Improving Process Operations Using Support Vector Machines and Decision Trees

Gorden T. Jemwa and Chris Aldrich
Dept. of Process Engineering, University of Stellenbosch, Matieland 7602, Stellenbosch, South Africa

DOI 10.1002/aic.10315
Published online in Wiley InterScience (www.interscience.wiley.com).

Statistical pattern-recognition methods are now widely applied in the analysis of process systems to achieve predictable and stable operating conditions. For example, multivariate statistical process control (MSPC) techniques use historical operating data to detect abnormal events, and assist engineers to focus their troubleshooting efforts to reduced subsets of variables in an otherwise broad operational space. Through an iterative process, it is hoped that the system variability remains bounded. Usually only a few samples collected under a state of statistical control are of interest, whereas the rest, which may be used to uncover potential improvement opportunities, are ignored. Beyond statistical control, an additional step is required to reduce the dispersion of process quality variables attributed to common causes. To achieve this goal, common and sustained causes not identified by MSPC must be interrogated. In this paper, a methodology based on kernel-based machine learning concepts is proposed to identify decision boundaries. A sparse set of instances or exemplars is identified that define a linear decision boundary in a feature space, which is equivalent to defining a nonlinear decision function in the associated input space. This is extended to defining operating strategies by integrating inductive learning into a decision support framework. Such an extension is founded on the fact that the success or failure of state-of-the-art approaches are invariably linked to the presence or absence of useful knowledge embedded in the system. © 2005 American Institute of Chemical Engineers *AIChE J,* 51: 526–543, 2005

Introduction

The importance of monitoring and management of quality in achieving business objectives is now widely recognized in process engineering. Large volumes of data being generated in plants invariably uncover useful insights and information if properly analyzed. Indeed, data analysis and information management processing are becoming key enablers for many firms in the competitive business environment. To this end, process operations are compelled to continuously challenge existing process performance levels and search for improvement opportunities. Ultimately, the goal is to ensure that both process capabilities and product quality are within the narrow operating or specified ranges.

Statistical pattern recognition provides a principled framework in the analysis, interpretation, and design of efficient decision support systems based on operating data. A group of methodologies that have proved useful in process engineering manufacturing include statistical process control (SPC), the Six Sigma approach to manufacturing, and other related statistical tools. These are almost used routinely in enhancing process performance and product quality among other critical factors necessary for successful operations. Statistical techniques provide tools to monitor process capability and are aimed at constraining the dynamic evolution of a process within a state of statistical control (or predictable control). Univariate statistical process control charts such as Shewart charts,[1] CUSUM plots,[2] and EWMA charts[3] compare current process performance against statistically derived control limits of normal

Correspondence concerning this article should be addressed to C. Aldrich at ca1@sun.ac.za.

process behavior. Multivariate statistical process control (MSPC) methods are an extension of the classical SPC methods that additionally exploit correlation information in process data.[4-7] These methods allow for a systematic search for, and subsequent elimination of, abnormal process conditions and other assignable causes. Eventually, the process system's behavior collapses into a region of bounded variability and exhibits a state of statistical control. Residual variation is then attributed to unavoidable or common causes that "cannot" be eliminated from the plant.

Saraiva and Stephanopoulos[8] discuss a framework for exploring possible improvement opportunities by challenging current performance levels and operating strategies. In particular, they suggest a *nonexperimental* empirical learning–based improvement strategy that integrates analogical reasoning and symbolic induction. Use is made of data collected under a "state-of-statistical-control." Although these data are normally of limited interest to both operators and process engineers, it has been estimated that the majority of process problems occur under this state. In determining the operating regions and/or operational strategies that yield potential improvement opportunities, a mapping relating process conditions and process trends is conceived using only information generated by the system. Such a mapping ideally must be nonlinear to adequately capture the underlying complex dynamics exhibited in practice.

Despite its appeal, the empirical learning approach is confronted by many issues; more often than not measured process data are discretely sampled (often nonuniformly), noisy, and have an unknown relationship to state variables. In addition, the measured data form a sparse distribution in a high-dimensional measurement space, resulting in ill-posed problems. Also, the underlying process descriptive model may not have a simple closed representation. Perhaps an even more fundamental issue is that, although data used in learning the decision function (training data) may result in a consistent hypothesis, they may incorrectly classify yet-to-be-seen future data. This is a well-known problem in most learning algorithms, such as neural networks, decision trees, and the like. In the last few years developments in statistical learning theory have presented a different approach to the learning problem. Instead of optimizing the performance of the decision function over training data, a different property—*generalization capacity*—is optimized. Support vector machines (SVMs) possess a number of desirable properties that will be shown to be particularly attractive in the specific problem to be addressed in the present work.

