############################## Association Rule #####################################

**#Objective: Prepare rules for the all the data sets**

**#1) Try different values of support and confidence. Observe the change in**

**#number of rules for different support, confidence values**

**#2) Change the minimum length in apriori algorithm**

**#3) Visualize the obtained rules using different plots**

**#Data : my\_movies.csv**

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

install.packages("arules")

library("arules") # Used for building association rules i.e. apriori algorithm

**##Step1 : Data Exploration**

movies<-read.csv('D:\\Shilpa\\Datascience\\Assignments\\Association Rule\\my\_movies.csv')

View(movies)

# making rules using apriori algorithm

# Keep changing support and confidence values to obtain different rules

?apriori

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

**#Step 2: Training a model on the data**

#install.packages("arulesViz")

library("arulesViz") # for visualizing rules

#set better support and confidence levels to learn more rules

**# Building rules using apriori algorithm support=0.02,confidence=0.5**

rules <- apriori(as.matrix(movies[6:15]),parameter=list(support=0.02, confidence = 0.5,minlen=5))

rules

#Apriori

#Parameter specification:

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

#0.5 0.1 1 none FALSE TRUE 5 0.02 5 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 ... [5 rule(s)] done [0.00s].

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

**#Step 3: Evaluating model performance**

# summary of movies association rules

summary(rules)

#set of 5 rules

#rule length distribution (lhs + rhs):sizes

#5

#5

#Min. 1st Qu. Median Mean 3rd Qu. Max.

#5 5 5 5 5 5

#summary of quality measures:

# support confidence coverage lift count

#Min. :0.1 Min. :1 Min. :0.1 Min. :1.667 Min. :1

#1st Qu.:0.1 1st Qu.:1 1st Qu.:0.1 1st Qu.:5.000 1st Qu.:1

#Median :0.1 Median :1 Median :0.1 Median :5.000 Median :1

#Mean :0.1 Mean :1 Mean :0.1 Mean :4.333 Mean :1

#3rd Qu.:0.1 3rd Qu.:1 3rd Qu.:0.1 3rd Qu.:5.000 3rd Qu.:1

#Max. :0.1 Max. :1 Max. :0.1 Max. :5.000 Max. :1

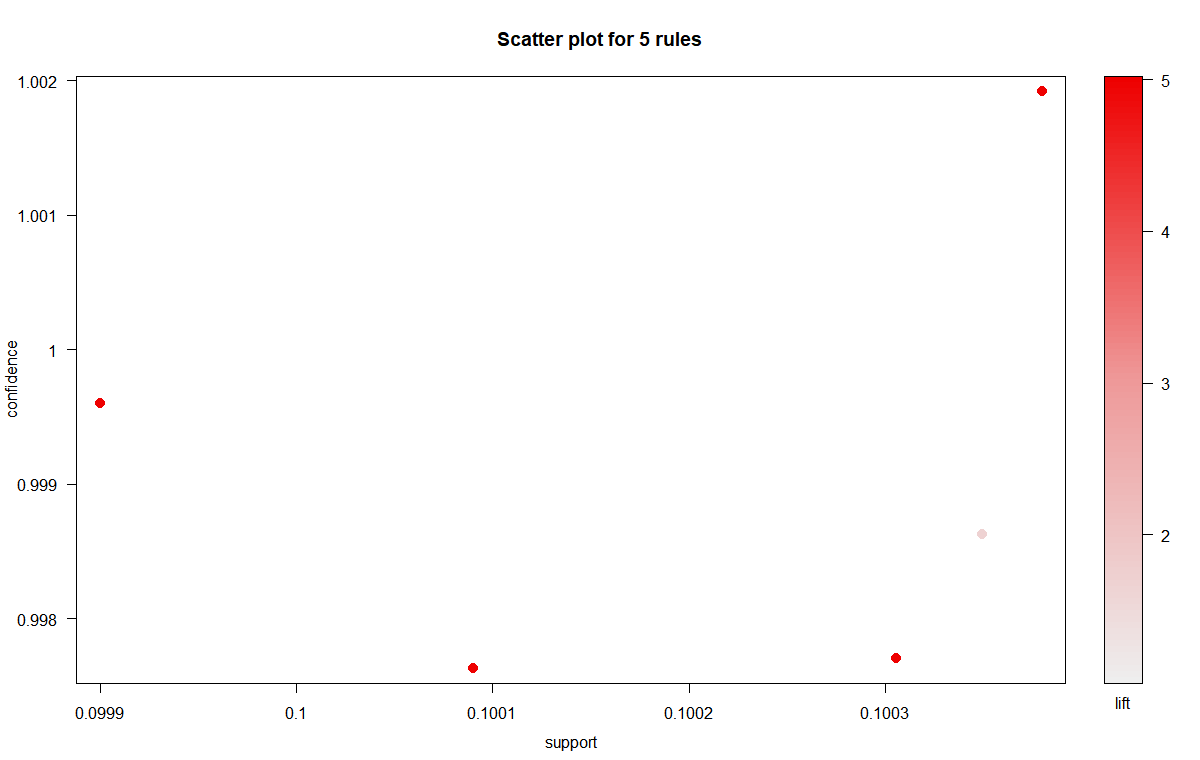
#mining info:

# data ntransactions support confidence

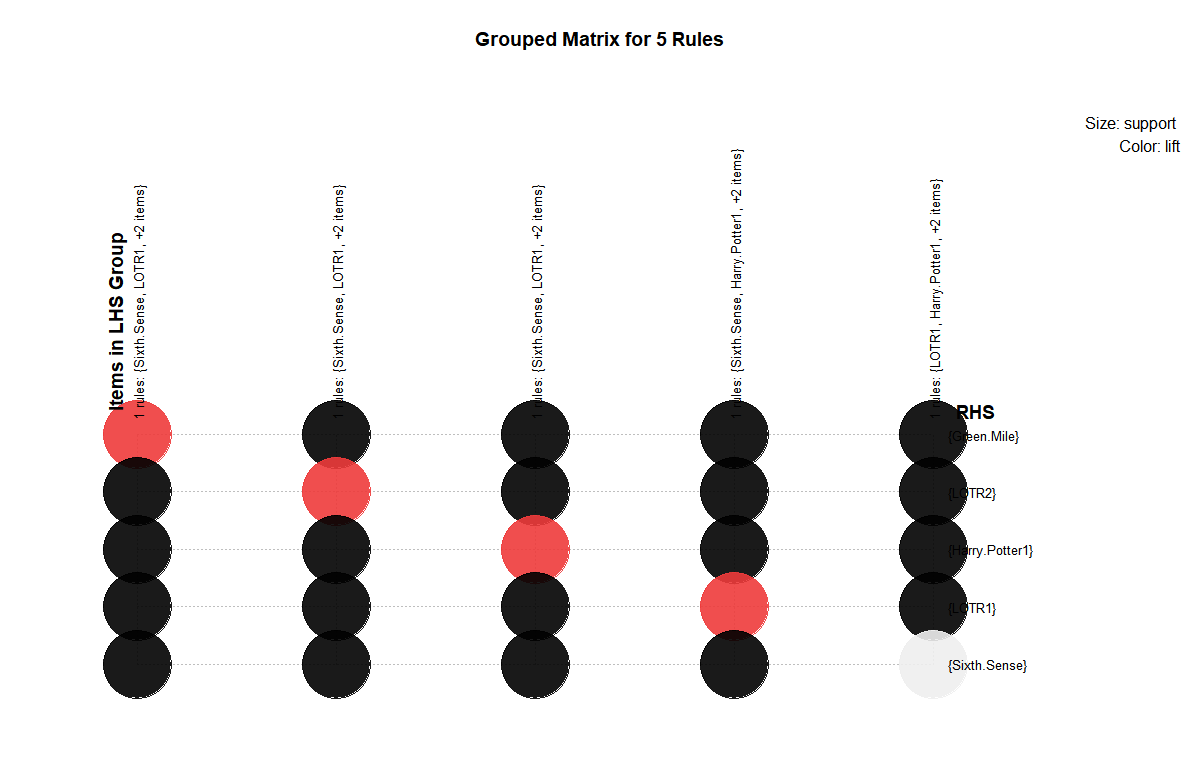
#as.matrix(movies[6:15]) 10 0.02 0.5

**#Step 4: Visualization**

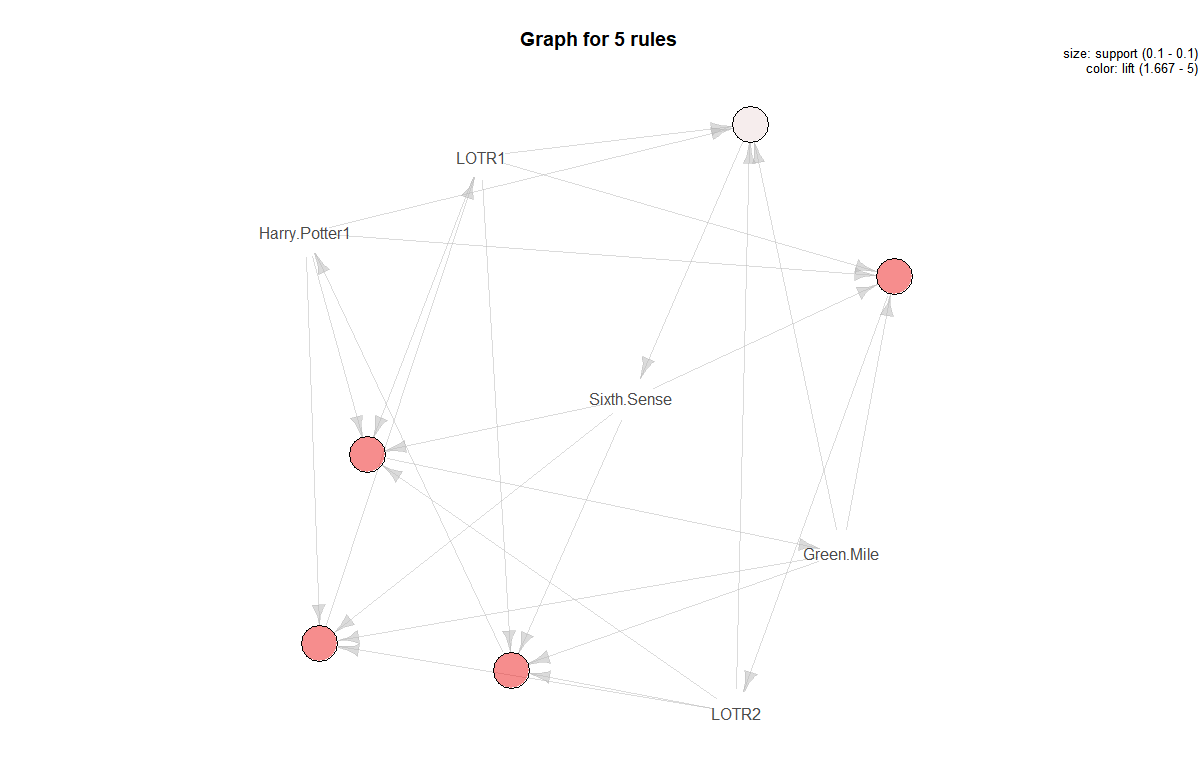
plot(rules,method = "scatterplot")



plot(rules,method = "grouped")



plot(rules,method = "graph")



**#Step 5: Improving model performance**

#sorting movie rules by lift

rules1 <- sort(rules,by="lift")

inspect(rules1[1:4])

