Knowledge Discovery in Data with FIT-Miner

Michal Šebek, Martin Hlosta and Jaroslav Zendulka

Faculty of Information Technology, Brno University of Technology,
Božetěchova 2, Brno
{isebek,ihlosta,zendulka}@fit.vutbr.cz

Abstract. The paper deals with a data mining system FIT-Miner that has been developed at the Faculty of Information Technology of the Brno University of Technology. The system is based on a component approach where the essential functionality is encapsulated in components. A data mining task is specified graphically as a network of interconnected components. This approach makes good maintainability and scalability possible. The FIT-Miner includes components for data preprocessing, data mining and presentation of results. Implemented data mining algorithms cover all typically mined kinds of knowledge, such as frequent patterns, association rules; and classification, prediction and clustering models.

Keywords: Data mining system, data preprocessing, frequent patterns, classification, prediction, clustering.

1 Introduction

A lot of data has been collected and stored in databases since data intensive application appeared in 60s of the last century. The speed of the growth has increased during last two decades as a result of multimedia processing and, mainly, as a result of worldwide using of the Internet and its service WWW. In addition, currently there are other data sources than databases that produce huge amount of data that cannnot or need not be stored permanently, as sensor networks, remote sensing, or measurements. Interesting and potentially useful patterns or implicit knowledge can exist in this data. Data mining is a branch that provides methods and techniques to discover this knowledge in huge amount of data. Many data mining algorithms have been developed and implemented. Some of them are currently available in data mining systems, packages of algorithms and libraries as commercial or open source solutions. There are also several standards concerning data mining, for example SQL/MM Data Mining, JSR-073 Data Mining API, PMML. Support for analytic functions and data mining is now provided by database servers of database producers as an extension of the basic DBMS functionality, for example Oracle Data Mining (ODM), Microsoft SQL Server Analysis Services, IBM DB2 Data Warehouse Edition (replaced the former DB2 Intelligent Miner).

SAS Enterprise Miner, which is a part of SAS analytics software [15] is an example of the complex commercial system for data mining. The system provides a graphical interface for definition of the data mining task and a number of different analytics and mining methods. Another example of the data mining system we can mention is *Rapid Miner* [14]. Several systems are results of academic projects. Probably the most popular is a *WEKA project* [17] which provides a suite of machine learning software written in Java under GNU General Public License. In our country, one of such projects is a *LISp-Miner* [12] which includes algorithms for association analysis and classification. It is based on the GUHA technique of mechanized hypothesis formation [6].

In this paper we introduce a data mining system called FIT-Miner, which has been developed at the Faculky of Information Technology of the Brno University of Technology. The seeds of the system were sown by the Peter Kotasek's disseration, in which he developed a DMSL (Data Mining Specification Language) [9]. It is an XML-based language purpose of which is to provide the framework for platform-independent definition of the whole data mining task, and exchange and sharing of data mining projects among different applications, possibly operating in heterogeneous environments. The idea of the DMSL was similar to one of the PMML, but the latter focused mainly to mining models representation. Instead, the DMSL provides support for representation of the whole process.

There were two reasons that motivated us to develop our own system. First, we needed a data mining system that could be used in our course Knowledge discovery in databases (open in the academic year 2006/7 for first). We required the system to be simple enough so that students could easily use it in their project. Second, we wanted to involve students in data mining tools development by providing them with appropriate topics of their bachelor's and master's thesis. Although several leading producers of data mining and business intelligent solutions offer academic licenses of their data mining systems that could be used in the course for universities, the prices are still high. In addition, modifications and extensions of the systems by including additional algorithms developped by students are very limitted, if any. On the other hand, the usage of free and open source solutions is not often so easy as we would require.

The organization of the paper is following. In section 2, the system structure and basic functionality is introduced. The section 3 deals with concrete data mining algorithms which are implemented in the FIT-Miner. Finally, section 4 summarizes our work and presents ideas of future work.

