*D. Braha (ed.), Data Mining for Design and Manufacturing, pp. 1-40,*
*2001 Kluwer Academic Publishers*

# DATA MINING: AN INTRODUCTION

Ishwar K. Sethi
Intelligent Information Engineering Laboratory
Department of Computer Science and Engineering
Oakland University
Rochester, MI 48309
isethi@oakland.edu

## Abstract

This chapter provides an introductory overview of data mining. Data mining, also referred to as knowledge discovery in databases, is concerned with nontrivial extraction of implicit, previously unknown and potentially useful information from data in databases. The main focus of the chapter is on different data mining methodologies and their relative strengths and weaknesses.

## INTRODUCTION

In order to understand data mining let us first consider the distinction between data, information, and knowledge through an example. Suppose having completed weekly grocery shopping you are at your local grocery store in one of the checkout lanes. When your grocery goes past the checkout scanner, data is being captured. When your grocery store looks at the accumulated data for different customers at some point in time and finds certain grocery shopping patterns, the captured data is transformed into *information*. It is important to note that whatever it is that converts data into information resides external to the data and in the interpretation. When we have a very high degree of certainty or validity about information, we refer to it as *knowledge*. Continuing with our example, if the grocery shopping patterns discovered by our local grocery store are found to hold at many other grocery stores also, we have a situation where data is finally transformed into knowledge. Thus, we see that information and knowledge are both derived from data.

The modern technology of computers, networks, and sensors has made data collection an almost effortless task. Consequently, data is being captured and stored at a phenomenal pace. However, the captured data needs to be converted into information and knowledge to become useful. Traditionally, analysts have performed the task of extracting information and knowledge from recorded data; however, the increasing volume of data in modern industrial and business enterprises calls for computer-based methods for this task. Such methods have come to be known as *data mining* methods and the entire process of applying computer-based methodology is known as *knowledge discovery*. Note that this data mining viewpoint does not impose any restriction on the

nature of the underlying computer data analysis tools. This is the viewpoint that is held by most of the vendors of data mining products. However, some people, especially those belonging to the artificial intelligence community, have a slightly narrower definition for data mining. According to their viewpoint, the underlying data analysis tools must be based on one or more sub-technologies of artificial intelligence, for example machine learning, neural networks, or pattern recognition, to qualify as the data mining method.

## Importance In Business Decision Making

Data mining technology is currently a hot favorite in the hands of decision-makers as it can provide valuable hidden business intelligence from historical corporate data. It should be remembered, however, that fundamentally, data mining is not a new technology. The concept of extracting information and knowledge discovery from recorded data is a well-established concept in scientific and medical studies. What is new is the convergence of several factors that have created a unique opportunity for data mining in the corporate world. Businesses are suddenly realizing that the data that they have been collecting for the past 15-20 years can give them an immense competitive edge. Due to the client-server paradigm, data warehousing technology, and the currently available immense desktop computing power, it has become very easy for an end-user to look at stored data from all sorts of perspectives and extract valuable business intelligence. Data mining is being used to perform market segmentation to launch new products and services as well as to match existing products and services to customers' needs. In the banking, healthcare, and insurance industries, data mining is being used to detect fraudulent behavior by tracking spending and claims patterns.

## Data Classification

One can classify data into three classes: (1) *structured data*, (2) *semi-structured data*, and (3) *unstructured data*. Most business databases contain structured data consisting of well-defined fields of numeric or alphanumeric values. Semi-structured data has partial structure. Examples of semi-structured data are electronic images of business and technical documents, medical reports, executive summaries, and repair manuals. The majority of web documents also fall in this category. An example of unstructured data is a video recorded by a surveillance camera in a departmental store. Such visual or multimedia recordings of events or processes of interests are currently gaining widespread popularity due to reducing hardware costs. This form of data generally requires an extensive amount of processing to extract contained information. Structured data is often referred to as *traditional data* while the semi and unstructured data are lumped together as *non-traditional data*. Because of the presence of structure in it, traditional data has no ambiguity. On the other hand, non-traditional data is difficult to interpret and often has multiple interpretations. Most of the current data mining methods and commercial tools are meant for traditional data; however, development of data mining tools for non-traditional data is growing at a rapid rate.

Another way of looking at data, especially traditional data, is to look at the behavior of recorded attributes with respect to time. Certain attributes, for example a customer's social security number, do not change with time. A database containing only such kinds of records is considered to have *static data*. On the other hand, there are attributes, for example a customer's monthly utility consumption, that change with time. A database containing such records is considered to have *dynamic* or *temporal data* as well. The majority of the data mining methods are more suitable for static data and special consideration is often required to mine dynamic data.

## DATA MINING AND DATA WAREHOUSING

In this section, the relationship between data warehousing and data mining is addressed. Although the existence of a data warehouse is not a prerequisite for data mining, in practice the task of data mining is made a lot easier by having access to a data warehouse.

A data warehouse can be viewed as a data repository for an organization set up to support strategic decision-making. The architecture and the features of a data warehouse are very different from other databases in an organization, which are operational databases designed to support day-to-day operations of an organization. The function of a data warehouse is to store historical data of an organization in an integrated manner to reflect the various facets of the organization's business. The data in a warehouse is never updated but used only to respond to queries from end-users, who are generally the decision-makers. This is in contrast with the users of operational databases, whose interaction with the database consists of either reading some records from it or updating them. Unlike operational databases, data warehouses are huge in size storing billions of records. In many instances, an organization may have several local or departmental data warehouses. Such data warehouses (see Figure 1) are often called *data marts* due to their smaller size.

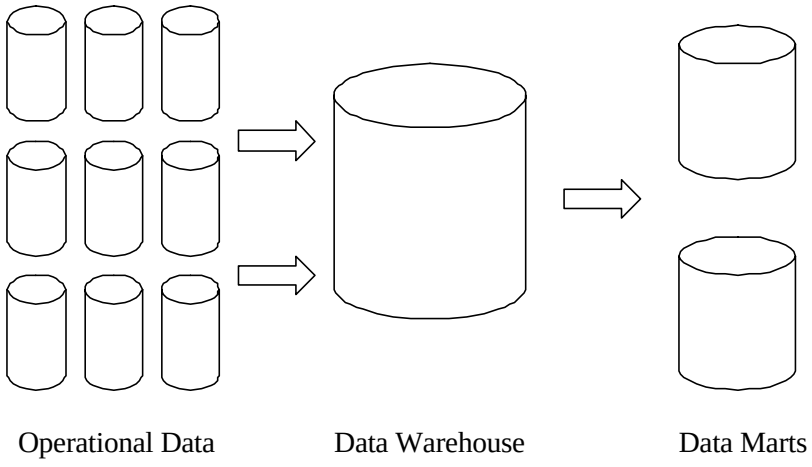


Figure 1. Operational databases, a data warehouse, and data marts

The link between data mining and data warehousing is a mutually reinforcing link. It is difficult to say whether the prospects of an informed and focused decision-making through data mining are responsible for a surge in industry-wide interest in data warehousing; or whether the availability of clean, well-formatted historical data in warehouses is the cause of recent boom in data mining. In any case, data mining is one of the major applications for data warehousing since the sole function of a data warehouse is to provide information to end-users for decision support. Unlike other query tools explained later, the data mining tools provide an end-user with a capability to extract hidden information. Such information, although more difficult to extract, can provide a bigger business advantage and yield higher returns on data warehousing investments for a company.

## DATA MINING AND QUERY TOOLS

All databases provide a variety of query tools for users to access information stored in the database. For ease of operation, these query tools generally provide a graphical interface to users to express their queries. In the case of relational databases, the query tools are known as *SQL* tools because of the use of *Structured Query Language* to query the database. In the case of dimensional databases, the query tools are popularly known as the *on-line analytical processing* (*OLAP*) tools. Following the viewpoint that data mining is concerned with the extraction of information and knowledge from databases, one obvious question to raise is "How is data mining different from structured query language (SQL) and on-line analytical processing (OLAP) tools?" In this section, we try to provide an answer to this question.

### Data Mining and SQL

SQL is the standard language for relational database management systems. SQL is good for queries that impose some kind of constraint on data in the database to extract an answer. In contrast, data mining is good for queries that are exploratory in nature. For example, if we want to obtain a list of all utility customers whose monthly utility bill is greater than some specified dollar amount, we can get this information from our database using SQL. However, SQL is not a very convenient tool if we want to obtain the differences in customers whose monthly bills are always paid on time with those customers who are usually late in their payments. It is not that this question cannot be possibly answered by SQL. By several trial and error steps, perhaps one can arrive at the answer through SQL. Data mining is, on the other hand, very good at finding answers to the latter types of questions. Thus, we can say that SQL is useful for extracting obvious information, i.e. shallow knowledge, from a database but data mining is needed to extract not so obvious, i.e. hidden, information from a database. In other words, SQL is useful when we know exactly what we are looking for; but we need data mining when we know only vaguely what we are looking for. Thus, SQL and data mining are complementary and both are needed to extract information from databases.

## Data Mining and OLAP

OLAP tools have become very popular in recent years as they let users play with data stored in a warehouse by providing multiple views of the data. In these views, different dimensions correspond to different business characteristics, e.g. sales, geographic locations, product types etc. OLAP tools make it very easy to look at dimensional data from any angle or to "slice-and-dice" it. For example, it is easy to answer questions like "How have increased advertising expenditures impacted sales in a particular territory?" with OLAP tools. To provide such answers, OLAP tools store data in a special format which corresponds to a multi-dimensional hyper-box structure. Although OLAP tools like data mining tools provide answers that are derived from data, the similarity between the two sets of tools ends here. The derivation of answers from data in OLAP is analogous to calculations in a spreadsheet; OLAP tools do not learn from data; nor do they create new knowledge. They are simply special-purpose visualization tools that can help an end-user learn patterns in the data. In contrast, the data mining tools obtain their answers via learning the relationships between different attributes of database records. Often, these discovered relationships lead to creation of new knowledge by providing new insights to business. Thus, OLAP tools are useful for data mining because of their capabilities to visualize relationships between different data dimensions; however, they are not a substitute for data mining.