#lhs rhs support confidence coverage lift

#[1] {Sixth.Sense,LOTR1,Harry.Potter1,LOTR2} => {Green.Mile} 0.1 1 0.1 5

#[2] {Sixth.Sense,LOTR1,Harry.Potter1,Green.Mile} => {LOTR2} 0.1 1 0.1 5

#[3] {Sixth.Sense,LOTR1,LOTR2,Green.Mile} => {Harry.Potter1} 0.1 1 0.1 5

#[4] {Sixth.Sense,Harry.Potter1,LOTR2,Green.Mile} => {LOTR1} 0.1 1 0.1 5

#count

#[1] 1

#[2] 1

#[3] 1

#[4] 1

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

**#Association with support =0.01,confidence = 0.06,minlen=4**

movie\_rules<-apriori(as.matrix(movies[6:15]),parameter = list(support = 0.01,confidence = 0.06,minlen=4))

#Apriori

#Parameter specification:

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

#0.06 0.1 1 none FALSE TRUE 5 0.01 4 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 ... [29 rule(s)] done [0.00s].

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

**#Step : Evaluating model performance**

# summary of movie\_rules association rules

summary(movie\_rules)

#set of 29 rules

#rule length distribution (lhs + rhs):sizes

#4 5

#24 5

#Min. 1st Qu. Median Mean 3rd Qu. Max.

#4.000 4.000 4.000 4.172 4.000 5.000

#summary of quality measures:

# support confidence coverage lift count

#Min. :0.1 Min. :1 Min. :0.1 Min. : 1.429 Min. :1

#1st Qu.:0.1 1st Qu.:1 1st Qu.:0.1 1st Qu.: 5.000 1st Qu.:1

#Median :0.1 Median :1 Median :0.1 Median : 5.000 Median :1

#Mean :0.1 Mean :1 Mean :0.1 Mean : 4.360 Mean :1

#3rd Qu.:0.1 3rd Qu.:1 3rd Qu.:0.1 3rd Qu.: 5.000 3rd Qu.:1

#Max. :0.1 Max. :1 Max. :0.1 Max. :10.000 Max. :1

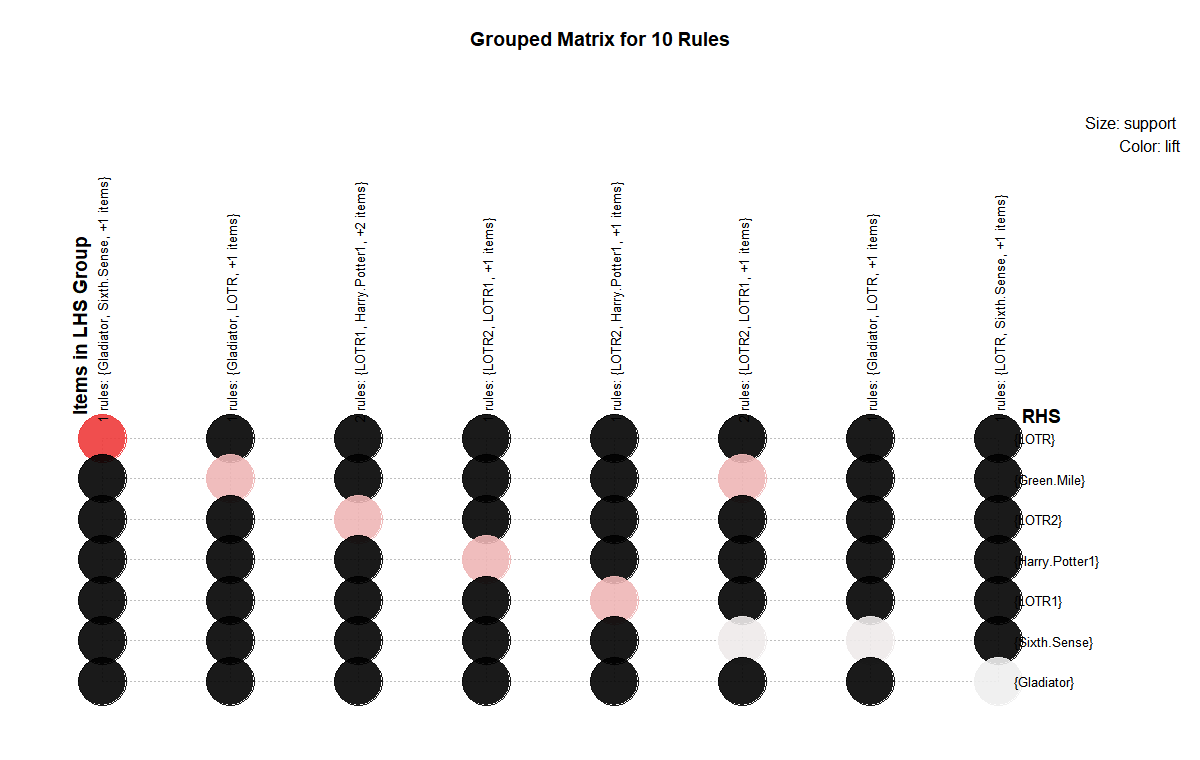
#mining info:

