Association Rule Mining - Algorithms & Examples

Solon Ioannou

7/24/2020

## Data Preparation

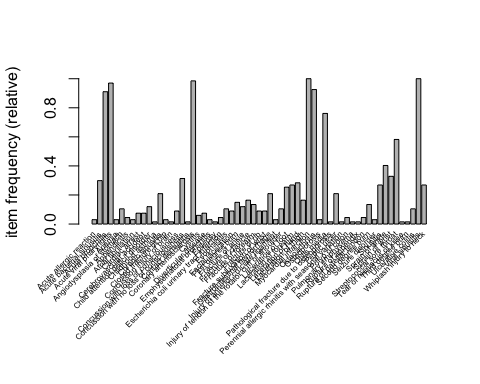
#convert relevant variables to factor  
cols <- c("rowId", "covariateId")  
df[, cols] <- lapply(df[, cols], factor)

# Preparing dataset without temporal information  
df\_input <- as.data.frame(df\_input)  
class(df\_input)  
trans\_sets <- as(split(df\_input[,"covariateId"], df\_input[,"rowId"]), "transactions")

trans\_sets2 <- as(split(df\_input2[,"covariateLabel"], df\_input2[,"rowId"]), "transactions")

## Data Exploration

### Frequency of events in the dataset



## Analysis

### Apriori

#Define parameters for apriori algorithm. See ?APparameter for more details  
ap\_params <- list(  
 support = 0.5,   
 confidence = 0.5,   
 minlen = 1, #number of items per itemset  
 maxlen = 50, #maximal number of items per itemset   
 arem = "chi2", #additional rule evaluation  
 aval = TRUE, #return additional rule measure  
 minval = 0, #default value for minimal value of additional rule measure  
 maxtime = 0 #Disabling he limit for subset checking  
)

rules1 <- apriori(trans\_sets2, parameter = ap\_params)

rules1

## set of 704 rules

Generated rules with varying support threshold

|  |  |  |  |
| --- | --- | --- | --- |
| minSupport | numberRules | numberMIrules | MI% |
| 0.05 | 118270 | 12610 | 10.66204 |
| 0.10 | 28016 | 3456 | 12.33581 |
| 0.15 | 9316 | 1244 | 13.35337 |
| 0.20 | 5632 | 784 | 13.92045 |
| 0.25 | 3428 | 500 | 14.58576 |
| 0.30 | 1616 | 224 | 13.86139 |
| 0.35 | 1216 | 160 | 13.15789 |
| 0.40 | 1036 | 132 | 12.74131 |
| 0.45 | 1024 | 128 | 12.50000 |
| 0.50 | 704 | 96 | 13.63636 |
| 0.55 | 660 | 92 | 13.93939 |
| 0.60 | 448 | 64 | 14.28571 |
| 0.65 | 448 | 64 | 14.28571 |
| 0.70 | 368 | 56 | 15.21739 |
| 0.75 | 212 | 36 | 16.98113 |
| 0.80 | 192 | 32 | 16.66667 |
| 0.85 | 172 | 30 | 17.44186 |
| 0.90 | 72 | 16 | 22.22222 |

#induction <- ruleInduction(rules1, trans\_sets, control = list(verbose =TRUE) )

### Selecting rules with MI in the RHS

mi\_subset <- subset(rules1, subset = rhs %in% 'Myocardial infarction')  
inspect(head(sort(mi\_subset, by = "lift")))

## lhs rhs support confidence  
## [1] {} => {Myocardial infarction} 1.0000000 1   
## [2] {Streptococcal sore throat} => {Myocardial infarction} 0.5820896 1   
## [3] {Otitis media} => {Myocardial infarction} 0.7611940 1   
## [4] {Acute bronchitis} => {Myocardial infarction} 0.9104478 1   
## [5] {Osteoarthritis} => {Myocardial infarction} 0.9253731 1   
## [6] {Coronary arteriosclerosis} => {Myocardial infarction} 0.9850746 1   
## chi2 coverage lift count  
## [1] 0 1.0000000 1 67   
## [2] 0 0.5820896 1 39   
## [3] 0 0.7611940 1 51   
## [4] 0 0.9104478 1 61   
## [5] 0 0.9253731 1 62   
## [6] 0 0.9850746 1 66

### ECLAT

itemsets\_ec <- eclat(trans\_sets2, parameter = list(supp=0.5, maxlen=30))

## Eclat  
##   
## parameter specification:  
## tidLists support minlen maxlen target ext  
## FALSE 0.5 1 30 frequent itemsets TRUE  
##   
## algorithmic control:  
## sparse sort verbose  
## 7 -2 TRUE  
##   
## Absolute minimum support count: 33   
##   
## create itemset ...   
## set transactions ...[61 item(s), 67 transaction(s)] done [0.00s].  
## sorting and recoding items ... [8 item(s)] done [0.00s].  
## creating bit matrix ... [8 row(s), 67 column(s)] done [0.00s].  
## writing ... [191 set(s)] done [0.00s].  
## Creating S4 object ... done [0.00s].

rules\_ec <- ruleInduction(itemsets\_ec, trans\_sets2, confidence = 0.5)  
  
