Applied Intelligence 19, 125–141, 2003


# A Knowledge Discovery System with Support for Model Selection and Visualization

TU BAO HO, TRONG DUNG NGUYEN, HIROSHI SHIMODAIRA AND MASAYUKI KIMURA
*Japan Advanced Institute of Science and Technology, Ishikawa, 923-1292 Japan*
bao@jaist.ac.jp
nguyen@jaist.ac.jp
sim@jaist.ac.jp
kimura@jaist.ac.jp



**Abstract.** The process of knowledge discovery in databases consists of several steps that are iterative and interactive. In each application, to go through this process the user has to exploit different algorithms and their settings that usually yield multiple models. Model selection, that is, the selection of appropriate models or algorithms to achieve such models, requires meta-knowledge of algorithm/model and model performance metrics. Therefore, model selection is usually a difficult task for the user. We believe that simplifying the process of model selection for the user is crucial to the success of real-life knowledge discovery activities. As opposed to most related work that aims to automate model selection, in our view model selection is a semiautomatic process, requiring an effective collaboration between the user and the discovery system. For such a collaboration, our solution is to give the user the ability to try various alternatives and to compare competing models quantitatively by performance metrics, and qualitatively by effective visualization. This paper presents our research on model selection and visualization in the development of a knowledge discovery system called D2MS. The paper addresses the motivation of model selection in knowledge discovery and related work, gives an overview of D2MS, and describes its solution to model selection and visualization. It then presents the usefulness of D2MS model selection in two case studies of discovering medical knowledge in hospital data—on meningitis and stomach cancer—using three data mining methods of decision trees, conceptual clustering, and rule induction.

**Keywords:** KDD, model selection, visualization, user's participation

## 1. Introduction

*Knowledge discovery in databases* (KDD)—the rapidly growing interdisciplinary field of computing that evolves from its roots in database management, statistics, and machine learning—aims at finding useful knowledge from large databases. The process of knowledge discovery is complicated and should be seen inherently as a process containing several steps. The first step is to understand the application domain, to formulate the problem, and to collect data. The second step is to preprocess the data. The third step is that of data mining with the aim of extracting useful knowledge as patterns or models hidden in data. The fourth step is to post-process discovered knowledge. The fifth step is to put discovered knowledge to practical use [1]. These steps are inherently *iterative* and *interactive*, i.e., one cannot expect to extract useful knowledge by just pushing one time a large amount of data into a black box without the user's participation.

In this work we adopt the view in [2] regarding a *pattern* as a local structure and a *model* as a global representation of a structure. While a pattern is related to just a handful of variables and a few cases such as a rule, a model, e.g. a decision tree or a regression hyperplane, summarizes the systematic component underlying the data or describes how the data may have arisen. In a

broader sense, we will refer hereafter to a set of discovered patterns as a model.

The KDD primary goals of prediction and description are concerned with different tasks in the data mining step of the KDD process, such as those for characterization, discrimination, association, classification, and clustering [3]. Also, there are different tasks of data cleaning, integration, transformation, and reduction in the preprocessing step, and those of interpretation, evaluation, exportation, and visualization of results in the post-processing step. Moreover, each of these tasks can be done by different methods and algorithms. To solve a given discovery problem, the user usually has to go through these steps several times, with each time corresponding to an exploitation of a series of algorithms.

It is well-known that there is no inherently superior method/model in terms of generalization performance. The No Free Lunch theorem [4] states that in the absence of prior information about the problem, there are no reasons to prefer one learning algorithm or classifier model over another. The problem of *model selection*—choosing appropriate discovered models or algorithms and their settings for obtaining such models in a given application—is difficult and non-trivial because it requires empirical comparative evaluation of discovered models and meta-knowledge on models/algorithms. The user often has to do a trial-and-error process to select the most suitable models from competing ones. Clearly, trying all possible options is impractical, and an informed search process is needed to attain expected models. Informed search requires both performance metrics and model characteristics that often are not available to the user. Moreover, the user's interest in discovered models is a subjective matter that depends much on his/her domain knowledge and sometimes is very independent of performance metrics provided by the system. Current data mining provides multiple algorithms within a single system, but the selection and combination of these algorithms are external to the system and specified by the user. This makes the KDD process difficult and possibly less efficient in practice.

Unlike the major research tendency that aims to provide the user with meta-knowledge for an automatic model selection as described in the next section, in our view model selection is semiautomatic and requires an effective collaboration between the user and the discovery system. In such a collaboration, visualization has an indispensable role because it can give a deep understanding of complicated models that the user cannot achieve if using only performance metrics. The research on visualization integrated with model selection is significant because there is currently very limited visualization support for the process of building and selecting models in knowledge discovery [5, 6].

The goal of this work is to develop a research system for knowledge discovery with support for model selection and visualization. The system called D2MS (Data Mining with Model Selection) first provides the user with the ability to try various algorithm combinations and their settings. Each combination of algorithms is registered in a list called a plan. After executing registered plans, the user is supported in evaluating competing models quantitatively and qualitatively before making his/her final selection. The quantitative evaluation can be obtained by performance metrics provided by the system, while the qualitative evaluation can be obtained by effective visualization of the discovered models.

Within D2MS several data mining methods have been implemented, including decision tree induction method CABRO [7], CABROrule [8], conceptual clustering method OSHAM and its variants [9], [10], and rule induction method LUPC [11]. These programs have been integrated with several advanced visualization techniques in D2MS such as tightly-coupled views [12], fish-eye views [13], and particularly the proposed technique T2.5D (Tree 2.5 Dimensions) for large hierarchical structures [14].

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 gives an overview of the system. Section 4 introduces the D2MS solution to model selection integrated with visualization. Section 5 briefly summarizes three data mining methods—decision tree induction, conceptual clustering, and rule induction—that were implemented in D2MS, and presents two case studies of discovering knowledge in hospital data—on meningitis and stomach cancer—with these methods. Section 6 summarizes the work and outlines further research.

## 2. Related Work

Model selection is a general statistical notion, and a probability model is generally a mathematical expression representing some underlying phenomenon [15]. The model selection has usually been viewed in the statistics community in terms of estimation of

expected discrepancy $E\Delta(f, g_{\hat{\theta}})$ between the underlying model $f$ (that is unknown in practice) and approximating models $g_{\hat{\theta}}$, where $\theta$ is the parameter vector [16]. The standard methods of model selection in statistics include classical hypothesis testing, maximum likelihood, Bayes method, minimum description length, cross-validation, bootstrap, and Akaike's information criteria—see for example [16, 17].

The model selection problem has attracted researchers in machine learning and recently in knowledge discovery because of its significant importance. Most research on model selection in this community has focused on two issues: the preselection of models/algorithms that is typically concerned with finding measures of relation between the dataset characteristics and the models/algorithms performance (referred to as *model characteristics*); and the post-selection that is concerned with the evaluation of models/algorithms using multi-criteria of performance (referred to as *performance metrics*).

In [18] the author introduced an approach that uses knowledge about the representational biases of a set of learning algorithms coded in the form of rules in order to pre-select models or algorithms. In [19] the authors introduced three ranking methods of classification algorithms: using average ranks, success rate ratios, and significant wins. These methods generate rankings by results obtained by algorithms on the training datasets, and the rankings are evaluated by their distance from the ideal ranking built in advance. In [20] the authors proposed to use learning algorithm profiles to address the model selection problem. The algorithm profiles consist of meta-level feature-value vectors which describe learning algorithms in terms of their representation, functionality, efficiency, robustness, and practicality. NOEMON [21] is a prototype system in the context of the classification task that suggests the most appropriate classifier, deciding on the basis of morphological similarity between the new dataset and the existing collection.

Most research on performance metrics focuses on multi-criteria for doing post-selection of algorithms. In [22] the authors proposed a multi-criteria evaluation metric that aims to take into account both positive and negative properties of data mining algorithms, and provides a fast and comprehensive evaluation of various algorithms. In [23] the author strengthened the practical role of stratified cross validation in model selection for many common application domains by a large number of careful experiments.

The above-mentioned research efforts aimed to find meta-knowledge towards an automatic selection of models/algorithms and did not take into account the participation of the user during model selection. Also, they often limited themselves to the classification problem with supervised data in which performance metrics can be computed and served for the prediction goal. The situation is somewhat different in the case of the clustering problem where the main goal is description and several usual performance metrics such as accuracy cannot be computed from unsupervised data. While D2MS provides model characteristics and performance metrics for a quantitative evaluation, the major distinguishing feature of the D2MS system is its tight integration of visualization with model selection that supports a qualitative evaluation so that the user can make a better final selection based on his/her insight into discovered models.

