Data Mining and Knowledge Discovery: A Review of Issues and a Multistrategy Approach

Ryszard S. Michalski and Kenneth A. Kaufman

P97-3

MLI 97-2

March 1997

Data Mining and Knowledge Discovery: A Review of Issues and a Multistrategy Approach[*]

Ryszard S. Michalski and Kenneth A. Kaufman

Abstract

An enormous proliferation of databases in almost every area of human endeavor has created a great demand for new, powerful tools for turning data into useful, task-oriented knowledge. In efforts to satisfy this need, researchers have been exploring ideas and methods developed in machine learning, pattern recognition, statistical data analysis, data visualization, neural nets, etc. These efforts have led to the emergence of a new research area, frequently called data mining and knowledge discovery. The first part of this paper is a compendium of ideas on the applicability of symbolic machine learning methods to this area. The second part describes a multistrategy methodology for *conceptual data exploration*, by which we mean the derivation of high-level concepts and descriptions from data through symbolic reasoning involving both data and background knowledge. The methodology, which has been implemented in the INLEN system, combines machine learning, database and knowledge-based technologies. To illustrate the system's capabilities, we present results from its application to a problem of discovery of economic and demographic patterns in a database containing facts and statistics about the countries of the world. The presented results demonstrate a high potential utility of the methodology for assisting a user in solving practical data mining and knowledge discovery tasks.

Keywords: Multistrategy Learning, Integrated Learning, Knowledge Discovery, Conceptual Clustering, Inductive Generalization. Constructive Induction, Temporal Reasoning, Structured Data

Acknowledgments

This research was done in the Machine Learning and Inference Laboratory of George Mason University. The authors thank their past and current collaborators from the Laboratory, who provided research feedback, comments, and assistance at different stages of the development of many of the ideas and computer programs described in this chapter. In particular, we thank Eric Bloedorn, Mark Maloof, Jim Ribeiro, Janusz Wnek, and Qi Zhang for their participation in these research efforts. Support for the Machine Learning and Inference Laboratory's research activities has been provided in part by the National Science Foundation under Grants No. DMI-9496192 and IRI-9020266, in part by the Office of Naval Research under Grant No. N00014-91-J-1351, in part by the Defense Advanced Research Projects Agency under Grant No. N00014-91-J-1854 administered by the Office of Naval Research, and in part by the Defense Advanced Research Projects Agency under Grants No. F49620-92-J-0549 and F49620-95-1-0462 administered by the Air Force Office of Scientific Research.

[*] This paper is based on the authors' Chapter 2 of *Machine Learning and Data Mining: Methods and Applications*, edited by R.S. Michalski, I. Bratko and M. Kubat, John Wiley & Sons publishers, 1997 (to appear).

1 Introduction

The current information age is characterized by an extraordinary expansion of data that are being generated and stored about all kinds of human endeavors. An increasing proportion of these data is recorded in the form of computer databases, in order that the computer technology may easily access it. The availability of very large volumes of such data has created a problem of how to extract from them useful, task-oriented knowledge.

Data analysis techniques that have been traditionally used for such tasks include regression analysis, cluster analysis, numerical taxonomy, multidimensional analysis, other multivariate statistical methods, stochastic models, time series analysis, nonlinear estimation techniques, and others (e.g., Daniel and Wood, 1980; Tukey, 1986; Morganthaler and Tukey, 1989; Diday, 1989; Sharma, 1996). These techniques have been widely used for solving many practical problems. They are, however, primarily oriented toward the extraction of quantitative and statistical data characteristics, and as such have inherent limitations.

For example, a statistical analysis can determine covariances and correlations between variables in data. It cannot, however, characterize the dependencies at an abstract, conceptual level, and produce a causal explanation of reasons why these dependencies exist. Nor can it develop a justification of these relationships in the form of higher-level logic-style descriptions and laws. A statistical data analysis can determine the central tendency and variance of given factors, and a regression analysis can fit a curve to a set of datapoints. These techniques cannot, however, produce a qualitative description of the regularities and determine their dependence on factors not explicitly provided in the data, nor can they draw an analogy between the discovered regularity and a regularity in another domain.

A numerical taxonomy technique can create a classification of entities, and specify a numerical similarity among the entities assembled into the same or different categories. It cannot, however, build qualitative descriptions of the classes created and hypothesize reasons for the entities being in the same category. Attributes that define the similarity, as well as the similarity measures, must be defined by a data analyst in advance. Also, these techniques cannot by themselves draw upon background domain knowledge in order to automatically generate relevant attributes and determine their changing relevance to different data analysis problems.

To address such tasks as above, a data analysis system has to be equipped with a substantial amount of background knowledge, and be able to perform symbolic reasoning tasks involving that knowledge and the data. In summary, traditional data analysis techniques facilitate useful data interpretations, and can help to generate important insights into the processes behind the data. These interpretations and insights are the ultimate knowledge sought by those who build databases. Yet, such knowledge is not created by these tools, but instead has to be derived by human data analysts.

In efforts to satisfy the growing need for new data analysis tools that will overcome the above limitations, researchers have turned to ideas and methods developed in machine learning. The field of machine learning is a natural source of ideas for this purpose, because the essence of research in this field is to develop computational models for acquiring knowledge from facts and background knowledge. These and related efforts have led to the emergence of a new research area, frequently called data mining and knowledge discovery, (e.g., Michalski, Baskin and Spackman, 1982; Zhuravlev and Gurevitch, 1989; Michalski, 1991; Michalski et al, 1992; Van Mechelen et al, 1993; Fayyad et al, 1996; Evangelos and Han, 1996).

The first part of this paper is a compendium of ideas on the applicability of symbolic machine learning methods to data mining and knowledge discovery. The second part describes a methodology for *conceptual data exploration,* by which we mean the derivation of high-level concepts and descriptions from data. The methodology, stemming mainly from various efforts in machine learning, applies diverse methods and tools for determining task-oriented data characterizations and generalizations. These characterizations are expressed in the form of logic-style descriptions, which can be easily interpreted and used for decision-making. The term *task-oriented* emphasizes the fact that an exploration of the same data may produce different knowledge; therefore, the methodology tries to connect the task at hand with the way of exploring the data. Such task-orientation naturally requires a multistrategy approach, because different tasks may need to employ different data exploration and knowledge generation operators.

The aim of the methodology is to produce knowledge in a form that is close to data descriptions that an expert might produce. Such a form may include combinations of different types of descriptions, e.g., logical, mathematical, statistical, and graphical. The main constraint is that these descriptions should be easy to understand and interpret by an expert in the given domain, i.e., they should satisfy the “principle of comprehensibility” (Michalski, 1993). Our first efforts in developing a methodology for multistrategy data exploration have been implemented in the system INLEN (Michalski et al, 1992). The system combines a range of machine learning methods and tools with more traditional data analysis techniques. These tools provide a user with the capability to make different kinds of data explorations and to derive different kinds of knowledge from a database.

The INLEN methodology for intelligent data exploration directly reflects the aims of the current research on data mining and knowledge discovery. In this context, it may be useful to explain the distinction between the concepts of data mining and knowledge discovery, as proposed in (Fayyad, Piatetsky-Shapiro and Smyth, 1996). According to this distinction, data mining refers to the application of machine learning methods, as well as other methods, to the “enumeration of patterns over the data,” and knowledge discovery refers to the process encompassing the entire data analysis lifecycle, from the identification of data analysis goals and the acquisition and organization of raw data to the generation of potentially useful knowledge, its interpretation and testing. According to these definitions, the INLEN methodology incorporates both data mining and knowledge discovery techniques.

2 Machine Learning and Multistrategy Data Exploration

This section shows a close relationship between ideas and methods developed in the field of machine learning to the goals of data mining and knowledge discovery. Specifically, it describes how methods of symbolic machine learning can be used for automating or semi-automating a wide range of tasks concerned with conceptual exploration of data and a generation of task-oriented knowledge from them. Let us briefly review some of these methods.

2.1 Determining General Rules from Specific Cases

A major class of tools for multistrategy data exploration is based on methods for symbolic inductive learning from examples. Given collections of examples of different decision classes (or cases of a relationship), and problem-relevant knowledge ("background knowledge"), an inductive learning method hypothesizes a general description of each class. Some methods use a fixed criterion for choosing the description from a large number of possibilities, and some allow the user to define a criterion that reflects the problem at hand. A description can be in the form of a set of decision rules, a decision tree, a semantic net, etc. A decision rule can also take on many different forms. Here we will assume the following form:

$$\text{CLASS} \Leftarrow \text{CONDITION}$$

where CLASS is a statement indicating a class, decision, or a concept name to be assigned to an entity (an object or situation) that satisfies CONDITION; CONDITION is a conjunction of elementary conditions on the values of attributes characterizing the objects; and ⇐ denotes implication.

We will also assume that if CLASS requires a disjunctive description, then several such (conjunctive) rules relate to the same CLASS. To illustrate this point, Figure 1 gives an example of a disjunctive description of a class of robot-figures in EMERALD (a large system for demonstrating machine learning and discovery capabilities—Kaufman and Michalski, 1993).

Rule A: ***Class 1*** ⇐ ***Jacket Color is Red, Green or Blue &***
Head Shape is Round or Octagonal

Rule B: ***Class 1*** ⇐ ***Head Shape is Square &***
Jacket Color is Yellow

Figure 1. A two-rule description of Class 1.

To paraphrase this description, a robot belongs to *Class 1* if the color of its jacket is red, green or blue, and its head is round or octagonal, or, alternatively, its head is square and the color of its jacket is yellow.

The EMERALD system, mentioned above, combines five programs that display different kinds of learning capabilities (Kaufman and Michalski, 1993). These capabilities include rule learning from examples (using program AQ15), learning distinctions between structures (INDUCE), conceptual clustering (CLUSTER/2), prediction of object sequences (SPARC), and derivation of equations and rules characterizing data about physical processes (ABACUS). Each of these programs is directly applicable to conceptual data exploration. For example, the rules in Figure 1 were generated by the AQ15 rule module [MMHL86], [HMM86] from a set of "positive" and "negative" examples of Class 1 of robot-figures.

AQ15 learns *attributional* descriptions of entities, i.e., descriptions involving only their attributes. More general descriptions, *structural* or *relational*., also involve relationships among components of the entities, the attributes of the components, and quantifiers. Such descriptions are produced, for example, by the INDUCE module of EMERALD (Larson, 1977; Bentrup, Mehler and Riedesel, 1987). Constructing structural descriptions requires a more complex description language that includes multi-argument predicates, for example, PROLOG, or Annotated Predicate Calculus (Michalski, 1983; Bratko, Muggleton and Karalic, 1997).

For database exploration, attributional descriptions appear to be the most important and the easiest to implement, because typical databases characterize entities in terms of attributes, not relations. One simple and popular form of attributional description is a decision or classification tree. In such a tree, nodes correspond to attributes, branches stemming from the nodes correspond to attribute values, and leaves correspond to individual classes (e.g., Quinlan, 1986). A decision tree can be transformed into a set of decision rules (a ruleset) by traversing all paths from the root to individual leaves. Such rules can be often simplified by detecting superfluous conditions in them (e.g., Quinlan, 1993). The opposite process of transforming a ruleset into a decision tree is not so direct (Imam, 1995), because a rule representation is more powerful than a tree representation. The term “more powerful” means in this context that a decision tree representing a given ruleset may require superfluous conditions (e.g., Michalski, 1990).

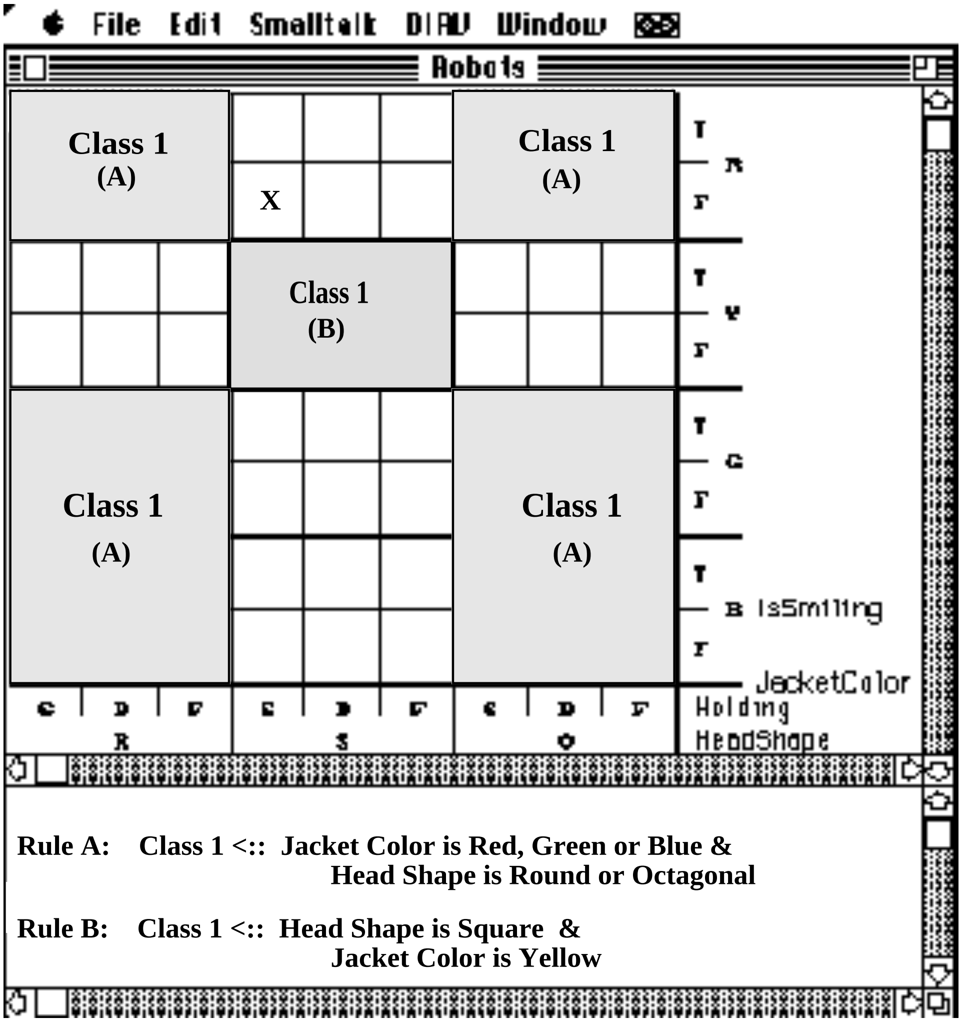
The input to an attributional learning program consists of a set of examples of individual classes and “background knowledge” (BK) relevant to the given learning problem. The examples (cases of decisions) are in the form of vectors of attribute-value pairs associated with some decision class. Background knowledge is usually limited to information about the legal values of the attributes, their type (the scale of measurement), and a *preference criterion* for choosing among possible candidate hypotheses. Such a criterion may refer to, for example, the computational simplicity of the description, and/or an estimate of its predictive accuracy. In addition to BK, a learning method may have a *representational bias*, i.e., it may constrain the form of descriptions to only a certain type of expressions, e.g., single conjunctions, decision trees, sets of conjunctive rules, or DNF expressions.

