

Task Report for the

**Energy Efficient and Affordable Small
Commercial and Residential Buildings
Research Program**

*a Public Interest Energy Research Program
sponsored by the California Energy Commission*

**Project 2.5 – Pattern-Recognition Based Fault
Detection and Diagnostics**

Task 2.5.2 - Select Pattern-Recognition Techniques

R. S. Briggs, Battelle Northwest Division

April 2001

Prepared for
Architectural Energy Corporation

Battelle Northwest Division
Richland, Washington 99352

THIS REPORT WAS PREPARED AS A RESULT OF WORK SPONSORED BY THE CALIFORNIA ENERGY COMMISSION (COMMISSION). IT DOES NOT NECESSARILY REPRESENT THE VIEWS OF THE COMMISSION, ITS EMPLOYEES, OR THE STATE OF CALIFORNIA. THE COMMISSION, THE STATE OF CALIFORNIA, ITS EMPLOYEES, CONTRACTORS, AND SUBCONTRACTORS MAKE NO WARRANTY, EXPRESS OR IMPLIED, AND ASSUME NO LEGAL LIABILITY FOR THE INFORMATION IN THIS REPORT; NOR DOES ANY PARTY REPRESENT THAT THE USE OF THIS INFORMATION WILL NOT INFRINGE UPON PRIVATELY OWNED RIGHTS. THIS REPORT HAS NOT BEEN APPROVED OR DISAPPROVED BY THE COMMISSION NOR HAS THE COMMISSION PASSED UPON THE ACCURACY OR ADEQUACY OF THE INFORMATION IN THIS REPORT.

Contents

1	Executive Summary.....	1
2	Purpose of This Task Report.....	4
3	Pattern-Recognition Techniques.....	5
3.1	Pattern Recognition Defined.....	5
3.2	Overview of Pattern-Recognition Techniques.....	6
3.2.1	General Issues.....	6
3.2.2	Fixed models.....	7
3.2.3	Parametric Methods.....	7
3.2.4	Nonparametric (data-driven) methods.....	8
3.2.5	Classification of Pattern-Recognition Algorithms.....	9
3.2.6	Data Preprocessing.....	11
3.3	Selecting Pattern-Recognition Techniques.....	12
4	Pattern Recognition for Building Fault Detection and Diagnostics.....	14
4.1	Description of Opportunities in Buildings.....	14
4.1.1	General.....	14
4.1.2	Project Focus.....	15
4.2	Evaluation Relative to Technique Selection Criteria.....	15
4.2.1	Classification vs. Estimation.....	15
4.2.2	State of Knowledge of the Underlying Phenomena.....	16
4.2.3	Availability of Training Data.....	16
5	Conclusions and Further Work on Algorithm Selection.....	19
6	References.....	20
7	Appendix – Glossary of Terms.....	21

1 Executive Summary

This is the second task report for Project 2.6 – Pattern-Recognition-Based Fault Detection and Diagnostics. The objective of this project is to develop automated diagnostics for building systems using pattern recognition techniques as the basis for automation. The diagnostics themselves will be derived from diagnostic techniques developed by Architectural Energy Corporation (AEC) for use with their ENFORMA software. In the first task report, we reported on the selection of boilers and chillers as the primary components on which to focus this project. Cooling towers were added afterwards because of their importance to the performance of many chiller-based systems. This task report provides an overview of pattern-recognition techniques that may be applicable in detecting and diagnosing operational problems and discusses their potential for application to building systems. The specific techniques that will be used in the prototype tools developed in this project will be selected in Task 2.5.3 after promising candidate techniques are tested.

Pattern recognition can be defined as "the association of an observation to past experience or knowledge" (Kennedy 1998). Pattern recognition processes can be divided into two major types—classification and estimation—both of which are applicable to the fault detection and diagnostic problems in buildings. Classification is a process of applying one of a finite set of labels to an observation; e.g., classifying the cooling tower approach temperature as *normal* or *abnormal*. Estimation is the process of applying one of a potentially infinite set of numerical labels to an observation; e.g., the cooling tower approach temperature is 8.3°F. Classification techniques can provide information in classes that map directly into conclusions or actions. Estimation is relevant when numerical precision or high resolution is required. Both classes of methods should prove valuable in automating fault detection and diagnosis.

The review of pattern-recognition techniques is organized around three major types of approaches or models used for pattern recognition—*fixed models*, *parametric methods*, and *nonparametric methods*—plus a set of generic techniques referred to as *data preprocessing*, which can be used with any of these three types of approaches. The distinctions between the three major approaches can be blurry, and pattern-recognition applications can use these techniques in various combinations and hybrid forms.

Fixed models are used where the underlying process is well understood and equations are available that adequately represent the behavior of the system. The term *fixed models* is used because all of the parameters in the model are fixed, having been defined at the outset of execution of the pattern-recognition process.

Parametric methods offer a second approach that can be used where some conceptual understanding of system behavior exists but that understanding is insufficient to create a complete or acceptably accurate model. The term *parametric* is used because the parameters (or variables) necessary to characterize system behavior are specified based on prior knowledge. Models of this type are classified as *parametric* rather than *fixed* because some aspects of the models (e.g., values of coefficients) are determined empirically rather than based on *a priori* knowledge.

In contrast with fixed and parametric methods, *nonparametric models* are derived almost exclusively from empirical data representing past behavior of the system to be modeled, hence they can be characterized as *data-driven* methods. Nonparametric methods are suitable where the structure of the problem and the relationships among parameters cannot be defined based on prior knowledge; where they can, parametric methods are usually simpler and more effective. Nonparametric methods can be effective with very large and complex pattern-recognition problems, even when the underlying processes are not adequately understood to create fixed or parametric models. Therein lies their major advantage over fixed and parametric models, which are limited in application to problems that lie well within the

limits of human comprehension. Nonparametric methods require *training data*. As a consequence, where little or no empirical data are available—as when commissioning a new built-up HVAC system—data-driven methods may not be appropriate.

Data *preprocessing* is used in some form with most pattern-recognition approaches to make the systems more efficient to develop and more effective when deployed in the field. Data preprocessing is performed using a collection of crosscutting techniques that may be applicable with fixed-model, parametric, and nonparametric approaches. Data preprocessing may be applicable to both training data used during development of the pattern-recognition application and to the data streams that a pattern-recognition application monitors during operation. The purpose of data preprocessing is to simplify the data inputs to the system and to increase the strength and clarity of the signal relative to the noise in the data. Data preprocessing can involve the use of a variety of techniques to manipulate the input data. The usefulness of these various techniques is highly problem dependent. Common techniques include averaging values, applying thresholds or filters, normalizing data, and combining correlated variables. Data preprocessing is an important step that can often mean the difference between a system that performs well and a system that performs poorly.

Selection of pattern-recognition techniques is largely an empirical process—some view it as an art. The process is highly iterative and involves applying selected techniques, testing their performance, and selecting the one that best meets the need to explain variations in the data but subject to other constraints. Selecting a model usual involves a trade-off between performance characteristics such as accuracy, memory requirements, training time, and execution speed. Often a simple model (such as linear regression) is applied to the problem and then used as a baseline with which to compare the performance of other models.

Under a broad definition of the term *pattern recognition*, there is a range of techniques that appear to be capable of supporting useful fault detection and diagnostic capabilities for building systems. In many respects, the problems of pattern recognition for building systems fault detection and diagnostics appear less challenging than many other pattern-recognition problems with which approaches reviewed in this report have shown good success.