Many KDD systems provide visualization of data and knowledge, and different visualization techniques have been developed or adapted to the KDD process. MineSet [24] provides several 3D visualizers, in particular a 3D Tree Visualizer. In [25] the authors develops an interactive visualization in decision tree construction for supporting an effective cooperation of the user and the computer in classification. In [26] the authors tightly integrate visualization with five steps of the KDD process. D2MS shares common features with the above systems in human-machine interaction and process visualization, and contributes two additional features. Firstly, it integrates visualization into model selection by providing visual information of competing models for the qualitative evaluation of the user. Secondly, it provides the effective view T2.5D for large hierarchical structures that can be seen as an alternative to powerful techniques for representing large hierarchical structures such as 3D cone trees [27] or 2D hyperbolic trees [28]. For large trees, T2.5D can give an effective view by displaying subtrees of the user's interest in 2D space and the rest of the tree in a virtual 3D space [14].

D2MS is a KDD research system that shares several common points with some KDD research systems. MLC++ [29] is a powerful library of classification algorithms with guidance to compare and select appropriate algorithms for the classification task. It also provides a visualization and interfaces between programs for both end-users and software developers. Sharing the class library approach, while MLC++ is only suitable for well-trained programmers, D2MS can orient a user without much computer knowledge.

DBMiner [3] is a KDD system that provides a wide spectrum of data mining functions. As interactive data mining environments, while DBMiner focuses on a tight integration of data mining methods with on-line analytical processing, D2MS focuses on a tight integration of data mining methods with model selection and visualization. FlexiMine [30] is a flexible platform for KDD research and application development. It is designed as a testbed for data mining research and a generic knowledge discovery tool. While sharing with Fleximine the integration of most KDD functions and the flexibility to do data mining, D2MS focuses on supporting the active participation of the user.

## 3. Overview of the System

Figure 1 presents the conceptual architecture of D2MS. The system consists of eight modules: Graphical user interface, Data interface, Data processing, Data mining, Evaluation, Plan management, Visualization, and Application. Additionally, it has a Plan base and a Model base to store registered plans and discovered models, respectively. Importantly, these modules follow the *cooperative mechanisms*: the modules are constructed in such a way that each of them can cooperate with all others to achieve the final goal.

D2MS is able to deal with various aspects in knowledge discovery: large datasets and high dimensionality, the degree of supervision, different types of data in different forms, the changing nature of data and knowledge, and interaction and visualization [10]. This ability makes the choice of algorithms a strong effect on discovered results.

*Graphical user interface*. A graphical user interface is designed to facilitate the complicated interaction between the user and the system in the knowledge discovery process. It allows the user to register different combinations of algorithms in the KDD process and their parameter settings via plans. Also, it provides the user with facilities to participate in the KDD process. More importantly, the graphical user interface supports the user in evaluating competing models in order to select the appropriate ones.

*Data interface*. The native input format of the system is a flat form of data files (filestem.data and filestem.names) like that of C4.5 data files format [31]. The system supports other common database formats by using standard interfaces (e.g., ODBC).

*Data preprocessing*. This module is to provide different programs for the four main tasks in the preprocessing step: data cleaning, integration, transformation, and reduction. The current version of D2MS has algorithms to solve the most common problems in preprocessing: discretization of continuous attributes, filling up missing values, and instance and feature selection. Discretization methods include rough set-based and entropy-based programs for supervised data, as well as a $k$-means-based discretization program for unsupervised data [32]. Missing-value techniques include three programs for filling up missing values: natural cluster based mean-and-mode (NCBMM), attribute rank cluster based mean-and-mode (ARBMM), and $k$-means clustering based mean-and-mode (kMCMM) [33]. Sampling programs and method SFG for feature selection [34] are also available in D2MS.

*Data mining*. The system provides several algorithms for prediction and description. In the current version of D2MS, classification algorithms consist of $k$-nearest neighbor, Bayesian classification, decision tree induction method CABRO [7], CABROrule [8], and LUPC [11], whereas CABRO and CABROrule have their variants with R-measure, gain ratio [31],



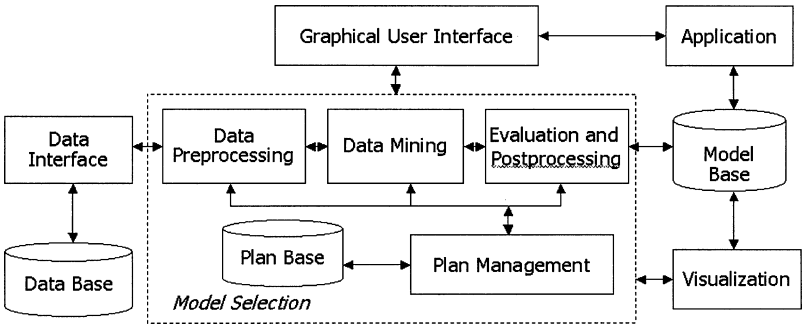



*Figure 1*. Conceptual architecture of the system.

gini-index [35], and $\chi^2$ [36], and each of them can produce trees or rule sets by either error-based pruning [31] or the post-pruning technique in [8]. Clustering algorithms include: (1) $k$-means and its variants for mixed numerical and categorical attributes using an efficient mixed similarity measure MSM [37], and (2) conceptual clustering algorithm OSHAM [9, 10]. These programs can read data in the common input form, and they can represent discovered models in a form readable by other related modules (post-processing, visualization, and plan management).

*Post-processing and evaluation*. This module provides tools for post-processing and evaluation of discovered models. Its available tools include those for $k$-fold stratified cross validation integrated with data mining programs [7], for post-pruning of decision tree [8], for automatically generating tables containing synthesized information of discovered models, and for exporting competing models into other easy-to-understand forms for the user (e.g., the spreadsheet format [10]), as well as forms readable for visualization programs (e.g., T2.5D for visualizing the hierarchical structures [14]).

*Plan management*. The main function of this module is to manage a plan base containing plans declared by the user. It also coordinates modules on data mining, evaluation, and visualization in model selection.

*Data and knowledge visualization*. This module consists of three visualizers for data, rules, and hierarchical structures. The algorithms in other modules can frequently invoke this module in order to help the user make effective use of them.

*Application*. This module contains utilities that help the user to use discovered models. A model can be used to match an unknown instance, or to generate an interactive dialogue in which the system conducts a series of questions/answers in order to predict the outcome.

## 4. Model Selection and Visualization

### *4.1. Model Selection*

The *interestingness* of discovered patterns/models is commonly characterized by several criteria: *evidence* indicates the significance of a finding measured by a statistical criterion; *redundancy* amounts to the similarity of a finding with respect to other-findings and measures to what degree a finding follows from another one; *usefulness* relates a finding to the goal of the users; *novelty* includes the deviation from prior knowledge of the user or system; *simplicity* refers to the syntactical complexity of the presentation of a finding, and *generality* is determined by the fraction of the population a finding refers to. The interestingness can be seen as a function of the above criteria [38] and strongly depends on the user as well as his/her domain knowledge.

Working with medical experts for more than two years, we eventually came to realize that their evaluation are not always concerned with performance metrics provided by discovery systems. In our case studies on meningitis and stomach cancer domains presented in Section 5, domain experts were often interested in models that do not necessarily have the highest predictive accuracy but those that give them something new. It turns out that model selection in a KDD system would be a user-centered process that depends strongly on active participation by the user.

The key idea of our solution to model selection in D2MS is to support an effective participation of the user in this process. Concretely, D2MS first supports the user in doing trials on combinations of algorithms and their parameter settings in order to produce competing models, and then supports the user to evaluate them quantitatively and qualitatively by providing both performance criteria as well as visualization of these models.

The model selection in D2MS mainly involves the three steps of data processing, data mining, and post-processing, as shown in Fig. 1. There are three phases in doing model selection in D2MS, all of which are managed by the plan management module: (1) registering plans of selected algorithms and their settings; (2) executing the plans to discover models; (3) selecting appropriate models by a comparative evaluation of competing models.

