Intelligent Assistance for the Data Mining Process:

An Ontology-based Approach

Abraham Bernstein
Corresponding Author
Department of Information Systems
Leonard Stern School of Business
New York University
44 West 4th Street, Suite 9-76
New York, NY 10012
Phone: 212 – 998 0803
Fax: 212 – 995 4228
Email: bernstein@stern.nyu.edu

Shawndra Hill
Department of Information Systems
Leonard Stern School of Business
New York University
44 West 4th Street, Suite 9-181
New York, NY 10012
Phone: 212 – 998 0837
Fax: 212 – 995 4228
Email: shill@stern.nyu.edu

Foster Provost
Department of Information Systems
Leonard Stern School of Business
New York University
44 West 4th Street, Suite 9-71
New York, NY 10012
Phone: 212 – 998 0806
Fax: 212 – 995 4228
Email: fprovost@stern.nyu.edu

CeDER Working Paper IS-02-02

Center for Digital Economy Research, Stern School of Business, New York University

44 W. 4th St., New York, NY 10012-1126, USA

Intelligent Assistance for the Data Mining Process: An Ontology-based Approach

Abraham Bernstein, Foster Provost, and Shawndra Hill

Department of Information Systems
Leonard Stern School of Business
New York University

Abstract

A data mining (DM) process involves multiple stages. A simple, but typical, process might include preprocessing data, applying a data-mining algorithm, and postprocessing the mining results. There are many possible choices for each stage, and only some combinations are valid. Because of the large space and non-trivial interactions, both novices and data-mining specialists need assistance in composing and selecting DM processes. We present the concept of Intelligent Discovery Assistants (IDAs), which provide users with (i) systematic enumerations of valid DM processes, in order that important, potentially fruitful options are not overlooked, and (ii) effective rankings of these valid processes by different criteria, to facilitate the choice of DM processes to execute. We use a prototype to show that an IDA can indeed provide useful enumerations and effective rankings. We discuss how an IDA is an important tool for knowledge sharing among a team of data miners. Finally, we illustrate all the claims with a comprehensive demonstration using a more involved process and data from the 1998 KDDCUP competition.

Index Terms

Data mining, data-mining process, intelligent assistants, knowledge discovery

1 Introduction

Knowledge discovery from data is the result of an exploratory process involving the application of various algorithmic procedures for manipulating data, building models from data, and manipulating the models. The Knowledge Discovery (KD) process [Fayyad, Piatetsky-Shapiro & Smyth, 1996] is one of the central notions of the field of Knowledge Discovery and Data mining (KDD). The KD process deserves more attention from the research community; processes comprise multiple algorithmic components, which interact in non-trivial ways. Even data-mining specialists are not familiar with the full range of components, let alone the vast design space of possible processes. Therefore, both novices and data-mining specialists are apt to overlook useful instances of the KD process. We consider tools that will help data miners to navigate the space of KD processes systematically, and more effectively. In particular, this paper focuses on a subset of stages of the KD process—those stages for which there are multiple algorithm components that can apply; we will call this a data mining (DM) process (to distinguish it from the larger knowledge discovery process). For most of this paper, we consider a prototypical DM process template, similar to the one described by Fayyad et al. [1996] and [Chapman et al., 2000], which is shown in Figure 1. We concentrate our work here on three DM-process stages: automated preprocessing of data, application of induction algorithms, and automated post-processing of models. We have chosen this set of steps because, individually, they are relatively well understood—and they can be applied to a wide variety of benchmark data sets.[2] In the final case study, we expand our view to a more involved DM process.

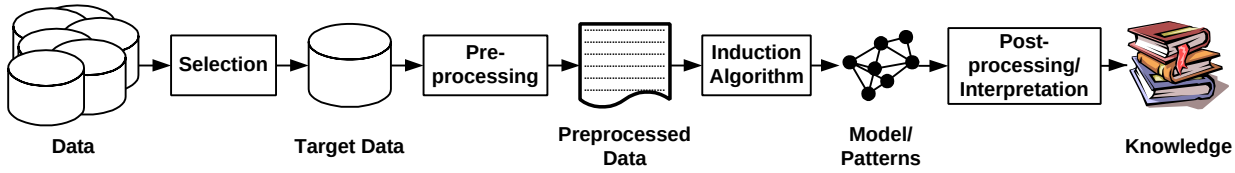

Figure 1: The KD process (adapted from Fayad et al. [1996])

Figure 2 shows three simple, example DM processes.[3] Process 1 comprises simply the application of a decision-tree inducer. Process 2 preprocesses the data by discretizing numeric attributes, and then builds

[2] More generally, because we will assemble these components automatically into complete processes that can be executed by a user, the scope of our investigation is necessarily limited to KD-process stages for which there exist automated components, and for which their requirements and functions can be specified. Important but ill-understood stages such as "business process analysis" or "management of discovered knowledge" are not included [Senator, 2000]. We also do not consider intelligent support for more open-ended, statistical/exploratory data analysis, as has been addressed by St. Amant and Cohen [St. Amant & Cohen, 1998].

[3] Descriptions of all of the techniques can be found in a data mining textbook [Witten & Frank, 2000].

a naïve Bayesian classifier. Process 3 preprocesses the data first by taking a random subsample, then applies discretization, and then builds a naïve Bayesian classifier.

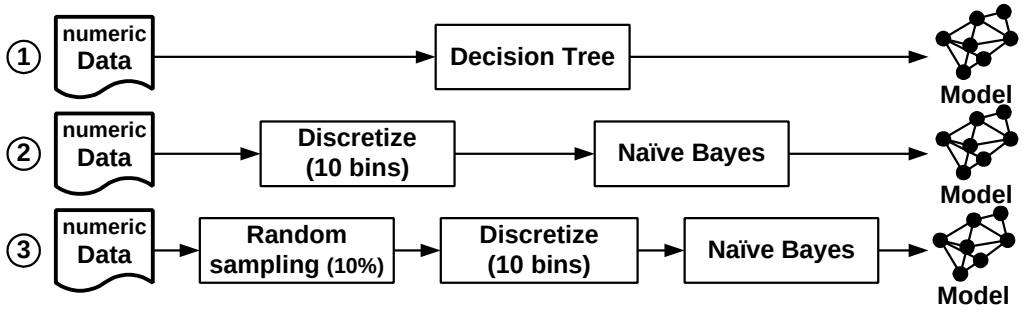

Figure 2: Three valid DM processes

Intelligent Discovery Assistants (IDAs) help data miners with the exploration of the space of valid DM processes. A valid DM process violates no fundamental constraints of its constituent techniques. For example, if the input data set contains numeric attributes, simply applying naïve Bayes is not a valid DM process—because (strictly speaking) naïve Bayes applies only to categorical attributes. However, Process 2 is valid, because it preprocesses the data with a discretization routine, transforming the numeric attributes to categorical ones. IDAs take advantage of an explicit ontology of data-mining techniques, which defines the various techniques and their properties. Using the ontology, an IDA searches the space of valid processes. Applying each search operator corresponds to the inclusion in the DM process of a different data-mining technique; preconditions constrain its applicability and there are effects of applying it. Figure 3 shows some (simplified) ontology entries (cf., Figure 2).

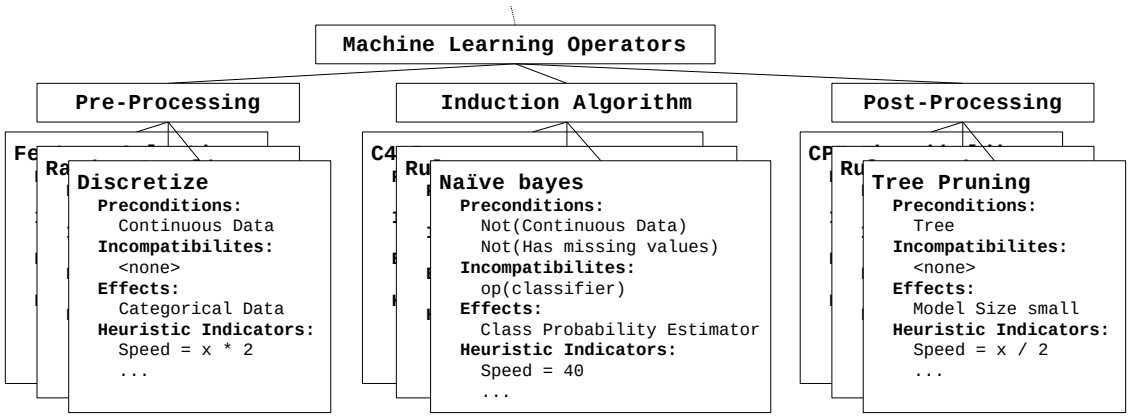

Figure 3: Simplified elements of a DM ontology

Above we said that an IDA *helps* a data miner. More specifically, an IDA determines characteristics of the data and of the desired mining result, and enumerates the DM processes that are valid for producing

the desired result from the given data. Then the IDA assists the user in choosing processes to execute, for example, by ranking the process (heuristically) according to what is important to the user. Results will need to be ranked differently for different users. The ranking shown in Figure 2 (based on the number of techniques that form the plan) would be useful if the user were interested in minimizing fuss. A different user may want to minimize run time, in order to get results quickly. In that case the reverse of the ranking shown in Figure 2 would be appropriate. There are other ranking criteria: accuracy, cost sensitivity, comprehensibility, etc., and many combinations thereof.

In this paper, we claim that IDAs can provide users with three benefits:

1. a systematic enumeration of valid DM processes, so they do not miss important, potentially fruitful options;
2. effective rankings of these valid processes by different criteria, to help them choose between the options;
3. an infrastructure for sharing knowledge about data-mining processes, which leads to what economists call network externalities.

We support the first claim by presenting in detail the design of effective IDAs, including a working prototype, describing how valid plans are enumerated based on an ontology that specifies the characteristics of the various components. We then show plans that the prototype produces, and argue that they would be useful not only to novices, but even to expert data miners. We provide support for the second claim with an experimental study, using ranking heuristics. Although we do not claim to give an in-depth treatment of ranking methods, we demonstrate the ability of the IDA prototype to rank potential processes by speed and by accuracy (both of which can be assessed objectively). We also demonstrate that an IDA can perform along the tradeoff spectrum between speed and accuracy. Finally, we provide additional support for all the claims with an empirical demonstration, using the KDDCUP 1998 data-mining problem, showing how an IDA can take advantage of knowledge about a problem-specific DM process, and we discuss how the insertion of such knowledge could improve the performance of a data-mining team. For most of the paper we use simple processes, such as those presented in Figure 2, to provide support for our claims. The final demonstration goes into more depth (but less breadth) with a particular, more complex process.

2 Motivation and General Procedure

It has been argued that when engaged in design activities, people rarely explore the entire design space [Ulrich and Eppinger, 1995, p. 79]. There is evidence that when confronted with a new problem, data miners, even data-mining experts, do not explore the design space of DM processes thoroughly. For example, the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining holds

an annual competition, in which a never-before-seen data set is released to the community, and teams of researchers and practitioners compete to discover the "best" knowledge (evaluated differently each year). KDDCUP-2000 received 30 entrants (teams) attempting to mine knowledge from electronic-commerce data. As reported by Brodley and Kohavi [Brodley & Kohavi, 2000], most types of data-mining algorithm were tried by only a small fraction of participants.

There are several reasons why even expert data miners would ignore the vast majority of approaches. They may not have access to the tools; however, readily (and freely) available data-mining toolkits make this reason suspect. More likely, even experts are not facile with many data-mining tools—especially those that require additional pre- and post-processing. Indeed, the only algorithm that was tried by more than 20% of the KDDCUP-2000 participants was decision-tree induction, which often performs reasonably well on a wide variety of data with little pre- and post-processing.

