install.packages("arules") # install the package

library("arules") # invoke the package

**#Visualizing Association Rules Package arulesViz supports visualization**

install.packages("arulesViz")

library("arulesViz")

**#importing data set**

Movie<-read.csv(file.choose(),header=T)

View(Movie)

**# making rules using apriori algorithm**

**# Keep changing support and confidence values to obtain different rules**

**# here i am taking 6 to 15 column for creating rules, it is in binary format**

**# convert binary data int a matrix format using as. matrix**

Movie\_rules1 <- apriori(as.matrix(Movie[,6:15]),parameter=list(support=0.01,confidence=0.7))

Apriori

Parameter specification:

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

0.7 0.1 1 none FALSE TRUE 5 0.01 1 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 ...[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 ... [79 rule(s)] done [0.00s].

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

> Movie\_rules1

set of 79 rules

Movie\_rules1

Set of 79 rules

#here creating 79 rules

#display the rules using inspect command

inspect(Movie\_rules1[1:10])

lhs rhs support confidence coverage lift count

[1] {} => {Gladiator} 0.7 0.7 1.0 1.000000 7

[2] {Harry.Potter2} => {Harry.Potter1} 0.1 1.0 0.1 5.000000 1

[3] {Braveheart} => {Patriot} 0.1 1.0 0.1 1.666667 1

[4] {Braveheart} => {Gladiator} 0.1 1.0 0.1 1.428571 1

[5] {LOTR} => {Green.Mile} 0.1 1.0 0.1 5.000000 1

[6] {LOTR} => {Gladiator} 0.1 1.0 0.1 1.428571 1

[7] {LOTR} => {Sixth.Sense} 0.1 1.0 0.1 1.666667 1

[8] {LOTR1} => {LOTR2} 0.2 1.0 0.2 5.000000 2

[9] {LOTR2} => {LOTR1} 0.2 1.0 0.2 5.000000 2

[10] {Green.Mile} => {Sixth.Sense} 0.2 1.0 0.2 1.666667 2

**# check the presence of redundant rules**

redundant\_rule1<-is.redundant(Movie\_rules1)

redundant\_rule1

[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE

[15] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE

[29] FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE

[43] FALSE TRUE FALSE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[71] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

summary(redundant\_rule1)

summary(redundant\_rule1)

Mode FALSE TRUE

logical 30 49

**#remove the redundant rules**

Movie\_rules1<- Movie\_rules1[!redundant\_rule1]

Movie\_rules1

set of 30 rules

**# we can see that there we have 30 rules remains**

**#display first 10 rules by highest lift ratios**

inspect(sort(Movie\_rules1[1:10],by="lift"))

lhs rhs support confidence coverage lift count

[1] {Harry.Potter2} => {Harry.Potter1} 0.1 1.0 0.1 5.000000 1

[2] {LOTR} => {Green.Mile} 0.1 1.0 0.1 5.000000 1

[3] {LOTR1} => {LOTR2} 0.2 1.0 0.2 5.000000 2

[4] {LOTR2} => {LOTR1} 0.2 1.0 0.2 5.000000 2

[5] {Braveheart} => {Patriot} 0.1 1.0 0.1 1.666667 1

[6] {LOTR} => {Sixth.Sense} 0.1 1.0 0.1 1.666667 1

[7] {Green.Mile} => {Sixth.Sense} 0.2 1.0 0.2 1.666667 2

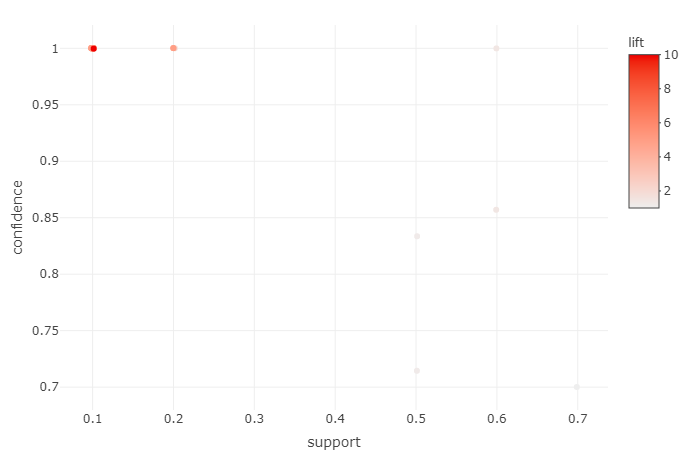
[8] {Braveheart} => {Gladiator} 0.1 1.0 0.1 1.428571 1

