A Methodology for Providing User Support for Developing Knowledge Discovery Applications

Robert Engels[1], Guido Lindner[2] and Rudi Studer[1]

[1] Universität Karlsruhe, Institut AIFB, D-76128, Karlsruhe,
Email: {engels,studer}@aifb.uni-karlsruhe.de

[2] Daimler Benz AG, Forschung und Technik, F3S/E
p.o. Mercedes Benz AG, T-402 D-70322 Stuttgart
Email: lindner@str.daimler-benz.com

Abstract

Knowledge Discovery in Databases (KDD) currently receives much attention from both the research as well as the industrial world. This might be due to the fact that companies are often faced with continuously growing databases that become increasingly important for decision making, whereas traditional data analysis approaches might not be able to handle all the requirements of decision makers. In such cases, KDD can offer an additional way for looking at data and extracting interesting knowledge in addition to more traditional approaches as e.g. statistics. Unfortunately, experiences with the application of Data Mining techniques show us that application of such techniques requires (thorough) expertise, and is often not so straightforward as hoped. Therefore, providing User Support for KDD processes is seen as necessary. The project described here is performed at the institute AIFB since early 1995[1]. An architecture for a User Guidance Module (UGM) is developed that aims at providing user support for KDD. The methodology that forms the basis for this architecture will be shortly described in this project report, together with some aspects of the prototype that is implemented in order to provide a proof of concept.

1 Introduction

Application of less traditional data analysis techniques gets more and more popular to overcome drawbacks of traditional data analysis techniques like regression analysis, Principle Component Analysis, Discriminant Analysis and so

[1]Papers concerning this project are available from the AIFB WWW-server:
http://www.aifb.uni-karlsruhe.de/WBS/publications/

on. In our project we aim at providing support for development of KDD applications. This project description will shortly describe our approach to offer such a support, the concepts and architecture that forms the basis, the means that are used for the implementation of a prototype as well as some future work.

2 User Support for KDD

Although the central role in a KDD process is played by Data Mining techniques that perform some analysis on the data, many applications however show that the key success factor of KDD is a proper problem definition and application development process (i.e. proper definition of the steps chosen for pre-processing etc.). Analysis of a KDD problem as well as analysis and preprocessing of the data that is to be used seems to consume about 80 percent of the time that is needed for the development and performance of a KDD process. At the same time, both definition of the problem and data pre-processing is often performed iteratively at a loose basis, following informal (re-)definitions between project partners resulting in ad-hoc data transformations (which are sometimes dominated by pragmatic decisions based on available hard- and software) and typically such steps are not documented.

In order to overcome such shortcomings, and at the same time deliver some support, the definition of a methodology for application of learning techniques in general is therefore an obligatory part of our work. Work that is performed towards definition of such a methodology can be found in [EEHV97], [Eng96], [ELS97] and [VE97]. User Guidance and Support aims at a decreased development time as well as better results of executed KDD processes. Next to a description of the context in which a KDD process is to be executed (characteristics of the data to be used, requirements on the solution, etc.), our approach also uses previous experiences where new applications where developed which might be reused for new KDD processes. Support is provided for selection of appropriate algorithms and algorithm settings, based on objective characteristics of the data that are used. The methodology that we developed also helps in building an initial, executable KDD process prototype that can be used and altered for further development and analysis.

3 Knowledge Discovery in Databases Processes

Typically we will call the application of Data Mining algorithms in a realistic environment (larger, real-world datasets, realistic problem definitions, etc) a KDD process (Knowledge Discovery in Databases)[2]. Such applications normally cycle through the following stages:

[2] We will use both terms interchangeably, although we are aware of the potential for discussion.

