1 Knowledge Discovery in Databases: An Overview

William J. Frawley, Gregory Piatetsky-Shapiro, and Christopher J. Matheus
GTE Laboratories Incorporated

Computers have promised us a fountain of wisdom
but delivered a flood of data
– A frustrated MIS executive

Abstract

This chapter presents an overview of the state of the art in research on knowledge discovery in databases. We analyze *Knowledge Discovery* and define it as the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. We then compare and contrast database, machine learning, and other approaches to discovery in data. We present a framework for knowledge discovery and examine problems in dealing with large, noisy databases, the use of domain knowledge, the role of the user in the discovery process, discovery methods, and the form and uses of discovered knowledge.

We also discuss application issues, including the variety of existing applications and propriety of discovery in social databases. We present criteria for selecting an application in a corporate environment. In conclusion, we argue that discovery in databases is both feasible and practical and outline directions for future research, which include better use of domain knowledge, efficient and incremental algorithms, interactive systems, and integration on multiple levels.

1.1 Introduction

It has been estimated that the amount of information in the world doubles every 20 months. The size and number of databases probably increases even faster. In 1989, the total number of databases in the world was estimated at five million, although most of them are small dbaseIII databases. The automation of business activities produces an ever-increasing stream of data because even simple transactions, such as a telephone call, the use of a credit card, or a medical test, are typically recorded in a computer.

Scientific and government databases are also rapidly growing. The National Aeronautics and Space Administration already has much more data than it can analyze. Earth observation satellites, planned for the 1990s, are expected to generate one terabyte (10^{15} bytes) of data every day—more than all previous missions combined. At a rate of one picture each second, it would take a person several years (working nights and weekends) just to look at the pictures generated in one day. In biology, the federally funded Human Genome project will store thousands of bytes for each of the several billion genetic bases. Closer to everyday lives, the 1990 U.S. census data of a million million bytes encode

patterns that in hidden ways describe the lifestyles and subcultures of today's United States.

What are we supposed to do with this flood of raw data? Clearly, little of it will ever be seen by human eyes. If it will be understood at all, it will have to be analyzed by computers. Although simple statistical techniques for data analysis were developed long ago, advanced techniques for intelligent data analysis are not yet mature. As a result, there is a growing gap between data generation and data understanding. At the same time, there is a growing realization and expectation that data, intelligently analyzed and presented, will be a valuable resource to be used for a competitive advantage.

The computer science community is responding to both the scientific and practical challenges presented by the need to find the knowledge adrift in the flood of data. In assessing the potential of AI technologies, Michie (1990), a leading European expert on machine learning, predicted that "the next area that is going to explode is the use of machine learning tools as a component of large-scale data analysis." A recent National Science Foundation workshop on the future of database research ranked data mining among the most promising research topics for the 1990s (Silberschatz, Stonebraker, and Ullman 1990). Some research methods are already well enough developed to have been made part of commercially available software. Several expert system shells use variations of ID3 for inducing rules from examples. Other systems use inductive, neural net, or genetic learning approaches to discover patterns in personal computer databases.

Many forward-looking companies are using these and other tools to analyze their databases for interesting and useful patterns. American Airlines searches its frequent flyer database to find its better customers, targeting them for specific marketing promotions. *Farm Journal* analyzes its subscriber database and uses advanced printing technology to custom-build hundreds of editions tailored to particular groups. Several banks, using patterns discovered in loan and credit histories, have derived better loan approval and bankruptcy prediction methods. General Motors is using a database of automobile trouble reports to derive diagnostic expert systems for various models. Packaged-goods manufacturers are searching the supermarket scanner data to measure the effects of their promotions and to look for shopping patterns.

A combination of business and research interests has produced increasing demands for, as well as increased activity to provide, tools and techniques for discovery in databases. This book is the first to bring together leading-edge research from around the world on this topic. It spans many different approaches to discovery, including inductive learning, Bayesian statistics, semantic query optimization, knowledge acquisition for expert systems, information theory, and fuzzy sets. The book is aimed at those interested or involved in computer science and the management of data, to both inform and inspire further research and applications. It will be of particular interest to professionals work-

ing with databases and management information systems and to those applying machine learning to real-world problems.

1.2   What Is Knowledge Discovery?

The chapters in this collection vary widely in their focus and content, reflecting the immense diversity of current research on knowledge discovery in databases. To provide a point of reference for these works, we begin here by defining and explaining relevant terms.

1.2.1   Definition of Knowledge Discovery

Knowledge discovery is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. Given a set of facts (data) F, a language L, and some measure of certainty C, we define a *pattern* as a statement S in L that describes relationships among a subset F_S of F with a certainty c, such that S is simpler (in some sense) than the enumeration of all facts in F_S. A pattern that is interesting (according to a user-imposed interest measure) and certain enough (again according to the user's criteria) is called *knowledge*. The output of a program that monitors the set of facts in a database and produces patterns in this sense is *discovered knowledge*.

These definitions about the language, the certainty, and the simplicity and interestingness measures are intentionally vague to cover a wide variety of approaches. Collectively, these terms capture our view of the fundamental characteristics of discovery in databases. In the following paragraphs, we summarize the connotations of these terms and suggest their relevance to the problem of knowledge discovery in databases.

Patterns and languages: Although many different types of information can be discovered in data, this book focuses on patterns that are expressed in a high-level language, such as

> If Age < 25 and Driver-Education-Course = No
> Then At-fault-accident = Yes
> With Likelihood = 0.2 to 0.3.

Such patterns can be understood and used directly by people, or they can be the input to another program, for example, an expert system or a semantic query optimizer. We do not consider low-level patterns, such as those generated by neural networks.

Certainty: Seldom is a piece of discovered knowledge true across all the data. Representing and conveying the degree of certainty is essential to determining how much faith the system or user should put into a discovery. As we examine later, certainty involves

several factors, including the integrity of the data; the size of the sample on which the discovery was performed; and, possibly, the degree of support from available domain knowledge. Without sufficient certainty, patterns become unjustified and, thus, fail to be knowledge.

Interesting: Although numerous patterns can be extracted from any database, only those deemed to be interesting in some way are considered knowledge. Patterns are interesting when they are novel, useful, and nontrivial to compute. Whether a pattern is novel depends on the assumed frame of reference, which can be either the scope of the system's knowledge or the scope of the user's knowledge. For example, a system might discover the following: If At-fault- accident = yes Then Age > 16. To the system, this piece of knowledge might be previously unknown and potentially useful; to a user trying to analyze insurance claims records, this pattern would be tautological and uninteresting and would not represent discovered knowledge. This example also suggests the notion of utility. Knowledge is useful when it can help achieve a goal of the system or the user. Patterns completely unrelated to current goals are of little use and do not constitute knowledge within the given situation. For example, a pattern relating at-fault-accident to a driver's age, discovered while the user's intent was to analyze sales figures, would not be useful to the user.

Novelty and utility alone, however, are not enough to qualify a pattern as discovered knowledge. Most databases contain numerous novel and useful patterns, such as the total sales for 1990, the average cost of an insurance claim, and the maximum intensity of a spectral line. These types of patterns would not typically be considered knowledge because they are trivial to compute. To be nontrivial, a system must do more than blindly compute statistics; the results of the directed calculation of straightforward statistics are, to our way of thinking, readily available to the database user. A discovery system must be capable of deciding which calculations to perform and whether the results are interesting enough to constitute knowledge in the current context. Another way of viewing this notion of nontriviality is that a discovery system must possess some degree of autonomy in processing the data and evaluating its results.

