Lab3 Report

Théo Lambert, William Didier

Theoritical Calculations

Question 1

We have the following formulas :
* 1-support item : s(A) = P(A)
* 2-support item : s(A => B) = P(A inter B)
* 2-confidence item : alpha(A => B) = P(A|B)
* 2-item lift : lift(A => B) = s(A => B)/(P(A)P(B))

Question 2

```
cond_probs <- read.table("~/Applied_Probabilities/TP3/group11_CondProbs.csv", header=TRUE, sep=",", dec=
cond_probs <- data.matrix(cond_probs)
cond_probs <- cond_probs[, -1]
```

Question 3

```
# initialisation of the 1-support, 2-support
nobs <- dim(cond_probs)[1]
n <- dim(cond_probs)[2]
namesvec <- colnames(cond_probs)
first_item_prob <- c(0.02, 0.2, 0.2, 0.2, 0.05, 0.01, 0.02, 0.3)
support1 <- array(0,n)
support2 <- matrix(0, ncol=n, nrow=n, dimnames=list(namesvec, namesvec))

# conditions on support, confidence and lift
support_thresh <- 0.15
confidence_thresh <- 0.5
lift_thresh <- 1.1

# computing support of one-item rules
support1 <- t(cond_probs)%*%first_item_prob
View(support1)

# computing support of 2-item rules
support2 <- matrix(nrow = 8, ncol = 8, dimnames = c(list(namesvec),list(namesvec)))
for(i in 1:8){
  for (j in 1:8){
    support2[i,j] = sum(cond_probs[,i]*cond_probs[,j]*first_item_prob)
  }
}
diag(support2) <- support1
View(support2)
```

```r
# computing confidence matrix for 2-item rules
confidence2 <- matrix(nrow = 8, ncol = 8, dimnames = c(list(namesvec),list(namesvec)))

for (i in 1:8){
  for (j in (i):8){
    confidence2[i,j] = support2[i,j]/support1[j]
    confidence2[j,i] = support2[i,j]/support1[i]
  }
}
diag(confidence2) <- array(1,8, dimnames = list(namesvec))
View(confidence2)

# computing lifting matrix for 2-item rules
tmp <- matrix(c(support1), nrow=n, ncol=n, byrow=TRUE)
lift2 <- confidence2/tmp
View(lift2)
```

Unfortunately we weren't able to answer this question without using for loops. Since the number of iterations is really small comapred to the second part of the lab (where we managed to avoid using for loops), we thought that it wouldn't be a big problem in terms of performance.

Question 4

```r
rules_matrix <- (support2>=support_thresh)*(confidence2>=confidence_thresh)*(lift2>=lift_thresh)
diag(rules_matrix) <- 0

significant_rules <- NULL
for (i in 1:n) {
  if (sum(rules_matrix[i,]) > 0) {
    significant_rules <- c(significant_rules, paste(namesvec[i], "->", namesvec[rules_matrix[i,]==1], s
  }
}
significant_rules
```

```
## [1] "Screws->Nails"       "Screws->Screwdriver" "Screws->Wrench"
```

Question 5

The rules that we found are hard to interprate. We thought we would find more obvious rules such as hammer -> nails, or screwdriver -> screws rather than screws -> screwdriver

Data simulation and basket analysis of the simulated dataset

Question 1

```r
set.seed(34567)
```

Question 2

```
baskets = matrix(0, nrow=5000, ncol=8, dimnames=list(NULL, namesvec))
first_item <- sample.int(8, size = 5000, replace = TRUE, prob = first_item_prob)
baskets[cbind(1:5000,first_item)] = 1
```

We have now drawn the first item of the basket according to the first_item probabilities that we were given. We will now use the conditional probabilities we have computed to simulate the rest of the baskets.

```
unif = matrix(runif(5000*8,0,1), nrow = 5000, ncol = 8)
matrix_compare = matrix(cond_probs[first_item,], nrow = 5000, ncol = 8, dimnames = list(NULL, namesvec))
baskets <- (matrix_compare > unif) * 1
colMeans(baskets)
```

```
##      Hammer       Nails      Screws Screwdriver      Wrench       Level
##      0.1420      0.5360      0.6124      0.5452      0.1762      0.1006
##       Drill       Brush
##      0.1290      0.5872
```

We defined two new matrixes : the first one, unif, is a draw of uniform law between 0 and 1 in a matrix that has the same dimension as baskets. We then compare the conditionnal probabilities knowing the first item that was drawn (matrix_compare), to this simulation of the uniform law. If the value in unif is smaller than the conditionnal probability, we set the corresponding value of baskets to 1. The output seems to be coherent since the mean of any column is clode to the theoretical support1 we computed erlier on.

```
support1_exp <- colMeans(baskets)
```

Question 3

```
support2_exp <- t(baskets) %*% baskets / 5000
```

The structure of our matrix, composed of 0 and 1 allows us to use the matricial product to sum the number of times we have A and B simultaneously in all the lines of the matrix. We simply have to divide by the number of simulations to get an empirical probability. Now, let's compute the confidence of the 2 item rules.

```
confidence2_exp <- support2_exp / support1_exp
```

To compute the lift of the 2 item rules, we need to divide each column of the confidence by the corresponding line of the support of the 1 item rules. In order to do so, we build a diagonal matrix that we'll multiply with our confidence matrix.

```
diag_matrix = matrix(0,8,8)
diag(diag_matrix) <- 1/support1_exp
lift2_exp <- confidence2_exp %*% diag_matrix
```

We now use the same piece of code as earlier on to print the significant rules we obtained through the simulation.

```
rules_matrix_exp <- (support2_exp>=support_thresh)*(confidence2_exp>=confidence_thresh)*(lift2_exp>=lif
diag(rules_matrix) <- 0

significant_rules_exp <- NULL
for (i in 1:n)
   if (sum(rules_matrix_exp[i,]) > 0)
       significant_rules_exp <- c(significant_rules, paste(namesvec[i], "->", namesvec[rules_matrix_exp

significant_rules_exp
```

```
## [1] "Screws->Nails"       "Screws->Screwdriver" "Screws->Wrench"
## [4] "Brush->Brush"
```

The rules we find are coherent with are theoritcal calculations, except the last one that shouldn't be displayes but we didn't find how to suppress it