





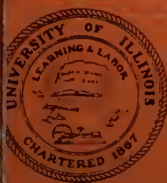
UNIVERSITY OF  
ILLINOIS LIBRARY  
AT URBANA-CHAMPAIGN  
BOOKSTACKS





330  
B385  
No. 1300 COPY 2

STX



# BEBR

FACULTY WORKING  
PAPER NO. 1300

Applying Inductive Learning to Enhance  
Knowledge-based Expert Systems

*Michael Shaw*

College of Commerce and Business Administration  
Bureau of Economic and Business Research  
University of Illinois, Urbana-Champaign



# **BEBR**

FACULTY WORKING PAPER NO. 1300


College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

November 1986

Applying Inductive Learning  
to Enhance Knowledge-based Expert Systems

Michael Shaw, Asst. Professor  
Department of Business Administration



Digitized by the Internet Archive  
in 2011 with funding from  
University of Illinois Urbana-Champaign



Applying Inductive Learning  
to Enhance Knowledge-based Expert Systems\*

Michael Shaw

Decision and Information Sciences Group  
Department of Business Administration  
University of Illinois  
Champaign, IL 61820, USA

ABSTRACT

This paper describes the use of inductive learning in MARBLE, a knowledge-based expert system I have developed for assisting business loan evaluation. Inductive learning is the process of inferring classification concepts from raw data; I use this technique to generate loan-granting decision rules based on historical and pro-forma financial information. A learning method is presented in this paper that can induce decision rules from training examples.

Key Words: Expert Systems for Business Loan Evaluation, Knowledge Acquisition, Inductive Learning.

---

\*This research is supported in part by a grant from the Office of Information Management.

7.

SHIPPED 6 1992

## I. Introduction

A major issue in designing knowledge-based expert systems for decision support is the process of knowledge acquisition: the encoding of human expertise into decision rules that can be incorporated in the expert system. The knowledge acquisition process is the main bottleneck in building expert systems for several reasons. First, even an expert of the given problem domain may not be aware of exactly what decision rules he/she has been applying; usually it would take a knowledge engineer to spend tedious interview sessions with the expert to identify a useful set of rules that can capture the necessary expertise and experience in the given domain. Second, there may not be experts in some domains or, when there are several experts specialized in the same area, it is often difficult to get consensus on the set of decision rules to use. Third, even when the decision rules have been determined and employed in the knowledge base, the expert system still needs to have a means to refine the rules continuously.

This paper describes a research aimed at automating the knowledge acquisition process for knowledge-based decision support. The principal objective is to investigate machine learning technique for deriving decision rules in the expert systems. Specifically, an inductive learning method that can extract concepts or decision rules from raw data is described; such a method can be characterized as learning by example because the set of raw data provides decision examples made by human experts. Throughout this paper, I shall describe the application of inductive learning in MARBLE, a knowledge-based expert system for business loan evaluation.

The remainder of the paper is organized as follows. Section II reviews the knowledge-based approach to decision support and describes a prototype I have developed for business loan evaluation, emphasizing the importance of machine learning in such a system. Section III introduces concepts acquisition by inductive learning. Section IV describes an algorithm for inductive learning. Section V describes an example on commercial loan evaluation using the approach presented in Sections III and IV. Finally, Section VI discusses such related issues as probabilistic reasoning and conceptual clustering in the context of inductive learning and knowledge-based decision support.

## II. MARBLE: A Knowledge-based Decision-Support System

### II.1 An Overview of the System

The research described in this paper is part of an ongoing effort to develop a knowledge-based expert system specializing in financial decision support for commercial banks. The system, referred to as MARBLE (standing for "an expert system for managing and recommending business loan evaluation"), is a MYCIN-based system [7] consisting of decision rules for evaluating commercial loans. It applies the judgment exercised by experienced loan officers in arriving at lending decisions for commercial loans.

MARBLE's architecture consists of three major program modules: the Consultation Module, the Explanation Module, and the Knowledge Acquisition Module, as shown in Figure 1. The Consultation Module interacts with the user to obtain information about the factual information for the problem, so as to generate decision information. An example of the question/answering session between the Consultation Module and the user



of MARBLE is shown in Appendix 1. In the process of decision support, the Explanation Module can provide justifications for the rules taken or explain the question posed to the user. The Knowledge-Acquisition Module is used to derive decision rules or to refine MARBLE's knowledge base. The inductive learning method described in this paper is to be embedded in this module.

-----  
Insert Figure 1 Here  
-----

Characterized by the often large amount of data and program modules (models) involved, a decision-support system is usually linked with an external database and a model base [2]. It has been shown that the knowledge-based expert system provides a very good environment for this type of decision support [25]. The system's problem-solving process, then, consists of a sequence of operations utilizing information from the knowledge-base, external database, dynamic database (sometimes referred to as working memory), and model base. In the case of MARBLE, the model base can contain program modules for financial analysis, forecasting, simulation, or regression. The external database typically contains the historical loan data and financial information of companies applying loans. Therefore, special care has been taken to handle the interface between the system's knowledge-base, model base, and database [26].

## II.2 Modeling the Loan-evaluation Decision

Typically, the evaluation of a business-loan application is a subjective decision process made independently by loan officers, bank controllers, auditors, and bank examiners. The loan-granting decision

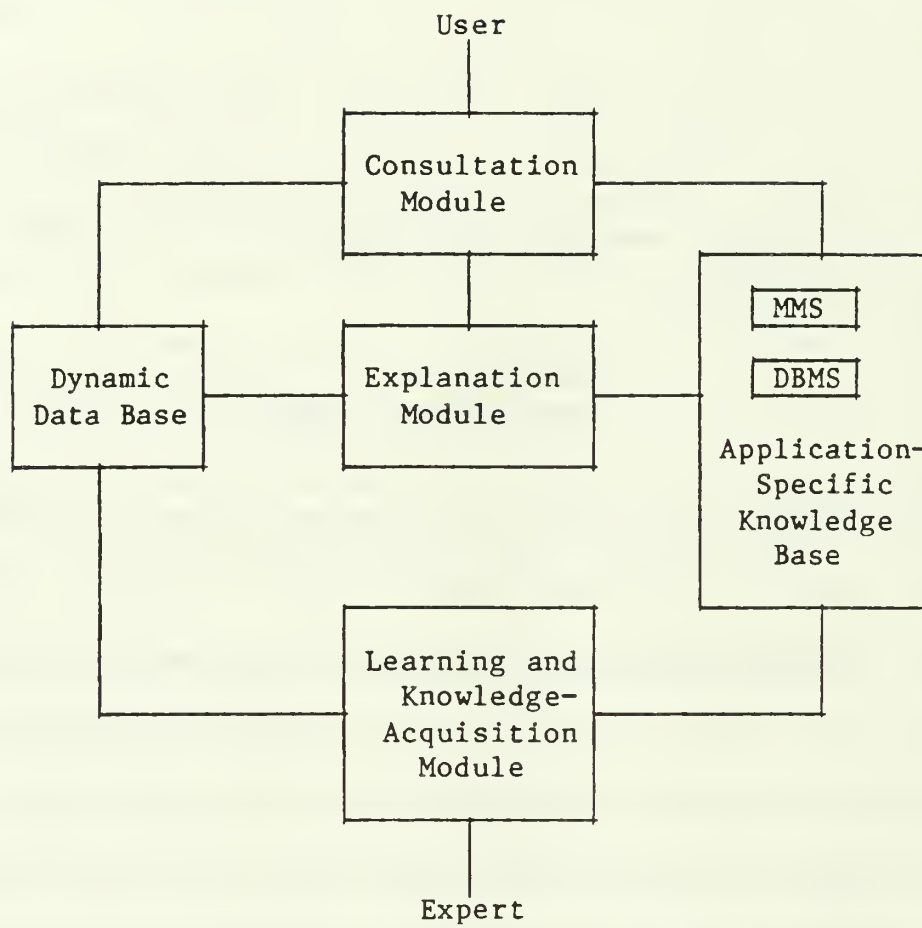


Figure 1

The Organization of MARBLE

usually relies on examining a large amount of historical and pro forma financial information and on judgmental evaluation on the company's market characteristics, industry performance, management competence, and accuracy of the information obtained.