In this paper we present an operational process methodology that searches for process improvement opportunities by systematically reducing variances in measured process variables. Support vector learning theory is used in the identification of the pivotal features defining separation boundaries between different regions of variable space. These pivotal "features," called support vectors (SVs), contain discriminative information separating different classes, which allows for a sparse solution representation. Using the identified SVs, a mapping relating process conditions to process trends is then formulated using classification decision trees.

In the following sections a review of the general statistical pattern-recognition problem is presented. Key ideas from the original methodology proposed in Saraiva and Stephanopoulos,[8] which inspired this work, are highlighted, including a brief overview of classification decision trees. Machine learning concepts foundational to support vector formulations are discussed using a binary classification problem on nonseparable data. Some examples are introduced to clarify certain elements, leading to a detailed discussion of the proposed methodology. We illustrate various elements of the proposed methodology using a simple simulated first-order reaction occurring in a continuous stirred tank reactor (CSTR). Finally, a demonstration on the application of the approach to data generated from a process operation is given.

The Statistical Pattern-Recognition Problem

Pattern recognition is a powerful and useful tool for interpreting process data. Instead of logical deduction (as in expert systems) pattern recognition involves inductive reasoning by extracting generalization rules using historical operating data. An important characteristic of pattern recognition is extraction of low-level detail of process behavior hidden in data.

The pattern-recognition task can be formulated as follows: Given an input pattern with M feature variables or attributes associated with the state of the process, assign the pattern into at least one distinct class that is either predefined (supervised) or unknown (unsupervised).[9] (The terms *features* and *attributes* are used interchangeably in pattern-recognition literature. However, in this article we reserve the term "feature" for a pattern mapped into a higher-dimensional space (the Reproducing Kernel Hilbert Space) as discussed later.) We will focus on supervised classification, where the classes are specified by the designer and the inductive principle is learned from a training set of known input–output pairs. Furthermore, it is assumed the measured attributes are relevant to the classification task. Mathematically, the pattern-recognition task can be formulated as defining discriminant functions $d(\mathbf{x})$, which establish decision boundaries in the input space S. For an n-class problem, the input space is thus partitioned into n disjoint regions $S(i)$, such that

$$S(i) \cap S(j) = \varnothing \qquad i, j = 1, 2, \ldots, n \text{ and } i \neq j \quad (1)$$

with

$$\bigcup_{i=1}^{n} S(i) = S \quad (2)$$

A pattern is thus assigned to a class $S(i)$ if and only if

$$d_i(x) > d_j(x) \qquad \text{for all } i \neq j \quad (3)$$

Different techniques have been proposed to solve the general statistical pattern recognition as formulated above. These can be broadly grouped as either *parametric* or *nonparametric*. Each chosen approach has its merits. It is generally accepted that nonparametric approaches perform better than parametric methods.[9,10] However, they imposed a computational overload because improved performance requires retaining into memory a large database of patterns. Also, it is difficult to gain insight into the underlying process behavior using nonparametric ap-

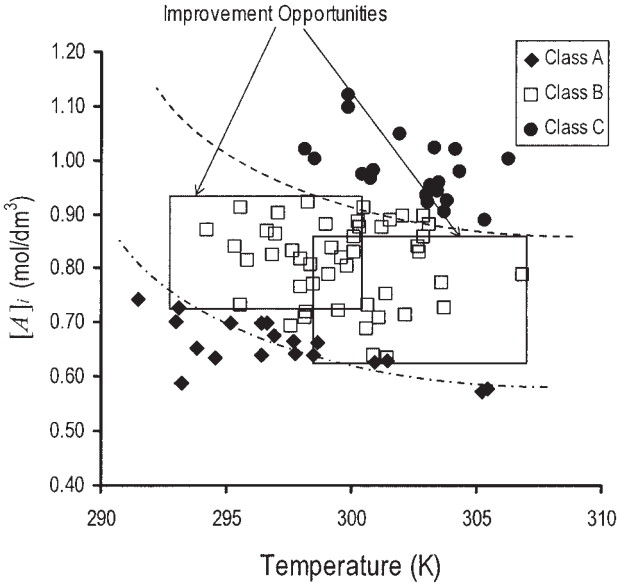

Figure 1. Performance improvement through partitioning of regions of space embedding process data points.

proaches. In certain applications where a smooth discriminant function is appropriate, it is advisable to use parametric approaches such as neural networks. Later we present the theoretical framework of support vector machines that, although strictly a parametric approach, have some nonparametric properties.

