Paper from: *Data Mining III*, A Zanasi, CA Brebbia, NFF Ebecken & P Melli (Editors).
ISBN 1-85312-925-9

KDB2000: An integrated knowledge discovery tool

A. Appice, M. Ceci & D. Malerba
Dipartimento di Informatica, University of Bari, Italy.

Abstract

Knowledge discovery in databases (KDD) is not a straightforward application of a single method, but rather a long lasting, iterative and interactive process during which the user has to model a multitude of data derivation processes, execute them and interpret the results in order to formulate new derivation processes. Having a single tool that supports the user in the selection of data, their preprocessing and transformation, as well as their mining, is an important requirement of KDD practitioners. Moreover, it is important to assist the user of a KDD system in the selection of the right parameters or methods.
This paper describes KDB2000, a tool that integrates database access, data preprocessing and transformation techniques, a full range of data mining algorithms as well as pattern validation and visualization. This integration aims to support the user in the entire KDD process enabling him/her to see the same problem from many different angles for a thorough investigation. In addition, KDB2000 makes use of the agent technology to assist the user in some data mining tasks, such as choosing the best pruning strategy for induced decision tree in accordance to both data features and used needs.

1 Introduction

The amount of data being collected in databases today far exceeds our ability to analyze them without the use of automated analysis techniques. The field of *knowledge discovery in databases* (KDD) is evolving to provide automated analysis solutions. Knowledge discovery is defined as the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data and describing them in a concise and meaningful manner [1]. This process is interactive and iterative, involving numerous steps with many decisions being made by the user [2]. Information flows forwards from one stage to the next, as well as backwards to previous stages. The main step is *data mining*, which involves in the selection of the best algorithm for the task in hand, the application of the algorithm to the selected data and the generation of plausible patterns occurring in the data. Although this is clearly important, the other steps of the KDD process are equally critical for successful applications of data mining algorithms to real-world data.

The need to analyze and extract useful knowledge from the volume of data collected in real applications leads to a generation of systems that automate knowledge discovery from large databases. Currently, most commercial data mining systems produce only a description of the knowledge discovered from data sets according to a limited set of tasks, such as classification or rule extraction. Moreover, there are some underestimated aspects to most of the current tools, such as the detecting and removing noise from selected data or the evaluation of data mining results. Consequently, the data analyst is forced to work with different tools in order to select, process and mine the data properly, as well as to visualize and evaluate the results. Difficulties due to the different data representations and results require the development of *integrated* KDD systems, which support the users in all steps of the knowledge discovery process.

This paper provides an overview of the KDD system KDB2000 (www.di.uniba.it/~malerba/software/KDB2000) and illustrates how it integrates database access, data preprocessing and transformation techniques, analytical data mining, as well as pattern validation and visualization. In addition to supporting users in *all* the steps of the KDD process, the system also assists the user in the choice of some critical parameters in some data mining tasks.

The paper is organized as follows. In the next section, a brief overview of the architecture of the system is reported. In Section 3, we explain how it is possible to manage the KDD process step-by-step by using KDB2000. Finally, the development of a user assistant for the selection of a decision tree pruning algorithm is reported in Section 4.

2 System architecture

The general architecture of KDB2000, shown in Figure 1.a, integrates a relational database management system, such as Microsoft Access, with a set of Data Mining modules, as well as with Extraction, Transformation and Loading techniques, Visualization and Evaluation tools.

Paper from: *Data Mining III*, A Zanasi, CA Brebbia, NFF Ebecken & P Melli (Editors).
ISBN 1-85312-925-9

