MiningMart: Sharing Successful KDD Processes

Timm Euler, Katharina Morik, and Martin Scholz
University of Dortmund
D-44221, Dortmund, Germany
{euler, morik, scholz}@ls8.cs.uni-dortmund.de

1 Introduction

Knowledge Discovery in Databases (KDD) can be considered a high-level query language for databases, aimed at generating quality reports which support decision making on various levels of an institution. However, today there are three main obstacles to overcome before the high-level support can be satisfactory.

Firstly, most tools for data mining do not offer database access, but demand the user to draw a sample from the database and to convert it into a specific format. Secondly, data preprocessing, which is crucial for the success of the data mining step, is not sufficiently supported. Preprocessing includes aggregation, discretisation, data cleaning, and treatment of missing values. Feature generation and selection are also major challenges for KDD [3].

Although up to 80 percent of the efforts in KDD processes are spent on clever preprocessing it is still common practice to carry out this task manually by applying low level methods including SQL statements and Perl scripts. As the third obstacle the selection of an appropriate algorithm for data mining, given the data, is not yet well understood, but remains the result of a trial and error process.

We would like to offer support to developers of KDD applications to ease the complex tasks of preprocessing and the selection of a suitable data mining algorithm. The idea behind the MiningMart approach stems from case-based reasoning: it is much easier to solve a task if a solution of a similar task is known. MiningMart thus offers an environment to develop, document and share complete data processing chains from the raw data tables in a relational database to the final data mining application. These processing chains are called *cases* in MiningMart. The cases as well as the data itself are modelled in the MiningMart meta-model, which is made executable by a compiler. The case models can be published on the MiningMart web platform[1] to be downloaded and modified by other users for their own applications.

The main advantages of this metadata-based approach are:

Abstraction: Metadata are given at different levels of abstraction, a conceptual and a relational level. This makes an abstract case understandable and re-usable.

Data and Case documentation: The database objects (tables or views) as well as their conceptual counterparts are declaratively stored. So is the chain of preprocessing operations, including all operators' parameter settings etc. All entities can be given explaining names and further free text descriptions. Thus all details of a case can be restored, modified and executed again at any time.

Ease of case adaptation: In order to run a given sequence of operators on a new database, only the relational metadata and their mapping to the conceptual metadata have to be written.

[1] `http://mmart.cs.uni-dortmund.de/`

2 The MiningMart Meta Model, M4

In order to exchange successful knowledge discovery cases, a formalism to describe them in abstract, yet operational terms proved to be adequate. A *conceptual data model* is used to reference data by common everyday notions rather than by specific database names. All data transactions are described at this level. Of course, this conceptual level must be mapped to the actual data, which is described by the relational data model. After that users only work with the convenient conceptual model.

The formalism behind the conceptual data model is an ontology. It introduces *concepts* and *relationships* between them. To further organise concepts and relationships the conceptual data model offers the opportunity to use inheritance. Examples for concepts are `Customer` and `Product`; they might be connected by a relationship called `Buys`. `Customer` could have *subconcepts* like `Private Customer` and `Business Customer`. All these objects can be considered a part of an abstract model of the application domain, as concepts allow to bundle information from different tables. Given a conceptual data model, a graphical tool supports the creation of a mapping of the involved entities to the corresponding database objects (the relational model).

The next step is to implement *operators* that perform data transformations such as discretisation, treatment of `null` values, aggregation of attributes into a new one, or collecting sequences from time-stamped data. The sequence of operators is called the *case model*. Setting up or adjusting cases is supported in MiningMart by a special graphical editor. Together, the three models (conceptual and relational data model, and case model) form the MiningMart meta model, M4. Thus the same formalism is used for the metadata and for the description of the KDD process. M4 is stored in separate tables of the relational database used.

The system is designed such that new operators can easily be integrated. By modelling real world cases in future applications, further useful operators will be identified, implemented and added to the repository.

To ease the process of editing cases, applicability constraints on the basis of metadata are provided as formalised knowledge and are automatically checked by the human

computer interface. In this way only valid sequences of steps can be produced by a case designer.

From the case model of the operator chains, the MiningMart *compiler* creates SQL code that performs the data processing steps as specified by the chains (see section 3).