The first phase is to register plans. A *plan* is an ordered list of algorithms associated with their parameter settings that can yield a model or an intermediate result when being executed. The plans are represented in a tree form called *plan tree* whose nodes are selected algorithms associated with their settings (the top-left window in Fig. 2). The nodes on a path of the plan tree must follow the order of preprocessing, data mining, and post-processing. A plan may contain several algorithms (nodes) of each step, for example filling in missing values by the NCBMM and discretizing continuous attributes by the entropy-based algorithm in

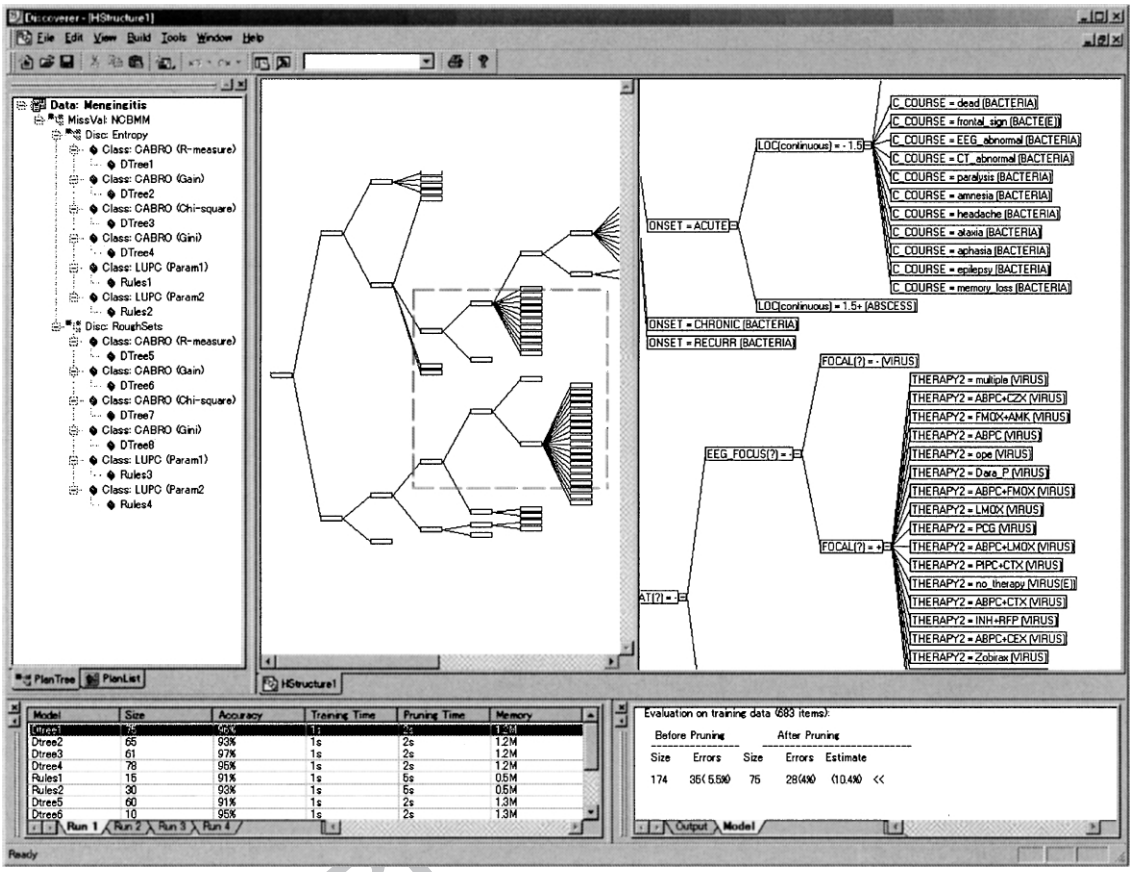



*Figure 2.* A screen of D2MS in selecting models learned from meningitis data: the top-left window shows the plan tree; the bottom-left window shows the summary table of performance metrics of discovered models; the bottom-right window shows detail information of the model being activated (highlighted); and the top-right window shows tightly-coupled views of the model in which the detailed view in the right part corresponds to the field-of-view specified in the left part.



preprocessing, then mining by CABRO and LUPC, and evaluating discovered results by $k$-fold cross validation in post-processing.

The plan management module provides the user with a friendly user-interface to register, modify, and delete plans on the plan tree. D2MS allows the user to register plans fully or gradually. A plan can be fully registered by determining straightaway all of its component algorithms and their settings so that it can yield a model. However, a plan can be registered gradually by determining algorithms and their settings step by step. This mode of registration is useful in mining large datasets. For example, to do a task in the KDD process, the user can try several algorithms/settings, and then evaluate their results and select only promising ones for doing further tasks.

The plan management module maintains *profiles* of algorithms available in preprocessing, data mining, and post-processing of D2MS. An algorithm profile contains information about the algorithm functions, its requirements, types and effect of its parameters. An algorithm is registered via a dialog box when it is added into a plan. For example, to generate a decision tree by CABRO, the user needs to choose the minimum number of instances at leaf nodes, the lowest accepted error rate, the attributes to be grouped, the method for filling in missing values, etc. The top-left window in Fig. 2 shows a plan tree created for mining the meningitis data (Section 5). This plan tree consists of NCBMM for imputing missing values, a discretization based on entropy and rough sets, and CABRO with different selection measures of LUPC with several parameter settings.

The second phase is to execute registered plans. While gradually registering a plan, the user can run an algorithm just after adding it to the plan, then

evaluate its results before deciding whether to continue the plan from its current stage with other algorithms, or to backtrack and try the plan with another algorithm or setting. The user also can run a plan after fully registering it, or even register a number of plans and then run them altogether. The intermediate results, the discovered models and their summaries and exported forms will be automatically created and stored in the *model base*.

The third phase is to select appropriate models. D2MS provides a summary table presenting performance metrics of discovered models according to executed plans (the bottom-left window in Fig. 2). However, the user can evaluate each model in depth by visualizing it, browsing its structure, checking its relationship with the dataset, etc. (the top-right and bottom-right windows in Fig. 2). The user also can visualize several models simultaneously to compare them (see Fig. 4). By getting a real insight into the competing models, the user certainly can make a better selection of models.

The performance metrics on discovered models are reported according to the types of algorithms. D2MS reports a multidimensional measure for classification supervised algorithms consisting of the following sub-measures: (1) accuracy, (2) size (number of leaf nodes or rules), (3) training time, (4) pruning time, (5) resource demand (memory). D2MS also reports a multidimensional measure for unsupervised clustering algorithms with the following sub-measures: (1) size (number of clusters); (2) number of hierarchy levels (3) training time, (4) resource demand (memory).

Section 4.2 addresses the visualization in model selection, and Section 5 will illustrate the D2MS's model selection with more details in regards to medicine applications.

### 4.2. *Visualization Support for Model Selection*

In D2MS, visualization helps the user to interpret data and models as well their connections in order to understand and evaluate models better. Through visualization the user also can see better the effect of algorithm settings on resulted models so that he/she can adjust settings to reach adequate models.

After a description of available visualizers in D2MS, this subsection focuses on two highlighted characteristics of its visualization: visualization of hierarchical structures and a tight integration of visualization with model selection.

***4.2.1. Visualizers.*** There are three visualizers in D2MS: data visualizer, rule visualizer, and hierarchy visualizer.

*Data visualizer*. It provides the user graphical views on the statistics of the input data and relations between attributes. These include multiple forms of viewing data such as tables, cross-tabulations, pie or charts, cubes. The data visualizer supports the user in the preprocessing step and in the selection of algorithms when registering plans.

*Rule visualizer*. It allows the user to view rules generated by CABROrule or LUPC. The visualizer provides graphical views on statistics of conditions and conclusions of a rule, correctly and wrongly matched cases in both training and testing data, and links between rules and data.

*Hierarchy visualizer*. It visualizes hierarchical structures generated by CABRO and OSHAM. The visualizer provides different views that may be suitable for different types and sizes of hierarchies. The user can view an overall structure of a hierarchy together with the detailed information of each node. D2MS provides several visualization techniques that allow the user to visualize large hierarchical structures effectively. The tightly-coupled views [12] simultaneously display a hierarchy in normal size and tiny size that allows the user to determine quickly the field-of-view and to pan to the region of interest (Fig. 2). The fish-eye view [13] distorts the magnified image so that the center of interest is displayed at high magnification, while the rest of the image is progressively compressed. Also, the new technique T2.5D [14] is implemented in D2MS for visualizing very large hierarchical structures.

***4.2.2. Trees 2.5 Dimensions.*** The user might find it difficult to navigate a very large hierarchy, even with tightlycoupled and fish-eye views. To overcome this difficulty, we have been developing a new technique called T2.5D (stands for Trees 2.5 Dimensions).