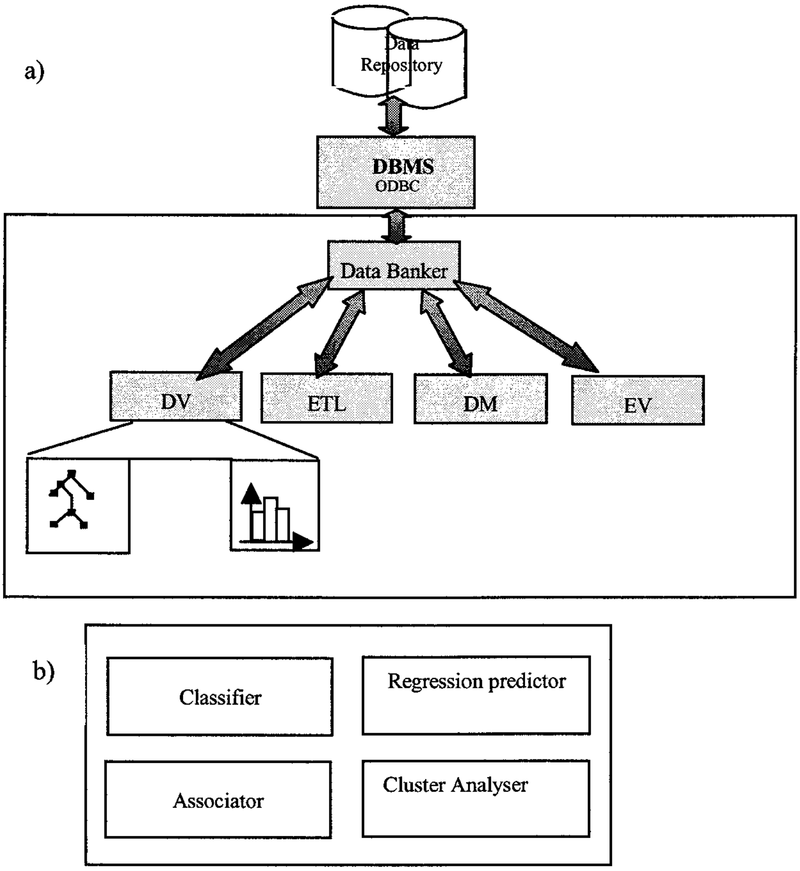


Figure 1: KDB2000: a) General architecture; b) Data Mining modules.

The *Data Banker* component allows access to data by some external relational data sources using the ODBC driver. It is the Data Banker's job to connect to a specified database, execute the SQL request, and return the result to the system making the request in the expected format. This component also allows the requesting system to obtain metadata on table names, table columns, column data types and so on. This information can be used as part of the data processing. Therefore, Data Banker extracts data from the data source and orchestrates the moving of it from one KDB2000 component to another. It also includes a storage subsystem that holds the user data both in a database and in a file.

Data Visualization tools (DV) are used to display data selected by the user and patterns discovered by the mining algorithms. This component is driven by the need to visualize the data resulting from any KDD step (from selection to mining) in order to permit an interactive exploration of large volumes of data.

The *Extraction*, *Transformation* and *Loading* techniques (ETL) operate at the heart of the KDD, process extracting data from data sources and transforming the data to make them accessible to business analysis. Functions supported by these tools go beyond the simple extraction of data. They are designed to take data in various format, cleanse and transform them into a usable format and load them into any repository.

The *Data Mining* module (DM) of KDB2000, shown in Figure 1.b, includes powerful algorithms to discover patterns in selected data according to the user task (classification, regression, clustering or association rule discovery). More precisely, the *classifier* analyses a set of objects whose class label is known and constructs a model for each class, based on the features in the data. The *regression predictor* learns a function that maps trained objects into a real-valued prediction variable. The *associator* discovers a set of association rules and the *cluster analyzer* groups a selected set of objects into a set of clusters.

Finally KDB2000 architecture provides an *Evaluation* module (EV) to help in the collection of results produced by performance experiments on the data mining algorithms.

3 Managing the KDD process with KDB2000

Fayyad, Piatetsky-Shapiro and Smyth [2] identify nine steps in the KDD process. In this paper we present a simplified version consisting of five stages, namely data selection, data pre-processing, data transformation, data mining, and evaluation/interpretation of mined patterns.

In most cases we begin with an original database, usually developed for tasks other than KDD. According to the end-user goals, a target data set is created by means of extracting the data that is needed for the preliminary analysis, in preparation for further mining. For each of the selected variables, associated semantic information, the metadata, it is necessary to understand what each of the variables means. The metadata could include the business definitions of selected data, the description of data types, potential values, original source and data formats.

In the data acquisition step, the KDB2000 user can select an existing data source or can create a new data source to access and manage database information. Data of interest for the application (or target data) can be selected by means of an SQL query. The query can be created:

- by using a text editor (for advanced users),
- by using a visual interface, or
- by recalling a previously issued query stored in a repository.

The visual interface allows the user to create complex queries without using SQL statements. It guides the user through the process of building a query, step by step, selecting the tables involved in the query, creating joins with selected tables, choosing the columns, deciding how far to summarize the data, how to calculate the summaries, how to sort the results and choosing filters to include or exclude data according to some search criteria.

Target data could contain errors, introduced during the input process, either by a clerk entering data incorrectly or by a faulty data collection device. The aim of data pre-processing (second step) is to ensure the quality of the selected data and to make the data consistent and clean. The data pre-processing step begins with a general review of the data structure and some assessing of its quality. Statistical methods and data visualization techniques are used in basic operations such as removing outliers, if appropriate, collecting the necessary information to model the noise or deciding on strategies for handling missing data.