In some methods, BK may include more information, e.g., constraints on the interrelationship between various attributes, rules for generating higher level concepts, new attributes, as well as some initial hypothesis (Michalski, 1983). Learned rules are usually *consistent* and *complete* with regard to the input data. This means that they completely and correctly classify all the original “training” examples. Sections 5 and 8 present consistent and complete example solutions from the inductive concept learning program AQ15c (Wnek et al, 1995). In some applications, especially those involving learning rules from noisy data or learning *flexible* concepts (Michalski, 1990), it may be advantageous to learn descriptions that are incomplete and/or inconsistent (Bergadano et al, 1992).

Attributional descriptions can be visualized by mapping them into a planar representation of a discrete multidimensional space (a diagram) spanned over the given attributes (Michalski, 1978), (Wnek et al, 1990). For example, Figure 2 shows a diagrammatic visualization of the rules from Figure 1. The diagram in Figure 2 was generated by the concept visualization program DIAV (Wnek et al, 1990; Wnek, 1995).

Each cell in the diagram represents one specific combination of values of the attributes. For example, the cell marked by an X represents the vector: (HeadShape=<u>S</u>quare, Holding=<u>S</u>word, JacketColor=<u>R</u>ed, IsSmiling=<u>F</u>alse). The four shaded areas marked Class1 (A) represent rule A, and the shaded area marked Class 1 (B) represents rule B. In such a diagram, conjunctive rules correspond to certain regular arrangements of cells that are easy to recognize (Michalski, 1978).

The diagrammatic visualization can be used for displaying the *target concept* (the concept to be learned), the training examples (the examples and counter-examples of the concept), and the actual concept learned by a method. By comparing the target concept with the learned concept, one can determine the *error area*, i.e., the area containing all examples that would be incorrectly classified by the learned concept. Such a diagrammatic visualization method can illustrate any kind of attributional learning process (Wnek et al, 1990).



Rule A: **Class 1 <:: Jacket Color is Red, Green or Blue &**
Head Shape is Round or Octagonal

Rule B: **Class 1 <:: Head Shape is Square &**
Jacket Color is Yellow

Figure 2. A diagrammatic visualization of rules from Figure 1.

Two types of data exploration operators can be based on methods for learning concept descriptions from examples:

- Operators for determining general symbolic descriptions of a designated group or groups of entities in a data set. Such descriptions express the common properties of the entities in each group. The operators can use abstract concepts that are not present in the original data via the mechanism of *constructive induction* (see below). These operators are based on programs for learning *characteristic concept description*s.
- Operators for determining differences between different groups of entities. Such differences are expressed in the form of rules that define properties that characterize one group but not the other. These operators are based on programs for learning *discriminant concept descriptions*.

Section 5 will illustrate these two types of descriptions. For more details and their definitions see (Michalski, 1983). Basic methods for concept learning assume that examples do not have errors, that all attributes have a specified value in them, that all examples are located in the same database, and that concepts to be learned have a precise ("crisp") description that does not change over time. In many situations one or more of these assumptions may not hold. This leads to a variety of more complex machine learning and data mining problems:

- *Learning from incorrect data*, i.e., learning from examples that contain a certain amount of errors or noise (e.g., Quinlan, 1990; Michalski, Kaufman and Wnek, 1991). These problems are important to learning from complex real-world observations, where there is always some amount of noise.
- *Learning from incomplete data*, i.e., learning from examples in which the values of some attributes are unknown (e.g., Dontas, 1988; Lakshminarayan et al, 1996).
- *Learning from distributed data*, i.e., learning from separate collections of data that must be brought together if the patterns within them are to be exposed (e.g., Ribeiro, Kaufman and Kerschberg, 1995).
- *Learning drifting or evolving concepts*, i.e., learning concepts that are not stable but changing over time, randomly or in a certain general direction. For example, the "area of interest" of a user is often an evolving concept (e.g., Widmer and Kubat, 1996).
- *Learning concepts from data arriving over time*, i.e., incremental learning in which currently held hypotheses characterizing concepts may need to be updated to account for the new data (e.g., Maloof and Michalski, 1995).
- *Learning from biased data*, i.e., learning from a data set that does not reflect the actual distribution of events (e.g., Feelders, 1996).
- *Learning flexible concepts*, i.e., concepts that inherently lack precise definition and whose meaning is context-dependent; some ideas concerned with this topic include *fuzzy sets* (e.g., Zadeh, 1965; Dubois, Prade and Yager, 1993), *two-tiered concept representations* (e.g., Michalski, 1990; Bergadano et al, 1992), and *rough sets* (e.g., Pawlak, 1991; Slowinski, 1992; Ziarko, 1994).
- *Learning concepts at different levels of generality*, i.e., learning descriptions that involve concepts from different levels of generalization hierarchies representing background knowledge (e.g., Kaufman and Michalski, 1996).
- *Integrating qualitative and quantitative discovery*, i.e., determining sets of equations that fit a given set of data points, and qualitative conditions for the application of these equations (e.g., Falkenhainer and Michalski, 1990).
- *Qualitative prediction,* i.e., discovering patterns in sequences or processes and using these patterns to qualitatively predict the possible continuation of the given sequences or processes (e.g., Davis, 1981; Michalski, Ko and Chen, 1985; 1986; Dieterrich and Michalski, 1986).

Each of these problems is relevant to the derivation of useful knowledge from a collection of data (static or dynamic). Therefore, methods for solving these problems developed in the area of machine learning are directly relevant to data mining and knowledge discovery, in particular, to conceptual data exploration.

2.2 Conceptual Clustering

Another class of machine learning methods relevant to data mining and knowledge discovery concerns the problem of building a conceptual classification of a given set of entities. The problem is similar to that considered in traditional cluster analysis, but is defined in a different way. Given a set of attributional descriptions of some entities, a description language for characterizing classes of such entities, and a classification quality criterion, the problem is to partition entities into classes in a way that maximizes the classification quality criterion, and simultaneously to determine general (extensional) descriptions of these classes in the given description language. Thus, a conceptual clustering method seeks not only a classification structure of entities (a dendrogram), but also a symbolic description of the proposed classes (clusters). An important, distinguishing aspect of conceptual clustering is that, unlike in cluster analysis, the properties of class descriptions are taken into consideration in the process of determining the classes (clusters).

To clarify the difference between conceptual clustering and conventional clustering, notice that a conventional clustering method typically determines clusters on the basis of a similarity measure that is a function solely of the properties (attribute values) of the entities being compared, and not of any other factors:

Similarity(A, B) = f(properties(A), properties(B))

where A and B are entities being compared.

In contrast, a conceptual clustering program clusters entities on the basis of a *conceptual cohesiveness*, which is a function of not only properties of the entities, but also of two other factors: the *description language* L, which the system uses for describing the classes of entities, and of the *environment*, E, which is the set of neighboring examples:

Conceptual cohesiveness(A, B) = f(properties(A), properties(B), L, E)

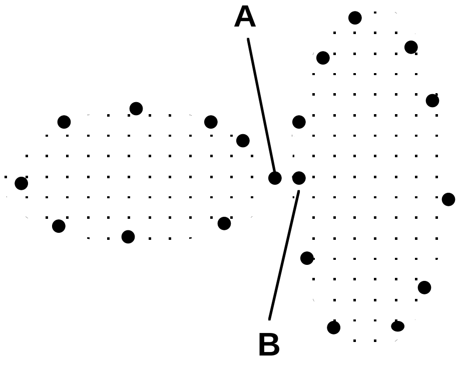


Figure 3. An illustration of the difference between closeness and conceptual cohesiveness.

Thus, two objects may be similar, i.e., close according to some distance (or similarity) measure, while having a low conceptual cohesiveness, or vice versa. An example of the first situation is shown in Figure 3. The points (black dots) A and B are “close” to each other; they therefore would be placed into the same cluster by any technique based solely upon the distances between the points. However, these points have small conceptual cohesiveness due to the fact that they belong to configurations representing different concepts. A conceptual clustering method, if equipped with an appropriate description language, would cluster the points in Figure 3 into two "ellipses," as people normally would.

A classification quality criterion used in conceptual clustering may involve a variety of factors, such as the *fit* of a cluster description to the data (called sparseness), the *simplicity* of the description, and other properties of the entities or the concepts that describe them (Michalski, Stepp and Diday, 1981). An example of conceptual clustering is presented in Section 5.

Some new ideas on employing conceptual clustering for structuring text databases and creating concept lattices for discovering dependencies in data are in (Carpineto and Romano, 1995a; 1995b). The concepts created through the clustering are linked in lattice structures that can be traversed to represent generalization and specialization relationships.

2.3 Constructive Induction

Most methods for learning rules or decision trees from examples assume that the attributes used for describing examples are sufficiently relevant to the learning problem at hand. This assumption does not always hold in practice. Attributes used in the examples may not be directly relevant, and some attributes may be irrelevant or *nonessential*. An important advantage of symbolic methods over statistical methods is that they can relatively easily determine irrelevant or nonessential attributes. An attribute is *nonessential* if there is a complete and consistent description of the classes or concepts to be learned that does not use this attribute. Thus, a nonessential attribute may be either irrelevant or relevant, but will by definition be dispensable. Inductive learning programs such as the rule-learning program AQ, or the decision tree-learning ID3, can cope relatively easily with a large number of nonessential attributes in their input data.

If there are very many nonessential attributes in the initial description of the examples, the complexity of a learning process may significantly increase. Such a situation calls for a method that can efficiently determine the most relevant attributes for the given problem from among all those given initially. Only the most relevant attributes will be used in the description learning process. Determining the most relevant attributes is therefore a useful data exploration operator. Such an operator can also be useful for the data analyst on its own merit, as it may be important to know which attributes are most discriminatory for a given_set of classes. By removing less relevant attributes, the representation space is reduced, and the problem becomes simpler. Thus, such a process can be viewed as a form of improving the representation space. Some methods for finding the most relevant attributes are described in (Zagoruiko, 1972; Baim, 1982).

In many applications, the attributes originally given may be only weakly or indirectly relevant to the problem at hand. In such situations, there is a need for generating new, more relevant attributes that may be functions of the original attributes. These functions may be simple, e.g., a product or sum of a set of the original attributes, or very complex, e.g., a Boolean attribute based on the presence or absence of a straight line or circle in an image (Bongard, 1970). Finally, in some situations, it will be desirable to abstract some attributes, that is, to group some attribute values into units, and thus reduce the attribute's range of possible values. A quantization of continuous attributes is an example of such an operation.

All the above operations—removing less relevant attributes, adding more relevant attributes, and abstracting attributes—are different forms of improving the original representation space for learning. A learning process that consists of two (intertwined) phases, one concerned with the construction of the “best” representation space, and the second concerned with generating the “best” hypothesis in the found space is called *constructive induction* (Michalski, 1978; 1983; Wnek and Michalski, 1994). An example of a constructive induction program is AQ17 (Bloedorn, Wnek and Michalski, 1993), which performs all three types of improvements of the original representation space. In this program, the process of generating new attributes is done by combining initial attributes by mathematical and/or logical operators and selecting the “best” combinations, and/or by obtaining advice from an expert (Bloedorn, Wnek and Michalski, 1993; Bloedorn and Michalski, 1996).

2.4 Selection of the Most Representative Examples

When a database is very large, determining general patterns or rules characterizing different concepts may be very time-consuming. To make the process more efficient, it may be useful to extract from the database the most representative or important cases (examples) of given classes or concepts. Most such cases are those that are either most typical or most extreme (assuming that there is not too much noise in the data). One method for determining the latter ones, the so-called “method of outstanding representatives,” is described in (Michalski and Larson, 1978).

2.5 Integration of Qualitative and Quantitative Discovery

In a database that contains numerical attributes, a useful discovery might be an equation binding these attributes. For instance, from a table of planetary data including planets’ masses, densities, distances from the sun, periods of rotation, and lengths of local years, one could automatically derive Kepler’s Law that the cube of the planet’s distance from the sun is proportional to the square of the length of its year. This is an example of quantitative discovery. The application of machine learning to quantitative discovery was pioneered by the BACON system (Langley, Bradshaw and Simon, 1983), and then explored by many systems since, such as COPER (Kokar, 1986), FAHRENHEIT (Zytkow, 1987), and ABACUS (Falkenhainer and Michalski, 1990). Similar problems have been explored independently by Zagoruiko (1972) under the name of empirical prediction.

Some equations may not apply directly to data, because of an inappropriate value of a constant, or different equations may apply under different qualitative conditions. For example, in applying Stoke’s Law to determine the velocity of a falling ball, if the ball is falling through a vacuum, its velocity depends on the length of time it has been falling and on the gravitational force being exerted upon it. A ball falling through some sort of fluid will reach a terminal velocity dependent on the radius and mass of the ball and the viscosity of the fluid.

A program ABACUS (Greene, 1988; Falkenhainer and Michalski, 1990; Michalski, 1991) is able to determine quantitative laws under different qualitative conditions. It partitions the data into example sets, each of which adheres to a different equation determined by a quantitative discovery module. The qualitative discovery module can then determine conditions/rules that characterize each of these example sets (in the case of Stoke's Law, the rules would be based on the medium of descent).