An Intelligent Discovery Assistant (IDA) helps a user to explore the space of valid data-mining processes, expanding the portion of the space that they consider. The overall meta-process followed by an IDA is shown in Figure 4. An IDA interacts with the user to obtain data, metadata, goals and desiderata. Then it composes the set of valid DM processes, according to the constraints implied by the user inputs, the data, and/or the ontology. This composition involves choosing induction algorithm(s), and appropriate pre- and post-processing modules (as well as other aspects of the process, not considered in this paper). Next, the IDA ranks the suitable processes into a suggested order based on the user's desiderata. The user can select plans from the suggestion list, hopefully aided by the ranking. Finally, the IDA will produce code for and can execute (automatically) the suggested processes on the selected data.

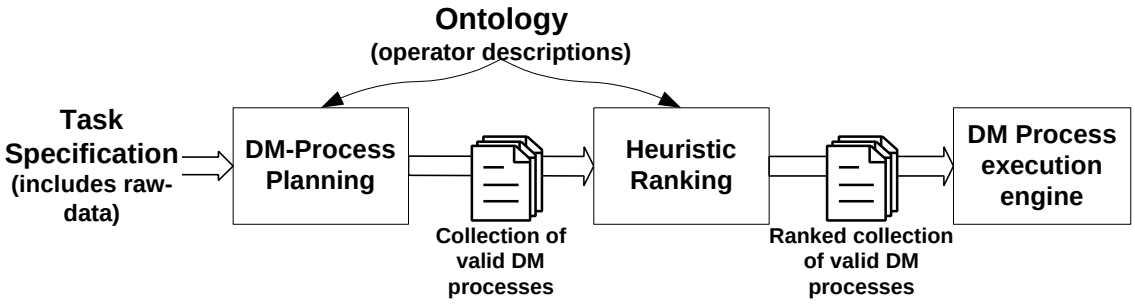

Figure 4: The overall process followed by an IDA

3 Enumerating Valid Data Mining Processes

Our first claim is that ontology-based IDAs can enumerate DM processes useful to a data miner. We support our claim in two ways. First, we describe how the ontology can enable the composition of only valid plans. Second, we describe process instances produced by our prototype (IDEA), in order to pro-

vide evidence that they can be non-trivial. Later we will describe how problem-specific elements can be incorporated into IDAs; for clarity and generality first we concentrate on domain-independent elements of the DM process. For example, when presented with a data set to mine, a knowledge-discovery worker (researcher or practitioner) generally is faced with a confusing array of choices [Witten & Frank, 2000]: should I use C4.5 or naive Bayes or a neural network? Should I use discretization? If so, what method? Should I subsample? Should I prune? How do I take into account costs of misclassification?

3.1 Ontology-based Intelligent Discovery Assistants

Consider a straightforward example: a user presents a large data set, including both numeric and categorical data, and specifies *classification* as the learning task (along with the appropriate dependent variable). The IDA asks the user to specify his/her desired tradeoffs between accuracy and speed of learning (these are just two possible desiderata). Then the IDA determines, of all the possible DM processes, which are appropriate. With a small ontology, there might be few; with a large ontology there might be many. For our example task, decision-tree learning alone might be appropriate. Or, a decision-tree program plus subsampling as a pre-process, or plus pruning as a post-process, or plus both. Are naive Bayes or neural networks appropriate for this example? Not by themselves. Naive Bayes only takes categorical attributes. Neural networks only take numeric attributes. However, a DM process with appropriate preprocessing may include them (transforming the data type), and may fare better than the decision tree. What if the user is willing to trade some accuracy to get results faster?

The IDA uses the ontology to assist the user in composing valid and useful DM processes. In the prototype, the ontology contains for each operator:

- A specification of the conditions under which the operator can be applied, involving a precondition on the state of the DM process as well as its compatibility with preceding operators.
- A specification of the operator's effects on the DM process's state and on the data.
- Estimations of the operator's effects on attributes such as speed, accuracy, model comprehensibility, etc.
- Logical groups, which can be used to narrow the set of operators to be considered at each stage of the DM process.
- Predefined schemata for generic problems such as target marketing.
- A help function to obtain comprehensible information about each of the operators.

Figure 5 shows a structural view of the prototype ontology, which groups the DM operators into three major groups: pre-processing, induction, post-processing. Each of these groups is further sub-divided. At the leaves of this tree are the actual operators (not shown in the figure, except for two examples: C4.5, PART). Specifically important for the empirical demonstrations below, the induction algorithm group is

subdivided into classifiers, class probability estimators (CPEs), and regressors. Classifiers are further grouped into decision trees and rule learners; the former includes C4.5 [Quinlan, 1993] and the latter includes PART [Frank & Witten, 1998].

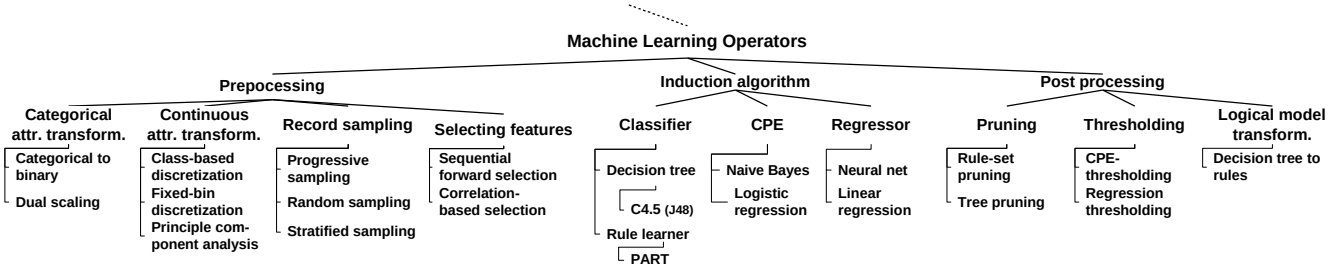

Figure 5: The Data-mining Ontology (partial view)

We have built an prototype IDA, the **I**ntelligent **D**iscovery **E**lectronic **A**ssistant (IDEA), that uses the ontology-based approach. Following the general framework for IDAs (see Figure 4), IDEA first gathers a *task specification* for the DM process, analyzes the data that the user wishes to mine and extracts the relevant meta-data, such as the types of attributes included (e.g., continuous, categorical). Using a GUI, the user then can complement the gathered information with additional knowledge about the data (such as structural attributes IDEA could not derive from the metadata), and can specify the type of information/model he/she wishes to mine and desired tradeoffs (speed, accuracy, cost sensitivity, comprehensibility, etc.). IDEA's first core component, the *DM-process planner*, then searches for DM processes that are valid given the task specification from within the design space of overall possible DM processes, as defined by the ontology. This is described in Section 3.2.

A collection of valid DM processes typically will contain a series of processes that are undesirable for certain user goals? they make undesirable trade-offs, such as sacrificing too much accuracy to obtain a model fast, etc. IDEA's second core component, the *heuristic ranker*, ranks the valid DM processes using one of several possible heuristic functions. The user's trade-off preferences are defined by weights entered through the GUI. Process ranking is treated in detail in Section 4. IDEA's GUI allows the user to sort the list of plans using any of the rankings (including a combined ranking derived from applying weights on the different characteristics), to examine the details of any process plan, and to generate code for and to run the process.

3.2 Enumerating Valid DM Processes: IDEA's procedure

Our first claim is that IDAs produce a systematic enumeration of DM processes that will be useful to data miners, and will keep them from overlooking important process instances. The general ontology-based methodology was outlined above. Now, we will describe the specific procedure used by the prototype IDEA, and will present some example DM processes enumerated for different DM tasks.

To enumerate (only) valid DM processes, IDEA performs a search of the space of processes defined by the ontology, constrained by the restrictions on operator application defined in the ontology. The structure of the search problem is amenable to more complex, AI-style planning, but so far the search-based approach has been sufficient. IDEA solves the search problem by constructing a step-by-step specification of DM operators (i.e., a DM process) that move from the start state (which includes some meta-data description of the data-set) to the goal state (typically a prediction model with some desired properties). Specifically, it starts with an empty process at the start state. At every state it then finds the applicable (or compatible) operators using the compatibilities, adds each operator to the partial process that brought it to the current state, and transforms the state using the operator's effects. From our example above, in order to apply Naïve Bayes, the current state must not contain numeric attributes; this would be the case only after discretization (or some other preprocessing). On the other hand, the planning would not apply discretization twice, because after the first application, the state no longer would contain numeric attributes, and thus the preconditions of discretization no longer would apply. The planner stops pursuing a given process when it has either reached the goal state or some "dead-end" state that will not lead to the goal state.

The central difference from traditional, AI planning techniques is that the algorithm does not stop executing when it has found a first viable solution, but instead searches for as many valid processes as possible. This approach is appropriate because knowledge discovery is an exploratory undertaking, and users often are not able to express their preferences precisely or completely before seeing possible available alternatives. This brings up a question of computational efficiency: will it be feasible to generate all possible processes in a reasonable amount of time? As long as the number of DM operators that will be available to an IDA is not huge, the speed of planning is unlikely to be problematic. For example, with the prototype DM ontology (currently incorporating a few dozen operators), the current DM-process planner can generate all valid processes (up to several hundred for problems with few constraints) in less than a second.

The constraints in the ontology are essential. For example, if we use the ontology whose overall structure is shown in Figure 5, give the goal of *classification*, and constrain the search only with the ordering of the logical groupings imposed by the prototype ontology (i.e., pre-processing precedes induction which precedes post-processing), IDEA generates 163,840 DM processes. Adding the constraints imposed by the pre- and post-conditions of the operators,[4] IDEA produces 597 valid process instances—less than one-half of one percent of the size of the unconstrained enumeration. Adding metadata (e.g., the data set contains numeric attributes) and/or user desiderata (e.g., the user wants cost-sensitive classification) allows the enumeration to be constrained even further.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

3.3 Enumerating Valid DM Processes: example enumerations from IDEA

The enumerations of processes produced by IDEA are not trivial. In many cases they would be valuable not only to novice data miners, but even to experts. As evidence, consider the following processes constructed by IDEA.

Example 1) When IDEA is given the goal of producing a *cost-sensitive classifier for a two-class problem*, it produces an enumeration comprising 189 DM processes. The enumeration includes building a class-probability estimator and setting a cost-specific threshold on the output probability. It includes building a regression model and determining (empirically) an effective threshold on the output score. The enumeration also includes using class-stratified sampling with any classification algorithm (which transforms an error-minimizing classifier into a cost-minimizing classifier). Novice data miners certainly do not consider all these options when approaching a cost-sensitive problem. In fact, we are aware of no single published research paper on cost-sensitive learning that considers one of each of these types of option [Turney, 1996].

Example 2) When we give IDEA the goal of producing *comprehensible classifiers*, the top-ranked DM process[5] is: `subsample the instances` → `feature selection` → `use a rule learner` → `prune the resultant rule set` (see Figure 6a). Although comprehensibility is a goal of much machine-learning research, we are not aware of this process being used or suggested. This process is interesting because each component individually has been shown to yield more comprehensible models; why shouldn't the composition yield even more comprehensible models? As another DM process highly ranked by comprehensibility, which in addition has a high accuracy ranking, IDEA suggests: `build a decision tree` → `convert tree to rules` → `prune rule set` (see Figure 6b). This also is a non-trivial suggestion: it is the process introduced by Quinlan [1987] and shown to produce a combination of comprehensibility and high accuracy. Although the addition to the ontology of `convert tree to rules` certainly was influenced by Quinlan's work, we did not "program" the system to produce this process instance. IDEA composed and ranked processes only based on knowledge of individual operators. This is particularly valuable, because the addition of a new operator to the ontology can have far-reaching effects (e.g., adding the "convert trees to rules" operator results in this plan being suggested strongly for comprehensible classification).

---

[4] These are not shown here, but are straightforward constraints such as: neural networks require numeric attributes, decision-tree pruning can only apply to decision trees, etc. (see the appendix).