The loan-evaluation decision is traditionally analyzed by statistical linear models, such as regression analysis [22] or multivariate discriminant analysis [13]. As pointed out by Haslem and Longbrake [12] and Kaplan and Dietrich [13], statistical analysis with linear models cannot capture the subjective judgments and the qualitative evaluation so important in the lending decision. In essence, the expert-system approach used by MARBLE is akin to the heuristic simulation method employed by Cohen, Gilmore, and Singer [6]; they both simulate the decision process of loan officers. MARBLE, however, employs production rules as the basic knowledge representation, which has been pointed out as an effective model of the human decision-making process [21]. In addition, the recent knowledge-based technology enables MARBLE to be equipped with uncertainty reasoning, explanation, and incremental refinement capabilities. As will be shown, inductive learning can be applied to enhance further MARBLE's performance by automatically acquiring decision rules for loan classification.

### II.3 Knowledge Acquisition in MARBLE

Knowledge acquisition is the transformation of problem-solving expertise from some knowledge source to a program. Potential sources of knowledge include domain experts, textbooks, and raw data. It is widely recognized that knowledge acquisition is a crucial process in the construction of knowledge-intensive systems because of the difficulty of

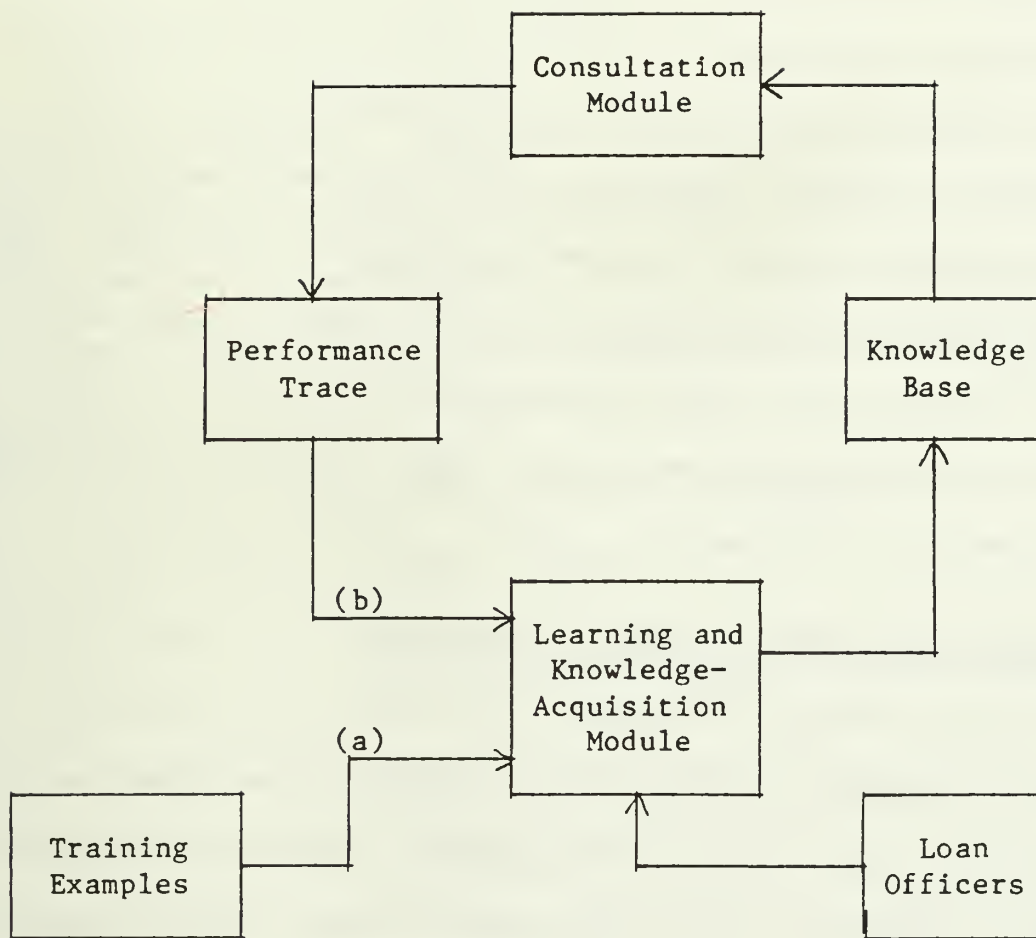
truthfully incorporating all specialized facts, procedures, and judgmental rules about the problem domain [7]. Currently, the knowledge-acquisition process in MARBLE is primarily achieved through learning by being told, which is essentially the transformation of explicit domain knowledge into the representation form used by the expert system, sometimes referred to as "knowledge engineering."

The ability to learn has long been recognized as an essential feature of intelligence. Dietterich et al. [9] categorizes learning methods into four areas: rote learning, learning by being told, learning from examples, and learning by analogy. This paper describes an inductive learning method that would help MARBLE augment its knowledge base through "learning by examples." The use of inductive learning to generate knowledge has been an important area of AI research. The work by Winston [29] pioneered the application of inductive learning to deriving conceptual descriptions for classifying block-world structures. Buchanan and Mitchell [3] developed a rule-learning method for discovering domain-specific knowledge used in inferring chemical structures from mass spectrum. Michalski and Chilausky [18] described PLANT, an expert system for soybean disease diagnosis; they performed an empirical study showing that the combination of learning by being told and learning from examples in PLANT can improve the diagnosis accuracy.

-----  
Insert Figure 2 Here  
-----

The primary application of inductive learning in MARBLE is to determine decision rules from examples of classification decisions done by domain experts (Figure 2). This capability would be very valuable





(a) Learning from Examples

(b) Rule Refinement

Figure 2

Interactions Between the Learning and Knowledge-Acquisition  
Module and Other Components of MARBLE

in credit analysis, where the problem is to extract the classification knowledge from a large amount of historical financial data. The widely used traditional data analysis techniques, such as factor analysis or discriminant analysis, only provide a scoring function with little interpretation. Inductive learning, on the other hand, can help detect interesting conceptual patterns or reveal structure in the data. The other application of machine learning shown in Figure 2 is to refine the decision rules based on past performance; this needs to be done empirically by analyzing the performance trace, and will not be included in the discussion of this paper.

### III. Concept Acquisition by Inductive Learning

#### III.1 Inductive Learning: A Review

Inductive learning can be defined as the process of inferring the description of a class from the description of some individual objects of the class. Each class can be viewed as a concept which is described by a concept recognition rule as a result of inductive learning; if an input data object satisfies this rule, then it represents the given concept. For example, a recognition rule for the concept "good customer" might be:

"A customer whose asset exceeds \$1,000,000.00, total-debt is less than \$250,000.00, and whose annual growth-rate is more than 10%."

Using first-order predicate calculus (FOPC) as the knowledge representation, such a concept can be represented by a conjunction of attribute descriptions:

$$\text{customer}(t) \wedge (\text{asset}(t) > 1,000,000) \wedge (\text{total-debt}(t) < \$250,000) \wedge (\text{AGR}(t) > 0.10) \rightarrow (\text{class}(t) = \text{'GOOD'})$$

An alternative way to represent such a concept is to use the variable-valued logic (VL) proposed by Michalski [16, 17]. The VL language is an extended form of if-then rules where many-valued variables are involved. The premise section of each rule is a conjunction of multivalued attribute variables; each variable is enclosed by a bracket with the corresponding attribute values. The aforementioned concept recognition rule can be represented by the VL formalism as follows:

[assets > \$1,000,000][total-debt < \$250,000]

[AGR > 0.10] → [class : 'GOOD'].