2.6 Qualitative Prediction

Most programs that determine rules from examples determine them from instances of various classes of objects. An instance of a concept exemplifies that concept regardless of its relationship to other examples. Contrast that with a sequence prediction problem, in which a positive example of a concept is directly dependent on the position of the example in the sequence. For example, Figure 4 shows a sequence of seven figures. One may ask what object plausibly follows in the eighth position? To answer such a question, one needs to search for a pattern in the sequence, and then use the pattern to predict a plausible sequence continuation. In *qualitative prediction*, the problem is not to predict a specific value of a variable (as in time series analysis), but to *qualitatively* describe a plausible future object, that is, to describe plausible properties of a future object.

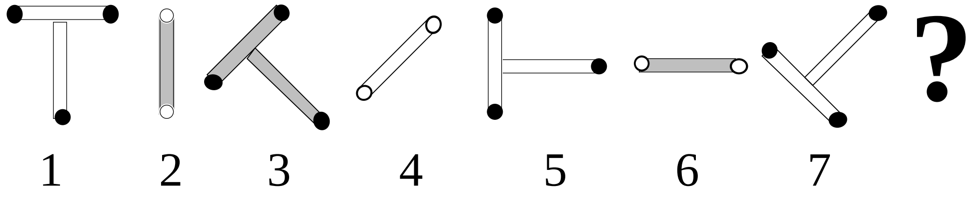


Figure 4. An example of a sequence prediction problem.

In the example in Figure 4, one may observe that the sequence consists of T-shaped figures with black tips and I-shaped figures with white tips. The figures may be white or shaded, and may be rotated in different orientations at 45-degree intervals. But is there a consistent pattern?

To determine such a pattern, one can employ different *descriptive models*, and instantiate the models to fit the particular sequence. The instantiated model that best fits the data is then used for prediction. Such a method is described in (Dieterrich and Michalski, 1986). The method employs three descriptive models—periodic, decomposition and DNF.

The *periodic model* is used to detect repeating patterns in a sequence. For example, Figure 4 depicts a recurring pattern that alternates T-shaped and I-shaped objects. In general, there can also be periodic sequences within the periodic sequences. In the figure, the T-shaped objects form a subsequence in which individual objects rotate leftward by 45 degrees.

The second model, the *decomposition model*, is used to characterize a sequence by decision rules in the following general form: "If one or more of the previous elements of the sequence have a given set of characteristics, then the next element will have the following characteristics." One such rule that applies to the sequence in Figure 4 would state that if an element in the sequence has a vertical component, then the next element in the sequence will have a shaded component; otherwise it will have no shaded components.

The third model, the DNF (disjunctive normal form) or "catch-all" model, tries to capture general properties characterizing the whole sequence. For example, for the sequence in Figure 4, it could instantiate to a statement such as "all elements in the sequence are T-shaped or I-shaped, they have white or shaded interiors, white or black tips, etc.

The program SPARC/G (Michalski, Ko and Chen, 1986) employs these three descriptive models to detect patterns in a sequence of arbitrary objects, and then uses the patterns to predict a plausible continuation for the sequence. For the sequence in Figure 4, SPARC/G found the following strong pattern based on the periodic model:

```
Period<    [Shape=T-shape] & [orientation(i+1)=orientation(i) - 45],
           [Shape = I-shape] & [orientation(i+1)=orientation(i) + 45] &
           [shaded(i+1)=unshaded(i)]>
```

The pattern can be paraphrased: There are two phases in a repeating period (their descriptions are separated by a comma). The first phase involves a T-shaped figure, and the second phase an I-shaped figure. The T-shaped figure rotates to the left, and the I-shaped figure rotates to the right by 45 degrees in relation to its predecessor. I-shaped figures are alternatingly shaded and unshaded. Based on this pattern, a plausible next figure in the sequence would be an unshaded I-shaped figure rotated clockwise 45 degrees in relation to the previous I-shaped figure.

The qualitative prediction capabilities described above can be useful for conceptual exploration of temporal databases in many application domains, such as agriculture, medicine, robotics, economic forecasting, etc.

2.7 Summarizing the Machine Learning-Oriented Approach

To help the reader to develop a rough sense of what is different and new in the above, let us summarize operations typically performed by traditional multivariate data analysis methods. These include computing mean-corrected or standardized variables, variances, standard deviations, covariances and correlations among attributes; principal component analysis (determining orthogonal linear combinations of attributes that maximally account for the given variance); factor analysis (determining highly correlated groups of attributes); cluster analysis (determining groups of data points that are close according to some distance measure); regression analysis (fitting an equation of an assumed form to given data points); multivariate analysis of variance; and discriminant analysis. All these methods can be viewed as primarily oriented toward a numerical characterization of a data set.

In contrast, the machine learning methods described above are primarily oriented toward developing symbolic logic-style descriptions of data, which may characterize one or more sets of data qualitatively, differentiate between different classes (defined by different values of designated output variables), create a “conceptual” classification of data, select the most representative cases, qualitatively predict sequences, etc. These techniques are particularly well suited for developing descriptions that involve nominal (categorical) and rank variables in data.

Another important distinction between the two approaches to data analysis is that statistical methods are typically used for globally characterizing a class of objects (a table of data), but not for determining a description for predicting class membership of future objects. For example, a statistical operator may determine that the average lifespan of a certain type of automobile is 7.3 years. Knowledge of the average lifespan of automobiles in a given class does not allow one to recognize the type of a particular automobile for which one obtained information about how long this automobile remained driveable. In contrast, a symbolic machine learning approach might create a description such as “if the front height of a vehicle is between 5 and 6 feet, and the driver’s seat is 2 to 3 feet above the ground, then the vehicle is likely to be a minivan." Such descriptions are particularly suitable for assigning entities to classes on the basis of their properties.

The INLEN methodology integrates a wide range of strategies and operators for data exploration based on machine learning research, as well as statistical operators. The reason for such a multistrategy approach is that a data analyst may be interested in many different types of information about the data. Different types of questions require different exploratory strategies and different operators.

3 Classification of Data Exploration Tasks

The problems described above can be simply illustrated by means of a *general data table* (GDT). Such a table is a generalization of a standard data table used in data analysis (Figure 5). It consists of a collection of relational tables (data tables) arranged in layers ordered by the time instance associated with each table. A GDT is used to represent a sequence of entities as they change over time. Examples of a GDT are a sequence of medical records of a patient (when each record is represented as a table of test results), a sequence of descriptions of a crop as it develops in the field, a sequence of data tables characterizing the state of a company during selected time instances, etc.

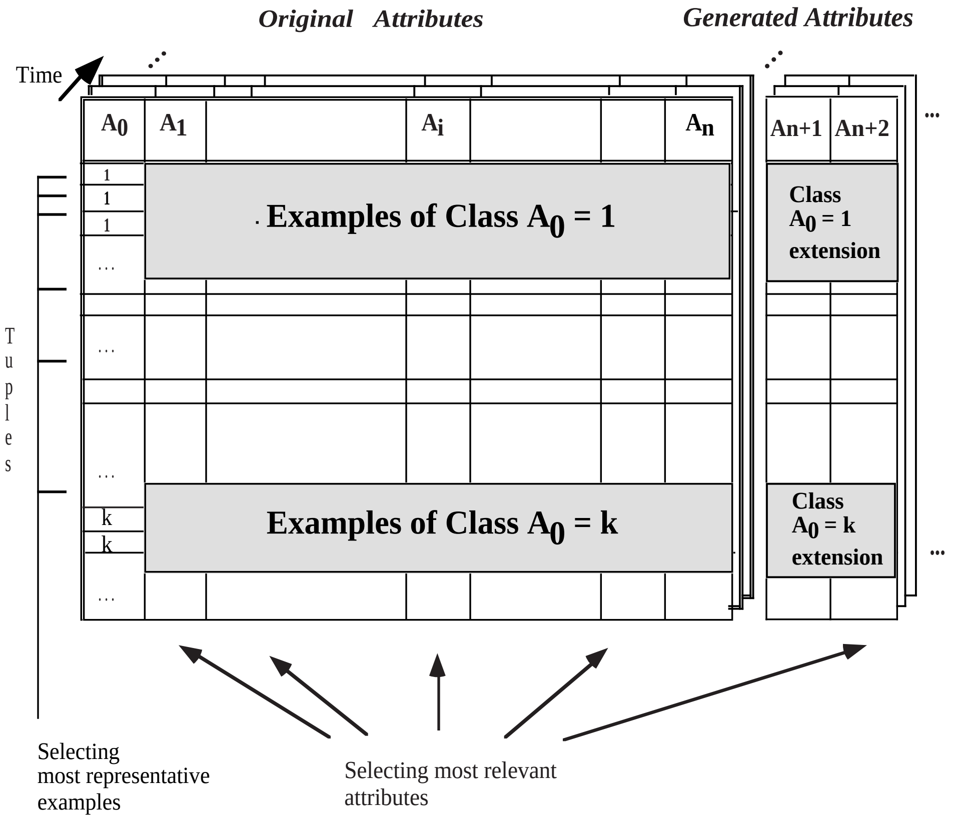


Figure 5. A GDT illustrating the role of different symbolic operators.

Columns in the tables correspond to attributes used to characterize entities associated with the rows. These may be initial attributes, given a priori, or additional ones generated through a process of *constructive induction* (e.g., Wnek and Michalski, 1994). Each attribute is assigned a *domain* and a *type*. The *domain* specifies the set of all legal values that the attribute can be assigned in the table. The *type* defines the ordering (if any) of the values in the domain. For example, the AQ15 learning program (Michalski et al, 1986) allows four types of attributes: nominal (no order), linear (total order), cyclic (cyclic total order), and structured (hierarchical order; see Kaufman and Michalski, 1996). The attribute type determines the kinds of operations that are allowed on this attribute's values during a learning process.

Entries in each row are values of the attributes for the entity associated with the row. Typically, each row corresponds to a single entity. However, in large databases whose records represent common, repeatable transactions, a column can be added to represent the number of occurrences of that particular transaction. With such information, discovery tools can incorporate a bias based on the frequency of instances.

Entries in the various columns of the table can be specific values of the corresponding attributes, the symbol "?," meaning that a value of this attribute is unknown for the given entity, or the symbol N/A, if an attribute does not apply to a specific entity. For example, "number of legs" usually applies to an animal, but would not apply to a plant.

An important problem of conceptual data exploration is to determine which attribute or attributes in a table functionally depend on other attributes. A related problem is to determine a general form of this relationship that would enable one to predict values of some attributes for future entities. For instance, when it is known that a nominal-scale attribute depends on other (independent) attributes, the problem is to hypothesize a general description of this relationship so that one can predict values of the nominal-scale attribute for future combinations of values of the independent attributes. This problem is equivalent to the problem of concept learning from examples, so methods developed in machine learning_directly apply. In such a case, the column in the data table that corresponds to the dependent attribute represents the *output attribute.* The values of that variable are classes whose descriptions are to be learned. In Figure 5, for illustration, it was assumed that the first_column (attribute A_0) represents values of the output variable. When there are no a priori classes to which entities belong, there is no such designated column. In this case, methods of conceptual clustering can be applied to determine a classification of entities.

Below we use the GDT (Figure 5) to relate machine learning techniques described in the previous section to data exploration problems.

Learning rules from examples:

Suppose that one discrete attribute in the GDT has been designated as the output attribute, and all or some of the remaining attributes as input (independent) attributes. A set of rows in the table for which the output attribute takes the same value can be viewed as a set of training examples of the decision class (concept) symbolized by this value. Any of the conventional concept learning techniques can be directly applied for determining a rule relating the output attribute to the input attributes. For a general analysis of the data set, every discrete attribute (and continuous attributes as well after quantization) can be considered as an output attribute, and a machine learning method can be applied to determine a relationship between that attribute and other attributes. The determination of such relationships (rules) can be guided by different rule quality criteria, for example, simplicity, cost, predictive accuracy, etc. In the INLEN system, the AQ learning method was applied due to the simplicity and the high comprehensibility of decision rules it generates (Wnek et al, 1995; Bloedorn and Michalski, 1996).

Determining time-dependent patterns:

This problem concerns the detection of temporal patterns in sequences of data arranged along the time dimension in a GDT (Figure 5). Among the novel ideas that could be applied for analyzing such time-dependent data is a multi-model method for qualitative prediction (Dieterrich and Michalski, 1986; Michalski, Ko and Chen, 1985; 1986). Another novel idea is a temporal constructive induction technique that can generate new attributes that are designed to capture time-dependent patterns (Davis, 1981; Bloedorn and Michalski, 1996).

Example selection:

The problem is to select rows from the table that correspond to the most representative examples of different classes. When a datatable is very large, is it important to concentrate the analysis on a representative sample. The "method of outstanding representatives" selects examples (tuples) that are most different from the other examples (Michalski and Larson, 1978).

Attribute selection:

When there are many columns (attributes) in the GDT, it is often desirable to reduce the data table by removing columns that correspond to the least relevant attributes for a designated learning task. This can be done by applying one of many methods for attribute selection, such as Gain Ratio(Quinlan, 1993) or Promise level (Baim, 1982).

Generating new attributes:

The problem is to generate additional columns that correspond to new attributes generated by a constructive induction procedure. These new attributes are created by using the problem's background knowledge and/or special heuristic procedures as described in papers on constructive induction (e.g., Bloedorn, Wnek and Michalski, 1993).

Clustering:

The problem is to automatically partition the rows of the table into groups that correspond to "conceptual clusters," that is, sets of entities with high conceptual cohesiveness (Michalski, Stepp and Diday, 1981). Such a clustering operator will generate an additional column in the table that corresponds to a new attribute "cluster name." The values of this attribute for each tuple in the table indicate the assigned class of the entity. Rules that describe clusters are stored separately in the Knowledge Base and linked to the entities via knowledge segments (see Section 4). An example of a clustering is presented in Section 5.