Efficiency: Finally, we are interested in discovery processes that can be efficiently implemented on a computer. An algorithm is considered efficient [1] if the run time and space used are a polynomial function of low degree of the input length. The problem of discovery of interesting sentences (concepts) that satisfy given facts is inherently hard.

---

[1] One of the earliest discovery processes was encountered by Jonathan Swift's Gulliver in his visit to the Academy of Labado. The "Project for improving speculative Knowledge by practical and mechanical operations" was generating sequences of words by random permutations and "where they found three or four Words that might make Part of a Sentence, they dictated them to . . . Scribes." This process, although promising to produce many interesting sentences in the (very) long run, is rather inefficient and was recently proved to be NP-hard.

Recent advances in computational learning theory (Valiant 1984; Haussler 1988) have shown that it is not possible to efficiently learn an arbitrary Boolean concept; the problem is NP-hard. However, these results are generally related to the worst-case performance of algorithms and do not eliminate the possibility that, on the average, we can find complex concepts fast. Efficient algorithms do exist for restricted concept classes, such as purely conjunctive concepts (for example, $A \wedge B \wedge C$), or the conjunction of clauses made up of disjunctions of no more than k literals (for example, $(A \vee B) \wedge (C \vee D) \wedge (E \vee F)$, for $k = 2$). Another possibility for efficiently finding concepts is to abandon the demand that the algorithm learn a desired concept with some guarantee and instead accept heuristic or approximate algorithms.

To summarize, knowledge discovery in databases exhibits four main characteristics:

- High-Level Language—Discovered knowledge is represented in a high-level language. It need not be directly used by humans, but its expression should be understandable by human users.

- Accuracy—Discoveries accurately portray the contents of the database. The extent to which this portrayal is imperfect is expressed by measures of certainty.

- Interesting Results—Discovered knowledge is interesting according to user-defined biases. In particular, being interesting implies that patterns are novel and potentially useful, and the discovery process is nontrivial.

- Efficiency—The discovery process is efficient. Running times for large-sized databases are predictable and acceptable.

The systems and techniques described in the chapters of this book generally strive to satisfy these characteristics. The approaches taken, however, are quite diverse. Most are based on machine learning methods that have been enhanced to better deal with issues particular to discovery in databases. Next, we briefly define database concepts and terminology and relate them to terms common to machine learning.

1.2.2   Databases and Machine Learning

In database management, a *database* is a logically integrated collection of data maintained in one or more files and organized to facilitate the efficient storage, modification, and retrieval of related information. In a relational database, for example, data are organized into files or tables of fixed-length records. Each record is an ordered list of values, one value for each field. Information about each field's name and potential values is maintained in a separate file called a *data dictionary*. A *database management system* is a collection of procedures for retrieving, storing, and manipulating data within databases.

Table 1.1
Translations between Database Management and Machine Learning Terms.

Database Management	Machine Learning
database: a logically integrated collection of dynamic files	a fixed set of examples
file	database, data set, set of instances
tuple, record	instance, example, feature vector
field, attribute	feature, attribute
field domain	possible field values
data dictionary	field type and domain information
relational data	a set of instances
object-oriented, structured data	relational data
logical condition	concept description

In machine learning, the term database typically refers to a collection of instances or examples maintained in a single file.[2] *Instances* are usually fixed-length feature vectors. Information is sometimes also provided about the feature names and value ranges, as in a data dictionary. A learning algorithm takes the data set and its accompanying information as input and returns a statement (for example, a concept) representing the results of the learning as output.

Table 1.1 informally compares the terminology in database management with that of machine learning. With the appropriate translation of terms, it seems that machine learning could be readily applied to databases: Rather than learning on a set of instances, learning is done on a file of records from a database. Knowledge discovery in databases, however, raises additional concerns that extend beyond those typically encountered in machine learning. In the real world, databases are often dynamic, incomplete, noisy, and much larger than typical machine learning data sets (table 1.2). These factors render most learning algorithms ineffective in the general case. Not surprisingly, much of the work on discovery in databases focuses on overcoming these complications.

1.2.3 Related Approaches

Although machine learning is the foundation for much of the work in this area, knowledge discovery in databases deals with issues relevant to several other fields, including database management, expert systems, statistical analysis, and scientific discovery.

---

[2]We are assuming a rather narrow view of machine learning—that is, supervised and unsupervised inductive learning from examples. See Carbonell, Michalski, and Mitchell (1983) for a broader and more detailed view.

Table 1.2
Conflicting Viewpoints between Database Management and Machine Learning.

Database Management	Machine Learning
Database is an active, evolving entity	Database is just a static collection of data
Records may contain missing or erroneous information	Instances are usually complete and noise-free
A typical field is numeric	A typical feature is binary
A database typically contains millions of records	Data sets typically contain several hundred instances
AI should get down to reality	"Databases" is a solved problem and is therefore uninteresting

Database management: A *database management* system provides procedures for storing, accessing, and modifying the data. Typical operations include retrieval, update, or deletion of all tuples satisfying a specific condition, and maintaining user-specified integrity constraints. The ability to extract tuples satisfying a common condition is like discovery in its ability to produce interesting and useful statements (for example, "Bob and Dave sold fewer widgets this year than last"). These techniques, however, cannot by themselves determine what computations are worth trying, nor do they evaluate the quality of the derived patterns. Interesting discoveries uncovered by these data-manipulation tools result from the guidance of the user. However, the new generation of deductive and object-oriented database systems (Kim, Nicolas, and Nishio 1990) will provide improved capabilities for intelligent data analysis and discovery.

Expert systems: *Expert systems* attempt to capture knowledge pertinent to a specific problem. Techniques exist for helping to extract knowledge from experts. One such method is the induction of rules from expert-generated examples of problem solutions. This method differs from discovery in databases in that the expert examples are usually of much higher quality than the data in databases, and they usually cover only the important cases (see also Gaines, chapter 29, for a comparison between knowledge acquisition from an expert and induction from data). Furthermore, experts are available to confirm the validity and usefulness of the discovered patterns. As with database-management tools, the autonomy of discovery is lacking in these methods.

Statistics: Although statistics provide a solid theoretical foundation for the problem of data analysis, a purely statistical approach is not enough. First, standard statistical methods are ill suited for the nominal and structured data types found in many databases (figure 1.2). Second, statistics are totally data driven, precluding the use of

available domain knowledge, an important issue that we discuss later. Third, the results of statistical analysis can be overwhelming and difficult to interpret. Finally, statistical methods require the guidance of the user to specify where and how to analyze the data. However, some recent statistics-based techniques such as projection pursuit (Huber 1985) and discovery of causal structure from data (Glymour et al. 1987; Geiger, Paz, and Pearl 1990) address some of these problems and are much closer to intelligent data analysis. We expect that methods using domain knowledge will be developed by the statistical community. We also believe that statistics should have a vital role in all discovery systems dealing with large amounts of data.

Scientific discovery: Discovery in databases is significantly different from scientific discovery (see also Żytkow and Baker, chapter 2) in that the former is less purposeful and less controlled. Scientific data come from experiments designed to eliminate the effects of all but a few parameters and to emphasize the variation of one or a few target parameters to be explained. However, typical business databases record a plethora of information about their subjects to meet a number of organizational goals. This richness (or confusion) both captures and hides from view underlying relationships in the data. Moreover, scientists can reformulate and rerun their experiments should they find that the initial design was inadequate. Database managers rarely have the luxury of redesigning their data fields and recollecting the data.

