

Knowledge-Based Data Mining

Sholom M. Weiss, Stephen J. Buckley, Shubir Kapoor, and Søren Damgaard
IBM T.J. Watson Research Center,
P.O. Box 218, Yorktown Heights, NY 10598, USA
sholom@us.ibm.com, sbuckley@us.ibm.com, shubirk@us.ibm.com, sddam@dk.ibm.com

ABSTRACT

We describe techniques for combining two types of knowledge systems: expert and machine learning. Both the expert system and the learning system represent information by logical decision rules or trees. Unlike the classical views of knowledge-base evaluation or refinement, our view accepts the contents of the knowledge base as completely correct. The knowledge base and the results of its stored cases will provide direction for the discovery of new relationships in the form of newly induced decision rules. An expert system called SEAS was built to discover sales leads for computer products and solutions. The system interviews executives by asking questions, and based on the responses, recommends products that may improve a business' operations. Leveraging this expert system, we record the results of the interviews and the program's recommendations. The very same data stored by the expert system is used to find new predictive rules. Among the potential advantages of this approach are (a) the capability to spot new sales trends and (b) the substitution of less expensive probabilistic rules that use database data instead of interviews.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications - Data Mining

Keywords

decision rule induction, expert systems, sales leads

1. INTRODUCTION

Expert systems and machine learning methods are two prominent approaches to encoding knowledge in a computer. Both approaches can employ similar representations for knowledge. For example, both can use logical decision rules for classification. A key distinguishing characteristic of the two approaches is that an expert system can be built from an expert's knowledge and not inferred from often incomplete

sample data. From this perspective, data are used for evaluation of an expert system's performance on test cases.

Expert systems and machine learning methods can be combined. Evaluation of a pre-existing knowledge base matches the true answer to the answer given by the expert system. The true answer is found by some independent means. For example, a medical diagnosis system will reach a conclusion that is not necessarily correct. We know this by comparing to an objective measure, such as the human expert's diagnosis or perhaps the result of some procedure that ascertains the exact answer like a costly lab test. Case data are run through a knowledge base, and the labels assigned by the expert system are compared to the known correct labels.

Evaluation is the central thesis of many mating schemes of data and rule-based systems. The knowledge base is considered an expert's theory, an encoding of domain knowledge. Generally, it is correct, but may have weaknesses. The evaluation helps uncover the weak areas, and the knowledge base is refined. The human engineer or expert modifies the program to improve its accuracy. Methods of automated knowledge base refinement[4] attempt to modify the knowledge base. The knowledge base is revised to resolve mislabeled sample cases, while maintaining the overall integrity of the expert-derived knowledge. An alternative approach is case-based reasoning[5], where learning methods are sometimes used to augment the expert's current store of cases.

In our application, we consider the knowledge base to be correct. We have no aspirations to modify the knowledge base. The data used in the knowledge base will provide a basis for data mining the stored results of previous applications of the knowledge. A key objective is to find less expensive ways of obtaining predictions usually in terms of likelihood instead of the complete certainty of the expert system. An expert system called SEAS has been developed that interviews executives of small and medium-sized companies and then recommends possible solutions for problem areas, such as supply chain software. Many of the questions asked by the program can only be answered by individuals intimately familiar with all segments of a company's business operations. The expert system provides very accurate results, but requires an interview, a hard-to-obtain objective. Can we find less expensive ways that may be probabilistic, but still provide increased likelihood for sales leads?

From one perspective, the problem is not unlike medical diagnostic testing. We only perform expensive tests when there is an increased likelihood of a suspected outcome. We usually settle for less predictive triage before embarking on a

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGKDD '03 Washington, D.C., USA

Copyright 2003 ACM 1-58113-737-0/03/0008 ...\$5.00.

Figure 2: Tasks of Unified System

1. Construct expert system
 2. Give consultations
 3. Store results:
 - question-answer pairs
 - program recommendations
 4. Induce decision rules from stored data
-

full protocol of testing. In a simple incarnation of a solution, a decision tree might be built, knowing that the terminal nodes are the conclusive results. If it is too costly to reach the terminal nodes, then we estimate the likelihood of reaching the terminal node from other nonterminal nodes. When the likelihood for an antecedent node reaching the terminal node is below a fixed threshold, no further tests are ordered.

2. METHODS AND PROCEDURES

Figure 1 describes the SEAS expert system for finding sales leads for small and medium companies. It interviews executives and recommends solutions for their problems. Figure 4 describes an extended system that unifies SEAS with data mining. Figure 2 is a list of the tasks that are performed in an integrated expert and data-mining system.

