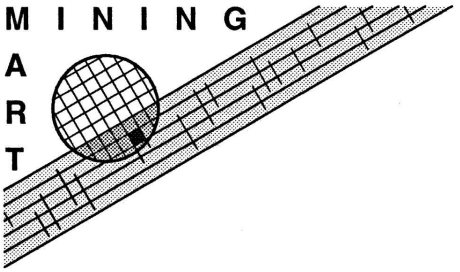

Extension of MiningMart
Contract No. IST-2001-35479
D20.5: Final Report Ext-MM

Ext-MM: Final Report

Katharina Morik, Martin Scholz, Timm Euler

Dortmund, April 17, 2003

Contents

Chapter 1

Overview

The Ext-MM project is an extension of the IST project "MiningMart: Enabling End-User Datawarehouse Mining", contract no. IST-1999-11993. In MiningMart, a collection of operational descriptions of knowledge discovery processes is developed, based on a metadata model. The MiningMart project is described in the Final Report on MiningMart, deliverable 20.4 of the MiningMart project.

The goals of Ext-MM were threefold:

- First, additional operators were to be developed that could be integrated into the MiningMart system. In MiningMart, operators realize the data processing steps in a knowledge discovery process. A set of operators had been developed within MiningMart, however it turned out that the discretization operators were not sufficient for handling all cases that the project would like to handle. Further, for the central task of automatic feature selection, only a few (wrapper-based) operators were present in MiningMart. Ext-MM has developed a range of discretization operators and additional, faster feature selection operators which are filter-based (see section 2.2). This task was fulfilled by the partner UEP.

- Second, MiningMart was to be evaluated by a new partner who had not taken part in the project development. This would put the usability and stability of the MiningMart system to a test. It would also give feedback to the system development and allow improvements early on. This task was fulfilled by the partner NIT.

- Third, a new MiningMart case was to be developed. This task is closely linked to the previous one. The partner NIT modelled one of its successful knowledge discovery processes in the MiningMart framework, and published it in the MiningMart case base, where it can serve as a template for similar knowledge discovery applications.

This document is the final report on Ext-MM. The chapters 2, 3 and 4 describe the project results with respect to the three tasks above. This document

contains pointers to MiningMart deliverables and technical reports, which can be found on the MiningMart web pages at the address:
`http://mmart.cs.uni-dortmund.de`.

Chapter 2

New MiningMart Operators

UEP has developed new operators which they integrated into the MiningMart system. Operators are part of a central component in the MiningMart system, the M4 compiler. The M4 compiler was given a modular design to allow the easy integration of new operators. A technical report on how to write new operators (TR 12-04) was available to UEP when they started their work. UEP developed two kinds of operators: for discretization and grouping, and for feature selection. The following gives an overview; details can be found in the MiningMart deliverables 16.1 and 14.4.

2.1 Discretization and Grouping Operators

The original machine learning algorithms were able to process symbolic, categorial data only. However, real-world problems (for example in medicine) involve both symbolic and numerical attributes. Therefore, it is an important issue of machine learning to *discretize* numerical attributes. The assignment of discretized values to numerical variables is well known to statisticians. Different approaches are used; for instance, discretization into a given number of categories using equidistant cutpoints, discretization into a given number of categories with the same cardinality (equifrequent discretization), or categorization based on mean and standard deviation. All these approaches are "class-blind" (or unsupervised), since they deal only with the discretized attribute.

In the framework of machine learning, the fact that the examples (objects) belong to different classes is taken into account. So the discretization algorithms are "class sensitive" (or supervised). The differences between the algortihms are in

- integration of machine learning algorithms (integrated or stand-alone as preprocessing tool),
- search strategy (top-down by splitting intervals or bottom-up by merging intervals),

- the impurity measure for evaluating potential intervals (entropy, information gain, minimum classification error),
- number of intervals (binarization or creating more intervals),
- stopping criterion (number of intervals, frequency of intervals),
- type of intervals (crisp vs fuzzy),
- number of processed attributes (most algorithms are univariate and consider only one numeric attribute at a time).

For implementation within the MiningMart project, discretization algorithms were chosen that are univariate, create crisp intervals, perform merging of intervals and have different stopping criterions.

Grouping values of a nominal attribute becomes important if the number of these values is too large (e.g. hundreds of ZIP codes or profession codes). To deal with each value separately can bring problems both during computation (e.g. branching a node) and interpretation. While discretization is a standard preprocessing operation, grouping of values of categorial attribute is less common.

For implementation within the MiningMart project, grouping algorithms were chosen that are cunterpart to the discretization ones.