1.2.4   A Framework for Knowledge Discovery

We have referred to discovery systems several times without specifying what a discovery system is. Although discovery systems vary considerably in their design, it is possible to describe a prototypical discovery system. Figure 1.1 depicts the basic components of our prototypical system for knowledge discovery in databases. At the core of the system is the discovery method, which computes and evaluates patterns on their way to becoming knowledge. The inputs to the discovery method include raw data from the database, information from the data dictionary, additional domain knowledge, and a set of user-defined biases that provide high-level focus. The output is discovered knowledge that can be directed to the user or back into the system as new domain knowledge. Because this model represents a prototypical system, a discovery system need not include all these aspects. The feedback of discovered knowledge into the store of domain knowledge, for example, is present in few existing systems (see Future Directions).

In the remainder of this chapter, we explore each aspect of this framework in detail. Specifically, we consider (1) the peculiarities of databases, (2) the use of domain knowledge, (3) the role of the user in the discovery process, (4) methods of knowledge discovery, and (5) the form and uses of discovered knowledge.

1.3   Database Issues

The fundamental input to a discovery system is the raw data present in a database. We previously mentioned that databases pose unique problems to discovery not typically confronted in machine learning. In particular, these problems arise from the fact that real-world databases are dynamic, incomplete, noisy, and large. Other concerns include whether the database contains adequate information for interesting discovery and how to deal with the overabundance of irrelevant information.

Dynamic data: A fundamental characteristic of most databases is that their contents are ever changing. Data can be time sensitive, and discovery is affected by the timeliness of data observations. Some data values, such as a patient's social security number, are constant over time; some vary more or less gradually over time (for example, weight and height); and some are so situation dependent that only a recently observed value will suffice (for example, pulse rate).

Irrelevant fields: Another key characteristic is the relevance of data, that is, whether an item of data is relevant to the current focus of discovery. When a patient database is being explored for interesting patterns of symptoms and diagnoses, nonmedical data, such as patient name or zip code, are irrelevant, and errors there are unimportant. On the other hand, pulse rate, a simple and typically recorded medical observation, is relevant, and errors here can affect what is discovered. If, however, we are looking for a geographic concentration of a particular disease, then a correct zip code becomes crucial. If the zip code is thought to be faulty, it can sometimes be inferred from related information, such as the patient's address.

An aspect somewhat related to relevance is the applicability of an attribute to a subset of the database; for example, a patient's pregnant field is not applicable to men (the class of patients with sex equal to male), but it is essential to know for female patients of childbearing age.

Missing values: The presence or absence of values for relevant data attributes can affect discovery. Whether a patient was comatose at the time of diagnosis can be so important that it does not allow the substitution of a default value; less important missing data can be defaulted. In an interactive system, the absence of an important datum can spawn a request for its value or a test to determine it. Alternatively, the absence of data can be dealt with as if it were a condition in itself, and the missing attribute can be assigned a neutral value, such as unknown. For example, the absence of some patient measurements might be found to imply that the patient was formerly in excellent health.

Noise and uncertainty: For relevant attributes, the severity of error can depend on the data type of the allowed values. Values of different attributes can be real numbers
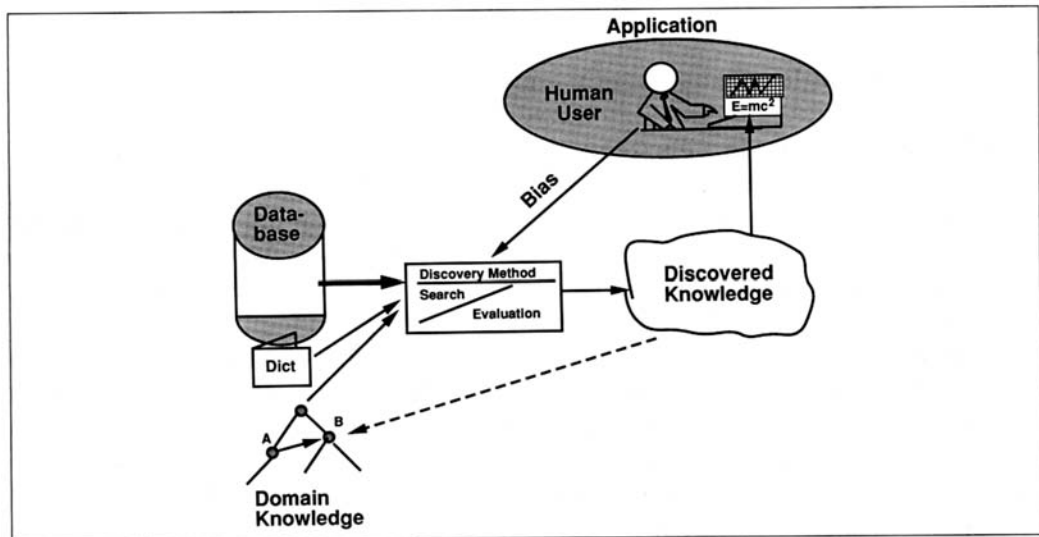
Figure 1.1
A Framework for Knowledge Discovery in Databases.

(charge), integer numbers (age), or strings (name), or can belong to a set of nominal values (patient type). Nominal values can be partially or completely ordered and can also have a semantic structure (figure 1.2).

Another aspect of uncertainty is the inherent or expected exactitude of data, that is the degree of noise in the data. Repeated measurements cluster around an average; based on a priori analysis or computations on the measurements themselves, a statistical model describing randomness is formulated and used to define expected and tolerable deviations in the data. Subjective input, such as whether a patient's condition is severe or moderate, can vary considerably about one or more cluster points. Often, statistical models are applied in an ad hoc manner to subjectively determined attributes to obtain gross statistics and judge the acceptability of (combinations of) attribute values.

Especially with regard to numeric data types, data precision can be a factor in discovery. For example, it is regarded as adequate to record body temperature to a precision of 0.1 degree. A sensitive trend analysis of body temperature would require even greater precision in the data. To a discovery system able to relate such trends to diagnosis, this imprecision would appear to be noise in the input.

Missing fields: An inadequate view of the database can make valid data appear to be in error. The database view is the totality of usable attributes or accessors that the discovery system can apply to a problem. It is assumed that the attributes differentiate cases of interest. When they don't, there appears to be some error. Suppose, for example, that a system is tasked to learn to diagnose malaria from a patient database that does

not include the red blood cell count. Patients whose records are correct and who are medically identical with respect to this given view might have different diagnoses, which, in turn, might be incorrectly blamed on data error.

1.4   Databases and Knowledge

> Ignorance is the curse of God
> Knowledge the wing wherewith we fly to heaven.
> – *William Shakespeare.*

A database is a logically integrated collection of files. Associated with a database is a data dictionary, which defines field names, the allowable data types for field values, various constraints on field values, and other related information. For example, the field *age* is a positive integer and the field *date-of-birth* has the form *MMDDYY*. Types of constraints include ranges of possible values for numeric fields, lists of possible values for nominal fields, or more complex conditions such as a department must always have exactly one manager. In a sense, the data dictionary defines the syntax of database use.