- **Data analysis** This stage is performed in order to get a more thorough insight in the domain data and their quality.
- **Data preprocessing** Data transformations often have to be performed since either data is not stored in a convenient format or contains either too much, too less or irrelevant information, a problem that often can be solved through execution of preprocessing techniques. Data transformation mainly is performed in two ways: horizontally (where one changes the dimensionality of the data) or vertical (changing number of data items).
- **Data Mining** Applying the techniques selected to perform data analysis also means that one has to set parameters, check if the techniques assumptions are fulfilled (assumptions like whether a certain technique requires multi-variate normal distribution of its dependent variables, posses restrictions on an eventual independent variable, whether a minimum of data tuples is present w.r.t. the significance of resulting models, etc.).
- **Evaluation** The produced models/knowledge has to be interpreted, evaluated and eventually iteration is performed, or the results are forwarded for employment (typically, once evaluated as good enough, one wants to perform a certain task with the results).
- **Storage and documentation** After succesfull execution of a KDD process the process that was used for producing the results, as well as the results and their evaluation should be documented and stored with possible reuse in mind.

Our methodology for user support follows such typical KDD process stages in so far that a user is supported according to these phases.

4 The UGM Approach to User Support: a Two Way Process

Our approach can be seen as containing a top down as well as a bottom up strategy for user support. The combination of these two is a major advantage, since it enables us to propose and extract algorithms, processes and process parts and at the same time find sensible parameter settings, based on the context (user provided as well as gathered from calculation of certain statistics). The next two sections will shortly describe these two strategies and their connections.

4.1 From Problem Descriptions to Task Decomposition

A typical KDD-process will start with the definition of the actual problem, where the goals of the application (functional as well as non-functional) are defined together with the context in which the problem will be solved.

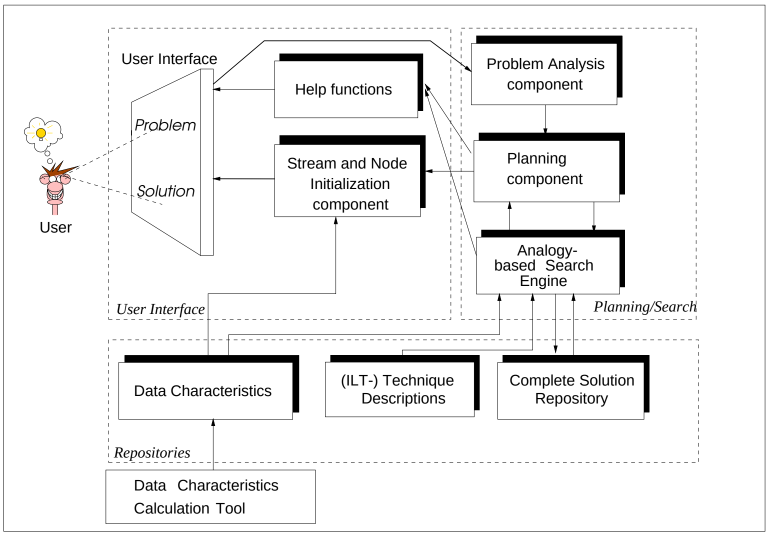

Figure 1: Architecture of the User Guidance Module

We guide KDD process definition by letting the user define his or her (KDD) problem, using a *Problem Analysis Component*. This component intelligently lets the user answer questions that are relevant to the problem at hand. So, on the one hand, according to an increasingly detailed problem description, the requirements on the solution are defined, where on the other hand this component will provide feedback over certain decisions and can make proposals for preprocessing or even the applicability of certain algorithms given the data. The *Problem Analysis Component* is for this purpose supported by a tool that retrieves Data Characteristics of the domain data that is to be used. An initial task decomposition results of this component, which can be used for later back tracking to specific decision points in the process of problem description. In this way the user has the possibility to redefine parts of his problem description and evaluate the altered KDD process after execution.

The PA-component then passes this information to the *Planning Component* that tries to identify process components from a repository via the *Analogy Based Search Engine* or otherwise plans a sequence of steps that can deliver the required functionality. Elements in this repository are divided in three groups, and can be either generic (i.e. sole algorithms or techniques), compositions of chains of such techniques with a certain (described) functionality, or whole processes that are indexed and stored, and that posses the functionality needed to solve the problem at hand. Such components can either stem from former successfully applied KDD solutions, or from the repository of components that describe solutions for parts of the task decomposition and are predefined (application independent). Such components will be called Reusable Process Units (RPU's).

