A Guided Tour through the Data Mining Jungle*

Robert Engels[1], Guido Lindner[2] and Rudi Studer[1]
[1] University of Karlsruhe, Institute AIFB, D-76128, Karlsruhe,
Email: {engels,studer}@aifb.uni-karlsruhe.de
[2] Daimler Benz AG, Research and Technology, F3S/E
p.o. Mercedes Benz AG, T-402 D-70322 Stuttgart
Email: lindner@str.daimler-benz.com

Abstract

An important success factor for the field of KDD lies in the development and integration of methods for supporting the construction and execution of KDD processes. Crucial aspects in this context are the (incremental) development of a precise problem description, a decomposition of this top level problem description into manageable and compatible subtasks which can be reused, and a selection and combination of adequate algorithms for solving these subtasks. In this paper we describe an approach for supporting the systematic decomposition of a KDD process into subtasks and for selecting appropriate problem-solving methods and algorithms for solving these subtasks. Our approach has been partially integrated into the CLEMENTINE system and has been used to develop a real world application in the area of prediction.

Keywords : *KDD-processes, User-support, Task Decompositions, Knowledge Acquisition*

1 Introduction

Knowledge Discovery in Databases (KDD) is currently a field that is seen as increasingly interesting. We think that an important succes indicator for KDD lies in the development and integration of methods for supporting the construction and execution of KDD processes. Crucial aspects in this context are the (incremental) development of a precise problem description, a decomposition of this top level problem description into manageable and compatible (reusable) subtasks and a selection and combination of adequate algorithms for solving these subtasks (Engels 1996). Therefore, methods and corresponding tool support are required which assist the developer of a KDD application in building up a high quality KDD process with as less effort as possible. Our approach is on the one hand based on the notion of task analysis and reusable problem-solving methods (Breuker & van de Velde 1994). We propose to use pre-/postconditions (i) to characterize (sub)-tasks and methods, (ii) to guide the decomposition process, and (iii) to handle the dependencies between the subtasks and the methods. On the other hand we exploit and integrate techniques to describe the characteristics of the available (database) data to further guide the selection of applicable methods and algorithms. We use statistic measures (see e.g. the result of the Statlog project (Michie, Spiegelhalter, & Taylor 1994)), measures from the field of machine learning, e.g. missing values or noise, as well as information available in the data dictionary, e.g. attribute types or the size of relations. Our approach has been partially integrated into the clementine[1] framework and has been used to develop a real world application in the area of prediction. The rest of the paper is organized as follows: we shortly describe the application problem and the solution we have developed using the clementine tool. In section 3 we introduce our approach for supporting the construction of KDD processes. How we have used this approach for solving the given application problem is described in section 4. Finally, we discuss related work and provide a conclusion.

2 Application Environment

2.1 The KDD-Tool

Our approach is partially integrated in the CITRUS project (Wirth *et al.* 1997). Tool support is built on Clementine. This tool supports many of the steps that are typically found in a KDD-process. Moreover, it facilitates defining a dataflow providing a graphical user interface. The user can select icons which represent data sources, data manipulation algorithms, data

 This work has been partially funded by the Daimler Benz AG, Research & Technology, project no. 096 4 965047 1 E B.

[1] Clementine, trademark by INTEGRAL SOLUTIONS LTD. (ISL).

mining algorithms, graphical and statistical data analysis techniques. However, no support is provided for the selection of the appropriate algorithms for a task. Part of the CITRUS project deals with user-guidance to give support during the different stages of the KDD-process and to interactively construct a KDD-solution for a given problem in the form of such streams (Engels 1996).

2.2 Application Scenario

We use a real world application from the automobile industry (Wirth & Reinartz 1996). We will use this example to illustrate our ideas on providing user guidance for KDD-processes. This example of a multistrategy KDD process was developed for an early warning system, which is called the early indicator approach. The main idea is to find and characterise subpopulations of faulty cars that in an early stage show behaviour of a whole population of cars at a later point in time. Such a subpopulation can then be used for prediction and thus facilitates reacting on quality problems with subparts of the population at an earlier time point.

2.3 The EIC-task Decomposed

User support also requires a task decomposition. The three steps of the EIC approach can be mapped to the KDD-process model.

Basically we decompose the approach in two steps, the first step (preprocessing) forms the main step, where subset selection is performed and the attribute space is changed. This step also comprises a KDD process where a model is generated for a certain point in time, which is then deployed for selecting data. The last step in the preprocessing phase performs a transformation of representation from the fault profile of cars to their configuration data. The second step in the approach forms the top level Data Mining step, that finally delivers the interpretable model that is looked for based on the defined set of attributes.

3 The Approach

This section deals with the approach that we take w.r.t. user guidance for KDD-processes. In this paper we will extend upon the terminology and ideas introduced in (Engels 1996). The following sections will deal with the two-staged process of mapping tasks to algorithm classes as well as the final selection of an algorithm that matches the task and data characteristics.

