Coursework7

Fortunat Mutunda

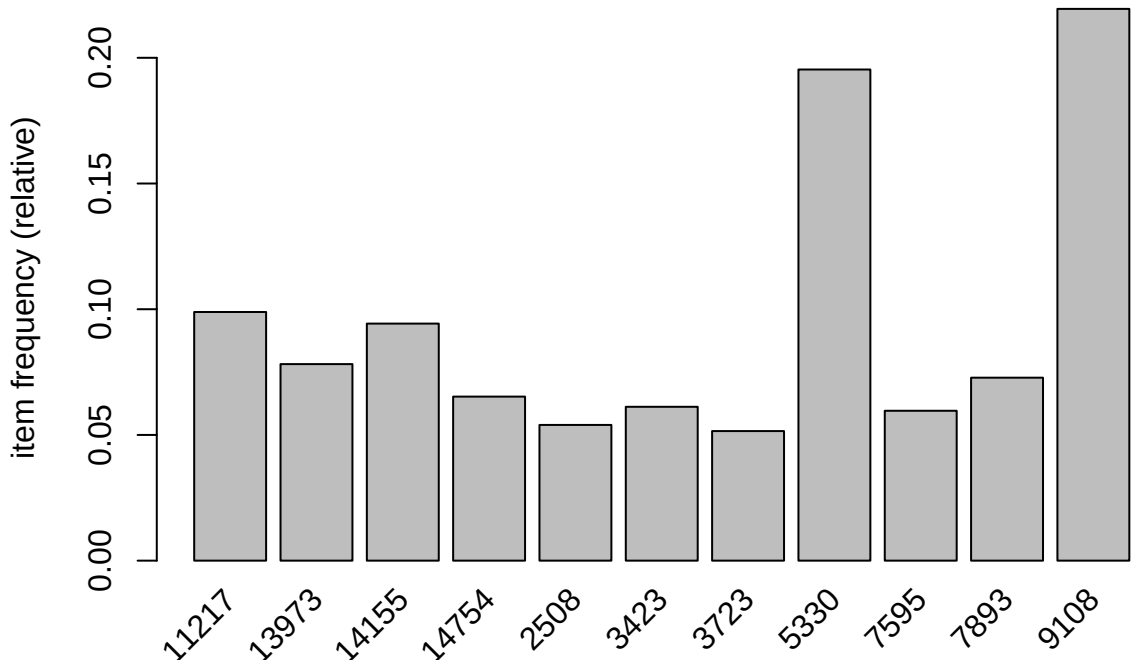
March 23, 2016

1. Report which tools you decided to use, how you used them, what were the first results. Also report the running times for the tools chosen.

For this first tast I have selected the *arules* library because that's the one I am used to, and it's the one I've been using for the couple past homeworks. It is not only easy to use but it is also very intuitive and pretty straight forward. With its nice inuilt function `read.transaction()` I can get all the transations without an hassle.

```
#df = read.csv("supermarket.txt", header = F, sep = " ")

#Starts from here
ptm <- proc.time()
trans = read.transactions("supermarket.txt", format = "basket" , sep = " ")
itemFrequencyPlot(trans, support = 0.05)
```