I shall use the VL formalism for knowledge representation throughout this paper for its simplicity and clarity.

In performing concept formation tasks, an induction program is presented with objects, usually consisting of a set of attribute-value pairs as object descriptions. The program is expected to generalize from these examples and derive the common concept so as to accurately classify new objects. Sometimes negative examples--i.e., objects which fail to exhibit the concept--are presented to facilitate the process and to improve the accuracy of the learning. Angluin and Smith [1] used the following specifications to define an inductive inference problem:

- (1) the class of rules being considered;
- (2) the hypothesis space, sometimes referred to as the description space, which consists of a set of concept descriptions such that each rule in the class has at least one description in the hypothesis space;

- (3) for each rule, its set of examples, and the sequences of examples that constitute admissible presentations of the rule;
- (4) the class of inference methods under consideration;
- (5) the criteria for successful inference.

Essentially, inductive learning is an inference process executed in the hypothesis space; this inference process reads in examples and outputs concept descriptions taken from the hypothesis space. The successful implementation of the inference process depends largely on the adequate handling of the following design issues:

1. Organization of the hypothesis space.
2. Representation of the inference rules.
3. The inference method.
4. Criteria for evaluating hypothesis.
5. Criteria for successful inference.

The remainder of this section will give a more detailed look at each of these issues.

### III.2 Hypothesis Space

Since the major step in inductive learning is concerned with the process of generalization, it is useful to organize the hypothesis space in such a fashion that the generalization relation is explicitly represented. The subsumption relation in predicate logic can provide such a structure. The subsumption relation formally represents the relative generality (or specificity, for that matter) between two logic descriptions. If  $A$  and  $B$  are two well-formed-formulas (wffs),  $A$  subsumes  $B$  if and only if there exists a substitution  $\sigma$  such that the descriptions in  $\sigma(A)$  are a subset of those in  $B$ . For example,  $RED-HAIR(x) \ \& \ TALL(x) \ \& \ TEACHER(y)$  subsumes  $RED-HAIR(a) \ \& \ TALL(a) \ \& \ TEACHER(g(a)) \ \& \ ENGLISH(g(a))$



with the substitution  $\sigma = \{a/x, g(a)/y\}$ . If A subsumes B, then A is more general than B, which implies that A can apply in more situations than B. Based on this subsumption relation, the descriptions in the hypothesis space can be placed in a partial order according to the generality of each description. This type of hypothesis space is used in [20], [28] and [17].

Another structure used for organizing the hypothesis space is the decision-tree structure where a node with b branches in the tree represents the corresponding attribute has b different values in the example set. Quinlan [23] and Lee and Ray [14] used decision tree to structure the hypothesis space of their learning programs.

### III.3 Inference Rules

In the general AI problem-solving process, rules are used as state-transformations in the effort to achieve the desired goal, which then provides the solution to the problem. In the same vein, inductive learning can be viewed as a process of transforming initial concept descriptions to intermediate concept description to, ultimately, the inductive concept descriptions. The transformations are achieved by using inference rules.

There are different types of inference rules used in inductive learning. Michalski [17] developed a set of generalization rules to facilitate the searching of the inductive concept descriptions and to guide the movement in the hypothesis space. The AM system described in [15] used roughly 40 heuristic rules to create new concepts; the rules are used to achieve such learning functions as generalization, specialization, permutation of function arguments, and reasoning by analogy.

These types of inference rules serve as "refinement operators" to improve hypothesis.

#### III.4 The Inference Method

The process of inductive learning is often implemented as a heuristic searching procedure [15, 20, 24]. Concept descriptions are derived through a sequence of transformations to generate the goal descriptions in the hypothesis space. Descriptions satisfying the training examples provide the initial condition; negative examples provide constraints to reduce the search space. Because there are enormous amounts of concept descriptions contained in the hypothesis space, successful inference methods often utilize heuristic information to guide the search and bypass unnecessary searching paths.

It is important to choose a representation for the rule space in which generalization can be accomplished by inexpensive operations. Mitchell's version-space algorithm takes advantages of the partial ordering of the hypothesis space. He defines the version space as the set of all concept descriptions that are consistent with all the training examples so far. Initially, the version space is the complete set of possible concepts. The version space is progressively reduced when more training examples are presented. Positive examples force the program to generalize and, consequently, the more specific concept descriptions are eliminated. Conversely, negative examples force the program to specialize, so the more general concept descriptions are removed from consideration. The version space gradually shrink in this manner until only the desired description remains.

Another approach for facilitating the process of generalization is to use the set of negative examples as constraints to rule out undesirable points. The central methodology underlying the learning programs in [17, 18, 19] is based on the concept of a star. A star of the example  $e$  against the set  $E$ , denoted  $G(e/E)$ , is defined as the set of all maximally general expressions that satisfy the positive example  $e$  and that do not satisfy any of the negative examples in  $E$ . This methodology essentially decomposes the problem of finding a complete description of a concept into subproblems with each subproblem aimed at finding the star that covers one positive example but none of the negative examples. In the process of generating the stars, the descriptions can be generalized and simplified by the aforementioned transformation rules or refinement operators. The rule-learning algorithm described in Section IV is based on this star methodology.

As previously stated, a classification rule can be represented in the form of a decision tree. The inference procedure to form classification rules in this context is then the construction of decision trees. Typically, the decision tree can be generated by a branch-and-bound procedure and a branching criterion is needed to determine the attribute-value to be included in the rule. [23] and [14] employed a branching criterion based on the expected information content of each node. The information content of a node is measured by  $-p^+ \log_2 p^+ - p^- \log_2 p^-$ , where  $p^+$  is the proportion of positive examples and  $p^-$  is the proportion of negative examples. The algorithm selects the next attribute to include in the rule based on the principle of maximizing expected information gain.

### III.5 Criteria for Evaluating Hypothesis

In the searching of inductive concept descriptions, the movement in the hypothesis space needs to be guided by some criteria which establish a basis for evaluating the hypothesis. Such criteria are used in selecting the most promising concept among a group of candidates. The possible criteria include:

(a) Simplicity of hypothesis. This criterion states that, while everything else being equal, the hypotheses (a concept description) with the least number of attributes should be selected.

(b) The set-theoretical goodness of fit. This criterion stems from the research in language learning. It states that given the example set  $S$ , the best learned concept description should satisfy every element in  $S$  and as few additional elements as possible.

(c) The decision theoretical measure. Based on Bayes' Theorem, this criterion looks for a hypothesis that has the maximal conditional probability given the set of examples. That is, for the given set of examples  $S$ , the best hypothesis  $h$  maximizes  $\Pr(h/S)$ . Since by Bayes' Theorem,  $\Pr(h/S) = (\Pr(h) \cdot \Pr(S/h))/(\Pr(S))$ , the criterion essentially is to maximize  $\Pr(h) \cdot \Pr(S/h)$ .

An interesting observation is that criterion (c) can combine both criteria (a) and (b). That is,  $\Pr(h)$  can be evaluated by the simplicity of  $h$  and  $\Pr(S/h)$  can be measured by the goodness-of-fit of  $h$  to the set of examples  $S$ --in other words, higher  $\Pr(h)$  means simpler hypothesis and higher  $\Pr(S/h)$  means better fit of  $h$  to  $S$ .



### III.6 Inference Criteria

The inference criteria are used to evaluate the inference procedure to derive inductive concept descriptions. Two such criteria are especially important: (a) the convergence of the inference procedure and (b) the solution quality of the inference procedure as measured by "completeness" and "consistency" of the procedure.

(a) Convergence. This criterion is typically used in the theoretic study of language learning [1]. Conceptually, suppose an inference procedure is run on a large collection of examples with the examples presented in a sequence. The inference procedure is said to converge correctly if it always derives the correct rules after some finite number of iterations.

