> movies <- read.csv(file.choose())

> View(movies)

> head(movies)

V1 V2 V3 V4 V5 Sixth.Sense Gladiator LOTR1

1 Sixth Sense LOTR1 Harry Potter1 Green Mile LOTR2 1 0 1

2 Gladiator Patriot Braveheart 0 1 0

3 LOTR1 LOTR2 0 0 1

4 Gladiator Patriot Sixth Sense 1 1 0

5 Gladiator Patriot Sixth Sense 1 1 0

6 Gladiator Patriot Sixth Sense 1 1 0

Harry.Potter1 Patriot LOTR2 Harry.Potter2 LOTR Braveheart Green.Mile

1 1 0 1 0 0 0 1

2 0 1 0 0 0 1 0

3 0 0 1 0 0 0 0

4 0 1 0 0 0 0 0

5 0 1 0 0 0 0 0

6 0 1 0 0 0 0 0

> summary(movies)

V1 V2 V3 V4

Length:10 Length:10 Length:10 Length:10

Class :character Class :character Class :character Class :character

Mode :character Mode :character Mode :character Mode :character

V5 Sixth.Sense Gladiator LOTR1 Harry.Potter1

Length:10 Min. :0.0 Min. :0.00 Min. :0.0 Min. :0.0

Class :character 1st Qu.:0.0 1st Qu.:0.25 1st Qu.:0.0 1st Qu.:0.0

Mode :character Median :1.0 Median :1.00 Median :0.0 Median :0.0

Mean :0.6 Mean :0.70 Mean :0.2 Mean :0.2

3rd Qu.:1.0 3rd Qu.:1.00 3rd Qu.:0.0 3rd Qu.:0.0

Max. :1.0 Max. :1.00 Max. :1.0 Max. :1.0

Patriot LOTR2 Harry.Potter2 LOTR Braveheart

Min. :0.0 Min. :0.0 Min. :0.0 Min. :0.0 Min. :0.0

1st Qu.:0.0 1st Qu.:0.0 1st Qu.:0.0 1st Qu.:0.0 1st Qu.:0.0

Median :1.0 Median :0.0 Median :0.0 Median :0.0 Median :0.0

Mean :0.6 Mean :0.2 Mean :0.1 Mean :0.1 Mean :0.1

3rd Qu.:1.0 3rd Qu.:0.0 3rd Qu.:0.0 3rd Qu.:0.0 3rd Qu.:0.0

Max. :1.0 Max. :1.0 Max. :1.0 Max. :1.0 Max. :1.0

Green.Mile

Min. :0.0

1st Qu.:0.0

Median :0.0

Mean :0.2

3rd Qu.:0.0

Max. :1.0

> describe(movies)