Process Improvement Strategies Using Classification of Operating Regions

The problem of process improvements can be reduced to a pattern classification task. Classification or partitioning of the feature space, using historical records collected during "normal" operating periods, can be used to detect regions of space in which the process would have a reduced variability.

Saraiva and Stephanopoulos[8] proposed a decision-support system for the continuous detection and formulation of process improvement opportunities for physical systems. Their methodology uses a machine-learning approach, integrating an instance-based learning component and an inductive symbolic learning procedure, to complement statistical tools used in process monitoring and fault detection and diagnosis.

Figure 1 illustrates the ideas underlying the approach. The process depicted is operating in a state of statistical control. Acceptance of the final product is determined by a suitable quality index, which assigns the products to one of three categories A, B, and C, of which the desirable products fall into class B. This delineation allows for the definition of decision boundaries between classes A and B, and classes B and C. In the absence of an appropriate fundamental model, an empirical-learning approach can be used to define implicit decision rules, which partition the operating zones into the designated classes. An essential requirement of the methodology is correct interpretation of the data that direct the operator to strategies that offer most promising and interesting hyper-rectangular zones in the decision space for improved process performance.

Saraiva and Stephanopoulos[8] suggest the identification of pivotal data points nearer the decision boundary surfaces using a nonparametric nearest-neighbor classifier. Only those points that lie close to the boundary surface *and* form *Tomek links* are used to form an active memory of exemplars that are subsequently used to induce a decision tree. (*Decision trees* are symbolic classifiers that provide a modularized description of the feature space with characteristics that make them suited to the requirements of the methodology. We will discuss these and other properties of decision trees in a subsequent section.) Eventually, the approach suggests changes to current operating strategies and/or design of a reduced set of confirmatory experiments.

Nonetheless, the methodology of Saraiva and Stephanopoulos[8] has certain limitations. First, the use of Tomek links results in a piecewise linear classifier. This may not be appropriate in instances where the decision boundary is correctly described by a nonlinear function. Also, the process engineer has no direct control over the number of training points necessary to induce a decision tree. This is particularly relevant in cases where there are uncertainties in the data. A related issue is that for systems with many variables, larger amounts of training data are required to identify the correct decision hyperplanes. In our experiments (and as suggested in the original work) the methodology has rather slow adaptive properties, making it unsuited for processes with rapidly changing parameters such as an ore milling plant getting feed from different ore bodies.

The basic elements used in the original development are based on two pattern-recognition tools: classification decision trees and memory-based pattern classification. Although we retain the classification decision tree, our proposal uses support vector machines for determining the class boundaries or discriminant functions. The theory and properties of decision trees and support vector machines are discussed next with emphasis on those elements essential in the implementation of the methodology.

Inductive learning using decision trees: a brief overview

Decision trees are hierarchical and structured classifiers that determine classification rules by partitioning the feature space using piecewise hyperlinear decision boundaries. Given a set of process trends or features and corresponding process conditions, a decision tree extracts generalized rules through mapping of the process features and quality variables. Decision trees discover classification rules by using a top-down, divide-and-conquer strategy that partitions the given set of objects into progressively smaller subsets in step with the growth of the tree. The derived decision rules are expressed in the form of complexes or conjunctions of conditions amenable to easy interpretation and implementation.

Figure 2 shows a typical binary decision tree induced using an m-dimensional space of feature vectors $\mathbf{x}_i = [x_1, x_2, \ldots, x_m]$, $i = 1, \ldots, N$. The decision tree identifies IF–THEN rules using splitting criteria that partition the data set at a node into two maximally homogeneous subsets. Table 1 is the corresponding decision list extracted from the tree. Once defined, the decision rule can be incorporated into an operational strategy. The splitting rules are based only on the pivotal attributes or variables and, therefore, easy to interpret. This is particularly relevant in situations where data are correlated and therefore

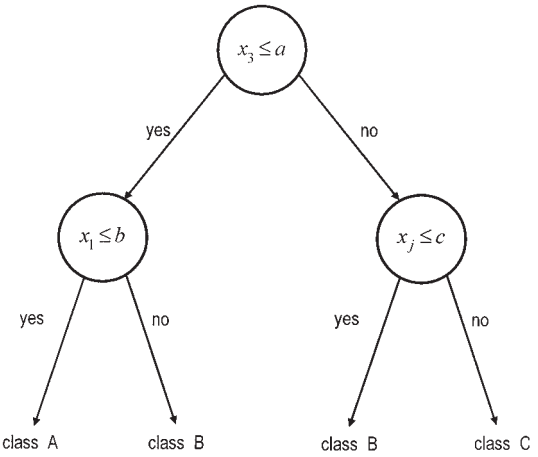

Figure 2. Rule extraction using binary decision trees.