[9] {LOTR} => {Gladiator} 0.1 1.0 0.1 1.428571 1

[10] {} => {Gladiator} 0.7 0.7 1.0 1.000000 7

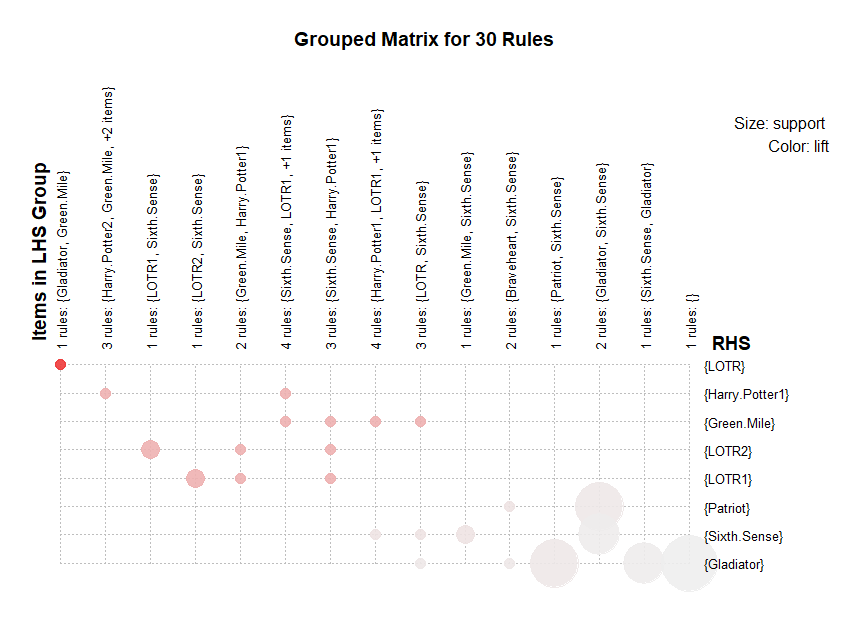
#visualize the rules

plotly\_arules(Movie\_rules1)



**##grouped matrix for association rules**

plot(Movie\_rules1,method="grouped")



################################################################

**#change support value and confident value**

Movie\_rules2 <- apriori(as.matrix(Movie[,6:15]),parameter=list(support=0.2,confidence=0.8))

Movie\_rules2

Apriori

Parameter specification:

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

0.8 0.1 1 none FALSE TRUE 5 0.2 1 10 rules TRUE

Algorithmic control:

filter tree heap memopt load sort verbose

0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 2

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

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

sorting and recoding items ... [7 item(s)] done [0.00s].

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

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

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

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

**#here creating 8 rules**

#display the rules using inspect command

inspect(Movie\_rules2)

lhs rhs support confidence coverage lift count

[1] {LOTR1} => {LOTR2} 0.2 1.0000000 0.2 5.000000 2

[2] {LOTR2} => {LOTR1} 0.2 1.0000000 0.2 5.000000 2

[3] {Green.Mile} => {Sixth.Sense} 0.2 1.0000000 0.2 1.666667 2

[4] {Patriot} => {Gladiator} 0.6 1.0000000 0.6 1.428571 6

[5] {Gladiator} => {Patriot} 0.6 0.8571429 0.7 1.428571 6

[6] {Sixth.Sense} => {Gladiator} 0.5 0.8333333 0.6 1.190476 5

[7] {Sixth.Sense,Patriot} => {Gladiator} 0.4 1.0000000 0.4 1.428571 4

[8] {Sixth.Sense,Gladiator} => {Patriot} 0.4 0.8000000 0.5 1.333333 4

**# check the presence of redundant rules**

redundant\_rule2<-is.redundant(Movie\_rules2)

redundant\_rule2

summary(redundant\_rule2)

summary(redundant\_rule2)

Mode FALSE TRUE

logical 6 2

**#remove the redundant rules**

Movie\_rules2<- Movie\_rules2[!redundant\_rule2]