vars n mean sd median trimmed mad min max range skew kurtosis

V1\* 1 10 1.9 1.29 1.0 1.75 0.00 1 4 3 0.73 -1.38

V2\* 2 10 4.0 1.49 5.0 4.25 0.00 1 5 4 -0.91 -0.89

V3\* 3 10 3.2 1.81 3.5 3.25 2.22 1 5 4 -0.16 -1.94

V4\* 4 10 1.2 0.42 1.0 1.12 0.00 1 2 1 1.28 -0.37

V5\* 5 10 1.1 0.32 1.0 1.00 0.00 1 2 1 2.28 3.57

Sixth.Sense 6 10 0.6 0.52 1.0 0.62 0.00 0 1 1 -0.35 -2.05

Gladiator 7 10 0.7 0.48 1.0 0.75 0.00 0 1 1 -0.75 -1.57

LOTR1 8 10 0.2 0.42 0.0 0.12 0.00 0 1 1 1.28 -0.37

Harry.Potter1 9 10 0.2 0.42 0.0 0.12 0.00 0 1 1 1.28 -0.37

Patriot 10 10 0.6 0.52 1.0 0.62 0.00 0 1 1 -0.35 -2.05

LOTR2 11 10 0.2 0.42 0.0 0.12 0.00 0 1 1 1.28 -0.37

Harry.Potter2 12 10 0.1 0.32 0.0 0.00 0.00 0 1 1 2.28 3.57

LOTR 13 10 0.1 0.32 0.0 0.00 0.00 0 1 1 2.28 3.57

Braveheart 14 10 0.1 0.32 0.0 0.00 0.00 0 1 1 2.28 3.57

Green.Mile 15 10 0.2 0.42 0.0 0.12 0.00 0 1 1 1.28 -0.37

se

V1\* 0.41

V2\* 0.47

V3\* 0.57

V4\* 0.13

V5\* 0.10

Sixth.Sense 0.16

Gladiator 0.15

LOTR1 0.13

Harry.Potter1 0.13

Patriot 0.16

LOTR2 0.13

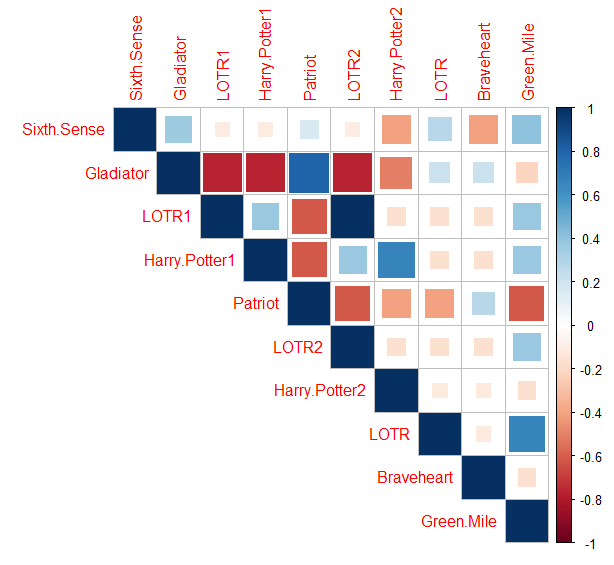
Harry.Potter2 0.10

LOTR 0.10

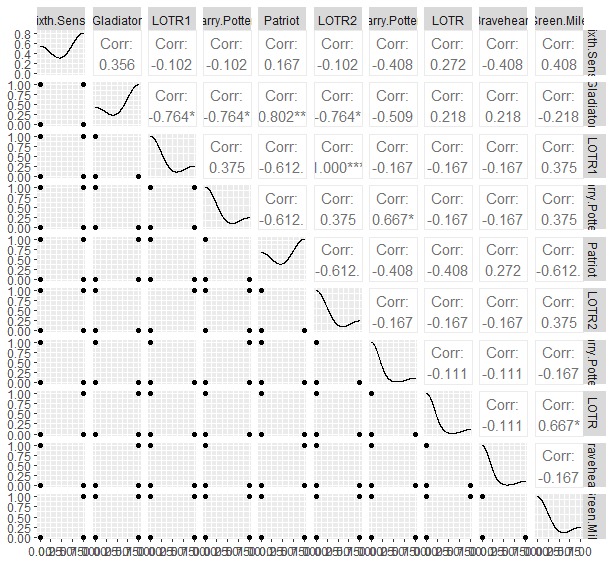
Braveheart 0.10

Green.Mile 0.13

> corrplot(cor(movies[,6:15]),method = "square",type = "upper")



> ggpairs(movies[,6:15])

  
#applying apriori algorithm

# confidence =0.05 , support =0.005 ,minlen =3

>movies\_rules <- apriori(as.matrix(movies[,6:15]),parameter = list(support = 0.005,confidence= 0.05,minlen=3))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen

0.05 0.1 1 none FALSE TRUE 5 0.005 3 10

target ext

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

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

>movies\_rules

Set of 77 rules

> inspect(head(sort(movies\_rules,by="lift")))

lhs rhs support confidence coverage

[1] {Gladiator,Green.Mile} => {LOTR} 0.1 1.0 0.1

[2] {Sixth.Sense,Gladiator,Green.Mile} => {LOTR} 0.1 1.0 0.1

[3] {Gladiator,LOTR} => {Green.Mile} 0.1 1.0 0.1

[4] {Sixth.Sense,LOTR} => {Green.Mile} 0.1 1.0 0.1

[5] {Sixth.Sense,Green.Mile} => {LOTR} 0.1 0.5 0.2

[6] {LOTR1,Harry.Potter1} => {LOTR2} 0.1 1.0 0.1

lift count

[1] 10 1

[2] 10 1

[3] 5 1

[4] 5 1

[5] 5 1

[6] 5 1

> inspect(head(sort(movies\_rules,by="confidence")))