T2.5D is inspired by the work of Reingold and Tilford [39] that draws clear trees in a reasonable time with reasonable storage. Different from tightly-coupled and fish-eye views that can be seen as location-based views (view of objects in a region), T2.5D can be seen as a relation-based view (view of related objects). The starting point of T2.5D is the observation that a large tree consists of many subtrees that are not usually and necessarily viewed simultaneously. The key idea of T2.5D is to represent a large tree in a virtual

3D space (subtrees are overlapped to reduce occupied space), while each subtree of interest is displayed in a 2D space. To this end, T2.5D determines the fixed position of each subtree (its root node) in two axes (X and Y) and, in addition, it computes dynamically a Z-order for this subtree in an imaginary axis Z. A subtree with a given Z-order is displayed in front of its siblings that have higher Z-orders.

When visualizing and navigating a tree, at each moment the Z-order of all nodes on the path from the root to a *node in focus* in the tree is set to zero by T2.5D. The *active wide path* to a node in focus, which contains all nodes on the path from the root to this node in focus and their siblings, is displayed in the front of the screen with highlighted colors to give the user a clear view. Other parts of the tree remain in the background to provide an image of the overall structure. With Z-order, T2.5D can give the user an impression that trees are drawn in a 3D space. The user can easily change the active wide path by choosing another node in focus [14].

We have tested T2.5D with various real and artificial datasets. One example is the case of the U.S. census bureau database consisting of 199,523 instances described by 40 numeric and symbolic attributes (103 Mbytes). T2.5D handles well a tree learned from this dataset that has nearly 20,000 decision and leaf nodes. The ease of navigation of such a large tree depends mainly on the depth of the tree (number of tree levels) and on the number of nodes having many subnodes (branches) but not on the number of nodes, since most of nodes are virtually represented. The understanding of such trees is supported by the use of different colors to label classes of leaf nodes. Figure 3 illustrates a pruned tree of 1795 nodes learned from stomach cancer data (Section 5). In general, more than 1,000 nodes in several levels can be displayed together on the screen [14].



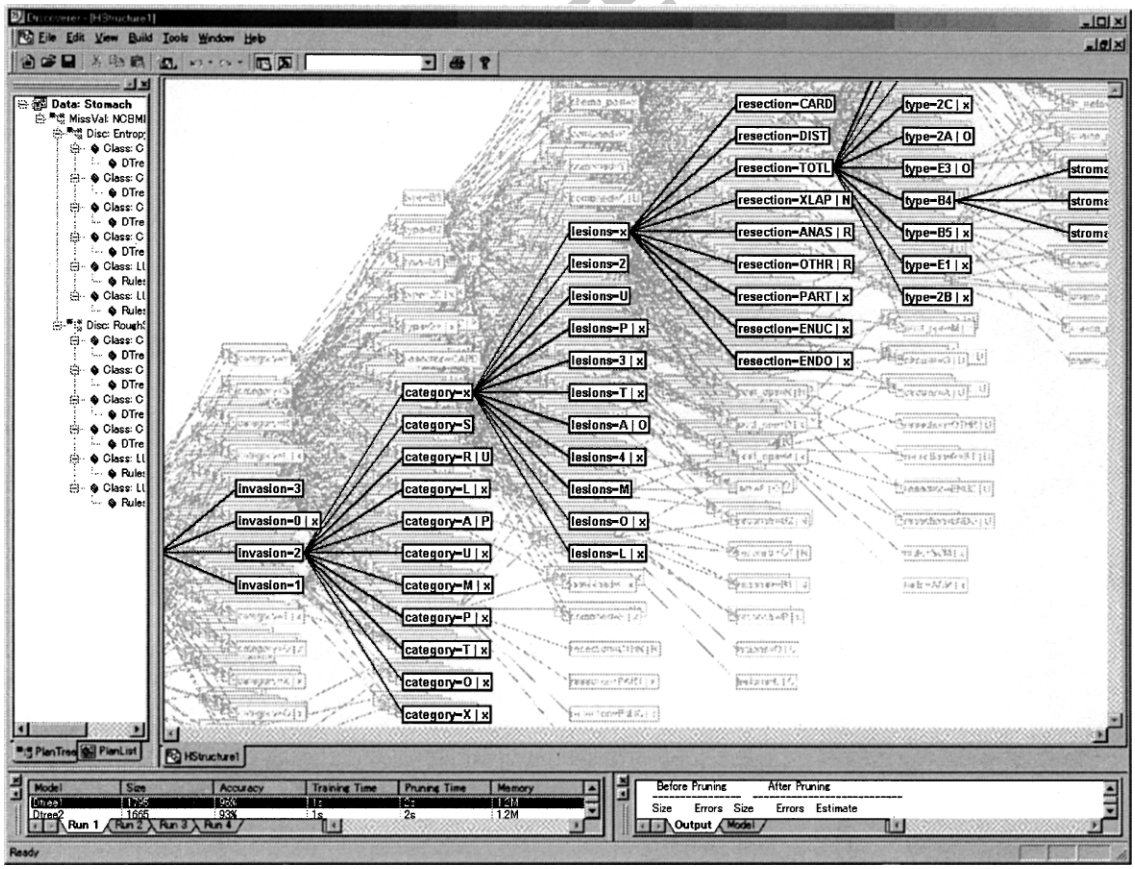



*Figure 3.* T2.5D provides views in between 2D and 3D. The pruned tree learned from the stomach cancer data by CABRO has 1795 nodes where the active wide path to a node in focus displays all of its direct relatives in a 2D view.

***4.2.3. Visualization in Model Selection.*** In D2MS, visualization is integrated with the steps of the KDD process and closely associated with the plan management module in support of model selection. The user can have either of these two views: executing a plan, or comparative views of discovered models.

If the user is interested in following the execution of one plan, he/she can view, for example, the input data, the derived data after preprocessing, the generated models with chosen settings, and the exported results. Thus, the user can follow and verify the process of discovery by each plan, and change settings to reach alternative results. If the user is interested in comparative evaluation of competing models generated by different plans, he/she can have multiple views of these models. The user can compare performance metrics of all activated plans that are always available in the summary table. Whenever the user highlights a row in the table, the associated model will be automatically displayed. Several windows can be opened simultaneously to display competing models in forms of trees, concept hierarchies, or rule sets. Figure 4 illustrates a view of two different concept hierarchies, with different sizes and structures, generated by OSHAM from the meningitis data.

## 5. Case Studies

This section illustrates the utility of model selection and visualization of D2MS by presenting two knowledge discovery case studies. It starts with a brief introduction to three data mining methods (CABRO, LUPC and OSHAM) implemented in D2MS, and two medical datasets (on meningitis and stomach cancer).