Favorable attributes of this problem are the industry's mature understanding of the underlying science and well-developed engineering methods. The available cause-and-effect models of system behavior can enable the pattern-recognition problem to be simplified and solutions made more robust. Other favorable attributes of our problem are the small number of variables required and lenient response-time requirements. The major unfavorable attribute of the problem is the potential lack of adequate training data under some deployment scenarios—in particular with new buildings and newly installed diagnostic systems, especially for diagnostic approaches that rely on models of faults themselves.

The wealth of first principles-based models in the building-systems domain suggests that fixed or parametric models will provide the most promising approaches. However, past efforts to correlate these first principles-based models with measured building performance suggest that some tuning of these models will be required for most systems. The need for free parameters to accomplish this tuning would argue for use of parametric methods rather than fixed models. Although, nothing in this review argues strongly for focusing on nonparametric approaches, such as neural networks, it would be premature to exclude them from further consideration. There are clearly some usage scenarios and important sub-problems in which the training data would likely be adequate for use of such methods.

Final selection of algorithms will necessarily be based on specific attributes of the detection and diagnostic problems. Proceeding under other project tasks are activities that will lead to final selection

of the detailed diagnostics. Final work on selecting pattern-recognition techniques awaits completion of those tasks, particularly algorithm testing in Task 2.5.3. Documentation of the final pattern-recognition technique selection decisions will be included in the task report on laboratory testing of algorithms, Task 2.5.3 Implement and Test Techniques.

2 Purpose of This Task Report

This is the second task report for a project whose object is to develop pattern-recognition techniques to detect and diagnose faults in the operation of building systems. The first task report documented the selection of the building systems to be addressed by these techniques. Boilers and chillers were selected as the primary components to be used in technique development. Cooling towers were added because of their importance to chiller performance in many systems. This second task report reviews available approaches and pattern-recognition techniques that may be applicable in detecting and diagnosing operational problems that are common in the selected components.

This report provides only a preliminary, high-level assessment of the suitability of pattern-recognition techniques to fault detection and diagnostic problems. Because Boilers, chillers, and cooling towers are each subject to multiple types of faults, the project scope comprises multiple diagnostic "subproblems." The most effective pattern-recognition techniques may vary among these subproblems. More detailed documentation and discussion of the final selection of pattern-recognition techniques will be included in the third task report, where findings can be based on tested techniques rather than entirely on paper studies.

3 Pattern-Recognition Techniques

This section provides an overview of pattern-recognition methods and techniques that may be applicable to fault detection and diagnostics for building systems.

3.1 Pattern Recognition Defined

For purposes of this project and report, we will use a rather broad definition of the term *pattern recognition*. Pattern recognition has been defined as "the association of an observation to past experience or knowledge." (Kennedy 1998) Pattern recognition has been an active area of research in the fields of psychology and neurophysiology for several decades. Humans excel at certain types of very complex pattern-recognition tasks. For example, consider the ease with which we recognize a face or the voice of a famous singer. Then contrast that ease with the daunting task of understanding the processing that goes on in the human mind or replicating that process in a computer-based device.

Some pattern-recognition tasks have been successfully automated using computer technology, and successful applications have been developed in such fields as manufacturing, marketing, medicine, and finance. Automated pattern-recognition procedures offer a number of advantages over reliance on humans to perform the same tasks. Computer-based systems can improve speed, provide continuous monitoring where only periodic monitoring is otherwise feasible, eliminate tasks that humans find tedious, substitute for human experts where they are scarce or too costly to use, and offer superior analytical capabilities for certain types of operations. An example of this later advantage would be in identifying the optimal time to service or replace air filters (while accounting for fan power) as opposed to relying on a fixed time interval or a predefined static pressure threshold.

Interest in pattern-recognition techniques has increased with the advent of very large databases—so called *data warehouses*. *Data Mining* is a term that has been used in marketing and a number of other fields for the process of extracting useful information from large datasets using pattern-recognition techniques. Other fields use such terms as *automatic discovery* and *exploratory agents* to describe what is essentially the same process. We use the term *pattern recognition* for these processes throughout the remainder of this report.

For most readers, the phrase "automated pattern recognition" probably conjures images of an actual graphical plot of data that is then interpreted graphically. While useful pattern-recognition applications probably could be developed that function by interpreting graphic images, the focus of discussions in this report is on methods that—while analogous to matching patterns in visual representations—accomplish the same thing mathematically inside a computer.

Pattern recognition processes can be divided into two major types—classification and estimation—both of which are applicable to the fault detection and diagnostic problems in buildings. Classification is a process of applying one of a finite set of labels to an observation; e.g., the cooling tower approach temperature is *normal* (or *abnormal*). Estimation is the process of applying one of a potentially infinite set of numerical labels to an observation; e.g., the cooling tower approach temperature is 8.3°F. While some define pattern recognition to include only classification, we include both classification and estimation in our definition. The reason for this is that estimation is clearly relevant to the larger problem of developing systems that perform automated fault detection and diagnostics for buildings.

3.2 Overview of Pattern-Recognition Techniques

This review of pattern-recognition techniques is organized around three major types of approaches—fixed models, parametric methods, and nonparametric methods—plus a set of generic techniques referred to as *data preprocessing*, which can be used with any of these three types of approaches.

The distinctions between the three major approaches can be blurry, and pattern-recognition applications can use these techniques in various combinations and hybrid forms. The reader is encouraged to view the three types of approaches as providing a useful way of organizing this discussion, and not as incompatible or entirely distinct methodologies.

3.2.1 General Issues

The core activity involved in creating an automated pattern-recognition capability involves building a system that accepts inputs and returns outputs. The inputs might include information about the systems (e.g., capacities, power requirements, numbers of cells, and control sequences for a cooling tower) and monitored data streams (e.g., sump temperature and electrical current to the tower). The outputs might include status information (e.g., that the tower approach, range, and power consumption are normal).

We refer to the system that accepts inputs and returns output as a *model*. Of course, software that performs automated pattern recognition would consist of more than just a computer model of the subject system's behavior. The software would require an interface to manage inputs, an interface to convey output to the user (or other entity capable of taking the requisite action), and, particularly in the case of estimation algorithms, logic to interpret model results. For example, a model might predict that under current operating conditions, the cooling tower approach temperature should be 8.5°F. At the same time, direct measurement of the actual temperatures might show the approach to be 11°F. Program logic (in addition to the model) would be needed to compare the observed temperatures to the model output, infer that a fault is present, and originate an appropriate message or signal to initiate corrective action. Figure 3-3.1 illustrates the process and needed software components graphically.

The focus of this discussion is on the models—whether fixed, parametric, or nonparametric—and not on the related software components that may be necessary to interpret results based on model output. While these other components may be necessary parts of a functioning system, they are peripheral to the core methodological decisions, which are the focus of this review.