The database contains the raw data to be processed by the discovery system. In practice, the discovery system must also use the additional information about the form of data and constraints on it. Some of this information can be stored in the data dictionary, but other information might exist in manuals or experts' heads. For example, a discovery system for a hospital database needs to know which diagnosis codes (DX) are grouped into which diagnostic-related groups (DRG), which, in turn, are grouped into major diagnostic categories (MDC) (figure 1.2). Then, if a patient's DX does not match his or her DRG, an unusual side effect or an error in record keeping is indicated.

Because discovery is computationally expensive, additional knowledge regarding the form and content of data, the domain(s) described by the database, the context of a particular discovery episode, and the purposes being served by discovery are often used to guide and constrain the search for interesting knowledge. We refer to this form of information as *domain knowledge* or *background knowledge*. Domain knowledge assists discovery by focusing search. However, its use is controversial because by telling a system what to look for and where to look for it, domain knowledge restricts search and can deliberately rule out valuable discovery. An example discussed at the IJCAI-89 Knowledge Discovery Workshop illustrates the trade-off between quickly finding conventional solutions and discarding unusual ones. In logistics planning, the search space is so large that it is impossible to find solutions without using constraints such as "trucks don't drive over water." This constraint, however, eliminates potentially interesting solutions, such as those in which trucks drive over frozen lakes in winter. Of equal importance to
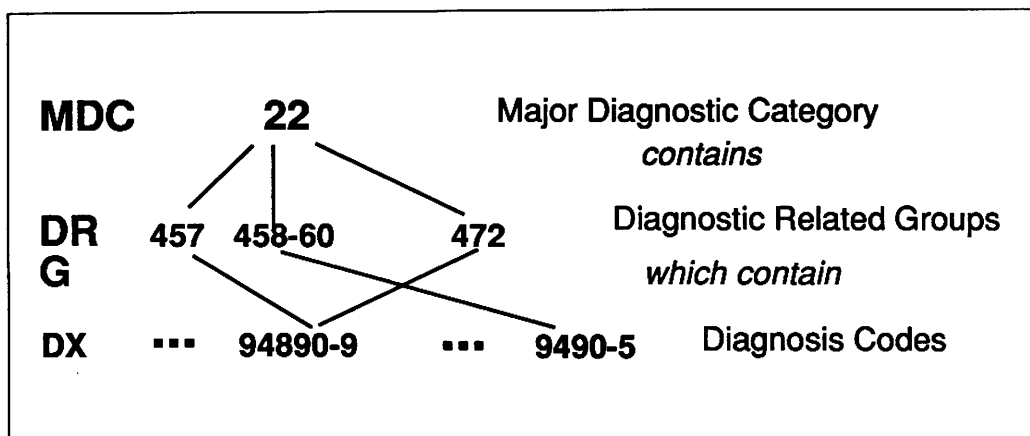
Figure 1.2
Data Dictionary: Relationships Between Fields and Values

this debate is the accuracy and timeliness of domain knowledge.

Background knowledge can take on a number of different forms. Data dictionary knowledge is the most basic and least controversial form of domain knowledge. Inter-field knowledge, such as weight and height being positively correlated, and inter-instance knowledge, such as knowing that federal regulation requires the largest charge for hand-icapped customers be no more than the average charge for regular customers, are closely related to the data but move toward the semantics of the domain. According to data dictionary knowledge, one shouldn't add age to hair-color, but it's a matter of domain knowledge that taking the sum of age and seniority makes sense, but adding age to weight does not.

Knowledge about the content of the database can also help make the discovered knowledge more meaningful to the end user. Domain knowledge is usually provided by a domain expert, although it is possible for domain knowledge to be discovered, suggesting a bootstrapping approach.

1.5 Discovered Knowledge

This section examines three important facets of discovered knowledge: its form, its representation, and its degree of certainty.

1.5.1 Form

The form of a discovered knowledge can be categorized by the type of data pattern it describes. *Interfield patterns* relate values of fields in the same record (for example procedure = surgery implies days-in-hospital > 5). *Inter-record patterns* relate values

aggregated over groups of records (for example, diabetic patients have twice as many complications as nondiabetics) or identify useful clusters, such as the group of profit-making companies. Discovery of inter-record patterns is a form of data summarization (part 4). In time-dependent data, inter-record relationships can also identify interesting trends (for example, produce-sales are up 20 percent over last year's sales).

We can also categorize the form of a discovery by its descriptive capacity. A quantitative discovery relates numeric field values using mathematical equations (see part 1). A qualitative discovery (see parts 2 and 3) finds a logical relationship among fields. We distinguish these two forms because different discovery techniques are often used in each case. For example, linear quantitative relationships are conveniently found using linear regression methods that are inappropriate for qualitative discoveries.

Qualitative and quantitative discoveries are often expressed as simple rules, such as $X > Y$, or A implies B. Discoveries, however, can take on more complex forms. Putting several simple implications together forms a causal chain or network. The discovery of relationships among simpler rules can lead to semantic models or domain theories. Models and theories of this sort imply complex relationships that can require applying of the discovery process to previous, simpler discoveries. We might refer to this discovery form as an interdiscovery discovery (see Future Directions).

1.5.2 Representation

Discoveries must be represented in a form appropriate for the intended user. For human end users, appropriate representations include natural language, formal logics, and visual depictions of information. Discoveries are sometimes intended for other computer programs, such as expert system shells; in this case, appropriate representations include programming languages and declarative formalisms. A third case arises when the intended user of the discovered knowledge is the discovery system itself. In these situations where discoveries are fed back into the system as domain knowledge, domain knowledge and discovered knowledge must share a common representation.

Natural language is often desirable from a human perspective, but it is not convenient for manipulation by discovery algorithms. Logical representations are more natural for computation and, if necessary, can be translated into a natural language form. Common logic representations include formalisms such as production rules (for example, if X, then Y), relational patterns ($X > Y$), decision trees (equivalent to ordered lists of rules), and semantic or causal networks. These representational forms offer different advantages and limitations. An appropriate choice for a discovery system depends on the expected knowledge complexity and the need for human comprehension.

For humans, some information is best presented visually (Tufte 1990). The relationships among the branches of a decision tree, for example, are more evident in a graphic

presentation than as a logical representation of nested conditional statements. The same is usually true for semantic and causal networks in which the global structure of the network relationships is more clearly depicted in a diagram. Information about the shape and density of clusters of records is another type of knowledge that is best presented visually. In this case, two- or three-dimensional plots can convey certain information more concisely and clearly than any logical representation.

1.5.3   Uncertainty

Patterns in data are often probabilistic rather than certain. This situation can result from missing or erroneous data, or it can reflect the inherent indeterminism of the underlying real-world causes. Capturing this sort of probabilistic information requires a method for representing uncertainty. One common technique is to augment logical representations with probabilistic weights that indicate probabilities of success, belief measures, or standard deviations (see chapters 6, 7, and 8). Alternatively, linguistic uncertainty measures, such as fuzzy sets, are sometimes used (see Yager, chapter 20). In visual presentations, probabilistic information can be readily conveyed by size, density, and shading.

The easiest way to deal with error and uncertainty is not to have any. For example, the occurrence of noisy, missing, and inapplicable data can be minimized by rigidly applying standardized protocols in data entry. Another approach is to assume that the data are absolutely correct and terminate any calculation that encounters error (see Ziarko, chapter 11). Presuming that erroneous data cannot lead to a valid result, the remaining discoveries are regarded as highly reliable. Of course, this presumption eliminates the possibility of discovering nearly certain or probable knowledge.