What follows is a list of the new operators that were implemented, with very brief descriptions. More details can be found in the MiningMart deliverable 16.1.

UserDefinedDiscretization The user defines the cut points for discretization, i.e. the boundaries of the intervals.

EquidistantDiscretizationGivenWidth The cut points are computed such that each interval has the given width.

EquidistantDiscretizationGivenNoOfInt The cut points are computed such that a given number of intervals with same width is the result.

EquifrequentDiscretizationGivenCardinality The cut points are computed such that a given number of objects falls within each interval, and the intervals contain approximately equally many objects.

EquifrequentDiscretizationGivenNoOfInt The cut points are computed such that a given number of intervals with approximately equally many objects is the result.

ImplicitErrorBasedDiscretization The cut points are computed such that the classification error is minimized.

ErrorBasedDiscretizationGivenMinCardinality The cut points are computed such that the classification error is minimized and a given minimum number of objects falls within each interval.

ErrorBasedDiscretizationGivenNoOfInt The cut points are computed such that a given number of intervals is the result and the classification error is minimized.

UserDefinedGrouping The user defines which values should be grouped together.

GroupingGivenMinCardinality Groups are computed such that each group contains the given minimum number of objects, and the most frequent values correspond to a group each.

GroupingGivenNoOfGroups The given number of groups is used, such that the most frequent values correspond to a group each.

ImplicitErrorBasedGrouping Groups are computed such that the majority of objects in each group has the same class.

ErrorBasedGroupingGivenMinCardinality Groups are computed such that each group contains the given minimum number of values, and the majority of objects in each group has the same class.

ErrorBasedGroupingGivenNoOfGroups Groups are computed such that the given number of groups is not exceeded, and the majority of objects in each group has the same class.

2.2 Feature Selection Operators

Feature selection is an often-needed task in knowledge discovery. Frequently, the data to be analysed has dozens of attributes (features) that could be taken into account. However, the information in them may be redundant with respect to the learning task, and a number of features may not contribute any information to the learning task. See the MiningMart deliverable 14.1 for details.

There are two basic feature selection methods, the wrapper and the filter approach. Both are explained in detail in the MiningMart deliverable 14.1. Briefly, the first considers the result of the given learning algorithm on a number of subsets of features to select the best subset. This approach is implemented by the feature selection operators for deliverables 14.2 and 14.3. The second is independent of the learning algorithm and considers only the data to choose a subset of features. This approach is implemented by UEP in Ext-MM. All of their operators use an information-theoretic measure of dependence which is explained in the MiningMart deliverable 14.4.

The subsets of features which are examined are found by a certain search strategy. Various search strategies are explained in the deliverables 14.1 and 14.4. The various feature selection methods can be distinguished by their method of searching. For the implementation in MiningMart, four rather simple bottom-up and top-down search strategies were used. The simple search operators start with an empty set of features, adding one feature at a time which is chosen by

the information measure until the stopping criterion is met, or they start with the full set of features and remove one feature at a time. A variant is introduced by simultaneously adding and removing under certain conditions, resulting in a floating search behaviour.

What follows is again a list of the implemented operators with brief explanations. Details can be found in deliverable 14.4.

UserDefinedFeatureSelection The user selects the features by hand.

SimpleForwardFeatureSelectionGivenNoOfAttributes Starting with the empty set, one feature at a time is added according to the information dependence criterion until the given number of features is reached.

SimpleBackwardFeatureSelectionGivenNoOfAttributes Starting with the full set of features, one feature at a time is removed according to the information dependence criterion until the given number of features is reached.

FloatingForwardFeatureSelectionGivenNoOfAttributes Starting with the empty set, one feature at a time is added according to the information dependence criterion until the given number of features is reached. However, at each step it is also considered to remove the least significant feature, according to a definition which is explained in deliverable 14.4.

FloatingBackwardFeatureSelectionGivenNoOfAttributes Starting with the full set, one feature at a time is removed according to the information dependence criterion until the given number of features is reached. However, at each step it is also considered to add the most significant feature, according to a definition which is explained in deliverable 14.4.

Chapter 3

MiningMart Evaluation

The evaluation of the MiningMart system within Ext-MM was done based on the case described in section 4 by the National Institute of Telecommunications (NIT). NIT developed the case during a time in which the MiningMart system was also still under development. They were therefore able to provide highly valuable feedback to the system developers. They also contributed two operators, UnionByKey and NumericalIntervalManualDiscretization, which are described in detail in the technical report TR12-02. Their detailed evaluation report forms the MiningMart deliverable 17.3b.