3.1 Recursiveness in KDD Processes

Decomposition of a task into subtasks is a refinement problem that we solve using a multi-strategy approach.

On the one hand we refine the toplevel task according to a library of so-called PSMs[2] and doing so, reuse predefined methods that solve certain specified tasks. In cases where no reusable components are available we can either look for a single technique solving the subtask or, when the problem is too complex to be solved by only one technique, use a planner in order to find a series of techniques that can solve the subtask. Many KDD-processes show recursiveness (as oposed to iterativeness, where certain subtasks are repeated until a certain criterion is reached) in the sense that a subtask of the task decomposition really is a (smaller) KDD-process on its own. Recursiveness also shows up in our example and lets us reuse the same PSM that describes the KDD-process at the top level as well as at the data preprocessing stage (see (Engels 1996)). A singular PSM can in this case be retrieved from a library and reused twice in different instantiations. The several stages that are defined in such a PSM introduce certain constraints on its subtasks. Such a framework is then used to guide an initial decomposition of the task at hand.

3.2 Assigning Algorithm Classes to Tasks

Decomposing an initial task using PSMs delivers a tree like structure describing the task at several levels of abstraction. Tasks, subtask and PSM's are described using the same concept of pre- and postconditions so that tasks can be mapped on methods.

Given a set of modelling algorithms that can be used in KDD-processes it might occur that only a subset of them is applicable. A selection of an algorithm class must then be made. Such an algorithm class should later be shrinked until a single algorithm is left.

3.3 Algorithm Selection using Data Characteristics

For selecting of an algorithm from a set of potentially useful algorithms we apply data characteristics to pick the algorithm with the highest utility.

Since most large databases come with a data dictionary, it is natural to use this data dictionary to extract characteristic information on the data. Furthermore simple statistic measures like standard deviations, means, possibilities etc. that characterise and describe the data can be calculated. In the Statlog project (Michie, Spiegelhalter, & Taylor 1994) such measures were used for determining the applicability of statistical and data mining algorithms. We aim at using similar measurements for selecting algorithms

[2]Problem Solving Methods. See also: (Breuker & van de Velde 1994), (Angele, Fensel, & Studer 1996) for more on PSM's.

which fit to the identified subtask. Here we want to exploit information on attribute types and possible sets, the percentage of positive and negative examples that the database contains, missing values, consistency of the dataset, etc.

4 How to Juggle the Jungle; the Example in our Approach

The idea of recursiveness and iterativeness as we found provides an example on the assigment of algorithm classes to subtask types. For the following we will presume that there already is a problem description available as well as a mapping to an initial appropriate task type. We will use task characteristics of the example to show how an initial algorithm class is selected.

4.1 Recursiveness in the Example

Our example shows the recursive nature of KDD-tasks. On the top level, the KDD-process is decomposed according to the steps of a standard KDD-process. When decomposing, the need for a preprocessing step arises. Relevant attributes (i.e. the average mileage between repairs), can be calculated from the database. These attributes are used for model generation in the preprocessing phase. The generated model in the first step of the preprocessing stage is then used to extract those examples out of a second data subset (from an earlier point in time) that show the same behaviour on the selected attributes. Lateron the static attributes (like the model, type, engine, accesories, etc.) are merged with the data subset resulting of this extraction.

It is clear that in order to be able to derive such a refinement structure one needs to define the functionality of tasks and use them for selecting the appropriate PSMs and finally the right algorithms. Part of this retrieval is supported by annotations of KDD-processes and algorithms.

4.2 Assigning Algorithm Classes to our Task

From a problem description one can retract pre- and postconditions. For the first round we can decide to take a class of supervised learning algorithms due to the fact that a domain specialist provided input that there are classes known that might be used for learning and that there is a need for an interpretable model. This gives us a set of algorithms that are possibly interesting, namely the set of supervised learning algorithms that generate interpretable models. Now we need to close in on one of them in order to guide the task decomposition process.

4.3 Closing in on a specific Algorithm using Data Characteristics

The dataset that is used in the EIC approach has data-characteristics that we will use in a kind of 'strike-through' approach for eliminating algorithms that would not work for our problem. We could retrieve a PSM for the preprocessing subtask based on the set of requirements and constraints we retrieved in the problem definition. We retrieve data characteristics from the Data Dictionary or using the parameters that are defined in the Statlog project. Doing so we get information on the set of classes that comprises our goal attribute (which is provided by the user). In our case this set was not so large (4 items), otherwise we should have started user interaction in order to decrease the number of classes, or maybe to ignore the concept and start data analysis without it (as in unsupervised learning tasks).

Some of the conditions that whe found in our approach lead to the advice to take a certain learning algorithm (f.e. C4.5) for the Data Mining step, since there are interesting classes that are provided by the user and those classes are not represented as numeric attributes and the results can be represented either in the form of a decision tree, or as a set of production rules (this also starts further user interaction in order to make such decisions). Such a decision for a specific algorithm then specifies the constraints (postconditions) for the prior step in our decomposition, together with the constraints that follow from data characteristics and the user input.

