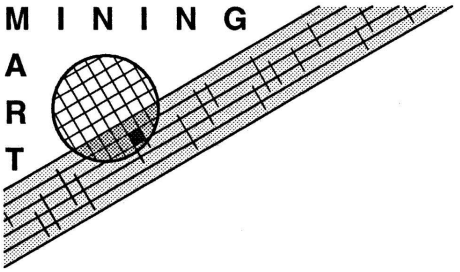

Enabling End-User Datawarehouse Mining
Contract No. IST-1999-11993
D20.4: Final Report

MiningMart: Final Report

Katharina Morik, Martin Scholz, Timm Euler

Dortmund, April 28, 2003

Contents

1 Overview **5**
- 1.1 Problems in Knowledge Discovery from Databases 5
- 1.2 Basic Ideas in MiningMart 8
- 1.3 Overview of this document 9

2 The MiningMart System **11**
- 2.1 M4 – The MiningMart MetaModel 11
  - 2.1.1 The Conceptual Data Model 15
  - 2.1.2 The Relational Model 15
  - 2.1.3 The Case Model 16
- 2.2 The Compiler 16
- 2.3 The Human-Computer Interface (HCI) 18
- 2.4 The Web Platform 21
  - 2.4.1 The Case Base 21
  - 2.4.2 The Business Level 22

3 Cases and Evaluations **25**
- 3.1 The drug store case 25
- 3.2 The call center case 27
- 3.3 The churn prediction case 28

4 Exploitation and Dissemination **31**
- 4.1 The MiningMart Websites 31
- 4.2 The One Day Seminar 32
- 4.3 Exploitation Plans 32
  - 4.3.1 Application Service Providing 33
  - 4.3.2 Research 33
  - 4.3.3 Embedding MiningMart into other KDD tools 33
  - 4.3.4 Product Development 34

5 Related Work and Conclusions **35**

6 Appendix **39**
6.1 Experiences with Partners from Associated States 39
6.2 List of documents related to MiningMart 40
6.2.1 Published Papers 40
6.2.2 Deliverables 40
6.2.3 Technical Reports 42
6.2.4 Other Documents 42

Chapter 1

Overview

1.1 Problems in Knowledge Discovery from Databases

The use of very large databases has enhanced in the last years from supporting transactions to additionally reporting business trends. The interest in analyzing the data has increased. One important topic is customer relationship management with the particular tasks of customer segmentation, customer profitability, customer retention, and customer acquisition (e.g. by direct mailing). Other tasks are the prediction of sales in order to minimize stocks, the prediction of electricity consumption or telecommunication services at particular day times in order to minimize the use of external services or optimize network routing, respectively. The health sector demands several analysis tasks for resource management, quality control, and decision making. Existing databases which were designed for transactions, such as billing and booking, are now considered a mine of information, and digging knowledge from the already gathered data is considered a tool for building up an organizational memory. Managers of an institution want to be informed about states and trends of their business. Hence, they demand concise reports from the database department.

On-line Analytical Processing (OLAP) offers interactive data analysis by aggregating data and counting the frequencies. This already answers questions like the following:

- What are the attributes of my most frequent customers?
- Which are the frequently sold products?
- How many returns did I receive after my last direct mailing action?
- What is the average duration of stay in my hospital?

Reports that support managers in decision making need more detailed information. Questions are more specific, for instance:

- Which customers are most likely to sell their insurance contract back to the insurance company before it ends?
- How many sales of a certain item do I have to expect in order to not offer empty shelves to customers and at the same time minimize my stock?
- Which group of customers best answers to direct mailing advertising a particular product?
- Who are the most cost-intensive patients in my hospital?

Knowledge Discovery in Databases (KDD) can be considered a high-level query language for relational databases that aims at generating sensible reports such that a company may enhance its performance. The high-level question is answered by a *data mining* step. Several data mining algorithms exist. However, their application is still a cumbersome process. There are three main obstacles to be overcome that have so far prevented KDD from becoming a standard procedure:

- Most tools for data mining need to handle the data internally and cannot access the database directly. Sampling the data and converting them into the desired format increases the necessary effort for data analysis.
- Preprocessing of the given data is decisive for the success of the data mining step. Aggregation, discretization, data cleaning, the treatment of null values, and the selection of relevant attributes are steps that still have to be programmed (usually in SQL) without any high-level support.
- The selection of the appropriate algorithm for the data mining step as well as for preprocessing is not yet well understood, but remains the result of a trial and error process.

The conversion of given data into the formats of diverse data mining tools is eased by toolboxes which use a common representation language for all the tools. Then, the given data need to be transformed only once and can be input into diverse tools. A first approach to such a toolbox was the development of a Common Knowledge Representation Language (CKRL), from which translators to several learning algorithms were implemented in the European project *Machine Learning Toolbox* [CCMR90, MCB91]. Today, the *weka* collection of learning algorithms implemented in JAVA with a common input format offers the opportunity to apply several distinct algorithms on a data set [WF00]. However, these toolboxes do not scale up to real-world databases naturally[1]. In contrast, database management systems offer basic statistical or OLAP procedures on the

[1]Specialized on multi-relational learning algorithms, the *ILP toolbox* from Stefan Wrobel (to be published in the network ILPnet2) allows to try several logic learning programs on a database.

given data, but do not yet provide users with more sophisticated data mining algorithms. Building upon the database facilities and integrating data mining algorithms into the database environment will be the synergy of both developments. We expect the first of the above obstacles for KDD applications to be overcome very soon.

The second obstacle is the most important one. If we inspect real-world applications of knowledge discovery, we realize that up to 80 percent of the efforts are spent on the clever preprocessing of the data. Preprocessing has long been underestimated, both in its relevance and in its complexity. If the data conversion problem is solved, the preprocessing is not at all done. Feature generation and selection[2] (in databases this means to construct additional columns and select the relevant attributes for further learning) is a major challenge for KDD [LM98]. Machine learning is not restricted to the data mining step, but is also applicable in preprocessing. This view offers a variety of learning tasks that are not as well investigated as is learning classifiers. For instance, an important task is to acquire events and their duration (i.e. a time interval) on the basis of time series (i.e. measurements at time points). Another example is the replacement of null values in the database by the results of a learning algorithm. Given attributes A_i without null values, we may train our algorithm to predict the values of attribute A_j on those records, which do have a value for A_j. The learning result can then be applied in order to replace null values in A_j. Records without null values are a prerequisite for the application of some algorithms. These algorithms become applicable as the data mining step because of the learning in the preprocessing. Preprocessing is a field of greatest potential to the scientific community in KDD.

The third obstacle, the selection of the appropriate algorithm for a data mining task has long been on the research agenda of machine learning. The main problem is that nobody has yet been able to identify reliable rules predicting when one algorithm should be superior to others. Beginning with the *Mlt-Consultant* [SOD89] there was the idea of having a knowledge-based system support the selection of a machine learning method for an application. The *Mlt-Consultant* succeeded in differentiating the nine learning methods of the Machine Learning Toolbox with respect to specific syntactic properties of the input and output languages of the methods. However, there was little success in describing and differentiating the methods on an application level that went beyond the well known classification of machine learning systems into classification learning, rule learning, and clustering. Also, the European *Statlog*-Project [MST94], which systematically applied classification learning systems to various domains, did not succeed in establishing criteria for the selection of the best classification learning system. It was concluded that some systems have generally acceptable performance. In order to select the best system for a certain purpose, they must each be applied to the task and the best selected through a test-method such as cross-validation. Theusinger and Lindner [TL98] are in the

[2]Specialized on feature generation and selection, the toolbox YALE offers the opportunity to try and test diverse feature sets for learning with the support vector machine [FKMR02]. However, the YALE environment does not access a database.

process of re-applying this idea of searching for statistical dataset characteristics necessary for the successful applications of tools. An even more demanding approach was started by Engels [Eng96]. This approach not only attempts to support the selection of data mining tools, but to build a knowledge-based process planning support for the entire knowledge discovery process. To date this work has not led to a usable system [ELS97]. The European project *MetaL* now aims at learning how to combine learning algorithms and datasets [Bra98]. Although successful in many respects, there is not enough knowledge available in order to propose the correct combination of preprocessing operations for a given dataset and task. The *IDEA* system now tries the bottom-up exploration of the space of preprocessing chains [BHP02]. Ideally, the system would evaluate all possible transformations in parallel, and propose the most successful sequence of preprocessing steps to the user. For short sequences and few algorithms, this approach is feasible. Problems like the collection of all data concerning one customer (or patient) from several tables, or the generation of most suitable features enlarge the preprocessing sequences considerably. Moreover, considering learning algorithms in preprocessing steps enlarges the set of algorithms per step. For long sequences and many algorithms this principled approach of *IDEA* becomes computationally infeasible.

1.2 Basic Ideas in MiningMart

If the pairing of data and algorithms is all that difficult, can we support an application developer at all? The difficulty of the principled approaches to algorithm selection is that they all start from scratch. They apply rules that pair data and algorithm characteristics, or plan a sequence of steps, or try and evaluate possible sequences for each application anew. However, there are similar applications where somebody has already done the cumbersome exploration. Why not use these efforts to ease the new application development? Normally, it is much easier to solve a task if we are informed about the solution of a similar task. This is the basic assumption of case-based reasoning and it is the basic idea behind MiningMart. It enables users to share knowledge about KDD processes by exchanging successful applications (called *cases*) of such processes.