[5] We discuss ranking next. Here it is sufficient to understand that these rankings are created by combining scores, included in the ontology, for the different operators that compose a KD process.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

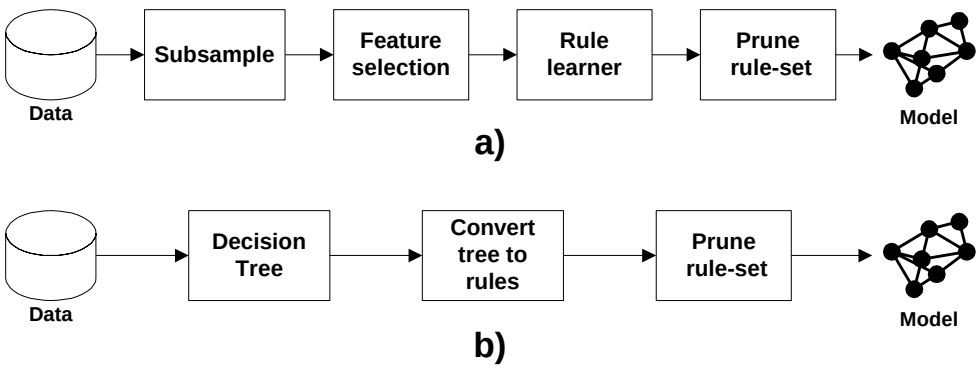

Figure 6: Two Plans for producing a comprehensible classifier

Example 3) Consider the case where the user is interested in classification, but wants to get results fast. As described in detail below, IDEA can rank processes quite well by speed, but does the enumeration contain particularly useful (fast) processes? Indeed it suggests processes that use fast induction algorithms, such as C4.5 (shown to be very fast for memory-resident data, as compared to a wide variety of other induction algorithms [Lim et al., 2000]). It also produces suggestions not commonly considered even by researchers studying scaling up inductive algorithms [Provost & Kolluri, 1999]. For example, the enumeration contains plans that use discretization as a preprocess. Research has shown that discretization as a preprocess can produce classifiers with comparable accuracy to induction without the preprocess [Kohavi & Sahami, 1996]; but with discretization, many induction algorithms run much faster. For example, as described by Provost and Kolluri, most decision tree inducers repeatedly sort numeric attributes, increasing the computational complexity considerably; discretization eliminates the sorting. IDEA's suggestions of fast plans also include plans that use subsampling as a preprocess. Most researchers studying scaling up have not considered subsampling explicitly, but of course it produces classifiers much faster—and for large data sets it has been shown to often produce classifiers with comparable accuracies [Oates & Jensen, 1997].

In sum, for a variety of types of tasks, IDEA's enumerations of DM processes are non-trivial: certainly for novices, and arguably even for expert data miners. In Section 6 we will present an extended example giving further support.

4 IDAs can produce effective rankings

The foregoing section argued that enumerating DM processes systematically is valuable, because it can help data miners to avoid missing important process instances. However, such enumerations can be unwieldy. It is important not only to produce an enumeration, but also to help the user choose from among the candidate processes. IDAs do this by first enumerating DM processes systematically, and then rank-

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

ing the resulting processes by characteristics important to the users (speed, accuracy, model comprehensibility, etc.).

Rankings of DM processes can be produced in a variety of ways. For example, static rankings of processes for different criteria could be stored in the system. We believe that flexible rankings also are important—so that as new ontological knowledge is added, the system can take advantage of it immediately. IDEA allows both static rankings and dynamic rankings. In particular, it produces rankings dynamically by composing the effects of individual operators. The ontology contains (in the form of scoring functions) estimations of the effects of each operator on each goal. For example, an induction algorithm may be estimated to have a particular speed (relative to the other algorithms). Taking a 10% random sample of the data as a preprocess might be specified to reduce the run time by a factor of 10 (which would be appropriate for algorithms with linear run times). Correspondingly, sampling might be specified to reduce the accuracy by a certain factor (on average), and to increase the comprehensibility by a different factor (cf., the study by Oates and Jensen [1997]). For a given DM process plan, an overall score is produced as the composition of the functions of the component operators.

The systematic enumeration of DM processes allows yet another method for ranking the resulting processes: because the processes are represented explicitly and reasoned about, the system can undertake auto-experimentation to help it produce rankings. Specifically, the system can run its own experiments to determine appropriate rankings by constructing processes, running them, and gathering statistics on their efficacy. Of course, it does not make sense to run a large number of processes to find out which would give results *fast*. On the other hand, if accuracy is crucial and speed is not a concern, it may make sense to run some or all of a process enumeration (e.g., automatically conducting a cross-validation study such as would be performed by an expert data miner).

Our next goal is to provide support for our claim that IDAs can provide useful rankings. We make no claim about what are the best ranking procedures.

4.1 Details of ranking experiments

In order to provide a demonstration to support our claim, we implemented a code generator for IDEA that exports any collection of DM processes, which then can be run (automatically). Currently it generates code for the Weka data-mining toolkit [Witten and Frank, 2000], and it generates Java code for executing the plans, as well as code for evaluating the resulting models based on accuracy and speed of learning. We chose to assess IDEA's ability to rank processes by speed and by accuracy, because these are criteria of general interest to users and for which there are well-accepted evaluation metrics (which is not the case for comprehensibility, for example). Furthermore, one expects a rough tradeoff between speed and accuracy [Lim, *et al.*, 2000], and a user of an IDA may be interested in points between the extremes—e.g., trading off some speed for additional accuracy. We return to this tradeoff in section 4.4.

For the experiments in this section, we restricted the ontology to a subset for which it is feasible to study an entire enumeration of plans thoroughly. The ontology subset uses seven common pre-processing, post-processing, and induction techniques (for which there were appropriate functions in Weka, see below). The experimental task is to build a *classifier*, and has as its start state a data set *containing at least one numeric attribute* (which renders some inducers inapplicable without preprocessing). Table 1 shows on the left the list of 16 valid process plans IDEA created for this problem; on the right is a legend describing the 7 operators used.[6] Even this small ontology produces an interesting variety of DM-process plans. For example, the ontology specifies that naïve Bayes only considers categorical attributes, so the planner needs[7] to include a preprocessor that transforms the data. Indeed, although the ontology for the experiments is very small, the diversity of plans is greater than in many research papers.

	steps	*heuristic rank*			
		credit-g accuracy	composition accuracy	credit-g speed	composition speed
Plan # 1	c4.5	3	6	13	13
Plan # 2	part	9	1	16	16
Plan # 3	rs, c4.5	14	14	2	5
Plan # 4	rs, part	4	9	8	10
Plan # 5	fbd, c4.5	5	8	12	11
Plan # 6	fbd, part	10	4	15	14
Plan # 7	cbd, c4.5	6	7	11	12
Plan # 8	cbd, part	8	2	14	15
Plan # 9	rs, fbd, c4.5	12	16	4	3
Plan # 10	rs, fbd, part	15	12	6	8
Plan # 11	rs, cbd, c4.5	7	15	5	4
Plan # 12	rs, cbd, part	16	10	6	9
Plan # 13	fbd, nb, cpe	1	4	8	6
Plan # 14	cbd, nb, cpe	2	2	10	7
Plan # 15	rs, fbd, nb, cpe	11	13	1	1
Plan # 16	rs, cbd, nb, cpe	13	11	3	2

Legend for operators used in plans	
acronym	*name/algorithm*
rs	Random sampling (result instances = 10% of input inst.)
fbd	Fixed-bin discretization (10 bins)
cbd	Class-based discretization (Fayyad & Irani's MDL method)
c4.5	C4.5 (using Witten & Frank's J48 implementation)
part	Rule Learner (PART, Frank & Witten)
nb	Naïve Byes (John & Langley)
cpe	CPE-thresholding post-processor

Table 1: 16 process plans and rankings

In Table 1, the first column ranks the plans by the number of operators in the plan. This may be interesting to users who will be executing plans manually, who may be interested in minimizing fuss. Not surprisingly decision-tree learning is at the top of the list, echoing the observation from the KDDCUP 2000 [Brodley & Kohavi, 2000]. We will not consider this ranking further except to reference plans by number.

The *heuristic rank* columns of Table 1 show two pairs of rankings computed by heuristics, one pair for accuracy and one for speed. The "credit-g" rankings are static rankings created by running all the plans

[6] The last operator in Table 1, `cpe`, which places an appropriate threshold on a class-probability estimator, becomes a no-op for Naïve Bayes (`nb`) in the Weka implementation, because Weka's implementation of `nb` thresholds automatically.

[7] This is not strictly true for the Weka implementation, for which naïve Bayes is augmented with a density estimator for processing numeric attributes. For this study, we considered strict naïve Bayes. The Weka implementation, to IDEA, would be considered naïve Bayes plus a different sort of numeric preprocessor.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

on one, randomly selected data set (viz., credit-g[8]). A static ranking makes practical sense if the flexibility to add new operators is not of primary importance. Adding new operators (or otherwise changing the ontology) changes the space of plans, in which case a static ranking would have to be updated or recomputed. The "composition" rankings were generated by a functional composition based on the accuracy and speed functions contained in the ontology. More specifically, to generate the heuristic rankings, the ontology specifies a base accuracy and speed for each learner, and specifies that all the preprocessing operators will reduce accuracy and will increase speed, by different amounts. The heuristic functions are subjective, based on our experience with the different data-mining techniques and on our reading of the literature (e.g., [Lim *et al.*, 2000]). The ranking functions were fixed before we began using Weka's particular implementations, with one exception: because speed ratings differ markedly by implementation, we ran Weka on one data set (again, credit-g) to instantiate the base speed for the three learning algorithms and the speed improvement factors for sampling and for discretization.

Our experiments are designed to assess the feasibility of using an IDA to provide rankings by speed and by accuracy. Specifically, the experiments compare the proposed rankings to rankings generated by actually running the plans on the data sets. For the experiments, we used 23 data sets from the UCI Repository [Blake & Merz, 2001], each containing at least one numeric attribute. The data sets and their total sizes are listed in Table 2. Unless otherwise specified, for each experiment we partitioned each data set randomly into halves (we will refer to these subsets as D_1 and D_2). We used ten-fold cross-validation within D_2 to compute average classification accuracy and average speed—which then are used to assess the quality of the ex-ante rankings, and to construct the "actual" (ex-post) rankings for all comparisons. (We will use the D_1s, later, to construct auto-experimentation rankings; the $\{D_1, D_2\}$ partitioning ensures that all results are comparable.)

[8] We did not use credit-g as a testing data set in our experiments.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

Dataset name	*Size*
heart-h	294
heart-c	303
ionosphere	351
balance-scale	625
credit-a	690
diabetes	768
vehicle	846
anneal	898
vowel	990
credit-g	1000
segment	2310
move	3029
dna	3186
gene	3190
adult10	3256
hypothyroid	3772
sick	3772
waveform-5000	5000
page	5473
optdigits	5620
insurance	9822
letter	20000
adult	32561

Table 2: Data set names and sizes

4.2 Ranking by Speed

Our first experiments examine whether the heuristics can be effective for ranking DM processes by speed. Since being able to rank well by speed is most important for larger data sets, let us consider the largest of our data sets: adult. Table 3 shows the two rankings from Table 1 and the actual (ex-post) ranking based on the average run times for all the plans. The table is sorted by the actual ranking, and the table entries are the positions of each plan in each ranking (i.e., 1 is the first plan in a ranking, 2 the next, and so on). Both heuristics rank very well. Using Spearman's rank-correlation statistic, r_s (recall that a perfect rank correlation is 1, no correlation is 0, and a perfectly inverted ranking is -1), to compare with the ideal ranking, we can examine just how well. For the credit-g ranking (on the adult data set), $r_s = 0.93$ and for the composition ranking, $r_s = 0.98$.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

Plan Name	credit-g ranking	composition ranking	D2 ("actual") ranking
Plan # 2	16	16	16
Plan # 6	15	14	15
Plan # 8	14	15	14
Plan # 1	13	13	13
Plan # 7	11	12	12
Plan # 4	9	10	11
Plan # 5	12	11	10
Plan # 14	10	7	9
Plan # 10	7	8	8
Plan # 12	7	9	7
Plan # 3	2	5	6
Plan # 13	9	6	5
Plan # 11	5	4	4
Plan # 9	4	3	3
Plan # 16	3	2	2
Plan # 15	1	1	1

Table 3: Adult data set rankings by speed

Table 4 shows for all the domains the correlations between the rankings produced by the heuristics and the ranking based on the actual speeds. Here and in the subsequent tables, the data sets are presented in order of increasing size (large ones toward the bottom). Highlighted in bold are the cases where $r_s > 0.5$ (all but the smallest data set).[9] Neither heuristic is superior, but both are effective; for both ranking heuristics, the average is approximately $r_s = 0.85$. These results show convincingly that it is possible for an IDA to rank DM processes well by speed.