lhs rhs support confidence coverage lift

[1] {Patriot,Braveheart} => {Gladiator} 0.1 1 0.1 1.428571

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

[3] {LOTR,Green.Mile} => {Gladiator} 0.1 1 0.1 1.428571

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

[5] {Gladiator,Green.Mile} => {LOTR} 0.1 1 0.1 10.000000

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

count

[1] 1

[2] 1

[3] 1

[4] 1

[5] 1

[6] 1

> inspect(head(sort(movies\_rules,by="support")))

lhs rhs support confidence coverage lift

[1] {Gladiator,Patriot} => {Sixth.Sense} 0.4 0.6666667 0.6 1.111111

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

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

[4] {Patriot,Braveheart} => {Gladiator} 0.1 1.0000000 0.1 1.428571

[5] {Gladiator,Braveheart} => {Patriot} 0.1 1.0000000 0.1 1.666667

[6] {Gladiator,Patriot} => {Braveheart} 0.1 0.1666667 0.6 1.666667

count

[1] 4

[2] 4

[3] 4

[4] 1

[5] 1

[6] 1

> inspect(head(sort(movies\_rules,by=c("count","lift"))))# max count = 4

lhs rhs support confidence coverage

[1] {Sixth.Sense,Patriot} => {Gladiator} 0.4 1.0000000 0.4

[2] {Sixth.Sense,Gladiator} => {Patriot} 0.4 0.8000000 0.5

[3] {Gladiator,Patriot} => {Sixth.Sense} 0.4 0.6666667 0.6

[4] {Gladiator,Green.Mile} => {LOTR} 0.1 1.0000000 0.1

[5] {Sixth.Sense,Gladiator,Green.Mile} => {LOTR} 0.1 1.0000000 0.1

[6] {Gladiator,LOTR} => {Green.Mile} 0.1 1.0000000 0.1

lift count

[1] 1.428571 4

[2] 1.333333 4

[3] 1.111111 4

[4] 10.000000 1

[5] 10.000000 1

[6] 5.000000 1

> head(quality(movies\_rules))