In order to exchange successful knowledge discovery cases, a formalism to describe them must be found. This formalism should abstract away from unnecessary details of a given solution (a given case). To this end, a *conceptual level* that describes the data using common everyday notions is introduced. All data transactions are described on this level. Of course, this conceptual level must be mapped to the actual data. The latter is described at a *relational level*. Both levels contain meta-information about the data; in other words, MiningMart is based on metadata, also called *ontologies*.

The next step is to implement *operators* that perform data transformations such as discretization, handling null values, aggregation of attributes into a new one, or collecting sequences from time-stamped data. The operators directly access the database and are capable of handling large masses of data. The oper-

ators are applied by the user on the conceptual level. The specifications of these operators are described using the same formalism as for the data. In addition, the particular sequence of operators as they were applied in a given KDD solution (a case) is also described in this formalism. In other words, there is an integrated data model for the metadata and for the description of the solution. It is called M4, the MiningMart MetaModel; it is stored in the same relational database as the data itself.

From the meta-descriptions of the operator chains, the MiningMart compiler creates SQL code that performs the data processing steps as specified by the chains.

With this equipment it is possible to develop and collect successful cases of knowledge discovery. Since most of the time is used to find chains of operator applications that lead to good answers to complex questions, it is cumbersome to develop such chains over and over again for very similar discovery tasks and data. Currently, in practice even the same task on data of the same format is implemented anew every time new data are to be analyzed. Therefore, the re-use of successful cases speeds up the process considerably. This is one of the main benefits of MiningMart: the metadata for each case is stored for future usage and for exchange of cases. This allows the re-use of cases.

1.3 Overview of this document

This document is the final report on the MiningMart project. It aims to relate the main parts of MiningMart to each other and give an overview of the technical goals achieved during the project, as well as an outlook into the future. Whereever relevant, it contains pointers to more detailed documents that were written during the MiningMart project, such as technical reports and deliverables. A complete list of these documents can be found in the appendix at the end of this document. The pointers refer to this list.

Chapter 2 describes the MiningMart system, which is the main achievement of the MiningMart project. Its various parts are presented in some detail. The second achievement of the project is the collection of successful knowledge discovery cases whose conceptual data are publicly available on the MiningMart webpages. An overview over these cases is given in chapter 3.

Chapter 4 describes the plans for the exploitation and dissemination of the system. Finally, the last chapter summarizes the work done in MiningMart and relates it to other relevant scientific work.

Chapter 2

The MiningMart System

Figure 2.1 shows an overview of the different parts of MiningMart. At the heart of MiningMart we find the Metamodel M4, which is motivated and described in section 2.1. Its operator-related parts are made executable by the M4 Compiler, as explained in section 2.2. The different parts of M4 can be filled with metadata using the editors which form the Human-Computer Interface of MiningMart, described in section 2.3. Finally, there is an internet access to the successful cases which have been collected by MiningMart users. Section 2.4 explains this in detail.

2.1 M4 – The MiningMart MetaModel

Before presenting M4 in more detail, we list the main advantages that the use of metadata gives to MiningMart. Ontologies or meta-data have been a key to success in several areas. They allow:

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

Data documentation: All attributes together with the database tables and views, which are input to a preprocessing chain are explicitly listed at both, the conceptual and relational part of the meta-data level. An ontology allows to organize all data by means of inheritance and relationships between concepts. For all entities involved, there is a text field for documentation. This makes the data much more understandable, e.g. by human domain experts, than just referring to the names of specific database objects. Furthermore, statistics and important features for data mining (e.g., presence of null values) are accessible as well. This extends the common use of meta-data in relational databases and gives a good impression of the data sets at hand.

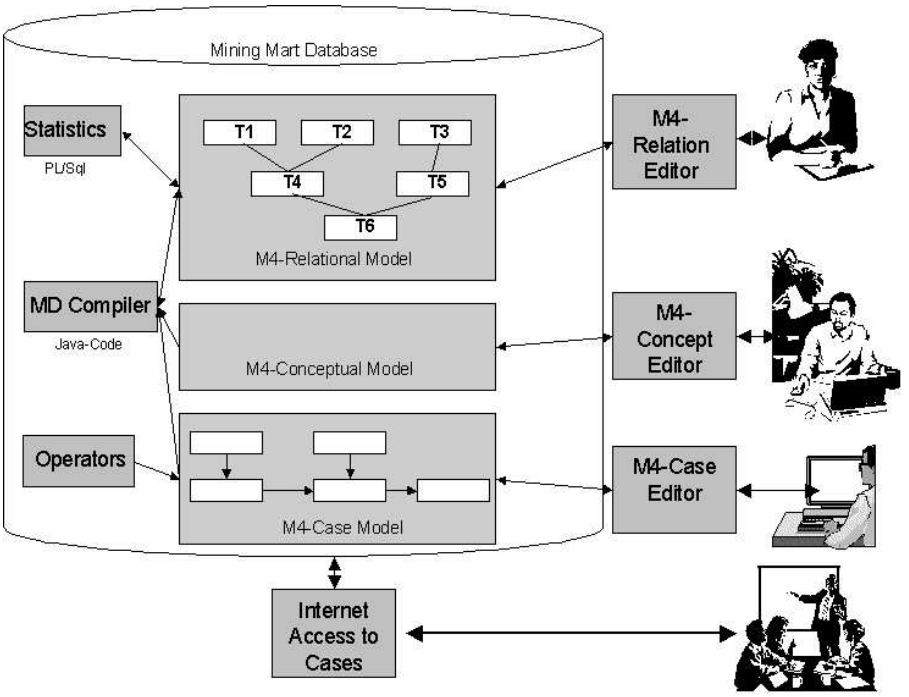

Figure 2.1: Overview of the MiningMart system

Case documentation: The chain of preprocessing operators is documented, as a result of storing its metadata: the declarative definition of an executable case in the M4 model can already be considered a documentation. Furthermore, apart from the opportunity to use self-explanatory names for steps and data objects, there are text fields to document all steps of a case together with their parameter settings. This helps to quickly figure out the relevance of all steps and makes cases reproducable. In contrast, the current state of documentation is most often the memory of the particular scientist who developed the case.

Ease of case adaptation: In order to run a given sequence of operators on a new database, only the relational meta-data and their mapping to the conceptual meta-data has to be written. A sales prediction case can for instance be applied for different kinds of shops, or a standard sequence of steps for preparing time series for a specific learner might even be applied as a template in very different mining contexts. The same effect eases the maintanance of cases, when the database schema changes over time. The user just needs to update the corresponding links from the conceptual to the relational level. This is especially easy when all abstract M4 entities are documented.

Figure 2.2 shows a simplified UML diagram of the M4 model. The remainder of this section explains how M4 works and how it provides the advantages listed above to the MiningMart system.

M4 is structured along two dimensions, topic and abstraction. The *topic* is either the data or the case. The data are the ones to be analyzed. The case is a sequence of (preprocessing) steps. The *abstraction* is either conceptual or relational. Where the conceptual level is expected to be the same for various applications, the relational level actually refers to the particular database at hand. The conceptual data model describes concepts like Customer and Product and relationships between them like Buys. The relational data model describes the business data that are analyzed. Most often it already exists in the database system in the form of the database schema. The meta-data written in the form as specified by M4 are stored in a relational database themselves.

As figure 2.2 shows, each case contains steps, each of which embeds an operator an parameters. Apart from values, not shown here, parameters may be concepts, base attributes, or a multi column feature (a feature containing multiple base attributes). This part is a subset of the conceptual part of M4. The relational part contains columnsets and columns. Columnsets either refer to database tables, or to virtual (meta-data only) or database views. Each columnset consists of a set of columns, each of which refers to a database attribute. On the other hand columns are the relational counterpart of base attributes. For columns and base attributes there is a predefined set of data types, which is omitted in Figure 2.2.

A complete technical description of M4 can be found in the technical report TR12-05 of the MiningMart project.