(b) Completeness and consistency. Inductive learning of concepts is essentially a process of generalization such that the resulting concept description for each class can correctly describe the individual examples of that class; the description is typically a conjunction of attribute-value pairs shared by all objects in the class. The completeness condition says that the concept description generated by the inductive learning process must correctly describe all positive examples; the consistency condition states that the concept description generated must not describe any of the negative examples.

### IV. An Inductive Learning Algorithm

In terms of algorithmic design, the process of inductive learning for multiple concepts begins with the separation of positive and negative decision examples among the whole set of training examples. Let

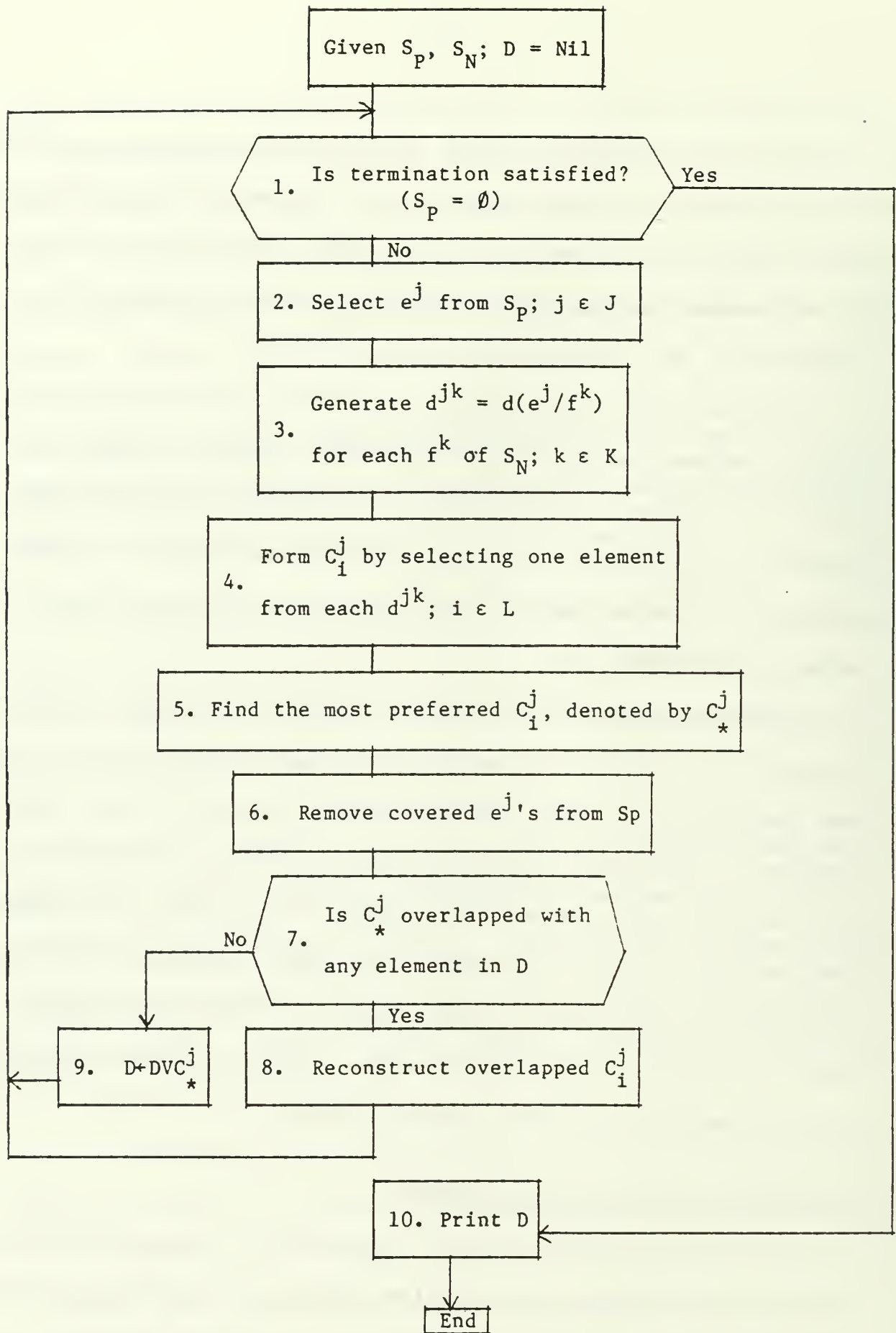


Figure 3

the set of positive examples be  $S_P$  and the set of negative examples be  $S_N$ , the goal of the algorithm is then to determine a conjunction of attribute values as the concept description that satisfies the completeness condition, the consistency condition, and the criterion of the induction.

The learning program would iteratively choose an element  $e^j$  in  $S_P$  and, for every element  $f^k$  in  $S_N$ , generate a discriminant  $d(e^j/f^k)$ , or  $d^{jk}$  for short. Mathematically, let

$$e^j = a_1^j \cdot a_2^j \cdot \dots \cdot a_i^j = \bigwedge_{i=1,n} a_i^j \text{ and}$$

$$f^k = f_1^k \cdot f_2^k \cdot \dots \cdot f_i^k = \bigwedge_{i=1,n} f_i^k,$$

where  $a_i^j$  and  $f_i^k$  are attributes in  $e^j$  and  $f^k$ , respectively.

Then  $d^{jk} = d(e^j/f^k) = \bigwedge_{i \in Q} a_i^j$ , where  $Q = \{i : a_i^j \neq f_i^k\}$  and  $d^{jk} =$

$\bigwedge_{i=1, \ell_{jk}} d_i^{jk}$ . That is,  $d(e^j/f^k)$  is a conjunction of attribute values that can be described by  $e^j$  but not  $f^k$ .

Next, the program will generate a set of all consistent complexes  $C^j$  associated with  $e^j$ ; each element  $C_i^j$  in  $C^j$  is a conjunction of attributes not described by any of the negative examples in  $S_N$ . Algorithmically,  $C_i^j$  is generated by taking an attribute out of each  $d^{jk}$ , for  $k = 1, \dots, n$ , and form a conjunction, that is,

$$C^j = \{(\bigwedge_{\ell=1,n} C_{i\ell}^j) : C_{i1}^j \in d_1^{jk}, C_{i2}^j \in d_2^{jk}, \dots, C_{in}^j \in d_n^{jk}\}.$$

Thus, there are a total of  $\ell_{j1} \cdot \ell_{j2} \cdot \dots \cdot \ell_{jn} = \prod_{k=1,n} \ell_{jk}$  consistent complexes for  $e^j$  (remember that  $\ell_{jk}$  is the number of attributes contained

in the discriminant  $d^{jk}$ ). Each of the complexes is consistent, since it does not cover any negative example.

The induction criterion should then be applied to choose the best complex  $C_{*}^j$  in  $C_i^j$ 's based on a utility function,

$$f(C_i^j) = w_1 P_1 + w_2 P_2 + \dots + w_r P_r,$$

where  $P_1, P_2, \dots, P_r$  are the induction criteria chosen by the user,  $w_1, w_2, \dots, w_r$  show the degree of importance the user gives to the preference criterion.  $C_{*}^j$  is chosen by selecting the highest  $f(\cdot)$  value.

For example, the induction criterion--"to satisfy as many positive examples as possible while not covering any of the negative examples" can be translated to the utility function:

$$\text{MAX } f(C_i^j) = N_P - W * N_N,$$

where  $N_P$  is the number of positive examples satisfied by  $C_i^j$  and  $N_N$  is the number of negative examples that can describe  $C_i^j$ .  $W$  is a very large number used to discourage  $N_N$  from taking any positive value.

Positive examples covered by  $C_{*}^j$  will be removed from  $S_P$ , and the same procedure will be applied to the remaining  $S_P$  and the original  $S_N$  again until all positive examples,  $e^j$ 's, are covered. All the complexes thus produced will be combined to form a complete disjunctive description which covers every positive examples. This algorithm is described by the flowchart shown in Figure 3.

