

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
NPTEL
NPTEL ONLINE CERTIFICATION COURSE
Business Analytics & Data Mining Modeling
Using R – Part II
Lecture-03
Association Rules – Part III
With
Dr. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology Roorkee

Business Analytics & Data Mining Modeling Using R - Part II

Lecture-03 Association Rules-Part III

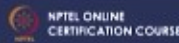


With
Dr. Gaurav Dixit
Department of Management Studies
Indian Institute of Technology Roorkee

Welcome to the course Business Analytics and Data Mining Modeling Using R Part 2, so in the previous lecture we were discussing association rules specifically we were talking about the transaction data formats, so we discussed item list format and also binary matrix formats, so we you know also in R studio we also went through an example how these two formats are different from each other, so we discussed all those details and did an exercise in R studio as well, so let's move forward.

Association Rules

- Rule Selection
 - Select rules which have confidence higher than user specified cut-off value for confidence
- Open RStudio
- Results Interpretation
 - Support of a rule indicates its impact w.r.t database
 - What proportion of transactions is affected?



17

Now as we have completed our discussion on the rule generation stage and also the assessment of rule strength, so next important step is the rule selection, so once the assessment of rule strength has been done, now how do we select the rules for implementation, so as you can see we can select the rules which have confidence value higher than the user specified cut off value for confidence, so again here also we require a certain input from user for selecting the rules, association rules, so any rule which is having higher confidence value, higher value then user specified confidence level, so those rules can be selected.

The screenshot shows the RStudio interface with the following R code in the editor:

```
13-assoR.R  
47 iL=as(CoverColPC, "transactions")  
18 inspect(iL)  
19 image(iL)  
20 summary(iL)  
21 iM=as(iL, "matrix"); iM  
22  
23 iLabels=c("red","white","blue","orange","green","yellow")  
24 iL=encode(CoverColPC, iLabels)  
25 inspect(iL1)  
26 image(iL1)  
27 iM1=as(iL1, "matrix"); iM1  
28  
29 # Convert transactions to transaction ID lists  
30 inspect(as(iL1, "tidlists"))  
31  
32 # 2. Binary matrix format  
291 (Top level):
```

The console output shows the binary matrix format:

```
> iM1=as(iL1, "matrix"); iM1  
red white blue orange green yellow  
[1,] TRUE TRUE FALSE FALSE TRUE FALSE  
[2,] FALSE TRUE FALSE TRUE FALSE FALSE  
[3,] FALSE TRUE TRUE FALSE FALSE FALSE  
[4,] TRUE TRUE FALSE TRUE FALSE FALSE  
[5,] TRUE FALSE TRUE FALSE FALSE FALSE  
[6,] FALSE TRUE TRUE FALSE FALSE FALSE  
[7,] FALSE TRUE FALSE TRUE FALSE FALSE  
[8,] TRUE TRUE TRUE FALSE TRUE FALSE  
[9,] TRUE TRUE TRUE FALSE TRUE FALSE  
[10,] TRUE TRUE TRUE FALSE TRUE FALSE
```

The Environment pane shows the objects: CoverColPC (List of 10), iL (Formal class transactions), iL1 (Formal class itemMatrix), iM (logi [1:10, 1:6] FALSE FALS...), and iM1 (logi [1:10, 1:6] TRUE FALS...).

The Plots pane shows a heatmap visualization of the binary matrix. The y-axis is labeled 'Elements (Rows)' and ranges from 1 to 10. The x-axis is labeled 'Items (Columns)' and ranges from 1 to 6. The heatmap shows a pattern of black and white squares representing the binary values in the matrix.

So let's open R studio and go through an exercise to understand this further, so in the previous lecture we were able to create this binary matrix format in the particular fashion that we

wanted, particular configuration that we wanted, few more things that would be of use later on for example we can convert these transactions to, transaction ID list, so for that we can use this particular again as function and you can see IL1 that particular variable is been used and then we can have this TID, transaction ID list, TIDlist as the format and let's run this, if we run this code you can see this is the format, so this is transaction ID you know list format, so here you can see the items all distinct items, all arranged in each row, so we have 6 distinct items, so we have 6 rows red, white, blue, orange, green and yellow, and for each of these items the transaction ID's where these items are occurring, so for example first row we have red, item red and the transaction ID's where this particular item is present, so you can see transaction ID 1, 4, 5, 8, 9, similarly white is the most frequent you know item, and here you see it is out of all 10 transaction, 8 transaction it is present. Similarly for blue, orange, green and yellow we can see the transaction ID's where these particular items are present, so this is the transaction ID you know list format.

The screenshot shows an RStudio session with the following components:

- Source Editor:** Contains R code for converting transactions to a binary matrix and then to transaction ID lists.
- Console:** Shows the output of the R code, displaying a list of transaction IDs for each item.
- Environment:** Lists the objects in the global environment, including 'il1', 'im', and 'im1'.
- Heatmap:** A 10x6 heatmap showing the presence of items in transactions. The y-axis is labeled 'Elements (Rows)' and the x-axis is labeled 'Items (Columns)'. The items are red, white, blue, orange, green, and yellow.

```

25 inspect(il1)
26 image(il1)
27 im1=as(il1, "matrix"); im1
28
29 # Convert transactions to transaction ID lists
30 inspect(as(il1, "tidlists"))
31
32 # 2. Binary matrix format
33 imn=apply(im1, c(1,2), as.numeric)
34 dimnames(imn)=list(c(1:nrow(im1)), itemLabels(il1)); imn
35
36 # Itemsets with support(count)>=2
37 isets=ec1at(il1, parameter = list(support=0.2), control = list(verbose=#))
38 isets1=inspect(isets)
39 isets1=isets1[order(size(isets)),]; isets1
40
321 (Top level)

```

```

> # Convert transactions to transaction ID lists
> inspect(as(il1, "tidlists"))
items transactionIDs
1 red {1,4,5,8,9}
2 white {1,2,3,4,6,7,8,9}
3 blue {3,5,6,8,9}
4 orange {2,4,7}
5 green {1,8}
6 yellow {10}

```

Let's move forward, so now what we can do is we can create a, so earlier we the matrix that we had created the binary matrix format that we had created it was logical, so we had true and false to indicate whether a particular item is present in a particular row that means a particular transaction, so we can change this a bit for example, we can make it this instead of using true false value, we can start using 0 and 1, so for that you can see that we are using apply function and the function that we are going to apply here is as dot numeric, so all the logical values are now going to be coerced into numeric value, so true and false are going to be converted into 0's and 1's, so let's run this code.