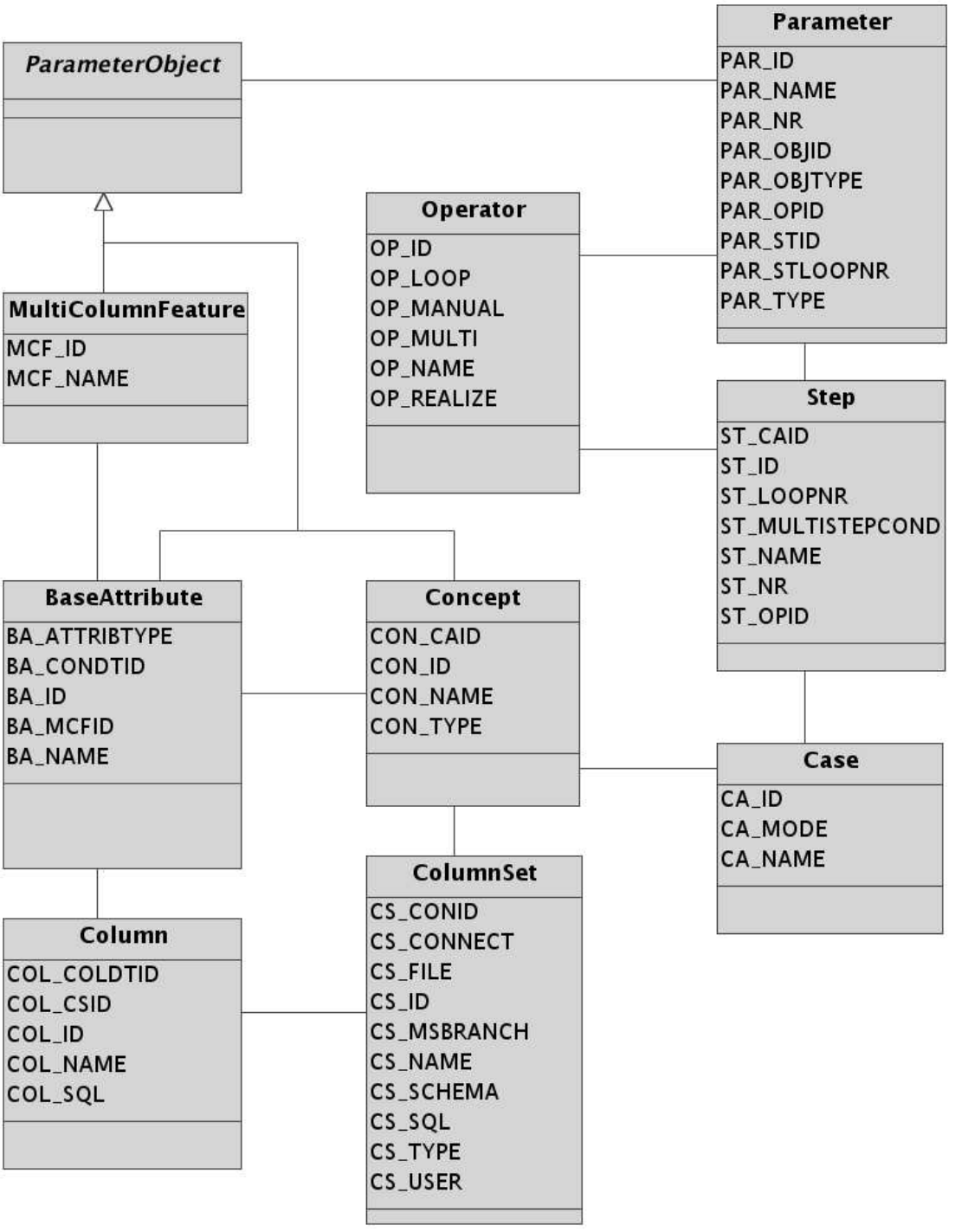

Figure 2.2: Simplified UML diagram of the MiningMart Meta Model (M4)

2.1.1 The Conceptual Data Model

As depicted in Figure 2.1, there are different kinds of experts working at different ends of a knowledge discovery process. First of all a domain expert will define a conceptual data model, using a concept editor. The entities involved in data mining are made explicit by this expert. The conceptual model of M4 is about *concepts* having *features*, and *relationships* between these concepts.

Examples for concepts are Customer and Product. Although at the current stage of development concepts refer to either database views or tables, they should rather be considered as part of a more abstract model of the domain. Concepts consist of features, either base attributes or multi column fetures. A base attribute corresponds to a single database attribute, e.g. the name of a customer. A multi column feature is a feature containing a fixed set of base attributes. This kind of feature should be used, when information is split over multiple base attributes. An example is to define a single multi column fetaure for the amount and the currency of a bank transfer, which are both represented by base attributes.

Relationships are connections between concepts. There could be a relationship named Buys between the concepts Customer and Product, for example. At the database level one-to-many relationships are represented by foreign key references, many-to-many relationships make use of cross tables. However, these details are hidden from the user at the abstract conceptual level.

To organize concepts and relationships the M4 model offers the opportunity to use inheritance. Modelling the domain in this fashion, the concept Customer could have *subconcepts* like Private Customer and Business Customer. Subconcepts inherit all features of their superconcept. The relationship Buys could for instance have a subrelationship Purchases on credit.

2.1.2 The Relational Model

Given a conceptual data model, a database administrator maps the involved entities to the corresponding database objects. The relational data model of M4 is capable of representing all the relevant properties of a relational database. The most simple mapping from the conceptual to the relational level is given, if concepts directly correspond to database tables or views. This can always be achieved manually by inspecting the database and creating a view for each concept. However, more sophisticated ways of graphically selecting features in the database and aggregating them to concepts increase the acceptance by end users and ease the adaptation of cases to other environments. In the MiningMart project, the relational editor is intended to support this kind of activity. In general it should be possible to map all reasonable representations of entities to reasonable conceptual definitions. A simple mapping of the concept Customer, containing the features Customer ID, Name, Address to the database would be to state that the table CUSTOMER holds all the necessary attributes, e.g. CUSTOM_ID, CUST_NAME and CUST_ADDR. Having the information about name and address distributed over different tables (e.g. sharing the key attribute CUS-

TOM_ID) is an example for more complex mappings. In this case the relation editor should be able to use a join operation.

2.1.3 The Case Model

The task of a case designer, ideally a data mining expert, is to find sequences of steps resulting in a representation well suited for the given data mining task. This work is done in the case editor (see section 2.3). To support the case designer a list of available operators and their overall categories, e.g. feature construction, clustering or sampling is part of the conceptual case model M4. The idea is to offer a fixed set of powerful pre-processing operators, in order to offer a comfortable way of setting up cases on the one hand, and ensuring re-usability of cases on the other. By modeling real world cases in the scope of the project further useful operators will be identified, implemented and added to the repository.

For each step the case designer chooses an applicable operator from the collection, sets all of its parameters, assigns the input concepts, input attributes and/or input relations and specifies the output. To ease the process of editing cases, applicability constraints on the basis of meta-data are provided as formalized knowledge and are automatically checked by the human computer interface. This way only valid sequences of steps can be produced by a case designer.

A sequence of many steps, namely a *case* in M4 terminology, transforms the original database into another representation. Each step and their ordering is formalized within M4, so the system is automatically keeping track of the performed activities. This enables the user to interactively edit and replay a case or parts of it.

2.2 The Compiler

All the information about the conceptual descriptions and about the according database objects involved are represented within the M4 model and stored within relational tables. M4 *cases* denote a collection of steps, basically performed sequentially, each of which changes or augments one or more concepts. Each step is related to exactly one M4 *operator*, and holds all of its input arguments. The M4 *compiler* reads the specifications of steps and executes the according operator, passing all the necessary inputs to it. This process requires the compiler to translate the conceptual entities, like input concepts of a step, to the corresponding relational entities, like database table name, the name of a view or the SQL definition of a virtual view, which is only defined as relational meta-data in the M4 model.

Two kinds of operators are distinguished, manual and machine learning operators. Manual operators just read the M4 meta-data of their input and add an SQL-definition to the meta-data for their output, establishing a virtual table. Currently, the MiningMart system offers 20 manual operators for selecting rows, selecting columns, handling time data, and generating new columns for the pur-

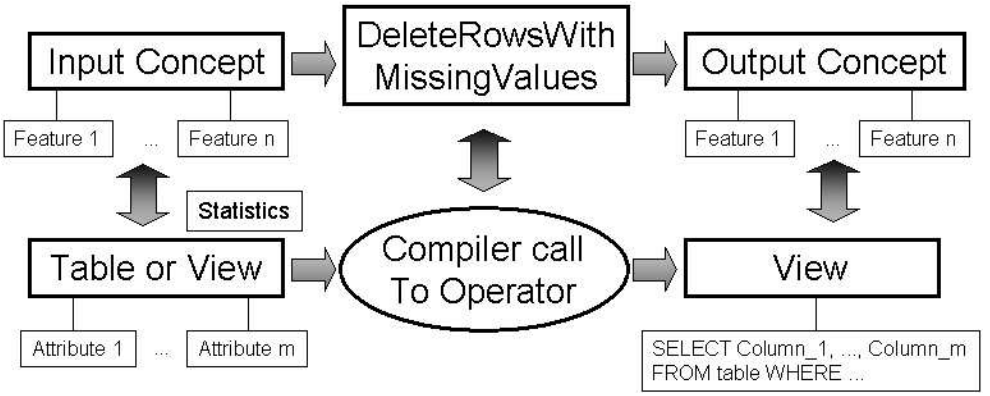

Figure 2.3: An illustration of the coupling of the abstract conceptual and executable level.

poses of, e.g., handling null values, discretization, moving windows over time series, gathering information concerning an individual (e.g.,customer, patient, shop).

External machine learning operators on the other hand are invoked by using a wrapper approach. Currently, the MiningMart system offers learning of decision trees, k-means, and the support vector machine as learning preprocessing operators[1]. The necessary business data are read from the relational database tables, converted to the required format and passed to the algorithm. After execution the result is read by the wrapper, parsed, and either stored as an SQL-function, or materialized as additional business data.

In any case the M4 meta-data will have to be updated by the compiler. A complex machine learning tool to replace missing values is an example for operators altering the database. In contrast, for operators like a join it is sufficient to *virtually* add the resulting view together with its corresponding SQL-statement to the meta-data.