3 MiningMart Components

MiningMart has got three main components: the metadata model M4, which is explained in the previous section; the compiler, which makes a case executable; and the graphical user editors. A fourth part of the project is the web platform which allows to exchange cases. The compiler, the editors and the web platform are described in the following sections 3.1, 3.2 and 3.3, respectively.

3.1 The M4 Compiler

The task of the compiler is to translate the operator sequences from the case model to executable SQL. To do so, it uses static information (stored in M4) about each operator. These operator specifications define what kind of input and output the operator expects. For example, an operator removing all records with missing values for a specific attribute expects as its input the names of the attribute and its associated concept. The output is a new concept, associated with a database view whose records do not have a `null` value in the specified attribute.

The M4 compiler reads the static operator specifications, finds the definitions of parameters in the case model and executes the operator for the corresponding database objects. Succeeding operators can then access the output concept and read its data like any other one.

3.2 The Graphical Editors

MiningMart's graphical editors offer an intuitive access to the knowledge discovery process. Operators form a directed acyclic graph. Their inputs and outputs are displayed and can be edited after double-clicking on the icons. New operators can be selected from a menu and be inserted at any point. In this way, the case model can be developed easily and intuitively.

Another editor allows to create and modify the conceptual data model, map it to database objects, inspect the data and its statistics.

3.3 The Web Platform

One of the basic ideas behind MiningMart is the aspect of sharing knowledge about successful cases. The MiningMart project has set up a central web platform which allows the public exchange and documentation of exported cases. The platform makes use of a special software called `InfoLayer` [2] and is available via the MiningMart website.

When an effective knowledge discovery chain has been found, it can easily be exported and added to the Internet repository. Only the conceptual metadata is submitted, so even if a case handles sensitive information it is still possible to distribute the valuable metadata.

Users looking for relevant cases in the repository are supported by a browsable representation of the conceptual models and the context of each case, especially the business it was used in.

4 Applications

Two telecommunication companies have so far successfully used MiningMart and contributed their cases to the Internet repository. We cannot give details here due to space limitations, but information about the cases can be found on our web platform (see section 3.3). In their evaluation, the two companies emphasised the ease with which they could develop a suitable chain of preprocessing operations to prepare their data mining applications.

5 Related Work

The relevance of supporting not only single steps of data analysis but sequences of steps has long been underestimated. Where a large variety of excellent tools offer algorithms for the data mining step, only very few approaches exist which tackle the task of making clever choices during preprocessing and combining these choices to an effective and efficient sequence. Few of them support an easy reuse of successful sequences, and none works directly on databases.

Two systems that focus on preprocessing are *Idea* [1] and *GLS* [5; 4]. In *Idea*, possible data operation chains are searched through and ranked based on applicability constraints and estimations about execution time and accuracy. In contrast, MiningMart exploits the findings of expert case designers. *GLS* also finds valid preprocessing chains automatically, supported by user interaction. However, there is no mechanism to apply a successful chain to similar but different databases, which is one of the strengths of MiningMart.

6 Conclusions

To sum up, MiningMart is a database oriented approach to the graphical support of creating, editing and storing successful KDD chains. It allows to exchange such cases between different users via a central web platform, and to adapt a case to a new database. It provides an extensible set of operators that allow flexible operations on given database tables, thus supporting the costly preparation of data for data mining. Successful KDD chains as well as the data model they are based on are automatically documented.

Acknowledgement

The MiningMart project was funded by the European Union under the contract number IST-1999-11993.

References

[1] Abraham Bernstein, Shawndra Hill, and Foster Provost. An intelligent assistant for the knowledge discovery process. Technical Report IS02-02, New York University, Leonard Stern School of Business, 2002.

[2] Stefan Haustein and Jörg Pleumann. Easing participation in the semantic web. In *International Workshop on the Semantic Web at WWW2002*, May 2002.

[3] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998.

[4] N. Zhong, C. Liu, and S. Ohsuga. A Way of Increasing both Autonomy and Versatility of a KDD System. In Z.W. Ras and A. Skowron, editors, *Foundations of Intelligent Systems*, pages 94–105. Springer, 1997.

[5] N. Zhong, C. Liu, and S. Ohsuga. Dynamically Organizing KDD Processes. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(3):451–473, 2001.