support confidence coverage lift count

1 0.1 1.0000000 0.1 1.428571 1

2 0.1 1.0000000 0.1 1.666667 1

3 0.1 0.1666667 0.6 1.666667 1

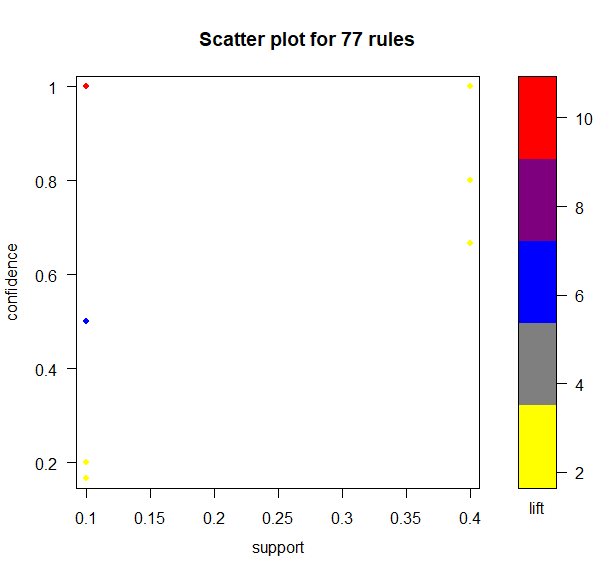
4 0.1 1.0000000 0.1 1.428571 1

5 0.1 1.0000000 0.1 5.000000 1

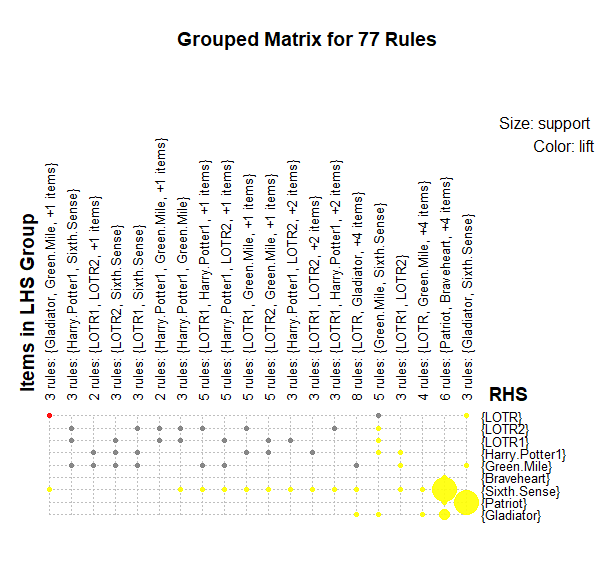
6 0.1 1.0000000 0.1 10.000000 1

> #visualisation

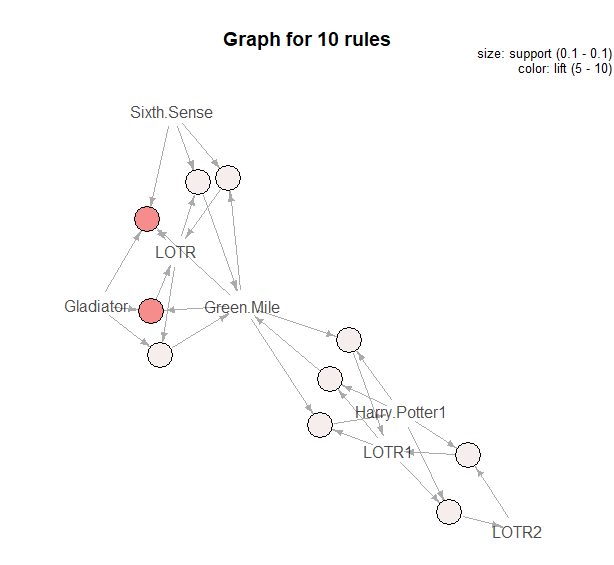
> plot(movies\_rules,method = "scatterplot",jitter=0,col=colfunction(5))



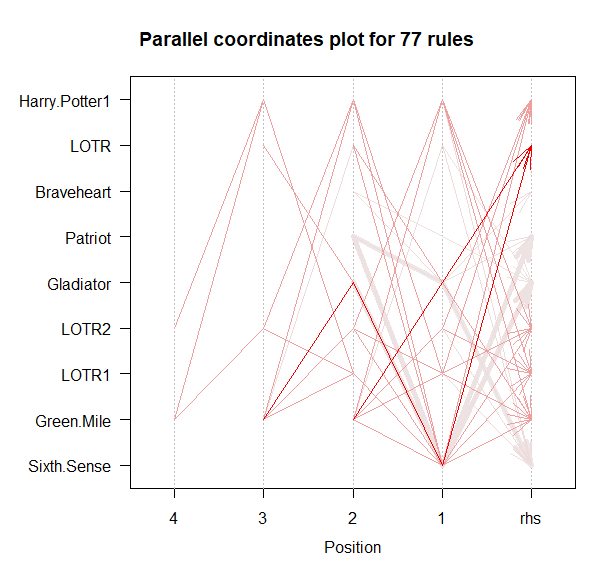
> plot(movies\_rules,method = "grouped matrix",col=colfunction(5))



> plot(head(sort(movies\_rules,by="lift"),n=10),method = "graph")



> plot(movies\_rules,method = "paracoord")



> plot(movies\_rules,method = "matrix")

Itemsets in Antecedent (LHS)

[1] "{Sixth.Sense,Gladiator,Green.Mile}"

[2] "{Gladiator,Green.Mile}"

[3] "{Sixth.Sense,LOTR1}"

[4] "{Sixth.Sense,Harry.Potter1}"

[5] "{Sixth.Sense,LOTR2}"

[6] "{Sixth.Sense,Gladiator,LOTR}"

[7] "{Sixth.Sense,LOTR1,Harry.Potter1}"

[8] "{Sixth.Sense,LOTR1,LOTR2}"

[9] "{Sixth.Sense,Harry.Potter1,LOTR2}"

[10] "{Sixth.Sense,LOTR1,Green.Mile}"

[11] "{Sixth.Sense,Harry.Potter1,Green.Mile}"

[12] "{Sixth.Sense,LOTR2,Green.Mile}"