Determining attribute dependencies:

The problem is to determine relationships, such as correlations, causal dependencies, logical or functional dependencies among the attributes (columns) in the given GDT, using statistical and logical methods.

Incremental rule update:

The problem is to update working knowledge (in particular, rulesets characterizing relationships among attributes in the GDT) to accommodate new instances or time slices in the table. To do so, an incremental learning program must be applied to synthesize the prior knowledge with the new information. The incremental learning process may be full-memory, partial-memory, or no-memory, depending on how much of the original training data is maintained in the incremental learning process (Hong, Mozetic and Michalski, 1986; Reinke and Michalski, 1988; Maloof and Michalski, 1995).

Searching for approximate patterns in (imperfect) data:

For some GDTs, it may not be possible (or useful) to find complete and consistent descriptions. In such cases, it is important to determine patterns that hold for a large number of cases, but not necessarily for all. An important case of this problem is when some entries in the table are missing or incorrect. The problem is then to determine the best (i.e., the most plausible) hypothesis that accounts for most of the available data.

Filling in missing data:

Given a data table in which some entries are missing, determine plausible values of the missing entries on the basis of analysis of the currently known data. An interesting approach to this problem is to apply a multi-line reasoning, based on the core theory of human plausible reasoning (Collins and Michalski, 1981; 1989; Dontas, 1988).

Determining decision structures from declarative knowledge (decision rules):

Suppose that a set of general decision rules (a declarative form of knowledge) has been hypothesized for a given data set (GDT). If this ruleset is to be used for predicting new cases (by a computer program, or by an expert), it may be desirable to convert it into the form of a decision tree (or a more general form, a decision structure) that is tailored to a given decision-making situation (e.g., by taking into consideration the cost of measuring attributes). A methodology for doing this and arguments for and against using such an approach (as opposed to the traditional method of learning of decision trees directly from examples) are discussed in (Imam and Michalski, 1993; Imam ,1995; Michalski and Imam, 1997).

Methods for performing the above operations on data tables have been implemented in various machine learning programs (e.g., Michalski, Carbonell and Mitchell, 1983; 1986; Forsyth and Rada, 1986; Kodratoff, 1988; Kodratoff and Michalski, 1990). Below we describe the INLEN system that aims at ultimately incorporating all of these programs as operators in one integrated system for the generation of knowledge from data.

4 Integration of Many Operators in INLEN

To make the data exploration operations described above easily available to a data analyst, and applicable in sequences in which the output from one operation is an input to another one, programs performing these operations need to be integrated into one system. This idea underlies the INLEN system (Kaufman, Michalski and Kerschberg, 1991; Michalski et al, 1992). The name INLEN is derived from **in**ference and **le**ar**n**ing. The system integrates machine learning programs, statistical data analysis tools, a database, a knowledge base, inference procedures, and various supporting programs under a unified architecture and graphical interface. The knowledge base is used for storing, updating and applying rules and other forms of knowledge that may be employed for assisting data exploration, and for reporting results from it.

The general architecture of INLEN is presented in Figure 6. The system consists of a database (DB) connected to a knowledge base (KB), and a set of operators. The operators are divided into three classes:

- *DMOs:* Data Management Operators, which operate on the database. These are conventional data management operators that are used for creating, modifying and displaying relational tables.

- *KMOs:* Knowledge Management Operators, which operate on the knowledge base. These operators play a similar role to the DMOs, but apply to the rules and other structures in the knowledge base.
- *KGOs*: Knowledge Generation Operators, which operate on both the data and knowledge bases. These operators perform symbolic and numerical data exploration tasks. They are based on various machine learning and inference programs, on conventional data exploration techniques, and on visualization operators for displaying graphically the results of exploration. The diagrammatic visualization method DIAV (Wnek ,1995) is used for displaying the effects of symbolic learning operations on data.

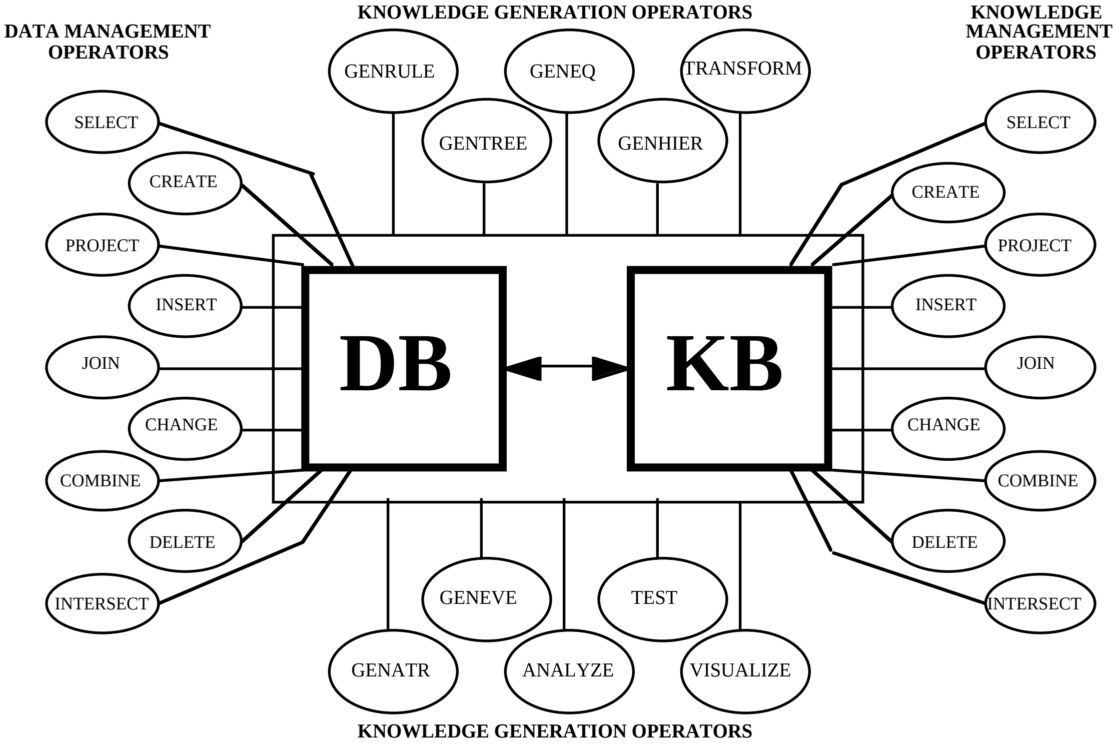


Figure 6. An architecture of the INLEN system for multistrategy data exploration.

The KGOs are the heart of the INLEN system. To facilitate their use, the concept of a *knowledge segment* was introduced (Kaufman, Michalski and Kerschberg, 1991). A knowledge segment is a structure that links one or more relational tables from the database with one or more structures from the knowledge base. KGOs can be viewed as modules that perform some form of inference or transformation on knowledge segments and, as a result, create new knowledge segments. Knowledge segments are both inputs to and outputs from the KGOs. Thus, they facilitate the passage of data and knowledge from one knowledge generation operator to another.

The execution of a KGO usually requires some background knowledge, and is guided by control parameters (if some parameters are not specified, default values are used). The background knowledge may contain some general knowledge as well as knowledge specifically relevant to a given application domain, such as a specification of the value sets and types of attributes, the constraints and relationships among attributes, initial rules hypothesized by an expert, etc. The KGOs can be classified into groups, based on the type of operation they perform. Each group includes a number of specific operators that are instantiated by a combination of parameters. The basic operator groups are as follows:

- GENRULE operators generate different kinds of decision rules from given facts. A specific operator may generate rules characterizing a set of facts, discriminating between groups of facts, characterizing a sequence of events, and determining differences between sequences, based on programs such as AQ15c (Wnek et al, 1995) and SPARC/G (Michalski, Ko and Chen, 1986). A KGO for learning rules can usually work in either incremental or batch mode. In the incremental mode, it tries to improve or refine the existing knowledge, while in the batch mode, it tries to create entirely new knowledge based on the facts in the database, and knowledge in the knowledge base.

- GENTREE operators build a decision structure from a given set of decision rules (e.g., Imam and Michalski, 1993), or from examples (e.g., Quinlan, 1993). A decision structure is a generalization of the concept of a decision tree in which nodes can be assigned an attribute or a function of attributes. Individual branches may be assigned a set of attribute values. Leaves may be assigned a set of decisions (Imam and Michalski, 1993; Imam, 1995).

- GENEQ operators generate equations characterizing numerical data sets and qualitatively describing the conditions under which these equations apply (e.g., Falkenhainer and Michalski, 1990).

- GENHIER operators build conceptual clusters or hierarchies. They are based on the program CLUSTER methodology (Michalski, Stepp and Diday, 1981). The operator in INLEN is based on the reimplementation in C of the program CLUSTER/2 (Stepp, 1984).

- TRANSFORM operators perform various transformations on the knowledge segments, e.g., generalization or specialization, abstraction or concretion, optimization of given rules, etc. according to user-provided criteria. For instance, one such operator climbs an attribute's generalization hierarchy to build more general decision rules (Kaufman and Michalski, 1996).

- GENATR operators generate new attribute sets by creating new attributes (Bloedorn and Michalski, 1996), selecting the most representative attributes from the original set (Baim, 1982), or by abstracting attributes (Kerber, 1992).

- GENEVE operators generate events, facts or examples that satisfy given rules, select the most representative events from a given set (Michalski and Larson, 1978), determine examples that are similar to a given example (Collins and Michalski, 1989), or predict the value of a given variable using an expert system shell or a decision structure.

- ANALYZE operators analyze various relationships that exist in the data, e.g., determining the degree of similarity between two examples, checking if there is an implicative relationship between two variables, etc. Statistical and symbolic operators alike may perform these tasks.

- TEST operators test the performance of a given set of rules on an assumed set of facts. The output from these operators is a confusion matrix—a table whose (i,j)th element shows how many examples from the class i were classified by the rules to be in class j. These operators can also be used to apply the rules to any given situation to determine a decision. The TEST operator implemented in INLEN is based on the ATEST program (Reinke, 1984).

- VISUALIZE operators are used to present data and/or knowledge to the user in a convenient, easy-to-understand format (Wnek, 1995).

Summarizing, INLEN integrates a large set of operators for performing various types of operations on the data base, on the knowledge base, or the data and knowledge bases combined.

5 Illustration of Clustering and Learning Operators

Among the most important knowledge generation operators implemented in INLEN are the operator for creating a classification of data (clustering), and the operator for learning general rules relating a designated concept (attribute) to other designated attributes. The first operator is realized by the CLUSTER/2 program for conceptual clustering (Stepp, 1984). The second operator is realized by the AQ15c rule learning program (Wnek et al, 1995). This section illustrates these operators through an application to a datatable characterizing hard drives (Figure 7). The datatable is based on information published in the October, 1994 issue of *MacUser*.

In the table presented in Figure 7, each row (except for the first one) describes a hard drive in terms of the attributes specified in the first row. Suppose that the task of data exploration is to develop a classification of the hard drives into some meaningful categories. For this task, the operator CLUSTER is applied. Let us assume that the operator will seek a clustering that maximizes the quality of classification, as defined by two criteria: the simplicity of the descriptions of generated categories, and the cohesiveness of the_descriptions (measured by the ratio of the number of instances in the datatable covered by a given description to the number of possible instances covered by the description). The input to the conceptual clustering operator is the table in Figure 7 (without the rightmost column, which, for the sake of saving space, represents already the result of clustering).

The result of applying the clustering operator is a knowledge segment containing two components—a new, extended data table, and a set of rules. The new table, in comparison to_the input table, has an additional column—the rightmost column in Figure 7, labeled "Group," which represents the category assignments of the drives by the clustering operator. The second component is the set of rules describing the categories that were generated. Here are the rules describing the categories created by the operator:

[Class 1] ⇐ [Toll_free_Support is yes] & [FCC_Class-B is yes] & [Encryption is no] & [SCSI_50-Pin is yes or no] & [Guarantee is yes or by dealer]

[Class 2] ⇐ [Toll_free_Support is no] & [SCSI_50-Pin is yes] & [5yr_Warranty is yes] & [Guarantee is yes or no] & [Loaners is yes or no]

[Class 3] ⇐ [Toll_free_Support is no] & [FCC_Class-B is yes] & [AC outlet is yes] & [Passwd_Protect is yes] & [5yr_Warranty is no] & [Guarantee is not by dealer] & [Loaners is yes or if available]

Hard Drive	AC Outlet	SCSI 50-Pin	FCC Class B	Passwd Protect	Encrypt	5yr Warranty	Toll-free Support	Guarantee	Loaners	Capacity	Group
Apple 1050	no	yes	yes	yes	no	no	yes	by dealer	by dealer	low	**1**
Micropolis	no	yes	yes	yes	yes	yes	no	no	no	low	**2**
SLMO 1000	no	yes	Class A	yes	no	yes	no	no	yes	low	**2**
Focus 1G	yes	yes	yes	yes	no	no	yes	yes	yes	low	**1**
GHD 1200S	no	yes	yes	no	no	yes	no	no	no	low	**2**
Joule 1080	yes	no	yes	yes	no	yes	yes	yes	no	low	**1**
Liberty 1GB	no	25 pin	yes	yes	no	no	no	yes	yes	low	**3**
Spitfire 1GB	yes	yes	yes	yes	no	yes	no	yes	no	low	**2**
PowerUser 1070	no	yes	yes	no	no	no	yes	yes	no	low	**1**
P1000	no	yes	yes	yes	no	no	yes	yes	no	low	**1**
Seagate 1075	yes	yes	yes	yes	no	no	yes	yes	no	low	**1**
Minipak 1000	no	yes	yes	yes	no	no	no	yes	yes	low	**3**
PowerCity 1GB	yes	yes	yes	yes	no	on mech.	yes	yes	no	low	**1**
Spin 1021	no	yes	yes	yes	no	yes	yes	yes	no	low	**1**
APS MS 1.7	no	yes	yes	yes	no	yes	yes	yes	no	high	**1**
Seagate 2GB	no	yes	yes	yes	yes	yes	no	no	no	high	**2**
SLMO 2000	no	yes	no	yes	no	yes	no	no	yes	high	**2**
Focus 2G	yes	yes	yes	yes	no	no	yes	yes	yes	high	**1**
FWB 1760MF	no	68 pin SCSI2	yes	yes	yes	no	no	no	if avail.	high	**3**
Liberty 2GB	no	no	yes	yes	no	no	no	yes	yes	high	**3**
Loviel L2000	yes	yes	yes	yes	no	yes	no	yes	yes	high	**2**
Seagate 2.1	yes	yes	yes	yes	no	yes	no	yes	no	high	**2**
PowerUser 1801	no	yes	yes	no	no	no	yes	yes	no	high	**1**
MacP Sg 28	no	yes	yes	yes	no	yes	no	yes	no	high	**2**

Figure 7. A datatable characterizing hard drives.