2 FIT-Miner Overview

The development of the system started in 2002. The first version of the system was implemented in Java with MySQL database. In 2005, we replaced MySQL with Oracle database and used some built-in data mining functionality for preprocessing provided by Oracle DBMS. Since 2006 a new version of the system implemented in Java on the NetBeans Platform has been developed. It uses Oracle database not only to store data and to perform preprocessing operations

but it also employs built-in functionality provided in Oracle 10g and Oracle 11g. This version of the FIT-Miner is presented in the following sections.

2.1 FIT-Miner Architecture

The system has a client-server architecture. This type of architecture is suitable because an efficient data access method is provided natively by the server and client requests can be preprocessed and postprocessed on the client side of the system. We use Oracle 10g or 11g) [13] as a database server. We employ not only its basic database functionality, but also its Oracle Data Mining extension (ODM) [13], which provides implementation of several algorithms for mining various kinds of knowledge. The client part of the system is implemented in Java on the the Netbeans Platform. As a result, this part is completely portable and based on unified GUI of Java project management.

The logical architecture of the FIT-Miner is presented in Figure 1. It shows a Data layer and three layers of the FIT-Miner. The Data layer contains an Oracle database and a DMSL document. The database stores data to be mined and some stored procedures for preprocessing operations. The DMSL document represents the data mining task. It contains information which is produced and shared by all components of the FIT-Miner.

The main idea of the design of the three other layers was to separate each essential functionality of the system and to make easy extension and modification possible. As a result, the system contains a Kernel and Mining Components. The Kernel consists of a data abstraction layer with Database Access and DMSL Access partitions, a System Core layer and a Kernel Components partition of the Plugable Modules layer. The Database Access and the DMSL Access provide an abstraction of a database connection and an interface to DMSL document, respectively. The System Core layer controls a data mining task composition and its run. The Plugable Modules layer contains a set of plugable components. A component is a fundamental concept in FIT-Miner. It encapsulates one basic step of the data mining task that is represented by an icon in a graphical specification of a data mining task (it will be described in section 2.2). The Kernel Components partition includes components of data selection, loading and preprocessing. The Mining Components partition includes components of data mining modules that implement algorithms for mining a particular kind of knowledge, for example association rules. Both kernel and mining components can be easily added.

2.2 Component-Based Definition of the Data Mining Task

A fundamental question of each data mining system is how to specify the data mining task. To find a simple solution is not easy because the data mining task is composed of many steps (subtasks). The basic structure of a data mining task in the data mining system can be the following [5]:

- data loading,

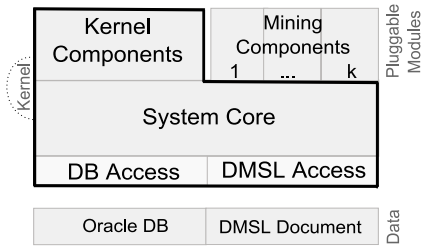

Fig. 1. Architecture of the FIT-Miner system.

– data preprocessing,
– data mining,
– presentation of the results.

Because the data mining task is a sequence of subtasks, we can encapsulate each subtask into components of the system. Then, the data mining task is represented by an oriented graph where nodes are components. Nodes cannot be interconnected randomly, for example oriented connection from a data mining result to the data loading component is not possible. Such logic restrictions are expressed as constraints of connections in the graph. In FIT-Miner, the user constructs the graph interactively by means of icons representing the components. It provides a simple and clear way of the data mining task specification. This graf will be referred to as a *task-graph* in this paper. An example is shown in Figure 2. The task-graph specifies a data source from which the data to be mined is loaded. Then, some transformations of the dataset such as data cleaning or attribute construction. In this example, the required data mining functionality is classification. Therefore, the partitioning of the dataset to subsets for training and testing is necessary. After that, the classification model is built and tested. In this case a Decision Tree and Naive Bayes classification methods are employed. Finally, the results are visualized.