In KDB2000 the user can use both statistical methods and data visualization techniques to ensure the quality of selected data. Statistical methods are also a useful way to understand the data content better. The system computes some basic statistics for each attribute, such as mean, mode, median, maximum, minimum, correlation coefficient and standard deviation. When combined, these measures provide a means for determining the presence of invalid and skewed data. Data visualization techniques are used to help the user understand and/or interpret data. Histograms and pie charts are available in the system. Visualization techniques help to identify distribution skews and invalid or missing values.

In the third step, the data could be transformed to produce the analytical data model that is needed for use in mining techniques. The analytical data model represents a consolidated, integrated, time dependent restructuring of the data which have been selected and pre-processed. This is a crucial step as the accuracy and validity of the final results depends on how the data analyst decides to structure and present the input. KDB2000 provides the following data transformation tools:

- *discretization*: converts quantitative variables into categorical ones by dividing the values of the input variables into intervals. Intervals are regular or preserve a uniform distribution of data.
- *scaling*: most data mining algorithms can accept only numeric input in the range [0.0,1.0] or [-1.0,1.0]. In these cases, continuous variable values must be scaled.
- *null value substitution*: substitutes null values of continuous and discrete variables with mean and mode, respectively.
- *sampling*: extracts a random set of items from the specified dataset. KDB2000 does the sampling using a *reservoir algorithm* (algorithm Z) [3] which selects a random sample of n objects without replacing them from a pool of N objects, where the value of N is previously unknown. The algorithm Z outperforms current sampling methods in the sense that it extracts a random sample of a set of objects in one pass, using constant space and in $O(n(1 + \log(N/n)))$ expected time, which is optimum.
- *binarization*: converts a discrete variable into a numeric representation using a set of attributes, one attribute for each possible nominal value. This transformation is especially useful for

association rule mining, where binary representations of transactions are required for typical basket analysis.

The data mining step is at the hearth of KDD process and uses the output of the previous steps. Data mining algorithms can be used to discover trend, characteristics, anomalies and more generally unknown patterns. The appropriate data mining algorithm should be identified, as it should be pertinent to the type of data to be analyzed and the purpose of the analyses (classification, regression, clustering, association rule discovery, and so on).

KDB2000 provides several algorithms for the following data mining tasks: classification, regression, clustering and association rule discovery.

Classification means learning a function that classifies a data item into one of several predefined classes. KDB2000 supports classification by means of a k-nearest neighbour (*K-NN*) algorithm or a decision tree induction algorithm. K-NN instance-based learning is simple and often works very well: the training examples are stored verbatim, and a distance function is used to determine which member of the training set is closest to an unknown classification instance. Once the nearest training instance has been located, its class is predicted for the classification instance [4]. The decision tree algorithm [5] builds a predictive model in the form of tree. Decision trees often suffer from the fragmentation problem. To minimize fragmentation the user can prune the tree using simplification techniques [6]. An expert agent assists the user in choosing the appropriate pruning method according to the dataset features (see next section).

Regression is a learning function that maps a data item into a real-valued prediction variable. KDB2000 supports regression by means of an innovative model tree induction algorithm named SMOTI (Stepwise MOdel Tree Induction) [7]. In SMOTI the top-down induction of model trees is performed stepwise by considering regression steps and splitting tests at the same level. This means that there are two types of nodes in the tree: regression nodes and splitting nodes. The former compute straight-line regression, while the latter partition the feature space. The complete piece-wise regression function is computed by combining the straight-line regression associated to each leaf with all univariate regression lines associated to the regression nodes along the path from the root to the leaf. Induced model trees can also be simplified by using a grafting or pruning operator. In addition, if required, every model tree can be translated into a set of rules, one for each leaf. The rules are in the form X->Y, where X is a conjunction of conditions on the attribute values and Y is the multiple linear regression function associated with the leaf.

Clustering is a descriptive task where one seeks to identify a finite set of *categories* or *clusters* to describe the data. Similar data items can be seen to be generated from the same component with a mixture of probability distribution. The clustering problem is to determine the parameters of the mixture distribution that generated a set of observed data items, where, for each item, the generating component is an unobserved feature. The algorithm available in KDB2000 performs k-means clustering in one scan of the dataset [8]. The k-means algorithm is a heuristic solution to the clustering problem based on the assumption that data points are drawn from a fixed number *k* of spherical Gaussian distributions. The algorithm is an iterative process of assigning cluster members and re-estimating cluster parameters. It terminates when data points no longer change their membership due to changes in the re-estimated cluster parameters. The algorithm requires one scan of the dataset because it operates within the confines of a limited memory buffer using a sampling-based approach.