-----  
insert Figure 3 here  
-----



Name	Description	Type
F1	the rating of management competence	nominal domain = {high, average, marginal, reject}
F2	the outside credit rating	nominal domain = {high, average, marginal, reject}
Current-assets	using the pro forma balance sheet, the amount of current current assets	linear
Net-worth	the amount of net worth	linear
Total-debt	the amount of total debt	linear
Funds	the funds for debt service to funds provided operations (three years average)	linear
Cash	the amount of cash	linear
Current-liabilities	the amount of current current liabilities	linear
Current-inventory	the amount of current inventory	linear
Average-inventory	the amount of three years average inventory	linear
Avg-profits	three-year average of net profits	linear
Past-account-evaluation	the evaluation of past account	structured
Customer-status	the applicant's customer status with the bank	nominal domain = {current, new}
Account-type	the applicant's account type either in this bank or from from other banks	structured

Figure 4. Relevant Attributes for Credit Rating

Decision rules derived from this algorithm satisfy both the completeness and the consistency conditions defined in Section III.1. A simple proof is described here according to the algorithm displayed in Figure 3. The algorithm is complete because the algorithm uses steps 2-5 to generate new complexes to describe uncovered positive examples until every positive example is covered by the disjunctive concept description,  $D$ , produced in step 9. The algorithm is also consistent because the concept is generated based on the discriminants  $d(e^j/f^k)$  in step 3, which ensures that none of the negative examples are covered by the inductive description represented by  $D$ .

#### V. An Example: Applying Inductive Learning in the MARBLE System

I shall use the loan evaluation as an example to illustrate the application of inductive learning in MARBLE. The objective is to determine the risk classification of commercial bank loans. In order to describe the default risk on a given commercial loan, a bank usually would use a five-category classification scheme [13]. Here, for the sake of simplicity, only three classes, represented by I, IA, II, are actually used in the set of training examples. There are a total of nine training examples: customers A, B, C for class I; D, E, F for class IA; and G, H, I for class II. The inductive procedure for learning classification rules can be described as follows:

##### (i) Choosing the relevant attributes for training examples

An initial set of attributes using historical and pro forma financial information are selected to be included in each input data case as training examples. As shown in Figure 4, this set of attributes

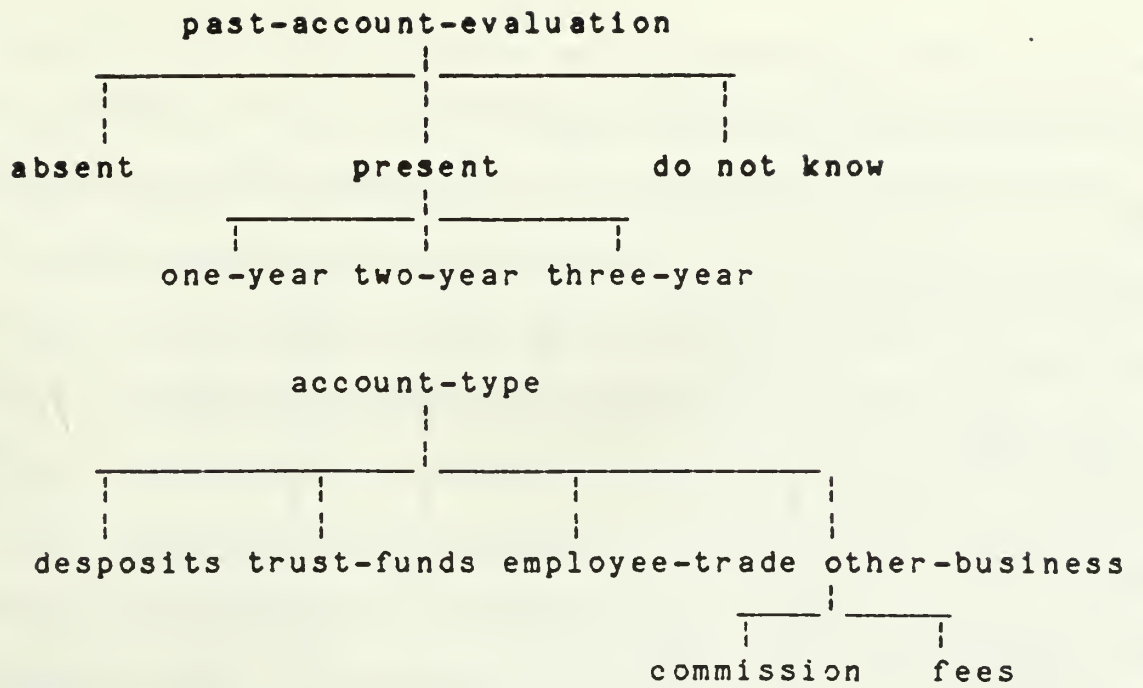


Figure 5. Examples of Structured Attributes

	A	B	C	D	E	F	G	H	I
F1	H	H	H	A	H	A	A	M	A
F2	H	H	A	A	A	A	M	A	A
Current assets	57	39	43	42	38	52	45	37	46
Net worth	57	55	49	37	46	40	38	29	36
Total debt	23	17	20	19	28	25	36	27	35
Funds	9	8	7	8	9	6	-9	7	5
Cash	4	3	5	6	4	5	6	6	5
Cur. Liability	39	28	47	55	39	45	57	53	57
Inventory	21	15	18	12	14	11	7	13	14
Avg inventory	9	14	11	6	6	5	3	5	6
Avg-profits	12	15	13	8	9	9	9	8	-0.8
Past-acc-eval	1Y	2Y	3Y	2Y	1Y	1Y	3Y	2Y	NA
Cust-status	C	C	N	C	C	N	N	C	C
Account-type	C	E	D	D	T	E	E	T	T

Figure 6. Data of 9 Customers  
(all figures in \$1,000)



includes nominal, linear, and structured attributes. In the more traditional data analysis techniques, such as regression or discriminant analysis, only linear and nominal attributes can be considered. The ability to process structural information constitutes one of the advantages of symbolic processing (as characterized by most AI programs) over numerical calculation (as characterized by statistical analysis). The domain of each structured attribute usually can be represented by a hierarchy of attribute values, corresponding to a generalization tree. Two structured attributes used in this example are shown in Figure 5. The tree structure will be used to apply appropriate generalization rules in the induction process.

(ii) Specifying the data description and the domain-specific knowledge

After choosing the relevant attributes, a set of data descriptions  $\{e_k^i\}$ ,  $i = 1, 2, 3$ , and the corresponding class  $\{K_i\}$ . ( $K_1 = I$ ,  $K_2 = IA$ ,  $K_3 = II$ ) are used as training examples (Figure 6).

-----  
insert Figures 4, 5, and 6 here  
-----

The domain-specific knowledge, represented by the generalization trees, can be specified by the following transformation rules:

R1:

[past-account-eval = one-year]V[past-account-eval = two-year]V  
[past-account-eval = three-year]  $\rightarrow$  [past-account-eval = present];

R2:

[account-type = commission]V[account-type = fees]  
 $\rightarrow$  [account-type = other-businesses].

(iii) Forming a induction criterion

The induction criteria are (1) to maximize the number of positive examples covered, while not covering any of the negative examples, and (2) to include the least number of attributes.

(iv) Applying the inductive learning algorithm

To derive the classification decision rule for class I, we start with the first positive example, corresponding to customer A, in class I.

Step 1

--The program produces the discriminant  $d^{jk}$  of customer A against each negative examples one by one, starting with customer D in the negative example set. This process generates the following conjunctive description:

[F1=H][F2=H][current-assets >\$42,000][net-worth >\$37,000]  
[total-debt >\$19,000][funds >\$8,000][cash <\$6,000]  
[cur-liability <\$55,000][inventory >\$12,000]  
[avg-inventory >\$6,000][avg-profits >\$8,000][past-acc-eval <2Y]  
[account-type =C].