# THE DATA MINING PROCESS

The data mining process consists of several stages and the overall process is inherently interactive and iterative [Fayyad et al, 1996]. The main stages of the data mining process are: (1) domain understanding; (2) data selection; (3) cleaning and preprocessing; (4) discovering patterns; (5) interpretation; and (6) reporting and using discovered knowledge.

## Domain Understanding

The domain understanding stage requires learning the business goals of the data mining application as well as gathering relevant prior knowledge. Blind application of data mining techniques without the requisite domain knowledge often leads to the discovery of irrelevant or meaningless patterns. This stage is best executed by a team of business and information technology persons to develop an all-around understanding of the data mining task being undertaken.

## Data Selection

The data selection stage requires the user to target a database or select a subset of fields or data records to be used for data mining. Having a proper domain understanding at this stage helps in the identification of useful data. Sometimes, a business may not have all the requisite data in-house. In

such instances, data is purchased from outside sources. Examples of data often purchased from outside vendors include demographic data and life-style data. Some applications of data mining also require data to be obtained via surveys.

## Cleaning and Preprocessing

This is the most time-consuming stage of the entire data mining process. Data is never clean and in the form suitable for data mining. The following are typical of data corruption problems in business databases:

Duplication - This kind of data corruption occurs when a record, for example a customer's purchases, appears several times in a database. It is one of the most common data corruption problems found in databases of businesses, such as direct mailers and credit card companies, dealing with individual customers. Misspelling due to typing/entry errors generally causes this kind of corruption. Sometimes customers are known to misspell deliberately to avoid linkage with their own past records.

Missing Data Fields - Missing fields are present in a database due to a variety of reasons. For example, a customer may simply get tired of filling in the desired information; or a missing field may be caused by a data entry error with an improper entry for a field. Filling in the missing values is generally a non-trivial task. Often, the records with missing fields are ignored for further processing.

Outliers - An outlier is a data value in a field, which is very different from the rest of the data values in the same field. The presence of outliers in a database is generally due to incorrect recordings of outlier fields. In many instances, outliers are easy to spot by considering domain consistency constraints. Sometimes, an outlier may be present due to exceptional circumstances such as a stolen credit card, when considering the monthly expenditure field in a credit card database. Detection of such outliers requires a considerable effort and often such a detection step itself is called the *data discovery step*.

## Preprocessing

The preprocessing step of the third stage of data mining involves integrating data from different sources and making choices about representing or coding certain data fields that serve as inputs to the data discovery stage. Such representation choices are needed because certain fields may contain data at levels of details not considered suitable for the data discovery stage. For example, it may be counter-productive to represent the actual date of birth of each customer to the data discovery stage. Instead, it may be better to group customers into different age groups. Similarly, the data discovery stage may get overwhelmed by looking at each customer's address and may not generate useful patterns. On the other hand, grouping customers on a geographical basis may produce better results. It is important to remember that the preprocessing step is a crucial step.

The representation choices made at this stage have a great bearing on the kinds of the patterns that will be discovered by the next stage of data discovery.

## Discovering Patterns

The data-pattern-discovery stage is the heart of the entire data mining process. It is the stage where the hidden patterns and trends in the data are actually uncovered. In the academic or research literature, it is only this stage that is referred to as data mining. There are several approaches to the pattern discovery stage. These include association, classification, clustering, regression, sequence analysis, and visualization. Each of these approaches can be implemented through one of several competing methodologies, such as statistical data analysis, machine learning, neural networks, and pattern recognition. It is because of the use of methodologies from several disciplines that data mining is often viewed as a multidisciplinary field (see Figure 2). Here, we will provide details about different approaches to pattern discovery. The details about different methodologies will be presented in the next section.

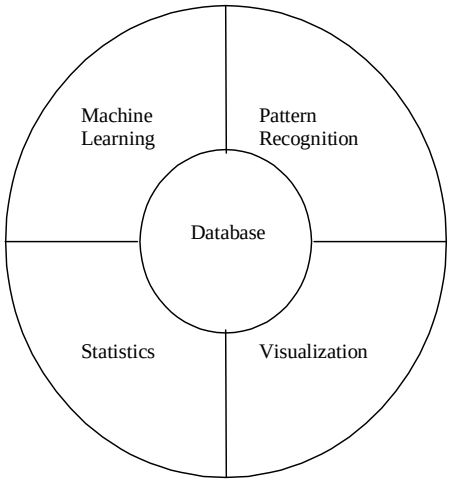


Figure 2. Core technologies for data mining

Association - This approach to data discovery seeks to establish associative relationships between different items in database records. The association approach is very popular among marketing managers and retailers who find associative patterns like “90% of customers who buy product X also buy product Y” extremely helpful for market targeting and product placement in stores. The association approach to data discovery is successful when one has an idea of different associations that are being sought out. This is because one can find all kinds of correlations in a large database. Statistics, machine learning, and neural networks are popular methodologies for the association approach to data discovery.

Classification - The classification approach to data discovery is perhaps the most widely used approach. It consists of classifying records into two or more pre-determined classes. As an example, consider a utility company planning to offer an appliance service plan to its customers. To get maximum response for a planned telemarketing effort, the utility may want to classify its customers into two classes – customers likely to respond and customers not likely to respond. Once such a classification is done, the telemarketers can concentrate on only those customers that fall in the first category. The application of the classification approach requires a classification rule, which is generally extracted from an existing set of pre-classified records. Such a set of records is often termed as a *training set* and the process of extracting the classification rule is commonly known as *learning*. Decision tree classifiers and neural network classifiers are two of the most popular methodologies for implementing the classification approach to data discovery.

Clustering - Clustering implies data grouping or partitioning. This approach to data discovery is used in those situations where a training set of pre-classified records is unavailable. The major applications of clustering are in market segmentation and mining of customers' response data. Clustering is also known as *exploratory data analysis* (EDA). Other terms for clustering are *unsupervised learning* and *self-organization*. Performing clustering with a known number of groupings is relatively easy in comparison with those situations when the number of groups is not known a-priori and must be determined by the clustering process itself. Clustering is a very popular data analysis tool and statistical pattern recognition, neural networks, and fuzzy logic offer a variety of clustering algorithms.

Sequence Analysis - This approach is used for discovering patterns in time-series data. For example, we could use this approach to determine the buying patterns of credit-card customers to predict their future purchases. Such predictive information can be used for identifying stolen credit cards. Sequence analysis is also used to establish associations over time. For example, it can be used to find patterns like “80% of customers who buy product X are likely to buy product Y in the next six months.” This allows marketers to target specific products and services that the customers are more likely to buy. The popular methodologies for sequence analysis are rooted in statistics and neural networks.

Visualization - The visualization approach to data mining is based on an assumption that human beings are very good at perceiving structure in visual forms. This approach thus consists of providing the user with a set of visualization tools to display data in various forms. The user while viewing the visualized data makes the actual discovery of the patterns in the data. An extreme of the visualization approach to data mining consists of creating an immersive virtual reality (VR) environment so that a user can move through this environment discovering hidden relations.

## Interpretation

The interpretation stage of the data mining process is used by the user to evaluate the quality of

discovery and its value to determine whether previous stages should be revisited or not. Proper domain understanding is crucial at this stage to put a value on discovered patterns.

## Reporting

The final stage of the data mining process consists of reporting and putting to use the discovered knowledge to generate new actions or products and services or marketing strategies. Without this step, the full benefits from data mining cannot be realized. Reporting can take many forms, including detailed graphical presentation of the patterns discovered and the transfer of the mined knowledge or model to the appropriate business application.

# DATA MINING METHODOLOGIES

This section presents basic concepts of different data mining methodologies. We present a few selected techniques from different methodologies. From statistical data analysis methods, we describe linear regression, logistic regression, linear discriminant analysis, and clustering techniques. From pattern recognition, we focus mainly on nearest neighbor classification. After presenting the basic neuron model, we describe briefly multiple-layer feed-forward network and self-organization feature map methods from neural network methodology of data mining. From machine learning, we describe decision tree methods and genetic algorithms.

Many different types of variables or attributes, i.e. fields in a database record, are common in data mining. Not all of the data mining methods are equally good at dealing with different types of variables. Therefore, we first explain the different types of variables and attributes to help readers determine the most suitable methodology for a given data mining application.

## Types of Variables

There are several ways of characterizing variables. One way of looking at a variable is to see whether it is an *independent* variable or a *dependent* variable, i.e. a variable whose value depends upon values of other variables. Another way of looking at a variable is to see whether it is a *discrete* variable or a *continuous* variable. Discrete variables are also called *qualitative* variables. Such variables are measured or defined using two kinds of non-metric scales – *nominal* and *ordinal*. A nominal scale is an order-less scale, which uses different symbols, characters or numbers, to represent the different states of the variable being measured. An example of a nominal variable is the customer type identifier, which might represent three types of utility customers – residential, commercial, and industrial, using digits 1, 2, and 3, respectively. Another example of a nominal attribute is the zip-code field of a customer's record. In each of these two examples, numbers designating different attribute values have no particular order and no necessary relation to one another. An ordinal scale consists of ordered discrete gradations, e.g. rankings. An example of an

ordinal attribute is the preference ordering by customers; say of their favorite pizza. An ordered scale need not be necessarily linear, e.g. the difference in rank orders 3 and 4 need not be identical to the difference in rank orders 6 and 7. All that can be established from an ordered scale is the greater-than or less-than relations. The continuous variables are also known as *quantitative* or *metric* variables. These variables are measured using either an *interval* scale or a *ratio* scale. Both of these scales allow the underlying variable to be defined or measured with infinite precision. The difference between the interval and ratio scales lies in how the zero point is defined in the scale. The zero point in the interval scale is placed arbitrarily and thus it does not indicate the complete absence of whatever is being measured. The best example of an interval scale is the temperature scale, where zero degrees Fahrenheit does not mean total absence of temperature. Because of the arbitrary placement of the zero point, the ratio relation does not hold true for variables measured using interval scales. For example, 80 degrees Fahrenheit does not imply twice as much heat as 40 degrees Fahrenheit. In contrast, a ratio scale has an absolute zero point and consequently the ratio relation holds true for variables measured using this scale. This is the scale that we use to measure such quantities as height, length, energy consumption, and salary.