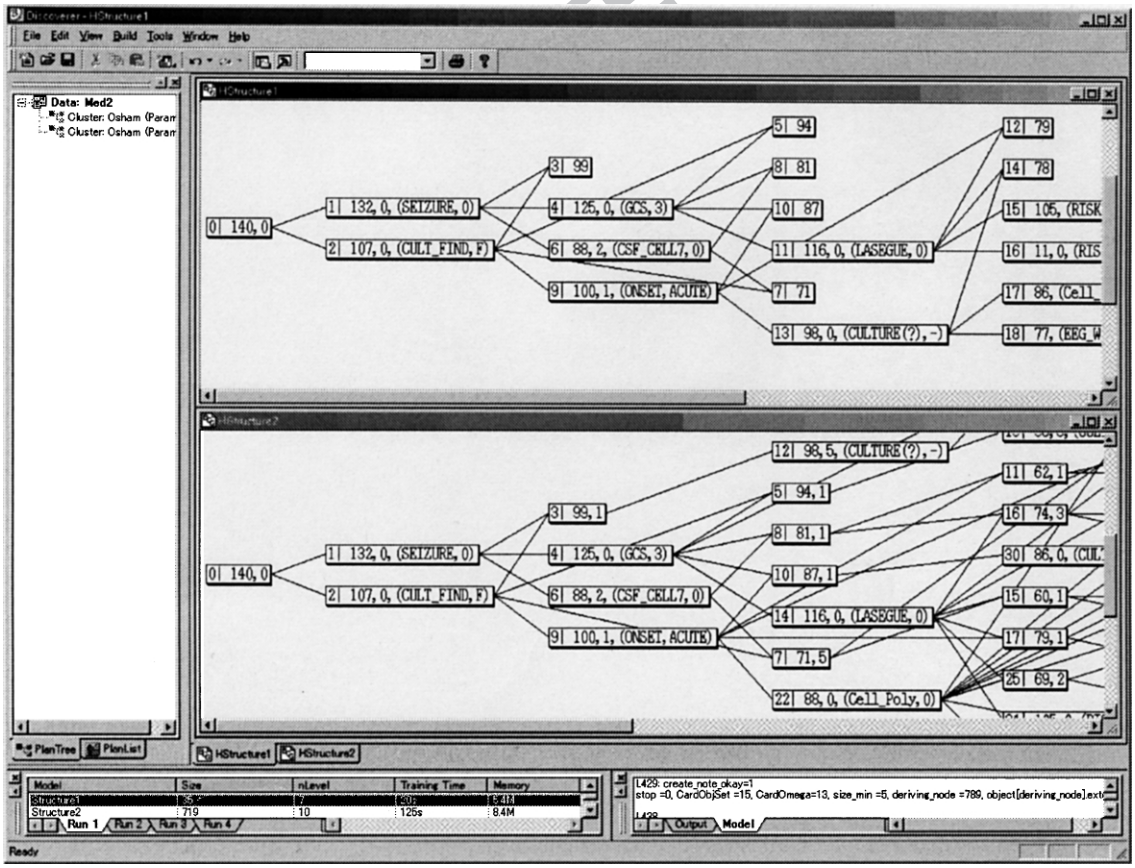



*Figure 4.* Viewing comparatively competing models obtained by different OSHAM's parameter settings in mining useful "concepts" from the meningitis data. Two concept hierarchies show different possible descriptions in terms of size and structure.

### *5.1. CABRO, LUPC, and OSHAM*

***5.1.1. Decision Tree Method CABRO.*** *Decision tree induction* (DTI) is one of the most widely used knowledge discovery methods for supervised data. From a given set of labeled instances, DTI induces a model (classifier) in the form of a decision tree that predicts classes of unknown instances. Typical decision tree methods include C4.5 [31] and CART [35].

The two main problems in DTI are attribute selection and pruning. Attribute selection is to decide which attribute will be chosen to branch a decision node [36]. Pruning is to avoid over-fitting by reducing the complexity of the tree [40]. In our current system, decision tree program CABRO (Construction d'une Base de Règles à partir d'Observations) [7] or its variant CABROrule [8] allow the user to generate different tree models or rule sets by combining one of four attribute selection measures (gain-ratio [31], the gini-index [35], $\chi^2$ [36] and R-measure [7]) with one of the following pruning algorithms: error-complexity, reduced-error, and pessimistic error [31, 35].

The following techniques resulting from our research on DTI have been implemented in the system: efficient visualization of hierarchical structures including tightly-coupled, fish-eye, and T2.5D views [14]; and rule post-pruning of large trees in the post-processing step [8]. These techniques offer advantages in data mining in which datasets are often large, and consequently so are the decision trees. From the U.S. census database, C4.5 produces a pruned tree of about 18,500 decision and leaf nodes, which are almost impossible to understand in text form. Also, C4.5 could not convert this tree into a rule set after two days of running on a Sun workstation of 128 Mbytes of RAM memory. CABRO, however, provides an excellent view of its discovered tree by T2.5D, and converted the tree into rules in a near linear time of $O(n \log n)$.

***5.1.2. Rule Induction Method LUPC.*** LUPC (Learning Unbalanced Positive Class) is another approach to learning descriptive rules from supervised data. LUPC permits two learning modes: learning one target class or learning all classes. If the user is interested in learning one target class, this class will be considered as the positive class $C^+$ and all other classes as the negative class $C^-$. With a suitable voting procedure and by sequentially taking each class as the target class, LUPC can learn all classes. It has been shown to be very effective when learning a minority class as the target class in a unbalanced dataset, and it is comparable to other well-known methods when learning all classes [11].

LUPC yields different rule sets depending on the settings of its four parameters: (1) minimum accuracy of a rule $\alpha$; (2) minimum coverage of a rule $\beta$; (3) number of candidate attribute-value pairs $\eta$; and (4) number of candidate rules $\gamma$. There are three main features that distinguish LUPC from related methods and allow it to learn effectively in unbalanced datasets. Firstly, it efficiently combines separate-and-conquer rule induction [41] with association rule mining by finding $\alpha\beta$-strong rules biased on accuracy and cover ratio with adaptive thresholds. Secondly, it exploits a property of rules in unbalanced datasets[1] (we skip our proof as it can be verified easily) to make an effective search with a large beam search parameter, as well as using data in the negative class to speed up the mining process on the minority positive class. Thirdly, it integrates prepruning and post-pruning in a way that can avoid overfitting. Note that the risk of overfitting to the particular dataset(s) given the hand-turning process of parameter settings is avoided in D2MS while using CABRO, CABROrule, and LUPC because of the pruning mode of these programs.

***5.1.3. Conceptual Clustering Method OSHAM.*** *Conceptual clustering* is a typical knowledge discovery method for unsupervised data. Its basic task is from a given set of unlabelled instances to find simultaneously a hierarchical model that determines useful object subsets and their intensional definitions. These two problems relate to another important problem, that of interpreting discovered concepts in the hierarchical structure.

The method OSHAM (Making Automatically Hierarchies of Structured Objects) employs a hybrid representation of concepts that combines advantages of classical, prototype, and exemplar views on concepts [9]. Depending on parameter settings, OSHAM can extract non-disjoint or disjoint hierarchical models of concepts. The algorithm OSHAM is non-incremental and divisive. For each discovered concept, OSHAM seeks recursively for more specific subconcepts according to a quality function defined in its hybrid representation. The form and the size of OSHAM's hierarchical models depend on plans and settings of the following parameters: (1) the concept hierarchy to discover is disjoint or non-disjoint; (2) the minimum size of each

node; (3) the threshold about the concept dispersion; and (4) the number of competitors for beam search. OSHAM is associated with an interpretation procedure to use discovered models. There are several variants of OSHAM: incremental I-OSHAM that can learn when databases are incrementally updated; and approximate A-OSHAM that can learn approximate concepts when data are uncertain and imprecise [10].

OSHAM yields different models with concept's description in the hierarchy when parameter settings are varied. The selection of appropriate models resulting from unsupervised methods is difficult because only a few quantitative indexes are available. However, OSHAM is implemented in the graphical environment of D2MS and it supports the use in evaluating competing models. In fact, the user can visualize the concept hierarchies discovered by OSHAM, the general-to-specific relation of the concepts, the description associated with each concept such as its typical attribute-value pairs, the dispersion among its members, the estimated occurrence probability of its members, etc. [9, 10]. Figure 4 shows D2MS support for comparatively evaluating two models discovered by OSHAM, with different size and structure, from meningitis data.

### 5.2. *The Meningitis and Stomach Cancer Datasets*

***5.2.1. Meningitis Data.*** The meningitis dataset was collected at the Medical Research Institute, Tokyo Medical and Dental University from 1979 to 1993. It contains data of patients who suffered from meningitis and who were admitted to the department of emergency and neurology in several hospitals. There are 38 numeric and categorical attributes presenting patients history, physical examination, laboratory examination, diagnosis, therapy, clinical course, final status, and risk factors. The tasks determined by medical experts in extracting knowledge from this dataset is to find factors important for (1) diagnosis of meningitis, (2) detection of bacteria or virus, and (3) prediction of prognosis [42]. The dataset and these tasks have been given to challenge the research community [11].

***5.2.2. Stomach Cancer Data.*** The stomach cancer dataset collected at the National Cancer Center in Tokyo during 1962–1991 is a very precious source for this research. It contains data of 7,520 patients: described originally by 83 numeric and categorical attributes. These include information on patients, symptoms, type of cancer, longitudinal and circular location, serosal invasion, metastasis, pre-operative complication, post-operative complication, etc. One goal is to use attributes containing patient information before the operation to predict the patient status after the operation. The domain experts are particularly interested in finding predictive and descriptive rules for the class of patients who died within 90 days after operation (among five outcome classes "death within 90 days", "death after 90 days", "death after 5 years", "alive", "unknown").