Describing components is done using a Description Logical approach, which enables the usage of a matching algorithm and provides us with a means for defining "goodness-of-fit". The metrics used for retrieving components are an extension of the ideas of Tversky ([Tve77]). Basically a ratio is defined between a (weighted) set of features that describe the similarity of the currently considered RPU w.r.t. the problem definition, and the (weighted) set of features that describe the distinction between the currently considered RPU and the problem definition as in (1).

$$Utility = \frac{\sum_{i=1}^{m} w_i SimFeature_i}{\sum_{j=1}^{n} v_j DistFeature_j} \tag{1}$$

A critical factor here is the definition of relevant features for the KDD environment that we deal with. Figure 2 gives an idea of features that are used by our UGM in order to calculate the Utility of a component. Note that this list is just a summary. As one can see, some of these features are abstractions of the contents of a component based on its functionality. Such abstractions are made at storage time by the component editor of the UGM. For example: A component contains a technique that derives a new attribute from several existing attributes (e.g. by means of a mathematical transformation). If the next technique that is executed cuts out the original attributes, a transformation of the dataset is performed, and both techniques together will get the value: 'attribute transformation' for the functionality slot. Input- and output constraints are used to finally provide a match between the selected component and the KDD process where the component should fit in.

component_type:	[name; model]
KDDprocess_phase:	['Data Analysis'; 'Data Preprocessing'; Data Mining; ....]
component_function:	['Attribute Agglomeration'; 'Attribute Definition'; 'Data Sampling'; 'Attribute Value Agglomeration'; 'Attribute Transformation'; ....]
component_contents:	[m ['number_of_nodes' number] ['list_of_nodes' 'list_of_nodes'] ['node_necessity' 'list_of_necessities']
input_constraints::	['logical expressions describing input constraints']
output_constraints::	['logical expressions describing restrictions on output']
component_annotations:	['comment text meant for end-users']
...............:	....

Figure 2: Descriptors of KDD process parts

Second, a distinction in our framework is made between necessary and optional features (that are also included in the annotation tool of the UGM) and is implemented through an appropriate weighting of the features, where necessary features are assigned larger weights than optional ones. Rationale behind that is that a component might have a certain functionally at storage time that is more extensive as strictly necessary for its proper functioning. One can think of a certain procedure for preprocessing of data that is extended with several visualisation techniques for the convenience of the user that defined the preprocessing process. These visualisation techniques are therefore present in the stored component, but not strictly necessary for performing the task the component is defined for. Currently we consider optimisations of this approach in order to depend less on an expert appropriately weighting the feature set on the hand, and on the other hand get a better estimation of the goodness of fit, together with the degree of non-agreement that the matcher finds among the RPU- and problem description. In our framework this means that a semantical net is used connecting the several features using weighted arcs. The weight of a path between dissimilar features then defines a measure for disagreement which is then used, in combination with (weighted) similar features to propose a number of applicable components from the repository to the user.

As far as components are found in the repository they are proposed to the user, in case of processes (either whole or parts) the user gets a graphical representation of the components to select from combined with some basic descriptions (like author of the RPU, creation data and comments in Natural Language) of the contents of the components. Him or her will then be given the possibility to check out annotations associated with the several components and decide on the most appropriate one. If none is found very appropriate (which can be the case if the repository can only deliver ill-fitting components) the user might pass the job to a planning system that tries to define a suitable process based on a knowledge base. The underlying methodology for process definition is based on the concept of task decomposition, which means that the user is guided through a recursive refinement of his problem. Based on a set of components (described by their pre- and postconditions and stored in the above mentioned repository) the planner can also be applied to propose a next-best technique to apply. Apart from the advantages of such a top down approach, such as a step by step refinement and the possibility to look at the problem from several levels of refinement, this architecture also enables the storage of parts of the final decomposition. On its turn, this architecture supports our concept of reuse of parts of KDD processes.

4.2 Providing more Objective Data Set descriptions

Another functionality of our User Guidance Module is the ability to use its knowledge base to analyse the problem description (defined by the user using the

Problem Analysis Component) in relation to the characteristics of the available data to: a) propose meaningful algorithms to a user for performing certain tasks of a task decomposition, and b) set the algorithms' parameters according to the context it has to be executed in.