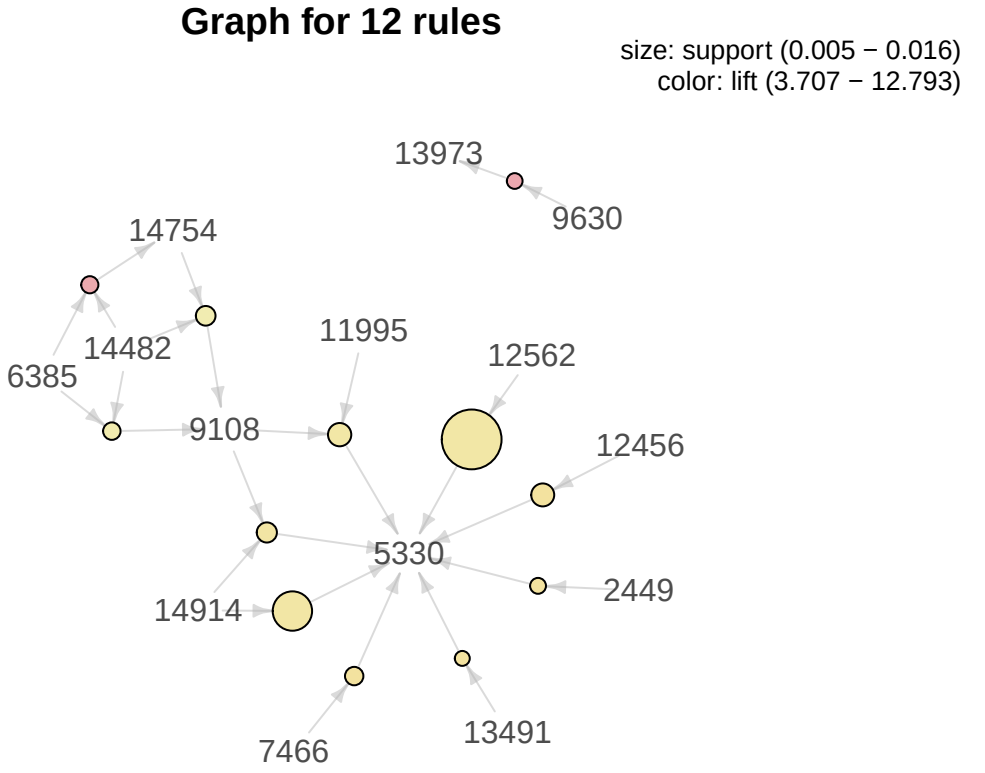
- First I've read the transaction then I have proceeded by getting which items are the most frequents in the transaction. And it appears that `9108` is the most frequent item in the given itemset, when I use a support of 0.05.

```
#rules
ruless <- apriori(trans, parameter= list(supp=0.005))
```

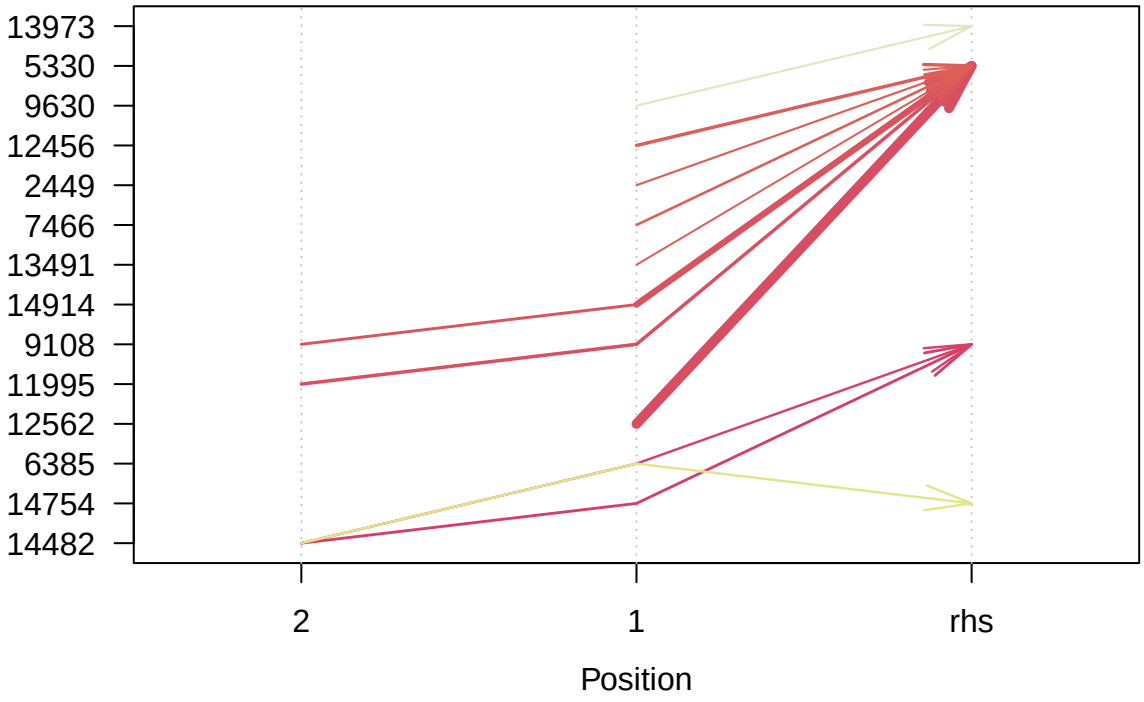
```
## Apriori
##
```

```
## Parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.8    0.1    1 none FALSE            TRUE   0.005      1     10
##  target   ext
##   rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 126
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[15699 item(s), 25382 transaction(s)] done [0.03s].
## sorting and recoding items ... [186 item(s)] done [0.00s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [12 rule(s)] done [0.00s].
## creating S4 object  ... done [0.01s].
```

```
plot(ruless, method = "graph")
```




```
plot(ruless, method="paracoord", control=list(reorder=TRUE))
```