In the previous section, the FIT-Miner components were classified into kernel and mining ones. In this example, Select Data, VIMEO, Transformations, Reduce/Partition and Insight are kernel components. The others are mining components.

2.3 DMSL and a DMSL document

The data mining system has to store the task specification internally. The FIT-Miner uses a DMSL document for this purpose. The DMSL language identifies five main primitives that play the major roles in the data mining task specification. These primitives are represented by five sections of a DMSL document: the data model, the data mining model, the domain knowledge, the mining task, the knowledge. Thus, each DMSL document can carry the complete information about a single data mining project. Primarily, DMSL is aimed to capture

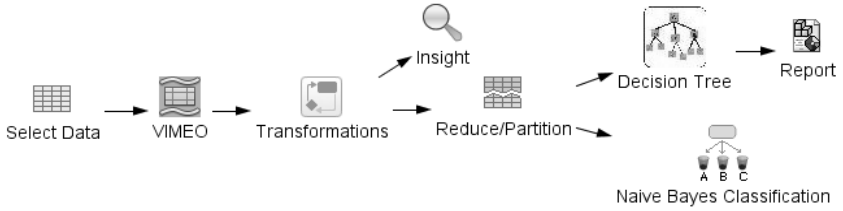

Fig. 2. A data mining task specified in the FIT-Miner.

all the information relevant to the whole data mining project, and to enable applications exchange and share projects. The notion of 'data mining project' refers to a particular instance of the data mining task. The sections have a form of XML elements in the document. The DataModel element represents a data schema that defines the shape of initial input data to be mined. The DataMiningModel element represents a data mining model that defines transformations of the initial input data defined by DataModel. The DataMiningTask element specifies the data mining step of the whole process. It is performed over a data mining model. The task can employ domain knowledge that is described by the DomainKnowledge element. The result of the data mining is held by the Knowledge element [10].

The structure of the DMSL document is mapped to corresponding objects in the System Core layer. Components of the data mining task use the document both as their input and output, i.e. they both read and create the content of the document. For example the Decision Tree component from Figure 2 reads metadata of the preprocessed and partitioned dataset created by the preceeding components in the task-graph and creates the DataMiningTask and Knowledge elements in the document.

2.4 Basic Kernel Components

The kernel components are shared by all kinds of data mining tasks. In the kernel of the FIT-Miner, the shared functionality is the data loading, preprocessing and a common base of result reporting. Following components are included in the FIT-Miner kernel [11, 16].

Select Data – data selection and loading is the first subtask of each data mining task. The Select Data component is responsible for this functionality in the FIT-Miner. The component displays a list of tables and their columns in a source database and the user selects tables and columns to be used in the data mining task. The component supports some advanced processing of the source data such as the joining of tables or import from CSV data files. This component deals with a special type of column which only references into an enumeration-type table. All selected columns are joined into one output table which is passed to next component of the task-graph.

VIMEO functions – input data are often incomplete or there are some noisy values in the dataset. As a result, the dataset has to be cleaned. The FIT-Miner includes a VIMEO component for this purpose. VIMEO is an abbreviation for a set of markers: Valid, Invalid, Missing, Empty and Outlier [9]. The cleaning is performed by VIMEO functions that accept the value to be cleaned and it returns a VIMEO marker.

The logical view of VIMEO functions is such that the marker of each value in the dataset is determined by a user-defined condition (e.g. a numeric value has to be from defined interval). The condition is checked for each value and the value is marked. The post-processing of the value depends on the marker and possible actions are to pass value, replace it with a defined value (e.g. mean, constant) or to remove the record with the value from the dataset.

Because of an efficiency the implementation is different. We have designed an algorithm which converts conditions to an optimized sequence of SQL statements (usually some sequence of UPDATE statements with complex WHERE condition). Then the marking and value post-processing is evaluated together.