Characteristics of the available data are retrieved using a tool that analyses the data source(s). It returns specific characteristics of the data that are calculated using techniques from statistics as well as information theoretical measures. Although the usage of data characteristics is known from projects like StatLOG and MLT/Consultant we went further down the road by selecting a quit different set of statistical techniques (for continuous attributes) and information theoretical techniques (for symbolic attributes), that are relatively robust against certain types of distributions of attributes, and are in that sense a better choice as simple distribution descriptors like e.g. skewness, kurtosis etc.. Besides techniques that are more robust in real-world settings we also use different techniques in order to be able to get a more thorough insight in the complexity of the domain and the possibilities for preprocessing of the data. An example is the calculation of so-called eigenvalues, which are used for an estimation of the possible dimensionality of classification problems. Here we also use a different technique that is also suitable for real-world distributed datasets. The techniques are implemented in a so called *Data Characteristics Calculation Tool* that is included in the UGM. In the UGM the part that analyses such measurements resembles a small expert system, that finally directs the interaction with the user and proposes preprocessing steps in accordance to these data. At the same time these measures can play a role in the selection of algorithms that are applicable (with use of a knowledge base).

Summarised, the UGM provides guidance and support on several levels for users that are no experts in performing KDD processes as well as support for relative experts who are looking for support while selecting applicable algorithms for a certain application while defining their KDD process mainly self. So, User Guidance Modelling aims at:

1. **Supporting** the description of the actual KDD-problem
2. **Matching** of a task on the problem definition of the current KDD problem (and **decomposition** of this task).
3. **Proposing** initial solutions that a user can either *execute* or take as basis for *further refinement* as well as algorithms that are selectable for the problem at hand.
4. **Annotating** solutions and solution parts for understandability and retrieval purposes (this is an additional feature of our UGM implementation that is not described in this paper).
5. **Managing** components of the solution for reuse.

From a methodological point of view seen such a component is relatively independent of a software tool, but in the course of the CITRUS-project ([WSG+97]) the UGM implementation is based on the Clementine system[3].
In the next session we will deal with our implementation of this architecture.

5 Implementation of the prototypical UGM

The UGM as prototype is implemented in pop11 ([BS87]) on a Sparc10 running Solaris. Clementine is a trademark by Integral Solutions LTD. (ISL) [She96] and is used as a basis for our UGM. Clementine comprises several data source interfaces, techniques for data preprocessing (visualisation, selection, transformation, agglomeration, sampling, etc.) as well as a variety of machine learning algorithms that all can cope with data tuples represented as a conjunction of attribute-value pairs. As machine learning and data analysis algorithms neural nets, C4.5, a rule learner, a Kohonen net and a regression analysis algorithm are included. In the course of the Daimler Benz project several algorithms are added concerning preprocessing and data analysis. Clementine's functionality provides us with the possibility to restrict our focus to the support of KDD process definitions while the algorithms used for KDD are implemented and easily executed. Furthermore Clementine provides an attractive interface that we extended for our purposes. The DataSet Analyser is implemented in C and is executable from our UGM tool.

6 Related Work

Although there seems a kind of general consensus on the necessity of User Guidance for KDD process definition problems of the kind described here (i.e. [FV97], [ELS97], [ZLKO97]), there is not too much research found in this direction. In the field of statistics selecting appropriate statistical techniques for a given problem plays an important role [Han94]. A similar need for user guidance raises in the field of Knowledge Discovery in Databases where more and more algorithms become available for performing the different subtasks of KDD-processes. KDD processes can become rather complex and obscure. The need for defining user guidance for KDD-processes shows the growing maturity of the field of KDD and deserves serious attention, since the definition of such user guidance requires the abstraction from the technical details of the different algorithms that are typically used in KDD processes and requires to think about the characteristics of the task at hand as well as the applicability of these algorithms under different circumstances. The UGM (User Guidance Module [Eng96]) is meant to provide such an assistance to a user.

[3] Trademark by INTEGRAL SOLUTIONS LTD. (ISL) [She96].

In statistics the question of how to identify appropriate techniques has been addressed [Han94]. Although the statistical problems that are mentioned are problems that are similar to KDD problems, there is not a real solution offered in neither statistics nor KDD.