Figure 2.3 illustrates, how the abstract and the executable or relational level interact. First of all just the upper sequence is given, an input concept, a step, and an output concept. The concept definitions contain features, the step contains an operator together with its parameter settings. Apart from operator specific parameters, the input and output concept are parameters of the step, as well. The compiler needs the inputs, e.g. the input concept and its features to be mapped to relational objects before execution. The mapping may either be defined manually, using the relation editor, or it may be a result of executing the preceeding step. If there is a corresponding relational database object for each input, then the compiler executes the embedded operator. In the example this is a simple operator named "DeleteRowsWithMissingValues". The corresponding

[1]Of course, the algorithms may also be used in the classical way, as data mining step operators.

executable part of this operator generates a view definition in the database and in the relational meta-data of M4. The latter is connected to the conceptual level, so that afterwards there is a mapping from the output concept to a view definition. The generated views may be used as inputs to subsequent steps, or they may be used by other tools for the data mining step.

Following the overall idea of declarative knowledge representation of the project, known pre-conditions and assertions of operators are formalized in the M4 schema. Conditions are checked at runtime, before an operator is applied. Assertions help to decrease the number of necessary database accesses, because necessary properties of the data can be derived from formalized knowledge, saving expensive database scans. A step replacing missing values might be skipped, for instance, if the preceding operator is known not not produce any missing values. If a user applies linear scaling to an attribute, then all values are known to lie in a specific interval. If the succeeding operator requires all values to be positive, then this pre-condition can be derived from the formalized knowledge about the linear scaling operator, rather than to recalculate this property by another database scan.

2.3 The Human-Computer Interface (HCI)

The main task of the HCI is to support the creation of metadata (concepts and features as well as operator chains) and the execution of the operator chains. To this end, there are two editing windows in MiningMart which correspond to different parts of M4. The *concept editor* allows to create, edit and delete M4 objects on the conceptual level; it also creates M4 objects on the relational level where necessary and allows to link the two levels. The *case editor* allows to set up a chain (or a directed acyclic graph) of data processing steps.

Figure 2.4 shows a screenshot of the concept editor, while it is used to list and edit base attributes. The right part of the lower window states, that the selected concept Sales Data is connected to another concept Holidays by a relationship week has holiday.

The concept editor also allows for each object on the conceptual level to map it to an object on the relational level (not shown). Apart from this, it offers a data viewer and is capable of displaying statistics of connected views or tables. Figure 2.5 shows an example of the statistics displayed. For each view or table the number of tuples and the numbers of nominal, ordinal and time attributes are counted. For numerical attributes the number of different and missing values is displayed, the minimum, maximum, average, median and modal value are calculated together with the standard deviation and variance. For ordinal and time attributes the most reasonable subset of this information is given. Finally we have information on the distribution of the values for all attributes.

Editing sequences of steps is done in the *case editor*. Figure 2.6 shows a screenshot of a rather small example case edited by this tool. Typically a pre-processing chain consists of many different steps, usually organized as a directed

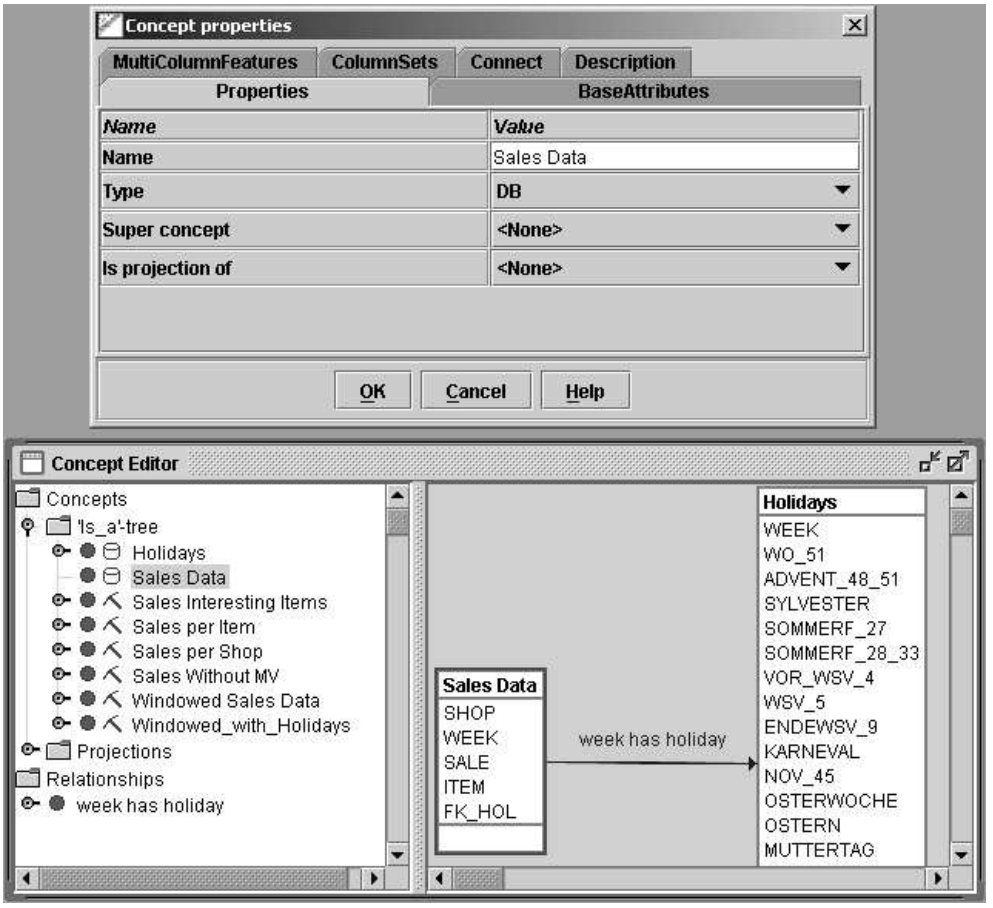

Figure 2.4: The Concept Editor

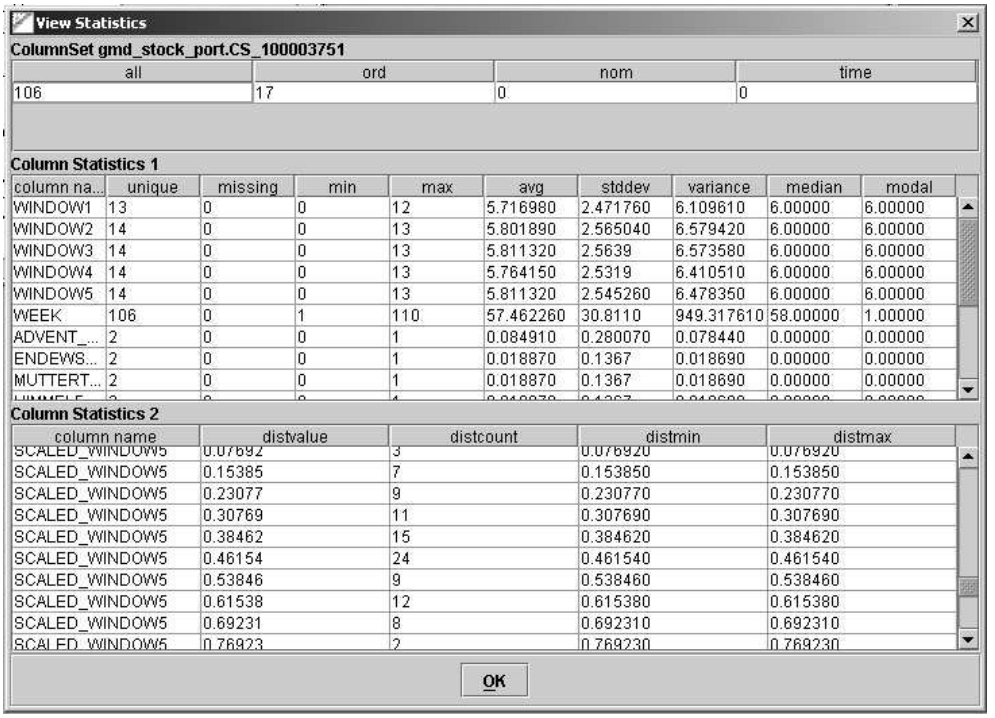

Figure 2.5: Statistics of a database view

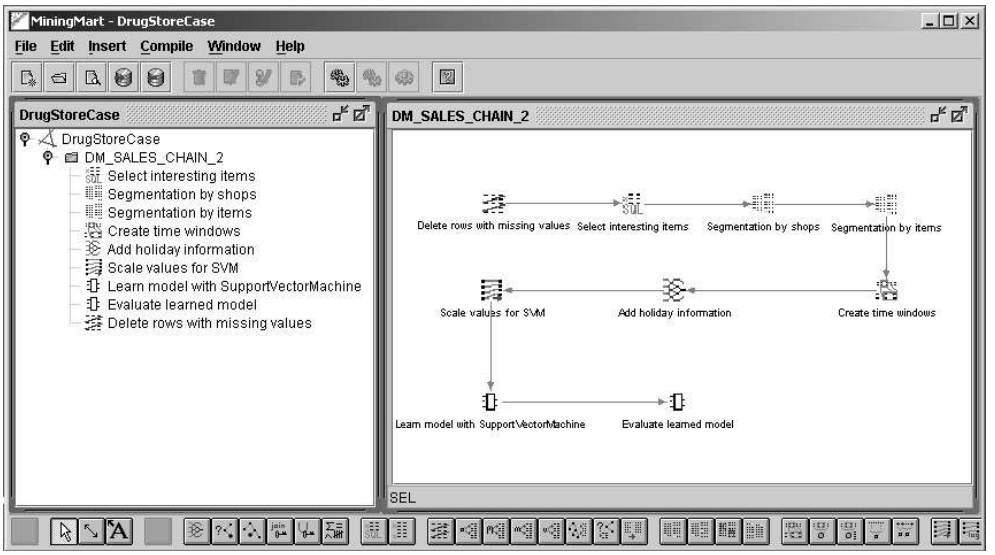

Figure 2.6: A small example case in the case editor.

acyclic graph, rather than as a linear sequence as the example case shown. The case editor gives additional support to the user by automatically defining output concepts of steps according to the meta-data constraints, and by offering property windows tailored to the demands of chosen operators.