[9] The choice of 0.5 was ad hoc, but was chosen before running the experiment. Examining hand-crafted rankings with various r_s values seemed to indicate that 0.5 gave rankings that looked good.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

	credit-g ranking	composition ranking
heart-h	0.39	0.30
heart-c	**0.62**	**0.59**
ionosphere	**0.80**	**0.70**
balance-scale	**0.82**	**0.81**
credit-a	**0.94**	**0.91**
diabetes	**0.55**	**0.64**
vehicle	**0.94**	**0.95**
anneal	**0.98**	**0.92**
vowel	**0.90**	**0.93**
segment	**0.89**	**0.92**
move	**0.90**	**0.95**
dna	**0.98**	**0.94**
gene	**0.92**	**0.95**
adult10	**0.97**	**0.97**
hypothyroid	**0.95**	**0.91**
sick	**0.95**	**0.89**
waveform-5000	**0.90**	**0.94**
page	**0.86**	**0.85**
optdigits	**0.89**	**0.87**
insurance	**0.95**	**0.93**
letter	**0.90**	**0.96**
adult	**0.93**	**0.98**
mean	**0.86**	**0.85**
median	**0.90**	**0.92**

Table 4: Spearman ranks for ranking heuristics for speed

4.3 Ranking by Accuracy

Ranking by speed is useful, but what about ranking DM processes in terms of the accuracy of the models they will produce? Our next set of experiments examines whether the IDA can be effective for ranking DM processes by accuracy. Note that one would not expect to be able to do nearly as well at this task as for ranking by speed. Nevertheless, it would be helpful to be able to give users guidance in this regard, especially when a system proposes a process containing a component with which the user is not familiar. If the process were ranked highly by accuracy, it would justify learning about this new component.

Credit-g and Composition Rankings

As in the speed experiments, we use the heuristic rankings to predict how the different DM processes would fare in terms of accuracy. Table 5 shows the correlations (using Spearman's r_s) between the heuristic rankings and the ranking determined empirically through cross-validation using D_2. As above, the table presents the test domains sorted by size. As expected, the accuracy results are less impressive than

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

the speed rankings (above). The mean r_s is 0.28 for the credit-g ranking and 0.53 for the composition heuristic. Examining the correlations for the composition ranking more closely, we see that in all but 3 (of 23) cases, the ranking is better than random, and in most cases it ranks surprisingly well by accuracy (17 of 23 have $r_s > 0.5$). However, for the diabetes data set the ranking is strikingly poor ($r_s = -0.52$),[10] pulling down the means (cf., the medians). We reiterate that our purpose was not to study the production of the best heuristic ranking functions; we believe that these could be improved considerably with further research. Nevertheless, these results clearly support our claim that IDAs can rank DM process plans (heuristically) by expected accuracy, and therefore can provide valuable assistance in choosing between different processes.

	credit-g heur	composite heur
heart-h	-0.23	0.12
heart-c	0.46	**0.62**
ionosphere	0.36	**0.76**
balance-scale	-0.16	-0.20
credit-a	-0.18	-0.15
diabetes	-0.64	-0.52
vehicle	**0.54**	**0.77**
anneal	**0.51**	**0.66**
vowel	**0.64**	**0.73**
segment	**0.55**	**0.86**
move	0.42	**0.82**
dna	**0.50**	**0.72**
gene	0.50	**0.91**
adult10	0.32	**0.75**
hypothyroid	0.34	**0.62**
sick	**0.62**	**0.70**
waveform-5000	0.46	**0.81**
page	0.00	0.23
optdigits	0.28	**0.54**
insurance	-0.14	0.31
letter	**0.58**	**0.84**
adult	0.35	**0.80**
mean	0.28	**0.53**
median	0.42	**0.70**

Table 5: Spearman ranks for heuristic ranking for accuracy

Auto-experimentation ranking

There is another option for producing accuracy rankings, which was not available for speed rankings. Specifically, an IDA can perform auto-experimentation, composing process plans and running its own

[10] Investigating this further we find that the differences between the accuracies of the different plans are statistically insignificant resulting in a high variance in the actual rankings.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

experiments to produce a ranking of the plans by accuracy.[11] Although this may initially seem ideal (albeit time consuming), we must remember that even careful experimental evaluations of the accuracies of predictive models are still only estimation procedures, with respect to the accuracy of the models on unseen data. The quality of the rankings of DM processes produced by such estimation will vary (e.g., by data-set size), and for any particular domain must be determined empirically. However, we know of no method of ranking by accuracy that performs better generally. Therefore, the auto-experimentation rankings can be considered an upper bound against which other ranking procedures can be compared.

We now present the results of an experiment to assess the effectiveness of such a procedure. For each domain, IDEA composed the DM process plans and generated Weka code for the plans (and for their evaluations via cross-validation). For each data set, the cross-validation was performed on data subset D_1 to produce an estimation of the accuracy that would result from running the plan on a data set from the domain. These accuracies were used to construct a ranking of the DM-process plans by accuracy for each data set. These rankings then were compared to the ranking produced on data set D_2 (identically to all previous experiments). Table 6 lists the resulting rank correlations. As expected, the auto-experimentation outperforms the other two rankings considerably. Notably, the empirically determined rankings are considerably better for the larger data sets. Consider the data sets with 5000 or more records. Averaged over these data sets, $r_s = 0.86$ for the empirically determined ranking, as compared to $r_s = 0.59$ for the heuristic ranking. A t test shows the difference in these means to be statistically significant at the $p < 0.05$ level ($p=0.011$), and the win:loss ratio of 6:0 also is significant (at $p<0.016$ by a sign test). Also of note, considering the auto-experimentation results as an upper bound places the results of the composition ranking in a much better light.

[11] This is not an option for speed rankings, because the auto-experimentation process itself may be (very) time consuming.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

	D1	credit-g ranking	composition ranking
heart-h	-0.06	-0.23	0.12
heart-c	0.06	0.46	**0.62**
ionosphere	0.20	0.36	**0.76**
balance-scale	**0.55**	-0.16	-0.20
credit-a	**0.71**	-0.18	-0.15
diabetes	0.49	-0.64	-0.52
vehicle	**0.91**	**0.54**	**0.77**
anneal	**0.90**	**0.51**	**0.66**
vowel	**0.90**	**0.64**	**0.73**
segment	**0.92**	**0.55**	**0.86**
move	**0.87**	0.42	**0.82**
dna	**0.91**	**0.50**	**0.72**
gene	**0.88**	0.50	**0.91**
adult10	**0.86**	0.32	**0.75**
hypothyroid	**0.96**	0.34	**0.62**
sick	0.18	**0.62**	**0.70**
waveform-5000	**0.94**	0.46	**0.81**
page	**0.74**	0.00	0.23
optdigits	**0.84**	0.28	**0.54**
insurance	**0.84**	-0.14	0.31
letter	**0.96**	**0.58**	**0.84**
adult	**0.86**	0.35	**0.80**
average	**0.70**	0.28	**0.53**
median	**0.86**	0.42	**0.70**

Table 6: Spearman rank correlation coefficient for the three different ranking methods

These results show that ranking by accuracy (not surprisingly) is difficult, but that via various methods an IDA can provide guidance as to which methods are expected to be more accurate. For small data sets, the composition heuristic and estimation via auto-experimentation perform comparably. For larger data sets, auto-experimentation outperforms the composition heuristic, but one pays a considerable run-time price as the data-set size grows.

4.4 Trading off Speed and Accuracy

Our long-term goal is not simply to be able to rank by speed or by accuracy, but to allow users to specify desired tradeoffs between different criteria. For example, consider larger data sets. For these, as shown in the previous section, auto-experimentation provides significantly better rankings than does the composition heuristic—but the auto-experimentation is time consuming. Presumably, as data sets get larger and larger, the accuracy of auto-experimentation will increase, but so will the computational cost. What if a user is willing to trade off some speed for a better accuracy ranking, but does not have the time for full-blown auto-experimentation (i.e., running all the plans on all the data)?

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

An alternative is to perform auto-experimentation on subsamples of the data for the purpose of estimating the accuracy ranking for the full data set. Presumably, as the estimation samples get larger the accuracy of the rankings improves, as does the computational cost. Our next experiment tests whether this in fact is the case—if it is, it will demonstrate that IDAs can allow users to trade off quality of assistance (in particular, the ranking of DM-process plans by expected accuracy) for timeliness of assistance.

For the experiment, IDEA ran the process plans for the six largest data sets (each having 5000 or more total records) on increasingly larger subsets of the data. Specifically, for each domain's D_1, we selected random subsets of 10%, 20%, …, 100% of the data. For each subset, IDEA performed cross-validation to determine empirically an expected accuracy ranking, identically to the previous experiment. For this experiment, we consider only the eight DM-process plans that do not contain random sampling.

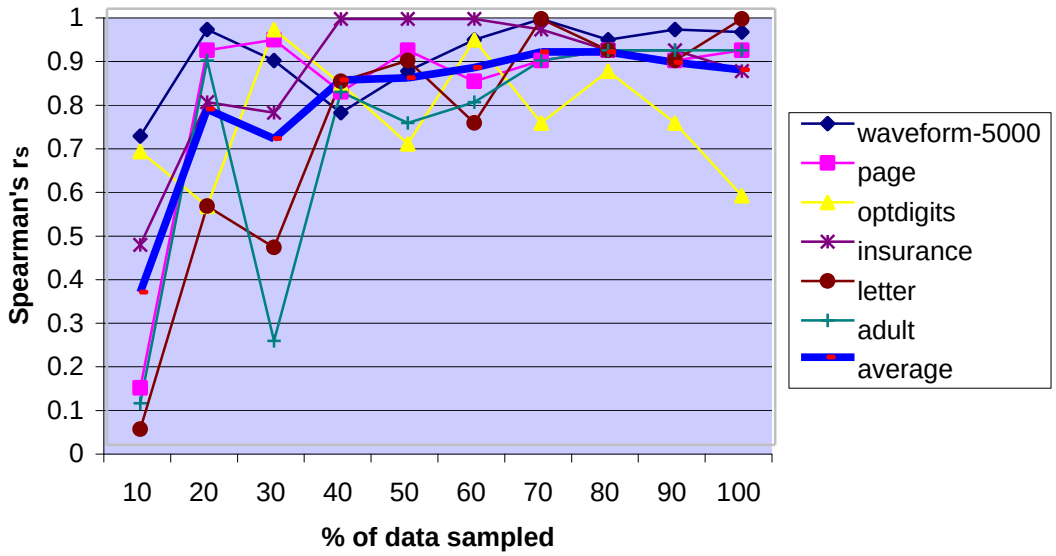

Figure 7: Rank correlations and sample size

Figure 7 plots the rank correlations as the size of the sample grows, for the six largest data sets, and in bold shows the average rank correlation as size grows. Clearly the largest samples give better rankings than the smallest ones. For the 100% sample, all are above 0.5, and all but optdigits are above 0.8. On the other hand, for several of the data sets (page, adult, letter) the rankings with the 10% sample are not much better than random. Recall from above that on these six (largest) data sets the composition ranking gives an average rank correlation of 0.59; comparing this with the results in Figure 7 suggests that even this rudimentary heuristic ranking is competitive with auto-experimentation until (on average) 20-30% of the data are used.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

With one notable exception, the rank correlations become relatively stable when about half of the data have been seen. The optdigits curve is unusual: the rank correlations do not increase and do not become more stable as more data are used. Further investigation shows that optdigits is, in an important sense, "too easy." Specifically, all methods perform extremely well, even with small training sets, so it is not possible to rank them meaningfully beyond a certain level (this still is substantially better than random). Figure 8 shows the graph without the optdigits data, showing that the average performance is as desired (generally increasing, but with decreasing marginal benefits).