### 5.3. *Knowledge Extraction from Meningitis Data*

The main feature of the problem of knowledge extraction from meningitis data is its multi-tasks. Each of the tasks (1), (2), and (3) can be done by two alternative target attributes. For example, the extraction of diagnosis knowledge can be done by using either attribute DIAG2 that groups diagnosis results into 2 classes (VIRUS and BACTERIA), or attribute DIAG that groups diagnosis results into 6 classes (ABSCESS, BACTERIA, BACTE(E), TB(E), VIRUS(E), VIRUS). Similarly, two tasks of extracting predictive and descriptive knowledge on the detection of bacteria/virus and prognosis can be carried out with pairs of attributes CULTFIND and CULTURE (2 and 13 classes), and COURSE and CCOURSE (2 and 12 classes), respectively. There are a total of 6 target attributes in this dataset, and accordingly, 6 datasets derived from the original one for the mining tasks. To find interesting knowledge in such a complicated problem, the interaction of and iteration with various parameter settings of data mining methods are indispensable in order to generate and select models. In our experiments, it was difficult to run iteratively with different parameter settings and to comparatively evaluate other data mining systems such as C4.5 [31], its successor See5, or CBA [43].

Different plans have been created in D2MS for doing these tasks. Each of them is typically a sequence of methods for discretization of numerical attributes, data mining, and visualization of extracted knowledge or its exportation to the Excel format. Two supervised discretization methods in D2MS yield 2 different derived datasets from each of 6 above-mentioned derived datasets as these methods depend on the class attribute. Thus, we obtained totally 12 different derived datasets among them 4 derived datasets can be

used for one mining task. The entropy-based discretization method often ignores many attributes and yields few discretized values on the remaining attributes. The rough set-based discretization method does not ignore any attributes and often yields more values for attributes to be discretized. When data mining methods CABRO (CABROrule), LUPC, and OSHAM with their different settings are applied to these two derived datasets, different possible models can be generated.

The plans associated with CABRO produce 16 competing decision trees, with the default parameter setting, for each of the mining tasks (1), (2), or (3) by using alternatively 4 attribute selection measures (R-measure, gain-ratio, gini-index, and $\chi^2$) on 4 derived datasets. Figure 2 illustrates such generated decision tree models. The bottom-left window reports the performance metrics on each discovered model. Each of them when highlighted will be visualized in the top-right window, and detailed information of the model can be seen in the bottom-right window. The generated trees can be visualized and navigated by fish-eyes and T2.5D visualizers. The utility of D2MS is supported by the fact that it is relatively easy to induce decision trees from a dataset but it is much more difficult to select the best among them. The convenient way of evaluating and comparing generated decision trees in D2MS is of great help to the user.

The plans associated with LUPC (also CABROrule) produce competing rule sets. LUPC runs in two modes: (1) learning all classes, and (2) learning one target class of the target attribute (one model will be generated for one target class, says, the class VIRUS of the attribute DIAG2). D2MS allows us to easily try LUPC with various parameter settings and to evaluate the obtained rule sets by the performance metrics and visualization of rules and data. By analyzing early results obtained by varying parameter settings of LUPC, we have determined a promising parameter setting for this dataset: minimum accuracy of a rule $\alpha = 95\%$, minimum coverage of a rule $\beta = 2$, number of candidate attribute-value pairs $\eta = 100$, number of candidate rules $\gamma = 30$. With these parameters we obtained totally 73 models from 12 derived datasets when running two learning modes (this number of models is smaller than the number of models possibly generated as we did not apply the learning mode of one target class to several classes of three attributes DIAG, CULTURE, and CCOURSE as they have very few training instances). In general, the number of discovered models depends on the parameter settings. Rules found for identifying factors important for diagnosis, detection of bacteria/virus, and predicting prognosis were well accepted by domain experts [11].

The plans associated with OSHAM produce different concept hierarchies depending on the settings of four parameters in OSHAM. Each concept hierarchy presents a hierarchical model of concepts with multiple links where each concept is characterized by a 10-tuple of the components including its intension, extension, etc. [9]. As the size and structure of generated concept hierarchies may greatly change when OSHAM's parameters are varied, it is particularly difficult to produce and select appropriate models. The tree visualizer of D2MS offers a great support to the user in observing and evaluating models, especially the model structure and concept components. For example, the tree visualizer allows us to answer a number of important questions in doing model selection, for example, which attribute-values pairs are the most significant in discovered clusters? Also, what is the relation between the model size (simplicity) and the quality of discovered concepts (e.g., its dispersion, occurence probability, etc.)?

Figure 4 illustrates two different models visualized simultaneously for the comparison purpose. In this example, the models consist of 75 and 114 concepts (nodes) generated by two plans containing OSHAM with its parameter settings (1, 7, 0.3, 3) and (1, 4, 0.3, 4), respectively. The user can observe the structure and links of these models, as well as the characteristics of each concept. Below is an example of a concept found by OSHAM from the meningitis data with parameters (1, 7, 0.3, 3). The components of this concept, which covers 88 cases of which 2 are exceptional, can be seen when navigating the concept hierarchy.

```
CONCEPT 59
Level = 4, Super_Concepts = {25},
  Sub_Concepts = {60 61 62 63 64}
Concept_dispersion = 0.297297
Local_instance_dispersion = 0.297297
Concept probability = 0.628571
Concept local conditional probability
  = 1.000000
Features = {(WBC,0), (SEIZURE,1),
  (EEG_FOCUS(?),-), (CSF_CELL, 0)}
Local_instances/Covered_instances = 2/88
Local_instance_dispersion = 0.297297
Local_instance_conditional_probability
  = 1.000000
```

### 5.4. *Knowledge Extraction from Stomach Cancer Data*

The extraction of prediction and description rules for the class "death within 90 days" from stomach cancer data is a difficult task because the classes in this dataset have a very imbalanced distribution and they are not well separated. Several well-known data mining systems C4.5, See5, CBA, and Rosetta have been applied to do this task. However, the obtained results were far from expectations: they have low support and confidence, and usually relate to only a small percentage of patients of the target class. In next subsections, we show that the method LUPC, with visualization tools of D2MS, allows us to detect and select some interesting rules.

***5.4.1. Preliminary Analysis of Data with Visualization Tools.*** The D2MS's visualization tools associated in LUPC allow us to examine the data and to gain better insight into complex data before learning. While the viewing mode of original data offers an intuition about the distribution of individual attributes and instances, the summarizing and querying modes can suggest irregular or rare events to be investigated, or to guide which biases could be used to narrow the huge search space.

It is commonly known that patients who have symptoms "liver_metastasis" of all levels 1, 2, or 3 will certainly not survive. Also, "serosal_invasion = 3" is a typical symptom of the class "death within 90 days." With the visualization tools, we found several unusual events. For example, among 2329 patients in the class "alive", 5 of them have heavy metastasis of level 3, and 1 and 8 of them have metastasis level 2 and 1, respectively. Moreover, the querying data allow us to verify some significant combination of symptoms such as "liver_metastasis = 3", and "serosal_invasion = 3" as shown in Fig. 5.

***5.4.2. Finding General Rules in Class "Death within 90 Days".*** The term "general rule" indicates extracted rules based on common measures of rule quality such as accuracy and coverage. Different



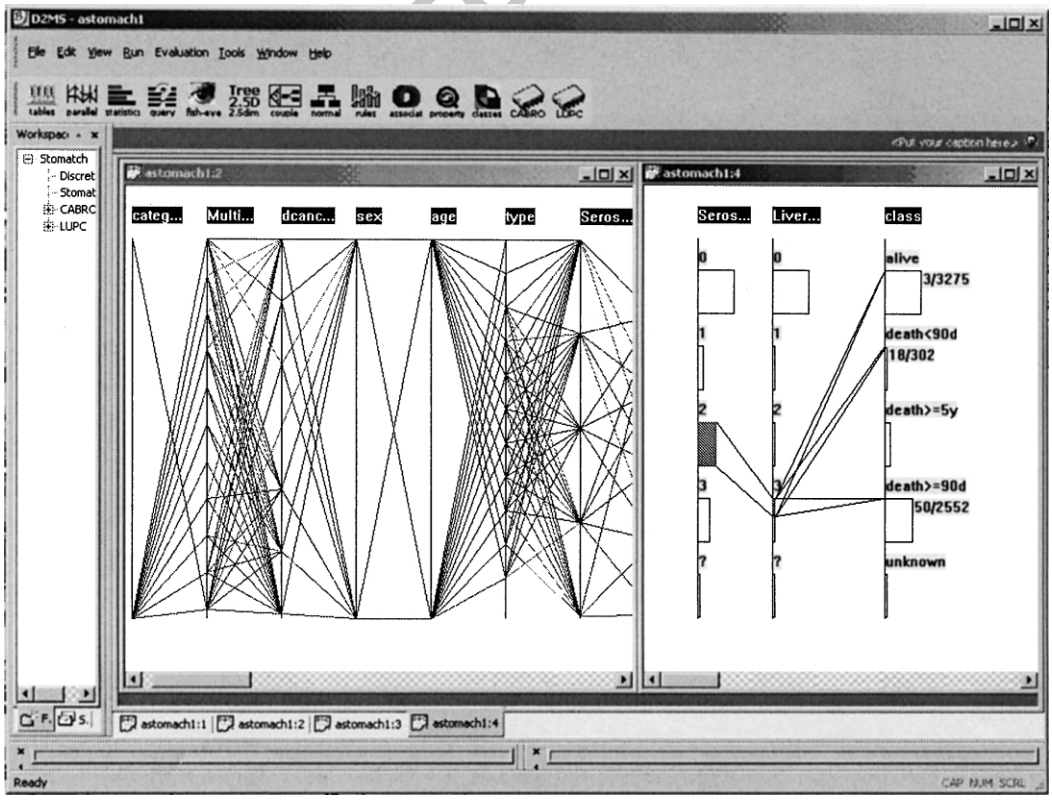



*Figure 5*. Visualization of data suggested rare events to be investigated.