Association rules discover a model that describes relations between attributes, by means of Apriori, AprioriTid and AIS algorithms [9] [10]. These relations are rules in the form X→Y, where X and Y are a set of binary literals called items and $X \cap Y = \emptyset$. Since the available association rule mining algorithms are devoted to binary data tuples, it is necessary for discrete variables to be transformed into a binary representation. In KDB2000 this is performed automatically by means of the binarization function, already implemented for the data transformation facilities. When a rule is identified it is *evaluated* in terms of support and confidence measures. Furthermore, the system assists the user to define of the taxonomies (*is-a* hierarchies) between classes of items [11], in order to extract generalized rules involving classes of attributes rather than single attributes. The hierarchies allow knowledge mining at multiple abstraction levels. They can be used in a *roll-up* operation. Patterns can be rolled up, or viewed at a more general level, by climbing up the concept hierarchy of an attribute replacing a lower level concept with a higher level one. Finally, the user can set filters according to user-defined templates, that provide a template of the association rules to be discovered.

When a pattern is identified, it should be *interpreted* and *evaluated* to determinate whether it is correct, novel and useful, according to some quality measures. This is the last step of the KDD process. When a mined pattern is considered relevant it can be incorporated into the knowledge base for use in subsequent iterations or simply for informing an interested user.

KDB2000 supports the user in the evaluation of the predictive accuracy of decision and model trees, by means of the *k-fold Cross-Validation* (k-CV) approach. In the k-CV approach, each dataset is divided into *k* blocks of near-equal size and near-equal distribution of class values, and then, for every block, the (decision or regression) method is trained on the remaining blocks and tested on the hold-out block. The estimate of the accuracy of the decision or model tree, built on the whole set of training cases, is obtained by averaging the accuracy computed for each hold-out block.

Note that, at many of the KDD steps, it may be necessary to go back to a previous step. For example, if no relevant patterns are discovered, new data should be selected for further analysis.

4 An intelligent assistant

The entire KDD process is user-centered and the user makes decisions on scientific aspects, e.g. the selection of an analysis method depending on characteristics of the data and the goal of the task. Indeed some efforts in the KDD field have been directed towards intelligent support of the KDD process.

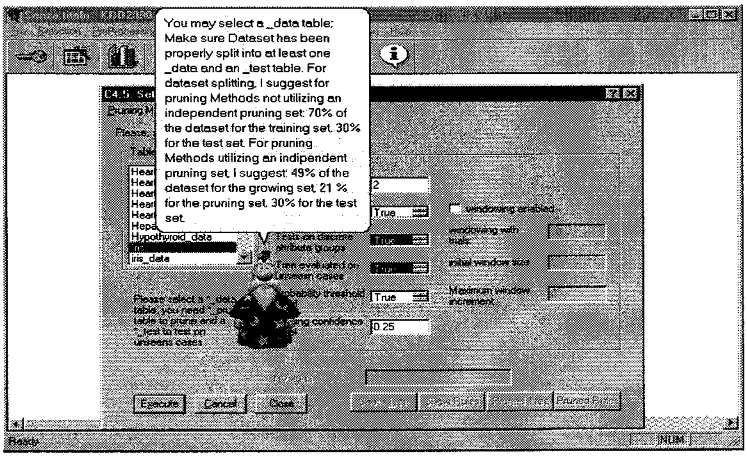
KDB2000 provides an intelligent assistant, named "KD-WIZ", which assists users in some critical choices, such as the appropriate pruning method for induced decision trees, according to the dataset features and user preferences (see Figure 2). The pruning method and parameters are evaluated on the basis of the following factors:

- The underlying model.
- The presence (or absence) of a dominant class.
- The theoretical bias defined by the user.

The underlying model is investigated in order to discover the presence of univariate or multivariate tests. The former are in the form of *age>10* or *rings <5* or *sex="M"*, while the latter are in the form of *age > rings*. If the underlying model is univariate, it may be represented by a smaller tree and a restrictive pruning method can be used [12]. Furthermore, if the underlying model is multivariate, it is advisable to use pruning methods that do not work on an independent pruning set, because the regions produced by a decision tree are all hyperrectangles. Therefore, it is advisable to train the decision tree on the entire training set to obtain a better predictive accuracy [5].

The goal of a decision tree induction algorithm is to partition the feature space to produce regions that contain points of a single class. This goal cannot be attained if the density points in a region are low. This is a typical example of a dominant class. Therefore, it is advisable to train and prune the decision tree on the entire training set to increase the number of training examples which fall into non-dominant classes [5].