Transformations – transformation of values in columns is another important step of the data mining task. Discretization, normalization, smoothing and dimensionality reduction can be performed by this component. The Oracle ODM functionality was used as much as possible for the implementation of transformation subtasks because of its high performance.

In addition, the component supports a construction of a new attribute through user-defined functions. In practice, a new attribute is derived (evaluated) from existing columns. A target column can be a numeric or a categorical attribute represented by string value. This function can be useful when the new attribute can be derived from several existing attributes. This trasformation is useful for dimensionality reduction. These user-defined functions are similar to VIMEO functions, only the type of the result differs. The functions are also converted to SQL statements and applied to data on the server-side of the system.

Reduce/Partition – some supervised mining algorithms require more than one input data set – a training dataset and a testing data set. The model is trained on the training set and verified on the testing dataset. The Reduce/Partition component allows splitting the input dataset into the training dataset and the testing one.

In some cases a quantity of the dataset can be enormous and data reduction is necessary. This subtask has to be performed by the Reduce/Partition component too. The reduction is based on sampling methods such as a sampling with/without return or a stratified selection.

Insight – the first step of the analysts is usually to get some basic statistical information about the input dataset. The Insight component provides functions of statistical analysis and data visualization. Most of the functions evaluate standard statistical measures over one column of the dataset. The supported basic measures include mean, median, modus, standard deviation, quartiles and deciles. These measures help to discover or estimate the distribution of the values

in a column. In addition, the analyst can use a χ^2-test for testing correlation of pairs of columns.

Usually the best way how to view the data is to visualize statistical information insted of displaying numeric statistical measures. The Insight component supports a number of different types of one and two dimensional charts such as histograms, quantile plots, scatter plots and box plots.

Result – the component provides a presentation of results. Because the way of the discovered knowledge is dependent on the knowledge kind, the component does not contain any specific presentation technique. It employs the functionality provided by a particular data mining component.

3 Supported Mining Algorithms and Types of Knowledge

The most important parts of the FIT-Miner are data mining modules. Each module contains one data mining component. The component includes a data mining algorithm, structures for knowledge representation and support for knowledge presentation. The presentation has to be closely connected with components because different types of knowledge require different presentation methods. A summary of implemented algorithms is shown in Table 1.

Table 1. Summary of supported data mining task and implemented algorithms. The note "ODM" means that the Oracle Data Mining was used for the implementation.

Task	Implemented Algorithms
Association Rules Mining	Apriori (ODM), ML-T2L1
Classification	Naive Bayes (ODM), Decision Tree (ODM), Back-Propagation Neural Network, Genetic Algorithm
Clustering	DBSCAN, OPTICS, Enhanced K-Means (ODM), Orthogonal Clustering (ODM), DENCLUE
Prediction	SVM (ODM), Generalized Linear Models (ODM)
Time Series Analysis	Exponential Smoothing, Winters
Outlier Value Analysis	One-class SVM (ODM)

3.1 Mining Frequent Patterns and Association Rules

Learning association rules is one of the basic machine learning tasks. Nowadays, the system includes two modules for mining frequent patterns and association rules. The first one serves for mining standard one-level association rules, the second one for mining multilevel association rules.

The module for mining one-level association rules is based on the ODM implementation of the Apriori algorithm. The implementation requires a transactional table with two columns as its input. The first column is an identification

of the transaction and the second one contains a set of items of the transaction. This format of the table is not usual in relational databases. Instead, the data is stored in a normalized table with several rows for each transaction. Therefore, an algorithm based on database views that transforms the relation table to the required format of the transactional table was implemented.

Parameters of the mining task are a minimal support, a minimal confidence, a rule length and the maximal number of mined rules. These parameters allow mining a required form of rules to the analyst. If the result set is large, the additional filtering rules can be applied to the rules. The rules are presented in a simple table-view with antecedent - consequent columns.