## Statistical Data Analysis

Statistical data analysis is the most well established methodology for data mining. Ranging from 1-dimensional analysis, e.g. mean, median, and mode of a qualitative variable, to multivariate data analysis simultaneously using many variables in analysis, statistics offers a variety of data analysis methods [Hair et al, 1987]. These data analysis methods can be grouped into two categories. The methods in the first category are known as *dependence* methods. These methods use one or more independent variables to predict one or more dependent variables. Examples of this category of methods include multiple regression and discriminant analysis. The second category of statistical data analysis methods is known as *interdependence* methods. These methods are used when all of the variables involved are independent variables. Examples of interdependence methods are different types of clustering methods and multidimensional scaling.

#### Dependence Methods

### *Multiple Linear Regression*

Multiple regression method is used when a single dependent quantitative variable (also called the *outcome variable*) is considered related to one or more quantitative independent variables, also known as *predictors*. The objective of regression analysis is to determine the best model that can relate the dependent variable to various independent variables. Linear regression implies the use of a general linear statistical model of the following form

$$y = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_k x_k + \varepsilon$$

where $y$ is the dependent variable and $x_1, x_2 \ldots, x_k$ are the independent variables. The quantities, $a_0, a_1 \ldots, a_k$, are called unknown parameters and $\varepsilon$ represents the random error. The unknown parameters are determined by minimizing the sum of squared error (SSE). It is defined as

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

where $y_i$ and $\hat{y}_i$, respectively, are the observed and predicted values of the dependent variable for the i-th record in the database of $n$ records. The process of model fitting, when only one independent variable is involved is equivalent to best straight-line fitting in least square sense. The term *simple regression* is often used in that situation. It should be noted that the term *linear* in the general linear model applies to the dependent variable being a linear function of the unknown parameters. Thus, a general linear model might also include some higher order terms of independent variables, e.g. terms such as $x_1^2, x_1 x_2,$ or $x_2^3$. The major effort on the part of a user in using the multiple regression technique lies in identifying the relevant independent variables and in selecting the regression model terms. Two approaches are common for this task: (1) sequential search approach, and (2) combinatorial approach. The sequential search approach consists primarily of building a regression model with a set of variables, and then selectively adding or deleting variables until some overall criterion is satisfied. The combinatorial approach is a brute force approach, which searches across all possible combinations of independent variables to determine the overall best regression model. Irrespective of whether the sequential or combinatorial approach is used, the most benefit to model building occurs from a proper understanding of the application domain.

*Logistic Regression*

In many applications, the dependent variable is a qualitative variable, e.g. the credit rating of a customer, which can be good or bad. In such cases, either logistic regression or discriminant analysis is used for prediction. Rather than predicting the state of the dependent variable, the logistic regression method tries to estimate the probability $p$ that the dependent variable will be in a given state. Thus, in place of predicting whether a customer has a good or bad credit rating, the logistic regression approach tries to estimate the probability of a good credit rating. The actual state of the dependent variable is determined by looking at the estimated probability. If the estimated probability is greater than 0.50, then the prediction is yes (good credit rating), otherwise no (bad credit rating). In logistic regression, the probability $p$ is called the *success probability* and is estimated using the following model:

$$\log(p/(1-p)) = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_k x_k$$

where $a_0, a_1 \ldots, a_k$ are unknown parameters. This model is known as the *linear logistic model*

and $\log(p/(1-p))$ is called the *logistic* or *logit* transformation of a probability. Unlike the multiple linear regression where the unknown parameters are estimated using the least squares method, the logistic regression procedure determines unknown parameters by the likelihood maximization method. With respect to the credit rating example, this means maximizing the likelihood of a good credit rating.

*Linear Discriminant Analysis*

Linear discriminant analysis (LDA) is concerned with problems that are characterized as classification problems. In such problems, the dependent variable is categorical (nominal or ordinal) and the independent variables are metric. The objective of discriminant analysis is to construct a discriminant function that yields different scores when computed with data from different classes. A linear discriminant function has the following form:

$$z = w_1 x_1 + w_2 x_2 + \cdots w_k x_k$$

where $x_1, x_2 \ldots, x_k$ are the independent variables. The quantity $z$ is called the *discriminant score*. The variables $w_1, w_2 \ldots, w_k$ are called *weights*. A geometric interpretation of the discriminant score is shown in Figure 3. As this figure shows, the discriminant score for a data record represents its projection onto a line defined by the set of weight values.

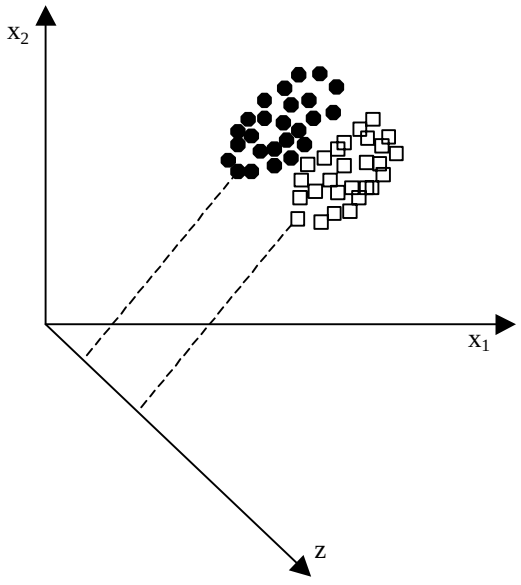


Figure 3. Geometric interpretation of the discriminant score

The construction of a discriminant function involves finding a set of weight values that maximizes the ratio of the between-groups to the within-group variance of the discriminant score for pre-classified records (training examples) from the database. Once constructed, the discriminant function is used to predict the class of a given data record, i.e. the state of the dependent variable from the independent variables. The classification is performed by the following classification rule. Assign the *i-th* data record to class A (e.g. good credit rating) if its discriminant score $z_i$ is greater than or equal to the cutting score; otherwise assign it to class B (i.e. bad credit rating). The cutting score thus serves as a criterion against which each individual record's discriminant score is judged. The choice of cutting score depends upon whether both classes of records are present in equal proportions or not, as well as the underlying distributions. It is common to assume the underlying distributions to be normal. Letting $\tilde{z}_A$ and $\tilde{z}_B$ be the mean discriminant scores of pre-classified data records from class A and B, respectively, the optimal choice for the cutting score, $z_{cut}$, is given as

$$z_{cut} = \frac{\tilde{z}_A + \tilde{z}_B}{2}$$

when the two groups of records are of equal size and are normally distributed with uniform variance in all directions. A weighted average of mean discriminant scores, calculated as follows, is used as an optimal cutting score when the groups are not of equal size:

$$z_{cut} = \frac{n_A \tilde{z}_A + n_B \tilde{z}_B}{n_A + n_B}$$

The quantities $n_A$ and $n_B$ in above, respectively, represent the number of records in each group.

While a single discriminant function is constructed for two-way classification, multiple discriminant functions are required when dealing with more than two classes. The term *multiple discriminant analysis* is used in such situations. For an *M*-way classification problem, i.e. a dependent variable with *M* possible outcomes, *M* discriminant functions are constructed. The classification rule in such situations takes the following form: "Decide in favor of the class whose discriminant score is highest." This is illustrated in Figure 4.

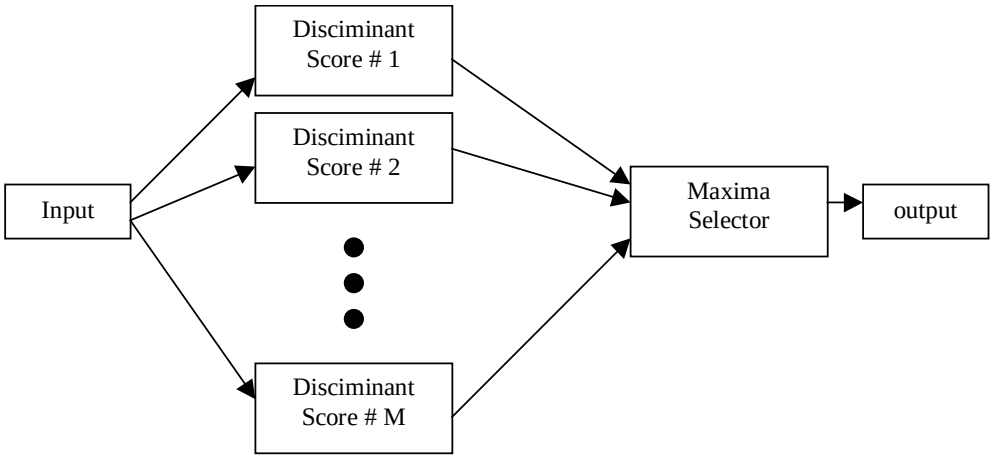


Figure 4. Illustration of classification in multiple discriminant analysis

**Interdependence Methods**

There are many applications where there is no dependent variable. In such applications, interdependence methods such as clustering and multidimensional scaling are used.