contain redundant information, a common characteristic of most multivariate measurements. Construction of a binary decision tree revolves on a few issues: specification of splitting rules at test nodes, termination criteria, and class assignment rules for the terminal nodes. There exist several splitting rules that have been implemented in many algorithms such as ID3, C4, and CART.[11-14]

Although decision trees possess a number of features that are useful in engineering applications,[15] they have certain structural limitations that may complicate their use. In particular, the divide-and-conquer strategy does not guarantee an optimal decision tree. It can be difficult to extract intelligible rules from a large and complex tree; and the piecewise linear decision boundaries associated with decision trees may be inadequate for boundaries better defined by nonlinear functions. Fortunately, it is possible to limit these restrictions by integrating decision trees and other empirical learning tools with the desired properties, such as support vector machines.

Support Vector Machines

The support vector machine (SVM) is a learning algorithm introduced by Vapnik and coworkers using a firm statistical learning theory approach.[16,17] In particular, they showed that the problem of function estimation (encountered in pattern-recognition, regression, and density-estimation tasks) can be conducted in a general statistical framework that minimizes an upper bound on the expected loss using observed data. In the following we illustrate the main ideas behind SVMs for a binary nonseparable classification problem. Comprehensive descriptions can be found in Burges,[18] Cristianini and Shawe-Taylor,[19] and Schölkopf and Smola.[20]

Given observed or training data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, N$, generated according to a (unknown) probability distribution $P(\mathbf{x}, y)$ with $\mathbf{x}_i \in \Re^m$ and $y_i \in \{-1, 1\}$, the pattern-recognition task finds a decision function $h(\mathbf{x})$ that associates the vector of attributes $\mathbf{x}_i$ to the target classes y_i. The classical approach finds the optimal decision boundary using realizations of the empirical risk minimization (ERM) induction principle, such as least-squares and maximum likelihood. The ERM principle optimizes the performance of the classifier on a training set. However, it is not guaranteed that such a classifier will perform as well when presented with the new data. It is possible to constrain the selected decision function to ensure good generalization properties. For example, one can introduce a regularization term to limit the complexity of the set of hypotheses H from which h is chosen. Such a heuristic defines a trade-off between complexity and accuracy of the selected hypothesis.

An alternative induction principle developed in statistical learning theory is the structural risk minimization (SRM).[17] Here, the complexity–accuracy trade-off is motivated by imposing statistical bounds on the generalization error. In other words, rather than minimizing the error on a training set (ERM principle), support vector machine learning seeks to minimize the expected error of misclassifying yet-to-be-seen data. It is this property that makes the SRM principle superior to the classical ERM principle.[21] The statistical formulation of SVM learning is derived from a sound theoretical basis and, therefore, avoids the dangers of a heuristic that may be based on misleading intuition.[19] However, as with all empirical learning methods, the approach is only as good as the data used.

The SVM algorithm finds the decision function as a linear combination of certain points of the training set (called *support vectors*), which condense the information contained in the training set. The decision boundary, which is linear in some possibly high-dimensional feature space, is obtained from solving a quadratic programming problem that depends on a regularization parameter. The main ideas behind support vector classification (SVC) are discussed next for the case of a binary nonseparable classification problem.

SVC of a binary nonseparable classification problem

The plane separating the classes (allowing for a few errors or misclassifications) is called the *separating hyperplane*. It is possible to define many such hyperplanes that can classify the two classes. Defining $d_+(d_-)$ as the shortest distance to a separating hyperplane, the *margin* of a separating hyperplane is then $(d_+ - d_-)$. It can be shown that the shortest distance and the weight vector specifying the direction of the hyperplane are related through $d_+(d_-) = 1/\|\mathbf{w}\|$. Thus, maximizing the margin is equivalent to minimizing the term $\|\mathbf{w}\|/2$. Assuming a 0/1 loss function the size of the allowable errors can be controlled by introducing a regularizing term $C \sum_{i=1}^{N} \xi_i$, where C is a constant and ξ_i is a slack variable (quantifying how far a vector is relative to the margin). The support vector algorithm searches for an optimal separating hyperplane that simultaneously maximizes the margin and minimizes the errors.