The ML-T2L1 algorithm is designed for mining multilevel association rules [4]. The multilevel association rules mining requires additional input information. The concept hierarchy of the categorical attributes is needed. It has to be defined by the user by assigning the attribute values into hierarchy tree before the mining phase. The mined rules are shown in the same way as for the one-level association rules module.

3.2 Classification and Prediction Tasks

A classification is other typical data mining task. There exist many different approaches for the data classification but, in general, the algorithms are based on machnine learning techniques. Therefore, each technique (algorithm) is implemented as a single module with one component. The following techniques are included in the current version of the system: Statistical Bayes Theorem, Decision Trees, Neural Networks, Evolutionary Algorithms.

To simplify the classification problem, all classification algorithms require only categorical attributes on the input. Therefore, if the numerical attribute is included in the source data, it has to be transformed – discretized before the classification. The Transformation component can be used for this purpose. Classification components (except Genetic Algorithm) need 2 input datasets because the classification task has typically two phases – a training phase and a testing phase. Both have to be performed on different non-overlapping datasets. The training step changes the classification model and the testing step tests the quality of the classification model on other data with known classes. The splitting of the input data set has to be performed by Reduce/Partition component before.

The ODM was used for the implementation of the Bayesian and Decision Tree classifiers. Hence the Naive Bayes Component and Decision Tree Component has very similar interface. The components are designed for a simple model creation and transparent presentation of the classification results. Because the system is used by students, the components also show statistics about the model creation and its parameters. Namely, a ROC curve and lift charts over the testing dataset are accessible by the Result component. For the visualization of model accuracy, the components can show the confusion matrix by 3D plot where x- and y-axes determines a pair of actual (known) and predicted classes, and z-axis is quantity of the pair in the matrix. Non-diagonal bars show errors on the accuracy chart.

In addition, the Decision Tree classification component can visualize the final decision tree model by the tree structure of decision conditions.

A neural network and a genetic algorithm are not supported by ODM. The neural network uses the most known Back-propagation algorithm. The implementation of the Genetic Algorithm is very rare in data mining systems. We have included this algorithm rather for experimental purposes. The major problems of the genetic algorithm are the finding of a suitable encoding of input data and the definition of the appropriate fitness function. The data encoding in the implemented genetic algorithm is based on the fitness function which compares the sizes of correctly and wrongly classified data [2]. The advantage of this method is that it can be simply transformed to the set of SQL queries and the dataset needn't be transferred to the client-side of the system. The result is presented in the form of a set of IF-THEN rules because a gene encodes an condition and the class of a data entity.

When the target is not a class (categorical) attribute but numeric attribute, the prediction component should be used. The FIT-Miner supports prediction by the Support Vector Machines (SVM) method and by the Generalized Linear Models implemented in ODM. Both algorithms are covered by a single prediction component and the analyst has to select one of them as a prediction method. The main input parameter is the target attribute to be predicted. Quality of the model is tested while training and it is shown by the Result component as e.g. Mean Squared Error and Mean Absolute Error measures or by comparisons of real and predicted values.

In contrast to association rules where the descriptive model is the final result, classification and prediction models are usually created in order to classify new non-classified data. For this purpose, the components allow applying the mined model to new imported data and allow performing a classification/prediction task over the imported data. The result is shown in a table with the target attribut filled by the classification algorithm.

3.3 Clustering Algorithms

There exist various clustering algorithms with different input parameters and also their output knowledge may have different form. Therefore, we designed a common interface, which every clustering algorithm has to implement. The component also provides four distance measures, three of them for interval variables and one for both nominal and interval variables. Based on DMSL type of the column, nominal or interval variable is used.