*Clustering*

Clustering or cluster analysis refers to methods for grouping objects – individuals, products, and services, in such a way that each object is more similar to objects in its own group than to objects in other groups. Since clustering methods are used in a wide range of disciplines, there exist a variety of names for clustering such as *unsupervised classification*, *Q analysis*, *typology*, and *numerical taxonomy*. A clustering method is characterized by how it measures similarity and what kind of method is employed to perform grouping. There are several ways of measuring similarity, with Euclidean distance being the most commonly used measure. Given two objects, A and B, the *Euclidean distance* between them is defined as the length of the line joining them. Other distance measures are the *absolute* and *maximum* distance functions. Irrespective of the distance measure being used, a small distance value between two objects implies high similarity and vice-versa a large distance value implies low similarity. Since objects in any application will have many attributes, each measured with a different scale, it is very common to use a normalized distance function in clustering. A normalized distance function incorporates a raw data normalization step so that each raw value is converted into a standard variate with a zero mean and a unit variance. The most commonly used normalized distance measure is the *Mahalanobis distance*, which not only

takes care of different scales for different attributes but also accounts for inter-correlations among the attributes.

Most of the common clustering methods can be classified into two general categories: (1) *hierarchical* and (2) *partitional*. A hierarchical clustering procedure achieves its clustering through a nested sequence of partitions, which can be represented in a treelike structure. On the other hand, a partitional clustering method performs clustering in one shot. Figure 5 shows these differences in two general types of clustering procedures.

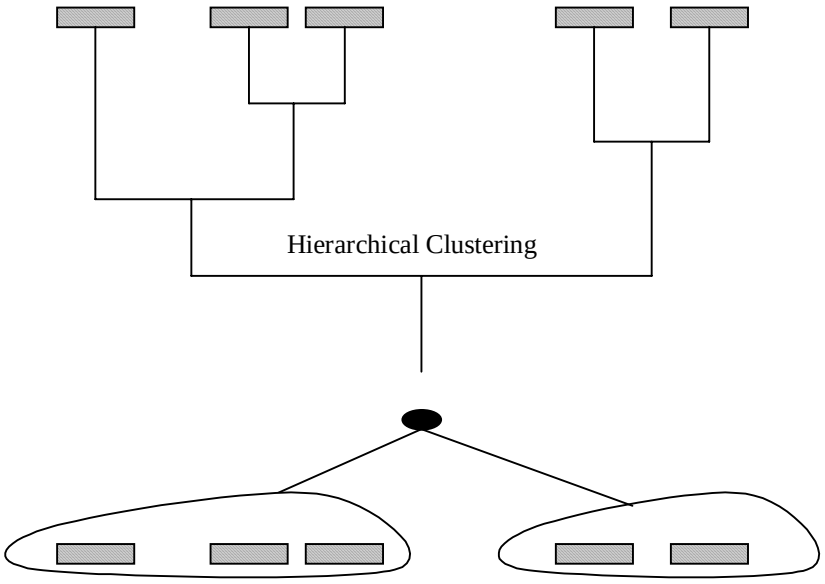


Figure 5. Differences in hierarchical and partitional clustering

## *Hierarchical Clustering*

Hierarchical clustering procedures can be further divided into two basic types – *agglomerative* and *divisive*. In agglomerative clustering, each object starts out as its own cluster. The subsequent stages involve pair-wise merging of two most similar clusters or objects. This process continues until the number of clusters is reduced to the desired number, or eventually all objects are grouped into a single cluster as shown in Figure 5. The treelike structure of Figure 5 is often referred to as a *dendrogram*. Divisive approach to hierarchical clustering is exactly opposite to the agglomerative approach. Here, we begin with one large cluster of all objects. In subsequent steps, objects most dissimilar are split off to yield smaller clusters. The process is continued until each object becomes a cluster by itself. Agglomerative procedures are much more popular and are provided by most statistical software packages. Some of the popular agglomerative clustering procedures are *single*

*linkage*, *complete linkage*, and *average linkage*. These methods differ in how the similarity is computed between clusters. Figure 6 illustrates these differences.

*Partitional Clustering*

Partitional clustering procedures can be classified as *sequential* or *simultaneous* procedures. In a sequential partitional clustering procedure, objects to be clustered are handled one by one and the ordering of presentation of objects usually has an influence on the final clusters. Simultaneous methods, in contrast, look at all objects at the same time and thus generally produce better results. Many partitional clustering methods achieve clustering via optimization. Such procedures are known as *indirect methods* in contrast with direct partitional clustering methods, which do not use any optimization method. The most well known partitional clustering method is the *K-means* method, which iteratively refines the clustering solution once the user specifies an initial partition or from a random initial partition.

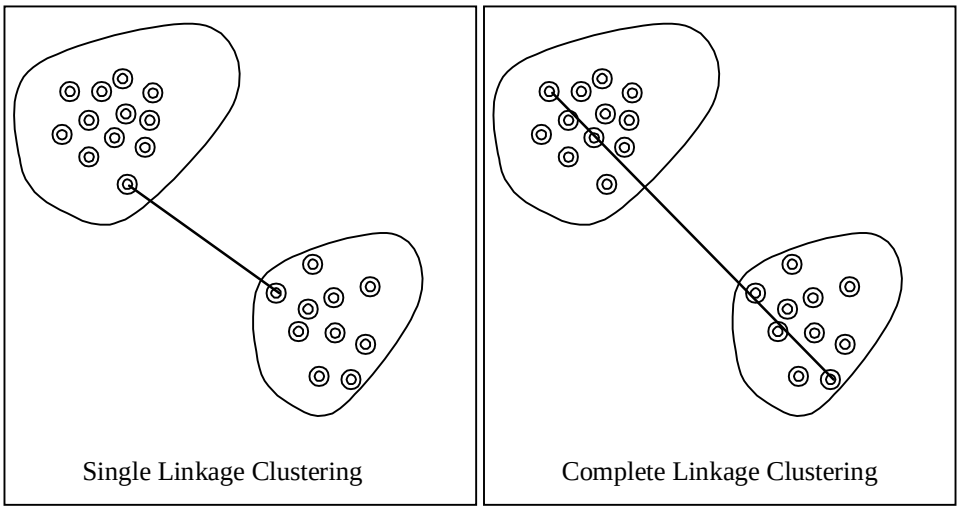


Figure 6. Similarity computation in single linkage and complete linkage clustering

## *Comparison of Hierarchical and Partitional Clustering*

Historically, the hierarchical clustering techniques have been more popular in biological, social, and behavioral sciences whereas partitional methods are more frequent in engineering applications. The main advantage of hierarchical procedures is their speed. The other advantage is that no knowledge of the number of clusters is required. The disadvantage of hierarchical methods

is that they sometime lead to artificial groupings because grouping mistakes cannot be reversed due to the hierarchical nature of the grouping process. In contrast, the partitional methods are relatively slow but tend to produce better results. However, these methods rely on the user to provide good initial seed points for clusters and thus demand a better domain understanding on the part of the user. Irrespective of the selected clustering procedure, it is generally advisable to compute several solutions before settling on one as the final solution.

*Multidimensional Scaling*

Multidimensional scaling (MDS) relies on a projection from a high dimensional space to a low dimensional space (two or three dimensions) to uncover similarities among objects. For the mapping from a high-dimensional space to a low-dimensional one to be useful in MDS, it is required that the mapping preserve inter-point distances as far as possible. MDS is also known as *perceptual mapping* and the resulting projection as the *perceived relative image*. The main application of MDS lies in evaluating customer preferences for products and services. There are two classes of MDS techniques – *decompositional* and *compositional*. The decompositional approach, also known as the *attribute-free* approach, is used in situations where only overall similarity data for different objects is available. In contrast, the compositional methods are used when detailed data across numerous attributes for each object is available. Most statistical software packages provide both kinds of MDS methods.

## Pattern Recognition

Pattern recognition theory and practice is concerned with the design, analysis, and development of methods for classification or description of patterns – objects, signals, and processes [Duda & Hart, 1973]. The classification is performed using such physical properties of patterns as height, width, thickness, and color. These properties are called *features* or *attributes* and the process of obtaining feature measurements for patterns is called *feature extraction*. Pattern recognition systems are used in two kinds of applications. The first kind of applications are those where a pattern recognition system provides cost and speed benefits. Examples of such applications are part location and identification in manufacturing, handwriting recognition in banking and offices, and speech recognition. The second kinds of applications are those where a pattern recognition system is used to perform a complex identification task either to assist or replace an expert. Examples of this kind of application include fingerprint classification, sonar signal classification, and flaw and crack detection in structures.

#### Pattern Recognition Approaches

There are three basic approaches to pattern recognition (PR) – *statistical*, *structural*, and *neural*. In the context of data mining, statistical and neural approaches are useful and will be discussed. The neural approach is discussed under neural networks. We shall limit ourselves here to

the statistical pattern recognition (SPR) approach. The statistical approach is rooted in statistical decision theory. This approach to pattern recognition treats each pattern (object or data record) as a point in an appropriate feature space. Similar patterns tend to lie close to each other, whereas dissimilar patterns, those from different classes, lie far apart in the feature space. The taxonomy of statistical pattern recognition techniques is shown in Figure 7. When complete information, i.e. a-priori probabilities and distribution parameters, about a pattern recognition task is available, the preferred PR approach is the *Bayes decision rule*, which provides optimal recognition performance. However, availability of complete information is rare and invariably a PR system is designed using a set of training or example patterns. This is analogous to data mining. When the example patterns are already classified, we say that we have *labeled patterns*. In such situations, *parametric* or *nonparametric* classification approaches are used. When example patterns do not have class labels, classification is achieved via clustering. The clustering methods in SPR are the same as those discussed earlier under statistical data analysis and, hence, will not be discussed any further.