The screenshot shows RStudio with the following R code in the editor:

```

25 inspect(iL1)
26 image(iL1)
27 iM1=as(iL1, "matrix"); iM1
28
29 # Convert transactions to transaction ID lists
30 inspect(as(iL1, "tidLists"))
31
32 # 2. Binary matrix format
33 iMn=apply(iM1, c(1,2), as.numeric)
34 dimnames(iMn)=list(c(1:nrow(iM1)), itemLabels(iL1)); iMn
35
36 # Itemsets with support(count)>=2
37 isets=eclat(iL1, parameter = list(support=0.2), control = list(verbose=F))
38 isets1=inspect(isets)
39 isets1=isets1[order(size(isets)),]; isets1
40
41 (Top level)

```

The console output shows the binary matrix format:

```

> dimnames(iMn)=list(c(1:nrow(iM1)), itemLabels(iL1)); iMn
red white blue orange green yellow
1 1 1 0 0 1 0
2 0 1 0 1 0 0
3 0 1 1 0 0 0
4 1 1 0 1 0 0
5 1 0 1 0 0 0
6 0 1 1 0 0 0
7 0 1 0 1 0 0
8 1 1 1 0 1 0
9 1 1 1 0 0 0
10 1 1 1 0 0 0

```

The Environment pane shows the objects created:

- Global Environment
- CoverColPC: List of 10
- iL: Formal class transactions
- iL1: Formal class itemMatrix
- iM: logi [1:10, 1:6] FALSE FAL...
- iM1: logi [1:10, 1:6] TRUE FALS...
- iMn: num [1:10, 1:6] 1 0 0 1 1 ..

The Viewer pane displays a heatmap of the binary matrix with 'Elements (ROWS)' on the y-axis (1-10) and 'Items (Columns)' on the x-axis (1-6). The heatmap shows black squares representing 1s and white squares representing 0s.

Now at the same time we'll also like to change the dimensions here, so these dimensions that we had created earlier, that we are going to use, now let's look at this format, so now you can see that all the values, all the cells are being indicated as 1 or 0, so if that particular item is present in that particular transactions so it is 1, otherwise it is 0 if it is absence then it is 0, so you can see the item levels or also like just like we created in the last lecture red, white, blue in that particular order and we have total 10 transactions.

The screenshot shows RStudio with the following R code in the editor:

```

40
41 # Minimum support=20% & minimum confidence=70%
42
43 (Top level)

```

The console output shows the binary matrix format:

```

2 0 1 0 1 0 0
3 0 1 1 0 0 0
4 1 1 0 1 0 0
5 1 0 1 0 0 0
6 0 1 1 0 0 0
7 0 1 0 1 0 0
8 1 1 1 0 1 0
9 1 1 1 0 0 0
10 0 0 0 0 0 1

```

The Environment pane shows the objects created:

- Global Environment
- CoverColPC: List of 10
- iL: Formal class transactions
- iL1: Formal class itemMatrix
- iM: logi [1:10, 1:6] FALSE FAL...
- iM1: logi [1:10, 1:6] TRUE FALS...
- iMn: num [1:10, 1:6] 1 0 0 1 1 ..

The Viewer pane displays the output of the `inspect` function:

Display Associations and Transactions in Readable Form

Description

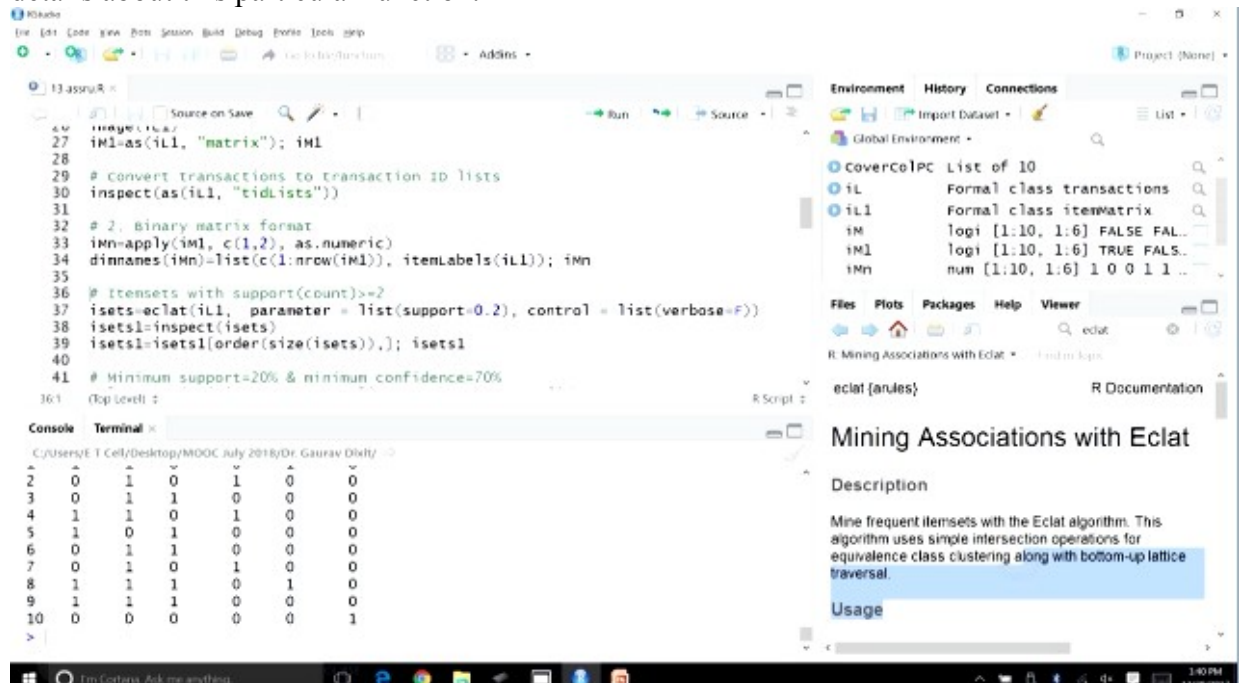
Provides the generic function `inspect` and S4 methods to display associations and transactions plus additional information formatted for online inspection.