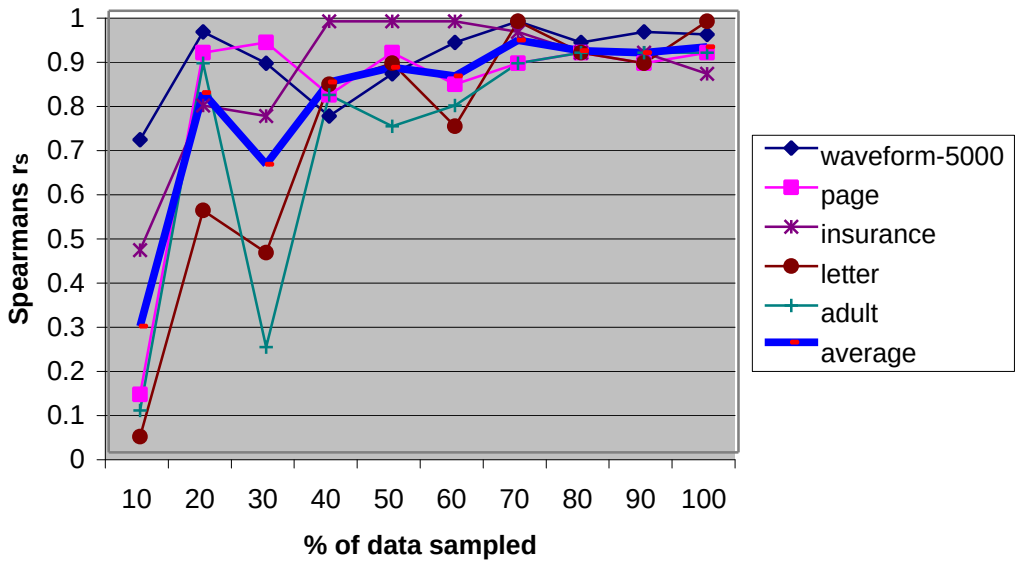

Figure 8: Rank correlations and sample size without the optdigits dataset

These results show clearly that it is possible to trade off longer response time for higher accuracy of recommendations (rankings). In particular, using fewer data reduces the (average) quality of the rankings produced, but does so considerably faster. Using more data improves the quality of the rankings, up to the maximally accurate ranking (the full-blown auto-experimentation ranking).

5 Knowledge Sharing and Network Effects

As we have argued, IDAs are particularly useful because they are systematic in their exploration of the design space of DM processes. Without such a tool users, even experts, seldom are systematic in their search of the DM-process space; it is unlikely that any user will consider all possible process plans. Therefore, users may overlook important, useful DM processes.

Up to this point for emphasis we have discussed novice users and expert users. However, this is not a true dichotomy—there is a spectrum of expertise along which users reside. For the most novice, any help

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

with DM process planning will be helpful. For the most expert, an IDA could be useful for double-checking his/her thinking, and for automating previously manual tasks, as well as for suggesting additional processes. For others along the expertise spectrum, IDAs will have both types of benefits. Furthermore, even among experts, different users have different expertise: a data miner trained in the statistics community and a data miner from the machine-learning community can be experts and novices with respect to different methods. An IDA may help to educate any user. For example, when the system produces a highly ranked plan that a user had not considered previously, the user can examine the ontology, and become educated on some new aspect of the DM process.

A unique benefit of an IDA based on an explicit ontology is the synergy it can provide between teams of users. If users contribute to the ontology, other users instantly receive the benefit of their contributions. Thus, IDAs exhibit what economists call *network externalities* or *network effects*: users get positive value from other people using the "network," and therefore the value of the network to each user increases as the network gets more users. In this case, an IDA becomes more valuable to each user as the number of contributing users grows. All users get the benefit of contributors' work automatically. Furthermore, no single member of the group has to be an expert in the entire body of data-mining technology.

Consider the following example of network effects in action. Georgia is a member of a large team of data miners, with several on-going projects. While reading the statistics literature she discovers a technique called *dual scaling* [Nishisato, 1994], a preprocessing operator that transforms categorical data into (scaled) numeric data, in a manner particularly useful for classification. Georgia codes up a new preprocessor (call it `DS`) and uses it in her work.

Such discoveries normally are isolated; they do not benefit a team's other projects. However, consider what happens if Georgia simply adds `DS` into the IDA, including adding the appropriate entry to the ontology. If another team member, Samuel, uses the system, DM-process plans may be generated that use `DS` (when appropriate). In some cases, these plans will be highly ranked (when `DS` is likely to do a good job satisfying the user's criteria). In such cases, Sam could experiment with `DS` immediately, or could read about it (using the documentation Georgia added), or could follow pointers to the literature, or could call Georgia directly and talk to her about it. Thereby, the tool brings to bear shared knowledge in the context of a particular need.

6 Demonstration with more complex DM process

We now present the results of a final set of experiments, to demonstrate further the power of IDAs. The prototypical DM-process template that we used for the discussions and experiments above was straightforward—as was necessary to introduce IDAs and to run experiments on a large number of benchmark data sets. However, in some real-world situations the DM process can be more complex

[Agrawal, 1998]. We assert that the potential value of IDAs is even greater in such cases, because there is even greater need for expertise in technique and in process itself.

The data we use for our demonstration were the subject of the 1998 KDDCUP. In 1998 the KDDCUP problem was to select a subset of a customer base to whom to mail solicitations, in order to maximize "profit" (revenues minus the cost of mailing). Participants built models from the training data, using a wide variety of different methods. To determine the "winners," the organizers evaluated (on a separate test set for which the true answers were hidden) how much profit each team's model would have garnered. More specifically, the 1998 KDDCUP was based on data from a fund-raising campaign undertaken by a national veterans association. The customer base was a set of individuals who donated in prior campaigns, and the goal was to select those from whom to solicit donations in the current campaign. Each observation in the data set is an individual, and includes (for example) the response to the prior campaign.

The training set from the competition consists of 95412 records and the test set consists of 96367 records. The mailing cost is $0.68 and the average donation is $15.60 with a range of $1-$200. The donation frequency is about 5% of the population. Using the default strategy of mailing to everyone, the average profit over the test set is $10,560. The results of the 1998 KDDCUP competition are presented in Table 7. For this experiment, we use the variables used in a study reported by Zadrozny and Elkan [Zadrozny and Elkan, 2001].[12]

Participants	Profit	%Gain
Urban Science	$14,712	39.32
SAS	$14,662	38.84
#3	$13,954	32.14
#4	$13,825	30.92
#5	$13,794	30.63
#6	$13,598	28.77
#7	$13,040	23.48
#8	$12,298	16.46
#9	$11,423	8.17
#10	$11,276	6.78
#11	$10,720	1.52
#12	$10,706	1.38
#13	$10,112	-4.24
#14	$10,049	-4.84
#15	$9,741	-7.76
#16	$9,464	-10.38
#17	$5,683	-46.18
#18	$5,484	-48.07
#19	$1,925	-81.77
#20	$1,706	-83.84
#21	($54)	-100.51

Table 7: Results of 1998 KDDCUP

[12] Note that selection and construction of features also is part of the KD process. We do not treat them in this paper, except in Limitations, below.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

This was a challenging competition: the spread between the different competitors is quite large. Notice that 9 of 21 entries produced lower profits than did the default strategy of mailing to everyone. In fact, the last-place entry actually lost money. The winners achieved a 39% increase in profit over the default strategy. Notice also that the winners are experts in this sort of data mining: Urban Science specializes in building models for target marketing (and in fact, they also won the 1997 KDDCUP). Close behind in second place is SAS, who also have extensive experience with this sort of modeling. The competitors with the lower scores most likely applied data mining tools in the manner typical of data-mining/machine-learning research. As we will demonstrate, the *straightforward* application of existing tools is insufficient for high-level performance on these data. However, the inclusion of application-specific, DM-process-related knowledge is.

We followed a methodology intended to mimic the algorithmic portion of the process that KDDCUP competitors would have taken. Specifically, we create rankings of DM processes considering only the training set (estimating the profit that would be obtained). To assess the quality of a ranking, we calculate the "actual" profits on the test set. The 1998 KDDCUP focused on a problem of cost-sensitive classification: classify into one of two categories, solicit or do not, taking into account the cost of false positives (the mailing costs) and the cost of false negatives (the lost revenue). We use a larger set of induction algorithms than in the experiments above, but for clarity, for this experiment we do not consider pre- and post-processing explicitly.

Process **NN**: Create dummies → Neural Network → Classification by regression

Process **Lin**: Create dummies → Linear Regression → Classification by regression

Process **Log(CPE)**: Create dummies → Logistic Regression(CPE) → CPE-Threshholding

Process **NB(CPE)**: Discretization → Naïve Bayes (CPE) → CPE-Threshholding

Process **Rule(CPE)**: Rule Learner(CPE) → CPE-Threshholding

Process **DT(CPE)**: Decision Tree(CPE) → CPE-Threshholding

Figure 9: DM processes generated for cost-sensitive classification

Figure 9 shows 6 DM processes generated for cost-sensitive classification. As mentioned above, a wider variety of learning algorithms (from Weka) is used here, and only one process with each algorithm is generated. Specifically, the first two processes produce regression models: process "NN" is the application of a neural network learner and process "Lin" is the application of linear regression. As mentioned in section 3.3, regression models can be converted to cost-sensitive classification models by a postprocessor which chooses (by experimenting with the training data) an appropriate threshold on the

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

predicted (output) value ("classification by regression"). Both of these algorithms require categorical variables to be preprocessed into a set of binary "dummy" variables. The last four processes use algorithms that create "class probability estimators," which give an estimation of the probability that a new example belongs to the class in question (here, "will donate"). Such a model can be converted to a cost-sensitive classifier with a postprocessor that chooses a threshold decision-theoretically, taking into account the misclassification costs. Process Log(CPE) uses logistic regression, which also requires preprocessing of categorical variables into dummies. Process NB(CPE) uses naïve Bayes, for which discretization is required as a preprocess. Processes Rule(CPE) and DT(CPE) build rule-based and decision-tree models, respectively; these do not require the preprocessing of numeric or categorical variables.

Table 8 shows the ranking of these processes by estimated profit, the actual profit calculated on the test set, and the resulting percentage gain over the default strategy of mailing to everyone. The profit was estimated by auto-experimentation (using cross-validation, as above) on the training data. Note that except for the neural network classifier, the ranking by estimated profit is perfect. Unfortunately, even without the error, the procedure would have placed only 9th (of 21) in the competition. What's worse, only one of the processes actually beats the default strategy of mailing to everyone. To be fair, this was a very difficult problem for data miners not intimate with modeling for problems such as target marketing. Indeed, the participants in the contest were serious data-mining researchers and tool vendors, and only half were able to do significantly better than the default strategy.