--Repeat the same procedure to the remaining negative examples, against customer E:

[F2=H][current-assets >\$38,000][net-worth >\$46,000]  
[total-debt <\$28,000][inventory >\$14,000][avg-inventory >\$6,000]  
[avg-profits >\$9,000] [account-type =T]

against customer F:

[F1=H][F2=H][current-assets >\$52,000][net-worth >\$40,000]  
[total-debt <\$25,000][funds >\$6,000][cash <\$5,000]  
[cur-liability <\$45,000][inventory >\$11,000]  
[avg-inventory >\$5,000][avg-profits >\$9,000]

[past-acc-eval = present][cust-status =N] [account-type =E]

against customer G:

[F1=H][F2=H][current-assets >\$45,000][net-worth >\$38,000]

[total-debt <\$36,000][funds >\$0][cash <\$6,000]

[cur-liability <\$57,000][inventory >\$7,000]

[avg-inventory >\$3,000][avg-profits >\$9,000][past-acc-eval = 1Y]

[cust-status =C][account-type =C]

against customer H:

[F1=H][F2=H][current-assets >\$37,000][net-worth >\$29,000]

[total-debt <\$27,000][funds >\$7,000][cash <\$6,000]

[cur-liability <\$53,000][inventory >\$13,000]

[avg-inventory >\$5,000][avg-profits >\$8,000][past-acc-eval =1Y]

[account-type =T]

against customer I:

[F1=H][F2=H][current-assets >\$46,000][net-worth >\$36,000]

[total-debt <\$35,000][funds >\$5,000][cash <\$5,000]

[cur-liability <\$57,000][inventory >\$14,000]

[avg-inventory >\$6,000][avg-profits >\$0][past-acc-eval =1Y]

[account-type =C].

Generalization rules as the extension-against rule and the climbing generalization tree rule have been applied in the foregoing process in deriving these discriminants.

## Step 2

--Form a set of complexes  $C^j$ s by taking an attribute out of each discriminant  $d^{jk}$  generated in Step 1.

For example, by taking out the first attribute in each discriminant generated above and alternatively taking the attribute of the discriminant against customer I,  $d^{AI}$ , the following 13 complexes ( $C_i^j$ s) are generated:

- (1) [F1=H][F2=H][F1=H][F1=H][F1=H][F1=H],
- (2) [F1=H][F2=H][F1=H][F1=H][F1=H][F2=H],
- (3) [F1=H][F2=H][F1=H][F1=H][F1=H][current-assets >\$46,000],
- .
- .
- .
- . . .
- (13) [F1=H][F2=H][F1=H][F1=H][F1=H][account-type=C]

Another example of the complex generated in this step would be

- (i) [avg-inventory >\$6,000][net-worth >\$46,000]
- [avg-inventory >\$5,000][avg-inventory >\$3,000]
- [net-worth >\$29,000][net-worth >\$36,000],
- .
- .
- .

This process continues until all the consistent complexes are generated.

These complexes can be simplified by removing redundant components or applying generalization rules. For example, complex (3) can be simplified to [F1=H][F2=H][current-assets > \$46,000] and complex (i) can be generalized to [avg-inventory  $\geq$ \$7,000] [net-worth  $\geq$ \$47,000] by applying the closing interval rule described in [17] (note that the unit of input data is \$1,000).

### Step 3

--Choose the best complex description from the set of complexes generated in Step 2 according to the preference criterion. The



preference criteria used in the example include (1) to maximize the number of positive examples covered, (2) to minimize the number of negative examples covered, and (3) to minimize the number of attributes. Then the complex selected is

[avg-inventory  $\geq$  \$7,000][net-worth  $\geq$  \$47,000],

which covers customers A, B, and C in the set of positive examples.

#### Step 4

--Remove the covered positive examples from the list of positive examples by the resulting description in Step 3, and apply the algorithm to the remaining positive examples. Since, in this case, all the positive examples have been covered by the single complex, the decision rule for class I is

[avg-inventory  $\geq$  \$7,000][net-worth  $\geq$  \$47,000]  $\rightarrow$  [class = I].

The same procedure produces the following decision rule for class IA:

[\$37,000  $\leq$  net-worth  $\leq$  \$48,000][inventory  $>$  \$8,000]

$\rightarrow$  [class = IA];

and for class II

[F1=H,A][total-debt  $\geq$  \$26,000]

$\rightarrow$  [class = II].

For the given set of training examples, the three classification rules thus generated covered all the positive examples but none of the negative examples, i.e., the induction process is both complete and consistent. These decision rules not only can be used for credit classification, each of the rules is also a description of a "concept" learned from observing the classification examples. The set of decision rules generated by the inductive learning program can then be added to

the expert system. For this example, the three decision rules just generated can then be stored in MARBLE as follows:

1. PREMISE: (\$AND (GREATERQ\* (VAL1 CNTXT AVG-INVENTORY) 7,000)  
(GREATERQ\* (VAL1 CNTXT NET-WORTH) 47,000));  
ACTION: (DO-ALL (CONCLUDE CNTXT CLASS-TYPE I TALLY 1000))
2. PREMISE: (\$AND (BETWEEN\* (VAL1 CNTXT NET-WORTH) 37,000 48,000)  
(GREATERQ\* (VAL1 CNTXT INVENTORY) 8,000))  
ACTION: (DO-ALL (CONCLUDE CNTXT CLASS-TYPE IA TALLY 1000)); and
3. PREMISE: (\$AND (\$OR (\$AND (SAME CNTXT F1 H)  
(SAME CNTXT F1 A)))  
(GREATERQ\* (VAL1 CNTXT TOTAL-DEBT) 26,000))  
ACTION: (DO-ALL (CONCLUDE CNTXT CLASS-TYPE II TALLY 1000)).

## V. Other Important Aspects of Inductive Learning

### V.1 Probabilistic Learning

In inductive learning, if the training examples are highly distinct in the attribute space and occur in separable clusters, then the generalization procedure may result in concepts which are unambiguous in characterizing the classes. However, in real life, the training examples given as input data are usually not perfect but, rather, are contaminated by a variety of "noise," thus causing errors. In the case of business loan evaluation, for example, the noise may be caused by incorrect financial information or inconsistent granting decisions made by loan officers. The inductive learning method just described would derive generalized descriptions even from the erroneous examples and generate decision rules accordingly. To account for the possible noise in input

examples, the learning system should be able to recognize the imperfection of the data and exploit the converging evidence for the conceptual descriptions. An easy approach to resolve this problem is to relax the preference criterion specified for the induction procedure so that the description of a class is allowed to cover some negative examples under a specified limit. Alternatively, rather than taking a deterministic view, the inductive learning system can derive concept descriptions probabilistically.

Most of the AI inductive learning research to date has been focussed on the deterministic aspect, although uncertainty reasoning has always been an important issue in designing expert systems. The research efforts in [10, 14, 24, 27] represent some recent developments dealing with probabilistic learning. The probabilistic learning system (PLS) developed by Rendell [24] adopted an information theoretical model to determine the probability associated with each candidate concept description. PLS uses a splitting algorithm which repeatedly dichotomizes the attribute space (sometimes referred to as the feature space) into smaller cells based on a dissimilarity measure. The splitting process continues until the training examples in each cell are as homogeneous as possible. The most relevant conceptual description can then be derived. The probabilistic rule generator (PRG) developed by Lee and Ray uses a modified branch-and-bound strategy to extract relevant attribute/value pairs for inclusion in the concept description. At each node of the search tree, PRG uses the information entropy as an evaluation function for selecting the branching attributes. An interesting aspect of PRG is that it can progressively select

the most relevant concept for configuring the decision rules, and thus is shown to be computationally more efficient than prior inductive learning methods [17].

Theoretically, a learning system that derives decision rules probabilistically should be, in general, more efficient computationally than the deterministic counterpart because the completeness and consistency conditions (Section III) are relaxed; instead, a probabilistic learning system carries out statistical tests to determine the significance of various concept descriptions. This computational advantage becomes important when the amount of data is getting larger. If a deterministic approach such as the one described in this paper is used, the set of training examples needs to be preprocessed so that the most representative examples are extracted to serve as input to the induction algorithm.