- Second I have applied the `apriori` alogrithm on the given transaction with a support of 0.05. And the I have plotted two differents plots. *Notice* that whenever a client almost get any kind of item they also end up getting the item `5330` but bigger chances are mostly when `12562` is gotten then `5330` will be gotten too or when `14914` it is still the same case.

```
confidence  0.8   |   minval 0.1  | smax 1  | arem none | aval FALSE   | originalSupport
TRUE | support  0.005 | minlen 1 | maxlen 10
```

- This is the running time of the whole process.

```
proc.time() - ptm
```

```
##    user  system elapsed
##   5.207   0.271   5.681
```

`proc.time` returns five elements for backwards compatibility, but its print method prints a named vector of length 3. The first two entries are the total user and system CPU times of the current R process and any child processes on which it has waited, and the third entry is the real' elapsed time since the process was started.

2. Report overall 5 different high-support, high-confidence, high-lift rules; provide the respective contingency tables and scores.

The code here might be a bit redundant but I am going to try to get first the top five 5 support

```
ruless.sorted = sort(ruless,by = "support")
five_rules = head(ruless.sorted,5)
inspect(five_rules)
```

```
##    lhs             rhs    support     confidence lift
## 7  {12562}      => {5330} 0.016231975 0.8673684  4.440408
## 6  {14914}      => {5330} 0.011110236 0.8924051  4.568581
## 12 {11995,9108} => {5330} 0.007091640 0.8737864  4.473265
## 5  {12456}      => {5330} 0.007012844 1.0000000  5.119403
## 8  {14914,9108} => {5330} 0.006303680 0.9142857  4.680597
```

Then the top 5 hight confidence

```
ruless.sorted = sort(ruless,by = "confidence")
five_rules = head(ruless.sorted,5)
inspect(five_rules)
```

```
##   lhs        rhs     support     confidence lift
## 1 {9630}  => {13973} 0.005200536 1          12.793347
## 2 {13491} => {5330}  0.005003546 1           5.119403
## 3 {7466}  => {5330}  0.005870302 1           5.119403
## 4 {2449}  => {5330}  0.005239934 1           5.119403
## 5 {12456} => {5330}  0.007012844 1           5.119403
```

Lastly the top 5 lift

```
ruless.sorted = sort(ruless,by = "lift")
five_rules = head(ruless.sorted,5)
inspect(five_rules)
```

```
##   lhs             rhs     support     confidence lift
## 1 {9630}       => {13973} 0.005200536 1.0000000  12.793347
## 9 {14482,6385} => {14754} 0.005555118 0.8245614  12.638296
## 2 {13491}      => {5330}  0.005003546 1.0000000   5.119403
## 3 {7466}       => {5330}  0.005870302 1.0000000   5.119403
## 4 {2449}       => {5330}  0.005239934 1.0000000   5.119403
```