Secondly, approaches towards user guidance can be found in the field of Machine Learning ([PSBK+96]). An example here is the MLT-project [Con93] where a so-called CONSULTANT tool has been built [CSG+92]. Here the approach is taken to build a knowledge base for defining which algorithm of a set of ten algorithms is appropriate for a certain problem. The knowledge base was mainly aiming at differentiating between these algorithms, whereas we describe the functionality of algorithms, thereby facilitating the integration of new algorithms. Further, Consultant could not deal with situations where several algorithms should be combined in order to solve a problem. From the KDD point of view there is also a need for defining a method for performing KDD. A few approaches in this direction can be found in ([BS95], [RW95]).

Third we support the generation of more complex processes, involving multiple algorithms, and also the reuse of formerly defined process parts. Reuse and task decomposition is known from the field of Knowledge Acquisition, and is found in approaches such as KADS [SWB93] and MIKE [AFS96]. Those ideas combine very well with our aim to provide a user with a possibility to reuse parts of previous problem solutions and also provide a means for storage of newly defined KDD-processes.

Work on defining similarity is known from the field of cognitive psychology (a.o. [Tve77]) and also plays an important role in the Case Based Reasoning research community. For KDD, [NS97] proposes an approach to the evaluation of applied algorithms, which is based on ideas of the Operation Research Community, and shows resemblances to the approaches known from cognitive psychology. The difference between our usage of the metric and [NS97]'s usage of the metrics lies for example in the fact that we aim at defining a similarity measure for selecting an artifact a priori, where they provide a metric for a posterior analysis of the applicability of an algorithm. Using an a posteriori approach to redefinition of quality criteria means that for every context one has to perform tests on applicability, after which one has to select the appropriate calculation for every setting encountered.

7 Conclusions and Future Work

Concluding we can say that the jungle of algorithms for data analysis and knowledge discovery, that all have a certain complexity and corresponding functionality, together with the ill-defined nature of most KDD-problems at the moment they are posed, there is a need to gear them to each other. Our approach aims at doing precisely that by providing a framework for deriving a precise problem description, formalising the problem description, decomposing the matching task and managing the storage and retrieval of formerly successfully applied data analysis and knowledge discovery processes. One important point thereby is the possibility for a user to reuse components that are defined previously. Another important point is the usage of statistical techniques together with information theoretical measurements by a small expert system in order to support preprocessing of the data and technique parameter instantiation.

Future work will include the further development of the ideas mentioned in this paper. The tool that calculates data characteristics for use by our UGM is tested and will eventually be extended if needs arise. On the other hand are we currently analysing implications of the calculated measurements for the other phases of KDD (data analysis, evaluation of learning results, etc.). As far as the task decomposition is concerned we will concentrate on the selection of RPU's for KDD, and if none is available planning alternative processes that can be used to solve a (sub-)process.

Acknowledgements This work is partly supported by Daimler Benz AG ('Research and Development'). We would like to thank the numerous people that added to this work during extensive discussions or by providing helpful feedback. Special thanx to the students that helped implementing parts of the several tools that the UGM comprises.

References

[AFS96] J. Angele, D. Fensel, and R. Studer. Domain and Task Modelling in MIKE. In A.G. Sutcliffe, F. van Assche, and D. Benyon, editors, *Domain Knowledge for Interactive System Design, Proceedings of the IFIP WG8.1/13.2 Joint Working Conference on Domain Knowledge for Interactive System Design.* Chapman & Hall, Geneva, May 1996.

[BS87] M. Burton and N. Shadbolt. *Pop-11 programming for artificial intelligence.* Addison-Wesley, 1987.

[BS95] C.E. Brodley and P. Smyth. Applying Classification Algorithms in Practice. In D. Aha, editor, *Proceedings of the Workshop on Applying Machine Learning in Practice at the ICML-95*, Tahoe City, CA, 1995.

[Con93] MLT Consortium. Final public report. Technical report, 1993. Esprit II Project 2154.

[CSG+92] S. Craw, D. Sleeman, N. Granger, M. Rissakis, and S. Sharma. CONSULTANT: Providing Advice for the Machine Learning Toolbox. In M.A. Bramer and R.W. Milne, editors, *Research and Development in Expert Systems*, pages 5–23, 1992.