Uncertainty cannot be ignored when the patterns of interest are inherently probabilistic; for example, "there's a 50 percent chance of rain tomorrow." Fortunately, databases are typically large enough for statistical analysis to determine these probabilities. In some situations, data values can be modeled as representing true information corrupted by random noise. The uncertainty in discovered knowledge can then be represented in terms of a derived probability distribution. For example, for a discovered rule, a product distribution is derived from the conjuncts making up its left-hand side and the correlation to the success of the right-hand side. More simply, a discovered rule can be matched against all entries in the database to accrue an overall success rate. The same approach applies to more complex structures, such as decision trees or classification hierarchies: Match each entry to the ordered nodes, and count the successes and failures.

When databases are very large, with records in the millions, complete analysis of all the data is infeasible. Discovery algorithms must then rely on some form of sampling, whereby only a portion of the data is considered. The resulting discoveries in these cases are necessarily uncertain. Statistical techniques, however, can measure the degree of

uncertainty (see Piatetsky-Shapiro, chapter 13, for one approach toward estimating the accuracy of rules discovered from a sample). They can also be used to determine how much additional sampling would be required to achieve a desired level of confidence in the results.

1.6   Discovery Algorithms

Discovery algorithms are procedures designed to extract knowledge from data. This activity involves two processes: identifying interesting patterns and describing them in a concise and meaningful manner. The identification process categorizes or clusters records into subclasses that reflect patterns inherent in the data. The descriptive process, in turn, summarizes relevant qualities of the identified classes. In machine learning, these two processes are sometimes referred to as unsupervised and supervised learning respectively. In discovery systems, user supervision can occur in either process or, in the ideal case, can be completely absent.

1.6.1   Pattern Identification

One way to look at a pattern is as a collection or class of records sharing something in common: customers with incomes over \$25,000, patients between 20 and 30 years old, or questionable insurance claims. Discovering pattern classes is a problem of pattern identification or clustering. There are two basic approaches to this problem: traditional numeric methods and conceptual clustering.

Traditional methods of clustering come from cluster analysis and mathematical taxonomy (Dunn and Everitt 1982). These algorithms produce classes that maximize similarity within classes but minimize similarity between classes. Various measures of similarity have been proposed, most based on Euclidean measures of distance between numeric attributes. Consequently, these algorithms only work well on numeric data. An additional drawback is their inability to use background information, such as knowledge about likely cluster shapes.

Conceptual clustering attempts to overcome these problems. These methods work with nominal and structured data and determine clusters not only by attribute similarity but also by conceptual cohesiveness, as defined by background information. Recent examples of this approach include AutoClass (Cheeseman et al. 1988), the Bayesian Categorizer (Anderson and Matessa 1990), Cluster, and Cobweb (Fisher 1987).

Although useful under the right conditions, these methods do not always equal the human ability to identify useful clusters, especially when dimensionality is low and visualization is possible. This situation has prompted the development of *interactive clus-*

tering algorithms that combine the computer's computational powers with the human user's knowledge and visual skills.

1.6.2   Concept Description

Once identified, useful pattern classes usually need to be described rather than simply enumerated. In machine learning, this process is known as supervised concept learning from examples: Given a set of objects labeled by class, derive an intensional description of the classes. Empirical learning algorithms, the most common approach to this problem, work by identifying commonalities or differences among class members. Well-known examples of this approach include decision tree inducers (Quinlan 1986), neural networks (Rummelhart and McClelland 1986), and genetic algorithms (Holland et al. 1986).

The main drawback to empirical methods is their inability to use available domain knowledge. This failing can result in descriptions that encode obvious or trivial relationships among class members. For example, a description of the class of pregnant patients that includes the term *sex=female* would be empirically accurate but would not provide any new information to a hospital administrator. Some learning approaches, such as explanation-based learning (Mitchell, Keller, and Kedar-Cabelli 1986), require a set of domain knowledge (called a domain theory) for use in explaining why an object falls into a particular class. Other approaches combine empirical and knowledge-based methods. Discovery in large, complex databases clearly requires both empirical methods to detect the statistical regularity of patterns and knowledge-based approaches to incorporate available domain knowledge.

1.6.3   Discovery Tasks

Discovery is performed for various reasons. The appropriateness of a specific discovery algorithm depends on the discovery task. One task we've already mentioned is class identification or clustering. For the process of concept description, we can identify at least three additional tasks:

- Summarization: Summarize class records by describing their common or characteristic features. Example: A summary of sales representatives with increased earnings last quarter might include that they are all from the Midwest and drive blue cars.

- Discrimination: Describe qualities sufficient to discriminate records of one class from another. Example: To determine whether a salesperson is from the Midwest it might be sufficient to look at the color or his or her car: if the color is blue, the salesperson is from the Midwest; otherwise the salesperson is from the East or West Coast.

- Comparison: Describe the class in a way that facilitates comparison and analysis with other records. Example: A prototypical Midwest salesperson might own a blue car, have increased sales, and average 100 phone calls a week. This description might serve as the basis against which individual salespeople are judged.

Because these different tasks require different forms and amounts of information, they often influence discovery algorithm selection or design. For example, a decision tree algorithm produces a description intended for discriminating between class instances that might exclude characteristic class qualities.

1.6.4 Complexity

Discovery algorithms for large databases must deal with the issue of computational complexity. Algorithms with computational requirements that grow faster than a small polynomial in the number of records and fields are too inefficient for large databases. Empirical methods are often overwhelmed by large quantities of data and potential patterns. The incorporation of domain knowledge can improve efficiency by narrowing the focus of the discovery process but at the risk of precluding unexpected but useful discovery. Data sampling is another way of attacking the problem of scale; it trades a degree of certainty for greater efficiency by limiting discovery to a subset of the database (see previous section on uncertainty).

1.7 Application Issues

This section starts by listing areas where discovery in databases has been applied, then discusses the key features that characterize a suitable application in a corporate environment, and ends with a cautionary note regarding the ethics of uncovering hidden knowledge in data about people.

1.7.1 Applications of Discovery in Databases

A decade ago, there were only a few examples of discovery in real data. The notable ones include discovery of mass spectrometry rules by MetaDendral (Buchanan and Mitchell 1978), new diagnostic rules for soybean disease (Michalski and Chilausky 1980), and drug side effects in a rheumatism patient database (Blum 1982).

Since this time, the discovery approach has been tried in many more domains, including those given in the following list. This list is by no means exhaustive and is meant to give representative examples for the kinds of applications where discovery in databases is possible. The largest databases used for discovery had several millions of records, and larger ones are being considered.

- Medicine: biomedicine, drug side effects, hospital cost containment, genetic sequence analysis, and prediction

- Finance: credit approval, bankruptcy prediction, stock market prediction, securities fraud detection, detection of unauthorized access to credit data, mutual fund selection

- Agriculture: soybean and tomato disease classification

- Social: demographic data, voting trends, election results

- Marketing and Sales: identification of socio-economic subgroups showing unusual behavior, retail shopping patterns, product analysis, frequent flying patterns, sales prediction

- Insurance: detection of fraudulent and excessive claims, claims "unbundling." Of course, all insurance data analysis can be considered a form of knowledge discovery in databases.

- Engineering: automotive diagnostic expert systems, Hubble space telescope, computer-aided design (CAD) databases, job estimates

- Physics and Chemistry: electrochemistry, superconductivity research

- Military: intelligence analysis, data fusion, and . . . (the rest is classified)

- Law Enforcement: tax and welfare fraud, fingerprint matching, recovery of stolen cars

- Space Science: astronomy, search for extraterrestrial intelligence, space data analysis (this will become more important as huge amounts of data are gathered in future space missions)

- Publishing: custom editions of journals.