[13] "{Sixth.Sense,LOTR1,Harry.Potter1,LOTR2}"

[14] "{Sixth.Sense,LOTR1,Harry.Potter1,Green.Mile}"

[15] "{Sixth.Sense,LOTR1,LOTR2,Green.Mile}"

[16] "{Sixth.Sense,Harry.Potter1,LOTR2,Green.Mile}"

[17] "{LOTR1,Harry.Potter1}"

[18] "{Harry.Potter1,LOTR2}"

[19] "{LOTR1,Green.Mile}"

[20] "{Harry.Potter1,Green.Mile}"

[21] "{LOTR2,Green.Mile}"

[22] "{Gladiator,LOTR}"

[23] "{LOTR1,Harry.Potter1,LOTR2}"

[24] "{LOTR1,Harry.Potter1,Green.Mile}"

[25] "{LOTR1,LOTR2,Green.Mile}"

[26] "{Harry.Potter1,LOTR2,Green.Mile}"

[27] "{Sixth.Sense,LOTR}"

[28] "{Sixth.Sense,Green.Mile}"

[29] "{LOTR1,LOTR2}"

[30] "{Gladiator,Braveheart}"

[31] "{Gladiator,LOTR,Green.Mile}"

[32] "{LOTR1,Harry.Potter1,LOTR2,Green.Mile}"

[33] "{LOTR,Green.Mile}"

[34] "{Sixth.Sense,Gladiator}"

[35] "{Patriot,Braveheart}"

[36] "{Sixth.Sense,Patriot}"

[37] "{Sixth.Sense,LOTR,Green.Mile}"

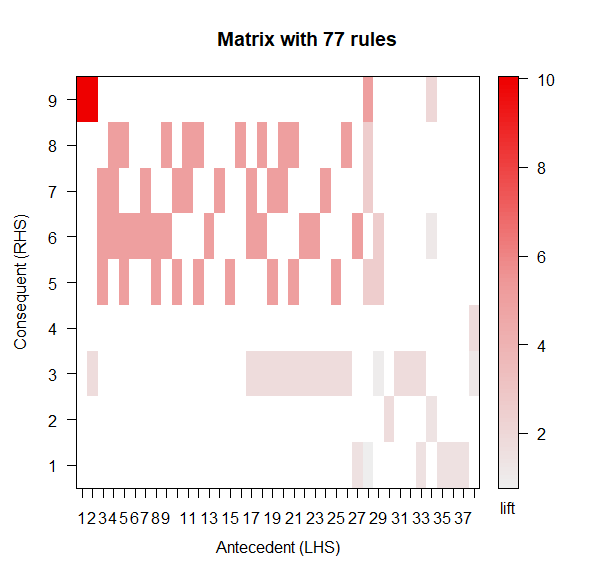
[38] "{Gladiator,Patriot}"

Itemsets in Consequent (RHS)

[1] "{Gladiator}" "{Patriot}" "{Sixth.Sense}" "{Braveheart}"

[5] "{Harry.Potter1}" "{Green.Mile}" "{LOTR2}" "{LOTR1}"

[9] "{LOTR}"



> movies\_rules2 <- apriori(as.matrix(movies[,6:15]),parameter = list(support =0.005, confidence =0.5,minlen=2))

Apriori

Parameter specification:

confidence minval smax arem aval originalSupport maxtime support minlen maxlen

0.5 0.1 1 none FALSE TRUE 5 0.005 2 10

target ext

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 ... [105 rule(s)] done [0.00s].

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

> movies\_rules2

set of 105 rules

> inspect(head(sort(movies\_rules2,by="lift")))

lhs rhs support confidence

[1] {Gladiator,Green.Mile} => {LOTR} 0.1 1.0

[2] {Sixth.Sense,Gladiator,Green.Mile} => {LOTR} 0.1 1.0

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

[4] {Harry.Potter1} => {Harry.Potter2} 0.1 0.5

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

[6] {Green.Mile} => {LOTR} 0.1 0.5

coverage lift count

[1] 0.1 10 1

[2] 0.1 10 1

[3] 0.1 5 1

[4] 0.2 5 1

[5] 0.1 5 1

[6] 0.2 5 1

> inspect(head(sort(movies\_rules2,by="confidence")))