Movie\_rules2

**# we can see that there we have 30 rules remains**

#display rules by highest lift ratios

inspect(sort(Movie\_rules2,by="lift"))

lhs rhs support confidence coverage lift count

[1] {LOTR1} => {LOTR2} 0.2 1.0000000 0.2 5.000000 2

[2] {LOTR2} => {LOTR1} 0.2 1.0000000 0.2 5.000000 2

[3] {Green.Mile} => {Sixth.Sense} 0.2 1.0000000 0.2 1.666667 2

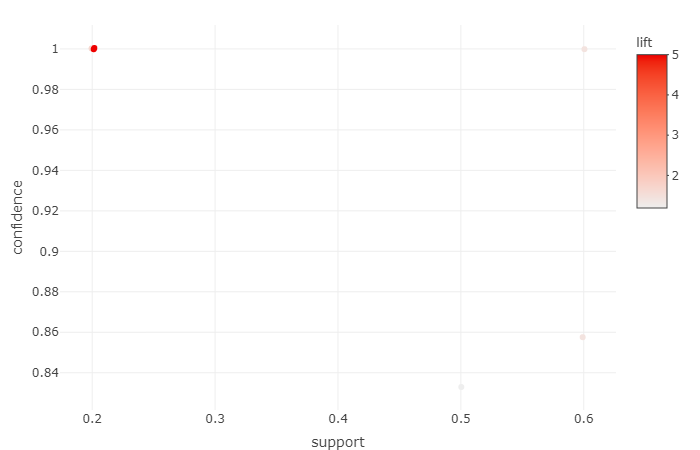
[4] {Patriot} => {Gladiator} 0.6 1.0000000 0.6 1.428571 6

[5] {Gladiator} => {Patriot} 0.6 0.8571429 0.7 1.428571 6

[6] {Sixth.Sense} => {Gladiator} 0.5 0.8333333 0.6 1.190476 5

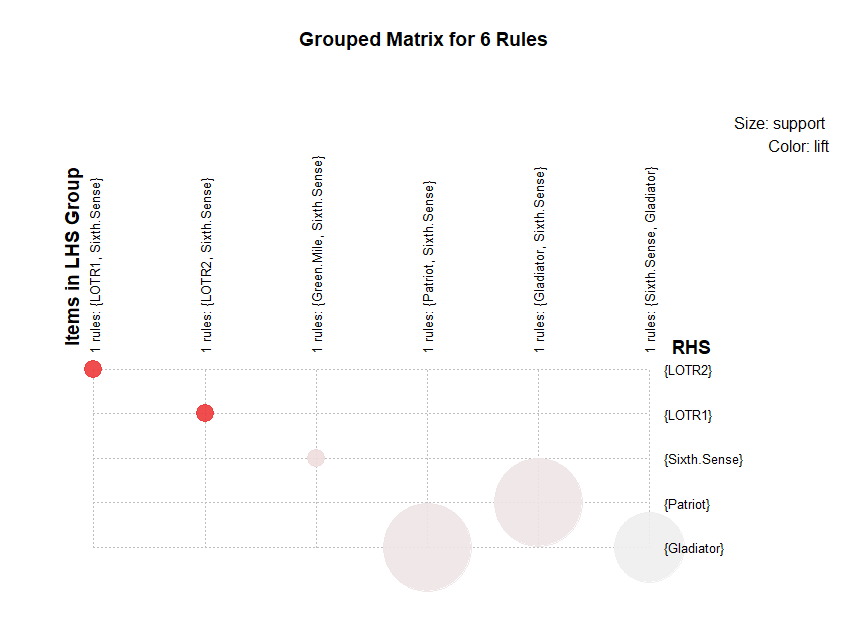
**#visualize the rules**

plotly\_arules(Movie\_rules2)



**##grouped matrix for association rules**

plot(Movie\_rules2,method="grouped")



##############################################################33

#**change support value and confident value**

Movie\_rules3 <- apriori(as.matrix(Movie[,6:15]),parameter=list(support=0.03,confidence=0.85))

|  |
| --- |
| Parameter specification:  confidence minval smax arem aval originalSupport maxtime support minlen maxlen target ext  0.85 0.1 1 none FALSE TRUE 5 0.03 1 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 ...[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 ... [75 rule(s)] done [0.00s].  creating S4 object ... done [0.00s]. |
|  |
| |  | | --- | | > | |