More details on how to use the HCI can be found in the MiningMart User-Guide.

2.4 The Web Platform

As soon as an efficient chain of preprocessing has been found, it can easily be exported and added to an Internet repository of best-practice MiningMart cases. Only the conceptual meta-data is submitted, so even if a case handles sensitive information, as is true for most medical or business applications, it is still possible to distribute the valuable meta-data for re-use, while hiding all the sensitive data and even the local database schema.

2.4.1 The Case Base

One of the project's objectives was to set up a case-base of successful cases on the Internet. The shared knowledge allows all Internet users to benefit from a new case. Submitting a new case of best practice is a safe advertisement for KDD specialists or service providers, since the relational data model is kept private. To support users in finding the most relevant cases, their inherent structure

is exploited. An Internet interface is accessible that visualizes the conceptual meta-data. It is possible to navigate through the case-base and to investigate single steps, to see which operators were used on which kind of concepts. The Internet interface reads the data directly from the M4 tables in the database, avoiding additional efforts and redundancies.

2.4.2 The Business Level

Additionally to the data explicitly represented in M4, a business level has been added. This level aims at relating the case to business goals and to give several kinds of additional descriptions, like which success criteria were important for the case. Figure 2.7 shows a screenshot of a case's business level description. For instance, the sales prediction answers the question "How many sales of a particular item do I have to expect?" where the business goal is that it must not happen that the item is sold out, but the stock should be minimized. A particular application need is that the forecast can only be used if it predicts the sales 4 weeks ahead because of delivery times. Especially the more informal descriptions should help decision makers to find a case tailored for their specific domain and problem. The additional information is stored in an XML-representation, directly connected to the M4 entities. On the Internet these connections are represented by hyperlinks. Figure 2.8 shows the ontology of the business level.

It is possible to start the search for a case at each category of the business level or conceptual level. In this sense the cases are indexed by all the categories part of the conceptual M4 model and the business model. If a user considers a case useful, then its conceptual data can be downloaded from the server. The downloadable case itself is a category in the XML framework. The locally installed MiningMart system offers an import facility for installing the meta-data into the user's M4 tables. If problems arise, or further help is necessary, the business level holds a category for the case designer or the company providing service.

The project has developed four cases, which are described and downloadable in the internet case base except for the first one:

- analysis of insurance data for direct mailing [KZV00, KZFB00],
- call center analysis for marketing (see MiningMart deliverable 17.2b),
- analysis of data about calls and contracts for fraud detection in telecommunication (see MiningMart deliverable 17.2a), and
- analysis of sales data for sales prediction [Rue99].

More on these cases can be found in chapter 3. The web address for the case base is:
`http://kissen.cs.uni-dortmund.de:8080/mmart/index.html`

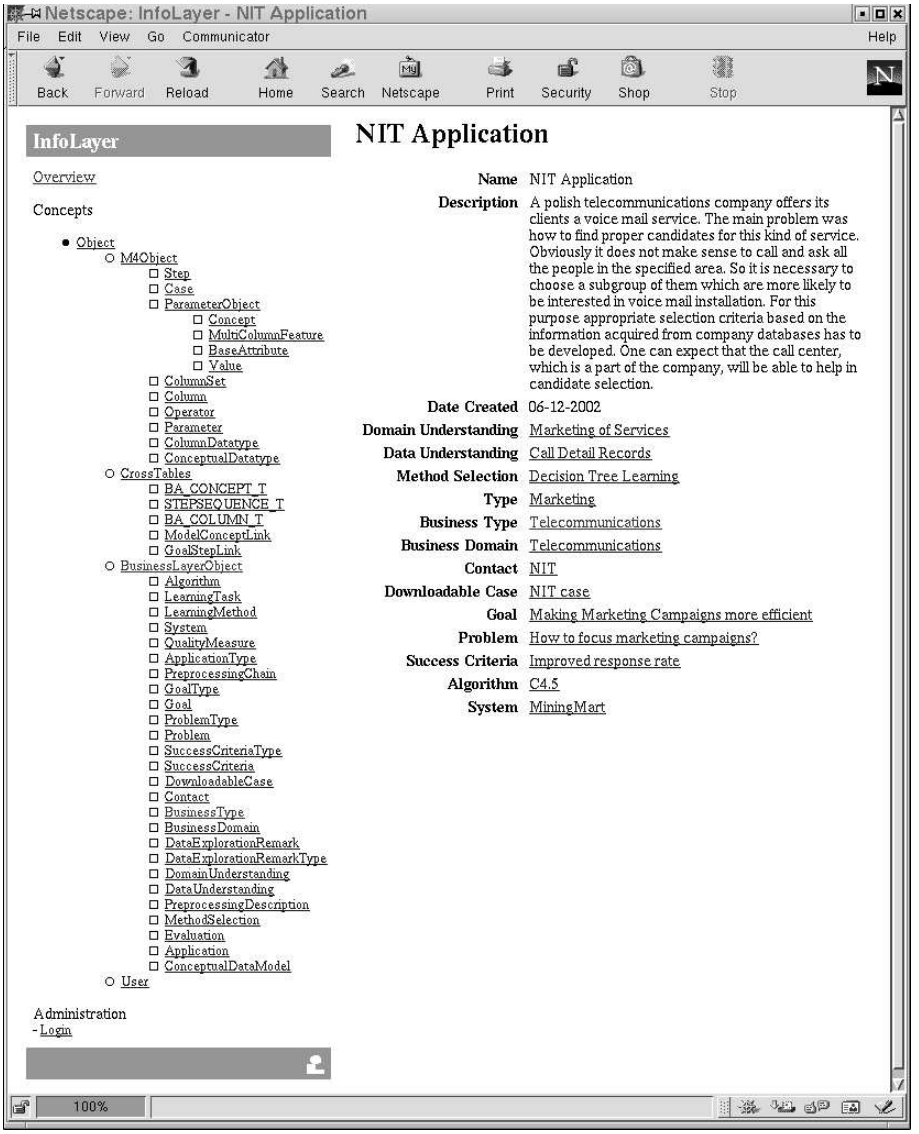

Figure 2.7: The Internet interface to the case base visualizes all cases, their steps, embedded operators, and parameters in HTML format. Entities related in the M4 schema are connected by hyperlinks. Additionally, a business level is part of the interface. It describe the available cases in terms like the addressed business goals of the data analysis. After deciding for a case with the help of conceptual M4 and business layer descriptions, the user can simply download the one addressing the most similar problem. The case adaption facilities of The MiningMart system helps to quickly adjust the case to the user's environment.

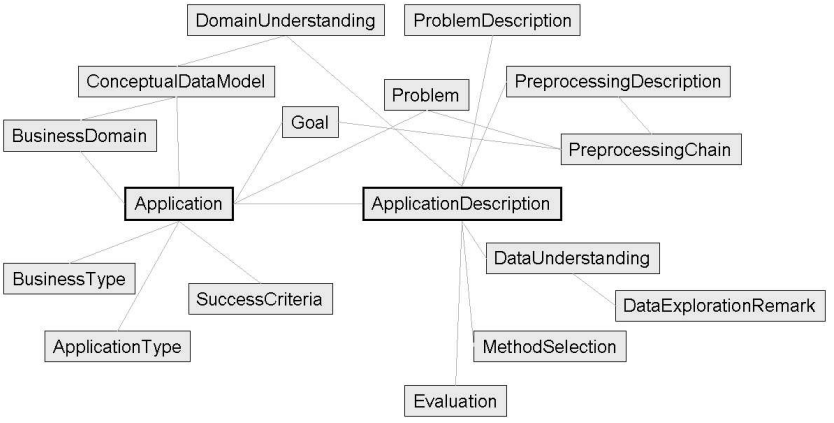

Figure 2.8: The ontology of the business layer, used to describe M4 cases in business terms.

Chapter 3

Cases and Evaluations

The MiningMart system was used during the project time to model two big industrial knowledge discovery applications. The result are two executable cases that demonstrate the usability and usefulness of the MiningMart system. These cases are published in the internet case base (see section 2.4). They are described in detail in the MiningMart deliverable 17.2 (parts a and b). Further, it was evaluated how well the system lended itself for the development of these cases and in how far it speeded up the modelling process. These evalutations are described in detail in the MiningMart deliverable 17.3 (parts a and b). In addition, a smaller case that was developed at the University of Dortmund was modelled during the project and used as a testbed.

This chapter gives an overview of these cases and the evaluations of MiningMart that are based on them. For details, please refer to the deliverables or the information in the internet case base.