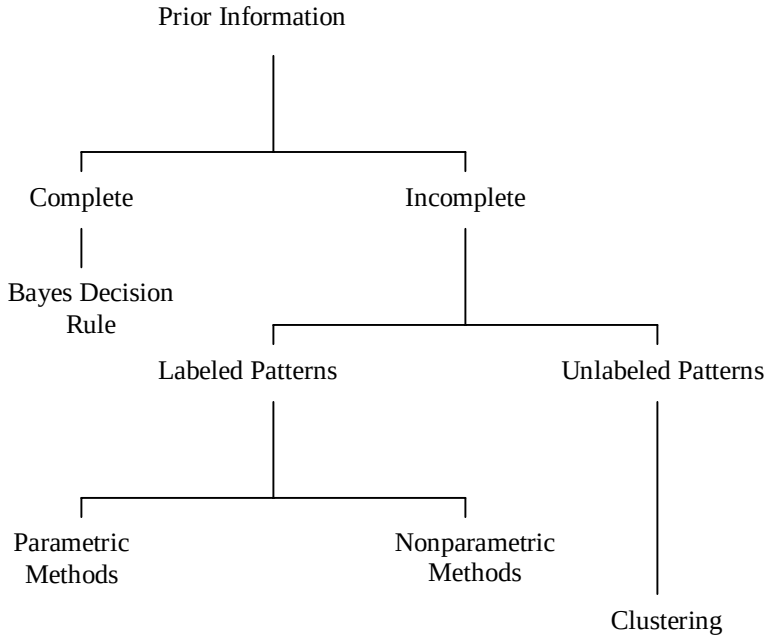


Figure 7. Taxonomy of statistical pattern recognition techniques

*Parametric Methods*

The parametric methods are used when the form of class conditional densities is known. In practice, these densities are commonly assumed to be multivariate Gaussian. The Gaussian assumption leads to linear or quadratic classifiers. To implement these classifiers, the parameters of

the class-conditional density functions are estimated using the available set of pre-classified training patterns.

*Nonparametric Methods*

The nonparametric methods are used in those situations where the form of the underlying class conditional densities is unknown. There are two basic nonparametric approaches – *density estimation approach* and *posteriori probability estimation approach*. The most well known example of the density estimation approach is the *Parzen window* technique where a moving window is used for interpolation to estimate the density function. The most well known example of the posteriori-probability estimation approach is the *k-nearest neighbor* (k-NN) method, which leads to the following rule for classification. Classify an unknown pattern to that class which is in majority among its k-nearest neighbors taken from the set of labeled training patterns. When k=1, this method is simply called the *nearest neighbor classifier*. It is easy to see that this classification rule is like a table look up procedure. The k-NN rule is a very popular classification method in data mining because it is purely a data-driven method, and does not imply any assumptions about the data. Furthermore, the k-NN rule is capable of producing complex decision boundaries. The main disadvantage of the k-NN rule is its computational burden, as an unknown pattern must be compared against every pattern in the training set, which can be exceedingly large. Many efficient implementation schemes are available in the literature to offset this drawback of the k-NN classifier. All of the classification approaches – parametric and nonparametric, discussed so far are single shot approaches, i.e. a classification decision is made in one step. Decision tree classifiers in contrast offer a multistage decision methodology where stepwise exclusion of alternatives takes place to reach a final decision. Furthermore, the decision tree methodology does not require any assumption about class conditional densities. Consequently, tree classifiers are very popular in statistics, pattern recognition, machine learning, and neural networks. We shall discuss them later under machine learning.

## Neural Networks

Artificial neural networks (ANNs) are currently enjoying tremendous popularity with successful applications in many disciplines. The interest in artificial neural networks is not new; it dates back to the work of McCulloch and Pitts, who about fifty years ago proposed an abstract model of living nerve cells or neurons [Zurada, 1992]. Since then, a very diverse set of researchers has been interested in ANNs because of a variety of different reasons. A main reason that has led many to look at ANNs is their non-algorithmic learning capability to solve complex classification, regression, and clustering problems.

#### Neuron Model

A neural network consists of a number of elementary processing units, called *neurons*. Figure

8 shows a typical neuron model. A neuron receives a number of inputs $x_1, x_2 \ldots, x_k$. Each input line has connection strength, known as *weight*. The connection strength of a line can be *excitatory* (positive weight) or *inhibitory* (negative weight). In addition, a neuron is given a constant bias input of unity through the bias weight $w_0$. Two operations are performed by a neuron –summation and output computation. The summation operation generates a weighted linear sum of inputs and the bias according to the following equation:

$$net = \sum_{i=0}^{k} w_i x_i$$

The output computation is done by mapping the weighted linear sum through an activation function:

$$y = f(net)$$

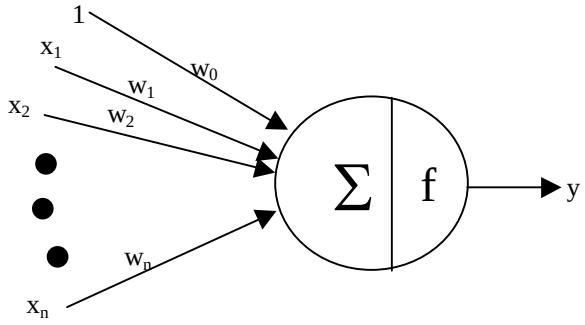


Figure 8. A typical neuron model

There are two basic types of activation functions – hard and soft. Hard activation function implies that the output of a neuron can exist in only one of the two possible states as shown below.

$$y = \text{sgn}(net - w_0) = \begin{cases} 1, & net > 0 \\ 0, & net < 0 \end{cases}$$

Such neurons are generally called *discrete* neurons or *perceptrons*. Sometimes the two allowed states are 1 and -1. Such neurons are called *bipolar discrete* neurons. Neurons with soft activation

functions are called *soft* neurons or *continuous* neurons. An example of a soft activation function is the *sigmoidal* activation function:

$$y = \frac{1}{(1 + \exp(-\alpha(net - w_0))}$$

which produces a continuously varying output in the range [0 1]. The quantity $\alpha$ in the above equations determines the slope of the activation function.

**Neural Network Models**

A neural network is a collection of interconnected neurons. Such interconnections could form a single layer or multiple layers. Furthermore, the interconnections could be unidirectional or bi-directional. The arrangement of neurons and their interconnections is called the *architecture* of the network. Different neural network models correspond to different architectures. Different neural network architectures use different learning procedures for finding the strengths (weights) of interconnections. Learning is performed using a set of training examples. When a training example specifies what output(s) should be produced for a given set of input values, the learning procedure is said to be a *supervised* learning procedure. This is the same as using a set of pre-classified examples in statistical data analysis and pattern recognition. In contrast, a network is said to be using an *unsupervised* learning procedure when a training example does not specify the output that should be produced by the network. While most neural network models rely on either a supervised or an unsupervised learning procedure, a few models use a combination of supervised and unsupervised learning.

There are a large number of neural network models, as shown in Figure 9, which have been studied in the literature [Looney, 1997]. Each model has its own strengths and weaknesses as well as a class of problems for which it is most suitable. We will briefly discuss only two models here that are common in data mining applications. These are (1) *multiple-layer feed-forward network*; and (2) *self-organizing feature map*. For information on other models, the reader should refer to books on neural networks.

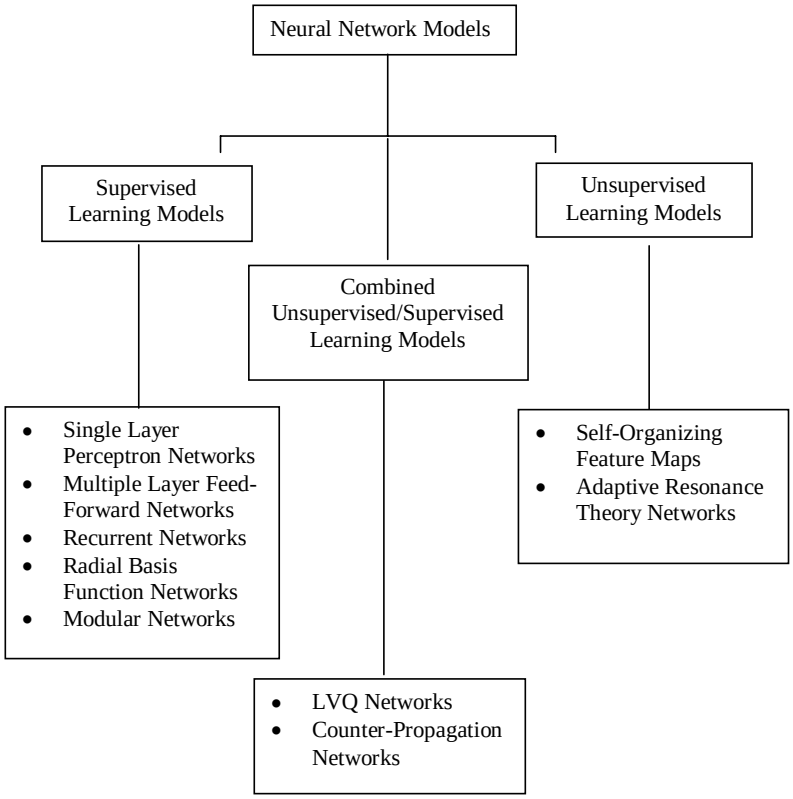


Figure 9. A classification of neural network models

*Multiple-Layer Feedforward Network Model*

The multiple-layer feedforward neural network model is perhaps the most widely used neural network model. This model consists of two or more layers of interconnected neurons, as shown in Figure 10. Generally, all neurons in a layer are connected to all neurons in the adjacent layers through unidirectional links. The leftmost layer is called the *input* layer, the rightmost the *output* layer. The rest of the layers are known as *intermediate* or *hidden* layers. It is known that a three-layer feed-forward network is capable of producing an arbitrarily complex relationship between inputs and outputs. To force a feedforward network to produce a desired input-output relationship requires training the network in an incremental manner by presenting pairs of input-output mapping. This training is done following an error minimization process, which is known as the *generalized delta rule* [Rumelhart, Hinton, and Williams, 1986]. However, the term *backpropagation* is more widely used to denote the error-minimization training procedure of multiple layer feedforward neural networks, which are often termed as *backpropagation neural networks* (BPN). One critical

aspect of using a feedforward neural network is that its structure must match well with the complexity of the input-output mapping being learned. Failure to do so may result in the trained network not having good performance on future inputs. That is, the network may not generalize well. To obtain a good match, it is common to try several network structures before settling on one. Despite certain training difficulties, the multiple layer feedforward neural networks have been employed for an extremely diverse range of practical predictive applications with great success. These networks have also been used for sequence data mining. In such applications, data within a moving window of a certain pre-determined size is used at each recorded time instant as an input to capture the temporal relationships present in the data.