[EEHV97] R. Engels, B. Evans, J. Herrmann, and F. Verdenius, editors. *Workshop on Machine Learning Application in the real world; Methodological Aspects and Implications (at the ICML-97)*, Nashville, TN, July 12th, 1997.

[ELS97] R. Engels, G. Lindner, and R. Studer. A Guided Tour through the Data Mining Jungle. In D.Pregibon D. Heckerman, H.Manilla, editor, *Proceedings of the 3rd International Conference on Knowledge Discovery in Databases*, Newport Beach, CA, 1997. AAAI Press, Menlo Park, CA.

[Eng96] R. Engels. Planning Tasks for Knowledge Discovery in Databases; Performing Task-oriented User-Guidance. In E. Simounis, J. Han, and U. Fayyad, editors, *Proceedings of the 2nd Int. Conference on Knowledge Discovery in Databases*, pages 170–175, Portland, Oregon, 1996. AAAI-Press.

[FV97] B. Evans J. Herrmann F. Verdenius, R. Engels. Workshop Summary Report: Workshop on Machine Learning Application in the real world; Methodological Aspects and Implications. Technical report, In conjunction with the: International Conference on Machine Learning-'97, Vanderbilt University, Nashville, TN, http://www.aifb.uni-karlsruhe.de/WBS/ICML97/wssummary.html, 1997.

[Han94] D.J. Hand. Deconstructing Statistical Questions. *Journal of the Royal Statistical Society*, pages 317–356, 1994.

[NS97] Ch. Nakhaeizadeh and A. Schnabl. Development of Multi-Criteria Metrics for Evaluation of Data Mining Algorithms. In D. Heckerman, H. Manilla, and D. Pregibon, editors, *Proceedings of the 3rd International Conference on Knowledge Discovery in Databases*, Newport Beach, CA, August 14 - 17th 1997. AAAI Press, Menlo Park, CA.

[PSBK+96] G. Piatetsky-Shapiro, R. Brachman, T. Khabaza, W. Kloesgen, and E. Simoudis. An Overview of Issues in Developing Industrial Data Mining and Knowledge Discovery Applications. In E. Simoudis, J. Han, and U. Fayyad, editors, *Proceedings of the 2nd Int. Conference on Knowledge Discovery in Databases*, pages 89–95, Portland, Oregon, 1996. AAAI press.

[RW95] T.P. Reinartz and R. Wirth. Towards a Task Model for KDD. In Y. Kodratoff, G. Nakhaeizadeh, and C. Taylor, editors, *Proceedings of the Workshop on Statistics, Machine Learning, and Knowledge Discovery in Databases. MLNet Familiarisation Workshop*, pages 19–24, Heraklion, Crete, April 28-29, 1995.

[She96] C. Shearer. User Driven Data Mining. In *Unicom Data Mining Conference*, London, 1996.

[SWB93] A.T. Schreiber, B.J. Wielinga, and J.A. Breuker. *KADS: A Principled Approach to Knowledge-Based System Development.* Academic Press, London, 1993.

[Tve77] A. Tversky. Features of Similarity. *Pschygological Review*, 84:327–352, 1977.

[VE97] F. Verdenius and R. Engels. A Process Model for Developing Inductive Applications. In W. Daelemans and P. Flach, editors, *Proceedings of the 7th Belgian-Dutch Conference on Machine Learning (Benelearn)*, Tilburg, Netherlands, 1997.

[WSG+97] R. Wirth, C. Shearer, U. Grimmer, Th. Reinartz, J. Schloesser, Chr. Breitner, R. Engels, and G. Lindner. Towards Process-Oriented Tool Support for KDD. In *Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD-97)*, Trondheim, Norway, June 1997.

[ZLKO97] N. Zhong, Chunnian Liu, Y. Kakemoto, and S. Ohsuga. KDD Process Planning. In D. Heckerman, H. Mannila, D. Pregibon, and R. Uthurasamy, editors, *Proceedings of the 3th International Conference on Knowledge Discovery and Data Mining*, pages 291–294, 1997.