3.1 The drug store case

This is the smallest case; it was developed and applied at the University of Dortmund before the MiningMart project and was later modelled as a MiningMart case as a testbed for the development of the system. The application is based on data from a drug store chain. For 20 shops of this chain and about 10000 items they sell, the sales data for every week of a two year period is given. The data mining task is to predict the sales of a specific item in a specific shop for a given week, either the next week or a few weeks ahead.

The input data is given in a single table containing one column each for the shop, item, week and number-of-sales information. The case developers decided that a Support Vector Machine (SVM) algorithm should be applied in the data mining step. To do this, the data has to be brought into a suitable format. Specifically, the SVM algorithm can not deal directly with the time series format (sales per week) in which the data is given. Therefore the data is transformed using a windowing method, such that in the resulting table, each row contains a

number of successive sales values for a particular item; thus each row can serve as a training example for the SVM.

Since the task is to predict the sales for a particular item in a particular shop, the input data is segmented into shops and items before the application of the windowing operation. Further, only a few of the 10000 items are of interest.

It turned out during the development of the case that the prediction performance of the SVM can be significantly improved if additional time-related information is provided. The drug store sales numbers are different in different seasons of the year; for many items, they are highest in the weeks before Christmas, but this is not true for items like sun tan lotion, for example. To be able to benefit from such regularities, information about the public holidays during the year was encoded by the case developers in a binary format, such that for each week, the presence or absence of a particular public holiday like Christmas is indicated in a separate database table. A foreign key link between the sales data and the holiday data exists.

We now have all the necessary information to describe the drug store case step by step.

Step 1 There are some missing values in the input data in the `sales` column. All rows missing such values are removed. This is done using the MiningMart operator `DeleteRowsWithMissingValues`.

Step 2 The items of interest (here, 3 items with certain Ids) are selected. After this step, only these 3 items are in the data that is further processed.

Step 3 The data is segmented such that each segment contains the data about one of the 20 shops. The MiningMart operator `SegmentationStratified` does this, but it is only done on the relational level. Thus, there is only one output concept of this step which has 20 segments as 20 columnsets mapped to it.

Step 4 The data is segmented further such that each segment contains the data about one of the three items selected in step 2 and about one of the shops. Thus there are now 60 segments (columnsets) attached to the output concept of this step. The data mining step will be applied to each of them because the task was to predict the sales for each item and each shop separately.

Step 5 This is the windowing step. In the output of this step, each row contains as many sales values for the item as the parameter `WindowSize` for the operator `Windowing` determines.

Step 6 Now the windowed data is joined with the holiday information from the other input table, such that the binary holiday information is included for the last value in the window (which is the target value for the learning task).

Step 7 The SVM algorithm works better on values that are scaled to the same range. This step applies the MiningMart operator `LinearScaling` in a loop to all BaseAttributes of the input so that the scaled versions can be input for the SVM.

Step 8 The SVM algorithm is applied in regression mode. It is told what values to learn from (parameter `ThePredictingAttributes`) and what is the target value (parameter `TheTargetAttribute`). The operator creates a BaseAttribute called `PredictedSales` which contains the SVM-predicted values.

Step 9 The predicted values are compared to the actual values in this last step. This operator is a pseudo-operator because it does not have any output on the conceptual level; it just prints out an error value when executed.

For details on the case, please refer to the internet case base (see section 2.4).

3.2 The call center case

This case was developed by the National Institute of Telecommunications at Warsaw, Poland, 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.

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.

Evaluation In the evaluation of the MiningMart system, which was done based on this case, the National Institute of Telecommunications (NIT) points out that the documentation of a preprocessing case that MiningMart enforces was especially important to them, because they often re-use cases. Further, the application of pre-defined operators avoids many programming errors. NIT points 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. Finally, the easy-to-understand graphical interface and chain concept were valued highly by NIT.

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 section 2.4).

3.3 The churn prediction case

This case was developed at the Telecom Italia Labs. The term "churn" refers to a customer ending their contract with the telecom. The task in this case was to

predict a segment of customers likely to do a churn in the near future.

The input data for this case included again *call detail data*, but aggregated on a monthly basis already, and *customer data*. The call detail data is available for five months in the data sample available for MiningMart; the task is to predict churn for the sixth month. The target attribute for learning is in a third data table which provides the *contract information* for each customer. Finally, another input table contains the *revenues* that the customers generated in the five months' sample.

Again the main tasks that this preprocessing case models are described in an overview below.

Task 1: Handle missing values Missing values in the call detail data are replaced by the average value in the rest of the data. However, this is done for every customer and every type of call separately, in order to get sensible average values. Therefore this task consists of several steps which first segment the input data according to customer and type of call, then assign the average value to missing values and finally unsegment the data.

Task 2: Create relational representation In the call detail table as well as the revenues table, data is stored on a monthly basis for each customer. However, for data mining it is useful to have only one record per customer. Therefore the data in the two tables are aggregated in this task in a way similar to task 1 in the call center case (section 3.2).

Task 3: Derive additional features Further attributes are computed for the data mining step, such as the sum of all call lengths for each month, the difference in call length to the latest month for each month (which could indicate an unusual rise in call length), and discretized versions of the revenue and length of subscription attributes. Also only certain tariff plans are selected because the others were not of interest to the telecom company.

Task 4: Learn prediction model Separating customers according to the discretized value of their revenues, a decision tree operator is applied to random samples of the data to predict the churn behaviour of each customer for the sixth month. The predicted and the actual value of the churn attribute are now available in the same table and can be compared.

Evaluation The Telecom Italia Labs (TILab) evaluated the MiningMart system using this case as a basis. The evaluation was done under the following criteria.

- Usability: TILab found the usability of the system to be on a par with leading commercial tools. Especially the option to subsume a number of steps into chains and the import/export functionality were valued highly.

- Mining Process Speed-up: TILab points out that operator constraints and the visualization of the preprocessing chain allow a fast development of correct chains. Testing a chain is always possible during its development. Modelling the above case took five man-days of work with MiningMart instead of 24 without. When a case is re-used, speed-up is even more significant.

- Scalability: TILab measured the time needed for preprocessing the data using the chain described above for different sizes of datasets. Preprocessing time was found to scale linearly with respect to the amount of data involved.

- Mining Process Quality: It was pointed out by TILab that due to the application of pre-defined operators, programming errors are excluded, while they are bound to occur when SQL programming is used. Further, conceptual mistakes are avoided by the visualization of the preprocessing chain, the possibility to inspect the data at any stage, and the constraints on the operators.

For details on the case and its evaluation, please refer to the deliverables 17.2 (a) and 17.3 (a) and to the internet case base (see section 2.4).

Chapter 4

Exploitation and Dissemination

The main goal of the MiningMart project was to create a running system which demonstrates the usefulness and feasibility of the basic ideas. While from the system architectural view, the core of MiningMart might be seen to be the metamodel and the compiler that makes it executable, the basic application scenario is that of sharing knowledge, i.e. cases, over the internet between many users. Therefore the internet case base, with the application-sharing possibilities that it offers (see section 2.4), is the main entry point for new users and a fundamental tool for the application of MiningMart. Thus the MiningMart web pages, which present the system and its possibilities to new users, are of particular importance to the future of the project and are described in section 4.1.

Regarding the future plans for the exploitation and dissemination of MiningMart, the internet presence is one of the main ways to raise interest in and knowledge about MiningMart. Another was a workshop held in Dortmund on the 18th of February 2003, which is described in deliverable 11.1 and summarized below in section 4.2. The third dissemination activity is the exploitation by partners of the project, who plan to apply MiningMart in a number of ways, which is expected to bring *their* clients and project partners into touch with MiningMart. These activities will also lead to new cases in the public case base, which will enhance the basis for new applications. More on these plans can be found in section 4.3. In addition, MiningMart has produced scientific publications which will be published in 2003 and which will raise publicity in the academia (see the appendix).

4.1 The MiningMart Websites

MiningMart addresses different kinds of users: those who work more in a business-related way and others who are closer to the data processing of an institution. The web pages offer different aspects of the system for the different users.

MiningMart addresses knowledge discovery applications. It wants to share successful applications between all kinds of users. Therefore one main goal for the web pages is to explain what knowledge discovery in databases is to users who have no experiences with this. There is a *User* section of the web pages which achieves this. Since the internet case base contains examples for knowledge discovery applications, it can be used to demonstrate the basic concepts. Thus all sections of the web pages are linked to the case base. However, the user section also provides a non-technical overview of how the MiningMart system works.

MiningMart was also a research project in the field of knowledge discovery in databases. For users with a research background, there is a *Research* section of the web pages which explains the basic ideas in MiningMart from a scientific perspective.

Of course, all *publications* related to MiningMart can be found on the web pages as well as the *downloadable system* itself together with the user guide. A *link* section contains pointers to related projects and information about KDD. Finally, as mentioned before, the *case base* can easily be reached from the web pages.

4.2 The One Day Seminar

The "*One Day Seminar – Data Mining in Practice*" was carried out by and held at the University of Dortmund on February 18th, 2003. The objective of the seminar was to present the results of the project and to attract contributions to the case-base. In preparation to this workshop it was announced at conferences and on several mailing lists for KDD, Machine Learning and neighbouring disciplines. There were about 70 registrations from the industrial and scientific sector. The programme consisted of a variety of presentations by industrial and academic partners that illustrated the benefit of using the MiningMart system for real world applications in KDD. A system demo was included.