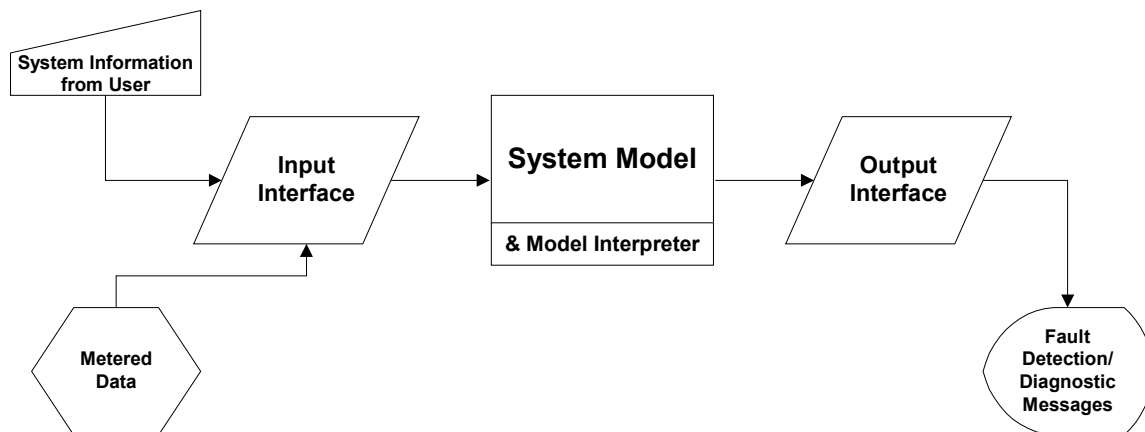


Figure 3-3.1 Conceptual Model of Diagnostic System

3.2.2 Fixed models

The first category of approaches we review is called fixed models, which represents the simplest and most straightforward pattern-recognition approach. Fixed models are used where the underlying process is well understood and equations are available that adequately represent the behavior of the system. The term *fixed models* is used because all of the parameters in the model are fixed, having been defined at the outset of execution of the pattern-recognition process. Most fixed models can be readily translated into pattern-recognition applications using common procedural programming methods.

An example of a pattern-recognition problem amenable to solution using fixed models is fault detection for air-side economizers. A model of correct economizer operation can readily be constructed based on conventional system design and outside-air damper control logic. Comparisons of model inputs of room thermostat, and outside-air and mixed-air temperatures can be used to construct a system capable of classifying economizer status as either OK, providing excessive outside air, or providing insufficient outside air under a wide variety of conditions. This approach was used in the economizer and outdoor-air diagnostic procedure in the Whole-Building Diagnostician (WBD) developed at PNNL

Expert systems are a particular type of fixed model, and are important to mention because of their potential application to building systems. Expert systems are created through interviews with human experts. Knowledge is elicited by a "knowledge engineer" and then formulated as a series of logical statements. The computer-based expert system uses the resulting "knowledge base" and software that embeds a component referred to as an "inference engine" to reason about the subject domain.

The major impediment to use of fixed models in most fields is that many pattern-recognition problems are not well-enough understood and adequately expressed in suitable form; i.e., as equations, algorithms, or available knowledge bases. Expert systems are time consuming to create, and issues related to uncertainty of inputs and confidence levels of outputs can pose significant impediments to their use, particularly where high levels of reliability are required.

Inadequate theory and equations make the use of fixed models very limited in such applications of pattern recognition as voice or character recognition—two applications that have enjoyed good success using other pattern-recognition approaches. However, fixed models offer a promising approach for many fault detection and diagnostic problems in buildings. In the buildings domain, both engineering theory and practice are well developed, and equations and algorithms capable of modeling system performance with adequate accuracy are readily available.

3.2.3 Parametric Methods

Parametric methods offer a second approach that can be used where you have some conceptual understanding of how to predict system behavior but that understanding is insufficient for you to create a complete or acceptably accurate model. The term *parametric* is used because the parameters (or variables) necessary to characterize system behavior are specified based on prior knowledge. Models of this type are classified as *parametric* rather than *fixed* because some aspects of the models (e.g., coefficients) are defined empirically rather than based on *a priori* knowledge.

An example of a parametric model for building systems can be found in ASHRAE/IESNA Standard 90. For many years, Standard 90 has used a parametric model that estimates building envelope loads for demonstrating compliance. The parameters in this simplified model were defined based on prior knowledge, but large numbers of computer simulations were used to tune the parametric model.

While there are a variety of ways to develop parametric models, linear regression is probably the most common and straightforward. Regression analysis is a statistical technique for fitting equations to (most often) empirical data. It can be used to readily develop mathematical models for estimating the behavior of systems for which there is empirical data describing past behavior. Identifying the parameters (i.e., the independent variables in the resulting regression model) is a key step in successful use of parametric methods. Knowledge of the structure of the problem and relationships among parameters is extremely helpful in developing robust parametric models that will perform well even under conditions that are not well represented in the original data. For example, knowing that the heat flow through a building envelope assembly varies directly with the assembly's area and inversely with its effective R-value is structural knowledge about the problem that is helpful in building an effective model.

An example application of parametric methods to pattern recognition might involve a system that monitors chiller efficiency and detects performance degradation. Chiller efficiency can be predicted using fixed models based on standard engineering algorithms that use inputs of rated COP and manufacturers data representing off-design performance characteristics. However, such fixed models might not yield the necessary accuracy for continuous monitoring purposes. A variety of factors may cause installed performance to differ from design performance, such as sensor placement or calibration, simplifications inherent in the engineering algorithms, pipe losses, or a large number of other conditions unique to the particular chiller or its installation. Regression analysis could be applied to measured data from the chiller, and the resulting coefficients could then be used to construct an empirical-based parametric model of chiller efficiency. In effect, this strategy would create a "tuned" model of chiller performance. This model would likely better match "normal" chiller performance than would the fixed-model alternative, and hence would likely provide a more sensitive detector of faults that cause the chiller's performance to differ from normal.

In addition to linear regression, there are two other algorithms used to develop parametric models for use in pattern recognition that warrant brief mention here—logistic regression and unimodal Gaussian. Logistic regression is a statistical algorithm that like linear regression can be used to build a model that maps inputs to outputs. Logistical regression algorithms are normally implemented using iterative methods and offer the advantage of embedding probabilities in the terms of the resulting model. However, logistic regression algorithms are not suited to problems in which input parameters are correlated, which make them ill suited to many pattern-recognition problems. Unimodal Gaussian is a relatively simple parametric algorithm, but is suitable only for classification problems. It performs well only when modeling data having Gaussian distributions. Because of its simplicity, unimodal Gaussian is a good algorithm to try first, if only to establish a useful benchmark against which to evaluate the performance of more complicated algorithms.

3.2.4 Nonparametric (data-driven) methods

In contrast with fixed and parametric methods, nonparametric models are derived almost exclusively from data representing past behavior of the system to be modeled, hence they can be characterized as *data-driven* methods. A key premise underlying data-driven methods is that the patterns in the data used to create the models (i.e., relationships between inputs and outputs) will recur in the future.

The use of nonparametric methods has grown rapidly in recent years in concert with the growth in the size of databases that organizations are creating and the increases in computational power available for analyzing these massive quantities of data. Very large datasets are being created in many fields, and nonparametric pattern-recognition methods can provide an efficient way to begin to mine the wealth of knowledge these data contain.

Nonparametric methods are suitable where the structure of the problem and the relationships among parameters cannot be defined based on prior knowledge; where they can, parametric methods are usually simpler and more effective. Nonparametric methods can be effective with very large and complex pattern-recognition problems, even when we do not fully understand and cannot model the underlying processes. Therein lies their major advantage over fixed and parametric models, which are limited in application to problems that lie well within the limits of human comprehension.

Nonparametric methods usually require a large number of example patterns of past behavior from which to build a model. These example patterns are usually referred to as *training data*. In contrast with fixed models, which represent an expression of how a system should perform based on theory (or first principles), nonparametric models do not offer any theoretical benchmark. The only basis they offer for how a system should perform is based on how it has performed. This fact has significant implications for the application of nonparametric methods to automated fault detection and diagnostics for building systems. Where little or no empirical data are available—as when commissioning a new built-up HVAC system—data-driven methods may not be appropriate. However, after successful commissioning of that HVAC system and several weeks or months of within-spec operation, continuous monitoring of the system's performance may well be a suitable application for data-driven methods. And, whether the building has been commissioned or not, with adequate training data nonparametric methods should be capable of detecting changes in system performance that occur over time.

