Universidad de La Sabana
Facultad de Ingenería

Felix Mohr
2020.2

Business Intelligence

- Sheet 9 -

Exercise 1 (3 Points)

1. We have seen that there is a strong connection between conviction and lift. We now want to use certain observations about purely positive rules to possibly derive rules with negation. Let $X \to Y$ be some rule. Now show that

   a) for any $k > 0$ it holds that $conv(X \to Y) < \frac{1}{k} \Leftrightarrow lift(X \to \neg Y) > k$.

   b) $lift(X \to Y) < \alpha \Leftrightarrow lift(X \to \neg Y) > \alpha$.

   Summarize both assertions in your own words and how to use it (at most two sentences per assertion).

2. Prove the following: A rule is redundant if one or more items in the antecedent of the rule are independent of both the other items in the antecedent and the consequent.

3. In this exercise, we identify another type of obsolete rules (weak rules). Let $X, Y \neq \emptyset$ be disjoint itemsets, $Z := X \dot\cup Y$ and suppose that $sup(X) = sup(Z)$. Show that for every rule inferable from X, the confidence of the rule remains the same if we add arbitrary items of Y to the conclusion.

Exercise 2 (5 Points)

1. Write functions `sup(D,X,Y=None)`, `conf(D,X,Y)`, `lift(D,X,Y=None)`, `leverage(D,X,Y)`, `jaccard(D,X,Y)`, `conviction(D,X,Y)`, `oddsRatio(D,X,Y)`, `imp(D,X,Y)` that compute for any rule $X \longrightarrow Y$ the respective metric given the dataset `D`, where `D` is a list of lists of item IDs. Note that, for the support and lift, the conclusion is optional. This is to allow the calculation of support and lift of patterns (itemsets). Add a function `getRuleMetric(D,X,Y,metric)` that computes the metric `metric` (given as a string name in $\{sup, conf, lift, leverage, jaccard, conviction, oddsratio, imp\}$) for the rule $X \longrightarrow Y$ in the data `D`.

2. Write a function `getRuleBasedScore(D,X,metric,agg)` that computes for all bi-partitions of `X` the score of metric `metric` (given as a string name in $\{sup, conf, lift, leverage, jaccard, conviction, oddsratio, imp\}$) and aggregates it based on one of the three keywords `min`, `max`, `avg` for the `agg` parameter.

3. Write a function `filterProductiveRules(D, R)` that takes a database `D` as above and a list `R` of tuples (X,Y) corresponding to rules $X \longrightarrow Y$. It should return all rules of `R` that are productive.

4. Use the algorithms in the template file to compute all the strong rules of the shop database and filter them so that you only retain productive ones; use minsup = 500 and minconf = 0.95. Now create for each 3-combination of rule metrics we have seen a 3D scatter plot that contains one (labeled) point for each rule. Based on these plots, can you identify particularly useful rules? Which are they and why?