# data ntransactions support confidence

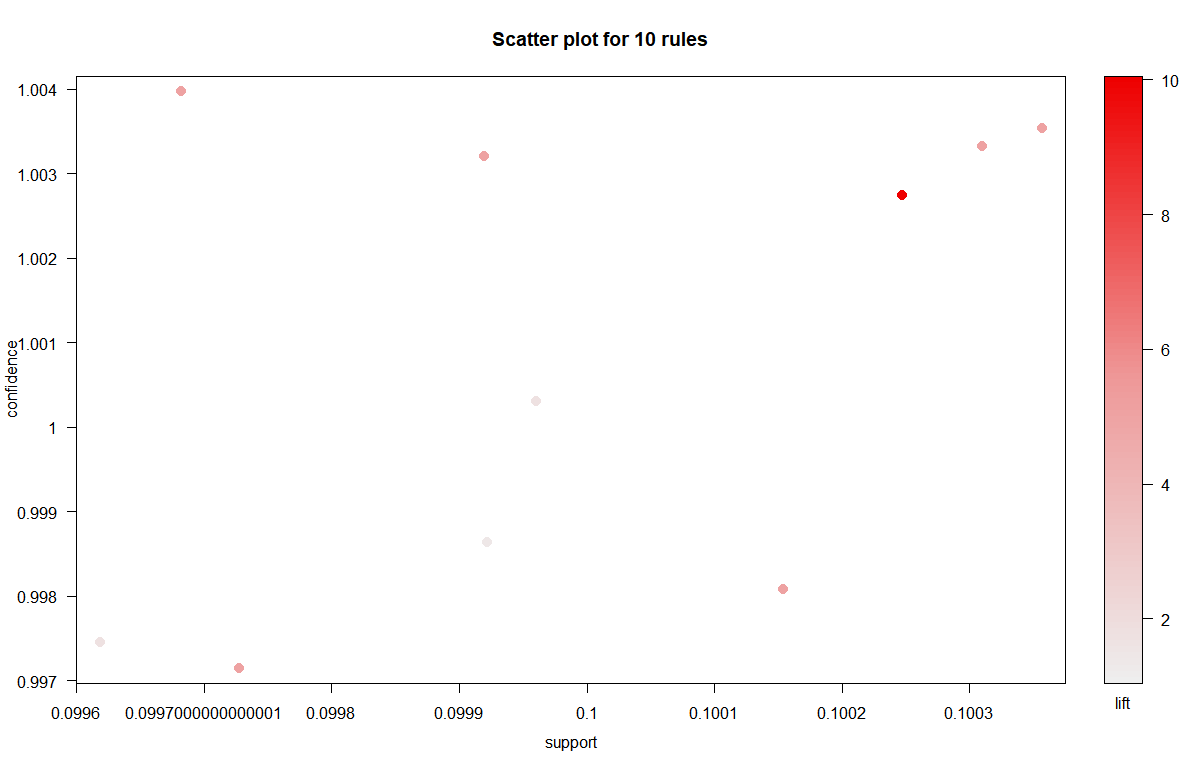
#as.matrix(movies[6:15]) 10 0.01 0.06

**#Step:Visualization**

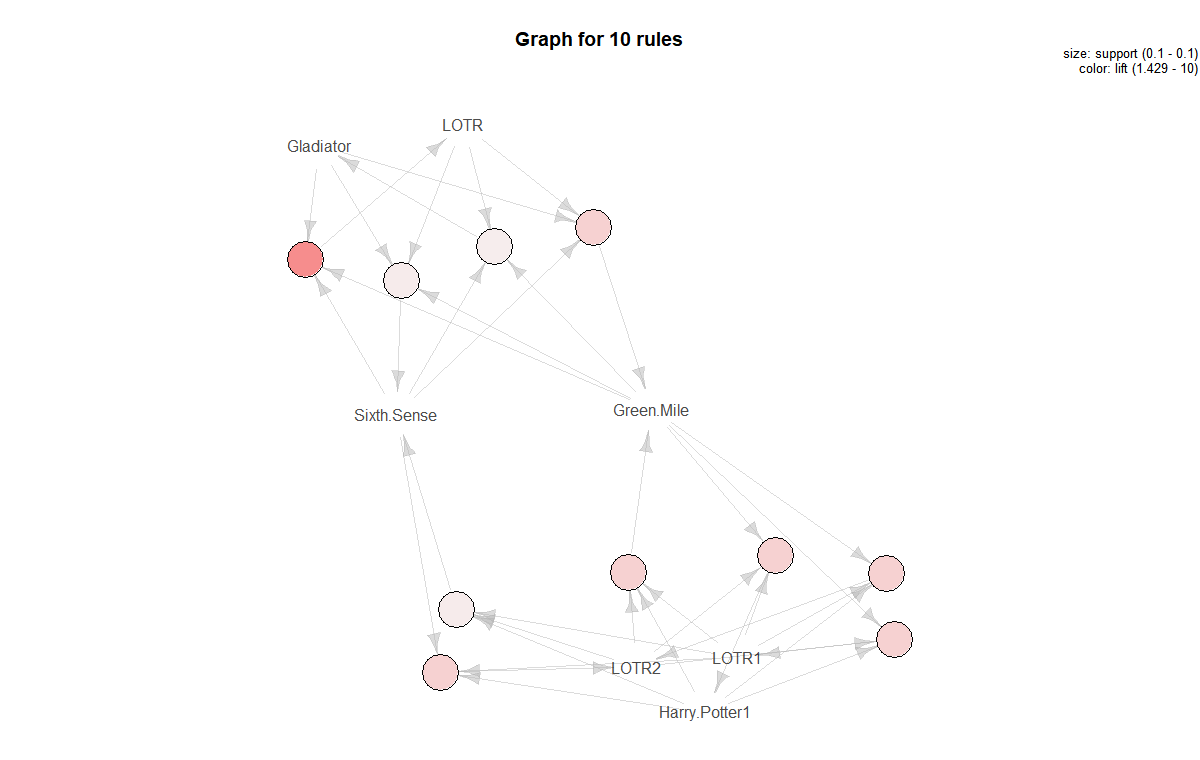
plot(head(sort(movie\_rules), n = 10), method = "grouped", control = list(cex = 0.2))



plot(head(sort(movie\_rules), n = 10), method = "scatterplot", control = list(cex = 0.8))



plot(head(sort(movie\_rules), n = 10), method = "graph")



**#Step: Improving model performance**

#sorting movie rules by lift

rules2 <- sort(movie\_rules,by="lift")

inspect(rules2[1:4])

#lhs rhs support confidence coverage lift count

#[1] {Sixth.Sense,Gladiator,Green.Mile} => {LOTR} 0.1 1 0.1 10 1

#[2] {Sixth.Sense,Gladiator,LOTR} => {Green.Mile} 0.1 1 0.1 5 1

#[3] {LOTR1,Harry.Potter1,LOTR2} => {Green.Mile} 0.1 1 0.1 5 1

#[4] {LOTR1,Harry.Potter1,Green.Mile} => {LOTR2} 0.1 1 0.1 5 1

#The first rule, with a lift of about 10,

#implies that people who watch Sixth.Sense,Gladiator,Green.Mile movies are more

#likely to watch LOTR movie also .

**#writing the rules to a CSV file**

write(rules1, file="movie\_rules.csv",sep=",")