2.1 The Knowledge Base

Our concept pre-supposes an expert system and its associated knowledge base. Many new applications may not have readily accessible sample data, and the complexity of a knowledge base may necessitate a very large data sample to test all program variations. In our specific application, many of the questions asked by the program cannot be answered by lookup in commercial or public data sources. It's up to the data-mining process to discover new relationships or assign probabilities to partial relationships for questions and answers that are present in the knowledge base and have surrogates in commercial databases. If these discoveries are made, substitutes for the interview process can be found.

Expert systems are a relatively mature field, at least for rule-based systems. The type of system for our application is sometimes called a consultation system. The programs ask questions and reach conclusions. A consultation system needs two components: (a) a questioning strategy and (b) a classification strategy. The questioning strategy interviews a person with a problem, and the classification reaches conclusions and make recommendations. Many expert systems are rule-based, and their questioning strategies examine the structure of the decision rules and their satisfied components to determine the most plausible next question. Alternatively, as is the case of SEAS, our sales-lead system, the topics are well-organized and have their own local structure. For example, the program can ask whether a company is competitive on production costs, and if not ask further questions about the sources of increased costs.

The questioning strategy accesses valid questions and answer pairs. The complete set of pairs is the list of items that

Figure 3: Data Acquisition Procedure

1. Ask question, record answer, and branch within XML questions
 2. Invoke decision rules to identify problem types
 3. Recommend solutions corresponding to problem types
 4. Save question-answer pairs and recommended solutions
-

can be mined, the attributes. In this application, they are all binary, where numerical attributes have been discretized to simplify the questioning: a user is asked to pick one of the specified ranges. A collection of decision trees can implement a questioning strategy for a circumscribed domain, and our application system uses a standard XML representation for this purpose.

The classification mechanism is a set of decision rules. They identify problem types and relate these problems to possible solutions. For our data mining application, we will store both the attributes, i.e. question-answer pairs, and the program's recommendations. Figure 3 describes the expert systems strategies for data acquisition from interviews.

2.2 The Data

In a standard evaluation of a classification system, we would have an additional item recorded for each example, each interview in our application. The correct answers would be stored, and we might measure accuracy. In the lead generation application, our expert system is assumed correct. The questions that we ask are implicitly expensive to obtain from any source, other than interviewing someone intimately familiar with the company. But once this information is obtained, the solution path is exact. We can accept the correctness of the knowledge base, and the recommendations of the expert system.

We have two components to any data-mining application for prediction: the measured attributes and the labels. Certainly, we know the attributes. These are the question-answer pairs. We would be very interested in using the same set of attributes to make a higher level prediction, not yet captured by the expert system. What would be a reasonable prediction? In our application, the purpose of the expert system is to obtain sales leads. Thus the ultimate prediction is whether a sale closes and its projected revenue. Knowing the eventual outcome, we could learn some form of score or classification, that if successful, would find patterns for the closed sales and not the failures. Most interviews do not end in a sale for many reasons including expense of purchase or a loss to a competitor. The very same attributes of the expert system are appropriate for machine learning. However, we must obtain a new set of labels, those recorded by tracking the eventual outcomes, much like we measure the efficacy of a medical treatment by the eventual outcome to a patient.

If we could obtain these labels for outcomes, we have a classical prediction problem for labeled data. In many circumstances, it is not easy to track outcomes. The results are only known over time; the results of an initial contact may

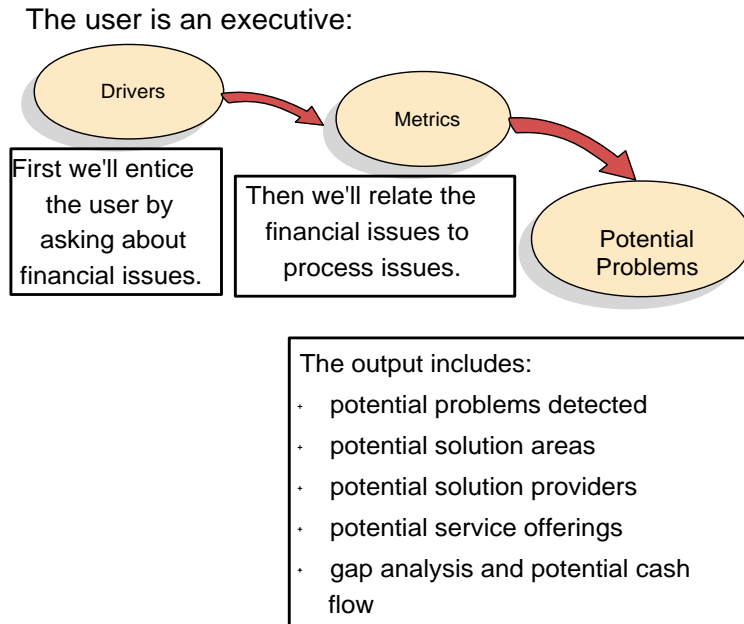


Figure 1: Overview of SEAS

not be stored in the same database as the ultimate outcome. In our application for lead determination, the outcome is not reliably tracked, passing through several databases with non-unique company identifiers.

