Assignment 2

Mining association rules

The objectives of this assignment are:

1) To understand the impact of the parameters on the number of frequent itemsets and rules extracted from the data.
2) To generate an interpretation of the rules and filter them, to identify the most relevant ones.

To this aim, we will use data from the AOL (America On Line) search engine, as described in the paper *A Picture of Search*, presented by G. Pass, A. Chowdhury, and C. Torgeson at the First International Conference on Scalable Information Systems, Hong Kong, June, 2006. In particular, for a subset of anonymous users in the original data we have listed the queries they submitted to the search engine for a period of three months, in 2006.

TASK 1: warm up) Read the www.csv data into Rapid Miner Studio. This file can be found on Studium. Each row represents a search query. The first field is the anonymous user id, the second field contains the search query, and each of the following columns corresponds to a frequent keyword and takes the values **t**(rue) or **f**(alse) depending on the presence of that keyword in the search query. As usual, you can configure the Read CSV operator using the "Import Configuration Wizard". After reading it, have a look at the data.

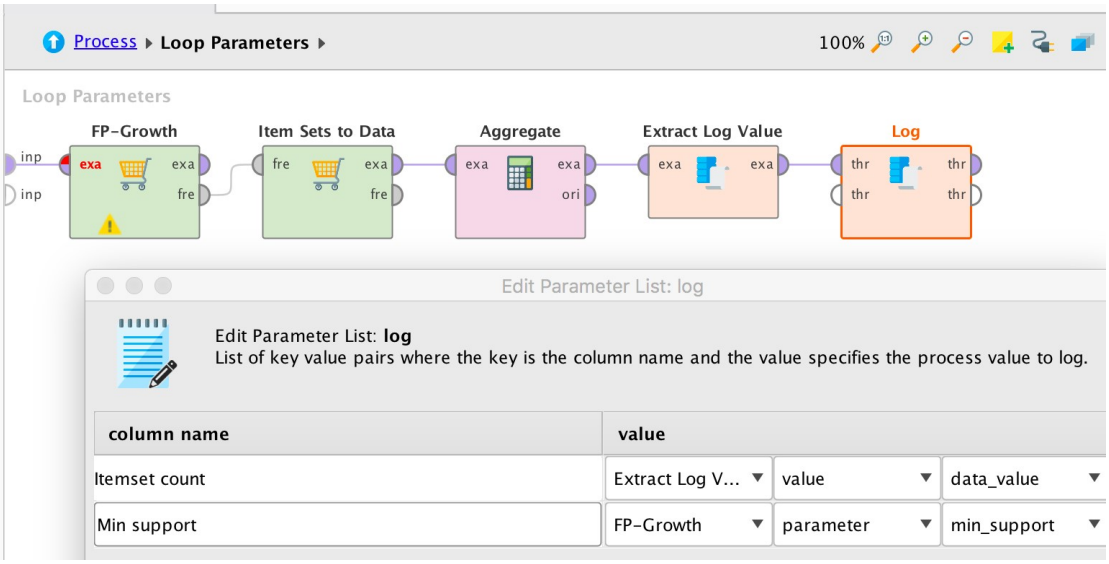
TASK 2: Generation of Frequent Itemsets) As you know, you can split the Association Rule Mining task into two parts. First, you should find the frequent itemsets. To do this, select from the data all the attributes except UserId and Query (that is, all the items) and apply the FP-Growth operator. This operator implements an efficient algorithm to identify all the frequent itemsets, and its result is the same as the one computed e.g. by the Apriori algorithm. (If you are interested in the details, FPGrowth is also described in your textbook.)

2.1) Set min-support so that an itemset must be present in at least 100 search queries to be considered frequent. While setting up the operator, do not forget to specify that the value indicating that an item is present in the search query is **t** (this parameter is called *positive value*).
[Reminder: the *support* of an itemset is the fraction of records containing the items in the itemset (in this case, the keywords), and *min-support* is the minimum value of support to consider an itemset *frequent*] **2.2)** Execute the process and examine the results.

TASK 3: Effect of support) Execute the previous process varying the min-support. Choose different values in the range [0.001, 0.01] and for each of these check:
3.1) the number of the found frequent itemsets, and **3.2)**
the maximum size of the found frequent itemsets.
You can automate this using the Loop Parameter operator. This sub-process executes its internal operators several times, varying the values of all the parameters you specify in

its settings – in this case the min-support parameter of the FP-Growth operator. Inside Loop Parameter you also need to log some summary values for the frequent itemsets (number and max size). One option is to:

1) Transform the itemsets produced by FP-Growth into a "regular" data table (Item Sets to Data).
2) Apply an aggregate function, i.e., count(items) or a max(size) respectively for Task 3.1 and Task 3.2.
3) Make the result of the aggregate function available to the logging operator (Extract Log Value).



In this way, you can log both the min-support parameter and the result of the aggregate function (number of itemsets or maximum itemset size).

Please notice that after executing the process the Result perspective will not open automatically, because there is no connection to the output ports. Therefore, you have to open it manually and look for the *log* tab, where you can use a Plot view to visualize the result.

3.3) From this analysis, choose a value for min-support leading to the discovery of around 150 frequent itemsets and use this value in the next tasks.

TASK 4: Generating the rules) Add a "Create Association Rules" operator taking the frequent itemsets as input. For now, set a minimum confidence of 80%.

4.1) Execute the process and inspect the result.
4.2) Find, among the identified rules, an example of a high-confidence rule X -> Y where Y -> X does not have high confidence. Why is the confidence of Y -> X lower?

TASK 5: impact of confidence) Execute again the previous process varying the minconfidence. This time you can do it manually (although you should now know how to automate this if you want): choose 5 different values of min-confidence in the range

[0.1, 1] and for each of these indicate the number of generated rules. See how the number of rules changes.

TASK 6: rule interpretation) In this final activity, you should test different evaluation functions with respect to their ability to highlight the *relevant* rules. Therefore, you are requested to provide your own subjective judgment on the identified rules and compare this against the values assigned by the algorithm:

6.1) Execute the process with a very low support, to extract a larger number of rules.

6.2) Choose up to 3 most **interesting** rules (notice that this is a subjective measure: you decide what is interesting). (Can you find 3?)

6.3) Choose up to 3 most **unexpected** rules (notice that this is also a subjective measure). (Can you find 3?)

6.4) For each of these 6 rules, provide their ranking according to confidence and lift (that is, at which position do the rules that you have selected appear, when rules are sorted by confidence/lift?).

6.5) Describe (if possible) the features characterizing the relevant (interesting and/or unexpected) rules, and update your process so that irrelevant rules are no longer identified (still preserving the interesting ones).

Hint: you can filter out additional (irrelevant) attributes in the Select Attributes operator and reduce the min-support.