Usage

```
inspect(x, ...)
```

So now this format that we have just created now this is binary matrix format and the values are also being indicated using 1's and 0's, if we want to identify item sets which are having support or count value greater than or equal to 2, we can do so using this particular function Eclat, so

we are interested in finding more details about Eclat what it does, so it is actually it will be able to help us in creating the rules using that particular transaction dataset, so let's look at the more details about this particular function.



So Eclat as you can see here, so this particular function, so you can see the description mine frequent item sets with the Eclat algorithm, this algorithm uses simple interaction, intersection operations for equivalence class clustering along with bottom up lattice traversal, so essentially this particular algorithm can be used to find to mine frequent item sets, so this is the function that we are going to use here, so let's run this code, so other arguments of this code if you see that we are passing the IL1 which is the you know, the item list format that we have matrix format that binary matrix format that we have, then you can see we have specified the support as 0.2 right, because we want the, we want to identify all the item sets having you know support greater than 2 and other parameters are also specified, so let's run this. Once this is done we can use the inspect function to analyze to see the results, so you can see here, you have the list now, so all the item sets which are having support greater than 0.2, you can see here support and count, 0.3, 0.2 in this fashion, we are able to see, so all the item, all the sets which are having this, which are having support or count value greater than this can be seen here.

```

27 iM1=as(iL1, "matrix"); iM1
28
29 # Convert transactions to transaction ID lists
30 inspect(as(iL1, "tidlists"))
31
32 # 2. Binary matrix format
33 iMn=apply(iM1, c(1,2), as.numeric)
34 dimnames(iMn)=list(c(1:nrow(iM1)), itemLabels(iL1)); iMn
35
36 # Itemsets with support(count)>=2
37 isets=eclat(iL1, parameter = list(support=0.2), control = list(verbose=F))
38 isets1=inspect(isets)
39 isets1=isets1[order(size(isets)),]; isets1
40
41 # Minimum support=20% & minimum confidence=70%
42 rules1=apriori(iL1, parameter=list(support=0.2, confidence=0.7))
39:12 (Top level)

```

	items	support	count
[1]	{white,orange}	0.3	3
[2]	{red,white,green}	0.2	2
[3]	{white,green}	0.2	2
[4]	{red,green}	0.2	2
[5]	{red,white,blue}	0.2	2
[6]	{white,blue}	0.4	4
[7]	{red,blue}	0.3	3
[8]	{red,white}	0.4	4
[9]	{white}	0.8	8
[10]	{red}	0.5	5

Now if we want to arrange this list in the order of count, in the order of support so we can do this, so we can see this code if we run this we'll get the list as per the order so you can see white is the most frequent you know item set, so this single item this is coming 8 times, this is present in the database, then red 5 times, then blue 5 times, in this fashion you can see the databases presented here.

```

32 # 2. Binary matrix format
33 iMn=apply(iM1, c(1,2), as.numeric)
34 dimnames(iMn)=list(c(1:nrow(iM1)), itemLabels(iL1)); iMn
35
36 # Itemsets with support(count)>=2
37 isets=eclat(iL1, parameter = list(support=0.2), control = list(verbose=F))
38 isets1=inspect(isets)
39 isets1=isets1[order(size(isets)),]; isets1
40
41 # Minimum support=20% & minimum confidence=70%
42 rules1=apriori(iL1, parameter=list(support=0.2, confidence=0.7))
43 rules1.sorted=sort(rules1, by="lift")
44 inspect(rules1.sorted)
45 round(quality(rules1.sorted), digits = 2)
46 summary(rules1.sorted)
47
42:28 (Top level)

```

	items	support	count
[9]	{white}	0.8	8
[10]	{red}	0.5	5
[11]	{blue}	0.5	5
[12]	{green}	0.2	2
[13]	{orange}	0.3	3
[1]	{white,orange}	0.3	3
[3]	{white,green}	0.2	2
[4]	{red,green}	0.2	2
[6]	{white,blue}	0.4	4

Now this particular ordering is if you look at this particular ordering is based on the size of item sets, so the first you know the item sets having just singular item or coming first then followed by the item sets which are having 2 item sets, and then 3 and 4.

Now if we want to run Apriori algorithm here, so we have this a Apriori function, so we want to mine the you know frequent item sets we can use this particular function, Apriori we are passing IL1 this binary matrix format, you can see the parameter support is specified as 0.2, and the confidence is specified as 0.7, so the user specified minimum support level is 20%, and the user specified minimum confidence level is 70%, so we can use the Apriori function and we can pass on the first order and being the matrix, and then the support and confidence value, so we'll get the association rules here, so let's run this.

The screenshot shows the RStudio interface with the following components:

- Script Editor:** Contains R code for running the Apriori algorithm. The code includes:


```

33 itemMatrix = as.matrix(c(1,1,2), as.numeric)
34 dimnames(itemMatrix) = list(c(1:nrow(itemMatrix)), itemLabels(itemMatrix))
35
36 # Itemsets with support(count) >= 2
37 itemsets = eclat(itemMatrix, parameter = list(support = 0.2), control = list(verbose = F))
38 itemsets = inspect(itemsets)
39 itemsets = itemsets[order(size(itemsets)),]; itemsets
40
41 # Minimum support = 20% & minimum confidence = 70%
42 rules = apriori(itemMatrix, parameter = list(support = 0.2, confidence = 0.7))
43 rules.sorted = sort(rules, by = "lift")
44 inspect(rules.sorted)
45 round(quality(rules.sorted), digits = 2)
46 summary(rules.sorted)
47
48 # Find redundant rules
49 (topLevel =
      
```
- Environment:** Shows objects created during the process:
 - `itemMatrix`: `logi [1:10, 1:6] TRUE FALSE...`
 - `itemMatrix`: `num [1:10, 1:6] 1 0 0 1 1 ..`
 - `itemsets`: `Formal class itemsets`
 - `itemsets.sorted`: `13 obs. of 3 variables`
 - `rules`: `Formal class rules`
 - `rules.sorted`: `Formal class rules`
- Console:** Displays the output of the `summary(rules.sorted)` command as a table:

lhs	rhs	support	confidence	lift	count
[1] {green}	=> {red}	0.2	1.0	2.00	2
[2] {white,green}	=> {red}	0.2	1.0	2.00	2
[3] {green}	=> {white}	0.2	1.0	1.25	2
[4] {orange}	=> {white}	0.3	1.0	1.25	3
[5] {red,green}	=> {white}	0.2	1.0	1.25	2
[6] {}	=> {white}	0.8	0.8	1.00	8
[7] {blue}	=> {white}	0.4	0.8	1.00	4
[8] {red}	=> {white}	0.4	0.8	1.00	4
- Viewer:** Shows the documentation for the `eclat` function, titled "Mining Associations with Eclat".

Now let's sort this rules also by their lift value, so after the sorting we would like to see, so let's inspect the sorted rules, so now you can see 8 rules are there, and you can see they have been ordered with the lift values, so first rule green, if you know, if the green color mobile cover is purchased then red color mobile cover is also purchased, so this is having the highest lift value of 2. Then the second association rules where if the white and green mobile covers are purchased then red is also purchased, this particular association rule is also having the lift value of 2, so similarly we can look at others. So you can see here all the association rules which are having greater than 1 lift value they have been shown here, so as we talked about for our you know a particular association rules to be useful it should have a lift value of greater than 1, right.

The screenshot displays the RStudio interface. The script editor contains R code for performing Eclat mining. The console shows the execution of the code, resulting in a summary of quality measures for the rules. The R documentation pane on the right shows the 'eclat' function documentation, including a description of the algorithm and a usage section.

```

30 # itemsets with support <= 0.2
37 isets=eclat(il1, parameter = list(support=0.2), control = list(verbose=#))
38 isets=inspect(isets)
39 isets1=isets[order(size(isets)),]; isets1
40
41 # Minimum support=20% & minimum confidence=70%
42 rules1=apriori(il1, parameter=list(support=0.2, confidence=0.7))
43 rules1.sorted=sort(rules1, by="lift")
44 inspect(rules1.sorted)
45 round(quality(rules1.sorted), digits = 2)
46 summary(rules1.sorted)
47
48 # Find redundant rules
49 subset1=is.subset(rules1.sorted); subset1
50 subset1[lower.tri(subset1, diag=T)]=NA; subset1
51 rdd1=colSums(subset1, na.rm=T) >= 1; rdd1
481 (Top Level)

```

Console Output:

```

1 2 3
1 5 2

  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1.000  2.000   2.000   2.125  2.250   3.000

summary of quality measures:
  support    confidence      lift      count
Min.   :0.2000   Min.   :0.800   Min.   :1.000   Min.   :2.000
1st Qu.:0.2000   1st Qu.:0.800   1st Qu.:1.000   1st Qu.:2.000
Median :0.2500   Median :1.000   Median :1.250   Median :2.500

```

R Documentation: Mining Associations with Eclat

Description: Mine frequent itemsets with the Eclat algorithm. This algorithm uses simple intersection operations for equivalence class clustering along with bottom-up lattice traversal.

Usage: eclat (arules)

There are few other functions which can be used, if we are interested in looking at the matrix number, then quality function can be used, then few more details if we are interested we can look at the summary function, so we would be able to see few more details, for example if you just you know look at few more details here, you can see set of 8 rules, rule length distribution sizes LHS plus RHS is given there in the summary output, then we can look at these quintiles values, median, first quintile, third quintile, although it values are also there then the summary of quality measures are also there.

Then it might so happen that some of these rules, some of these rules might be subset of some other rules, so we can always eliminate the subset rules so redundant rules can be identified, and they can be removed from the output, and then we'll have just the rules which are you know, redundant rules are removed and we have just the meaningful rules, so no reputation, so this is the code that can be use, so you can see we are using is dot subset function, so this function is being used to identify to its spot the rules which are subset of some other rules, right, so once this is done you can see because this output is going to be a matrix and then this is going to be you know lower triangle we are terming as an NA, so that the reputation is allowed, reputation is avoided and then we are summing all these cells, so if the sum is greater than 1, then probably you know that particular rule is subset to more than 1 rule, so therefore it has to be removed, so let's run this code.

The screenshot shows an R Studio session. The script editor contains the following R code:

```

41 # minimum support=0.2 & minimum confidence=0.5
42 rules1=apriori(i1, parameter=list(support=0.2, confidence=0.7))
43 rules1.sorted=sort(rules1, by="lift")
44 inspect(rules1.sorted)
45 round(quality(rules1.sorted), digits = 2)
46 summary(rules1.sorted)
47
48 # Find redundant rules
49 subset1=is.subset(rules1.sorted); subset1
50 subset1[lower.tri(subset1, diag=T)]=NA; subset1
51 rdd1=colSums(subset1, na.rm=T) >= 1; rdd1
52 which(rdd1)
53 # Prune them
54 rules1.pruned=rules1.sorted[!rdd1]
55 inspect(rules1.pruned)
56
581 (Top level)

```

The console shows the output of the `which(rdd1)` command, displaying a matrix of TRUE values:

```

      {red,green} {red,white,green} {white,green} {white,orange}
      TRUE      TRUE      TRUE      TRUE
{red,white,green} {white} {white,blue} {red,white}
      TRUE      TRUE      TRUE      TRUE
> which(rdd1)
      {red,green} {red,white,green} {white,green} {white,orange}
      1           2           3           4
{red,white,green} {white} {white,blue} {red,white}
      5           6           7           8

```

The sidebar on the right shows the 'Mining Associations with Eclat' documentation page, which includes a description of the algorithm and a usage section.

So a particular, so you can see the output as well, so this is the matrix and if a particular rule is subset of any other rule so it will have the finally when we sum that up, it will have the value equal to or greater than 1, so let's compute these values, so now which of these rules you can see, so these are some of the subsets, these are some of the item sets which are, so if true indicating that these are subsets to some other rules, right, so we can find out the indices of these rules, so you can see here, and then we can pruned them, so we'll just have to remove these rules and we'll be left with, we'll be left with remaining rules, so in this fashion we can create rules and identify and eliminate any redundant rules if there are any.