Having obtained outcome labels and solved the prediction problem, what have we achieved? Using the answers to the questions of the expert system, we can predict success or failure of the sales proposal. This may be useful, but we also know that many of the questions that the expert system asks may be difficult to obtain by alternative means, such as a commercial database. The interview must be given by the sales lead program.

In our procedures, we relax the requirements of obtaining the labeled outcomes and the complete expert system interviews. Without this information, we show how useful predictors may still be obtained. Using our medical analogy, our goal is to perform less expensive screen tests, that may have weaker predictive power, but still give useful indications of likelihood of outcomes.

2.3 Specifying the Problem

Our goal is to mine the data accumulated from sessions of the expert system. In our application, interviews results, expressed as questions and answers, are stored. We have also stored the program's conclusions. If we map the expert system's tasks into a classification and prediction problem, the objective is to determine mappings from measurements

to labels. Thus we need to specify the the usual two sets, the measurements and the labels, or for our application the question-answer pairs $\{QA\}$ and the labels or recommended solutions $\{S\}$. These are already the information provided by the knowledge base. If we continue with the usual prediction model, then our objective is to find mappings from $\{QA\}$ to $\{S\}$. However, we already have a system that does this correctly, the expert system. Moreover, if we use the very same representation as the knowledge base and have a large enough sample of data, we may actually induce the very same rules that we already have in the knowledge base.

First, let's consider the labels. The outcomes, in terms of win or loss, are best, but the recommendations of the expert system are a reasonable proxy. Looking at our application, we accept the recommendations as correct, and we expect that customers who need the specified solution are far better targets than those who do not. Therefore, they are good leads if we can find them, especially without the interview. We will maintain the same labels as the knowledge base.

Instead of using all attributes, we will work with a subset, $\{QA'\}$, those that are least costly to obtain, preferably available in a commercial or public database. For example, the knowledge base may ask demographic questions about a company's industry or number of employees. Circumscribing empirical learning to those attributes that are inexpensive may weaken the likelihood of obtaining the strongest rules, but it will guarantee inexpensive rules. Under the

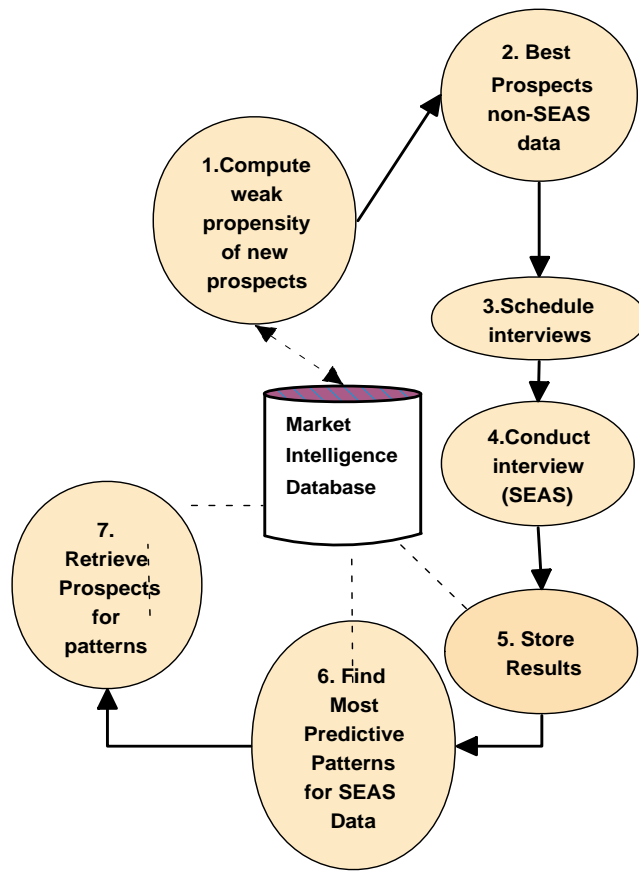


Figure 4: Unifying SEAS with Data Mining

best of circumstances, decision rules may be induced that are inexpensive, simpler, yet serve as full surrogates for the original rules for the knowledge base.