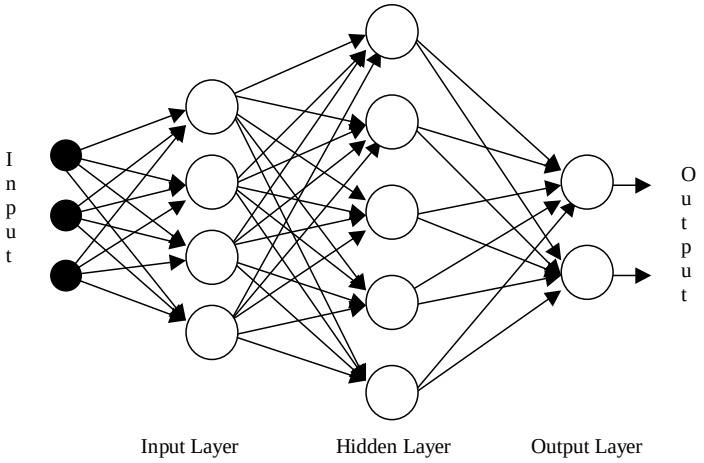


Figure 10. An example of a multiple-layer feedforward neural network

*Self-Organizing Feature Map Model*

The backpropagation training procedure is applicable only when each example in the training set has an associated output response. As mentioned before, however, many important applications do not involve any dependent variable. In order to perform data analysis in such instances, a neural network must be able to self-organize, i.e. it should be able to analyze and organize data using its intrinsic features without any external guidance. Kohonen's self-organizing feature map (SOFM) is one such neural network model that has received large attention because of its simplicity and the neuro-physiological evidence of the similar self-organization of sensory pathways in the brain [Kohonen, 1982]. The basic structure for a SOFM neural network is shown in Figure 11. It consists of a single layer of neurons with limited lateral interconnections. Each neuron receives an identical input. The network training is done following the *winner-takes-all* paradigm. Under this paradigm,

each neuron competes with others to claim the input pattern. The neuron producing the highest (or smallest) output is declared the winner. The winning neuron and its neighboring neurons then adjust their weights to respond more strongly, when the same input is presented again. This training procedure is similar to k-means clustering of statistical data analysis, and suffers from the same merits and de-merits. Next to the feedforward neural networks, the SOFM networks are the most widely used neural networks. In addition to performing clustering, these networks have been used for feature extraction from raw data such as images and audio signals. A variation of SOFM is the *learning vector quantization* (LVQ) model that seeks to combine supervised and unsupervised modes of learning. In training LVQ, rough classification rules are first learned without making use of the known classification information for training examples. These rough rules are refined next using the known classification information to obtain finely tuned classification rules.

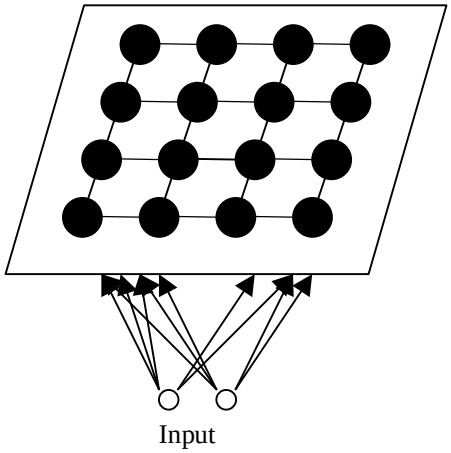


Figure 11. Kohonen's SOFM model

In addition to the above three models, other important models gaining widespread popularity include the *radial-basis function* (RBF) network model, and the *neural tree* model. The RBF network model consists of three layers and exhibits performance similar to multiple layer feedforward neural networks. However, the training time for RBF networks is much shorter. The neural tree is another class of feedforward neural networks. Such networks have limited interconnectivity, similar to a decision tree structure, but are more powerful than tree classifiers in terms of predictive capability. We will discuss neural trees under tree-based methods.

*Applying Neural Networks*

A typical neural network application requires consideration of the following issues: model selection; input-output encoding; and learning rate. The choice of the model depends upon the

nature of the predictive problem, its expected complexity, and the nature of the training examples. Since inputs and outputs in neural networks are limited to either [0-1] or [-1-1], the encoding of inputs and outputs requires careful considerations. Often, the encoding scheme for input-output has a large influence on the resulting predictive accuracy and training time. Another important factor in neural networks is the choice of learning rate. The learning rate determines the magnitude of weight changes at each training step. An improper learning rate (too small or too large) can cause an inordinately long training time, or can lead to sub-optimal predictive performance. Consequence, the use of neural networks is often called as an art.

## Machine Learning

Machine learning is a sub-discipline of artificial intelligence. The goal of machine learning is to impart computers with capabilities to autonomously learn from data or the environment. The machine learning community has been a major contributor of several new approaches to data mining. These include tree-based methods for learning, genetic algorithms, intelligent agents, and fuzzy and rough set-based approaches.

#### Tree-Based Methods

The tree-based methods for classification and regression are popular across several disciplines – statistical data analysis, pattern recognition, neural networks, and machine learning. Many similar tree-based algorithms have been developed independently in each of these disciplines. There are several reasons for the popularity of tree-based methods. First, a tree-based method allows a complex problem to be handled as a series of simpler problems. Second, the tree structure that results from successive decompositions of the problem usually provides a better understanding of the complex problem at hand. Third, the tree-based methods generally require a minimal number of assumptions about the problem at hand; and finally, there is usually some cost advantage in using a tree-based methodology in certain application domains.

*Decision Tree Classifiers*

The term *decision tree* is commonly used for the tree-based classification approach. As shown in Figure 12, a decision tree classifier uses a series of tests or decision functions to assign a classification label to a data record. The evaluation of these tests is organized in such a way that the outcome of each test reduces uncertainty about the record being classified. In addition to their capability to generate complex decision boundaries, it is the intuitive nature of decision tree classifiers, as evident from Figure 16 that is responsible for their popularity and numerous applications. Like any other data mining methodology for classification, the use of decision tree classification requires automatic extraction of a tree classifier from training data. Several automatic algorithms exist for this purpose in the pattern recognition and machine learning literature. Most of these decision tree induction algorithms follow the top-down, divide-and-conquer strategy wherein

the collection of pre-classified examples is recursively split to create example subsets of increasing homogeneity in terms of classification labels until some terminating conditions are satisfied.

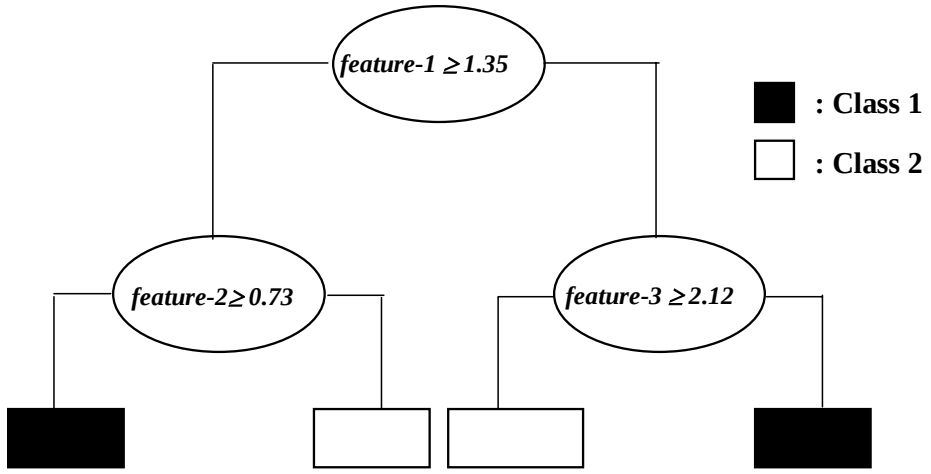


Figure 16. An example of a decision tree classifier. The left branches correspond to positive outcomes and right branches to negative outcomes of the tests at internal tree nodes.

The top-down, divide-and-conquer decision-tree-induction methodology consists of four components. First, it needs a splitting criterion to determine the effectiveness of a given split on training examples. Second, it requires a method to generate candidate splits. Third, a stopping rule is needed to decide when to stop growing the tree. Finally, it needs a method to set up a decision rule at each terminal node. The last component is the easiest part of the tree induction process. The majority rule is often used for this purpose. Different decision tree induction methods differ essentially in terms of the remaining three components. In fact, the differences are generally found only in the splitting criterion and the stopping rule.

