Submission for DB-Dag

• **Title:** Aggregation and Selection in Relational Data Mining

• **Authors:** Celine Vens, Anneleen Van Assche, Hendrik Blockeel, and Sašo Džeroski

• **Contact:**
  
  – **Adress:**
  K.U.Leuven - Department of Computer Science
  Celestijnenlaan 200A
  3001 Heverlee
  Belgium

  – **E-mail:**
  {celine,anneleen,hendrik}@cs.kuleuven.ac.be

• **Position:**
  
  – Celine Vens: Phd-student
  – Anneleen Van Assche: Phd-student
  – Hendrik Blockeel: Professor
  – Sašo Džeroski: Professor at Jozef Stefan Institute, Slovenia
Abstract

The fact that data is divided over several tables in relational databases causes many problems in the practice of data mining, since the patterns discovered by a data mining process have to take into account existing foreign key relationships.

Among the many techniques for mining relational data that currently exist, an important distinction can be made with respect to how they handle one-to-many and many-to-many relationships. A first approach uses SQL-like aggregation to summarize the relational data in one big table and applies a traditional data mining technique (e.g. a decision tree inducer). Let us call these techniques aggregating methods. A second group of techniques exploits the relational structure of the data and therefore uses a relational data mining technique (e.g. a relational decision tree inducer). These techniques usually handle sets of related entities by testing for the existence of specific elements and are hereafter called selective methods.

In order to illustrate the difference between the two groups, consider a database with just a single relation “Person” with attributes Id, Mother, Father, and Sex (Mother and Father being foreign keys to Id). Suppose we want to classify persons based on properties of a set of persons related to them. An aggregating method could easily express a concept such as “People with two children” and a selective method could easily express a concept “People with a son” (that is, at least one son). However, none of the two approaches could easily express a concept as “People with two sons”, because it contains both an aggregate and a selection (select all male children and count only those).

The aim of this work is to be able to express concepts such as the third one in a natural way. This goal can be achieved by combining aggregation and selection in a relational data mining setting. We chose to start from Tilde, a relational decision tree inducer, that requires the relational database to be in a Prolog format.

Recall that decision trees are one of the most widely used and practical methods for classifying data. They are usually learned with a divide and conquer algorithm where each internal node of the tree gets associated with a specific test of some attribute. Classification of a new instance is done by sorting it down the tree. If the test in a given node succeeds (fails), the instance is propagated to the left (right) subtree. The predicted class corresponds to the label of the leaf node where the instance arrives.

The trees output by Tilde are relational, in the sense that they contain conjunctions of first order logic literals (e.g. \((\text{person}(X), \text{child}(X,Y), \text{sex}(Y,\text{male}))\)) in the tests of internal nodes.

In order to be able to combine aggregation and selection, we first enlarged the set of tests considered at each node in the tree by including aggregate functions defined by the user (e.g. \((\text{person}(X), \text{count}(Y, \text{child}(X,Y), C), C=2))\)). Secondly, we also allowed tests that put an extra selection inside an aggregate condition (e.g. \((\text{person}(X), \text{count}(Y, (\text{child}(X,Y), \text{sex}(Y,\text{male})), C), C=2))\)).

This last extension causes an explosion of possible tests at the tree nodes. In order to deal with this problem, we can make use of so called “(relational) random forests”. This means that a forest of (relational) decision trees is build. Prediction is done by combining the outcome of the single trees by, for example, a simple voting mechanism. The method to build trees is slightly randomized in order to increase variability between the trees. For example, when building a node, instead of choosing the best test out of all possible tests, first a random subset of all possible tests is taken and the best one is chosen from this subset. We use this property to reduce the size of the search space when combining aggregation and selection: for each node, only a subset of all possible tests (e.g. 10% or the square root) has to be tested against the database, which results in an efficiency gain. We experimentally validated and compared several variants: relational random forests without aggregates, with aggregates, and with a combination of aggregates and selection.