Returning to our medical test analogy, we now have a mechanism of inferring less expensive rules for the original database. These rules may be probabilistic as opposed to certain. These rules may also vary over time. During one time frame, some industries' fortunes may change, making them less or more likely to need some of the offered solutions. This information could be valuable in determining leads, for example switching to industries whose prospects are strengthening during that time period.

Because a consultation system is the basis of gathering data, we cannot expect to have large samples. Over time, as more interviews are conducted, the sample will grow, but never to the size of those from fully automated batch systems. Moreover, the sample may not be completely stationary, and we may decide to keep only the most recent data. The knowledge base of our application has 1200 binary attributes—most questions are multiple choice. Some 200 possible non-mutually exclusive potential recommendations are the labels for our application.

2.4 Learning from the Data

The goal is to induce decision rules for the expert system's recommendations. Although all recommendations and answers to questions have been stored, only the least expensive

questions will be examined. The problems will be treated as binary classification problems, the same tasks performed by the expert system. Rule induction will be the method used to find classifiers, an approach completely compatible with the expert system's method of classification. Because the number of stored examples will be relatively small, we want a method that is compatible with resampling techniques like crossvalidation. The expert system is growing in use; the sample will also grow in size, but there are reasons why it may not grow large. The main reason is that trends may not continue, and a time window on the sample may be necessary. Moreover, it may be worthwhile to segment the data by geography. Trends in Europe or Asia may not be the same in the USA, especially for a program geared to small and medium-size businesses.

Figure 5 shows an example of typical DNF rules for a single class, where f_i are attributes. Together all rules for a class form a rule-set. In a binary classification problem, we can use the failure of any rules to be satisfied as indicators of class 2, i.e. to not recommend the specified solution. The complexity of a rule-set is described with two measurements: (a) the number of rules and (b) the total number of components in the rule-set.

Because we are looking for maximum interpretability of the decision rules, complex solutions, such as those generated by ensemble methods [6],[1], will be less desirable. Instead we want a simple set of DNF rules like those illustrated

Figure 5: Typical DNF Rule-Set for a single class

1. $\{f_1 \leq 5.2 \text{ AND } f_2 \leq 3.1 \text{ AND } f_7 \leq .45\} \Rightarrow \text{Class1}$
 2. $\{f_1 \leq 2.6 \text{ AND } f_3 \leq 3.9 \text{ AND } f_8 \leq 5.0\} \Rightarrow \text{Class1}$
 3. ... $\Rightarrow \text{Class1}$
-

Figure 6: Greedy Rule Induction

1. Grow conjunctive term T (until the maximum length or until false positive errors are zero) by greedily adding conditions that minimize error.
 2. Record T as the next rule R . Remove cases covered by T , and continue with step 1 until all cases are covered.
-

in Figure 5. As illustrated for rules of our application in Table 2, our rule process is even simpler because all attributes are binary. The classical methods of rule induction, using covering rules and pruning strategies, are most compatible with the original expert system. The covering strategies of classical methods for noisy data are well-known. Either a covering tree is used, or a simple greedy method such as illustrated in Figure 6 is used. We used a greedy algorithm with some local optimizations[7],[3].

The more interesting variations of our application are the pruning strategy and its rule selection procedure. While the objective is to identify as many strong prospects as possible, the strongest rules with highest predictive values are the most interesting. So false positives are worse errors than false negatives because of the large space of companies potentially having the characteristics identified by the rules. We emphasize the most promising candidates.

Given the moderate size of the sample, weakest link pruning[2] is a reasonable choice. It allows us to match different-size sets of rules, and then pick one set by some standard, usually minimum or near-minimum error. The method prunes rule-sets by a complexity measure, such as a statistical test with confidence levels. The measure that we use is err/var , where we prune a rule-set at the point where the number of errors introduced per number of discarded components is minimum. The procedure is repeated on the new smaller rule-set. The result of this process is a series of k rule-sets, $RS(1)...RS(k)$, ordered by complexity $C(1)..C(k)$. Each one of these can be tested by crossvalidation or independent test data, and a error rate measured. An example of this process is illustrated in Table 1, where we have 7 rule-sets. The covering rule-set has 9 rules and 10 components, and its error is estimated at .1236. The ‘*’ is the minimum error, and the ‘**’ is within 1 standard error of the minimum, as are all the intermediate rule-sets. Figure 7 describes a procedure for selecting the best rule-set.