The three most well known decision-tree-induction methodologies in pattern recognition, statistical data analysis, and machine learning literature are AMIG [Sethi and Sarvarayudu, 1982], CART [Breiman et al, 1984], and ID3 [Quinlan, 1986]. AMIG and ID3, both follow an information theory based measure, the *average-mutual information gain*, to select the desired partitioning or split of training examples. Given training examples from $c$ classes, and a partitioning $P$ that divides them into $r$ mutually exclusive partitions, the average mutual information gain measure of partitioning, $I(P)$, is given as

$$I(P) = \sum_{i=1}^{r} \sum_{j=1}^{c} p(r_i, c_j) \log_2 \frac{p(c_j / r_i)}{p(c_j)}$$

where $p(r_i, c_j)$ and $p(c_j / r_i)$, respectively, are the joint and conditional probabilities and $p(c_j)$ is the class probability. Using the maximum likelihood estimates for probabilities, the above measure can be written as

$$I(P) = \sum_{i=1}^{r} \sum_{j=1}^{c} \frac{n_{ij}}{N} \log_2 \frac{n_{ij} N}{N_i n_j}$$

where $n_j$ is the number of training examples from class $c_j$ and $n_{ij}$ is the number of examples of class $c_j$ that lie in partition $r_i$. The quantity $N$ is the total of all training examples of which $N_i$ lie in partition $r_i$. The split of training examples providing the highest value of $I(P)$ is selected. The CART procedure uses the *Gini index of diversity* to measure the impurity of a collection of examples. It is given as

$$G = 1 - \sum_{j=1}^{c} p^2(c_j)$$

The split providing maximum reduction in the impurity measure is then selected. The advantage of this criterion is its simpler arithmetic.

Determining when to stop top-down splitting of successive example subsets is the other important part of a decision-tree-induction procedure. The AMIG procedure relies for stopping on the following inequality that specifies the lower limit on the mutual information to be provided by the induced tree. The tree growing stops as soon as the accumulated mutual information due to successive splits exceeds the specified limit. CART and ID3 instead follow a more complex but a better approach of growing and pruning to determine the final induced decision tree. In this approach, the recursive splitting of training examples continues until 100% classification accuracy on them is achieved. At that point, the tree is selectively pruned upwards to find a best sub-tree according to some specified cost measure.

The generation of candidate splits at any stage of the decision-tree-induction procedure is done by searching for splits due to a single feature. For example in AMIG, CART*, and ID3, each top-down data split takes either the form of "Is $x_i \geq t$?" when the attributes are ordered variables or the form of "Is $x_i$ true?" when the attributes are binary in nature. The reason for using single feature splits is to reduce the size of the space of legal splits. For example with *n* binary features, a single feature split procedure has to evaluate only *n* different splits to determine the best split. On the other

* CART provides for limited Boolean and linear combination of features.

hand, a multi-feature-split procedure must search through a very large number of Boolean combinations, $2^{2^n}$ logical functions if searching for all possible Boolean functions, to find the best split. In recent years, many neural network-based methods, called *neural trees*, for finding decision tree splits have been developed [Sethi and Yoo, 1994; Sirat and Nadal, 1990]. These methods are able to generate efficiently multi-feature splits. Most of these neural tree methods make use of modified versions of the perceptron training procedure. Although neural trees do not provide predictive capabilities identical to those exhibited by multi-layer feedforward networks, their popularity rests on the intuitive appeal of step-wise decision making.

*Regression Trees*

While the majority of tree-based methods are concerned with the classification task, i.e. the data mining situations where the dependent variable is a discrete variable, the tree-based approach is equally suitable for regression. In such situations, the tree is called a *regression tree*. The well-known examples of the regression tree approach are CART (Classification and Regression Trees) and CHAID (Chi-Square Automatic Interaction Detection). CHAID is an extension of AID (Automatic Interaction Detection). The difference between the two is that AID is limited to applications where the dependent variable has a metric scale (interval or ratio). In contrast, CHAID is much broader in scope and can be applied even when the dependent variable happens to be a categorical variable; for example in market segmentation applications using brand preference data [Myers, 1996]. The steps involved in developing a regression tree from training examples are similar to the decision-tree-induction process discussed earlier. The major difference is in the splitting criterion. Unlike using the average-mutual information gain type measures suitable for classification tasks, measures such as the least square regression error are used in building a regression tree. There are two main differences between CART and CHAID. First, the trees produced by CART are binary trees, i.e. each node divides data into two groups only. In CHAID, the number of splits can be higher. The second difference is that CHAID is a pure top-down procedure, while CART methodology also uses bottom-up pruning to determine the proper tree size.

**Genetic Algorithms**

Genetic algorithms (GAs) belong to the broad area of evolutionary computing, which in recent years has gained widespread popularity in machine learning [Goldberg, 1989]. *Evolutionary computing* is concerned with problem solving by applying ideas of natural selection and evolution. Essentially, genetic algorithms are derivative-free search or optimization methods that use a metaphor based on evolution. In this approach, each possible solution is encoded into a binary bit string, called a chromosome. Also associated with each possible solution is a fitness function, which determines the quality of the solution. The search process in genetic algorithms begins with a

random collection of possible solutions, called the *gene pool* or *population*. At each search step, the GA constructs a new population, i.e. a new generation, using genetic operators such as crossover and mutation. Just like the natural evolution process, members of successive generations exhibit better fitness values. After several generations of genetic changes, the best solution among the surviving solutions is picked as the desired solution.

The application of the GA to a problem requires consideration of several factors, including the encoding scheme, choice of the fitness function, parent selection, and genetic operators. The selection of an encoding scheme often depends upon how well it captures the problem-specific knowledge. The selected encoding scheme also influences the design of the genetic operators. The choice of fitness function is tied with the nature of the problem being solved. For classification problems, the fitness function may be the classification accuracy on the training data. On the other hand, the fitness function may be related to the mean square error for regression problems. The parent selection component of the GA specifies how the members of the present population will be paired to create the next generation of possible solutions. The usual approach is to make the probability of mating dependent upon a member's value of the fitness function. This ensures that members with better fitness values reproduce, leading to survival of the fittest behavior. Crossover and mutation operators are the two genetic operators that determine the traits of the next generation solutions. The *crossover operator* consists of interchanging a part of the parent's genetic code. This ensures that good features of the present generation are retained for future generations. In *one-point crossover operation*, a point on the genetic code of a parent is randomly selected and two parent chromosomes are interchanged at that point. In *two-point crossover operation*, two crossover points are selected and the genetic code lying between those points is interchanged. While crossover operators with many more points can be defined, one and two-point crossover operators are typically used in GAs.

The *mutation operator* consists of randomly selecting a bit in the chromosome string and flipping its value with a probability equal to a specified mutation rate. The mutation operator is needed because crossover operation alone cannot necessarily provide a satisfactory solution, as it only involves exploiting the current genes. Without mutation, there is a danger of obtaining locally optimum solutions. It is common to use a very low mutation rate to preserve good genes. Furthermore, a high mutation rate produces behavior similar to random search, and is, therefore, not used. In addition to selection, crossover, and mutation, GAs often use additional rules to determine the members of the next generation. One popular rule in this regard is the *principle of elitism*, which requires a certain number of best members of the present population to be cloned.

While GAs are often employed on their own to solve problems, it is not uncommon to see GAs being used in conjunction with other data mining methods. For example, GAs have been used in the decision tree methodology to determine data partitions. In neural networks, these have been used for network pruning or for finding the optimal network configuration. In addition to the attractiveness of the evolution paradigm, there are several other reasons that have contributed to the overall popularity of GAs. First, GAs are inherently parallel, and thus can be implemented on parallel processing machines for massive speedups. Second, GAs are applicable to discrete as well as continuous optimization problems. Finally, GAs are less likely to get trapped in a local minimum. Despite these advantages, these algorithms have not yet been applied to very large-scale problems. One possible reason is that GAs require a significant computational effort with respect to other

methods, when parallel processing is not employed.

**Intelligent Agents**

Intelligent-agents (IA) is another machine learning approach that is rapidly becoming very popular in several applications, including data mining. Like GAs, the IA approach is also a population-based approach. However, unlike GAs where a member of the population interacts with another member to generate a new solution, the intelligent agents approach consists of finding solutions through social interaction among population members, each known as an intelligent agent [Epstein and Axtell, 1996]. Each intelligent agent behaves like an "individual" with different intelligent agents having different habits and behavior patterns. The habits and behavior patterns of agents are assigned based on existing data. As these simulated agents interact with each other, they adjust and adapt to other agents. Monitoring of behavior of a large collection of agents yields valuable information hidden in the data. For example, letting each agent simulate the buying pattern behavior of a consumer and running IA simulation for a sufficiently long time, we can count how many units of the new products are sold and at what rate. The information thus coming out of simulation can tell us whether the new product will be successful or not.

**Fuzzy and Rough Sets**

All of the methods discussed thus far have no provision for incorporating the vagueness and imprecision that is common in everyday life. The concepts of fuzzy sets and rough sets provide this provision and are useful in many applications. The fuzzy set concept was introduced by Lotfi Zadeh in 1965 [Zadeh, 1965] and since then has been applied to numerous applications. The last few years have especially seen a rapid increase in interest in fuzzy sets. Unlike conventional sets where an object either belongs to a set or not, objects in fuzzy sets are permitted varying degrees of memberships. As an example, consider the set of "tall" persons. In the conventional approach, we would define a cutoff height, say 5'10'', such that every person with height equal to or greater than the cutoff height would be considered tall. In contrast, the fuzzy set of "tall" people includes everyone in it. However, each person belongs to the "tall" fuzzy set with a different grade of membership in the interval of 0-1. The main advantage of fuzzy sets is that it allows inference rules, e.g. classification rules, to be expressed in a more natural fashion, providing a way to deal with data containing imprecise information. Most of the fuzzy set-based methods for data mining are extensions of statistical pattern recognition methods. The application of fuzzy set-based methods, e.g. the fuzzy k-means clustering procedure, to clustering is especially appealing. The statistical clustering techniques assume that an item can belong to one and only one cluster. Often in practice, no clear boundaries exist between different clusters, as shown in the several examples of Figure 13. In such situations, the notion of fuzzy clustering offers many advantages by letting each object have a membership in each cluster.