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

> inspect(head(sort(movies\_rules2,by="support")))

lhs rhs support confidence coverage lift count

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

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

[3] {Gladiator} => {Sixth.Sense} 0.5 0.7142857 0.7 1.190476 5

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

[5] {Patriot} => {Sixth.Sense} 0.4 0.6666667 0.6 1.111111 4

[6] {Sixth.Sense} => {Patriot} 0.4 0.6666667 0.6 1.111111 4

> inspect(head(sort(movies\_rules2,by=c("count","lift"))))# max count = 6

lhs rhs support confidence coverage lift

[1] {Patriot} => {Gladiator} 0.6 1.0000000 0.6 1.428571

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

[3] {Gladiator} => {Sixth.Sense} 0.5 0.7142857 0.7 1.190476

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

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

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

count

[1] 6

[2] 6

[3] 5

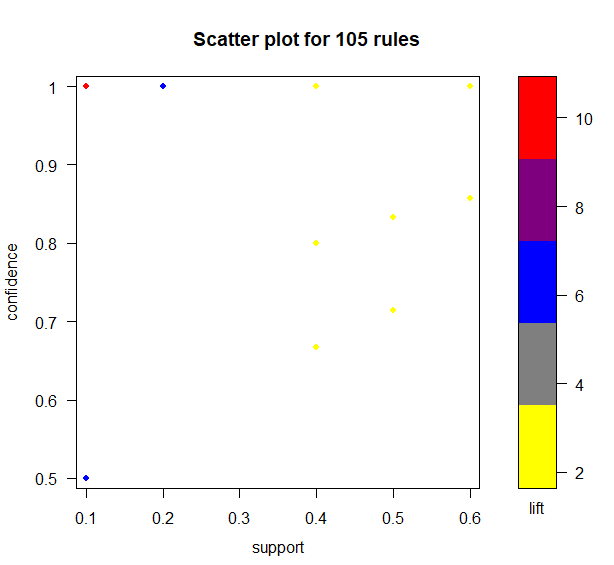
[4] 5

[5] 4

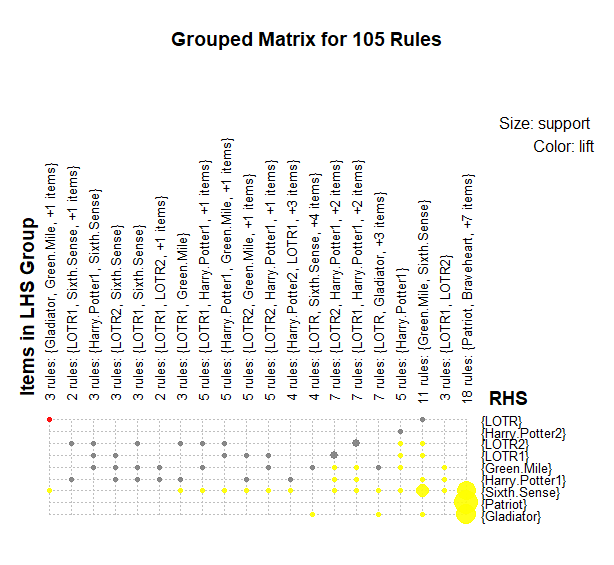
[6] 4

> #visualisation

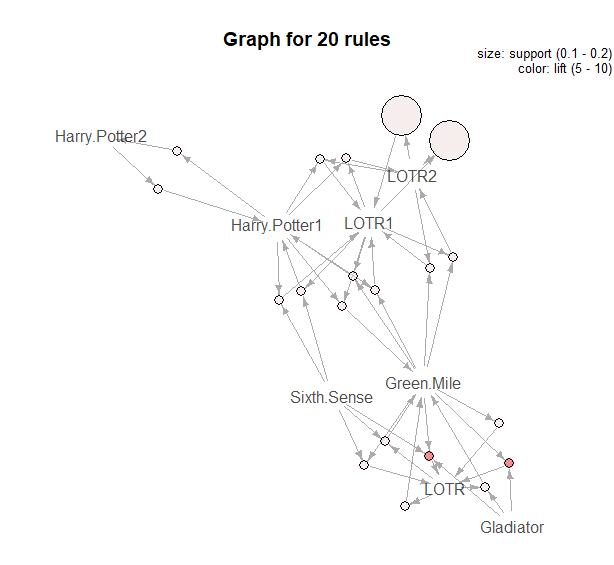
> plot(movies\_rules2,method = "scatterplot",jitter=0,col=colfunction(5))