There are many algorithms that are nonparametric in nature or that can be used nonparametrically, and there are a number of different ways these algorithms can be classified and described. The discussion in this report provides a general overview and classification of some of the more widely used algorithms available.

Parametric algorithms (such as linear regression) are included in this discussion because they can be used nonparametrically; i.e., you need not exploit knowledge of the structure and relationships among parameters to use linear regression. The decision on which algorithm to use for a given application is highly problem dependent. In most cases, there are multiple algorithms capable of producing acceptable performance, and the selection decision may depend on tradeoffs between accuracy and development time, memory, training-time, and execution-time constraints in the final software implementation, or even the personal preference of the application developer.

3.2.5 Classification of Pattern-Recognition Algorithms

Table 3-1 lists a number of widely used algorithms for addressing pattern-recognition problems and classifies them with respect to several salient characteristics. Represented in the table are a range of approaches from traditional statistical techniques, such as regression and clustering, to artificial neural¹ network techniques, such as multilayer perceptron/backpropagation. Any discussion of these algorithms and their strengths and weaknesses would readily become quite technical and is beyond the scope of this report. A glossary of terms has been included as an appendix to this report, and the reader may find the glossary helpful in understanding Table 3-1. Readers seeking more detailed information about these algorithms are referred to Kennedy et al., *Solving Data Mining Problems Through Pattern Recognition*

¹ Artificial neural networks are analytical techniques that are so named because they function in a way that is thought to be analogous to neurobiological processes. Artificial neural networks have received considerable attention in recent years in connection with artificial intelligence and their potential for use with massively parallel computer architectures. Neural networks can be viewed as sets of interconnected nodes usually arranged in multiple layers. The nodes of the network serve as the computational elements and pass data to the other nodes to which they are connected. Neural networks can be thought of as nonparametric statistical algorithms capable of solving pattern-recognition problems for which no prior knowledge of the problem is available. Neural networks are capable of solving both classification and estimation problems. (StatSoft 2001)

(Kennedy, 1998). That work contains a discussion of each of the algorithms in Table 3-1 and is the source for the information presented in the table.

The characteristics used in this high-level classification in Table 3-3.1 are explained briefly below.

Table 3-3.1 Pattern-Recognition Algorithms Grouped by Kernel Function

Algorithm Type	Classification/ Estimation	Parametric/ Nonparametric	Kernel Function	Boundaries (classification)	Mapping (estimation)
Linear regression	Estimation	Parametric	Line	Hyperplanes	Best fit line
Projection pursuit	Estimation	Nonparametric	Line	Hyperplanes	Linear inner product
K nearest neighbors	Estimation	Nonparametric	Euclidean norm	Piece-wise linear	Linear combination of weighted Euclidean distances
Nearest cluster	Classification	Nonparametric			
Learning vector quantization	Classification	Nonparametric			
E-M clustering	Estimation	Nonparametric			
K means clustering	Estimation	Nonparametric			
Multilayer perceptron (MLP)/Backpropagation	Estimation	Nonparametric	Sigmoid	Hyperplanes	Weighted sum of inputs passed through a sigmoid nonlinearity
Logistic regression	Estimation	Parametric			
Radial basis functions	Estimation	Nonparametric	Gaussian	Overlapping radial fields	Weighted sum of Gaussian outputs
Unimodal Gaussian	Classification	Parametric			
Gaussian Mixture	Classification	Nonparametric			
Parzen's window	Estimation	Nonparametric			
Binary decision tree	Classification	Nonparametric	Decision tree	Hyperplanes parallel to input axes	N/A
Linear decision tree	Estimation	Nonparametric			
Multivariate analysis regression splines (MARS)	Estimation	Nonparametric	Polynomial decision tree	Piece-wise polynomial	Piece-wise polynomial
General method of data handling (GMDH)	Estimation	Nonparametric	Polynomial	Piece-wise polynomial	Nonlinear spline
Hypersphere classifier	Classification	Nonparametric	Hyperspheres	Overlapping hyperspheres	N/A
See Appendix – Glossary of Terms for explanation of terms in this table.					

Classification vs. Estimation - One important way to classify nonparametric algorithms is whether they are capable of both classification and estimation or just classification. Any algorithm capable of estimation is also capable of classification; however, the inverse of this statement is not true. For example, nearest cluster, binary decision tree, hypersphere classifier, and learning vector quantization are strictly classification algorithms, which are not useful with estimation problems. Table 3-1 identifies the algorithms useful for classification vs. estimation (i.e., estimation and classification).

Parametric vs. Nonparametric – We list algorithms that can be used parametrically (i.e., to exploit prior knowledge about the structure and nature of relationships between parameters) as parametric and those that cannot be used parametrically as nonparametric. Parametric algorithms are used most

effectively when data is used to "tune" free parameters in a model whose basic structure has been defined. Nonparametric algorithms do not assume there is any particular structure for the model.

Kernel Functions – Another way of characterizing algorithms is by their kernel functions or underlying computing elements. Algorithms sharing the same kernel functions tend to behave similarly. While in theory nonparametric methods are capable of use with any type of pattern-recognition problem, an algorithm will be most effective with problems whose underlying structures are compatible with the algorithm's kernel function. For example, linear problems are usually best addressed with linear algorithms, whereas problems that exhibit normal frequency distributions may be most effectively addressed with Gaussian functions. The nature of the kernel functions also determines the nature of the boundaries (in the case of classification problems) or mappings (in the case of estimation problems) that the algorithm is capable of defining.

3.2.6 Data Preprocessing

Data preprocessing is used in some form with most pattern-recognition approaches to make the systems more efficient to develop and more effective when deployed in the field. Data preprocessing is performed using a collection of crosscutting techniques that may be applicable with fixed-model, parametric, and nonparametric approaches. Data preprocessing may be applicable to both training data used during development of the pattern-recognition application and to the data streams that a pattern-recognition application monitors during operation.

The purpose of data preprocessing is to simplify the data inputs to the system and to increase the strength and clarity of the signal relative to the noise in the data. Data preprocessing can involve the use of a variety of techniques to manipulate the input data. The usefulness of these various techniques is highly problem dependent. While data preprocessing can be a highly technical topic, it is an important step that can often mean the difference between a system that performs well and a system that performs poorly. The objective of this discussion will be to provide the reader only a general description of the nature of the process and an overview of the types of techniques that are most frequently employed.

Techniques that are commonly employed in data preprocessing include averaging data values, thresholding data, reducing input space through such statistical techniques as principal component analysis, combining noncorrelated variables, normalizing data, and feature extraction. Each of these techniques is described briefly below.

Averaging data values – With time-series data, transient effects can impose noise in short time-step data received from sensors or data loggers; for example, when monitoring temperatures in ducts or piping loops. Using moving averages rather than the raw data, serves to average out fluctuations that are not really meaningful, enabling more accurate and sensitive fault detection.

Thresholding data – This technique can be used to discount or ignore inputs that are above or below some threshold, in effect removing noise due to transient effects (e.g., from equipment startup) from the signal.

Reducing input space – In some situations, there may be so much input data that identifying the signature of a problem is like looking for the proverbial needle in a hay stack. In cases of data overload, statistical techniques such as principal component analysis can identify the most meaningful data, enabling the rest to be discarded. These techniques can lead to dramatic reductions in the quantity of data that needs to be processed and to a clearer signal from the data that remains.