Plan	Rank	Profit	%Gain
NN	1	$6,919	-34.48
Lin	2	$11,968	13.33
Log(CPE)	3	$10,520	-0.37
Rule(CPE)	4	$9,924	-6.02
NB(CPE)	5	$9,538	-9.68
DT(CPE)	6	$8,496	-19.54

Legend for Operators Used in Plans	
acronym	**name/algorithm**
j48	Decision Tree
Log	Logistic Regression
NB	Naive Bayes
Rule	Rule Learner
Lin	Linear Regression
NN	Neural Network

Table 8: Process plans ranked by estimated profit, showing actual profit and gain over default strategy

What did the winners do differently? They did not use more complicated mining algorithms. Rather, *they used a different DM process*, one that is known by specialists to be particularly effective for target marketing. Specifically, as shown in Figure 10, a class probability estimator (CPE) is built to estimate the probability of donation; separately, a regression model is built (from the donors in the training set) to estimate the amount to be donated conditioned on the presence of a donation. These two models are used in combination: the product of the two, for any individual, estimates his/her expected donation. If the expected donation is greater than the cost of the promotion to that individual, in this case $0.68, then a

mailing should be sent. Otherwise it should not. This is the strategy used by the winner in the 1998 KDDCUP.

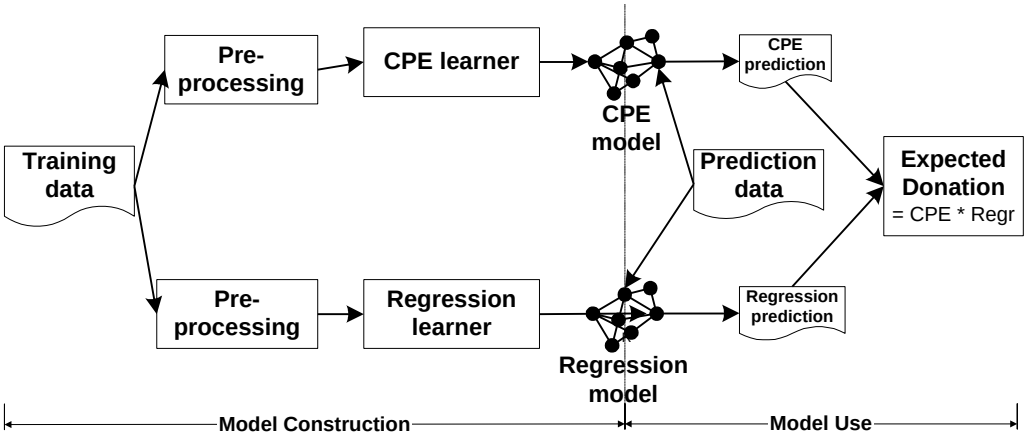

Figure 10: Target Marketing Process

We claim that such process knowledge, in this case about how to combine techniques to form effective special-purpose DM processes, can be added to an IDA's ontology by specialists, subsequently to be brought to bear by others. To the ontology we can add a template process for target marketing. Note that there still is a large degree of freedom, even given such a process template. What type of learner should be used for class-probability estimation? What type of regressor? Given the learner, what type of pre-/post-processing is required? The IDA will construct DM processes within the constraints imposed by this template, in addition to the simpler, default template (which we used in previous sections).

For our final experiment, we considered the cost-sensitive plans built with the default template and the plans built with the target-marketing template. In order not to bias the ranking with our prior knowledge (we know what the winners did), we use only auto-experimentation (cross-validation) to rank processes. In addition to the six process plans produced with the default (linear) DM process template, using the target-marketing template produces eight additional plans: the cross product of the available CPE learners (four) and the available regression learners (two). All the plans then are ranked by their estimated profit, produced via cross-validation on the training set. If one plan were to be submitted to a contest such as the KDDCUP competition, it would be the highest-ranking plan. Of course, we have the luxury of examining the entire list.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

Plan	Rank	Actual Profit	%Gain
Log(CPE) + NN	1	$14,914	41.23
Log(CPE) + Lin	2	$14,778	39.95
Rule(CPE) + NN	3	$13,672	29.47
Rule(CPE) + Lin	4	$13,456	27.42
DT(CPE) + NN	5	$11,055	4.69
NN	6	$6,919	-34.48
DT(CPE) + Lin	7	$10,843	2.68
Lin	8	$11,968	13.33
Log(CPE)	9	$10,520	-0.37
NB(CPE) + NN	10	$10,070	-4.64
RULE(CPE)	11	$9,924	-6.02
NB(CPE)	12	$9,538	-9.68
NB(CPE) + Lin	13	$10,113	-4.23
DT(CPE)	14	$8,496	-19.54

Legend for Operators Used in Plans	
acronym	name/algorithm
DT	Decision Tree
Log	Logistic Regression
NB	Naive Bayes
Rule	Rule Learner
Lin	Linear Regression
NN	Neural Network
CPE	Class Prob. Estimator

Table 9: Process plans ranked by estimated profit, showing actual profit and gain over default strategy

The fourteen process plans, ranked by cross-validated estimated profit, are listed in Table 9 along with their test-set profits and the percentage gain (loss) over the default mailing strategy. The estimated ranking reflects the actual profit ranking quite well (with a couple notable glitches; Spearman's $r_s = 0.798$). Indeed, the range of gains is remarkably similar to the actual ranking of submissions to the contest (note that we excluded processes such as: `(just) build a simple decision tree`, which produce zero profit). The top-ranked plans indeed are competitive with the winners' submissions. The penultimate plan is the one used by the winning submission, and performs comparably in terms of profit. We did not expect the IDA to perform this well, because we figured SAS and Urban Science must have left some tricks up their sleeves (e.g., proprietary twists on the modeling algorithms). The top-ranked process actually would have beaten the winning submission.

These results illustrate not only the power of the IDA generally to enumerate and to rank processes effectively, but also the power of the IDA as a knowledge-sharing device. If one specialist includes knowledge about the target-marketing process, and another includes knowledge about neural networks, and yet another includes knowledge about logistic regression,[13] other users would benefit from the IDA's composition of these to form a top-performing DM process.

7 Related Work

IDAs provide users with non-trivial, personalized "catalogs" of valid DM-processes, tailored to their task at hand, and help them to choose among the processes in order to analyze their data. We know of little work that directly studies IDAs for the overall DM-process, although some have argued that they are important [Brazdil, 1998; Morik 2000]. There is, however, quite a long tradition of work that ad-

[13] As was the case with us.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

dresses some of the same goals (such as recommending and ranking) or using similar techniques (e.g., planning, auto-experimentation, and the use of ontologies) for recommending and for ranking individual induction algorithms.

7.1 The Use of IDAs

Especially in the European community, researchers have argued for the importance of IDAs. Morik [2000], for example proposes to use a case-based repository to store successful chains of pre-processing operators.[14] As pre-processing chains are partial DM processes, the insights gained should complement our work, and ideally could be integrated with a system such as IDEA. The European MetaL project[15] has as one of its foci IDA-like systems; we are not aware of any existing system that uses background knowledge and/or experimentation to compose and rank DM *processes*, although Brazdil argues that it is important to do so [Brazdil, 1998].

The only implemented IDA-like system we are aware of was presented by Engels *et al.*, who describe a user-guidance module for DM processes called CITRUS ([Engels, 1996], [Engels *et al.*, 1997], [Wirth *et al.*, 1997], and [Verdenius and Engels, 1997]). In particular, the user-guidance module uses a task/method decomposition [Chandrasekaran *et al.*, 1992] to guide the user through a stepwise refinement of a high-level DM process, in order to help the user to construct the best plan using a limited model of operations. Finished plans are compiled into scripts for execution. The system is implemented by extending SPSS Inc.'s Clementine® system, which provides a visual interface to construct DM-processes manually.

This work is similar to our approach as it provides the user with assistance when constructing DM processes, and uses AI planning techniques. In contrast, our approach is based on two notions that have led us in a different direction. First, even with a well-specified goal it is very difficult to discern the one best plan, because the results of running data-mining methods are difficult to predict. Secondly, users' goals and desired tradeoffs often cannot be specified easily or completely at the onset of an investigation. This is because many desiderata are tacit and difficult to specify precisely (e.g., one may have an aversion to certain representations, based on experience with the domain experts). Moreover, knowledge discovery is an exploratory process; users must be given as much flexibility as possible. An IDA presents the user with many valid plans to choose from and helps him/her to choose among them, via rankings based on different criteria (and on combinations thereof). The user has no obligation to choose the highest-ranked plan in any given ranking—all of the plans in the ranking will be valid.

[14] see http://www-ai.cs.uni-dortmund.de/FORSCHUNG/PROJEKTE/MININGMART/index.eng.html
[15] MetaL stands for "Meta-Learning," the process of learning models of the performance of learning algorithms as a function of characteristics of data sets; see http://www.cs.bris.ac.uk/Research/MachineLearning/metal.html.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

7.2 Projects with Related Goals: Recommending and Ranking

A variety of research projects address issues regarding recommending/selecting optimal induction algorithms (rather than processes) and ranking induction algorithms. The MLT-Consultant [Craw 1992] was one of the first such systems. It used a MYCIN-type knowledge base [Davis 1984] with a Hypertext-based GUI to recommend to a user an algorithm to choose (from a machine-learning library). Several projects have since studied the selection of individual induction algorithms or subcomponents of algorithms based on certain forms of background knowledge. For example, Brodley [1995] chooses subcomponents to form a hybrid decision tree, based on expert knowledge of algorithm applicability. In Europe the StatLog project[16] [Michie et al., 1994] has investigated what induction algorithms to use given particular circumstances. Brazdil et al. [1994], Gama & Brazdil [1995], and others, use meta-rules drawn from experimental studies, to help predict which algorithms will be better; the rules consider measurable characteristics of the data (e.g., number of cases, number of attributes, kurtosis). This notion of "meta-learning" is the basis for the MetaL project, mentioned above. Finally, Hilario & Kalousis [2001] use a case-based system to advise users regarding which induction algorithm (and its respective parameter settings) to choose given a particular data-mining task.

A different tradition of meta-level systems for data mining [Buchanan et al., 1978], sometimes called "automatic bias selection," involves the selection of one of the following, based in part on feedback from the performance of the learner: vocabulary terms, the induction algorithm itself, components of the induction algorithm, parameters to the induction algorithm [desJardins and Gordon, 1995]. Bias-selection work generally assumes the goal is accuracy maximization, but also applies to other desiderata (exceptional examples include the work of Tcheng et al. [1989], who consider accuracy and speed, and that of Provost and Buchanan [1995], who consider accuracy, speed, and cost sensitivity).

Addressing the need for improved ranking methods, several research projects have studied the use of experimental comparison to rank individual induction algorithms. Brazdil [1998] summarizes some prior methods. This work is closely related to our ranking of DM processes (especially since one may put a conceptual box around a DM process and call it an induction algorithm, although this obscures important issues regarding the composition of processes). More recently, Brazdil and Soares have studied the ranking of individual induction algorithms, based on (functions of) their performances on previously seen data sets [Brazdil & Soares, 2000; Soares and Brazdil, 2000]. They compare various methods for ranking, which perform comparably, and they consider ranking combining accuracy and speed.

Generally, the knowledge generated from these and closely related projects could help to populate an IDA's ontology, as well as to inform the construction of more advanced heuristic functions for ranking DM processes.

[16] see http://www.ncc.up.pt/liacc/ML/statlog/index.html

7.3 Projects using similar techniques: Landmarking, Planning, Knowledge Management, and Ontologies

As we have seen, many of the component methods necessary for building IDAs have been the subject of recent study, especially in the European community. Several researchers have studied the notion of using fast processes (of different sorts) to help estimate the performance of less efficient ones. Pfahringer *et al*. [2000] and Fürnkranz and Petrak [2001] provide overviews of such "landmarking" techniques. In particular, Petrak [2000] presents a convincing analysis of the effectiveness of using subsamples from the data set in question to predict which learning algorithm will yield the lowest error on the entire data set; the technique works remarkably well—although it should be noted that for large data sets often one can achieve high accuracy with a surprisingly small subset of the data (cf., progressive sampling [Provost *et al*., 1999]). On the other hand, the relative performance of algorithms can change markedly with the amount of data [Perlich, *et al*., 2001].