The programme and the list of participants of this workshop can be found at the following web address:
`http://mmart.cs.uni-dortmund.de/content/oneDaySeminar.html`

Several attendants from the industrial sector expressed their interest in using the MiningMart system, though so far no prolonged cooperation has resulted from this. As the publically available repository of best-practice cases grows, we expect a higher interest from industrial side and valuable feedback from end users for the future.

4.3 Exploitation Plans

The plans for the exploitation of MiningMart can be divided into four different kinds of activity which are explained in the following sections.

4.3.1 Application Service Providing

While the MiningMart system is available to the public, it is yet possible to make profits from its application by using own expertise on knowledge discovery processes and selling this expertise. Using MiningMart enables the service provider to come up with new applications faster and re-use older applications easily, so that more revenue is gained than without MiningMart. Thus what is sold is not knowledge about MiningMart, which is freely available on the MiningMart web pages, but knowledge about Data Mining, which is developed in the MiningMart framework to benefit from the graphical interface, case documentation and easy re-use.

This kind of activity is pursued by Perot Systems Netherlands.

4.3.2 Research

There is a number of possible research directions in which MiningMart can play a role. MiningMart has shown the benefits of the maintainance of a metadata level in a database-related project. This approach can play a vital role in other database research. The conceptual level of MiningMart serves as an abstraction of given data, which can be used to connect different sources of data and provide a common view of them. This is envisioned in a research project to which the University of Dortmund is contributing. DISTA plan to use the MiningMart framework to develop new data mining technology more easily. Other cooperations are expected after the scientific contributions of MiningMart are published. The next publication in the scope of a conference will shortly be submitted to PKDD 2003 by the University of Dortmund. Further publications can be found in the appendix.

Another focus of the University of Dortmund is to teach students how to use the MiningMart system. Since April 2003 students attending lectures on KDD deepen their understanding by using the system. For October 2003 a one year project group will be offered, where students will have the opportunity to address real world KDD applications applying the MiningMart system.

4.3.3 Embedding MiningMart into other KDD tools

MiningMart has a well-defined interface: the M4 model in a relational database. Its input and output are standard database objects like tables and views. Thus MiningMart can be used in combination with existing tools for database management as well as Data Mining. However, to ease such combinations, the MiningMart system can be embedded into such existing tools, thus extending the range of applications for MiningMart and its versatility. AIS plan to apply MiningMart in the field of spatial data mining by combining it with their platform SPIN. Perot Systems Netherlands also have proprietory KDD tools which they plan to combine with MiningMart. This will support their application service providing activities.

4.3.4 Product Development

During the MiningMart project, the system was developed as a prototype. The current version of the software is publicly downloadable, it runs stable and has been tested on the most common operating systems. However, it is not yet a marketable product. Based on a licensing agreement with all partners, the system can be developed into such a product. Perot Systems Netherlands plan to do this after a period of testing the system in application service providing, depending on whether enough revenues will have been generated and the market situation appears to be promising.

Chapter 5

Related Work and Conclusions

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. The *Clementine* system offers processing chains to users. However, the focus is on the data mining step, not the preprocessing chain. The common data format in tool boxes such as *Spss* or *weka* provides users with the prerequisites to form their own sequences [WF00]. However, the user has to program the sequence and has to repeat this to solve very similar tasks.

Zhong and colleagues have proposed an agent system, *GLS*, which supports the complete KDD process, i.e. preprocessing, knowledge elicitation, and refinement of the result [ZLO01, ZLO97]. In some aspects, this system is similar to MiningMart. Its agents are our operators, its controller corresponds to our compiler, and both systems describe data and operators at the meta level. Where in MiningMart the operator description entails applicability conditions and pointers to the resulting table, in *GLS* the pre- and post-conditions for the application of an agent are stated. The hierarchy of agents in *GLS* corresponds to the inheritance hierarchy of operators as exploited in MiningMart. In addition, MiningMart offers an even more abstract level for the description of a case in business terms. The planning approach of *GLS* is also similar to the use of applicability constraints as done in MiningMart. In contrast to *IDEA*, no comparison of quality is performed for alternative chains of operators. Hence, both MiningMart and *GLS* produce valid sequences of steps, and none of them performs experiments – as does *IDEA* – in order to decide between several algorithms or agents.

Despite the similarities, the two systems do, of course, also differ. First, the set of algorithms (operators or agents, respectively) is different. Feature

generation and selection – a focus of MiningMart – is less developed in *GLS*. Data mining algorithms are less complete in MiningMart. This is no principal point, since both systems allow to extend the set of operators easily. Second, the relation to the database is different. The interaction between *GLS* and the database is not the primary focus of the research in [ZLO01, ZLO97]. In contrast, MiningMart resides to a large degree within the database, compiles metadata into SQL code, and many of its operators are directly integrated into the database. This allows to work on very large databases. Third, the use of human expertise is different. In *GLS*, some user interaction is required in order to optimize the automatically generated valid sequences. However, the notion of a complete case at the meta level is not part of the metamodel. This means that the various trials to establish an optimal sequence of agent activities are not documented. Therefore, experience of failed selections, groupings, or parameter settings is not available and cannot prevent users from trying such settings. By the same token, experience of successful cases is not stored. There is no mechanism to apply a successful chain to similar but different databases, which is one of the strengths of MiningMart. We believe that the re-use of best-practice cases and the case documentation are extremely important.

The recent *Idea* system is also similar to the MiningMart approach [BHP02]. Chains of operators are composed according to a ranking of algorithms in order to detect the best choice of an algorithm given data characteristics. Meta data describing the data as well as the algorithms are used in order to check the validity of operator sequences or incorporate an additional step which allows to apply the best operator. The difference lies first in MiningMart's orientation towards very large databases. *Idea* uses the *weka* data format and, hence, is restricted to smaller files. The data transformations and aggregations incorporated as manual operators in the MiningMart system are not taken into account in *Idea*, because they are not needed in the single table small sample representation of *weka* tools. The second distinction is the approach to finding the best sequence of preprocessing. Although the MiningMart system exploits applicability conditions of operators in order to check the validity of sequences, it does not aim at planning the best sequence or perform a ranking of possible algorithms at each step of an operator chain, as IDEA can do. Instead, MiningMart exploits the findings of expert case designers. Real-world applications of knowledge discovery comprise hundreds of steps in a KDD run (including manual operators) and ranking every algorithm at each of the steps would exhaust computing capacity. We feel that the adaptation of excellently solved KDD problems best combines human expertise and computational power.

We can now summarize the characteristics of MiningMart:

Very large databases: It is a database oriented approach which easily interacts with all SQL databases and scales up to real-world databases without problems. Several operators have been re-implemented in order to scale to very large data sets.

Sophisticated operators for preprocessing: Not only the data mining step,

but also preprocessing steps can make good use of machine learning. For instance, a learning result can be used to replace missing values by the learned (predicted) values. Feature generation and selection in the course of preprocessing can improve the quality of data that are the input to the data minig step.

Metadata driven code generation: MiningMart relies on metadata-driven software generation. Metadata about operators and data is used by the compiler in order to generate a running KDD application.

Case documentation: Metadata about a case also serves to document the overall KDD process with all operator selections and their parameter settings. In addition, a business layer offers the case description in less technical terms so that end-users of the KDD process are kept informed.

Case adaptation: The notion of a complete case in the metamodel allows to apply a given expert solution to a new database. The user only needs to provide the system with a new data model and the compiler generates the new case. For fine-tuning the new application, the human-computer interface allows to easily modifiy the case model with all operators.

Chapter 6

Appendix

6.1 Experiences with Partners from Associated States

As new partners from associate candidate states of Europe, the economical university of Prague (UEP) and the National Institute of Telecommunications (NIT) at Warsaw joined the MiningMart project. At Prague, operators for grouping and discretization have been developed. At Warsaw, the MiningMart system was applied to a knowledge discovery task which is in actual use at the institute. The comparison of handling the task using SAS with handling the task using MiningMart was the first real-world evaluation of the system and was extremely successful (see chapter 3).

The work of these partners is described in the final report of the Ext-MM project, deliverable 20.5.

The collaboration with the new partners was stimulating and productive from the very beginning. Since the new partners did not experience the discussions inside the project we could learn from them how understandable and easy to use the system is. Since one of the partners used the system in the sense of being an end-user (Warsaw), they provided the project with a realistic evaluation. Since the other partner (Prague) enhanced the system with new operators, we could learn how easy the extension of the system is and how we must describe the system in order to ease its further extension. Hence, the partners offered the project an "outsider's view" already during the development time. Of course, the partners ended up with an "insider's view" and became true and acknowledged team members.

In sum, it was a very successful step to include these partners into a european project and their contributions were highly welcome.

6.2 List of documents related to MiningMart

All of the documents listed below can be found on the MiningMart web pages. The address is:
`http://mart.cs.uni-dortmund.de/content/publications.html`

6.2.1 Published Papers

Bredeche, N. and Saitta, L. and Zucker, J.D., *A Wrapper Approach for Robot Visual Perception.* In: ICML Workshop on Machine Learning in Computer Vision, 2002.

Kietz, Jörg-Uwe and Vaduva, Anca and Zücker, Regina, *MiningMart: Metadata-Driven Preprocessing.* In: Proceedings of the ECML/PKDD Workshop on Database Support for KDD, 2001.