## V.2 Conceptual Clustering

As pointed out by Clancey [5], classification is a process critical to most problem-solving processes. The method for learning from examples described in Sections III and IV is concerned with forming conceptual descriptions for a set of predetermined classes, assuming that a set of data/class examples has been supplied by an external source (i.e., a domain expert). A different type of inductive learning can be achieved through "learning from observation," where the input data set consists of data cases without any associated classification. This type of learning process would create classes (clusters) among the data cases and then generate conceptual descriptions to characterize



each class. Such a machine-learning method is sometimes referred to as conceptual clustering [4, 11, 19].

In general, clustering is a procedure applied to classifying a set of data into mutually exclusive groups such that the data in the same group are similar while data of different groups are dissimilar to each other. Unlike the numerical taxonomy methods previously used for clustering, however, the conceptual clustering method characterizes each derived class by a conjunction of conceptual description, thus providing better interpretation of the classes. Moreover, as opposed to using a single similarity measure, conceptual clustering is also characterized by its ability to take into account nominal and structural attributes as well as linear numerical attributes.

Conceptual clustering can be used for knowledge acquisition in decision-support situations where the domain expert is either hard to find or is unreliable. In talking to several managers of commercial banks, for example, I found a general feeling existing among them that there are no agreed-upon criterion for classifying loan applications and the decision is often ad hoc. Thus, using historical loan data and the associated granting decision as training examples does have its shortcoming in that the input data may not be adequate examples for the classification decision. By contrast, conceptual clustering provides a method for grouping the input data into classes based on the inherent characteristics of the data. A conjunctive conceptual description is then generated to characterize each class, as to what can be achieved by the method described in Sections III and IV. Thus, conceptual clustering can be viewed as "learning without a teacher."

## VI. Conclusion

Features such as explanation ability, heuristic inference, reasoning with uncertainty, and capabilities for incremental refinement make the knowledge-based expert system an effective tool for decision support. In this paper I have described another aspect in designing such knowledge-based decision support systems: an inductive learning method that can help automate the knowledge-acquisition process and generate decision rules.

Although the inductive learning process described in this paper is primarily related to classification, because classification is a plausible paradigm for human problem solving, the inductive learning method is, therefore, applicable to general decision support problems as well. Moreover, inductive learning can help discover the structure and concepts associated with the data, enabling the decision support system continuously to refine its knowledge base. As an extension to this work, I am currently working on learning probabilistic rules and the integration of learning from examples with conceptual clustering.

REFERENCES

- [1] Angluin, D. and Smith, C., "Inductive Inference: Theory and Methods," Computing Surveys, Vol. 15, No. 3, (1983) pp. 237-269.
- [2] Bonczek, R. H., Holsapple, C. W. and Whinston, A. B., "Future Directions for Developing Decision Support Systems," Decision Science, Vol. 11, (1980), pp. 616-31.
- [3] Buchanan, G. B. and Mitchell, T. M., "Model-Directed Learning of Production Rules," Pattern-Directed Inference Systems, in Waterman, D., and Hayes-Roth, F. (eds.) (New York: Academic Press, 1978).
- [4] Cheng, Y. and Fu, K. S., "Conceptual Clustering in Knowledge Organization," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-7, No. 5 (September 1985).
- [5] Clancey, W. J., "Heuristic Classification," Artificial Intelligence, Vol. 27, (1985), pp. 289-350.
- [6] Cohen, K. J., Gilmore, T. C., and Singer, F. A., "Bank Procedures for Analyzing Business Loan Applications," Cohen, K. J., Hammer, F. S. (eds), in Analytical Methods in Banking, (1966), pp. 219-49.
- [7] Davis, R., "Interactive Transfer of Expertise: Acquisition of New Inference Rules," Artificial Intelligence, Vol. 12 (1979), pp. 121-57.
- [8] Dietterich, T., "Description of Inductive Program INDUCE 1.1," Technical report, (Urbana-Champaign: Department of Computer Science, University of Illinois, October 1978).
- [9] Dietterich, T. G., Lonclon, R., Clarkson, K., and Dromey, R., "Learning and Inductive Inference," D. Cohen and E. Feigenbaum (eds) in Handbook of Artificial Intelligence, (Los Altos: Kaufman, 1981), pp. 323-525.

- [10] Duda, R. O., Hart, P. E., and Nilsson, N. J., "Subjective Bayesian Methods for Rule-Based Inference Systems," in Proc. of the 1976 National Computer Conference, AFIPS Press, (1976).
- [11] Fisher, D. and Langley, P., "Approaches to Conceptual Clustering," Proceeding of Ninth International Joint Conference of Artificial Intelligence, (1985), pp. 691-97.
- [12] Haslem, J. A. and Longbrake, W. A., "A Credit Scoring Model for Commercial Loans, A Comment," Journal of Money, Credit and Banking, Vol. , (August 1972), pp. 733-34.
- [13] Kaplan, R. S. and Dietrich, J. R., "Empirical Analysis of the Commercial Loan Classification Decision," Accounting Review, Vol. 58, No. 1 (January 1982), pp. 18-38.
- [14] Lee, W. and Ray, S., "Probabilistic Rule Generator," Report No. U10CDCS-R-86-1263, (Urbana-Champaign: University of Illinois, Department of Computer Science, 1986).
- [15] Lenat, D. B., "AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search," Ph.D. dissertation, Stanford University, (1976).
- [16] Michalski, R. S., "Pattern Recognition as Rule-Guided Inductive Inference," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 2, (1980), pp. 249-361.
- [17] Michalski, R., "A Theory and Methodology of Inductive Learning," in Michalski, R., Carbonell, J., and Mitchell, T., (eds.), Machine Learning, (Palo Alto: Tioga, 1983).
- [18] \_\_\_\_\_ and Chilausky, R. L., "Learning by Being Told and Learning from Examples: An Experimental Comparison of the Two

- Methods of Knowledge Acquisition in the Context of Developing an Expert System for Soybean Disease Diagnosis," Policy Analysis and Information Systems, Vol. 4, No. 2, (June 1980), pp. 125-60.
- [19] Michalski, R. S. and Stepp, R., "Learning from Observation: Conceptual Clustering," in Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (eds), Machine Learning, (Palo Alto: Tioga, 1983).
- [20] Mitchell, T., "Generalization as Search," Artificial Intelligence, Vol. 18, (1982) pp. 203-26.
- [21] Newell, A. and Simon, H. A., Human Problem Solving (Englewood Cliffs: Prentice-Hall, 1972).
- [22] Orgler, Y. E., "A Credit Scoring Model for Commercial Loans," Journal of Money, Credit and Banking, (November 1970), pp. 435-45.
- [23] Quinlan, J. R., "Discovering Rules from Large Collection of Examples: A Case Study," in Michie, D. (ed.) Expert Systems in the Micro Electronic Age, (Edinburgh: University Press, 1979).
- [24] Rendell, L., "A New Basis for State-Space Learning Systems and a Successful Implementation," Artificial Intelligence, Vol. 20, (1983) pp. 369-92.
- [25] Shaw, M., "A Knowledge-Based Framework for Distributed Decision Support," Simulation and Modeling, Vol. 16, in W. Vogt and M. Mickle, (eds), (The Instrument Society of America, 1985).
- [26] \_\_\_\_\_ and Tu, P. L., "An Integrated Framework for Knowledge-Based Decision Support," Working Paper, University of Illinois, Department of Business Administration, (1986).



- [27] Shortliffe, E. H. and Buchanan, B. G., "A Model of Inexact Reasoning in Medicine," Mathematical Biosciences, No. 23 (1975), pp. 351-79.
- [28] Vere, S. A., "Induction of Concepts in the Predicate Calculus," Fourth International Joint Conference on Artificial Intelligence, Tbilisi, USSR, p. 281-87, (1975).
- [29] Winston, P. H., "Learning Structural Descriptions From Examples," in The Psychology of Computer Version, McGraw-Hill, New York, (1975).