An additional feature of task decompositions is that we can make use of the context that such a decomposition represents. It provides us possibilities to set parameters of (selected) algorithms according to their context.

5 Related Work

Approaches that include the user as the key-factor for succesfully performing data analysis problems are found in the field of machine learning ((Craw *et al.* 1992)), and the field of statistics (Hand 1994). Although this literature outlines possibilities or needs for a stronger inclusion of the user, there are no applications mentioned where the ideas are tested, as is the case in our approach.

Breaking down a task's complexity by decomposition is an approach that is known from the field of Knowledge Acquisition, and is found in approaches such as (among others) KADS (Wielinga, Schreiber, & Breuker 1992) and MIKE (Angele, Fensel, & Studer 1996)). Those ideas combine very well with our aim to provide user support. From the KDD point of view there is also a need for defining a method for performing KDD. The steps that are defined in such methodologies form the basics for our task decompositions and thus enable a first step decomposition, as shown in this paper. One particular survey dealt with the question if and how companies apply inductive learning techniques (Verdenius 1997). This research shares the conclusion with many other papers like (Brodley & Smyth 1995) that the process of machine learning application should primarily be user-driven, instead of data- or technology driven. Two approaches exist that deal with the support of selecting data mining algorithms for a certain task. One approach (Brazdil, Gama, & Henery 1994) is based on learning of a decision tree for the applicability of algorithms given data characteristics. The other approach is the more user centered Consultant part of the MLT-project (Consortium 1993).

6 Conclusions and Future work

The approach taken in this paper shows strengths and some weaknesses. The main problem is that we must deal with a huge unknown factor, and that is the user himself. A user might be unaware of some important problem characteristics. Although providing feedback to the user in cases is aimed at, it can happen that the system chooses the wrong track due to its ignorance in those areas where it relies on user provided information. Given this uncertainty we propose to extract some of the most important characteristics of such a user's problem, get hold of the characteristics in the data that he or she provides, and make use of a library containing reusable task decomposition parts that have been defined priorly. Supporting task decomposition means that a user gets a better overview of what task he/she should perform, and also enables us save parts of such decompositions for later reuse. In the future we will extend our framework and incorporate a planning mechanism that can fill gaps in cases that there are no PSM's available from the (user defined) library. Furthermore we will have to deal with the problem of finding a uniform representation for data mining algorithms and corresponding task characteristics. Finally, a knowledge base is needed that can help instantiate algorithms once their context is defined, in order to reduce the amount of effort that is needed to find a successful approach as much as possible.

Acknowledgements
We want to thank Daimler Benz for granting and providing the possibility to get industrial input in the form of real world problems. Furthermore we thank our collegues for many fruitful discussions.

References

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

Brazdil, P.; Gama, J.; and Henery, B. 1994. Characterizing the Applicability of Classification Algorithms Using Meta-Level Learning. In Bergadano, F., and Raedt, L. D., eds., *Proceedings of the European Conerence on Machine Learning (ECML 94)*, volume 784 of *Lecture Notes in Computer Science*, 83 – 102. Catania, Italy: Springer-Verlag.

Breuker, J., and van de Velde, W. 1994. *CommonKADS Library for Expertise Modelling.* IOS Press.

Brodley, C., and Smyth, P. 1995. Applying Classification Algorithms in Practice. In Aha, D., ed., *Proceedings of the Workshop on Applying Machine Learning in Practice at the ICML-95.*

Consortium, M. 1993. Final public report. Technical report. Esprit II Project 2154.

Craw, S.; Sleeman, D.; Granger, N.; Rissakis, M.; and Sharma, S. 1992. CONSULTANT: Providing Advice for the Machine Learning Toolbox. In Bramer, M., and Milne, R., eds., *Research and Development in Expert Systems*, 5–23.

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

Fayyad, U. M.; Piatetsky-Shapiro, G.; Smyth, P.; and Uthurasamy, R. 1996. *Advances in Knowledge Discovery and Data Mining.* Cambridge, London: MIT Press.

Hand, D. 1994. Decomposing Statistical Questions. *Journal of the Royal Statistical Society* 317–356.

Michie, D.; Spiegelhalter, D.; and Taylor, C. 1994. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood.

Verdenius, F. 1997. Applications of Inductive Learning Techniques: A Survey in the Netherlands. *AI communications* 10(1).

Wielinga, B.; Schreiber, A.; and Breuker, J. 1992. KADS: A Modelling Approach to Knowledge Engineering. Special Issue "The KADS Approach to Knowledge Engineering". *Knowledge Acquisition* 4(1):5–53.

Wirth, R., and Reinartz, T. 1996. Detecting Early Indicator Cars in an Automotive Database: A Multi-Strategy Approach. In Simounis, E.; Han, J.; and Fayyad, U., eds., *Proceedings of the 2nd Int. Conference on Knowledge Discovery in Databases.* Portland, Oregon: AAAI-press.

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