The resulting clustering model can be either flat or hierarchical, when hierarchical clustering algorithm is used. We have used three types of the visualisation of the model. The first type contains summarizing information about the model, tree representation and comparison of the clusters. The second one is designed for nominal variables to show their occurances in clusters. The third one is designed for interval variables. It contains 2D and 3D scatter plots. The clustering is the data mining task that can be used also in data preprocessing steps. Therefore, it is not a pure data mining component. Its output can be connected not

only to report component but also to another mining component. In the following mining component user can use the knowledge, to which cluster each data sample belongs.

Five clustering algorithms based on the common interface have been implemented and integrated into the system. Two of them are algorithms implemented in ODM and run on the server-side of the FIT-Miner, namely Enhanced K-Means, which is hierarchical extension to a well known K-Means, and Orthogonal Clustering (O-Cluster), which is a recursive hierarchical algorithm. The other three algorithms are density based ones, namely DBSCAN [1], OPTICS [3] and DENCLUE [7]. They have been implemented on the client-side[8]. The result of the OPTICS algorithm has additional knowledge. This knowledge is represented by an ordered structure which helps the user to find clusters in data of varying density. FIT-Miner provides the visualisation of this structure to the user too.

There is another component related to clustering available in the FIT-Miner. It allows the user to compare the quality of results from two or more clustering modules. The comparsion is based on computing of various validation indexes. Currently, the module provides two validation indexes - the Dunn's index and the ratio of averages. The example of such a comparsion is depicted in Figure 3. All clustering algorithms included in the FIT-Miner are compared on the Iris dataset in the diagram.

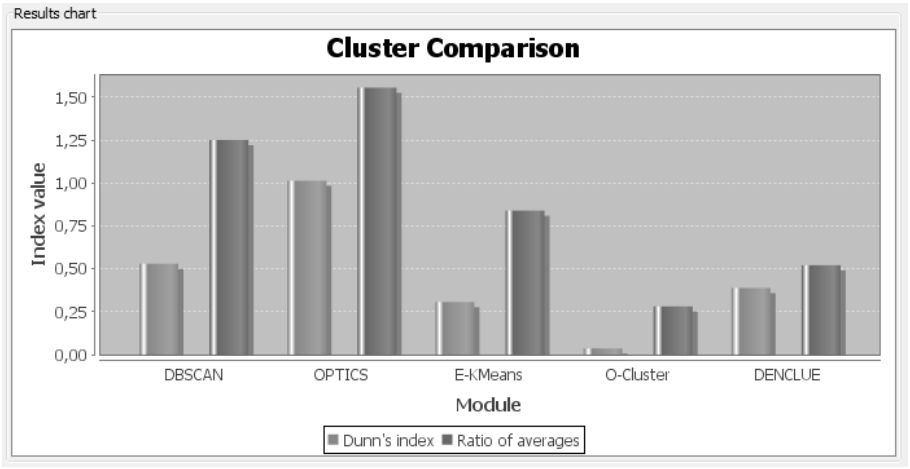

Fig. 3. Comparsion of all currently implemented clustering algorithms on Iris data set.

3.4 Mining Outlier Values

Sometimes, it is not interresting to search for patterns which are frequent, but for those which occure rarely. We call them anomalies or outliers. This task

can be considered as a one-class classification. We have information only about one type of the data - one class, everything else is considered to be an outlier value. In FIT-Miner system, we have used an One-class SVM algorithm which is implemented in ODM on the server-side. The algorithm works in two phases – a model creation and a model application. In the first phase, classification model is created for given input data which do not contain outliers, but only relevant data. Then the model is applied on new data which classifies into two classes - relevant data and outliers.

3.5 Mining Time Series

The methods and mining components described in previous sections mine in a relational or transactional table, which is the most common approach in data mining systems. The FIT-Miner also supports mining from time series, which is not so common.

There are several types of tasks for time series analysis and data mining, e.g. similarity based searching, trend analysis, segmentation, prediction (usually called *forecasting*). Currently, only forecasting is implemented as one of FIT-Miner components.