1.7.2   Selecting an Application in a Corporate Environment

As with any emerging technology, it is important to carefully select the initial applications for discovery in databases. We developed a list of criteria for selecting an application in a corporate environment. The criteria will be different for a university or a government project, but many important considerations will be the same. The knowledge tree in figure 1.3 illustrates our criteria.

The fruits of knowledge growing on the tree of data are not easy to pick. To get there, we need to climb a multistep ladder. The first step is the business need for discovery. The discovered knowledge should have the potential for significant and measurable financial benefit. There should be many unknown patterns in data that cannot be found easily by

conventional statistical methods. It is helpful if the problem is typical, but the need to solve unique problems with high payoff also exists (for example, location of human gene activators).

The second step is having sufficient and reliable data. Having at least 1,000 examples is desirable. The portion of the data that is incomplete or noisy should be relatively small. Most fields relevant to the discovery focus should be stored in the database. For example, it is hard to find likely potential customers for a mobile phone without information about a customer's car.

The next step is having the organizational support for the project. There should be an enthusiastic and influential supporter in the organization and the commitment for (potentially) long-term research.

The final step is to have significant but incomplete domain knowledge. The best chance for discovery is with things we almost but not quite know already. It is desirable to have user-specified hierarchies of field-value codes and rules relating possible field values. Domain knowledge should be codified and computer readable.

After climbing these steps, we can discover some useful knowledge (possibly, including something controversial) and receive the desired business payoff. We might also reach the research value cloud. The research value is high for complex applications where existing methods are insufficient. Such applications are characterized by a variety of data types, including numeric, nominal, and structured fields; a very large database (millions of records); noisy, incomplete, and contradictory data; and complex domain knowledge. Research value is high for difficult problems and, thus, is frequently in conflict with the business need to have quick results.

1.7.3   Will Discovery Open a Pandora's Box?

An important issue to consider in analyzing social or demographic databases is the appropriateness of discovery. A careless approach to discovery can open a Pandora's box of unpleasant surprises. Some kinds of discovery are actually illegal: Federal and state privacy laws limit what can be discovered about individuals. The use of drug traffickers' profiles by law enforcement agencies has been controversial, and use of some parts of the profile, such as race, have been ruled illegal. Political, ethical, and moral considerations can affect other discoveries. A Federal Bureau of Investigation proposal to establish a nationwide database of criminal suspects was dropped after congressional objections about possible invasion of privacy. Advanced algorithms for discovery in databases might also become a threat to database security (see O'Leary, chapter 30).

A pattern that involves racial or ethnic characteristics is likely to be controversial. The plans to market the Uptown cigarette, a campaign that was directed at young, black males (devised, possibly, after market research identified this group as smoking more
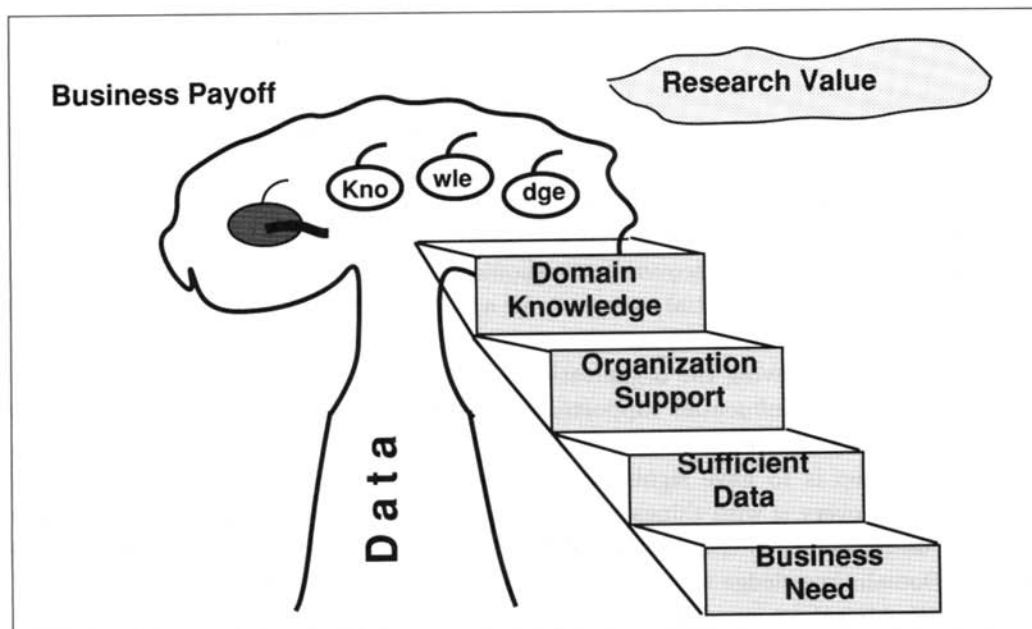
Figure 1.3
Selecting Application for Discovery in Databases in a Corporate Environment.

than average) was shelved after many protests (Wildavsky 1990). Another example is the Food and Drug Administration ban on blood donations by people from Haiti and Sub-Saharan Africa. The discovered pattern of the high incidence of AIDS in these groups was protested as being racially motivated because there was also a high incidence of AIDS in another geographically defined group, that is men from New York and San Francisco, whose members were not forbidden to donate blood. However, reports on this controversy did not make clear what the strength or coverage of these patterns was and what additional factors were considered. In such cases, it is desirable to give more detailed information, such as "Based on a sample of size S, people in group Z have 33 to 41 percent likelihood of developing the disease X. The nationwide risk of developing X is 10 to 12 percent." The public then has a clearer notion of the nature of discovered assertions and is better able to make an informed decision about them, even if they still remain controversial.

1.8   Future Directions

Although some aspects of discovery in databases, such as finding simple formulas to fit scientific data or inducing decision trees for classification, are relatively well under-

stood, many more aspects are in need of research. This research will not only be driven by academic considerations but also by the practical need to analyze more data that are more complex than ever before, including object-oriented, CAD-CAM, textual, and multimedia databases. Data complexity will make it necessary to use more domain knowledge. Much larger databases will require more efficient algorithms. Dealing with a fast-changing environment will demand much more incremental methods. Complex problems such as network control might require the integration of multiple approaches to discovery. The results should be presented to users in more understandable ways, using interactive approaches. Finally, discovery in social, business, and demographic databases requires caution about findings that may be illegal or unethical. The following paragraphs examine these directions in detail.

Domain knowledge can be used in all aspects of automated discovery, from data representation to selecting interesting features to making the discovered results more understandable. Using domain knowledge constraints to reduce search, however, is controversial, as illustrated by the "trucks don't drive over water" example mentioned earlier. The challenge is to use domain knowledge in such a way that we don't block the discovery of unexpected solutions. The use of domain knowledge taken to the extreme will produce a specialized learning algorithm (see Part 5, Domain-Specific Discovery Methods) that will outperform any general method in its domain but will not be useful outside it. A desirable compromise is to develop a framework for augmenting the general method with the specific domain knowledge. Some recent attempts are described in Quinlan (1990).

Efficient algorithms will be crucial. Exponential and even high-order polynomial algorithms will not scale up for dealing with large volumes of data. Although the efficient discovery of arbitrary rules is not possible in general (assuming $P \neq NP$), this problem can be sidestepped by using restricted rule types, heuristic and approximate algorithms, and careful use of sampling. Discovery algorithms should also use the latest advances in hardware and software. Parallel computers, large memories, and object-oriented and deductive databases not only can speed up existing methods but also present opportunities for new, faster algorithms.