## Appendix 1.

### An Example of MARBLE's question-answering mode

The following is a sample question-answering session with MARBLE, a loan evaluation decision support system.

(1) "What is the customer that is applying for LOAN-1?"

\*\*ASCEM INC.

{\*\* indicated the user input.}

(2) "Is the customer a present customer?"

\*\*YES

(3) "Do you want to do the extensive credit checking on this customer?"

\*\*YES.

(4) "What is the rating of management competence?"

\*\*F7

{F7 is a function key of showing WHY this question is needed.}

---

The rating of management competence is needed to determine the firm's credit-worthiness in relation to a particular loan application

#### RULE073

If 1) 1) the rating of management competence is high, and  
2) the outside credit rating of the firm is high, and  
3) credit rating based on the bank's financial analysis of the applicant's statements is high, or  
2) 1) a rating of management competence is HIGH, and  
2) the outside credit rating of the firm is HIGH, and  
3) credit rating based on the bank's financial analysis of the applicant's statements is AVERAGE, or  
3) 1) a rating of management competence is HIGH, and  
2) the outside rating of the firm is GOOD, and  
3) credit rating based on the bank's financial analysis of the applicant's statements is HIGH, or  
4) 1) a rating of management competence is AVERAGE, and  
2) the outside rating of the firm is HIGH, and  
3) credit rating based on the bank's financial analysis of the applicant's statements is HIGH.  
then 1) it is definite (100%) that the credit-worthiness is high.  
2) it is definite (100%) that extensive credit check is finished, and  
3) Inform the user of this decision.

---

{the user can input the value with some degree of certainty}

\*\*HIGH

\*\*8

(5) "What is the outside credit rating of the firm?"

\*\*F8

{F8 is a HOW function key. The user is able to see how the value of previous parameters have been determined.}

---

HOW did the system determine the value of  
MANAGEMENT-COMPETENCE OF LOAN-1

CUSTOMER-STATUS OF LOAN-1

CUSTOMER-NAME OF LOAN-1

{If the user choose management-competence, the following appears:}

MANAGEMENT-COMPETENCE was determined to be HIGH with certainty 80% by user input.

---

\*\*HIGH

(6) What is the type of loan?

\*\*TERM

(7) What is the firm's 3-year average net profits?

\*\*90,000,000

(8) What is the firm's trend in net profits?

\*\*F4

{F4 is an UNKNOWN function key}

(9) What is the industry median ratio of pretax profits to total tangible assets divided by prime rate of interest?

\*\*2

(10) What is the firm's ratio of pretax profits to total tangible assets divided by prime rate of interest?

\*\*2

(11) What is the firm's percentile in industry ratio of net profits to tangible net worth?

\*\*0.8

(12) What is the firm's percentile in industry inventory turnover ratio?

\*\*F4

(13) What is the firm's percentile in industry: net profits?

\*\*F4

(14) If the firm's percentile ratio of pretax profits to total tangible assets exceed prime rate, answer the ratio, otherwise press F4.

\*\*F4

(15) What is the industry median trend in net profits?

\*\*0.2

(16) What is the ratio of cash to total current liabilities?

\*\*0.2

(17) What is the percentile value of the firm's ratio of inventories to current assets?

\*\*0.2

(18) What is the percentile value of the firm's current ratio?

\*\*0.8

(19) What is the proposed loan amount?

\*\*10,000,000

(20) What is the legal loan limit?

\*\*120,000,000

(21) What is the maximum loan limit determined by either the firm's average deposits during the past year or during the past three years?

\*\*100,000,000

(22) What is the ratio of proposed loan to firm's total assets?

\*\*0.8

(23) What is the purpose of the loan?

\*\*PLANT AND EQUIPMENT

(24) What is the proposed maturity of the loan?

\*\*14

(25) Is the firm in a fully secured basis?

\*\*NO

(26) THE LOAN IS GRANTED WITH CERTAINTY FACTOR = 72%

## Appendix 2

In MARBLE, production rules are the basic form of knowledge representation. Rules can be subject to categorization in accordance with the context-types for which they are most appropriately invoked. For example, some rules deal with profitability, some with solvency, and still others deal with loan evaluation. The grammar of the rules, described by the BNF formalism, is shown in Table 1.

<pre>&lt;rule&gt; :: = &lt;premise&gt; &lt;action&gt;  &lt;premise&gt; :: = (\$AND &lt;condition&gt; ... &lt;condition&gt;)  &lt;condition&gt; :: = (&lt;func1&gt; &lt;context&gt; &lt;parameter&gt;)                   (&lt;func2&gt; &lt;context&gt; &lt;parameter&gt; &lt;value&gt;)                   (\$OR &lt;condition&gt; .. &lt;condition&gt;)  &lt;action&gt; :: = &lt;conclusion&gt;   &lt;actfunc&gt;                 (DO-ALL &lt;conclusion&gt; ... &lt;conclusion&gt;)                 (DO-ALL &lt;actfunc&gt; &lt;actfunc&gt; &lt;actfunc&gt;)  &lt;conclusion&gt; :: = (&lt;confunc&gt; &lt;context&gt; &lt;parameter&gt; &lt;value&gt; TALLY &lt;cf&gt;)</pre>
---

Table 1

To capture fully the decision rules used in business loan evaluation, MARBLE currently uses eight different context-types in its knowledge base:

LOAN	The loan application
EVALUATION	An evaluation of a new customer relationship, and
FEASIBLE	A feasibility appraisal



## Appendix 2 (cont'd)

RECOMMEND Detailed recommendations:

- S1 The credit-worthiness of the firm in relation to the proposed loan,
- S2 The indication of the extent that the customer will build the bank,
- S3 The evaluation of the expected profitability to the bank of a customer relationship with the firm.

PROFITABILITY The expected profitability of the firm:

SOLVENCY The expected solvency ability of the firm

The context-types instantiated during the consultation session are arranged hierarchically in a data structure termed the context tree, as shown in the following figure:

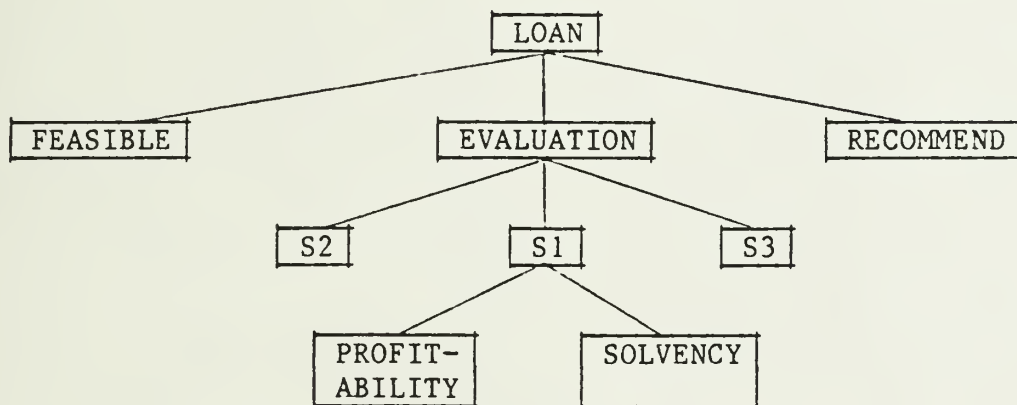


Figure A.1

The context tree helps to structure a knowledge base domain by allowing the knowledge engineer to separate a large amount of information of knowledge into logical entities. Each context can solve one part of the total problem and provide important information needed to solve the problem as a whole.







HECKMAN  
BINDERY INC.



**JUN 95**

Bound-To-Pleas<sup>®</sup> N. MANCHESTER,  
INDIANA 46962



UNIVERSITY OF ILLINOIS-URBANA



3 0112 005699951