Finally, the theoretical bias shows the user's preference on the characteristics of the pruned tree. In particular, the tree can be accurate or simple. The different user bias will affect the recommendation made by the agent.



You may select a _data table; Make sure Dataset has been properly split into at least one _data and an _test table. For dataset splitting, I suggest for pruning Methods not utilizing an independent pruning set: 70% of the dataset for the training set. 30% for the test set. For pruning Methods utilizing an indipendent pruning set, I suggest: 49% of the dataset for the growing set. 21 % for the pruning set. 30% for the test set.

Please select a *_data table, you need *_pr table to prune and a *_test to test on unseens cases

Figure 2: KDB2000: Expert agent that assists the user in choosing the appropriate pruning method for the classification task.

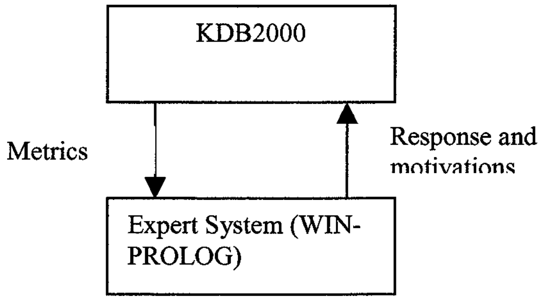


Figure 3: KDB2000: the interaction with the expert system of KD-WIZ.

KD-WIZ is composed of a *graphical interface*, a *measure extractor* and an *expert system*. The *graphical interface* interacts with the user, analyses his/her requests and visualises the results by means of the *msagent* technology[13]. The *measure extractor* computes some metrics to estimate the previously mentioned factors. The *expert system* (see Figure 3) uses the metrics to give advice on the appropriate pruning method and relative motivations.

5 Future research

KDB2000 is being extended in several directions. We are currently concentrating on identifying new data mining tasks and developing fast algorithms for their execution. Future developments of the system will include 3D-visualization tools and new simplification methods for model trees. We are also interested in improving the scalability and run-time performance of KDB2000 as well as its extension to more complex data sets with spatio-temporal dimensions.

Acknowledgment

This work is part of the MURST COFIN-2001 project on “Methods for the extraction, validation and representation of statistical information in a decision context”. The authors thank Lynn Rudd for her help in reading the paper.

References

[1] Frawley, W.J., Piatetsky-Shapiro, G., and Matheus, C. Knowledge Discovery In Databases: An Overview. In Knowledge Discovery In Databases, eds. G. Piatetsky-Shapiro, and W. J. Frawley, AAAI Press/MIT Press: Cambridge, MA., pp. 1-30, 1991

[2] Fayyad, U.M., Piatetsky-Shapiro, G., and Smyth, P. From Data Mining To Knowledge Discovery: An Overview. In Advances In Knowledge Discovery And Data Mining , eds. U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and

R. Uthurusamy, AAAI Press/The MIT Press: Menlo Park, CA., pp. 1-34, 1996.

[3] Vitter J.S. Random Sampling with a Reservoir, ACM Transactions on Mathematical Software, University of Brown, pp. 37-57, 1985

[4] Mitchell T. Machine Learning, McGraw Learning, 1997

[5] Quinlan J.R. C4.5 : Programs for Machine Learning. Morgan Kaufmann: San Matteo, California, 1993

[6] Esposito F., Malerba D., & Semeraro G. A Comparative Analysis of Methods for Pruning Decision Trees. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-19, 5, pp. 476-491, 1997

[7] Malerba D., Appice A., Ceci M., Monopoli M. Trading-off Local versus Global Effects of Regression Nodes in model Trees. 13th International Symposium on Methodologies for Intelligent Systems: Lyon, pp. 27-29 June 2002

[8] Farnstrom F., Lewis J., Elkan C. Scalability for clustering algorithm revisited. In SIGKDD Esplorations, 2(1), pp. 51-57, June 2000

[9] Agrawal R., Imielinski T., & Swami A. Mining association rules between sets of items in large databases. Proc. of the ACM SIGMOD Conference on Management of Data: Washington, D.C., 1993.

[10] Agrawal R., Srikant R. Fast Algorithms for Mining Association Rules. Proc. of the Twentieth VLDB Conference: Santiago, Cile,1994

[11] Srikant R. and Agrawal R., "Mining Generalized Association Rules", *Proc. of the 21st Int'l Conf. on Very Large Databases*: Zurich, Switzerland, Sep. 1995

[12] Shaffer, C. *Overfitting avoidance as bias*. Machine Learning, 10, 153—178, 1993.

[13] MsAgent technology, http://msdn.microsoft.com/msagent