Incremental methods are needed to efficiently keep pace with changes in data. More importantly, incremental discovery systems that can reuse their discoveries can bootstrap themselves and make more complex discoveries possible.

Interactive systems will provide, perhaps, the best opportunity for discovery in the near term. In such systems, a knowledge analyst is included in the discovery loop. This approach combines the best features of human and machine: Use human judgment but rely on the machine to do search and to crunch numbers. The interactive approach requires the discovered knowledge to be presented in a human-oriented form, whether as written reports (Schmitz, Armstrong, and Little 1990) or visual and sound patterns

(Smith, Bergeron, and Grinstein 1990). Such novel output presentations might allow the use of the phenomenal perceptual capabilities of humans. Tools need to be built to support effective interaction between the user and the discovery system. Also, algorithms need to be reexamined from the viewpoint of human-oriented presentation. A neural network, for example, might have to generate explanations from its weights (Gallant 1988).

Integration on many levels will be required for future systems. Accessing existing databases and data dictionaries, they will combine different types of learning and discovery. The results of discovery will not stand alone but feed back for more discoveries or feed forward to other systems. Recent examples of integrated learning systems are Silver et al. (1990) and Part 6, Integrated and Multiparadigm Systems. The incremental, knowledge-based, and interactive discovery methods may transform the static databases of today into evolving information systems of tomorrow.

1.9   Organization of the Book

A number of candidate themes and issues could be used to organize the following chapters. One is the nature of what is discovered and how it is expressed. Another is the treatment of data uncertainty. Also crucial is the user-supplied bias that determines what is interesting and novel during the discovery. There is also the controversial issue of whether and how to use domain knowledge. None of these issues provides a neat, linear progression describing ongoing research. For the following twenty-nine chapters, grouped into seven parts, we chose the path that leads from the general to the specific: from less reliance on domain knowledge to more, from less restricted forms of discovered knowledge to more specialized, from general methods to domain-specific techniques. Here is a quick overview.

1.9.1   Discovery of Quantitative Laws—Part 1

Four chapters explore domain-independent function-finding discovery programs whose purpose is to automate the discovery of numeric laws of the type commonly found in physics and chemistry. Here, the domain knowledge is used only indirectly by restricting results to a class of allowable functional forms.

Chapter 2, by J. M. Żytkow and J. Baker, describes how a numeric function-finding system Fahrenheit, designed for scientific discovery, was adapted to database exploration.

The next chapter, by Y.-H. Wu and S. Wang, uses the Reduction algorithm, which discovers a multivariate formula by finding its components from a predefined set of primitive functions.

Chapter 4, by E. P. D. Pednault, presents an approach to inferring probabilistic theories from low-dimensional data by using the minimum description length principle and shows a successful application to computer vision.

C. Schaffer, in chapter 5, outlines a framework for evaluating the systems that discover functional relationships in scientific data.

1.9.2   Data-Driven Discovery of Qualitative Laws—Part 2

These six chapters focus on the discovery of qualitative laws using techniques with little or no domain knowledge. These laws are represented as rules, decision trees, patterns, or dependency tables.

In chapter 6, K. C. C. Chan and A. K. C. Wong describe a statistical approach to extracting classificatory knowledge from data, capable of handling noisy data with mixed continuous- and discrete-valued fields.

Chapter 7, by D. K. Y. Chiu, A. K. C. Wong, and B. Cheung, presents an information-based method for the discovery of patterns using a hierarchical, maximum-entropy scheme that iteratively partitions the event space by minimizing the loss of information.

In chapter 8, R. Uthurusamy, U. M. Fayyad, and S. Spangler address the difficult problem of deciding when to stop specialization during top-down decision tree generation and present an algorithm that produced good results on a real-world diagnostic database.

In chapter 9, P. Smyth and R. M. Goodman present an information-based method for finding the best probabilistic rules relating a given set of discrete-valued domain variables.

J. Hong and C. Mao (chapter 10) present a method for incremental discovery of rules and structure through hierarchical and parallel clustering.

W. Ziarko, in chapter 11, describes how to find exact dependencies in databases, represented as the minimal subsets of data sufficient for uniquely predicting the target attribute.

1.9.3   Using Knowledge in Discovery—Part 3

In this part, six chapters explore the mechanisms by which the user can provide discovery-directing knowledge, including hierarchical structures, rule-interest measures, Horn clauses, and functions.

Y. Cai, N. Cercone, and J. Han (chapter 12) use attribute-oriented induction to extract characteristic rules and classification rules from relational databases. Their method performs generalization of field values guided by a concept tree and integrates database operations with the learning process.

In addressing the discovery of strong rules in databases, G. Piatetsky-Shapiro (chapter

13) discusses principles applicable to rule-interest functions, presents a fast algorithm for rule discovery, and analyzes the expected accuracy of discovered rules.

To overcome the practical limitations of conventional statistical classification methods, Q. Wu, P. Suetens, and A. Oosterlinck (chapter 14) examine the use of domain hierarchical classification structure to guide search and Bayesian classification; they show positive results in a biomedical domain.

In chapter 15, W. J. Frawley encodes both domain knowledge and contextual information as functions to be used as the building blocks of decision trees. This representation allows disjunctive concepts discovered in one pass over a database to be used in building subsequent trees.

F. Bergadano, A. Giordana, L. Saitta, F. Brancadori, and D. De Marchi (chapter 16) discuss their system to learn structured knowledge using both induction from examples and deduction from a logically specified domain theory.

In chapter 17, M. Manago and Y. Kodratoff show how the use of frames to represent examples extends the applicability of traditional machine learning induction to discovery in object-oriented databases.

1.9.4   Data Summarization—Part 4

Unlike the previous parts, which deal mostly with finding patterns that apply to an individual instance (interfield dependencies), these three chapters describe methods for finding patterns relating groups of instances (inter-record dependencies).

Chapter 18, by M. C. Chen and L. McNamee, shows how to adapt a decision tree algorithm for estimating the value of a summarized attribute and the cardinality of the queried category.

Chapter 19, by P. Hoschka and W. Klösegen, describes a system for searching for high-level, interesting statements that summarize subsets of data, with applications to marketing and vote analysis.

Finally, in chapter 20, R. Yager introduces an approach to linguistic data summarization using fuzzy set theory.

1.9.5   Domain-Specific Discovery Methods—Part 5

At one extreme of the knowledge versus data spectrum are methods whose domain models and representations determine and direct the discovery process. Each of the domains addressed by these four chapters is general enough to encompass a broad range of practical problems.

In the future, discovery methods will be used on a wealth of information now stored as text and in specialized databases. C.-S. Ai, P. E. Blower, and R. A. Ledwith (chapter

21) extract chemical reaction information from a full-text database and create a new database, understandable to humans and readable by computer, whose structured data objects represent chemical reactions described in the text.

A. J. Gonzalez, H. R. Myler, M. Towhidnejad, F. D. McKenzie, and R. R. Kladke (chapter 22) are working to create a complete knowledge base directly and automatically from computer-resident data sources. Their prototype system deals with CAD database files and produces a frame-based knowledge base.