Let's comeback to our discussion here, so how do we interpret the results, the exercise that we have just completed in R studio, how do we interpret some of these results, so first let's start with the support, so support of a rule indicates its impact with respect to database, so what proportion of transaction is affected, so that is the sense that we get by support of a rule, if higher the support that means in the impact of that particular rule is higher, and we'll also get the idea about the proportion of transactions that are being effected by a particular rules, so higher support is always preferable for the business and operational purposes.

Association Rules

- Results Interpretation
 - Lift ratio indicates efficiency of rule in finding consequents in comparison to random selection
 - Confidence indicates the rate of finding consequents
- Statistical significance and chance occurrence of rules
 - How sure are we about the meaningfulness of the rules?
 - Are we ending up with associations which are just chance occurrences?



If you look at the lift ratio then the lift ratio indicates the efficiency of a rule in finding consequence in comparison to random selection, so if the lift ratio value is higher then of course that rule would be consider to be more efficient in terms of finding the consequence, so whenever you know antecedent is present it is more likely that consequent is also going to be present, so that efficiency of a particular rule is going to be indicated or captured by lift ratio, so we will always like to go for, go for association rule which is having higher lift ratio.

Now if we look at the confidence matrix, so this particular matrix indicates the rate of finding consequence, so as we understood from the formula itself so out of all the transactions where antecedent item set is present, what is the proportion of you know item set where the consequent is going to be present as well, so in a sense we are getting the rate of finding the consequents, so in this fashion for a particular rule, so always if the value of confidence is higher than probably the rate of finally consequent is higher, so we would like to prefer a rule which is having higher support value, higher confidence value and higher lift ratio.

Now there is always question mark on data mining techniques because of them being data model and a lack of statistical influence, so whether these rules that we are trying to generate through an automated method whether they have some statistical significance or how do we ensure that they are not out of just chance occurrence, so this is always going to be a question mark on this, so how sure are we about the meaningfulness of these rules, so are we ending up with associations which are just chance occurrences, so how do we clear some of these questions, so that is also important so let's open R studio and we'll try to understand this particular aspect through an exercise.

So what we'll do, we'll create a you know randomly generated data, and we'll mine that data and see whether there are any meaningful association that are coming out of this randomly generated data, so since the data has been generated randomly and still if we are able to you know identify certain rules which are having high confidence, high support, and high lift ratio value then the chance occurrence is a serious problem in that case.

The screenshot shows an RStudio session with the following code and output:

```

53 # Prune them
54 rules1.pruned=rules1.sorted[!rdd1]
55 inspect(rules1.pruned)
56
57 # Chance occurrences?
58 # Example: Randomly generated data
59 # #items=9 & #transactions=50
60 rdtrans=random.transactions(nItems = 9, nTrans = 50, method = "independent",
61                             iProb = 1/9, lambda=3)
62
63
64 inspect(rdtrans)
65 size(rdtrans)
66
67 # Minimum support=4% & minimum confidence=70%
68 rules2=apriori(rdtrans, parameter=list(support=0.04, confidence=0.7))
69
70
71

```

The console output shows the results of the `random.transactions` function and the `apriori` function:

```

> which(rdd1)
      TRUE      TRUE      TRUE      TRUE
{red,green} {red,white,green} {white,green} {white,orange}
      1          2          3          4
{red,white,green} {white} {white,blue} {red,white}
      5          6          7          8

```

The RStudio interface also shows the Environment pane with the following objects:

- Global Environment
- iMn: num [1:10, 1:6] 1 0 0 1 1 ...
- isets: Formal class 'itemsets'
- isets1: 13 obs. of 3 variables
- rules1: Formal class 'rules'
- rules1.pru.: Formal class 'rules'
- rules1.sor.: Formal class 'rules'

The R Documentation pane shows the description of the `Eclat` function:

Mining Associations with Eclat

Description

Mine frequent itemsets with the Eclat algorithm. This algorithm uses simple intersection operations for equivalence class clustering along with bottom-up lattice traversal.

Usage

So let's go through an exercise, so you can see a chance occurrences whether a particular, whether association rules can fall into this trap, chance occurrences trap because of it being a data driven process, so in this case we are trying to randomly generate this data, the number of items that we are going to consider is 9, so these are the distinct number of items are 9, and the number of transaction that we are going to generate are 50, so the function that we are going to use is `random.transactions`, so these function `random.transactions` will create these 50 transactions using 9 distinct items, so you can see the first argument is number of items that is 9, then the number of transaction that is 50, and method you can see is `independent`, you know, because we want this to be randomly generated data, you know in built association will be you know these transactions, right, so probability value we are giving equal probability value for you know all of these items, so you can see $1/9$ is the probability value for all these items, so they have equal chance of occurrence these items have equal chance of occurrence, so let's run this code. More details about this particular function `random.transactions` you can always visit the help section and find out.

```

54 rules1.pruned=rules1.sorted[!rdd1]
55 inspect(rules1.pruned)
56
57 - #####
58 # Chance occurrences?
59 # Example: Randomly generated data
60 # #items=9 & #transactions=50
61 rdtrans=random.transactions(nItems = 9, nTrans = 50, method = "independent",
62                             iProb = 1/9, lambda=3)
63
64 inspect(rdtrans)
65 size(rdtrans)
66
67 # Minimum support=4% & minimum confidence=70%
68 rules2=apriori(rdtrans, parameter=list(support=0.04, confidence=0.7))
69
70

```

```

[34] {item2,item5,item7,item8} trans34
[35] {item1,item2,item6,item9} trans35
[36] {item4,item6,item7,item9} trans36
[37] {item1,item2,item5,item7} trans37
[38] {item3} trans38
[39] {item2,item4,item5,item7} trans39
[40] {item1,item2,item4} trans40
[41] {item2,item3,item7,item9} trans41
[42] {item1,item2} trans42
[43] {item1,item2,item3,item5,item6,item7,item9} trans43
[44] {item7} trans44