In their evaluation, they point out that the documentation of a preprocessing case that MiningMart enforces was especially important to them, because they often re-use cases. For example, certain business reports that are needed on a regular basis (monthly or quarterly) depend on a specific processing of business data with lots of parameters. Using MiningMart, these processing operations and their parameters are specified once and can then be re-used easily. Parameters can easily be adjusted if the new application demands it. New staff can easily and quickly understand the main steps of the processing chain, rather than having to understand low-level programming code. This saves large amounts of time.

NIT appreciate the use of a case as a template for other knowledge discovery processes, though this could not be tested in the available time. Further, they stress that the application of pre-defined operators avoids the programming errors that are bound to happen when using low-level programming, as they did in previous applications. Moreover, the selection of a pre-defined operation is automatically constrained by the system reflecting certain conditions and constraints that are operator-specific. NIT point out that by thus improving the quality of the preprocessing phase, the overall quality of the KDD process can be improved, because the data mining results depend on the quality of the data. MiningMart allows to improve the preprocessing phase based on earlier experiences, and to test more options for data representations in a shorter time than without MiningMart.

In addition to the documentation that is possible in the MiningMart system, the public case base offers the option to add a business level description to a case. This is explained in deliverable 10. NIT published their case in the case base and pointed out that this business level is a good entry point for new users and distinguishes MiningMart from other tools which often lack easy documentation possibilities.

Finally, the intuitive graphical interface and chain concept were valued highly by NIT, again with respect to saved manpower.

Chapter 4

A New Case For MiningMart

The new case was developed by NIT for a telecommunications agency. The purpose of this data mining application was to find a segment of customers who would be likely to respond positively to a mailing action which would offer them a particular telecommunication service.

There are three sources of data: first, the *call detail data* from the operating telecom, which is about single telephone calls for each client of the company; second, the *client data*, which is about client-related information like address etc.; and third, *call center data* which stems from a marketing action done by a call center, where clients were called and offered the service in question; their responses are stored in this data segment.

This case contains only the preprocessing steps; the data mining is done as a separate task. The main purpose of preprocessing in this case was to aggregate all the data that is given about the various calls that the clients made, and then to join it with the other data sources using the client id as the link.

Rather than listing all steps in detail, the following gives an overview of how the main preprocessing tasks are achieved with the available MiningMart operators. For details about the operators, please see the MiningMart technical report TR12-02.

Task 1: Aggregate statistics This task uses the call detail data. The original table that forms the input for the case contains information about each call that a client made, for many clients. This data is segmented such that each segment contains only the data about one client. This is done using the MiningMart operator `SegmentationStratified`. Afterwards, specific telephone calls, like to cellular phones, to cost-free numbers or to internet providers, are selected from each segment using the operator `RowSelectionByQuery`. To each selection, the operator `SpecifiedStatistics` is applied which allows to compute several statistical features, like the number of rows, the sum of values of a certain

attribute, and so on. Here its `count` function is used to find the number of calls of the particular type that each client made.

Afterwards, the different data segments (about one client each) are re-unified and the different selections according to the types of phone calls are joined, so that the output of this preprocessing "chain" (it is rather a graph) is that for each client, the statistics about how many phone calls of each type they made exist in one table, and there is only one row per client in that table.

Task 2: Join client and call center data This task joins the other two data sources, from the client data base and from the marketing responses, such that an attribute is constructed that contains the information whether the client has already got the service in question, and if not, contains the response from the marketing campaign. This attribute is used for data mining. Two steps are needed for this task.

Task 3: Join results of tasks 1 and 2 This is done using the MiningMart operator `JoinByKey`. The output concept of this step contains one data record for each client, where the record includes the statistics computed in task 1 and the attribute computed in task 2.

Task 4: Further cleaning The last two steps fill missing values with a default value, and create a binary attribute for some of the rare phone call types indicating whether any call of that type has been done or not. The final output is a table that can be used directly as input for a data mining algorithm.

For details on the case and its evaluation, please refer to the deliverables 17.2b and 17.3b and to the internet case base (see the MiningMart final report, deliverable 20.4).

Chapter 5

Conclusions

The Ext-MM project has proven to be an essential addition to the MiningMart project without which central goals of MiningMart could not have been achieved. In particular, the evaluation of MiningMart by a new partner who had not been inside the project, but who had substantial experiences in data mining, contributed to central developments in MiningMart and now forms the basis for further advertisement of the system.

The collaboration with the partners was stimulating and productive from the very beginning. They offered an "outsider's view" to MiningMart already during the development time. Of course, the partners ended up with an "insider's view" and became true and acknowledged team members. The integration of the work of the two projects was always smooth.

In sum, it was a very successful step to extend MiningMart by this additional project and very important work could be completed through it.