Movie\_rules3

#here creating 75 rules

#display the rules using inspect command

inspect(Movie\_rules3[1:10])

lhs rhs support confidence coverage lift count

[1] {Harry.Potter2} => {Harry.Potter1} 0.1 1 0.1 5.000000 1

[2] {Braveheart} => {Patriot} 0.1 1 0.1 1.666667 1

[3] {Braveheart} => {Gladiator} 0.1 1 0.1 1.428571 1

[4] {LOTR} => {Green.Mile} 0.1 1 0.1 5.000000 1

[5] {LOTR} => {Gladiator} 0.1 1 0.1 1.428571 1

[6] {LOTR} => {Sixth.Sense} 0.1 1 0.1 1.666667 1

[7] {LOTR1} => {LOTR2} 0.2 1 0.2 5.000000 2

[8] {LOTR2} => {LOTR1} 0.2 1 0.2 5.000000 2

[9] {Green.Mile} => {Sixth.Sense} 0.2 1 0.2 1.666667 2

[10] {Patriot} => {Gladiator} 0.6 1 0.6 1.428571 6

# check the presence of redundant rules

redundant\_rule3<-is.redundant(Movie\_rules3)

redundant\_rule3

[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE

[15] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE

[29] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE

[43] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[71] TRUE TRUE TRUE TRUE TRUE

summary(redundant\_rule3)

summary(redundant\_rule3)

Mode FALSE TRUE

logical 27 48

#remove the redundant rules

Movie\_rules3<- Movie\_rules3[!redundant\_rule3]

Movie\_rules3

# we can see that there we have 30 rules remains

#display rules by highest lift ratios

inspect(sort(Movie\_rules3[1:10],by="lift"))

lhs rhs support confidence coverage lift count

[1] {Harry.Potter2} => {Harry.Potter1} 0.1 1 0.1 5.000000 1

[2] {LOTR} => {Green.Mile} 0.1 1 0.1 5.000000 1

[3] {LOTR1} => {LOTR2} 0.2 1 0.2 5.000000 2

[4] {LOTR2} => {LOTR1} 0.2 1 0.2 5.000000 2

[5] {Braveheart} => {Patriot} 0.1 1 0.1 1.666667 1

[6] {LOTR} => {Sixth.Sense} 0.1 1 0.1 1.666667 1

[7] {Green.Mile} => {Sixth.Sense} 0.2 1 0.2 1.666667 2

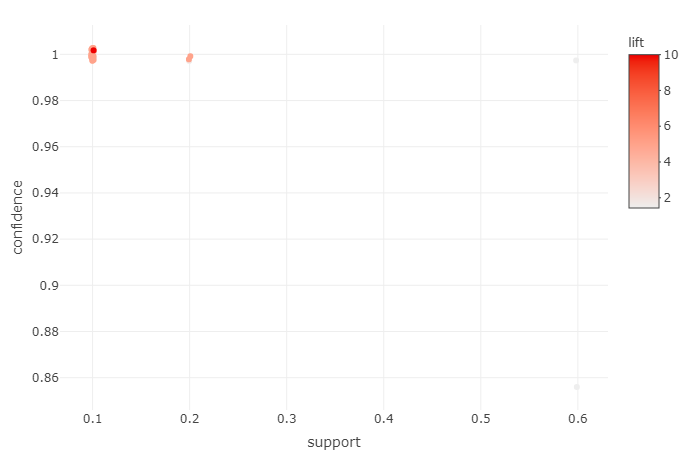
[8] {Braveheart} => {Gladiator} 0.1 1 0.1 1.428571 1