```

So once this is done, let's look at the, let's inspect the database, so you can see 50 transactions here, now let's scroll, now you can see we have the items so each transaction, each row is representing a transaction and first we have the items that are part of the transactions, so first transaction we have just the item 4, and the transaction ID is trans1, then the second transaction we have these items, item 3, item 5, 7, and the transaction ID is transaction 2, you can see that all these 50 transactions different item have been randomly selected depending on the parameters that we have passed, so these are the randomly generated transactions, now what we'll do is we will go about doing applying association rules mining, using this particular database, and find out whether chance occurrence has a role in this process or not.

```

64 inspect(rdtrans)
65 size(rdtrans)
66
67 # Minimum support=4% & minimum confidence=70%
68 rules2=apriori(rdtrans, parameter=list(support=0.04, confidence=0.7))
69 inspect(head(rules2, 9))
70

```

```

[45] {item4} trans45
[46] {item2,item6,item8} trans46
[47] {item1,item2} trans47
[48] {item1,item5,item7,item8,item9} trans48
[49] {item2,item4,item7} trans49
[50] {item6,item8} trans50

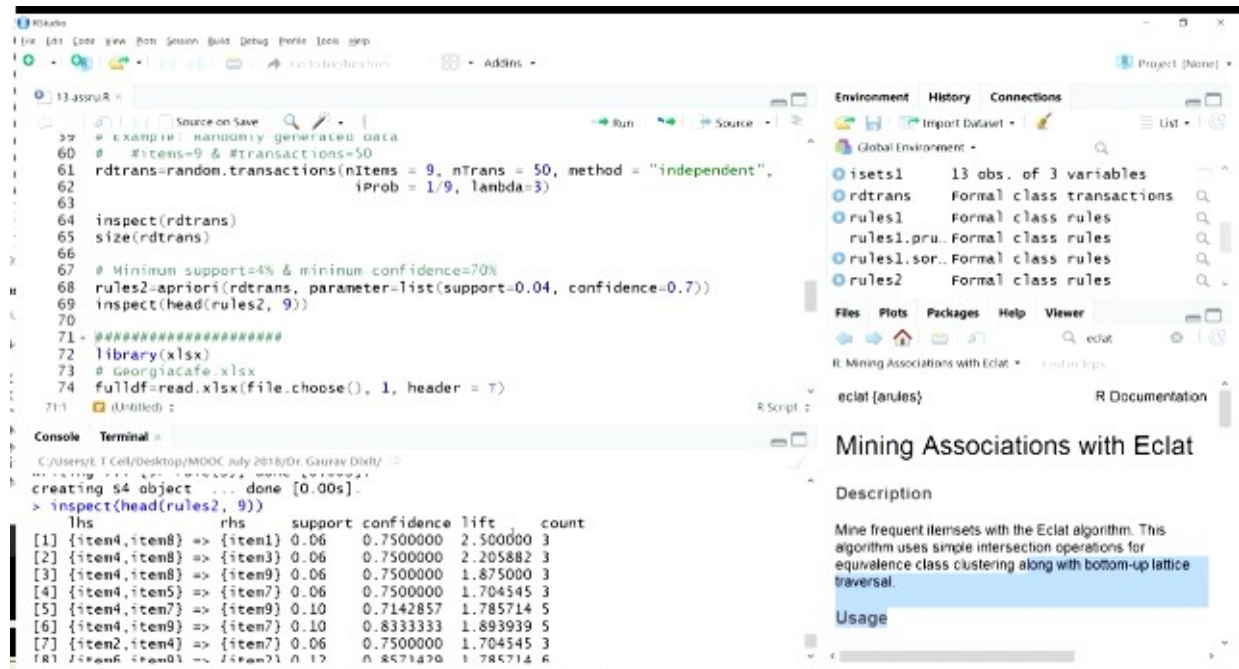
```

```

> size(rdtrans)
[1] 1 3 4 5 6 4 7 5 4 1 2 4 3 2 3 1 1 2 5 2 3 3 7 4 2 1 4 1 3 3 2 5 4 4 4 4 1 4
[40] 3 4 2 7 1 1 3 2 5 3 2

```

So another thing that we might be interested in size of this transactions, so you can see for each transaction number of items that are present, and each of those transaction that can also be seen through this output, however right now we are more interested in doing this association rules mining, so we'll run this Apriori, we'll call this Apriori function and use this randomly you know generated data base, and the parameter as you can see we are specifying the minimum support as 4%, 0.04 and the minimum confidence as 70%, so that keeping the support as quite low value so that the more number of rules are identified and the confidence is 70%, so let's run this code.



Now let's inspect first 9 rules so you can see here, we're just looking at first 9 rules, so let's scroll and see, so as you can see here that the first you know rule that has been selected is, where the, if item 4 and item 8 are purchased then item 1 is also purchased, the lift value of this particular rule is 2.5, so even though this whole data was randomly generated, and then we apply association rules mining on this database, but still we are able to find out some very strong associations, item 4, item 8, item 1 so this particular transaction and we did not had any you know inbuilt association, this was randomly generated, these were randomly generated transactions, and still we are able to identify certain association rules. For example, rule number 1, 2, 3, 4, 5 all are having you know, there are number of rules as we can see in this output which are having lift value greater than 1.5, two rules are having lift value greater than 2, one is having lift value of 2.5, so some strong associations are being identified even in this randomly generated data, so this is the cause of concerned, so how do we overcome this? So association rules mining the rules that we get, the rules that are having higher support, rule that are having you know higher confidence value and high lift value whether they are meaningful or not, whether they are useful or not, how do we find out?