St. Amant and Cohen [1998] study intelligent, computer-based support for open-ended, statistical/exploratory data analysis, which is akin to our approach. While focusing on somewhat different application areas—St. Amant's and Cohen's approach on statistical, exploratory data analysis and ours on the DM process—both approaches employ mixed-initiative planning, where an AI-planner proposes different courses of action. The two approaches differ, however, in how the human and the machine share control of the process. The application area of St. Amant's and Cohen's system, statistical/exploratory analysis, necessitates step-by-step guidance, where the user can (re-)evaluate each step and get advice on what to do next. Our approach, on the other hand, presents the user with all possible plans and forecasts of their (relative) performance. The user would then choose one (or more) of the plans, for example using those forecasts, run it, and then may re-run the system based on insights gained. This latter approach seems better suited in a domain (like knowledge discovery) where algorithms may run for extended periods of time. It would interesting, however, to explore a hybrid approach that would combine step-by-step guidance with overall planning allowing for the support of both integrated exploratory/ad-hoc and extended/long-running data analysis.

Kerber *et al.* [1998] document the DM process using active links to DM processes (that have been visually programmed) and to the rationale for major design choices. They collect these descriptions in a repository. This approach facilitates the reuse of DM processes, resulting in a knowledge management system for DM processes. It is complementary to our approach, as it emphasizes the documentation and retrieval of past knowledge, which could be integrated well with our notion of active support as represented by IDAs.

The only work of which we are aware that uses an explicit ontology within a meta-level machine-learning system is described by Suyama & Yamaguchi [1998]. As far as we can tell, this system uses the

ontology to guide the composition, by genetic programming, of fine-grained induction algorithm components (version space, star, entropy, entropy+information-ratio, etc.)

8 Limitations and Future Work

IDAs should not been viewed as automating the DM process totally. In contrast, intricate user interaction is critical to successful discovery. It is possible to provide automated, knowledge-based *assistance* for certain aspects of DM process design—as we have shown. We only have covered a few aspects so far, and for the most part only in the prototypical linear process. For example, our current prototype does not produce cyclic processes and our code generator does not yet produce code for more-complicated components, such as iterative feature selection (e.g., around a subprocess) or progressive sampling. This is the subject of on-going work—we do not believe that there are fundamental roadblocks. However, it should be clear that the space of DM processes will grow, and more knowledge or interaction may need to be brought to bear than is evident in the demonstrations we have provided here. On the other hand, this difficulty faces human data miners as well as IDAs, and the result seems to be that even expert humans end up using only a small set of tools, those with which they are familiar. Even moderately effective IDAs would expand this set.

Our experiments with rankings serve to demonstrate that valid processes can be ranked effectively. As stated above, we have not yet studied the production of rankings in depth. The related work on ranking induction algorithms should be very useful, as noted above, but also provides important caveats. For example, our use of the Spearman rank-correlation coefficient in effect weights equally the positions throughout a ranking. However, for our purposes, the processes near the top of the ranking probably would be the critical ones (especially for large number of generated process plans). Soares et al. [Soares, Costa & Brazdil, 2001] introduce a weighted modification to Spearman's coefficient, that takes into account position in the ranking. This same group of researchers also point out [Soares, Brazdil & Costa, 2000] other challenges in comparing rankings, stemming from the fact that the "ideal" ranking typically is based only on estimates of the true error rates. For example, one must deal with effective "ties" due to the lack of a demonstrable (statistically significant) difference between different ranked entities (algorithms for them, processes for us); Brazdil and Soares [2000] deal with this by averaging the Spearman's coefficients from the different folds of the cross-validation procedure (rather than using average accuracies from the folds to generate the ideal ranking). We will have to consider issues such as these if we want to study the ranking of process plans in more depth.

Furthermore, we only have considered here parts of the process that are relatively well understood. Preprocessing, induction algorithms, and post-processing have received relatively much attention in the literature. The existence of this body of knowledge has allowed us to construct an effective ontology. Other parts of the process are not as well understood or documented. For example, although feature con-

struction has received research attention for years, our understanding of when and how to use it effectively pales in comparison with our understanding of these other parts of the process. Consider the KDDCUP problem we presented above. We ignored the issue of feature construction, which (we assume) was crucial to success in the competition. Does enough knowledge exist to provide an IDA an ontology that will be effective to assist a user with feature construction? To our knowledge, this has yet to be shown convincingly. However, if generally effective methods or problem-specific heuristics exist, an IDA should be able to incorporate them. We also have assumed that the user will perform the selection of the discovery task(s) to perform. A future task is to extend the intelligent assistance to the part of the process involving the selection of discovery tasks. This typically is ignored in discussions of the knowledge discovery process, but was addressed in early knowledge discovery work by Lenat [1982] and recently by Livingston, Rosenberg, and Buchanan [2001a,b].

Finally, we believe that although studies such as this are necessary for the development of useful IDAs, we also need well-designed (and executed) user studies to assess whether IDAs actually are effective in helping real data miners. Such studies could, furthermore, also provide some indications what features of IDAs are most effective in supporting the knowledge discovery process and, therefore, provide guidance for further improvements of IDAs. We believe that the IDEA/WEKA combination will be sufficient to undertake such studies and hope to follow this path of investigation in future work.

9 Conclusion

Both novices and specialists need assistance in navigating the space of possible DM processes. We have introduced ontology-based IDAs, arguing that they can generate valid, non-trivial, and sometimes surprisingly interesting DM-process instances. Further, we have given empirical evidence that it is possible for IDAs to rank process instances effectively by various user criteria. Finally, we have argued that IDAs can be particularly useful as a knowledge-sharing environment for teams of data miners, creating network effects wherein the tool becomes increasingly valuable as it gets more and more users.

The knowledge discovery process has been a key concept in the field of KDD for a decade, but very little research addresses it explicitly. After having undertaken this work, we understand better why. Treating the DM process requires a tremendous breadth of knowledge of research and practical technique. Even most researchers in the area know only a fraction of what is necessary to do a comprehensive job of building an ontology (and we certainly have mistreated certain topics, although we have been careful). In retrospect, we believe even more strongly that in order for research on the knowledge discovery process to advance, systems like IDAs are essential—they document and automate parts of the process that are better understood, in order for research to concentrate on the large areas that are not well understood.

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

Acknowledgements

We thank Yan Mao for her assistance with the implementation of the prototype system, IBM for a Faculty Award, and Carlos Soares for extensive comments on a prior draft. We are indebted to the authors of Weka, for providing the free data-mining toolkit, to the librarians of and contributors to the UCI Repository, and to the organizers and participants of the 1998 KDDCUP.

References

[Agrawal, 1998] Agrawal, N., 1998, http://www.kdnuggets.com/meetings/kdd98/gain-kddcup98-release.html

[Blake & Merz, 2001] Blake, C.L. & Merz, C.J., 2000. UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

[Brazdil et al., 1994] P. Brazdil, J. Gama, & B. Henery. Characterizing the Applicability of Classification Algorithms Using Meta-Level Learning. In *Proceeedings of the European Conference on Machine Learning (ECML-94)*, pp. 83-102.

[Brazdil, 1998] Pavel. B. Brazdil. Data Transformation and Model Selection by Experimentation and Meta-learning. In Proceedings of ECML-98 Workshop on Upgrading Learning to the Meta-Level: Model Selection and Data Transformation, pp. 11-17.

[Brazdil & Soares, 2000] P. Brazdil & C. Soares. A Comparison of Ranking Methods for Classification Algorithm Selection. In *Proceedings 11th European Conference on Machine Learning (ECML-2000)*: 63-74.

[Brodley, 1995] C. Brodley. Recursive Automatic Bias Selection for Classifier Construction. *Machine Learning* 20(1-2): 63-94.

[Brodley and Kohavi, 2000] Brodley, C. and R. Kohavi. KDD-Cup 2000 Peeling the Onion. Talk at KDD-2000, Boston, August 2000.

[Buchanan et al., 1978] Buchanan, B., C. Johnson, T. Mitchell, and R. Smith, 1978. Models of Learning Systems. In J. Beltzer, (ed.), *Encyclopedia of Computer Science and Technology*, pp. 24-51.

[Chandrasekaran *et al.*, 1992] Chandrasekaran, B., Johnson, T. R., and Smith, J. W. 1992. Task-Structure Analysis for Knowledge Modeling. CACM 35, 9, 124-137

[Chapman et al. 2000] Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, and Rüdiger Wirth, CRISP-DM 1.0: Step-by-step data mining guide, *SPSS White paper – technical report CRISPWP-0800*, SPSS Inc., 2000

[Craw *et al.*, 1992] Craw, S., Sleeman, D., Graner, N., Rissakis, M., and Sharma, S. 1992. CONSULTANT: Providing Advice for the Machine Learning Toolbox. *Proceedings of the Research and Development in Expert Systems IX* CUP, Cambridge, 5 - 23

[Davis 1984] R. Davis. Interactive Transfer of Expertise. In B. Buchanan and E. H. Shortliffe, editors. Rule-based Expert Systems, pages 171-205. Addison-Wesley, Reading, MA. 1984

[desJardins and Gordon, 1995] desJardin, M., and D. Gordon (1995) Special Issue on Bias Evaluation and Selection. Machine Learning 20, 1/2.

[Engels, 1996] Engels, R. 1996. Planning Tasks for Knowledge Discovery in Databases; Performing Task-Oriented User-Guidance. Proceedings of the International Conference on Knowledge Discovery & Data Mining AAAI-Press, Portland, OR, 170-175

[Engels et al., 1997] Engels, R., Lindner, G., and Studer, R. 1997. A Guided Tour through the Data Mining Jungle. Proceedings of the 3nd International Conference on Knowledge Discovery in Databases. Newport Beach, CA

[Fayyad & Irani, 1993] Fayyad, U.M., and K. B. Irani. Multi-Interval discretization of continuous-valued attributes for classification learning. In Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence, Chambery, France. San Francisco: Morgan Kaufman, pp. 1022-1027.

[Fayyad, Piatetsky-Shapiro & Smyth, 1996] Fayyad, U., Piatetsky-Shapiro, G. and P. Smyth. The KDD process for extracting useful knowledge from volumes of data. CACM 39 (11), pp. 27-34.

[Frank & Witten, 1998] Frank, E., and Witten, I. H. Generating Accurate Rule Sets Without Global Optimization, 1998. In Proceedings of the Fifteenth International Conference on Machine Learning, 144-151.

[Fürnkranz & Petrak, 2001] J. Furnkranz and J. Petrak. An Evaluation of Landmarking Variants. In C. Giraud-Carrier, N. Lavrac, and S. Moyle (eds.), Working Notes of the ECML/PKDD 2001 Workshop on Integrating Aspects of Data Mining, Decision Support, and Meta-Learning, pp. 57-68.

[Gama & Brazdil, 1995] J. Gama and P. Brazdil: Characterization of Classification Algorithms. 7th Portuguese Conference on Artificial Intelligence, EPIA '95, pp. 189-200.

[Hilario & Kalousis 2001] M. Hilario and A. Kalousis, "Fusion of Meta-Knowledge and Meta-Data for Case-Based Model Selection," University of Geneva, Geneva UNIGE-AI-01-01.

[John & Langley, 1995] George H. John and Pat Langley. Estimating Continuous Distributions in Bayesian Classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. pp. 338-345. Morgan Kaufmann, San Mateo.

[Kerber et al., 1998] Kerber, R., Beck, H., Anand, T., and Smart, B. 1998. Active Templates: Comprehensive Support for the Knowledge Discovery Process. Intl. Conf. on Knowledge Discovery and Data Mining, 244-248

[Kohavi & Sahami, 1996] Ron Kohavi and Mehran Sahami, 1996. Error-Based and Entropy-Based Discretization of Continuous Features. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), pp. 114-119.