3. Discuss whether some other scores studied last week or in the lecture slides would help identify "more interesting" and different rules?

```
head(interestMeasure(ruless,c("oddsRatio", "leverage","addedValue", "chiSquared","certainty", "cosine",
```

```
##   oddsRatio    leverage addedValue chiSquared certainty    cosine
## 1        NA 0.004794033  0.9218344  1564.8599         1 0.2579385
## 2        NA 0.004026177  0.8046647   525.7950         1 0.1600474
## 3        NA 0.004723625  0.8046647   617.4155         1 0.1733564
## 4        NA 0.004216390  0.8046647   550.7666         1 0.1637844
## 5        NA 0.005642988  0.8046647   738.4322         1 0.1894771
##   imbalance
## 1 0.9334677
```

```
## 2 0.9743848
## 3 0.9699476
## 4 0.9731747
## 5 0.9640984
```

- The odds of finding X in transactions which contain Y divided by the odds of finding X in transactions which do not contain Y.
- leverage is defined as supp(X->Y) - (supp(X) supp(Y)). It measures the difference of X and Y appearing together in the data set and what would be expected if X and Y where statistically dependent. It can be interpreted as the gap to independence.
- addedValue Defined as conf(X -> Y) - supp(Y)
- chiSquared The chi-squared statistic to test for independence between the lhs and rhs of the rule. The critical value of the chi-squared distribution with 1 degree of freedom (2x2 contingency table) at alpha=0.05 is 3.84; higher chi-squared values indicate that the lhs and the rhs are not independent. Note that the contingency table is likely to have cells with low expected values and that thus Fisher's Exact Test might be more appropriate.
- The certainty factor is a measure of variation of the probability that Y is in a transaction when only considering transactions with X. An inreasing CF means a decrease of the probability that Y is not in a transaction that X is in. Negative CFs have a similar interpretation.
- cosine is defined as supp(X & Y)/sqrt(supp(X)supp(Y))
- imbalance, imbalance ratio is defined as |supp(X) - supp(Y)|/(supp(X) + supp(Y) - supp(X -> Y)) gauges the degree of imbalance between two events that the lhs and the rhs are contained in a transaction. The ratio is close to 0 if the conditional probabilities are similar (i.e., very balanced) and close to 1 if they are very different.

4. Given the ability to discover frequent itemsets and association rules, propose a strategy to use these tools to study different customer segments, shops, shopping times, or specific products.

Association rule mining is a popular data mining method available in R as the extension package arules. However, mining association rules often results in a very large number of found rules, leaving the analyst with the task to go through all the rules and discover interesting ones.

The most straight forward strategy for me would be to get first the support then lift lastly confidence. Then with these 3 I can already get to know what product are often baught, and which ones are always buaght together. For ex. Apples and Bananas or cats and cucumbers... etc. Knowing that we can create new marketing strategies, like placing items next to each others to increase sales... etc

5. Select some relatively high-support high-confidence rule (A->B) and based on that example describe the conditional probabilities P(A|B) and P(B|A), as well as the Bayes rule.

As I have understond the conditional probabilities I can ginve an example: The event is that you have Disease X. Let's call this Event D. Since only 2% of people in your situation have Disease X, the probability of Event D is P(D) = 0.02. If D' represents the probability that Event D is false, then P(D') = 1 - P(D) = 0.98. To define the diagnostic value of the test, we need to define another event: that you test positive for Disease X. Let's call this Event T. The diagnostic value of the test depends on the probability you will test positive given that you actually have the disease, written as P(T|D), and the probability you test positive given that you do not have the disease, written as P(T|D'). " Bayes' theorem then would give something like this:

P(D|T) = (P(T|D) P(D))/(P(T|D) P(D) + P(T|D')P(D'))

P(T|D) = 0.99

P(T|D) = 0.09

P(D) = 0.02

P(D') = 0.98

P(D|T) = ((0.99)(0.02))/((0.99)(0.02) + (0.09)(0.98)) = 0,1833

6. (Bonus 2p) Run Krimp on same data, provide commands and describe your findings and compare to Frequent Itemset Mining + Association rules. (link to Krimp documentation)

Krimp

```
# install.packages("~/Desktop/KrimpSourceBin20130201_win,lin.zip", repos = NULL, type = "win.binary")
# Error in install.packages : cannot install Windows binary packages on this platform
```