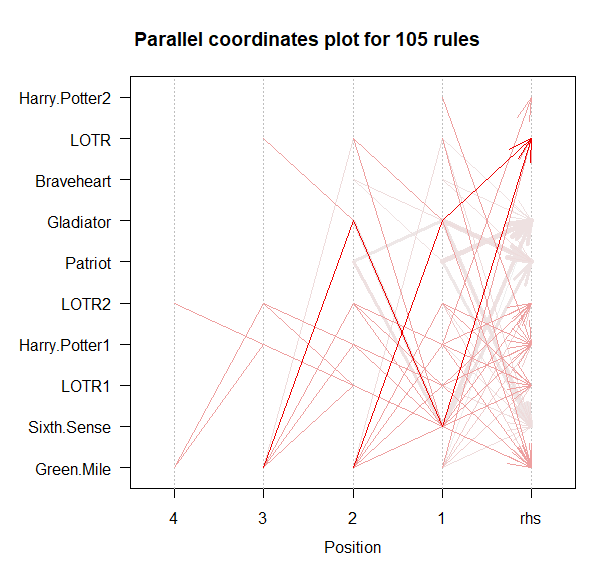
> plot(movies\_rules2,method = "grouped matrix",col=colfunction(5))



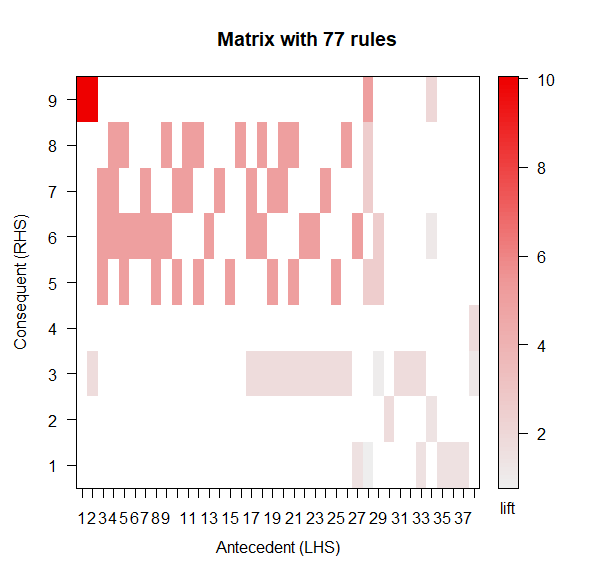
> plot(head(sort(movies\_rules2,by="lift"),n=20),method = "graph")



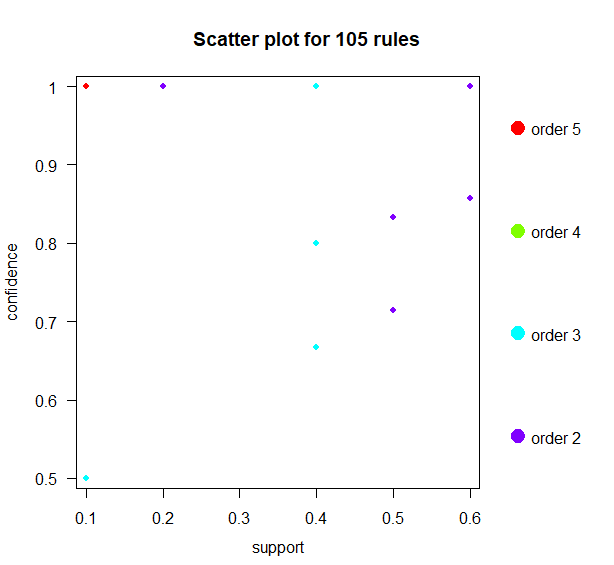
> plot(movies\_rules2,method = "paracoord")



> plot(movies\_rules,method = "matrix")



> plot(movies\_rules2,method = "two-key plot", jitter=0)



> plot(movies\_rules2,method = "matrix",engine = "3d")