Combining noncorrelated variables – Knowledge of the underlying system can often be exploited to translate multiple input variables into a more concise and meaningful form before pattern-recognition algorithms are applied. For example, for a rooftop direct expansion (Dx) unit, the efficiency of the unit can be calculated from input current, supply air-flow rate, and relevant air-stream temperatures. It would constitute an easier pattern-recognition problem to identify faults by monitoring the unit's efficiency versus ambient temperature than by monitoring current, flow rate, and temperatures before and after the coil vs. ambient temperature.

Normalizing data – Normalizing (or scaling) data is often necessary to keep numerically large inputs from overwhelming numerically smaller ones. The normalization problem is similar to the problem of scaling axes on a graph to effectively display trends in the data. Sometimes nonlinear transformations are necessary (like using a logarithmic scale) in order to enable pattern-recognition algorithms to perform most effectively.

Feature extraction – Feature extraction involves exploiting knowledge about the underlying problem in order to improve the performance of pattern-recognition algorithms. An example would be recognizing vertical and horizontal lines in alphanumeric characters (as opposed to pixels in a bitmap) as part of a character recognition application. In commissioning, feature extraction might involve filtering so that only data from selected time periods are used to monitor proper operation of the outside-air damper.

3.3 Selecting Pattern-Recognition Techniques

Research has shown that various algorithms can achieve similar levels of accuracy in solving many pattern-recognition problems (Kennedy 1998). However, algorithms may vary more significantly with respect to certain practical considerations, such as execution time, memory requirements, training time, complexity of the training process, and adaptability to new data. Kennedy et al. propose the following sequence of steps when selecting appropriate algorithms for use in pattern-recognition applications.

1. First, consider any hard constraints related to the final application. If the application requires an estimation result, algorithms capable only of classification results would not be appropriate. Execution time (i.e., the time needed to generate a classification or estimation result) can also represent a hard constraint. For example, an application providing quality assurance surveillance for an industrial production assembly line might have a rigid constraint on the execution speed for the deployed system, making use of certain types of algorithms infeasible. Kennedy (1998) contains a discussion of each of the algorithms addressed in Table 3-3.1 and rates their relative performance with respect to memory requirements, training times, and execution times (see Table 3-3.2).
2. Other constraints may be less rigid than the hard constraints listed above but may still represent important design tradeoffs. Requirements related to execution speed, training time, memory requirements for model training, and expected frequency of model retraining should be defined for the proposed application, for use assessing possible performance tradeoffs related to algorithm selection.
3. If you know the form of the model based on prior knowledge, use a parametric algorithm. Parametric algorithms allow you to reduce training times by reducing the number of training parameters. The resulting model is likely to provide greater accuracy, particularly if the data available for training is not representative of the entire region of interest. Parametric algorithms have major advantages where training data are scarce or costly to obtain. Prior knowledge can also influence the selection of nonparametric algorithms, as when the nature of the phenomenon being modeled is known to be better handled by one type of kernel function than another.

4. Certain algorithms offer unique advantages related to calculating probabilities that may be useful in generating confidence measurements. For example, the Gaussian mixture algorithm inherently produces estimates of probability density functions, and in some applications these can be useful in providing guidance on how much to trust the results.
5. An alternative approach to algorithm selection is to begin with the simplest algorithm—linear regression—and shift to more complex algorithms only after the easiest method is shown to produce inadequate results. This approach has the added advantage of establishing a performance benchmark against which progress can be measured as work proceeds using more complex algorithms.

Table 3-3.2 Performance Characteristics of Patter Recognition Algorithms

Algorithm Type	Memory	Training Time	Test Time
Linear regression	Very Low	Fast	Very Fast
Projection pursuit	Low	Medium	Fast
K nearest neighbors	High	Very Fast	Slow
Nearest cluster	Medium	Medium	Med-Fast
Learning vector quantization	Medium	Slow	Medium
E-M clustering	Medium	Medium	Medium
K means clustering	Med-Hi	Medium	Med-Fast
Multilayer perceptron (MLP)/Backpropagation	Low	Slow	Very Fast
Logistic regression	Very Low	Medium	Very Fast
Radial basis functions	Medium	Medium	Medium
Unimodal Gaussian	Very Low	Medium	Fast
Gaussian Mixture	Medium	Slow-Med	Medium
Parzen's window	High	Very Fast	Slow
Binary decision tree	Low	Fast	Very Fast
Linear decision tree	Low	Fast	Very Fast
Multivariate analysis regression splines (MARS)	Low	Medium	Very Fast
General method of data handling (GMDH)	Low	Med-Fast	Fast
Hypershpere classifier	Medium	Medium	Medium
After Table 10-7 in Kennedy (1998).			

4 Pattern Recognition for Building Fault Detection and Diagnostics

The selection of pattern-recognition approaches is highly problem dependent. For that reason, we begin our discussion of the application of pattern recognition to buildings with a description of the context in which fault detection and diagnostics for buildings would be deployed.

4.1 Description of Opportunities in Buildings

This discussion is presented in two parts: 1) a discussion of the context and opportunities for buildings generally and 2) a discussion of the context and opportunities for the component diagnostics that have selected for development and demonstration in this project. This two-part approach has been used because the intent of the project is to develop diagnostic approaches that can eventually be generalized to additional building systems and components—not just those used directly in developing diagnostic approaches for this project.

4.1.1 General

This section describes some salient attributes of (or perhaps assumptions about) this building-related problem that distinguish it from others for which pattern recognition-based applications have been successfully deployed.

- Most buildings are unique. Most components or systems in buildings that would warrant fault detection or diagnostic capabilities must be understood in the context of other systems with which they interact. Solutions that offer no capability to be adapted or tuned to the building in which they operate are unlikely to have broad application.
- The fault detections and diagnostics that this work primarily targets are by nature nonurgent, although they may prove to be costly over time and hence can be important. Major components such as chillers and boilers often have dedicated controls designed to deal with faults that could damage equipment or create dangerous situations. Other potentially serious faults from an occupant perspective, become readily apparent to occupants, such as the failure of a water heater to provide hot water.
- Particularly for early applications of pattern recognition-based diagnostic applications, reliability—and avoidance of false positive diagnoses) is considered very important. A lack of user acceptance is a potentially large impediment to successful deployment of building diagnostic systems and falsely identified problems could quickly discourage building operations from using automated diagnostics.
- An estimated half of the commercial-building floor area occurs in small buildings without on-site operating personnel. Servicing is provided by off-site personnel, who are summoned when operational problems are apparent. Under this service model, diagnostic messages that indicate the cause of the problem are probably less valuable than when operating personnel are on-site and possess both the knowledge and wherewithal to implement corrective action.

There are a number of different ways that fault detection and diagnostic capabilities can be deployed. The focus of this project is on development and demonstration of these methods and not on final implementations. However, it is helpful to keep in mind the variety of ways these methods could be deployed, as deployment options can carry with them a variety of constraints and opportunities. The most conventional vehicle for deployment of automated diagnostic methods is in the software of building automation systems (also known as energy management systems). On this platform, pattern recognition-based diagnostics might represent incremental change for current products. Pattern-recognition methods can be incorporated in testing procedures used in building or system commissioning and recommissioning procedures, often done using short-term metering capabilities. Related applications

might be as additions to routine servicing procedures used by equipment service personnel. Finally, diagnostic methods could be incorporated into the controllers sold with package equipment such as rooftop units or incorporated into thermostats or other control devices, that provide supervisory monitoring of a system or entire building or component monitoring via wireless link to remote sensors.