Thus, the operator created three categories of hard drives and described each category in the form of rules. Each rule shows all characteristics common to a given category, that is, it represents a *characteristic description* of a category (Michalski, 1983). (Note that some of the conditions in these rules appear to be redundant. For example, the last condition of the Class 2 rule says that Loaners is yes or no. This can be explained by the presence of a third value, “by dealer,” that neither guarantees, nor rules out a loaner.) These characterizations do not point out the most significant distinctions between a given category and other categories.

To create a description that points out the most significant distinctions, one needs to apply the operator that creates *discriminant descriptions* (Michalski, 1983). The operator (GENRULE) is applied to the extended data table in Figure 7, using the “Group” column as its output attribute. The result is a set of new decision rules:

[Class 1] ⇐ [Toll_free_Support is yes]

[Class 2] ⇐ [Toll_free_Support is no] & [5yr_Warranty is yes]

[Class 3] ⇐ [Toll_free_Support is no] & [5yr_Warranty is no]

The obtained rules are much simpler and easier to interpret than the rules generated by the CLUSTER operator that invented the three classes. The reason is that a discriminant description lists only those characteristics that are necessary to discriminate a given category from the other categories. Discriminant descriptions are designed to provide the minimum information needed for distinguishing between entities of different categories. Both characteristic and discriminant descriptions are *complete* and *consistent* with all the examples in Figure 7, i.e., they classify all examples in the same way.

6 Data and Rule Visualization

It is desirable for data analysts to be able to visualize the results of different operators in order to relate visually the input data to the rules that have been learned from them, to see which datapoints would corroborate or contradict these rules, to identify possible errors, etc. To this end, INLEN supports the visualization of data and knowledge through the *diagrammatic visualization* method implemented in the DIAV program (Michalski, 1978; Wnek, 1995).

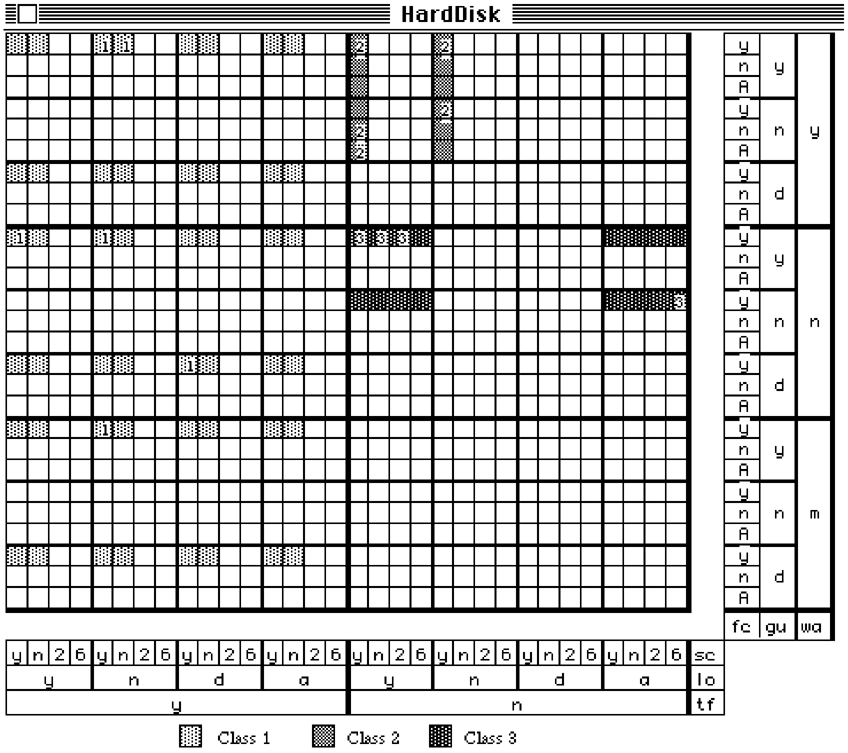

HardDisk
y n A y n A y n A y n A y n A y n A y n A y n A y n A
y n d y n d y n d
y n m
fc gu wa
y n 2 6 y n 2 6 y n 2 6 y n 2 6 y n 2 6 y n 2 6 y n 2 6 y n 2 6 sc
y n d a y n d a lo
y n tf
Class 1
Class 2
Class 3

Figure 8. A visualization of the *characteristic description* created by the conceptual clustering operator.

Let us illustrate the method with the hard disk classification problem presented in the previous section. The representation space, projected onto six attributes, is pictured in Figure 8. To simplify the visualization, the attributes used to span the diagram, Toll_free_Support (tf), Loaners (lo), SCSI_50-Pin (sc), FCC_Class-B (fc), Guarantee (gu), and 5yr_Warranty (wa), are only those that appeared most frequently in the characteristic descriptions created by the conceptual clustering operator. Each cell in the diagram corresponds to one combination of attribute values, specified by the annotations of the columns and rows. Thus the upper-leftmost cell corresponds to a datapoint in which all six of these attributes have the value yes (y).

The 24 examples from Figure 7 have been projected onto this space, and are represented by placing their class number in the corresponding cells. The shaded areas represent the characteristic descriptions of the classes generated by the clustering operator; the lightest color indicates Class 1, the intermediate shade represents Class 2, and the darkest one indicates Class 3. As can be seen in the diagram, the descriptions generated by the clustering operator are generalizations of the input instances, as they also cover instances that have not yet been observed (shaded areas without a number).

For comparison, Figure 9 is a visualization of the discriminant descriptions generated by the rule_learning operator from the input examples classified according to the previously generated clustering. The organization of the diagram in Figure 9 is the same as in Figure 8 with regard to the labeling of examples, classes, rows and columns. Because discriminant descriptions focus only on features that distinguish among the classes, they cover broader sections of the representation space. Thus, they are much more general than characteristic descriptions.

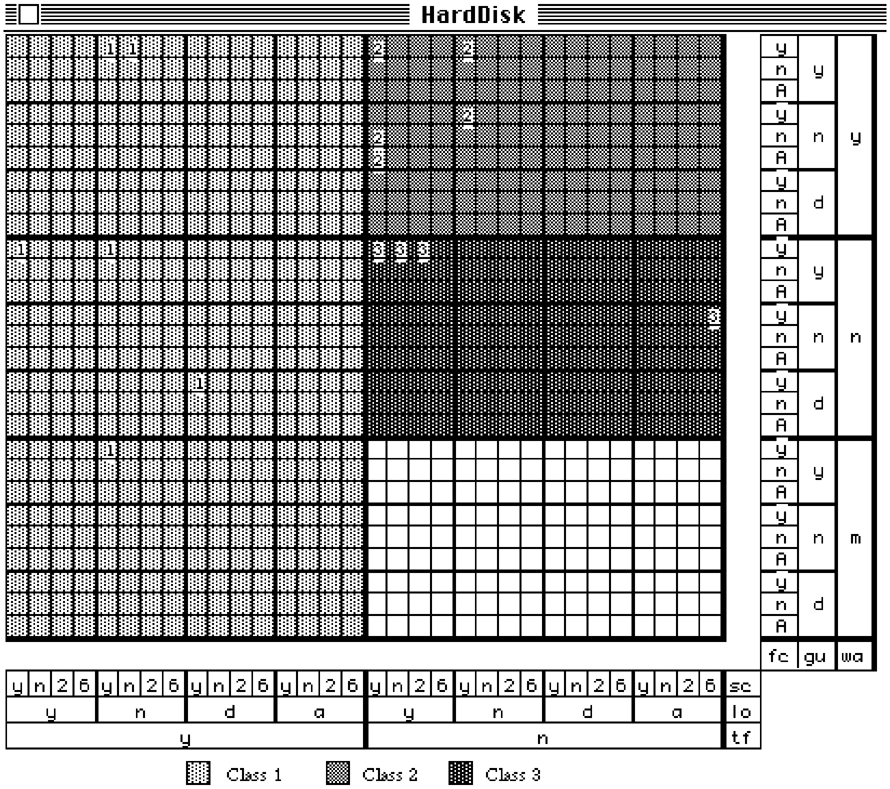


Figure 9. A visualization of the *discriminant rules* created by the inductive generalization operator.

The obtained discriminant descriptions divide the representation space into four sections, three corresponding to the rules for the three classes, and the fourth to the indeterminate portion of the event space, containing none of the known instances of the three categories. This latter section is defined by the combination of characteristics: Toll_free_Support = no and 5yr_Warranty = on_mechanism.

Note also that due to the greater generality of the discriminant descriptions, the indeterminate area is much smaller than in the case of characteristic descriptions (the blank area in Figure 8).

As can be seen from the diagram, the generated discriminant descriptions are consistent and complete with regard to all of the presented examples, that is, they preserve the classification of cases created by the clustering operator. Summarizing, the visualization method presented above makes it very easy to see how generated descriptions relate to the cases from which they were generated.

7 Learning Rules with Structured Attributes

In addition to conventional symbolic and numerical attributes, INLEN supports a new kind of attribute, called *structured*. Such attributes have value sets ordered into hierarchies (Michalski, 1980). In order to take advantage of the properties of structured attributes in executing inductive learning, new inductive generalization rules have been defined.

An inductive generalization rule (or transmutation) takes an input statement and relevant background knowledge, and hypothesizes a more general statement (Michalski, 1980; 1983; 1994). For example, removing a condition from the premise of a decision rule is a generalization transmutation (this is called a *dropping condition* generalization rule), since if the premise has fewer conditions, a larger set of instances can satisfy it.

A powerful inductive generalization operator used in the AQ learning programs is the *extension-against* operator. If rule **R1:** $\mathbf{C \Leftarrow [x_i = A] \;\&\; CTX1}$ characterizes a subset of positive concept examples, $\mathbf{E}^+$, of the concept **C**, and rule **R2:** $\mathbf{C \Leftarrow [x_i = B] \;\&\; CTX2}$ characterizes negative examples, $\mathbf{E}^-$ (where **A** and **B** represent disjoint subsets of the values of x_i, and the CTXs stand for any additional conditions), then the *extension of R1 against R2 along dimension x_i*

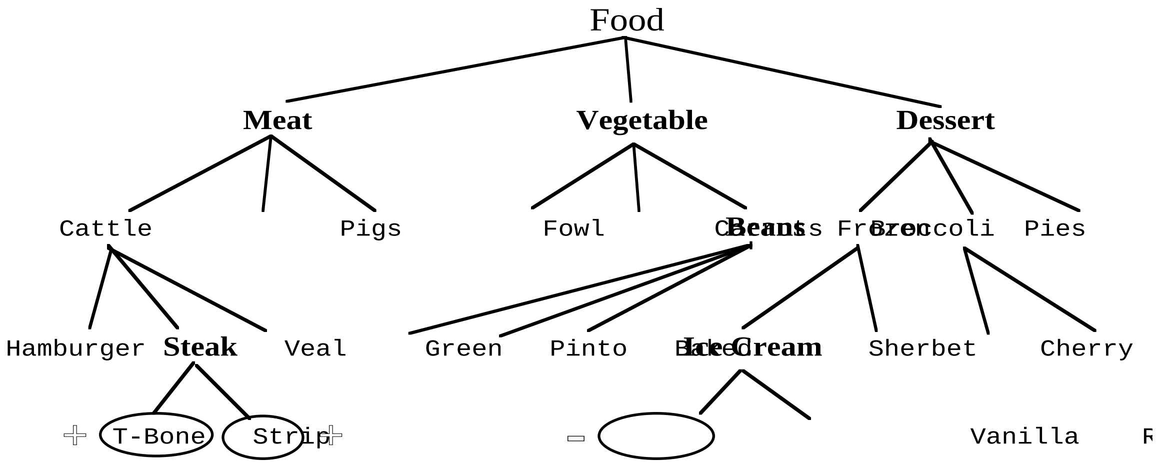
$$\mathbf{C \Leftarrow R1 \;—\!\!|\; R2} \;/x_i$$

produces a new rule **R3:** $\mathbf{[x_i \neq B \cup \varepsilon]}$, which is a *consistent generalization* of **R1,** that is, a generalization that does not intersect logically with **R2** (Michalski and McCormick, 1971; Michalski, 1983). The value of the parameter ε controls the degree of generalization. If ε is ø (the empty set), then **R3** is the *maximal consistent generalization* of **R1**. If ε is $\mathbf{D(x_i) \setminus (A \cup B)}$ (where $D(x_i)$ is the domain of x_i), then **R3** is the minimal consistent generalization of **R1** involving only x_i. In AQ programs, the extension-against operator is typically used with ε = ø.

By repeating the extension-against operator until the resulting rule no longer covers any negative examples, a consistent concept description (one that covers no negative examples) can be generated. Such a process can be applied in order to generate a description (cover) that is complete and consistent with regard to all the training examples.

By applying the extension-against operator with different values of the parameter ε, one_can generate descriptions with different degrees of generality. For instance, in AQ15c, in order to learn a characteristic rule, the output of the operator with ε initially set to ø is maximally specialized in such a way that it continues to cover all of the positive examples described by the initial extension. If discriminant rules are desired, the extension will be maximally generalized so long as it continues not to cover any negative examples of the concept.