[9] {LOTR} => {Gladiator} 0.1 1 0.1 1.428571 1

[10] {Patriot} => {Gladiator} 0.6 1 0.6 1.428571 6

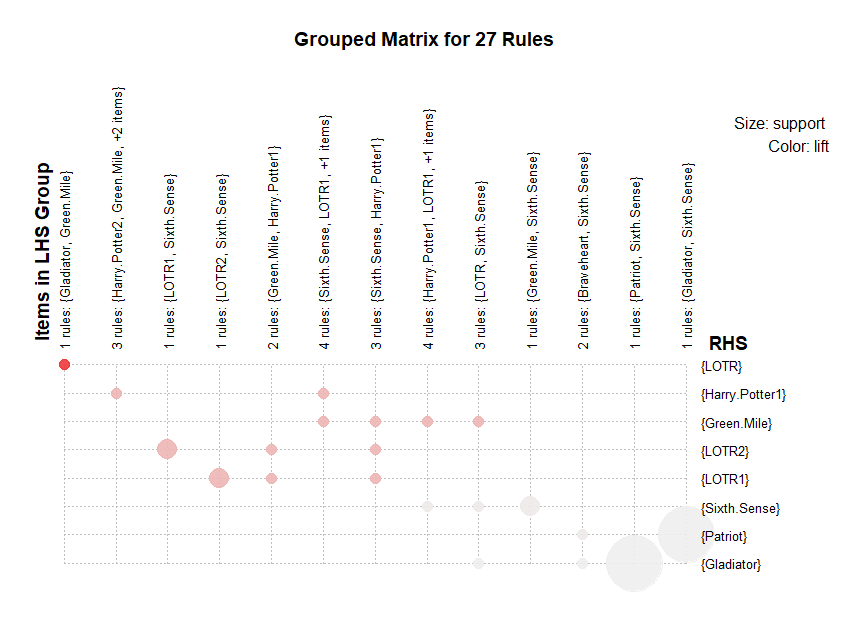
#visualize the rules

plotly\_arules(Movie\_rules3)



**##grouped matrix for association rules**

plot(Movie\_rules3,method="grouped")



##############################################################################

**#change support value and confident value**

Movie\_rules4 <- apriori(as.matrix(Movie[,6:15]),parameter=list(support=0.02,confidence=0.90))

Apriori

Parameter specification:

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

0.9 0.1 1 none FALSE TRUE 5 0.02 1 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 ...[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 ... [74 rule(s)] done [0.00s].

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

Movie\_rules4

#here creating 75 rules

#display the rules using inspect command

inspect(Movie\_rules4[1:10])

lhs rhs support confidence coverage lift count

[1] {Harry.Potter2} => {Harry.Potter1} 0.1 1 0.1 5.000000 1

[2] {Braveheart} => {Patriot} 0.1 1 0.1 1.666667 1

[3] {Braveheart} => {Gladiator} 0.1 1 0.1 1.428571 1

[4] {LOTR} => {Green.Mile} 0.1 1 0.1 5.000000 1

[5] {LOTR} => {Gladiator} 0.1 1 0.1 1.428571 1

[6] {LOTR} => {Sixth.Sense} 0.1 1 0.1 1.666667 1

[7] {LOTR1} => {LOTR2} 0.2 1 0.2 5.000000 2

[8] {LOTR2} => {LOTR1} 0.2 1 0.2 5.000000 2

[9] {Green.Mile} => {Sixth.Sense} 0.2 1 0.2 1.666667 2

[10] {Patriot} => {Gladiator} 0.6 1 0.6 1.428571 6

# check the presence of redundant rules

redundant\_rule4<-is.redundant(Movie\_rules4)

redundant\_rule4

[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

[15] FALSE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE TRUE

[29] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE

[43] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[57] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE

[71] TRUE TRUE TRUE TRUE

summary(redundant\_rule4)

summary(redundant\_rule4)

Mode FALSE TRUE

logical 26 48

**#remove the redundant rules**

Movie\_rules4<- Movie\_rules4[!redundant\_rule4]

Movie\_rules4

**# we can see that there we have 30 rules remains**

**#display rules by highest lift ratios**

inspect(sort(Movie\_rules4,by="lift"))