The rule-set procedure just described will select rules that are useful in minimizing the overall error. Within that

Figure 7: Rule-Set Selection Procedure

1. $i = 1$; $RS_1 =$ covering rule-set; Find C_i and save RS_i
 2. If RS_i has more than 1 component then $i = i + 1$ then repeat step 1.
 3. Evaluate all RS_i by crossvalidation
 4. Apply criteria to select best rule-set:
 - Find min test error
 - Consider all simpler rule-sets within range of 1 standard error of min test error:
 - Pick the one with highest predictive value, or ask expert to pick the one that makes sense to expert.
 - Consider only rule-sets with predictive value greater than a threshold.
-

framework, additional criteria are introduced to find the most desirable rules for the sales people. If the sample is very large, we can perhaps test individual rules, but in a smaller sample, this approach is amenable to effective testing by resampling. Most importantly, the experts can find a direction that is compatible with their own practice.

3. RESULTS AND DISCUSSION

These procedures were applied to 89 examples of expert system consultations. The original rules in the knowledge base are relatively shallow, typically having 2 or 3 components. The results of rule induction were summarized, an abstract of which is given in Table 2. The table lists the recommended solution, i.e. the class, the frequency that it occurred in the sample, the list of conditions in the decision rules, and the predictive value of the rule-set. At this stage, 60 of the 200 potential recommendations have sufficient data to support predictive rules. In addition, some rules were pruned by the experts as being artifacts of a small sample.

The SEAS expert system is a highly successful sales tool. Its success can be measured objectively in many millions of real dollars. Dozens of sales have been concluded following SEAS interviews. The initial contact is a simple enticement for the executive to get a free evaluation of the company’s operations. The SEAS program is currently being rolled out worldwide, and the program’s text has been translated into many languages. Using an XML representation of questioning and simple decision rules, the SEAS system is readily maintained and distributed, even with 1200 attributes and 200 classes.

At this stage, the data mining effort is an emerging technology that adds value to the original effort. Storage of data from the interviews has recently commenced, and the empirical results presented here are from an early stage of data collection. Examples of recorded interviews are rapidly growing in numbers. The rules found by rule induction may sometimes repeat those already known by the expert system. In other cases though, the rules are inherently less expensive

Table 1: Example of Rule Induction Error Summary

RSet	Rules	Vars	Train Err	Test Err	Test SD	MeanVar	Err/Var
1	9	10	.0000	.1236	.0349	9.9	.00
2*	6	7	.0337	.1011	.0320	7.0	1.00
3	5	5	.0787	.1236	.0349	5.0	2.00
4	4	4	.0899	.1236	.0349	4.0	4.00
5**	3	3	.1011	.1124	.0335	3.0	1.00
6	2	2	.1910	.1910	.0417	2.0	8.00
7	1	1	.3820	.3820	.0515	1.0	17.00

Table 2: Examples of Decision Rule Predictors

Solution Area Recommended	Frequency in Sample	Reason Recommended	Rule Accuracy
Procurement and Replenishment	45%	High cost of goods sold	100%
CRM Services	51%	Low revenue growth OR high fixed asset ratio OR high SG & A costs	98%
Supply Chain Services	62%	High cost of goods sold OR low revenue growth	96%

and can be invoked by direct lookup in a database. Instead of computing a score over all businesses in a database, the businesses satisfying a rule can be immediately retrieved using the standard database interface. By limiting the induction to inexpensive attributes, we may detect recent trends in sales.

Traditionally rule-based expert systems use data for evaluation or for refinement of the knowledge base. We have not revised the knowledge base with complementary rules found by a machine learning program. Rather, we have shown how a fully evaluated and presumed correct knowledge base can be used to guide a machine learning program in its endeavor to find alternative, very desirable ways to supplement the more expensive system. In our application for sales leads, we already know that the concept is technically feasible: we have very promising results for a relatively small sample. If results on larger samples, substantiate these results, then we have a strong addition to our marketing tools. Starting with an expert system requiring expensive interviews, we have a means of circumventing the interview and still finding promising leads. Additionally, some inexpensive attributes, like demographics for company size or industry codes, may uncover rapidly changing trends in sales needs.

Many consultation systems will have questions that are distinguished by cost of acquisition. We can expect that the techniques described here are readily adaptable to those systems.

4. REFERENCES

- [1] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.
- [2] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth, Monterrey, CA., 1984.
- [3] W. Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine*

Learning, pages 115–123, 1995.

- [4] A. Ginsberg, S. Weiss, and P. Politakis. Automatic knowledge base refinement for classification systems. *Artificial Intelligence*, 35:197–226, 1988.
- [5] J. Kolodner. *Case-based Reasoning*. Morgan Kaufmann, 1993.
- [6] R. Schapire. A brief introduction to boosting. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 1401–1405, 1999.
- [7] S. Weiss and N. Indurkha. Optimized rule induction. *IEEE EXPERT*, 8(6):61–69, December 1993.