*Table 1.* Number and quality of rules discovered by various data mining methods.

| | Nb. rules | | | | |
|---|---|---|---|---|---|
| Methods | cover $\geq$ 1, acc $\geq$ 50% | cover $\geq$ 7, acc $\geq$ 80% | cover $\geq$ 10, acc $\geq$ 50% | Method LUPC | Nb. rules |
| See5 (CF = 25%) | 16 | 0 | 1 | cover $\geq$ 1, acc $\geq$ 50% | 225 |
| See5 (CF = 99%) | 178 | 2 | 1 | cover $\geq$ 7, acc = 80% | 22 |
| CBA-CB | 106 | 1 | 0 | cover $\geq$ 7, acc $\geq$ 100% | 4 |
| CBA-CAR | 1,496 | 1 | 45 | cover $\geq$ 10, acc $\geq$ 50% | 45 |
| Rosetta | 19,564 | 5 | 0 | cover $\geq$ 10, acc $\geq$ 60% | 23 |
| | | | | cover $\geq$ 10, acc $\geq$ 70% | 6 |
| | | | | cover $\geq$ 10, acc $\geq$ 80% | 0 |

plans have been created in D2MS for finding rules of the target class "death within 90 days". The plans typically combine the feature selection algorithm SFG [34], rule induction method LUPC, and D2MS's visualization and exportation of discovered rules. SFG orders attributes according to their information gain, and the user can choose different subsets of attributes in the decreasing order of information gain. Note that SFG has also been used for preprocessing when applying See5, CBA, and Rosetta.

For the simplicity of model selection, we fixed the LUPC's default values of two parameters on the number of candidate attribute-value pairs ($\eta = 100$) as well as the number of candidate rules ($\gamma = 20$), and we varied two other parameters on minimum accuracy of rules $\alpha$ and minimum coverage of rules $\beta$. Given $\alpha$ and $\beta$, LUPC offers three types of search biases on either accuracy or coverage: (1) Rules are found with accuracy as high as possible, while coverage is kept equal or greater than $\beta$, (2) Rules are found with coverage as high as possible, while accuracy is kept equal or greater than $\alpha$, (3) Rules are found with both accuracy and coverage as high as possible by using alternatively two previous types of search biases with intermediate thresholds $\alpha'$ and $\beta'$, which are initially set with possible highest values and alternatively reduced until they become smaller than $\alpha$ and $\beta$, respectively.

Unlike other systems that often produce poor results in doing this task, LUPC allows us to run and visualize different models generated by these types of search biases, and select the best ones. Table 1 shows the number of rules discovered by four methods according to the required minimum of coverage (cover) and accuracy (acc) of rules. The left part of the table shows the number of rules discovered by See5, CBA (two modes CBA-CB and CBA-CAR), Rosetta, and the right part shows the number of rules discovered by LUPC. It is clear that with the same values of $\alpha$ and $\beta$, LUPC can find more and higher quality rules than the other systems. For example, when the required minimum coverage and accuracy are low (1 and 50%, respectively) See5, CBA, and Rosetta can find many rules, but when these thresholds are set highly (10 and 50%, 7 and 80%, respectively), these methods can discover only a few rules that cover a small number of instances in the target class. However, with these required thresholds on coverage and accuracy, or even higher thresholds, LUPC can find much more rules that cover a great number of instances of the target class as shown in the right part of Table 1.

***5.4.3. Finding Irregular Rules in Class "Death within 90 Days".*** It is commonly known that patients will die when liver metastasis occurs aggressively. Other learning methods when applied to this datasets often yield rules for the class "death within 90 days" containing "liver_metastasis" that are considered acceptable but not useful by domain experts. Also, these discovered rules usually cover only a subset of patients of this class. This low coverage means that there are patients of the class who are not included in "liver_metastasis" and, therefore, it is difficult to detect them.

Using visual interactive LUPC, we ran different trials and specified parameters and constraints to find only rules that do not contain the characterized attribute "liver_metastasis" and/or its combination with two other typical attributes, "Peritoneal_metastasis" and "Serosal_invasion." Below is a rule with accuracy 100% discovered by LUPC that can be seen as a rare and irregular event in the class.

Rule 8 accuracy = 1.0 (4/4),
cover = 0.001 (4/6712)
IF category = R AND sex = F
AND proximal_third = 3
AND middle_third = 1
THEN class = death within 90 days

***5.4.4. Finding Rare Events in Class "Alive".*** The prediction of rare events is becoming particularly interesting. When supposing that some attribute-value pairs may characterize some rare and/or significant events, LUPC, thanks to its associated visualization tools, allow us examine effectively the hypothesis space and identify rare rules with any given small support or confidence. An example is to find rules in the class "alive" that contain the symptom "liver_metastasis." Such events are certainly rare and influence human decision making. We found rare events in the class "alive", such as male patients getting "liver_metastasis" at serious level 3 can survive with the accuracy of 50%.

Rule 1 accuracy = 0.500 (2/4);
cover = 0.001 (4/6712)
IF sex = M AND type = B1
AND liver_metastasis = 3
AND middle_third = 1
THEN class = alive

## 6. Conclusion

We have presented the knowledge discovery system D2MS with support for model selection integrated with visualization. We emphasize the crucial role of the user's participation in the model selection process of knowledge discovery and have designed D2MS to support such a participation. Our basic idea is to provide the user with the ability of trying various alternatives of algorithm combinations and their settings, as well as to provide the user with performance metrics and effective visualization so that the user can get insight into the discovered models before making his/her final selection. D2MS with its model selection support has been used and shown to be advantageous in extracting knowledge from two real-world datasets, one on meningitis and one on stomach cancer.

The following objectives are under investigation: (1) to improve the construction of the system with described properties; (2) to validate and improve the effectiveness of the system through the use of it in real applications, in particular applications involving medical data with the participation of domain experts; (3) to enrich the system by adding other techniques for preprocessing, post-processing, and data mining, or by adding meta-knowledge in algorithm profiles and integrating on-line rules to the phase of plan registration.

## Acknowledgments

The authors would like to thank Prof. T. Tsumoto from Shimane Medical University for providing the meningitis data, Dr. N. Yamaguchi and his colleagues at the National Cancer Center in Tokyo for providing stomach cancer dataset. The authors also acknowledge Akinori Saitou, Saori Kawasaki, and DucDung Nguyen at JAIST for their contribution to this work.

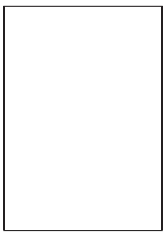
## Note

1. Given two thresholds $\alpha$ and $\beta$ on accuracy and coverage ratio, $0 \leq \alpha, \beta \leq 1$, a rule $R$ is not $\alpha\beta$-strong for any arbitrary $\beta$ if $cov^{-}(R) \geq \frac{1-\alpha}{\alpha} cover^{+}(R)$ where $cov^{+}(R)$ and $cov^{-}R$ denote the positive and negative coverage of the rule R.