4.1.2 Project Focus

The focus for this project work is to develop and demonstrate pattern recognition-based diagnostics for boilers, chillers, and cooling towers. The selection decision was documented in the Task 2.5.1 Report as boilers and chillers. Cooling towers were added following subsequent analysis of chiller diagnostics that revealed strong advantages to addressing chillers and cooling towers jointly.

Boilers, chillers, and cooling towers are components most commonly used in central built-up systems. These systems are most often used in large commercial buildings. These buildings typically have an on-site building manager responsible for operation of the HVAC system and are more likely to have a building automation system in place than are smaller buildings. Many of the sensors necessary for automated fault detection and diagnostics may already be in place as part of these building automation systems. In fact, diagnostic tools that rely on sensors already present for control are more likely to receive early acceptance than those requiring additional sensors. The financial rewards associated with efficient operation of the buildings in terms of energy-cost savings are much greater in large buildings than in smaller ones. Central systems do not enjoy the inherent redundancy and resiliency inherent in multiple small package systems, hence there is likely more interest in preventative action in response to early fault detection to avoid untimely downtime of key HVAC components. These factors all contribute to making these systems a favorable environment for initial deployment of automated fault detection and diagnostic capabilities. Although thorough commissioning of new buildings is by no means routine, these large buildings are more likely to receive some functional testing initially to ensure that components are performing according to specification when installed.

4.2 Evaluation Relative to Technique Selection Criteria

4.2.1 Classification vs. Estimation

The distinction between classification and estimation is a distinction between selecting from a finite set of states vs. assigning a value from a continuous range. The simplest fault detection and diagnostics can be addressed as binary classification problems; e.g., is the chiller operating normally—yes or no? or is the damper stuck at its minimum outside air position—yes or no?

Estimation approaches offer the potential to convey additional information that could provide greater value to the building operator. For example, in addition to operating normally, the chiller is currently operating 3% below rated efficiency or, in addition to the outside-air damper being at minimum set position, the outside air fraction is at 18% when it should be at 100%. The estimation result provides more information from which the user (or perhaps eventually a more sophisticated computer-based diagnostician) can infer the severity of the fault, the likely consequences of continued faulty operation, and what might be the root cause of the fault.

While it may be useful to view some of the problems in buildings as classification problems, there is potential added value in approaching at least some diagnostics as estimation problems. We expect that fault detection and diagnostic systems for buildings will need to be addressed as a mix of classification and estimation problems. For some simple systems and components, classification results may be fully satisfactory. For larger and more complex systems and components, estimation results may be

advantageous, if not initially in the long run because they offer greater potential for informative diagnostic results.

4.2.2 State of Knowledge of the Underlying Phenomena

An assessment of the state of knowledge of the underlying problem is important because it determines the feasibility of pursuing pattern-recognition solutions based on fixed-models or parametric algorithms as opposed to using nonparametric algorithms, which are generally more difficult to implement. There is a high level of understanding of the physics and engineering that governs the performance of all of the systems and equipment likely to be the subject of fault detection and diagnostic systems for buildings. In addition, engineering algorithms and computer models that adequately describe the performance of these systems and equipment are readily available.

Previous work, such as PNNL's outdoor-air/economizer diagnostic module in the Whole-Building Diagnostician, has used fixed models to perform fault detection and diagnostics for air-side economizers. While the system has performed well, the effort required to build the diagnostic capability was high, and because the system incorporates detailed engineering knowledge, results are not readily generalizable to other problems.

However, buildings never operate exactly the way they are expected to based on first principles. Most building systems are sufficiently complex and dependencies on weather, occupant behavior, installation conditions, and many other factors result in significant deviation from design performance. For all but the simplest building systems, variations between actual behavior and theoretical behavior are likely to be significant. Some "tuning" of models will likely be needed in most cases. As parametric approaches offer the capacity to adjust models of known form using empirical data, they are good candidates for use in this application. Fixed models may be appropriate for simple systems and those not strongly influenced by exogenous factors; such as weather and occupant behavior.

4.2.3 Availability of Training Data

Training data are usually empirical data that reveal the relationship between model inputs and outputs based on relevant conditions. Training data are necessary to tune parametric models and are essential for the creation of nonparametric models. Fault detection and diagnostic systems may be deployed under a variety of scenarios, and the availability of training data may vary widely between these scenarios. Some possible scenarios are discussed below.

Existing building scenario – The first scenario involves an existing building with months or even years of performance data that is assumed to represent "correct" (or within specification) behavior. Under this scenario, training data would be assumed to represent correct performance. It may be possible to detect deviations from this behavior to identify when problems are occurring or that improvements have been made. The specificity of such detections limited only by the detail in the data and the models developed from it.

For approaches relying on modeling of faulty behavior itself, training data for faulty conditions is required. It is possible to identify performance such data associated with faults as they occur happenstancially. Over time, that approach could result in valuable training data for the system, but such training data would, at best, accumulate slowly and would be unlikely to be captured at all, because the association of the data representing the faulty behavior with the root cause of the fault would necessarily have to be done manually. In theory, training data could also be generated by deliberately disabling components or perturbing systems in the building. This approach is unlikely to gain wide acceptance

because it would be disruptive to normal use of the building and because of the difficulty (or impossibility) of inducing some kinds of faulty behavior.

Computer simulations offer another way to create training data. However, there are at least two major challenges to this approach. First, available simulation tools are generally not designed to model faulty performance. While faulty performance can be simulated or approximated with some simulation tools, there is little guidance to be found within the technical literature on how to do this. Secondly, simulation tools rarely match actual performance closely, particularly over short time intervals. The discrepancies between the two are of comparable magnitude with the faults that an automated detection system would be designed to detect. This suggests that some tuning of the model would likely be necessary.

New building scenarios - A second scenario involves new buildings, new fault detection and diagnostic system installations in existing buildings, or existing buildings that were never been commissioned and whose performance is suspect. Under all of these scenarios, no training data would be available that represents within-spec performance. Under these conditions, the only benchmark for normal performance would need to come from first principles-based models—a severe limitation for reasons discussed above.

Manufactured component scenario - This final scenario involves a manufactured component, such as a roof-top Dx unit. A fault detection and diagnostic capability might be embedded within the unit's controller. In contrast with previous scenarios in which we assumed each build is unique, this scenario involves manufactured products, which may be produced in large quantities. A Dx unit is a somewhat simpler piece of equipment than a chiller and is far less unique than a built-up system. Under these circumstances, it may be reasonable to derive training data from other similar units.

The most plausible way for this training data to become available would be for the manufacturer to operate a unit (or a detailed computer model of the unit) under a range of faulty conditions corresponding with the scope of the fault detection system in a test facility. HVAC test facilities are capable of varying loads on a piece of equipment and subjecting it to various condenser and evaporator temperatures. Provided the training data developed this way represents the range of conditions the unit is normally expected to operate under, this scenario could yield appropriate training data to support a variety of pattern recognition-based diagnostics.

Summary - In general, limitations on the availability of training data appear to represent a potentially significant impediment to use of data-driven approaches for fault detection and diagnostics in buildings. However, there are notable exceptions, fault detection systems embedded in manufactured components being one example. In addition the Whole-Building Energy module of the Whole-Building Diagnostician developed at PNNL uses data-driven methods to identify changes in performance that are discernable at the whole-building and major-system (e.g., electricity for HVAC or gas usage) level. This tool and data-driven modeling approach it uses have proven quite effective in identifying deviations from expected behavior, albeit still limited to highly-aggregated energy consumption. This approach, however, in principle could be extended to greater levels of resolution provided the required data are available. Other Potential Constraints

For some applications, practical considerations can constrain the selection of pattern-recognition algorithms. This section reviews the requirements for building-related applications to determine if algorithm selection will likely be constrained by any of these practical considerations.

