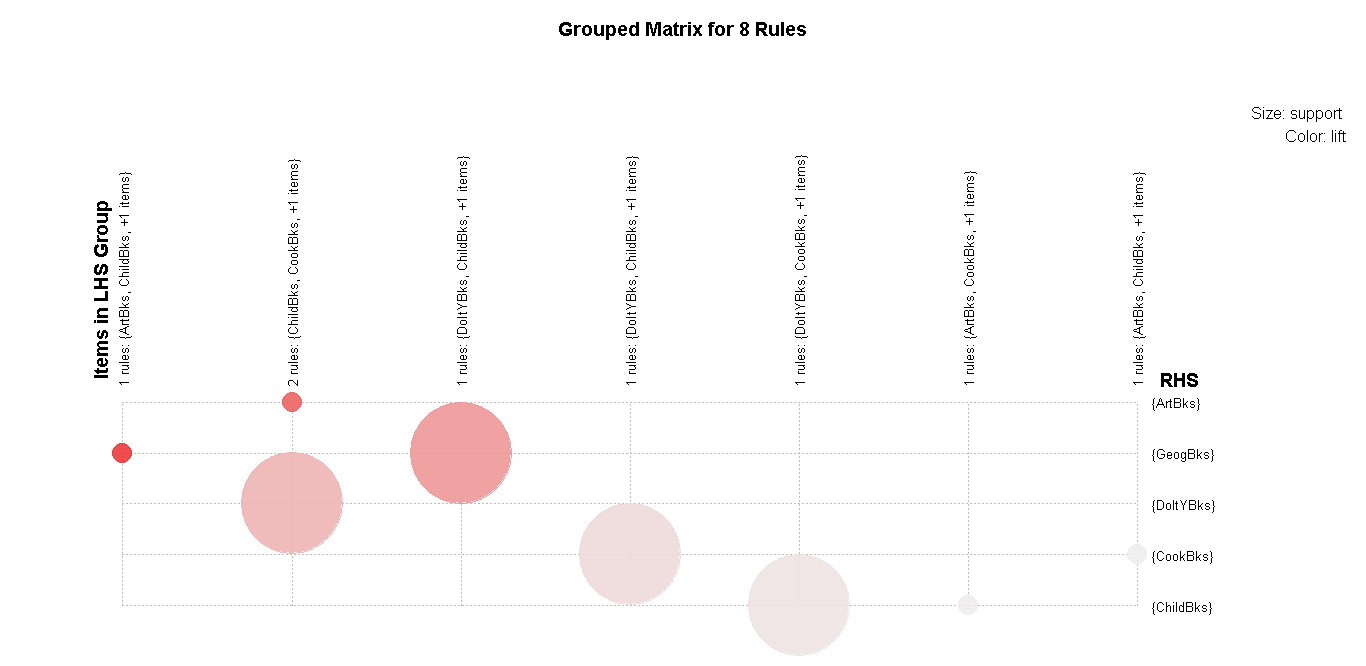
creating S4 object ... done [0.00s].

>

> plot(bk\_tran\_rules,method = "scatterplot")



> plot(bk\_tran\_rules,method = "grouped")



> plot(bk\_tran\_rules,method = "graph")