Association Rules

- Open RStudio
- Assessing rules for spuriousness due to chance effects
 - More no. of transactions for a rule, less chance of spuriousness
 - Large no. of transactions yield margins of error to a small range occurring due to sampling variations
 - Look to statistical confidence intervals on proportions
 - More no. of distinct rules, higher chance of spuriousness
 - Limit the no. of rules that can be considered from topdown to a no. which can be reasonably incorporated in human decision making process to guard against automated review of rules
- Open RStudio

So let's discuss more about this, so how do we assess rules for spuriousness due to chance effects, so two points that we are going to discuss here, so if you look at the example that we have just gone through the support you know level that we have kept there was quite low, right, so if we are able to have you know for a particular rule, if there are more number of transactions that are supporting a particular rule, then there is less chance for spuriousness, so if there are let say you know thousand transactions and if there are number of transaction let's say hundred or more than hundred or supporting a particular rule, then there are less chance of spuriousness, if we go back and look at the results that we had, if we see that you know the first rule that had

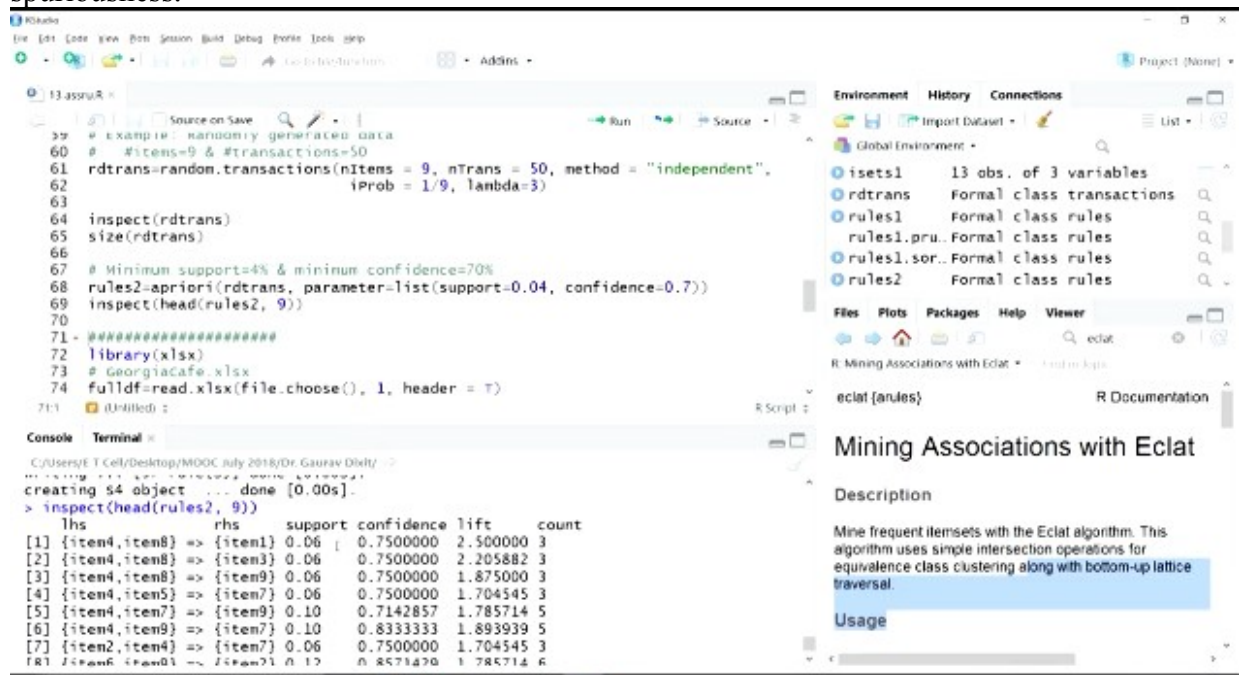
The screenshot shows an R script in RStudio. The code generates random transactions and mines association rules using the Eclat algorithm. The output table shows the following results:

	lhs	rhs	support	confidence	lift	count
[1]	{item4,item8}	{item1}	0.06	0.7500000	2.500000	3
[2]	{item4,item8}	{item3}	0.06	0.7500000	2.203882	3
[3]	{item4,item8}	{item9}	0.06	0.7500000	1.875000	3
[4]	{item4,item5}	{item7}	0.06	0.7500000	1.704545	3
[5]	{item4,item7}	{item9}	0.10	0.7142857	1.785714	5
[6]	{item4,item9}	{item7}	0.10	0.8333333	1.893939	5
[7]	{item2,item4}	{item7}	0.06	0.7500000	1.704545	3

The Eclat documentation is also visible, describing the algorithm as a simple intersection operation for equivalence class clustering along with bottom-up lattice traversal.

the lift value of 2.5 and the confidence of, confidence value of 75%, but it had just the 6% of support, right, so it has low support, so number of transaction out of the 50 that we had, so

number of transaction would be quite few, just about you know, you know the three transactions, so if more number of transaction or supporting a particular rule then we can, we will be able to, we might be able to eliminate this chance of spuriousness. So the rule of thumb could be you know have the, you know select those rules which are having a large number of transactions supporting it, so large number of transaction yield margins of error to a small range occurring due to sampling variations, if there are large number of transaction involved you know behind the rule then the you know margin of error that would be to a small range, right, so the rule is going to be more meaningful, more useful. Of course we can always look for statistical confidence, intervals on proportion to look at the margin of error, right, so this is one. The second point that we are discussing here is that if there are more number of distinct rules then there is higher chance of spuriousness, so if we are going with, you know if we are going with the, if we look at our business and operational capabilities, and if we are going with a number of you know a higher number of distinct rules then there is more chance that you know, that this particular spuriousness is going to occur, so how do we overcome this problem? So what we can do is we can limit the number of rules that can be considered from the top downs so in this sorted list using the lift ratio, so from that you know list, from the top down fashion we can you know limit the number of rules that have to be considered for implementation depending on you know, what can be reasonably incorporated in a human decision making process, so all this is to guard against, the automated review of rules that we have, so in this data driven modeling process, we are always going to generate a number of rules, this whole process association rules mining is always going to generate a number of a large number of rules, so we can restrict ourselves depending on what we can accommodate in our human decision making process, and just consider the top you know top rules, right, so in this fashion we would be able to you know control this problem of spuriousness.