In order to effectively apply the extension-against operator to structured attributes, new generalization rules need to be defined. Let us illustrate the problem by an example that uses a structured attribute “Food” shown in Figure 10. Each non-leaf node denotes a concept that is more general than its children nodes. These relationships need to be taken into consideration when generalizing given facts. Suppose that the concept to be learned is exemplified by statements: “John eats strip steak” and “John doesn’t eat vanilla ice cream.” There are many consistent generalizations of these facts, for example, “John eats strip steak,” “John eats steak,” “John eats cattle,” “John eats meat,” “John eats meat or vegetables,” or “John eats anything but vanilla ice cream.” The first statement represents the maximally specific description (no generalization), the last statement represents the maximally general description, and the remaining ones represent intermediate levels of generalization. A problem arises in determining the generalization of most interest for a given situation. We approach this problem by drawing insights from human reasoning.



Anchor nodes are shown in bold. Nodes marked by + and – are values occurring in positive and negative examples, respectively.

Figure 10. The domain of a structured attribute “Food.”

Cognitive scientists have noticed that people prefer certain nodes in a generalization hierarchy (concepts) over other nodes when creating descriptions (e.g., Rosch et al, 1976). Factors that influence the choice of a concept (node) include the concept typicality (how common are a concept’s features among its sibling concepts), and the context in which the concept is being used. For instance, upon seeing a robin (a typical bird), we may say, “There is a bird,” rather than “There is a robin,” assuming that the given situation does not require a specification of the type of bird. On the other hand, when we see a penguin, a much less typical bird, we are more likely to say “There is a penguin,” rather than “There is a bird”. This way a listener (who is not an observer) will not assign to the unseen bird characteristics typical to a bird, but rather the special characteristics of a penguin. This facilitates communication. Context also comes into play; at a gathering of bird watchers, the robin will not likely be called simply a bird, but rather will be referred to by its taxonomic name.

To provide some mechanism for capturing such preferences, INLEN allows a user to define *anchor nodes* in a generalization hierarchy. Such nodes should reflect the interests of a given application (Kaufman and Michalski, 1996). To illustrate this idea, consider Figure 10 again. In this hierarchy, vanilla and rocky road are kinds of ice cream; ice cream is a frozen dessert, which is a dessert, which is a type of food. In everyday usage, depending on the context, we will typically describe vanilla or rocky road as ice cream or dessert, but less likely as frozen dessert or food. Hence, we can designate dessert and ice cream as anchor nodes in the Food hierarchy. Using information about anchor nodes, different rule preference criteria can be specified, such as selecting the rule with the most general anchor nodes, or the one that generalizes positive examples to the next higher anchor node(s).

INLEN supports the use of structured attributes both as *independent* (input) and *dependent* (output) variables. Structured independent attributes represent hierarchies of values that are used to characterize entities. Structured dependent attributes represent hierarchies of decisions or classifications that can be made about an entity. Through the use of structured output attributes, INLEN's learning module can determine rules at different levels of generality.

While dependent attributes, like independent ones, can take on in principle different types (nominal, linear, cyclic or structured), in practical applications they are frequently either nominal or linear. A nominal output attribute is most frequently used in concept learning; its values denote concepts or classes to be learned. A linear output attribute (which is typically a measurement on a ratio scale) is used to denote a measurement whose values are to be predicted on the basis of the past data.

In many applications, it is desirable to use a structured attribute as a dependent variable. For example, when deciding which personal computer to buy, one may first decide the general type of the computer—whether it is to be IBM PC-compatible or Macintosh-compatible. After deciding the type, one can focus on a specific model of the chosen type. The above two-level decision process is easier to execute than a one-level process in which one has to directly decide which computer to select from a large set.

When a dependent variable is structured, the learning operator focuses first on the top-level values (nodes), and creates rules for them. Subsequently, it creates rules for the descendant nodes in the context of their ancestors. This procedure produces decision rules that are simpler and easier to interpret than rules learned with a flat (nominal) organization of the decision attribute.

8 Learning Decision Structures from Decision Rules

One of the main reasons for data exploration is to learn rules or patterns in data that will enable a data analyst to predict future cases. Thus, when such rules are learned, one needs a method for efficiently applying the rules for prediction. Since a convenient structure for implementing a decision process is a decision tree, the problem of how to transfer knowledge to a decision tree arises. In the conventional machine learning approach, decision trees are learned directly from training examples, thus avoiding the step of first creating rules (Hunt, Marin and Stone, 1966; Quinlan, 1986; 1993).

Learning a decision tree directly from examples, however, may have serious disadvantages in practice. A decision tree is a form of procedural knowledge. Once it has been constructed, it is not easy to modify it to accommodate changes in the decision-making conditions. For example, if an attribute (test) assigned to a high-level node in the tree is impossible or too costly to measure, the decision tree offers no alternative course of action other than probabilistic reasoning (Quinlan, 1986).

In contrast, a human making the decision would probably search for alternative tests to perform. People can do this because they typically store decision knowledge in a declarative form. From a declarative form of knowledge, such as a set of decision rules, one can usually construct many different, but logically equivalent, or nearly equivalent, decision trees. One such decision tree may be preferable to another in a given decision-making situation. Therefore, it is desirable to store knowledge decoratively and to transfer it only when the need arises to the procedural form that is most appropriate to the given situation.

Another weakness of decision trees is that they may become unwieldy and incomprehensible because of their limited knowledge representational power. To overcome the above limitations, a new approach has been developed that creates task-oriented *decision structures* from decision rules (Imam, 1995, Michalski and Imam, 1997). A decision structure is a generalization of a decision tree in which tests associated with nodes can refer not only to single attributes, but also to functions of multiple attributes; branches may be associated not only with single values/results of these tests, but also with a set of such values; and leaves can be assigned not only a single decision, but also a set of alternative decisions with appropriate probabilities.

This approach has been implemented in the AQDT-2 program, and employs an AQ-type learning algorithm (AQ15c and AQ17-DCI) for determining decision rules from examples. Among its advantages are the ability to generate a decision structure that is most suitable to a particular task and the ability to avoid or delay measuring costly attributes. Different users may want to generate different decision structures from a given set of rules, so that the structures are tailored to their individual situations. Furthermore, if an attribute is difficult to measure, or cannot be measured at all, the program can be instructed to build a decision structure from rules that tries to avoid this attribute, or measure it only when necessary.

Another advantage of this methodology is that once a rule set is determined, a decision structure can be generated from it far more rapidly than if it has to be determined from examples, hence processing time is very small. Also, a set of rules will take up less storage space than the data set from which it was learned.

Experiments with AQDT-2 indicate that decision structures learned from decision rules tend to be significantly simpler than decision trees learned from the same data, and frequently also have a higher predictive accuracy. For example, a decision structure learned by AQDT-2 for a wind bracing design problem had 5 nodes and 9 leaves, with a predictive accuracy of 88.7% when tested against a new set of data, while the decision tree generated by the popular program C4.5 had 17 nodes and 47 leaves with a predictive accuracy of 84% (Michalski and Imam, 1997). In another experiment, a decision tree learned from decision rules by AQDT to analyze Congressional voting patterns had 7 nodes and 13 leaves, with a predictive accuracy of 91.8% (when AQDT built an equivalent decision structure by combining some branches, the number of leaves was reduced to 8), while the decision tree learned by C4.5 from the same set of training examples had 8 nodes and 15 leaves, with a predictive accuracy of 85.7% (Imam and Michalski, 1993).

This methodology directly fits the philosophy of INLEN. A rule base may be provided either from an expert or through the use of a rule learning operator, thereby allowing for the generation of decision structures from rules.

9 Automatic Improvement of Representation Spaces

9.1 Determining Most Relevant Attributes

In a large database, many attributes may be used to characterize given entities. For any specific problem of determining rules characterizing the relationship between a designated output attribute and other attributes, it may be desirable to limit the independent attributes to the most relevant ones. To this end, one may use many different criteria for evaluating the relevance of an attribute for a given classification problem, such as gain ratio (Quinlan, 1993), gini index (Breiman et al, 1984), PROMISE (Baim, 1982), and chi-square analysis (Hart, 1984; Mingers, 1989).

These criteria evaluate attributes on the basis of their expected global performance, which means that those attributes with the highest ability to discriminate among *all* classes are selected as the most relevant.

When determining a declarative knowledge representation, such as decision rules, the goal is somewhat different. Here, each class is described independently from other classes, and the simplest and most accurate rules for each class are desired. Hence, if an attribute has a single value that characterizes very well just one specific class, the attribute with this value will be used effectively in a corresponding decision rule. In contrast, such an attribute may have a low global discriminating value, and is thus ignored in building a decision tree. It follows that the determination of attributes for decision trees and for decision rules needs to follow different criteria.

To illustrate this point, consider the problem of recognizing the upper-case letters of the English alphabet. Two of the attributes to be considered might be whether the letter has a tail and whether it is made up exclusively of straight lines. In a rule-based (declarative) representation, the letter Q can be distinguished from the rest of the alphabet by a simple and concise property, *if the letter has a tail, it is a Q*. Conversely, the straight line condition is alone insufficient to discriminate any specific letter, but is useful overall.

Thus, the attribute *has-tail* is very useful for learning one specific class, although not very useful for characterizing other classes. It is thus appropriate for use in rule learning. In decision-tree learning, however, it may be evaluated as having a relatively low overall utility and replaced by other attributes. This will most likely happen if Qs are relatively rare. Hence, testing the letter for a tail will be considered a wasted operation, as it only serves to eliminate the possibility of it being a Q, without making any progress in distinguishing between the other 25 letters. Meanwhile, testing the condition *all-straight-lines* immediately bisects the search space. It is better to pare down the set of hypotheses more rapidly, and only check for a tail as a last step when the set of possible letters has been reduced to O and Q. This way, the recognition of Q will require more tests than necessary, but at no expense to the recognition of other letters.

INLEN supports both global and local attribute evaluation criteria for selecting the most relevant attributes. The former is based on the PROMISE methodology (Baim, 1982), while the latter employs a variation of PROMISE that is oriented toward the maximum performance of some attribute value, rather than on the attribute's global performance.

9.2 Generating New Attributes

When the original representation space is weakly relevant to the problem at hand, or the concept to be learned is difficult to express in the from of attributional decision rules such as those employed in INLEN, there is a need to generate new attributes that are functions of the original ones and better suited to the given problem. This is done by a *constructive induction* operator based on the program AQ17-DCI (Bloedorn and Michalski, 1996).

In the case of a database that contains information on objects changing over time, one needs a mechanism for constructive induction that can take advantage of the time data ordering. For example, the database may contain information on the maximum temperature at a given location each day, with a field in each record indicating the day on which its temperature was recorded. Inherent in a timestamped representation are many attributes that can be generated through constructive induction, for example, date of the highest temperature, the minimum population growth rate during some period, weediness on date of planting, etc.

CONVART (Davis, 1981) uses user-provided and default system suggestions to search for useful time-dependent attributes that are added to the representation space. It uses the items on the suggestion list to generate new attributes and to test them for likely relevance to the problem. If they exceed a relevance threshold, it adds them to the representation space, repeating this procedure until a desired number of new attributes have been constructed. As part of its attribute construction capability, INLEN will incorporate such techniques for the generation of time-dependent attributes.

10 Exemplary Application: Discovery in Economic and Demographic Data

10.1 Motivation

Economic analysis is one domain in which conceptual data exploration tools can be of great value. The following example illustrates the role an intelligent data exploration system can play in the extraction of knowledge from data.

> *The United States government maintains records of the import and export of goods from various countries of the world. The different products and raw materials are divided and subdivided into different categories. In the early 1980s the data showed a sharp decline in the import of trucks from Japan and a corresponding increase in the import of auto parts from Japan. It took several years before analysts noticed that fact and concluded that Japan was shipping the chassis and truck beds separately to the US, where they would be subsequently assembled, thereby avoiding a high US tariff on imported trucks that was directed primarily at Europe and had been on the books since World War II. When United States analysts inferred this explanation, the US and Japan commenced trade negotiations pertaining to the import of trucks.*

How much sooner would that trend have been noticed had a conceptual data exploration program been applied to the data and pointed out the opposite changes in two related categories to an analyst? How much revenue did the undiscovered truth cost the US before they could finally work out a new agreement with Japan? Noticing economic trends and patterns like the one above is a difficult task, as humans can easily get overwhelmed by the amount of data.

Based on such motivation, the analysis of economic and demographic data has become one of the focus domains for INLEN development and testing. We illustrate some of its discovery capabilities through experiments involving two similar data sets: one provided by the World Bank consisting of information on 171 countries for the period of 1965 to 1990 (in terms of 95 attributes), and one extracted from the 1993 World Factbook (published by the Central Intelligence Agency) containing several databases of information on 190 countries (in terms of 17 attributes).

10.2 Experiment 1: Integration of Multiple Operators

The World Bank data enabled us to conduct a number of experiments for testing INLEN capabilities. One experiment focused on distinguishing between development patterns in Eastern Europe and East Asia, first by identifying such patterns, and then by generating discriminant rules (Kaufman, 1994).

A conceptual clustering operator determined a way of grouping the countries, based on each country's change in the percentage of its population in the labor force between 1980 and 1990. In this classification, the typical Eastern European country and the typical East Asian country fell into separate groups. Most of the European countries had a labor force change below a threshold determined for the region by the clustering program, while most of the Asian countries had changes in labor force participation above the threshold determined for their region.

Based on this grouping, the rule learning operator (using the AQ15c inductive learning program) was called upon first in characteristic mode to characterize the Asian-like countries (those above their regional thresholds) and the European-like countries (those below their regional thresholds), and then in discriminant rule-optimizing mode to condense those characterizations into simple discriminant rules. The discriminant rules obtained were:

Country is Asian-Like if:

A.1	Change in Labor Force Participation ≥ slight_gain,	*(9 countries)*
	or	
B.1	Life Expectancy is in 60s, and	
2	Working Age Population ≤ 64%,	*(2 countries)*

Country is European-Like if:

A.1	Change in Labor Force Participation is near 0 or decreasing, and	
2	Life Expectancy is not in 60s,	*(7 countries)*
	or	
B.1	Percentage of Labor Force in Industry ≥ 40.	*(1 country)*

The rules show that of the 10 attributes in the original data set, only four attributes are instrumental in distinguishing between the European-style and Asian-style development patterns, namely *Change in Labor Force Participation*, *Life Expectancy*, *Working Age Population* and *Percentage of Labor Force in Industry*. In both the Asian- and European-Like_cases, the first rule accounted for most of the countries fitting the class, while the second one described the remainder.