Two forecasting methods have been implemented. Both methods are based on exponential smoothing. First one is simple exponential smoothing and the second one is Winters method, which belongs to one of the best methods from the relative error of methods point of view[18].

Because the simple exponential smoothing does not deal with seasonality, we have added this functionality as one of preprocessing operation. It is based on a moving average method.

The results of our implementation have confirmed that the Winters method gives much better results than the simple exponational smoothing. The relative error on syntetic data was around 5% using Winters method. When dealing with real world data, results were worse due to the occurence of many random values in the dataset. The average of the relative error was 25% but it was still better than the error of the simple exponational smoothing.

4 Conclusions and Future Plans

The FIT-Miner already provides enough functionality to be employed in our course Knowledge discovery in databases. Currently, we use the SAS Enterprise Miner, where the approach of defining the data mining task specification is similar to our one. First testing of the FIT-Miner in the course started last academic year and will continue this year. Both the FIT-Miner and SAS Enterprise Miner are used and students can choose one of them.

We do not consider the development of the FIT-Miner data mining system to be completed. Its objective is not only to provide an appropriate tool that can be used in education, but, maybe more importantly, to serve as an experimental data mining development platform for our students and reseachers. For future

we plan extensions of the Kernel and Pluggable Components. We also plan to extend the system by some experimental algorithms for unusual types of data such as spatio-temporal data or data streams.

Acknowledgement

This work was partially supported by the BUT FIT grant FIT-S-10-2 and the research plan MSM0021630528.

References

1. Ankerst M.; Breunig M.; Kriegel H.P., Sander J. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'99)*, Philadelphia, 1999.
2. Cheonni S. Design and Implementation of a Genetic-Based Algorithm for Data Mining. *26th VLDB Conference*. Egypt, 2000.
3. Ester M.; Kriegel H.P.; Sander J., Xu X. OPTICS: Ordering Points To Identify the Clustering Structure. *2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96)*. Portland, 1996.
4. Han, J.; Fu, Y. Discovery of Multiple-Level Association Rules from Large Databases. *21th international Conference on Very Large Data Bases*. San Francisco, 1995.
5. Han, J.; Kamber, M. *Data Mining: Concepts and Techniques*. Elsevier Inc., 2006. ISBN 978-1-55860-901-3.
6. Hájek, P.; Havránek, T. *Mechanizing Hypothesis Formation Mathematical Foundations For a General Theory*. Springer-Verlag, 1978. ISBN 3-540-08738-9.
7. Hinnenburg A.; Keim D.A. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. Knowledge Discovery and Data Mining 1998.
8. Hlosta, M. *Cluster Analysis Module of a Data Mining System*. FIT BUT, Brno, 2010.
9. Kotásek, P. *DMSL: The Data Mining Specification Language*. FIT BUT, Brno, 2003.
10. Kotásek P.; Zendulka J. Describing the Data Mining Process with DMSL. *Advances in Database and Information Systems. Vol. 2: Research Communications.*, Bratislava, 2002, pp. 131-140, ISBN 80-227-1744-4.
11. Krásný, M. *Data Mining System in Oracle*. FIT BUT, Brno, 2008.
12. LISp-Miner. URL http://lispminer.vse.cz/
13. Oracle. Oracle Data Mining Application Developer's Guide. URL http://download.oracle.com/docs/cd/B28359_01/datamine.111/b28131/toc.htm.
14. Rapid-I. RapidMiner. URL http://rapid-i.com/content/view/181/190/lang,en/
15. SAS Institute Inc. Data mining with SAS Enterprise Miner. URL http://www.sas.com/technologies/analytics/datamining/miner/
16. Šebek, M. *Functionality Extension of Data Mining System on NetBeans Platform*. FIT BUT, Brno, 2009.
17. University of Waikato. Weka Machine Learning Project. URL http://www.cs.waikato.ac.nz/ ml/index.html
18. Wheelwright S.; Makridakis S. *Forecasting methods for managment*. Harvard, 1973.