In chapter 23, the experimental domain of the research of J. C, Schlimmer, T. M. Mitchell, and J. McDermott is the maintenance of large knowledge bases. They describe an approach and a prototype computer program that accepts rules from a knowledge base developer, attempts to justify the behavior of each rule, and then utilizes these justifications to suggest refinements to the rules.

In chapter 24, M. Siegel, E. Sciore, and S. Salveter, experimenting with a shipping database provided by Lloyd's of London, define a method for deriving and maintaining a set of rules for semantic query optimization by using intermediate results from the optimization process to direct the search for new rules.

1.9.6   Integrated and Multiparadigm Systems—Part 6

In the near future, we expect increased interest in multiparadigmatic discovery systems. Here, two chapters focus on the integration of techniques and the use of a number of modes of knowledge representation within a single system.

In chapter 25, B. Silverman, M. R. Hieb, and T. M. Mezher present an approach to discovery in the absence of human experts or trainers for domains for which large simulators exist.

K. A. Kaufman, R. S. Michalski, and L. Kerschberg (chapter 26) describe a system that combines database, knowledge base, and machine learning techniques to provide a user with an integrated set of tools for conceptually analyzing data and searching it for interesting relationships.

1.9.7   Methodology and Application Issues—Part 7

The final four chapters compare knowledge discovery methods, analyze the role of knowledge discovery in developing knowledge bases, and address the issue of database security.

J. W. Long, E. H. Irani, and J. R. Slagle are part of a data analysis effort associated with a major multiclinic clinical trial. As seen in chapter 27, their attempts to discover structural relationships in data are appropriate to databases in which it is unlikely for data to be missing or incorrect.

In chapter 28, M. McLeish, P. Yao, M. Garg, and T. Stirtzinger report on the appli-

cation of a number of machine learning and statistical methods to veterinary medicine databases, showing that diagnostic systems based on these methods can outperform domain experts.

Chapter 29, by B. R. Gaines, provides a quantitative framework for knowledge acquisition that encompasses machine learning and expertise transfer as related paradigms within a single spectrum and shows that techniques supporting the development of practical knowledge-based systems generally lie between these extremes.

In the concluding chapter, D. E. O'Leary addresses the database security aspects of knowledge discovery, in particular, unauthorized discovery from databases and the need to develop new database security controls. New controls can arise from the technology of knowledge discovery itself.

References

Anderson, J. R. and Matessa, M. 1990. A Rational Analysis of Categorization. In Proceedings of the Seventh International Machine Learning Conference, 76–84.

Blum, R. 1982. Discovery and Representation of Causal Relationships from a Large Time-Oriented Clinical Database: The RX Project. In *Lecture Notes in Medical Informatics*, no. 19. New York: Springer-Verlag.

Buchanan, B., and Mitchell, T. 1978. Model-Directed Learning of Production Rules. In *Pattern-Directed Inference Systems*, eds. D. A. Waterman and F. Hayes-Roth, 297–312. New York: Academic Press.

Carbonell, J. G.; Michalski, R. S.; and Mitchell, T. M. 1983. An Overview of Machine Learning. In *Machine Learning: An Artificial Intelligence Approach*, volume 1, eds. R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, 3–23. San Mateo, Calif.: Morgan Kaufmann.

Cheeseman, P.; Kelly, J.; Self, M.; Sutz, J.; Taylor, W.; and Freeman, D. 1988. AutoClass: a Bayesian Classification System. In Proceedings of the Fifth International Conference on Machine Learning. 54–64. San Mateo, Calif.: Morgan Kaufmann.

Dunn, G., and Everitt, B. S. 1982. *An Introduction to Mathematical Taxonomy.* Cambridge, Mass.: Cambridge University Press.

Fisher, D. H. 1987. Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning* 2(2): 139–172.

Gallant, S. I. 1988. Connectionist Expert Systems. *Communications of the ACM* 31(2): 153–168.

Geiger, D.; Paz, A.; and Pearl, J. 1990. Learning Causal Trees from Dependence Information. In Proceedings of the Eighth National Conference on Artificial Intelligence, 771–776. Menlo Park, Calif.: AAAI.

Glymour, C.; Scheines, R.; Spirtes, P.; and Kelly, K. 1987. *Discovering Causal Structure.* San Diego, Calif.: Academic Press.

Haussler, D. 1988. Quantifying Inductive Bias: AI Learning Algorithms and Valiant's Learning Framework. *Artificial Intelligence* 36(2): 177–221.

Holland, J. H.; Holyoak, K. J.; Nisbett, R. E.; and Thagard, P. R. 1986. *Induction: Processes of Inference, Learning, and Discovery.* Cambridge, Mass: MIT Press.

Huber, P. J. 1985. Projection Pursuit. In *The Annals of Statistics* 13(2): 435–474.

Kim, W.; Nicolas, J.-M.; and Nishio, S. 1990. *Deductive and Object-Oriented Databases.* Amsterdam: Elsevier.

Michalski, R. S.; and Chilausky, R. L. 1980. Learning by Being Told and Learning from Examples: An Experimental Comparison of the Two Methods of Knowledge Acquisition in the Context of Developing an Expert System for Soybean Disease Diagnosis. *International Journal of Policy Analysis and Information Systems* 4(2): 125–161.

Michie, D., 1990. March 15 Interview. *AI Week* 7(6): 7–12.

Mitchell, T. M.; Keller, R. M.; and Kedar-Cabelli; S. T. 1986. Explanation-Based Generalization: A Unifying View. *Machine Learning*, 1(1), 47–80.

Nordhausen, B., and Langley, P. 1990. An Integrated Approach to Empirical Discovery. In *Computational Models of Discovery and Theory Formation*, eds. J. Shrager and P. Langley, 97–128. San Mateo, Calif.: Morgan Kaufmann.

Quinlan, J. R. 1990. Learning Logical Definitions from Relations. *Machine Learning* 5(3): 239–266.

Quinlan J. R. 1986. Induction of Decision Trees. *Machine Learning*, 1(1): 81–106.

Rummelhart, D. E., and McClelland, J. L. 1986. *Parallel Distributed Processing, Volume 1.* Cambridge, Mass: MIT Press.

Schmitz, J.; Armstrong, G.; and Little, J. D. C. 1990. CoverStory—Automated News Finding in Marketing. In *DSS Transactions*, ed. L. Volino, 46–54. Providence, R.I.: Institute of Management Sciences.

Silberschatz, A.; Stonebraker, M.; and Ullman, J., 1990. Database Systems: Achievements and Opportunities, The "Lagunita" Report of the NSF Invitational Workshop, TR-90-22, Dept. of Computer Science, Univ. of Texas at Austin.

Silver, B.; Frawley, W.; Iba, G.; Vittal, J.; and Bradford, K. 1990. ILS: A Framework for Multi-Paradigmatic Learning", In Proceedings of the Seventh International Conference on Machine Learning, 348–356. San Mateo, Calif.: Morgan Kaufmann.

Smith, S.; Bergeron, D.; and Grinstein, G. 1990. Stereophonic and Surface Sound Generation for Exploratory Data Analysis. In Proceedings of the Conference of the Special Interest Group in Computer and Human Interaction (SIGCHI), 125–131. Reading, Mass.: Addison-Wesley.

Tufte, E. R. 1990. *Envisioning Information*, Cheshire Conn.: Graphics Press.

Valiant, L. G. 1984. A Theory of the Learnable, *Communications of the ACM* 27(11): 1134–42.

Wildavsky, B. 1990. Tilting at Billboards. *New Republic* August 20 and 27, 19–20.