This experiment demonstrated one of the cornerstone features of the methodology - an integration of different learning and discovery strategies that allows knowledge to be passed from one operator to another in a seamless way, leading to conclusions unreachable by any one individual program. It also shows that the rules created by the system are easy to understand and interpret.

10.3 Experiment 2: Detecting Anomalies in Subgroups

Another experiment with INLEN investigated the problem of detecting interesting regularities within the subgroups it creates. While the subgroups in a demographic domain may indicate that member countries or regions have something in common, notable exceptions may be exposed when a member of these constructed subsets shows a marked dissimilarity to the rest of the group. These exceptions in turn may prove to be a springboard for further discovery.

INLEN discovered several rules from the World Factbook PEOPLE database characterizing the 55 countries with low (less than 1% per year) population growth rates by invoking the rule learning operator in characteristic mode. One of the characteristic descriptions (Figure 11) had three conditions that together characterized 19 low growth countries and only one with higher population growth rates.

	Characteristic Description of Countries with Population Growth Rate below 1 per 1000 people:	Pos	Neg	Supp	Comm
1	Birth Rate = 10 to 20 or Birth Rate ≥ 50	46	20	69%	84%
2	Predominant Religion is Orthodox or Protestant or Hindu or Shinto	40	68	37%	73%
3	Net Migration Rate ≤ +20	32	104	23%	58%
	All 3 conditions:	19	1	95%	35%

Figure 11. A characterization of countries with low population growth.

In the characterization shown in Figure 11, the columns Pos and Neg respectively represent the number of positive and negative examples satisfying the condition. The *support level* (Supp) is defined as *Pos / (Pos + Neg)*, giving an indication of how much support the condition lends to the suggestion that a country's Population Growth Rate is less than 1%. The *commonality level* (Comm) is defined as *Pos / Total_Pos*, giving an indication of how commonly the condition occurs in countries with Population Growth Rates below 1% (in this example, Total_Pos = 55).

The first condition (and thus the strongest in terms of support level) states that countries with population growth rate below 1% have a low (under 20 per 1000 population) or very high (over 50 per 1000 population) birth rate. The presence of a very high birth rate in countries with low population growth is highly counterintuitive; examination of the 19 countries covered by the description pointed out that 18 had birth rates below 20, while only one, Malawi, had the high birth rate. When further attention was focused on Malawi, the explanation was clear. Malawi had a massive outward net migration rate of over 30 per 1000 population, by far the most extreme migration rate in the world. INLEN thus facilitated a discovery of a surprising exception to a normal pattern.

10.4 Experiment 3: Utilizing Structured Attributes

The rule shown in the previous example contained an attribute "predominant religion." This attribute was presented as a nominal attribute in the initial dataset. In order to examine how the structuring of attributes affects knowledge discovery, INLEN was applied to identical data sets with and without the Religion attribute being structured (Kaufman and Michalski, 1996). A portion of the attribute domain structure is shown in Figure 12.

One strong argument for structuring is that if the Predominant Religion attribute has been set up in an unstructured (nominal) manner, the statement “Predominant Religion is Lutheran” would be regarded as being as antithetical to “Predominant Religion is Christian” as it is to the statement “Predominant Religion is Buddhist,” since “Lutheran,” “Christian” and “Buddhist” are all considered equally different in a “flat” domain. This would lead to the possibility that some contradictions such as “Predominant Religion is Lutheran, but not Christian” might be generated.

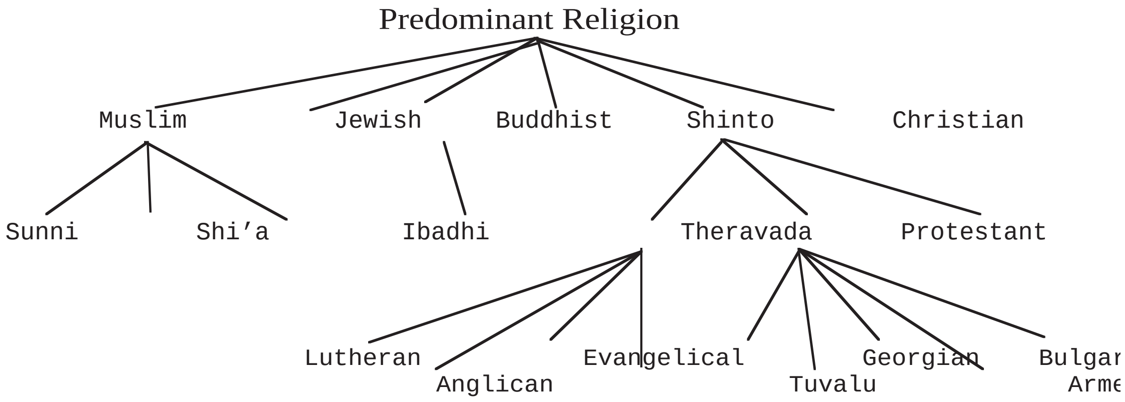


Figure 12. Part of the structure of the PEOPLE database’s *Religion* attribute.

Experiments using INLEN-2 have lent support to this and other hypotheses regarding the use of structured and non-structured attributes. Among the findings regarding their use as independent variables was that structuring attributes leads to simpler rules than when not structuring them. For example, when INLEN learned rules to distinguish the 55 countries with low population growth rate (less than 1%) from other countries, in a version of the PEOPLE database in which the attribute “Predominant Religion” was not structured, one of the rules it found was:

Population Growth Rate < 1% if: (20 examples)
1 Literacy = 95% to 99%,
2 Life Expectancy is 70 to 80 years,
3 Predominant Religion is Roman Catholic or Orthodox or Romanian or Lutheran or Evangelical or Anglican or Shinto,
4 Net Migration Rate ≤ +20 per 1000 population.

This rule was satisfied by 20 of the 55 countries with low growth rates. When the same experiment was run with “Religion” used as a structured attribute, a simpler pattern was discovered:

Population Growth Rate < 1% if: (21 examples, 1 exception)
1 Literacy = 95% to 99%,
2 Life Expectancy is 70 to 80 years,
3 Predominant Religion is Christian or Shinto,
4 Net Migration Rate ≤ +10 per 1000 population.

This rule has one exception (the United States, whose 1993 population growth rate was between 1% and 2%). If full consistency is required, the third condition could still be expressed in a simpler form than in an unstructured religion domain by performing a minimal specialization operation on the node Christian so that the rule would cover the same positive examples, but not the exception.

Similar differences were obtained by structuring dependent attributes. By arranging events into different levels of generality, rules classified them accordingly, which reduced the complexity and increased the informational significance of the rules at different levels of generalization.

These effects were especially visible at the lower levels of the hierarchy. In the unstructured dataset, five rules, each with two to five conditions, were required to define the 11 Sunni Muslim countries. The only one to describe more than two of the 11 countries was a rule with quite fragmented conditions:

Predominant Religion is Sunni_Muslim if: (4 examples)
1 Literacy ≠ 30% to 99%,
2 Infant Mortality Rate is 25 to 40 or greater than 55 per 1000 population
3 Fertility Rate is 1 to 2 or 4 to 5 or 6 to 7 per 1000 population,
4 Population Growth Rate is 1% to 3% or greater than 4%.

The value ranges in these conditions are divided into multiple segments, suggesting that this is not a strong pattern. In contrast, using a structured religion attribute, the learning operator produced two simple and easily understood patterns, each with one only condition:

Predominant Religion is Sunni_Muslim if: (10 examples, 1 exception)
1 Infant Mortality Rate ≥ 40 per 1000 population.

Predominant Religion is Sunni_Muslim if: (4 examples)
1 Birth Rate is 30 to 40 per 1000 population.

As described above, these rules apply only in the context of predominantly Islamic countries, and are based on the assumption that that determination has already been made.

10.5 Experiment 4: Applying Constructive Induction Operators

An experiment chronicled by Bloedorn and Michalski (1996) demonstrates the power of utilizing constructive induction as a knowledge discovery operator. Working from 11 economic attributes sampled over each of five consecutive years, 1986-1990 (for a total of 55 available attributes per record), the learning program attempted to discover rules to predict countries' changes in gross national product over the 5-year period. By applying three data-driven constructive induction operators—generating new attributes based on the existing attribute set, removing attributes less relevant to the goal concept, and abstracting numerical attributes into a small number of intervals—the predictive accuracy on new data increased by nearly half (from 41.7% to 60.5%).

Among the newly constructed highly relevant attributes were *Change in Energy Consumption Between 1986 and 1988*, *Ratio of Birth Rate in 1989 to Energy Consumption in 1990*, and *Average Annual Energy Consumption Over the 5-year Period*.

These results demonstrated that constructive induction can be a very useful tool for analyzing data, as it can build more adequate representation spaces for knowledge discovery.

11 Summary

The main thesis of this paper is that modern methods developed in symbolic machine learning have a direct and important application to the development of new operators for conceptual data exploration. A wide range of ideas on the applicability of various machine learning methods to this area were presented.

Two highly important operators are the construction of conceptual hierarchies (conceptual clustering), and the inductive derivation of general rules characterizing the relationship between designated output and input attributes. These rules represent high-level knowledge that can be of great value to a data analyst and directly usable in human decision-making. Other important operators include construction of equations along with logical preconditions for their application, determination of symbolic descriptions of time sequences, selection of most relevant attributes, generation of new, more relevant attributes, and selection of representative examples.

In contrast to many data mining approaches, the presented methodology requires a considerable amount of background knowledge regarding the data and the domain of discourse. This background knowledge may include, for example, a specification of the domain and the type of the attributes, the relationships among them, causal dependencies, theories about the objects or processes that generated the data, goals of the data analysis and other high-level knowledge. An important aspect of the methodology is its ability to take advantage of this knowledge.

The machine learning techniques implemented in the INLEN system allow a user to perform easily a wide range of symbolic data manipulation and knowledge generation operations. The illustrative examples demonstrate a significant potential utility of the described multistrategy methodology in solving problems of data mining and knowledge discovery.

References

Baim, P.W., “The PROMISE Method for Selecting Most Relevant Attributes for Inductive Learning Systems,” Report No. UIUCDCS-F-82-898, Department of Computer Science, University of Illinois, Urbana, 1982.

Bentrup, J.A., Mehler, G.J. and Riedesel, J.D., “INDUCE 4: A Program for Incrementally Learning Structural Descriptions From Examples,” *Reports of the Intelligent Systems Group*, ISG 87-2. UIUCDCS-F-87-958, Department of Computer Science, University of Illinois, Urbana, 1987.

Bergadano, F., Matwin, S., Michalski, R.S. and Zhang, J., “Learning Two-Tiered Descriptions of Flexible Concepts: The POSEIDON System,” *Machine Learning,* 8, pp. 5-43, 1992.

Bloedorn, E. and Michalski, R.S., "The AQ17-DCI System for Data-Driven Constructive Induction and Its Application to the Analysis of World Economics," *Proceedings of the 9th International Symposium on Methodologies for Intelligent Systems*, Zakopane, Poland, 1996.

Bloedorn, E., Wnek, J. and Michalski, R.S., "Multistrategy Constructive Induction," *Proceedings of the Second International Workshop on Multistrategy Learning*, Harpers Ferry, WV, pp. 188-203, 1993.

Bongard, N., *Pattern Recognition*, Spartan Books, New York (a translation from Russian), 1970.

Bratko, I., Muggleton, S. and Karalic, A., "Applications of Inductive Logic Programming," In Michalski, R.S., Bratko, I. and Kubat, M. (eds.), *Machine Learning and Data Mining: Methods and Applications,* London, John Wiley & Sons, 1997.

Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J., *Classification and Regression Trees*, Belmont, CA, Wadsworth Int. Group, 1984.

Carpineto, C. and Romano, G., "Some Results on Lattice-based Discovery in Databases," *Workshop on Statistics, Machine Learning and Knowledge Discovery in Databases*, Heraklion, pp. 216-221, 1995a.

Carpineto, C. and Romano, G., "Automatic Construction of Navigable Concept Networks Characterizing Text Databases," Gori, M. and Soda, G. (eds.), *Topics in Artificial Intelligence*, LNAI 992-Springer, pp. 67-78, 1995b.

Collins, A. and Michalski, R.S., "Toward a Formal Theory of Human Plausible Reasoning," *Proceedings of the Third Annual Conference of the Cognitive Science Society*, Berkeley, CA, 1981.

Collins, A. and Michalski, R.S., "The Logic of Plausible Reasoning: A Core Theory," *Cognitive Science*, 13, pp. 1-49, 1989.

Daniel, C. and Wood, F.S., *Fitting Equations to Data*, New York, John Wiley & Sons, 1980.

Davis, J., "CONVART: A Program for Constructive Induction on Time-Dependent Data," M.S. Thesis, Department of Computer Science, University of Illinois, Urbana, 1981.

Diday, E. (ed.), *Proceedings of the Conference on Data Analysis, Learning Symbolic and Numeric Knowledge*, Nova Science Publishers, Inc., Antibes, 1989.

Dieterrich, T. and Michalski, R.S., "Learning to Predict Sequences," Chapter in *Machine Learning: An Artificial Intelligence Approach Vol. 2*. Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (eds.), Morgan Kaufmann, pp. 63-106, 1986.

Dontas, K., "APPLAUSE: An Implementation of the Collins-Michalski Theory of Plausible Reasoning," M.S. Thesis, Computer Science Department, The University of Tennessee, Knoxville, TN, 1988.

Dubois, D., Prade, H. and Yager, R.R. (eds.), *Readings in Fuzzy Sets and Intelligent Systems*, Morgan Kaufmann, 1993.

Evangelos S. and Han, J. (eds.), *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Portland, OR, 1996.