[Lenat, 1982] D. Lenat, 1982. AM: Discovery in Mathematics as Heuristic Search. In R. Davis and D. Lenat (eds.), *Knowledge-based Systems in Artificial Intelligence*, pp. 3-255. New York: McGraw-Hill.

[Lim et al., 2000] Lim, T.-S., Loh, W.-Y., and Shih, Y.-S. A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms. Machine Learning 40 (3), 203-228.

[Livingston, Rosenberg & Buchanan, 2001a] G. Livingston, J. Rosenberg, and B. Buchanan, 2001. Closing the Loop: an Agenda- and Justification-Based Framework for Selecting the Next Discovery Task to Perform. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pp. 385-392. IEEE Computer Society Press.

[Livingston, Rosenberg & Buchanan, 2001b] G. Livingston, J. Rosenberg, and B. Buchanan, 2001. Closing the Loop: Heuristics for Autonomous Discovery. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pp. 393-400. IEEE Computer Society Press.

[Michie et al., 1994] D. Michie, D. Spiegelhalter, and C. Taylor (eds.). *Machine Learning, Neural and Statistical Classification.* Ellis Horwood.

[Morik, 2000] Morik, K., *The Representation Race - Preprocessing for Handling Time Phenomena*. Proceedings of the European Conference on Machine Learning 2000 (ECML 2000), Ramon López de Mántaras and Enric Plaza (ed.), Lecture Notes in Artificial Intelligence, Vol. 1810, Springer Verlag Berlin, 2000.

[Nishisato, 1994] Nishisato, S. *Elements of Dual Scaling: An Introduction to Practical Data Analysis.* Hillsdale: Lawrence Erlbaum Associates.

[Oates & Jensen, 1997]. Oates, T. and D. Jensen, 1997. The effects of training set size on decision tree complexity. In Proceedings of the Fourteenth International Conference on Machine Learning, pp. 254-262.

[Perlich, et al., 2001] C. Perlich, F. Provost, and J. Simonoff. "Tree Induction vs. Logistic Regression: A Learning-curve Analysis." CeDER Working Paper #IS-01-02, Stern School of Business, New York University, NY, NY 10012.

[Petrak, 2000]. J. Petrak. Fast Subsampling Performance Estimates for Classification Algorithm Selection. Technical Report TR-2000-07. Austrian Research Institute for Artificial Intelligence.

[Pfahringer, et al., 2000]. Meta-Learning by Landmarking Various Learning Algorithms. In P. Langley (ed.), Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000), pp. 743-750. San Francisco: Morgan Kaufmann.

[Provost & Buchanan, 1995] F. Provost & B.. Buchanan. Inductive Policy: The Pragmatics of Bias Selection. Machine Learning 20(1-2): 35-61 (1995).

[Provost et al. 1999] F. Provost, D. Jensen, and T. Oates. Efficient Progressive Sampling. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 23-32.

[Provost & Kolluri, 1999]. Provost, F. & V. Kolluri. A Survey of Methods for Scaling Up Inductive Algorithms. *Data Mining and Knowledge Discovery* 3 (2):131-169, 1999.

[Quinlan, 1987] J. R. Quinlan: Simplifying Decision Trees. *International Journal of Man-Machine Studies* 27(3): 221-234 (1987)

[Quinlan, 1993] J. R. Quinlan (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

[Senator, 2000] T. Senator (2000). Ongoing Management and Application of Discovered Knowledge in a Large Regulatory Organization: A Case Study of the Use and Impact of NASD Regulation's Advanced Detection System (ADS). In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 44-53.

[Soares & Brazdil, 2000] C. Soares & P. Brazdil. Zoomed Ranking: Selection of Classification Algorithms Based on Relevant Performance Information. In *Proceedings of Principles of Data Mining and Knowledge Discovery, 4th European Conference (PKDD-2000),* 126-135.

[Soares, Brazdil & Costa, 2000] C. Soares, P. Brazdil, & J. Costa. Measures to Evaluate Rankings of Classification Algorithms. In H. Kiers, J. Rasson, P. Groenen and M. Schader (eds.), *Data Analysis, Classification, and Related Methods, Proceedings of the Seventh Conference of the International Federation of Classification Societies (IFCS),* pp. 119-124. Springer.

[Soares, Costa & Brazdil, 2001] C. Soares, P. Brazdil, & J. Costa. Improved Statistical Support for Matchmaking: Rank Correlation Taking Rank Importance into Account. In *JOCLAD 2001: VII Jornadas de Classificao e Analise de Dados*, pp. 72-75.

[St. Amant & Cohen, 1998] St. Amant, R., and Paul R. Cohen. Intelligent Support for Exploratory Data

Analysis. Journal of Computational and Graphical Statistics, Volume 7, Number 4, Pages 545–558

[Suyama & Yamaguchi 1998] A. Suyama & T. Yamaguchi. Specifying and Learning Inductive Learning Systems using Ontologies. In *Working Notes from the 1998 AAAI Workshop on the Methodology of Applying Machine Learning: Problem Definition, Task Decomposition and Technique Selection*, pp. 29-36.

[Tcheng, et al., 1989] D. Tcheng, B. Lambert, S. Lu, L. Rendell: Building Robust Learning Systems by Combining Induction and Optimization. IJCAI 1989: 806-812.

[Turney, 1996] Turney, P., 1996. Cost-sensitive Learning Bibliography. http://extractor.iit.nrc.ca/bibliographies/cost-sensitive.html.

[Ulrich and Eppinger, 1995] Ulrich, K. T., and Eppinger, S. D. 1995. Product Design and Development. McGraw-Hill, New York, NY,

[Verdenius and Engels, 1997] Verdenius, F., and Engels, R. 1997. A Process Model for Developing Inductive Applications. Proceedings of the Seventh Belgian-Dutch Conference on Machine Learning Tilburg, NL, 119-128

[Wirth et al., 1997] Wirth, R., et al. Towards Process-Oriented Tool Support for KDD. Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery Trondheim, Norway

[Witten and Frank, 2000] Witten, I., and E. Frank (2000). Data Mininig: Practical machine learning tools and techniques with Java implementations. San Francisco: Morgan Kaufmann Publishers.

[Zadrozny and Elkan, 2001] Zadrozny, B. and C. Elkan (2001). Learning and Making Decisions When Costs and Probabilities are Both Unknown. Technical Report No. CS2001-0664, The University of California, San Diego, CA

Appendix A: Operators used by IDEA

Record Sampling

Random Sampling
- Preconditions
  - Large Number of Records
- Postconditions
  - NOT(Large Number of Records)
- Indices
  - Speed: **30**
  - Accuracy: **-50**
  - Model Comprehensibility: **10**

Stratified Sampling
- Preconditions
  - Large Number of Records
- Postconditions
  - NOT(Large Number of Records)
  - Binary Classification Task
- Indices
  - Speed: **20**
  - Accuracy: **0**
  - Model Comprehensibility: **0**

Progressive Sampling
- Preconditions
  - Large Number of Records
- Postconditions
  - NOT(Large Number of Records)
- Indices
  - Speed: **10**
  - Accuracy: **0**
  - Model Comprehensibility: **1**

Categorical attribute transformation

Categorical to binary
- Preconditions
  - Continous
- Postconditions
  - NOT(Continous)
  - Large Number of Attributes
- Indices
  - Speed: **-5**
  - Accuracy: **-10**
  - Model Comprehensibility: **-10**

Dual Scaling
- Preconditions
  - Categorical
- Postconditions
  - Continous
  - NOT(Categorical)
  - NOT(Speed)
  - In Comprehensible Vocabulary
- Indices
  - Speed: **-50**
  - Accuracy: **-1**
  - Model Comprehensibility: **-50**

Continous attribute transformation

Fixed Bin Discretization
- Preconditions
  - Continous
- Postconditions
  - NOT(Continous)
  - Categorical
- Indices
  - Speed: **21**
  - Accuracy: **-5**
  - Model Comprehensibility: **0**

Class Based Discretization
- Preconditions
  - Continous
- Postconditions
  - NOT(Continous)
  - Categorical
- Indices
  - Speed: **20**
  - Accuracy: **-1**
  - Model Comprehensibility: **-10**

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

Principle Component Analysis
- Preconditions
  - NOT(Categorical)
  - NOT(Large Number of Records)
  - Large Number of Attributes
- Postconditions
  - Continous
  - NOT(Large Number of Attributes)
  - NOT(Speed)
  - ModelSizeSmall
  - In Comprehensible Vocabulary
- Indices
  - Speed: **-50**
  - Accuracy: **-1**
  - Model Comprehensibility: **-5**

Feature Selection

Feature Selection
- Preconditions
  - Large Number of Records
- Postconditions
  - NOT(Large Number of Records)
  - NOT(Speed)
- Indices
  - Speed: **-50**
  - Accuracy: **10**
  - Model Comprehensibility: **20**

Speed Sampling (Features/Attributes)
- Preconditions
  - Large Number of Attributes
- Postconditions
  - NOT(Large Number of Attributes)
- Indices
  - Speed: **100**
  - Accuracy: **-20**
  - Model Comprehensibility: **10**

Induction algorithm

C4.5
- Preconditions
- Postconditions
  - Speed
  - Comprehensible Model
  - Classifier
  - Tree
  - NOT(Large Number of Attributes)
  - AND NOT(Large Number of Records)
- Indices
  - Speed: **30**
  - Accuracy: **20**
  - Model Comprehensibility: **40**

Naive Bayes
- Preconditions
  - NOT(Continous)
  - NOT(Has Missing Values)
- Postconditions
  - Speed
  - ModelSizeSmall
  - Comprehensible Model
  - Class Probability Estimator
  - Equation
- Indices
  - Speed: **30**
  - Accuracy: **30**
  - Model Comprehensibility: **20**

Logistic Regression
- Preconditions
  - NOT(Categorical)
  - NOT(Has Missing Values)
- Postconditions
  - Speed
  - ModelSizeSmall
  - Comprehensible Model
  - Class Probability Estimator
  - Equation
- Indices
  - Speed: **-10**
  - Accuracy: **30**
  - Model Comprehensibility: **20**

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02

Rule Learner
- Preconditions
- Postconditions
  - Speed
  - Comprehensible Model
  - Classifier
  - Rule Set
- Indices
  - Speed: **-20**
  - Accuracy: **30**
  - Model Comprehensibility: **50**

Neural Net
- Preconditions
  - Continous
  - NOT(Categorical)
- Postconditions
  - NOT(Speed)
  - NOT(ModelSizeSmall)
  - NOT(Comprehensible Model)
  - Regressor
- Indices
  - Speed: **-50**
  - Accuracy: **50**
  - Model Comprehensibility: **-50**

Post Processing

Decision Tree to rules
- Preconditions
  - Tree
- Postconditions
  - NOT(Tree)
  - Rule Set
- Indices
  - Speed: **-1**
  - Accuracy: **0**
  - Model Comprehensibility: **5**

Tree Pruning
- Preconditions
  - Tree
- Postconditions
  - ModelSizeSmall
  - Comprehensible Model
- Indices
  - Speed: **-10**
  - Accuracy: **10**
  - Model Comprehensibility: **10**

Rule Set Pruning
- Preconditions
  - Rule Set
- Postconditions
  - NOT(Speed)
  - ModelSizeSmall
  - Comprehensible Model
- Indices
  - Speed: **-20**
  - Accuracy: **10**
  - Model Comprehensibility: **20**

CPE-thresholding post-processor
- Preconditions
  - Class Probability Estimator
- Postconditions
  - Cost Sensitive
  - Classifier
  - NOT(Class Probability Estimator)
- Indices
  - Speed: **0**
  - Accuracy: **0**
  - Model Comprehensibility: **0**

Classification Via Regression
- Preconditions
  - Regressor
- Postconditions
  - Cost Sensitive
  - Classifier
  - NOT(Regressor)
- Indices
  - Speed: **5**
  - Accuracy: **0**
  - Model Comprehensibility: **0**

Center for Digital Economy Research
Stern School of Business
Working Paper IS-02-02