4.2.3.1 Execution Speed

Most building operational issues are not time-critical. Operational issues typically need to be addressed in days, hours, or minutes not seconds or milliseconds. Given that few building system faults would require attention in even a few minutes, it is unlikely that considerations of execution speed will impose any constraint on algorithm selection.

4.2.3.2 Memory Requirements

Memory requirements for diagnostic systems for buildings are expected to be low. This is in part a consequence of the small number of parameters that it is practical and cost-effective to monitor today. This may change in the future if installed costs for sensors can be significantly reduce, but currently few sensors beyond those required for rudimentary control are installed in most commercial buildings. While data loggers and building automation systems typically employ time steps that are measured in seconds, time-series data is usually summed and stored using much longer time steps. Memory considerations are unlikely to constrain algorithm selection due to the combined effects of using few parameters, the lack of need to store short time-step data, and low perceived value of retaining large amounts of historic data.

4.2.3.3 Frequency of Retraining

Retraining of pattern-recognition systems may be necessary with any significant new addition of training data. Retraining may be desired after retrofits or major repairs (e.g., replacement of a boiler or chiller, but the need for frequent system retraining appears unlikely. Given the anticipated modest quantities of training data, even algorithms that are slow to train should pose no problems for buildings-related systems.

5 Conclusions and Further Work on Algorithm Selection

Under a broad definition of the term *pattern recognition*, there is a range of techniques that appear to be capable of supporting useful fault detection and diagnostic capabilities for building systems. In many respects, the problems of pattern recognition for building systems fault detection and diagnostics appear less challenging than many other pattern-recognition problems with which approaches reviewed in this report have shown good success.

Favorable attributes of this problem are the industry's mature understanding of the underlying science and well-developed engineering methods. The available cause-and-effect models of system behavior can enable the pattern-recognition problem to be simplified and solutions made more robust. Other favorable attributes of our problem are the small number of variables required and lenient response-time requirements. The major unfavorable attribute of the problem is the lack of adequate training data under some deployment scenarios—in particular with new buildings and newly installed diagnostic systems.

The wealth of first principles-based models in this domain suggests that fixed or parametric models will provide the most promising approaches. However, past efforts to correlate these first principles-based models with measured building performance suggest that some tuning of these models will be required for most systems. The need for free parameters to accomplish this tuning would argue for use of parametric methods rather than fixed models. Although, nothing in this review argues strongly for focusing on nonparametric approaches, such as neural networks, it would be premature to exclude them from further consideration. There are clearly some usage scenarios and important subproblems in which the training data would likely be adequate for use of such methods.

Final selection of algorithms will necessarily be based on specific attributes of the detection and diagnostic problems. Proceeding under other project subtasks are activities that will lead to final selection of the detailed diagnostics. Final work on selecting pattern-recognition techniques awaits completion of those tasks. Documentation of the final pattern-recognition technique selection decisions will be included in the task report on laboratory testing of algorithms, Task 2.5.3 Implement and Test Techniques.

6 References

Architectural Energy Corporation (AEC). 2000. *ENFORMA, Demonstration and Installation CD-ROM, HVAC Analyzer v3.116, Lighting Evaluation System v.2.2, MicroDataLogger-DataManager Software v3.0.2.1*. Boulder, Colorado.

Architectural Energy Corporation (AEC). *ENFORMA Portable Diagnostic Solutions HVAC Analyzer User's Guide*. Boulder, Colorado.

ASHRAE/IESNA. 1999. *ASHRAE/IESNA Standard 90.1-1999: Energy Standard for Buildings Except Low-Rise Residential Building*. American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc. Atlanta, GA.

Kennedy, R. L., Y. Lee, B. Van Roy, C. D. Reed, and R. P. Lippmann. 1998. *Solving Data Mining Problems through Pattern Recognition*. Prentice Hall, Upper Saddle River, NJ.

Lippmann, R.P. 1994. "Neural Networks, Bayesian a posteriori Probabilities and Pattern Classification," in *From Statistics to Neural Networks. Theory and Pattern Recognition Applications*, V. Cherkassky, J.H. Friedman, and H. Wechsler, eds., Springer-Verlag, New York, NY.

StatSoft, Inc. 2001. *Electronic Statistics Textbook*. Tulsa, OK: StatSoft. WEB: <http://www.statsoft.com/textbook/stathome.html>.

Winkler, R. L and W.L. Hays. 1975. *Statistics: Probability, Inference, and Decision*. Holt, Rinehart and Winston, New York, NY.

7 Appendix – Glossary of Terms

<i>architecture</i>	<i>Architecture</i> refers to the basic structure of a pattern-recognition model. Training algorithms are used to tune free parameters in a pattern-recognition model. For example, k means clustering is an algorithm that can be used with nearest cluster architecture. The distinction between algorithms and architecture is subtle and is explained here primarily to clarify the use of terminology in this report.
<i>Artificial neural networks</i>	<i>Artificial neural networks</i> are analytical techniques that are so named because they function in a way that is thought to be analogous to neurobiological processes. Neural networks can be viewed as sets of interconnected nodes usually arranged in multiple layers. The nodes of the network serve as the computational elements and pass data to the other nodes to which they are connected. Neural networks can be thought of as nonparametric statistical algorithms capable of solving pattern-recognition problem for which no prior knowledge of the problem is available. Neural networks are capable of solving both classification and estimation problems. The multi-layered perceptron/backpropagation is a particular type of neural network that is discussed in this report.
<i>binary decision tree</i>	<i>Binary decision trees</i> is a type of model used for classification pattern-recognition problems in which a simple top-down tree structure is used. Binary decision trees have two branches (or leaves) below each decision node. See also <i>decision tree</i> .
<i>decision tree</i>	<i>Decision trees</i> are a type of model that can be used for both classification and estimation pattern-recognition problems. Use of decision trees is popular because they are relatively easy to understand. A decision tree consists of a tree-like structure in which each decision is represented by a node, no branches intersect, and final decisions are represented by the leaves (lowest level) of the decision tree. Decision trees can be divided into two type binary and linear (See <i>binary decision tree</i>).
<i>E-M clustering</i>	<i>Estimate-maximize clustering</i> is a nonparametric algorithm used for classification and estimation pattern-recognition problems. It easily accommodates categorical and continuous data.
<i>Euclidean</i>	<i>Euclidean</i> refers to something of or related to Euclidean geometry. <i>Euclidean</i> geometry is the study of points, lines, planes, and other geometric figures, using a modified version of the assumptions of Euclid (c.300 BC). The most controversial assumption of Euclidean geometry is the parallel postulate, which states that there is one and only one line that contains a given point and is parallel to a given line. Modifications of Euclid's parallel postulate provide the basis for non-Euclidean geometry.