Falkenhainer, B.C. and Michalski, R.S., “Integrating Quantitative and Qualitative Discovery in the ABACUS System,” In Kodratoff, Y. and Michalski, R.S. (eds.), *Machine Learning: An Artificial Intelligence Approach Vol. III*, San Mateo, CA, Morgan Kaufmann, pp. 153-190, 1990.

Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P., “Knowledge Discovery and Data Mining: Toward a Unifying Framework,” *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. Portland, OR, pp. 82-88, 1996.

Fayyad, U.M. Piatetsky-Shapiro, G. Smyth, P. and Uhturusamy, R. (eds.), *Advances in Knowledge Discovery and Data Mining*, San Mateo, CA, AAAI Press, 1996.

Feelders, A. Learning from Biased Data Using Mixture Models,” *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Portland, OR, pp. 102-107, 1996.

Forsyth, R. and Rada, R., *Machine Learning: Applications in Expert Systems and Information Retrieval*, Pittman, 1986.

Greene, G., “The Abacus.2 System for Quantitative Discovery: Using Dependencies to Discover Non-Linear Terms,” Reports *of the Machine Learning and Inference Laboratory*, MLI 88-4, Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, 1988.

Hart, A., “Experience in the Use of an Inductive System in Knowledge Engineering,” In M Bramer (Ed.), *Research and Developments in Expert Systems*, Cambridge, Cambridge University Press, 1984.

Hong, J., Mozetic, I. and Michalski, R.S., “AQ15: Incremental Learning of Attribute-Based Descriptions from Examples: The Method and User’s Guide,” *Reports of the Intelligent Systems Group*, ISG 86-5, UIUCDCS-F-86-949, Department of Computer Science, University of Illinois, Urbana, 1986.

Hunt, E., Marin, J. and Stone, P., *Experiments in Induction*, Academic Press, New York, 1966.

Imam, I.F., “Discovering Task-Oriented Decision Structures from Decision Rules,” Ph.D. dissertation, School of Information Technology and Engineering, George Mason University, Fairfax, VA, 1995.

Imam, I.F. and Michalski, R.S., “Should Decision Trees be Learned from Examples or From Decision Rules?” *Proceedings of the Seventh International Symposium on Methodologies for Intelligent Systems (ISMIS-93)*, Trondheim, Norway, 1993.

Kaufman, K., “Comparing International Development Patterns Using Multi-Operator Learning and Discovery Tools,” *Proceedings of the AAAI-94 Workshop on Knowledge Discovery in Databases*, Seattle, WA, pp. 431-440, 1994.

Kaufman, K.A. and Michalski, R.S., "EMERALD: An Integrated System of Machine Learning and Discovery Programs to Support Education and Experimental Research," *Reports of the Machine Learning and Inference Laboratory*, MLI 93-10, Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, 1993.

Kaufman, K. and Michalski, R.S., "A Method for Reasoning with Structured and Continuous Attributes in the INLEN-2 Knowledge Discovery System," *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, Portland, OR, pp. 232-237, 1996.

Kaufman, K., Michalski, R.S. and Kerschberg, L., "Mining for Knowledge in Data: Goals and General Description of the INLEN System," In Piatetsky-Shapiro, G. and Frawley, W.J. (Eds.), *Knowledge Discovery in Databases,* Menlo Park, CA: AAAI Press, pp. 449-462, 1991.

Kerber, R., "ChiMerge: Discretization of Numeric Attributes. *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, San Jose, CA, pp. 123-127, 1992.

Kodratoff, Y, *Introduction to Machine Learning*, Pittman, 1988.

Kokar, M.M., "Coper: A Methodology for Learning Invariant Functional Descriptions," Chapter in *Machine Learning: A Guide to Current Research*, Michalski, R.S., Mitchell, T.M and Carbonell, J.G. (Eds.), Kluwer Academic Publishers, Boston, MA 1986.

Kodratoff, Y. and Michalski, R.S. (eds.), *Machine Learning: An Artificial Intelligence Approach, Vol. III*. San Mateo, CA, Morgan Kaufmann., 1990

Lakshminarayan, K., Harp, S.A., Goldman, R. and Samad, T., "Imputation of Missing Data Using Machine Learning Techniques," *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Portland, OR, pp. 140-145, 1996.

Langley, P., Bradshaw G.L. and Simon, H.A., "Rediscovering Chemistry with the BACON System," *Machine Learning: An Artificial Intelligence Approach*, Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (Eds.), Morgan Kaufmann, San Mateo, CA, pp. 307-329, 1983.

Larson, J.B, "INDUCE-1: An Interactive Inductive Inference Program in VL21 Logic System," Report No. 876, Department of Computer Science, University of Illinois, Urbana, 1977.

Maloof, M.A. and Michalski, R.S, "Learning Evolving Concepts Using Partial Memory Approach," Working Notes of the 1995 AAAI Fall Symposium on Active Learning, Boston, MA, pp. 70-73, 1995.

Michael, J., "Validation, Verification and Experimentation with Abacus2," *Reports of the Machine Learning and Inference Laboratory*, MLI 91-8, Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, 1991.

Michalski, R.S., "A Planar Geometrical Model for Representing Multi-Dimensional Discrete Spaces and Multiple-Valued Logic Functions," ISG Report No. 897, Department of Computer Science, University of Illinois, Urbana, 1978.

Michalski, R.S., "Inductive Learning as Rule-Guided Generalization and Conceptual Simplification of Symbolic Descriptions: Unifying Principles and a Methodology," *Workshop on Current Developments in Machine Learning*, Carnegie Mellon University, Pittsburgh, PA, 1980.

Michalski, R.S., "A Theory and Methodology of Inductive Learning," In Michalski, R.S. Carbonell, J.G. and Mitchell, T.M. (eds.), *Machine Learning: An Artificial Intelligence Approach*, Palo Alto: Tioga Publishing, pp. 83-129, 1983.

Michalski, R.S, "Learning Flexible Concepts: Fundamental Ideas and a Method Based on Two-tiered Representation," In Kodratoff, Y. and Michalski, R.S. (eds.), *Machine Learning: An Artificial Intelligence Approach, Vol. III*, San Mateo, CA, Morgan Kaufmann, pp. 63-102, 1990.

Michalski, R.S., "Searching for Knowledge in a World Flooded with Facts," in *Applied Stochastic Models and Data Analysis*, Vol. 7, pp. 153-l63, January, 1991.

Michalski, R.S., "Inferential Theory of Learning: Developing Foundations for Multistrategy Learning," In Michalski, R.S. and Tecuci, G. (eds.), *Machine Learning: A Multistrategy Approach*, San Francisco: Morgan Kaufmann, pp. 3-61, 1994.

Michalski, R.S., Baskin, A.B. and Spackman, K.A., "A Logic-based Approach to Conceptual Database Analysis," *Sixth Annual Symposium on Computer Applications in Medical Care* (SCAMC-6), George Washington University, Medical Center, Washington, DC, pp. 792-796, 1982.

Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (eds.), *Machine Learning: An Artificial Intelligence Approach*, Palo Alto, CA, Tioga Publishing, 1983.

Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (eds.), *Machine Learning: An Artificial Intelligence Approach Vol. 2*, San Mateo, CA, Morgan Kaufmann, 1986.

Michalski, R.S. and Imam, I.F., "On Learning Decision Structures," *Fundamenta Matematicae*, Polish Academy of Sciences, 1997 (in press).

Michalski, R.S., Kaufman, K. and Wnek, J., "The AQ Family of Learning Programs: A Review of Recent Developments and an Exemplary Application," *Reports of the Machine Learning and Inference Laboratory*, MLI 91-11, Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, 1991.

Michalski, R.S., Kerschberg, L., Kaufman, K. and Ribeiro, J., "Mining for Knowledge in Databases: The INLEN Architecture, Initial Implementation and First Results," *Journal of Intelligent Information Systems: Integrating AI and Database Technologies*, 1, pp. 85-113, August 1992.

Michalski, R. S., Ko, H. and Chen, K., "SPARC/E(V.2), An Eleusis Rule Generator and Game Player", *Reports of the Intelligent Systems Group*, ISG No. 85-11, UIUCDCS-F-85-941, Department of Computer Science, University of Illinois, Urbana, 1985.

Michalski, R.S., Ko, H. and Chen, K., "Qualitative Prediction: A Method and a Program SPARC/G," Chapter in *Expert Systems*, Guetler, C. (ed.), Academic Press, Inc., London, 1986.

Michalski, R.S. and Larson, J.B., "Selection of Most Representative Training Examples and Incremental Generation of VL1 Hypotheses: The Underlying Methodology and the Description of Programs ESEL and AQ11," Report No. 867, Department of Computer Science, University of Illinois, Urbana, 1978.

Michalski, R.S. and McCormick, B.H., "Interval Generalization of Switching Theory," *Proceedings of the 3rd Annual Houston Conference on Computer and System Science*, Houston, TX, 1971.

Michalski, R.S. , Mozetic, I., Hong, J. and Lavrac, N., "The AQ15 Inductive Learning System: An Overview and Experiments," ISG Report 86-20, UIUCDCS-R-86-1260, Department of Computer Science, University of Illinois, Urbana, 1986.

Michalski, R.S., Stepp, R. and Diday, E., "A Recent Advance in Data Analysis: Clustering Objects into Classes Characterized by Conjunctive Concepts," Chapter in Kanal, L. and Rosenfeld, A. (eds.), *Progress in Pattern Recognition, Vol. 1*, pp. 33-55, 1981.

Mingers, J., "An Empirical Comparison of Selection Measures for Decision-Tree Induction," *Machine Learning,* 3, pp. 319-342, 1989.

Morgenthaler, S. and Tukey, J.W., "The Next Future of Data Analysis," In Diday, E. (ed.), *Proceedings of the Conference on Data Analysis*, *Learning Symbolic and Numeric Knowledge*, Nova Science Publishers, Inc., Antibes, 1989.

Pawlak, Z, *Rough Sets: Theoretical Aspects of Reasoning about Data,* Kluwer Academic Publishers, Dordrecht, 1991.

Quinlan, J.R., "Induction of Decision Trees," *Machine Learning*, 1, pp. 81-106, 1986.

Quinlan, J.R., "Probabilistic Decision Trees," In Y. Kodratoff and R.S. Michalski, (Eds.), *Machine Learning: An Artificial Intelligence Approach, Volume III*, Morgan Kaufmann, San Mateo, CA, pp. 140-152, 1990.

Quinlan, J.R., "*C4.5: Programs for Machine Learning*," Morgan Kaufmann, Los Altos, CA, 1993.

Reinke, R.E., "Knowledge Acquisition and Refinement Tools for the ADVISE Meta-Expert System," Master's Thesis, Department of Computer Science, University of Illinois, Urbana, 1984.

Reinke, R.E. and Michalski, R.S., "Incremental Learning of Concept Descriptions: A Method and Experimental Results," *Machine Intelligence*, 11, pp. 263-288, 1988.

Ribeiro, J., Kaufman, K. and Kerschberg, L., "Knowledge Discovery from Multiple Databases," *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, Montreal PQ, pp. 240-245, 1995.

Rosch, E., Mervis, C., Gray, W., Johnson, D. and Boyes-Braem, P., "Basic Objects in Natural Categories," *Cognitive Psychology,* 8, pp. 382-439, 1976.

Sharma, S., *Applied Multivariate Techniques*, London, John Wiley & Sons, Inc., 1996.

Slowinski, R. (ed.), *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory*, Dordrecht/Boston/London, Kluwer Academic Publishers, 1992.

Stepp, R., "A Description and User's Guide for CLUSTER/2, A Program for Conjunctive Conceptual Clustering," Report No. UIUCDCS-R-83-1084, Department of Computer Science, University of Illinois, Urbana, 1984.

Tukey, J.W., *The Collected Works of John W. Tukey, Vol. V, Philosophy and Principles of Data Analysis: 1965-1986*, Jones, L.V. (ed.), Wadsworth & Brooks/Cole, Monterey, CA, 1986.

Van Mechelen, I., Hampton, J., Michalski, R.S. and Theuns, P. (eds.), *Categories and Concepts: Theoretical Views and Inductive Data Analysis*, London, Academic Press, 1993.

Widmer, G. and Kubat, M., "Learning in the Presence of Concept Drift and Hidden Concepts," *Machine Learning*, 23, pp. 69-101, 1996.

Wnek, J., "DIAV 2.0 User Manual, Specification and Guide through the Diagrammatic Visualization System," *Reports of the Machine Learning and Inference Laboratory,* MLI 95-5, George Mason University, Fairfax, VA, 1995.

Wnek, J., Kaufman, K., Bloedorn, E. and Michalski, R.S., "Selective Induction Learning System AQ15c: The Method and User's Guide," *Reports of the Machine Learning and Inference Laboratory*, MLI 95-4, Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, 1995.

Wnek, J. and Michalski, R.S., "Hypothesis-driven Constructive Induction in AQ17-HCI: A Method and Experiments," *Machine Learning*, 14, pp. 139-168, 1994.

Wnek, J., Sarma, J., Wahab, A. and Michalski, R.S., "Comparing Learning Paradigms via Diagrammatic Visualization: A Case Study in Single Concept Learning using Symbolic, Neural Net and Genetic Algorithm Methods," *Reports of the Machine Learning and Inference Laboratory*, MLI 90-2, Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, 1990.

Zadeh, L., "Fuzzy Sets," *Information and Control*, 8, pp. 338-353, 1965.

Zagoruiko, N.G., "*Recognition Methods and Their Application*," Sovietsky Radio, Moscow (in Russian), 1972.

Zhuravlev, Y.I. and Gurevitch, I.B., "Pattern Recognition and Image Recognition," Chapter in Zhuravlev, Y.I. (ed.), *Pattern Recognition, Classification, Forecasting: Mathematical Techniques and their Application. Issue 2*, Nauka, Moscow, pp. 5-72 (in Russian), 1989.

Ziarko, W.P. (ed.), *Rough Sets, Fuzzy Sets and Knowledge Discovery*, Berlin, Springer-Verlag, 1994.

Zytkow, J.M., "Combining Many Searches in the FAHRENHEIT Discovery System," *Proceedings of the Fourth International Workshop on Machine Learning*, Irvine, CA, pp. 281-287, 1987.