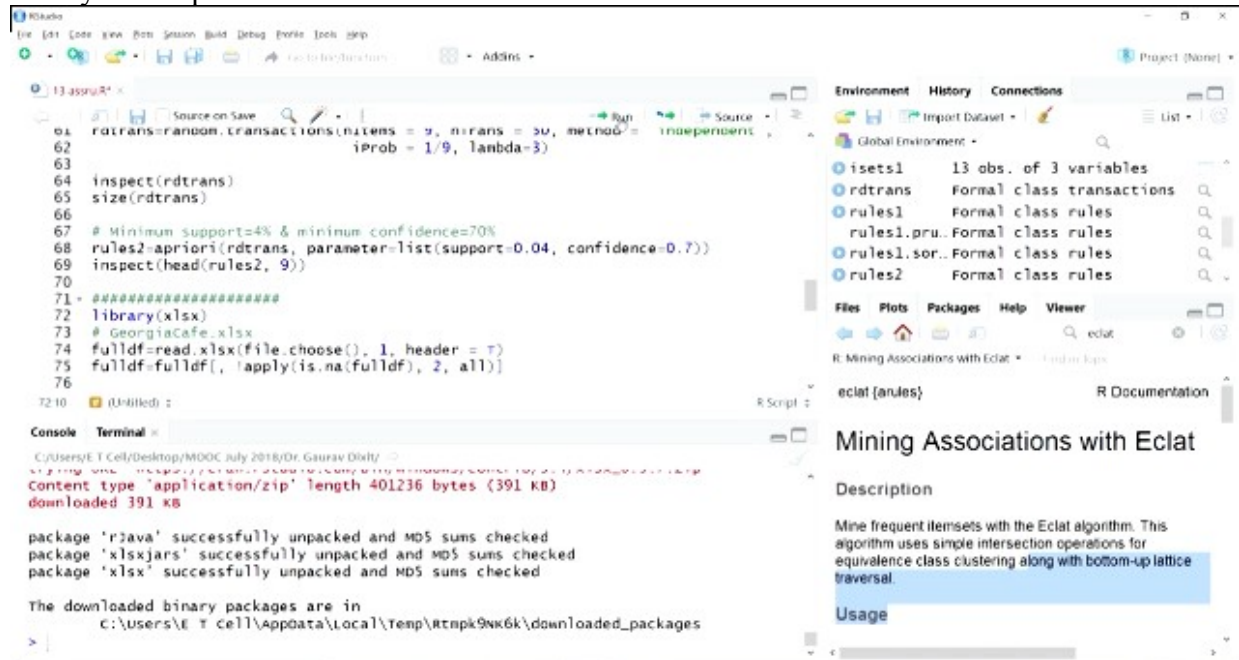


What we can look at right now is a database which is again this particular databases also manufactured, but this particular database is you know based on a, you know, a real you know manufacture based on a real life situation, so for this, this is GeorgiaCafe.xlsx file that we have,

so this dataset is about customers purchasing different you know items from a café, so let's look at this database, so we will go through all this whatever we have discussed till now, association rule mining process, we'll go through this process using this particular dataset.

So let's load this library, so this particular library right now not installed, so let's install this, double quotes let's use the name of this package, now once this package is installed so we would be able to load this particular you know, this particular function we would be able to load this library, library xlsx and then we'll have a look at this particular dataset that we are talking about.

So as I talked about this particular dataset Georgia café this is about the customers, transactions of customers purchasing certain food items, other snacks, and other items, beverage items from this café and each transaction is going to represent the items that are going to be, that have been purchased by that particular customers and we are going to use, so this is, this dataset is though, even though if this is manufactured this is representing a real situation, real life situation, so we will try to understand this dataset, and then mine, so whether which items are we will try to mine these datasets, these transaction and find out which items are being purchased together, so once this package is installed, so it is about to get installed, must this install, we will load this library and import this dataset.



```
61 rdtrans=random.transactions(n.items = 5, n.trans = 50, method = "independent",
62                             iProb = 1/9, lambda=3)
63
64 inspect(rdtrans)
65 size(rdtrans)
66
67 # Minimum Support=4% & minimum confidence=70%
68 rules2=apriori(rdtrans, parameter=list(support=0.04, confidence=0.7))
69 inspect(head(rules2, 9))
70
71 #####
72 library(xlsx)
73 # GeorgiaCafe.xlsx
74 fulldf=read.xlsx(file.choose(), 1, header = T)
75 fulldf=fulldf[, !apply(is.na(fulldf), 2, all)]
76
```

Environment History Connections

- Global Environment
- isets1 13 obs. of 3 variables
- rdtrans Formal class transactions
- rules1 Formal class rules
- rules1.pru... Formal class rules
- rules1.sor... Formal class rules
- rules2 Formal class rules

Files Plots Packages Help Viewer

R Mining Associations with Eclat

eclat [anules] R Documentation

Mining Associations with Eclat

Description

Mine frequent itemsets with the Eclat algorithm. This algorithm uses simple intersection operations for equivalence class clustering along with bottom-up lattice traversal.

Usage

So we can see, so the package installed so let's load the library, so there is one more package that we need to install, so because there are so many packages that are involved in while we do our, you know, modeling in R, sometimes we'll have to install all these packages on the go, and load them, so we need to install this R java package, since you can see there was an error, so we need to install this. So looks like now this time we'll be able to load, package we're not able to install, so we'll stop here, in the next lecture we'll have this particular dataset loaded imported, and the R studio environment and then we'll do our mining using this particular dataset. Thank you.

For Further Details **Contact**



Coordinator, Educational Technology Cell
Indian Institute of Technology Roorkee
Roorkee- 247 667
E Mail: etcell@iitr.ernet.in, etcell.iitrke@gmail.com
Website: www.nptel.iitm.ac.in

For Further Details Contact
Coordinator Educational Technology Cell
Indian Institute of Technology Roorkee
Roorkee – 247 667
E Mail:-etcell@iitr.ernet.in, iitrke@gmail.com
Website: www.nptel.iitm.ac.in

Acknowledgement

Prof. Ajit Kumar Chaturvedi
Director, IIT Roorkee

NPTEL Coordinator

IIT Roorkee

Prof. B. K Gandhi

Subject Expert

Dr. Gaurav Dixit

Department of Management Studies

IIT Roorkee

Produced by

Mohan Raj.S

Graphics

Binoy V.P

Web Team

Dr. Nibedita Bisoyi

Neetesh Kumar

Jitender Kumar

Vivek Kumar

Dharamveer Singh

Gaurav Kumar

An educational Technology cell

IIT Roorkee Production
© Copyright All Rights Reserved
WANT TO SEE MORE LIKE THIS
SUBSCRIBE