Kietz, Jörg-Uwe and Vaduva, Anca and Zücker, Regina, *Mining Mart: Combining Case-Based-Reasoning and Multi-Strategy Learning into a Framework to reuse KDD-Application.* In: Proceedings of the fifth International Workshop on Multistrategy Learning (MSL2000), R.S. Michalki and P. Brazdil (ed.), 2000.

Kietz, Jörg-Uwe, *On the Learnability of Description Logic.* In: Proceedings of the 12th International Conference on Inductive Logic Programming, 2002.

Knobbe, Arno J. and Haas, Marc de and Siebes, Arno, *Propositionalisation and Aggregates.* In: Proceedings of PKDD 2001.

Morik, Katharina and Scholz, Martin, *The MiningMart Approach to Knowledge Discovery in Databases.* In: Handbook of Intelligent IT, Ning Zhong and Jiming Liu (ed.), IOS Press, 2003, to appear.

Morik, Katharina and Scholz, Martin, *The MiningMart Approach.* In: Workshop Management des Wandels der 32. GI Jahrestagung, 2002.

Zücker, Regina and Kietz, Jörg-Uwe, *How to preprocess large databases.* In: Data Mining, Decision Support, Meta-learning and ILP: Forum forPractical Problem Presentation and Prospective Solutions, 2000.

6.2.2 Deliverables

D1: Saitta, Lorenza and Kietz, Jörg-Uwe and Beccari, Giuseppe, *Specification of Pre-Processing Operators Requirements.*

D2.1: Wettschereck, Dietrich and Mueller, Stefan, *MiningMart Deliverable D2.1.*

D3: Morik, Katharina and Liedtke, Harald, *Learning about Time.*

D4.1: Lorenza Saitta, Giuseppe Beccari and Alessandro Serra, *Informed Parameter Setting.*

D4.2: Lorenza Saitta,Marco Botta, Giuseppe Beccari and Ralf Klinkenberg, *Studies in Parameter Setting.*

D5: Arno Knobbe,Adriaan Schipper and Peter Brockhausen, *Domain Knowledge and Data Mining Process.*

D6.2: Kietz, Jörg-Uwe and Fiammengo, Anna and Beccari, Giuseppe and Zücker, Regina, *Data Sets, Meta-data and Preprocessing Operators at Swiss Life and CSELT.*

D7a: Regina Zücker, *Description of the M4-Relational Metadata-Schema within the Database.*

D7b: Regina Zücker, *Description of the Metadata-Compiler using the M4-Relational Metadata-Schema.*

D8/9: Katharina Morik and Marco Botta and Klaus R. Dittrich and Jörg-Uwe Kietz and Luigi Portinale and Anca Vaduva and Regina Zücker, *M4 - The MiningMart Meta Model.*

D9: Stefan Haustein, *Internet Presentation of MiningMart Cases.*

D10: Olaf Rem, *Case Base of Preprocessing.*

D11.1: Martin Scholz, *One Day Seminar – Data Mining in Practice.*

D11.3: Olaf Rem and Marten Trautwein, *Best practices report.*

D12.2: Bert Laverman and Olaf Rem, *Description of the M4 Interface used by the HCI of WP12.*

D12.3: Michael May and Detlef Geppert, *Description of the HCI for Pre-Processing Chains.*

D12.4: Olaf Rem and Erik Darwinkel, *The Concept Editor.*

D13: Jörg-Uwe Kietz, *On the Learnability of Description Logic Programs.*

D14.1: Luigi Portinale and Lorenza Saitta, *Feature Selection.*

D14.3: Timm Euler, *Feature Selection with Support Vector Machines.*

D14.4: Petr Berka and Radim Jirousek and Pavel Pudil, *Feature Selection Operators based on Information Theoretical Measures.*

D15: Peter Brockhausen and Marc de Haas and Jörg-Uwe Kietz and Arno Knobbe and Olaf Rem and Regina Zücker and Nico Brandt, *Mining Multi-Relational Data.*

D16.1: Petr Berka, *Discretization and Grouping operators.*

D17.0: Marco Richeldi and Alessandro Perucci, *Mining data with the MiningMart system - Evaluation Report.*

D17.2: Marco Richeldi and Alessandro Perrucci, *Churn Analysis Case Study.*

D17.2b: Cezary Chudzian and Janusz Granat and Wieslaw Traczyk, *Call Center Case.*

D17.3: Marco Richeldi and Alessandro Perrucci, *Mining Mart Evaluation Report.*

D17.3b: Janusz Granat and Wieslaw Traczyk and Cezary Chudzian, *Evaluation report by NIT.*

D18: Martin Scholz and Timm Euler and Lorenza Saitta, *Applicability Constraints on Learning Operators.*

D19: Ronnie Bathoorn, Nico Brandt, Marc de Haas and Olaf Rem, *Problem Modeling.*

D20.4: Katharina Morik, Martin Scholz and Timm Euler, *MiningMart Final Report.* (This document.)

D20.5: Katharina Morik, Martin Scholz and Timm Euler, *Ext-MM Final Report.*

6.2.3 Technical Reports

TR 12-02: Timm Euler, *Operator Specifications*

TR 12-04: Timm Euler, *How to implement M4 operators*

TR 12-05: Martin Scholz and Timm Euler, *Documentation of the MiningMart Meta Model (M4)*

TR 18-01: Martin Scholz, *Representing Constraints, Conditions and Assertions in M4*

TR 18-02: Martin Scholz, *Using Constraints, Conditions and Assertions*

6.2.4 Other Documents

The *MiningMart User Guide*, which includes installation instructions, is available from the downloads section of the MiningMart web pages.

Bibliography

[BHP02] 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.

[Bra98] Pavel Brazdil. Data Transformation and Model Selection by Experimentation and Meta-Learning. In C.Giraud-Carrier and M. Hilario, editors, *Workshop Notes – Upgrading Learning to the Meta-Level: Model Selection and Data Transformation*, number CSR-98-02 in Technical Report, pages 11–17. Technical University Chemnitz, April 1998.

[CCMR90] Karine Causse, Marc Csernel, Katharina Morik, and Céline Rouveirol. MLT Deliverable 2.2: Specification of the Common Knowledge Representation Language of the MLToolbox. GMD (German Natl. Research Center for Computer Science), P.O.Box 1240, W-5205 St. Augustin 1, Germany, September 1990.

[ELS97] Robert Engels, Guido Lindner, and Rudi Studer. A Guided Tour through the Data Mining Jungle. In *Proceedings of the 3nd International Conference on Knowledge Discovery in Databases (KDD-97)*, August 14–17 1997.

[Eng96] Robert Engels. Planning Tasks for Knowledge Discovery in Databases; Performing Task–Oriented User–Guidance. In *Proc. of th 2nd Int. Conf. on Knowledge Discovery in Databases*, August 1996.

[FKMR02] Simon Fischer, Ralf Klinkenberg, Ingo Mierswa, and Oliver Ritthoff. YALE: Yet Another Learning Environment – *tutorial*. Technical Report CI-136/02, Collaborative Research Center 531, University of Dortmund, Dortmund, Germany, 2002. ISSN 1433-3325.

[KZFB00] Joerg-Uwe Kietz, Regina Zuecker, Anna Fiammengo, and Giuseppe Beccari. Data Sets, Meta-data and Preprocessing Operators at Swiss Life and CSELT. Deliverable D6.2, IST Project MiningMart, IST-11993, 2000.

[KZV00] Jörg-Uwe Kietz, Regina Zücker, and Anca Vaduva. Mining Mart: Combining Case-Based-Reasoning and Multi-Strategy Learning into a Framework to reuse KDD-Application. In R.S. Michalski and P. Brazdil, editors, *Proceedings of the fifth International Workshop on Multistrategy Learning (MSL2000)*, Guimares, Portugal, May 2000.

[LM98] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining.* Kluwer Academic Publishers, 1998.

[MCB91] K. Morik, K. Causse, and R. Boswell. A Common Knowledge Representation Integrating Learning Tools. In *Proc. of the 1st International Workshop on Multistrategy Learning*, Harpers Ferry, 1991.

[MST94] D. Michie, D. J. Spiegelhalter, and C. C. Taylor. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, New York u.a., 1994.

[Rue99] Stefan Rueping. Zeitreihenprognose für Warenwirtschaftssysteme unter Berücksichtigung asymmetrischer Kostenfunktionen. Master's thesis, Universität Dortmund, 1999.

[SOD89] D. Sleeman, R. Oehlman, and R. Davidge. Specification of Consultant–0 and a Comparision of Several Learning Algorithms. Deliverable D5.1, Esprit Project P2154, 1989.

[TL98] C. Theusinger and G. Lindner. Benutzerunterstützung eines KDD-Prozesses anhand von Datencharackteristiken. In F. Wysotzki, P. Geibel, and K. Schädler, editors, *Beiträge zum Treffen der GI-Fachgruppe 1.1.3 Machinelles Lernen (FGML-98)*, volume 98/11 of *Technical Report.* Technical University Berlin, 1998.

[WF00] Ian Witten and Eibe Frank. *Data Mining – Practical Machine Learning Tools and Techniques with JAVA Implementations.* Morgan Kaufmann, 2000.

[ZLO97] 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.

[ZLO01] N. Zhong, C. Liu, and S. Ohsuga. Dynamically Organizing KDD Processes. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(3):451–473, 2001.