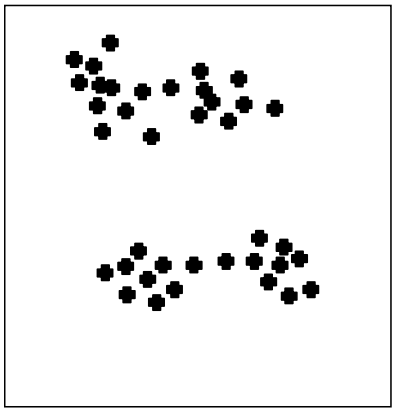

Figure 13. Examples of fuzzy clusters

The concept of rough sets is relatively new; Pawlak introduced it in 1982 [Pawlak, 1982]. The basic philosophy of rough set methodology is that lowering the degree of data precision makes data regularities easier to find and characterize in terms of rules. However, this lowering of precision is not without risk, as it can lead to loss of information and differentiation between different objects. The theory of rough sets provides tools to deal with this problem by letting roughness vary in a controlled manner to find a data precision level that ensures sufficient pattern discrimination. Often, rough sets are confused with fuzzy sets as they both deal with imperfect knowledge. However, both deal with different aspects of imperfection. The imperfection dealt with in fuzzy sets is with respect to objects within the same class. In contrast, rough sets deal with imperfections between groups of objects in different classes. The main application of rough sets so far has been in classification tasks. Furthermore, most of the reported applications have been for relatively small size problems. However, this is expected to change as rough sets gain more popularity and commercial software starts becoming available.

## ACCURACY OF THE MINED MODEL

Irrespective of the data mining methodology selected for a particular task, it is important to assess the quality of the discovered knowledge. There are two components to this assessment – *predictive accuracy* and *domain consistency*. Predictive accuracy implies how well the discovered classification or regression model will perform on future records. Domain consistency means whether the discovered knowledge is consistent with other domain knowledge that the end user might have. Since the model assessment based on domain consistency is problem specific, we will discuss only the predictive assessment component in the following.

The goal of assessing predictive accuracy is to find the true error rate - an error rate that would be obtained on an asymptotically large number of future cases. The most common approach for

assessing predictive accuracy is the train-and-test error rate estimation approach. In this approach, the entire data set is randomly divided into two groups. One group is called the training set and the other the testing set. The selected data mining methodology is applied to the test set to build the predictive model. Once the model is built, its performance is evaluated on the test set to determine the test-sample error rate, which provides a very good indication of the predictive performance on future records. When the test set consists of 5000 test cases, selected independent of the training set, then the test-sample error rate is considered virtually the same as the true error rate. For test sets of smaller size, the test-sample error rate is considered a slightly optimistic estimate of the true error rate. It is therefore a good practice to have as large a test set as possible. When this is not possible due to the small size of the total data set, a situation not likely to occur in data mining, various re-sampling techniques such as cross-validation and boot-strapping are used to obtain the predictive accuracy [Weiss and Kulikowski, 1991].

## COMPARISON OF DATA MINING METHODOLOGIES

Having described a number of methodologies, a natural question to ask is "Which data mining methodology is best?" This is a difficult question to answer. All that can be said is that there is no universally best methodology. Each specific application has its own unique characteristics that must be carefully considered and matched with different methodologies to determine the best method for that specific application. However, there are a few general remarks that can be made with respect to different methodologies.

The strength of statistical methods is that these methods come from a mature field. The methods have been thoroughly studied and have a highly developed theoretical foundation. In consequence, the knowledge discovered using these methods is more likely to be accepted by others. However, the weakness of the statistical methods is that they require many assumptions about the data. Furthermore, these methods require a good statistical knowledge on the part of an end-user to properly interpret the results. Another weakness of the statistical methods is that they do not scale well for large amounts of non-numeric data.

Many of the statistical pattern recognition methods share the same strengths and weaknesses as those of the statistical methods. However, the case of the k-NN rule is entirely different. It is a true data-driven method, which does not require any assumption about the data. Its decision making model is also intuitively understandable. The weakness of this method is its predictive capability; it cannot provide accuracy at par with other methods. This weakness of the k-NN rule, however, is offset by its zero learning time.

The neural network and machine learning methods are relatively new. In many cases, these techniques have not been theoretically well studied. However, the main strength of neural network and machine learning methods is that they require relatively fewer assumptions about data, and are thus data-driven. The neural network methods are criticized mainly on two counts. First, the results are difficult to replicate, and second, the concern about interpretability of the behavior of a trained neural network. The concern about replicating the results arises due to the algorithmic nature of neural network training. Since several components, such as the random initial weights and the presentation order of training examples, of this algorithmic process are not necessarily duplicated

every time a network is trained, it is generally difficult to replicate a network performance. The interpretability concern about neural network is associated with the *black box* image of neural networks. Since the explanations about a neural network behavior are stored in its weights, they are not easily comprehensible to a user. This has led to the black box view of neural networks. This view is further compounded by the fact that obtaining a proper neural network model for an application involves iteratively identifying proper network size, learning rate, and stopping criteria. This iterative process often overwhelms a new user. In consequence, the acceptance of neural networks is occasionally hard to come by. However, the black box image of neural networks is slowly changing as researchers are finding ways to uncover the knowledge captured by a neural network [Lu et al, 1995; Sethi and Yoo, 1996].

In comparison to other methods, the decision tree methodology appears to offer most advantages – a competitive predictive accuracy, minimal assumptions about data, better computational efficiency, and good scalability. Furthermore, the tree-based methods are able to deal with all types of variables. Consequently, this methodology is highly popular in data mining. The decision tree methodology also yields a mined model that is extremely easy to understand and use. On the negative side, the performance of decision trees is occasionally unsatisfactory because of single feature splits for data partitioning. Another criticism of decision tree methods is their greedy tree growing methodology, which often hides better models from discovery. It should be noted that many new decision tree induction methods combining machine and neural learning are being developed to address the above drawbacks of the decision tree methodology. Their availability in commercial software packages is expected to lead to more usage of the tree methodology in data mining.

## SUMMARY

This chapter has provided an introduction to data mining and the core technologies behind it. Data mining is not a single technique but rather a collection of techniques that allows us to reach out to valuable hidden information present in large business databases. Although computer-based learning technologies - neural networks, pattern recognition, decision trees - form the core of data mining, there is more to data mining than simply using a neural network or decision tree algorithm. It is an interactive and an iterative process of several stages driven with the goal of generating useful business intelligence. The success of a data mining effort is not determined by the accuracy of the resultant predictive or classification model but by the value of the model to the business. Effective data mining not only requires a clear understanding of the business issues involved but also needs an inordinate amount of data preparation - identifying important variables, cleaning data, coding and analyzing data. Without proper data preparation, data mining is apt to generate useless information. The proverbial garbage-in, garbage-out is never more apt than in a data mining application without a proper understanding of the business aspects of the problem and careful data preparation.

Data mining has emerged as a strategic tool in the hands of decision-makers to gain valuable business intelligence from their corporate data. Such business intelligence is helping companies

improve their operations in many critical areas including marketing, product development and customer services. In consequence, the applications of data mining continue to grow.

## REFERENCES


Breiman L., J. Friedman, R. Olshen, and C.J. Stone, Classification and Regression Trees, Belmont, CA: Wadsworth Int'l Group, 1984.

Duda R. and P. Hart, Pattern Classification and Scene Analysis, New York: John Wiley and Sons, 1973.

Epstein J.M. and R. Axtell, Growing Artificial Socities, Washington, DC: Brookerings Institution Press, 1996.

Fayyad U.M., G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, Cambridge, MA: AAAI Press/MIT Press, 1996.

Goldberg D.E., Genetic Algorithms in Search, Optimization, and Machine Learning, Reading, MA: Addison-Wesley, 1989.

Hair Jr. J.F., R.E. Anderson, R.L. Tatham and W.C. Black, Multivariate Data Analysis, New York: Macmillan Publishing Company, 1987.

Kohonen T., "Self-Organized Formation of Topologically Correct Feature Maps," Biological Cybernetics, 43: 59-69, 1982.

Lu, H., R. Setiono and H. Liu, "NeuroRule: A Connectionist Approach to Data Mining," in Proceedings of the 21st VLDB Conference, pp. 478-489, 1995.

Looney C.G., Pattern Recognition Using Neural Networks, New York: Oxford University Press, 1997.

Myers J.H., Segmentation and Positioning for Strategic Marketing Decisions, Chicago: American Marketing Association, 1996.

Pawlak Z., Rough Sets: Theoretical Aspects of Reasoning About Data, Dordrecht, The Netherlands: Kluwer Academic Publishers, 1991.

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

Rumelhart, D.E., G.E. Hinton, and R.J. William, "Learning Internal Representation by Error Propagation," in Parallel Distributed Processing, MIT Press: Cambridge, MA, 1986.

Sethi I.K. and J.H. Yoo, "Design of Multicategory Multifeature Split Decision Trees Using Perceptron Learning," Pattern Recognition, 27(7): 939-947, 1994.

Sethi I.K. and J.H. Yoo, "Symbolic Mapping of Neurons in Feedforward Networks," Pattern Recognition Letters, 17(10): 1035-1046, 1996.

Sirat J.A. and J.-P. Nadal, "Neural Trees: A New Tool for Classification," Networks, 1: 423-438, 1990.

Weiss S.M. and C.A. Kulikowski, Computer Systems That Learn, San Mateo, CA: Morgan Kaufmann Publishers, 1991.

Zadeh L.A., "Fuzzy Sets," Information and Control, 8: 338-353, 1965.

Zurada J.M., Artificial Neural Systems, St. Paul, MN: West Publishing, 1992.