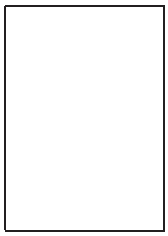
## References

1. H. Mannila, "Methods and problems in data mining," in *Inter. Conf. on Database Theory, Springer-Verlag*, 1997, pp. 41–55.
2. D.J. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*, The MIT Press, 2001.
3. J. Han and M. Kamber, *Data Mining. Concepts and Techniques*, Morgan Kaufmann, 2001.
4. D.H. Wolpert, "The relationship between PAC, the statistical physics framework, the bayesian framework, and the VC framework," in *The Mathematics of Generalization*, edited by D.H. Wolpert, Addison-Wesley, 1995, pp. 117–214.
5. R.J. Brachman and T. Anand, "The process of knowledge discovery in databases," in *Advances in Knowledge Discovery and Data Mining*, edited by U.M. Fayyad et al., AAAI Press/MIT Press, 1996, pp. 37–57.
6. A.W. Crapo, L.B. Waisel, W.A. Wallace, and T.R. Willemain, "Visualization and the process of modeling: A cognitive-theoretic view," in *Sixth Inter. Conf. on Knowledge Discovery and Data Mining KDD'00*, 2000, pp. 218–226.
7. T.D. Nguyen and T.B. Ho, "An interactive graphic system for decision tree induction," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 1, 1999, pp. 131–138.
8. T.D. Nguyen, T.B. Ho, and H. Shimodaira, "A scalable algorithm for rule post-pruning of large decision trees," in *Fifth Pacific-Asia Conf. on Knowledge Discovery and Data Mining PAKDD'01*, LNAI 2035, Springer, 2001, pp. 467–476.
9. T.B. Ho, "Discovering and using knowledge from unsupervised data," in *Decision Support Systems*, Elsevier Science, 1997, vol. 21, no. 1, pp. 27–41.

10. T.B. Ho, "Knowledge discovery from unsupervised data in support of decision making," *Knowledge Based Systems: Techniques and Applications*, edited by C.T. Leondes, Academic Press, 2000, pp. 435–461.
11. T.B. Ho, S. Kawasaki, and D.D. Nguyen, "Extracting predictive knowledge from meningitis data by integration of rule induction and association mining," in *Inter. Workshop Challenge in KDD*, JSAI Conference 2001, LNAI 2253, Springer, pp. 508–515.
12. H.P. Kumar, C. Plaisant, and B. Shneiderman, "Browsing hierarchical data with multi-level dynamic queries and pruning," *Inter. Journal of Human-Computer Studies*, vol. 46, no. 1, pp. 103–124, 1997.
13. G.W. Furnas, "The FISHEYE view: A new look at structured files," *Bell Laboratories Technical Memorandum* #81-11221-9, 1981.
14. T.D. Nguyen, T.B. Ho, and H. Shimodaira, "A visualization tool for interactive learning of large decision trees," in *Twelfth IEEE Inter. Conf. on Tools with Artificial Intelligence ICTAI'2000*, 2000, pp. 28–35.
15. D.J. Hand, *Construction and Assessment of Classification Rules*, John Willey & Sons, 1997.
16. W. Zucchini, "An introduction to model selection," *Journal of Mathematical Psychology*, vol. 44, pp. 41–61, 2000.
17. M.R. Forster, "Key concepts in model selection: Performance and generalizability," *Journal of Mathematical Psychology*, vol. 44, no. 1, pp. 205–231, 2000.
18. C.E. Brodley, "Recursive automatic bias selection for classifier construction," *Machine Learning*, 1995, vol. 20, pp. 63–94.
19. P.B. Brazdil and C. Soares, "A comparison of ranking methods for classification algorithm selection," in *Eleventh European Conf. on Machine Learning ICML'2000*, 2000, pp. 63–74.
20. M. Hilario and A. Kalousis, "Building algorithm profiles for prior model selection in knowledge discovery systems," *Engineering Intelligent Systems*, vol. 8, no. 2, pp. 77–87, 2000.
21. A. Kalousis and T. Theoharis, "NOEMON: Design, implementation and performance results for an intelligent assistant for classifier selection," *Intelligent Data Analysis Journal*, vol. 3, no. 5, pp. 319–337, 1999.
22. G. Nakhaeizadeh and A. Schnabl, "Development of multi-criteria metrics for evaluation of data mining algorithms," in *Third Inter. Conf. on Knowledge Discovery and Data Mining KDD'97*, 1997, pp. 37–42.
23. R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Inter. Joint Conf. on Artificial Intelligence IJCAI'95*, 1995, pp. 1137–1143.
24. C. Brunk, J. Kelly, and R. Kohavi, "MineSet: An integrated system for data mining," in *Third Inter. Conf. on Knowledge Discovery and Data Mining KDD'97*, 1997, pp. 135–138.
25. M. Ankerst, M. Ester, and H.P. Kriegel, "Towards an effective cooperation of the user and the computer for classification," in *Sixth Inter. Conf. on Knowledge Discovery and Data Mining KDD'00*, 2000, pp. 197–188.
26. J. Han, and N. Cercone, "RuleViz: A model for visualizing knowledge discovery process," in *Sixth Inter. Conf. on Knowledge Discovery and Data Mining KDD'2000*, 2000, pp. 244–253.
27. G.G. Robertson, J.D. Mackinlay, and S.K. Card, "Cone trees: Animated 3D visualization of hierarchical information," in *ACM Conf. on Human Factors in Computing Systems*, 1991, pp. 189–194.
28. J. Lamping and R. Rao, "The hyperbolic browser: A focus + context techniques for visualizing large hierarchies," *Journal of Visual Languages and Computing*, vol. 7, no. 1, pp. 33–35, 1997.
29. R. Kohavi, D. Sommerfield, and J. Dougherty, "Data mining using MLC++, a machine learning library in C++," *International Journal of Artificial Intelligence Tools*, vol. 6, no. 4, pp. 537–566, 1997.
30. C. Domslak, D. Gershkovich, E. Gudes, N. Liusternik, A. Meisels, T. Rosen, and S.E. Shimony, "FlexiMine—A flexible platform for KDD research and application construction," in *Fourth Inter. Conf. on Knowledge Discovery and Data Mining KDD'98*, 1998, pp. 184–188.
31. J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
32. J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," in *Twelfth Inter. Conf. on Machine Learning ICML'95*, 1995, pp. 194–202.
33. Y. Fujikawa and T.B. Ho, "Cluster-based algorithms for filling missing values," *6th Pacific-Asia Conf. Knowledge Discovery and Data Mining*, Lecture Notes in Artificial Intelligence 2336, Springer, 2002, pp. 549–554.
34. H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*, Kluwer Academic Publishers, 1998.
35. L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Wadsworth: Belmont, CA, 1984.
36. J. Mingers, "An empirical comparison of selection measures for decision tree induction," *Machine Learning*, vol. 3, pp. 319–342, 1989.
37. N.B. Nguyen and T.B. Ho, "A mixed similarity measure in near-linear computational complexity for distance-based methods," in *4th European Conf. on Principles of Data Mining and Knowledge Discovery PKDD'2000*, LNAI 1910, Springer, 2000, pp. 211–220.
38. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, "From data mining to knowledge discovery: An overview," in *Advances in Knowledge Discovery and Data Mining*, edited by U.M. Fayyad et al., AAAI Press/MIT Press, 1996, pp. 1–36.
39. E.M. Reingold and J.S. Tilford, "Tidier Drawings of Trees," *IEEE Transactions on Software Engineering*, vol. SE-7, no. 2, pp. 223–228, 1991.
40. J. Mingers, "An empirical comparison of pruning methods for decision tree induction," *Machine Learning*, vol. 4, pp. 227–243, 1989.
41. J. Furnkranz, "Separate-and-conquer rule learning," *Journal Artificial Intelligence Review*, vol. 13, pp. 3–54, 1999.
42. S. Tsumoto, "Comparison and evaluation of knowledge obtained by KDD methods," *Journal of Japanese Society for Artificial Intelligence*, vol. 15, no. 5, pp. 790–797, 2000.
43. B. Liu, W. Hsu, and Y. Ma, "Integrating classification and association rule mining," in *Fourth Inter. Conf. on Knowledge Discovery and Data Mining KDD'98*, 1998, pp. 80–86.
44. A. Ohrn, *Rosetta Technical Reference Manual*, Norwegian University of Science and Technology, 1999.

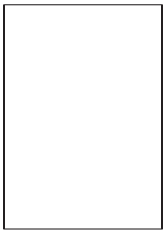


**Tu Bao Ho**

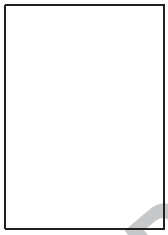


**Hiroshi Shimodaira**

Au: Pls. provide authors photo and bio.



**Trong Dung Nguyen**



**Masayuki Kimura**