lhs rhs support confidence coverage lift count

[1] {Harry.Potter2} => {Harry.Potter1} 0.1 1 0.1 5.000000 1

[2] {LOTR} => {Green.Mile} 0.1 1 0.1 5.000000 1

[3] {LOTR1} => {LOTR2} 0.2 1 0.2 5.000000 2

[4] {LOTR2} => {LOTR1} 0.2 1 0.2 5.000000 2

[5] {Braveheart} => {Patriot} 0.1 1 0.1 1.666667 1

[6] {LOTR} => {Sixth.Sense} 0.1 1 0.1 1.666667 1

[7] {Green.Mile} => {Sixth.Sense} 0.2 1 0.2 1.666667 2

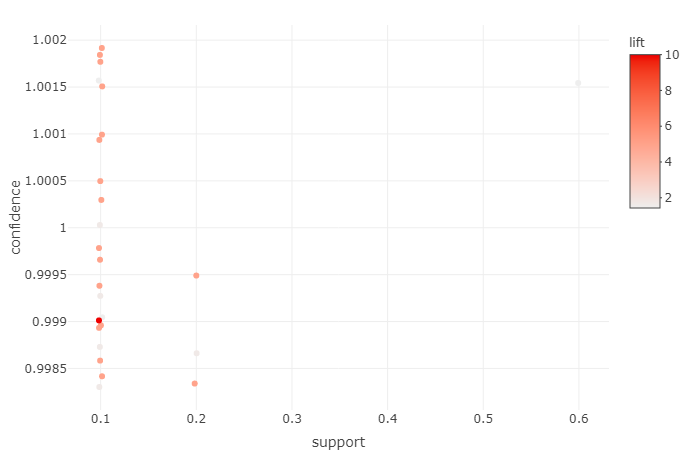
[8] {Braveheart} => {Gladiator} 0.1 1 0.1 1.428571 1

[9] {LOTR} => {Gladiator} 0.1 1 0.1 1.428571 1

[10] {Patriot} => {Gladiator} 0.6 1 0.6 1.428571 6

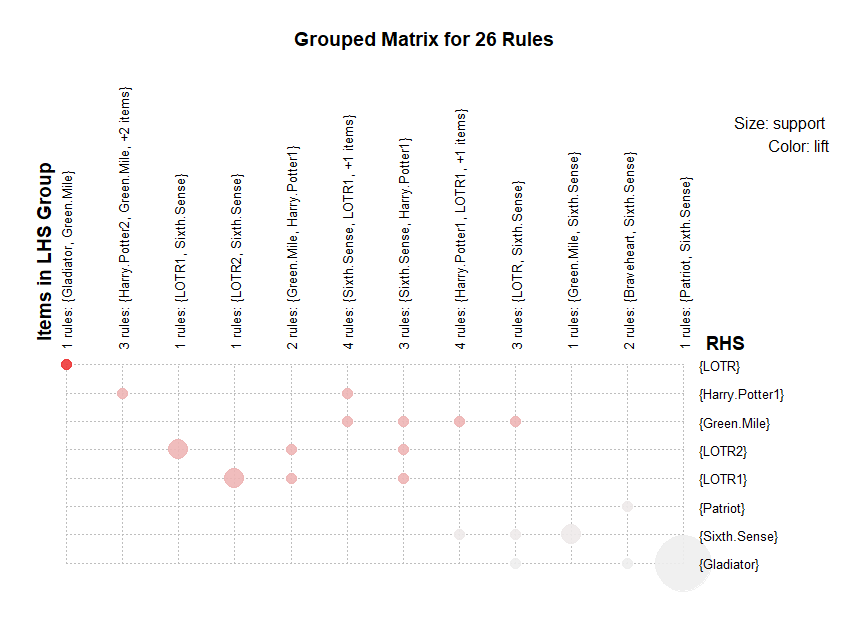
#visualize the rules

plotly\_arules(Movie\_rules4)



**##grouped matrix for association rules**

plot(Movie\_rules4,method="grouped")



Following are top 10 rules of movies

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column1** | **rules** | **support** | **cinfidence** | **coverage** | **lift** | **count** |
| 1 | {Harry.Potter2} => {Harry.Potter1} | 0.1 | 1 | 0.1 | 5 | 1 |
| 4 | {LOTR} => {Green.Mile} | 0.1 | 1 | 0.1 | 5 | 1 |
| 7 | {LOTR1} => {LOTR2} | 0.2 | 1 | 0.2 | 5 | 2 |
| 8 | {LOTR2} => {LOTR1} | 0.2 | 1 | 0.2 | 5 | 2 |
| 2 | {Braveheart} => {Patriot} | 0.1 | 1 | 0.1 | 1.666667 | 1 |
| 6 | {LOTR} => {Sixth.Sense} | 0.1 | 1 | 0.1 | 1.666667 | 1 |
| 9 | {Green.Mile} => {Sixth.Sense} | 0.2 | 1 | 0.2 | 1.666667 | 2 |
| 3 | {Braveheart} => {Gladiator} | 0.1 | 1 | 0.1 | 1.428571 | 1 |
| 5 | {LOTR} => {Gladiator} | 0.1 | 1 | 0.1 | 1.428571 | 1 |
| 10 | {Patriot} => {Gladiator} | 0.6 | 1 | 0.6 | 1.428571 | 6 |