#inspect(sort(rules\_ec, by = "lift"))  
subset.rules\_ec <- subset(rules\_ec, subset = rhs %in% 'Myocardial infarction') # get subset rules in vector  
length(subset.rules\_ec) #

## [1] 95

inspect(head(sort(subset.rules\_ec, by = "lift")))

## lhs rhs support confidence lift itemset  
## [1] {Acute bronchitis,   
## Acute viral pharyngitis,   
## Coronary arteriosclerosis,   
## Osteoarthritis,   
## Streptococcal sore throat,   
## Viral sinusitis} => {Myocardial infarction} 0.5373134 1 1 1  
## [2] {Acute bronchitis,   
## Acute viral pharyngitis,   
## Coronary arteriosclerosis,   
## Osteoarthritis,   
## Streptococcal sore throat} => {Myocardial infarction} 0.5373134 1 1 3  
## [3] {Acute bronchitis,   
## Coronary arteriosclerosis,   
## Osteoarthritis,   
## Streptococcal sore throat,   
## Viral sinusitis} => {Myocardial infarction} 0.5522388 1 1 4  
## [4] {Acute bronchitis,   
## Coronary arteriosclerosis,   
## Osteoarthritis,   
## Streptococcal sore throat} => {Myocardial infarction} 0.5522388 1 1 6  
## [5] {Acute bronchitis,   
## Acute viral pharyngitis,   
## Osteoarthritis,   
## Streptococcal sore throat,   
## Viral sinusitis} => {Myocardial infarction} 0.5373134 1 1 8  
## [6] {Acute bronchitis,   
## Acute viral pharyngitis,   
## Osteoarthritis,   
## Streptococcal sore throat} => {Myocardial infarction} 0.5373134 1 1 10

### FP-growth

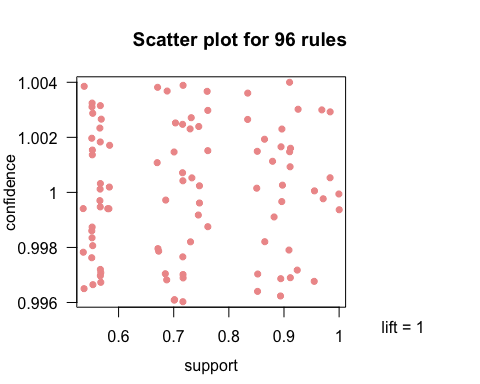
#rules\_fp <- rCBA::fpgrowth(trans\_sets2, support=0.5, confidence=0.5, consequent = "covariateLabel", parallel=FALSE)

## Visualizations

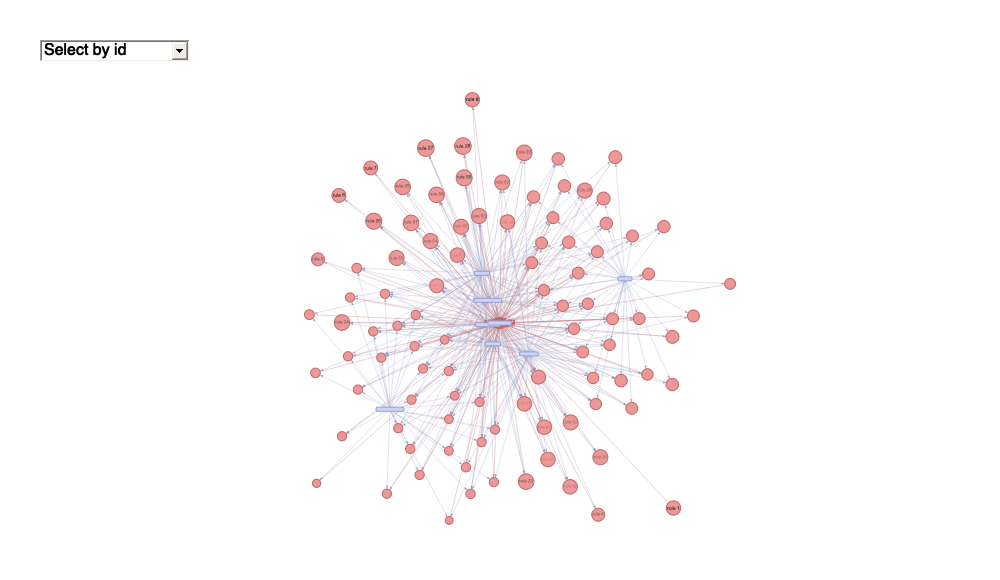
### Apriori

plot(mi\_subset)

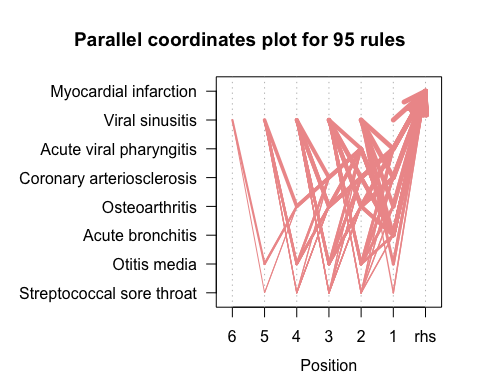
## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.



plot(mi\_subset, method = "graph", engine = "htmlwidget")



plot(mi\_subset, method = "paracoord", reorder = TRUE)



### ECLAT

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