<i>Gaussian</i>	Of or pertaining to the <i>Guassian</i> (or Normal) distribution. The bell-shaped Gaussian distribution is best characterized by its probability density function $f(x) = 1/[2\pi\sigma^2]^{1/2} \exp\{-(1/2)[(x-\mu)/\sigma]^2\}$, where μ is the mean of the distribution and σ is the standard deviation of x about μ .
<i>Gaussian mixture</i>	The <i>Gaussian mixture</i> method involves fitting mixtures of Gaussian distributions to multivariate data. The mixture is the weighted sum of the individual Gaussians. This mixture can be used to approximate arbitrarily complex distributions. The parameters in the Gaussians and the weightings are determined using estimate maximization (E-M), which is an iterative process in which the objective is to maximize the likelihood that the given data points were generated by the mixture of Gaussians.
<i>group method of data handling (GMDH)</i>	<i>Group method of data handling (GMDH)</i> is a nonparametric architecture for classification and estimation pattern-recognition problems. GMDH uses heuristic methods to simplify a polynomial-based estimation model and keep it from becoming intractably complex. The process begins with low-order polynomials, which are combined to form higher-order polynomials. Unimportant terms are removed from the equations and terms are added back to enable even higher order polynomials. The training process is terminated when successive iterations do not resulting in improvements to the estimation model.
<i>hyperplane</i>	An N -dimensional construct, which divides an $N+1$ dimensional space into two, much like a line ($N = 1$) divides a 2-dimensional space in two and a plane ($N = 2$) divides a 3-dimensional space in two.
<i>hypersphere</i>	An N -dimensional analogy of a sphere. The surface of an N -dimensional sphere is $N-1$ dimensional, much like the surface of a (3-dimensional sphere is 2-dimensional.
<i>hypersphere classifier</i>	<i>Hypersphere classifiers</i> are a type of boundary-forming classifier. Boundary-forming classifiers have binary outputs that form decision regions that designate the output class. They are trained by minimizing overall classification error rates. Hypersphere classifiers are used as the basis for some neural network algorithms which are trained using supervised learning.
<i>K means clustering</i>	<i>K means clustering</i> is an algorithm used to define a given number of clusters from a training data set, which can then be used in classifying observations. K means clustering minimize the overall mean distances between observations in the clusters. The k means clustering algorithm can be used with radial basis functions and other cluster-based pattern-recognition architectures.

K nearest neighbors

K nearest neighbors (KNN) is a simple architecture for pattern-recognition based on cluster analysis. The classification or estimation result is based on a selected number (K) of the closest observations in the training set, where "closeness" is based on a calculated distance metric, which is usually Euclidean. KNN is quick to train (because the entire training data set is used), but is generally reserved for low-dimensional problems and small data sets because it is very memory intensive and slow to execute.

learning vector quantization

Learning vector quantization (LVQ) uses a nonparametric architecture identical to that of nearest cluster. It is suitable for classification pattern-recognition problems only. LVQ differs from nearest cluster in the way the model is trained. After clusters are defined using the k-means clustering algorithm, cluster centers are incrementally adjusted to minimize misclassifications using the training data, often resulting in improved model performance.

linear decision tree

Linear decision trees are a type of architecture that can be used for both classification and estimation pattern-recognition problems. The nodes in a decision tree represent decisions. Linear decision trees represent these as inequalities of linear form that may involve multiple variables. Training of linear decision trees is more complicated than for binary decision trees and is based largely on heuristic processes.

linear regression

Linear regression is a category of parametric modeling problems where the objective is to estimate the value of a continuous output variable from some input variables, where the relationships between the output variable and input variables is assumed to be linear. For one output variable, y , the linear regression model takes the form

$$y = c_0 + c_1 x_1 + c_2 x_2 + c_3 x_3 + \dots + c_n x_n,$$

where $x_1, x_2, x_3, \dots, x_n$ are the input variables, the coefficients $c_1, c_2, c_3, \dots, c_n$ and the constant c_0 are selected to optimize the model's accuracy (usually by minimizing the sum of the squared errors) and $n-1$ is the number of independent variables.

logistic regression

Logistic regression creates a parametric mapping between input variables, x_1, x_2, \dots, x_N to an output y , according to the logistics function (S-shaped)

$$y = 1/[1 + \exp(- \)],$$

where

$$= w_0 + \sum_{i=1}^N w_i x_i,$$

where the w_i 's are free parameters, the x_i 's are the independent input variables, and N is the number of input variables. The weights are determined by minimizing what is known as the cross-entropy error for the training set,

$$E = \sum_{i=1}^{N_{\text{training}}} E_k,$$

where

$$E_k = d_k \ln(1/y_k) + (1 - d_k) \ln[1/(1 - y_k)],$$

is known as the cross-entropy cost function, y_k is the output produced by the k th input vector, d_k is the desired output for the k th input vector, and N_{training} is the number of training input vectors.

multilayer perceptron (MLP)/ backpropagation

Multilayer perceptron (MLP) is a nonparametric architecture used in with classification and estimation pattern-recognition problems. MLP is used with the backpropagation algorithm and is considered a type of artificial neural network (see *artificial neural network*). MLP consists of a network of interconnected nodes usually in multiple layers. Each node outputs a weighted sum of the nodes to which it is connected, and it is the weights of each of the nodes that represent the free parameters in the MLP models.

multivariate adaptive regression splines (MARS)

Multivariate adaptive regression splines (MARS) is a nonparametric architecture for classification and estimation pattern-recognition problems. MARS involves partitioning inputs into an n -dimensional grid. Heuristics are used to merge partitions to reduce size and complexity where doing so will not impact the accuracy of the model. Low-order polynomials are fit to the training data for each partition, but the regressions splines are constrained so as to ensure smooth transitions across partition boundaries. Because the number of initial partitions grow exponentially with the number of dimensions, MARS becomes intractable for high-dimensional problems (i.e., > 10 input parameters).

nearest cluster

Nearest cluster is a pattern-recognition architecture used primarily for classification problems. Nearest cluster architecture is similar to k nearest neighbors (KNN), but the training data is partitioned into clusters—see *k nearest neighbors*. *Nearest cluster* yields similar results to KNN but executes more quickly and is less memory intensive. Like KNN, nearest cluster is used mostly for small problems.

Parzen's windows

Parzen's windows is a nonparametric architecture for classification and estimation pattern-recognition problems and is similar to the Gaussian and Gaussian mixture architectures. The method uses weighted averages of radial Gaussian basis functions to create the probability density function model. The method is less reliable than other algorithms when used with limited training data.

projection pursuit

Projection pursuit is nonparametric architecture based on regression analysis. Rather than developing a regression model with a large number of dimensions, the model development process is decomposed into an iterative process that develops a series of low-dimensional regression models designed to minimize error relative to the training data set.

radial basis functions

Radial basis functions (RBF) is a nonparametric architecture used for classification and estimation pattern-recognition problems. With RBF, probability density functions in the model are radial Gaussian distributions. RBF models can be trained using iterative gradient descent methods or more quickly by k means clustering algorithms. A *sigmoid* is an S-shaped curve, with a near-linear central response and saturating limits. Examples include the logistics function and the hyperbolic tangent (tanh).

Sigmoid

training algorithm

Training algorithms or just algorithms are used to tune free parameters in pattern-recognition models. (See also *architecture*.)

unimodal

A unimodal distribution is a distribution with one mode, such as the Gaussian distribution. The mode is the value of the random variable, x , at which the probability density function peaks. A unimodal distribution has only one peak, whereas multi-modal distributions (e.g., the bimodal) have multiple peaks.

Unimodal Gaussian classifier

The unimodal Gaussian classifier is a parametric pattern recognition method that relies on the assumption that the probability distributions for input vectors for each class are Gaussian. The output of the method is the probability density function of the input vector X given the class were known to be C_j . The method serves as a good benchmark for comparison of results from more complex methods.

Sources: Kennedy 1998, Lippmann 1994, StatSoft 2001, Winkler and Hays 1975.