Mathematically, this can be formulated as follows: Given a linearly nonseparable training set $\{\mathbf{x}_i, y_i\}$, $i = 1, \ldots, N$, the hyperplane $(\mathbf{w}, b)$ that solves the optimization problem

$$\min_{(\mathbf{w},b)} \left(\frac{1}{2} \|\mathbf{w}\| + C \sum_{i=1}^{N} \xi_i \right)$$

Table 1. Summary of Decision Rules Induced by the Classification Tree in Figure 2

Rule	Antecedent (IF)	Consequent (THEN)
1	$x_3 \le a$ AND $x_1 \le b$	**x** belongs to class A
2	$x_3 \le a$ AND $x_1 > b$	**x** belongs to class B
3	$x_3 > a$ AND $x_j \le c$, $j \le M$	**x** belongs to class B
4	$x_3 > a$ AND $x_j > c$, $j \le M$	**x** belongs to class C

$$\text{subject to } \begin{cases} y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \\ 0 \leq \xi_i \qquad i = 1, \ldots, N \end{cases} \tag{4}$$

realizes the optimal margin hyperplane with geometric margin $\gamma = 1/\|\mathbf{w}\|$, where $\langle \cdot, \cdot \rangle$ is a dot product function. (In this study we restrict ourselves to the Euclidean 2-norm.)

The solution to the optimization problem in Eq. 4 can be solved by means of the classical method of Lagrange multipliers.[22] More specifically, denoting $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_2)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_N)$ as the nonnegative Lagrange multipliers associated with the constraints in Eq. 4, the solution to the optimization problem is equivalent to determining the saddle point of the primal Lagrange functional

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \tfrac{1}{2}\|\mathbf{w}\| + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha[\, y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 + \xi_i] - \sum_{i=1}^{N} \beta_i \xi_i \tag{5}$$

The corresponding dual formulation of Eq. 5 is found by noting that at the saddle point $L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ has a minimum for $\mathbf{w}$ and b and a maximum for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$

$$\max_{(\boldsymbol{\alpha}, \boldsymbol{\beta})} [\min_{(\mathbf{w}, b, \boldsymbol{\xi})} L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})] \tag{6}$$

Therefore, differentiating Eq. 5 with respect to $\mathbf{w}$ and b, respectively, gives

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial b} = 0 \Rightarrow 0 = \sum_{i=1}^{N} \alpha_i y_i$$

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \boldsymbol{\xi}} = 0 \Rightarrow \alpha_i + \beta_i = C \tag{7}$$

Finally the dual formulation thus obtained is obtained by substituting the relations in Eq. 7 into the primal function (Eq. 5)

$$\min \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^{N} \alpha_i$$

$$\text{subject to } \begin{cases} \sum_{i=1}^{N} y_i \alpha_i = 0 \\ 0 \leq \alpha_i \leq C \qquad i = 1, \ldots, N \end{cases} \tag{8}$$

The weight vector $\mathbf{w}$ is a linear combination of the training points, as indicated in Eq. 7. The value of the bias term at the optimal point is found by applying the Karush–Kuhn–Tucker (KKT) complementarity conditions[19,20]

$$\alpha_i[\, y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1 + \xi_i] = 0, \qquad i = 1, \ldots, N \tag{9}$$

$$\xi_i(\alpha_i - C) = 0 \qquad i = 1, \ldots, N \tag{10}$$

Therefore, only inputs $\mathbf{x}_i$ for nonzero α_i contribute to the decision function, which results in a sparse representation of the solution to the optimization problem. Points with nonzero α_i are referred to as *support vectors* (SVs) and are informative patterns relevant to the classification task. The magnitude of each α_i determines the importance of that pattern in the decision function, with large $|\alpha_i|$ values being more important.

Given a test pattern $\mathbf{x}$, the decision function for the binary classification problem is then obtained as

$$f(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b) = \text{sgn}\left(\sum_{i \in SVs} y_i \alpha_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b \right) \tag{11}$$

We can further characterize the SVs using the KKT conditions (Eqs. 9 and 10) and the values of ξ_i. For $0 < \alpha_i < C$, condition 9 leads to $\xi_i = 0$. These SVs lie on the margin, which is at a distance $\gamma = 1/\|\mathbf{w}\|$ from the optimal separating hyperplane; thus these are referred to as *margin vectors*. Support vectors with $0 < \xi_i \leq 1$ and $\alpha_i = C$ are correctly classified vectors but lie a distance inside the margin. Support vectors with $\xi_i > 1$ and $\alpha_i = C$ are misclassified points, or error support vectors. Figure 3 illustrates the characterizations of the support vectors for a nonseparable problem.

All the support vectors are involved in the decision function (Eq. 11), except for the determination of the bias, where only the margin support vectors are used. This property of support vector classification will be used later for the identification of

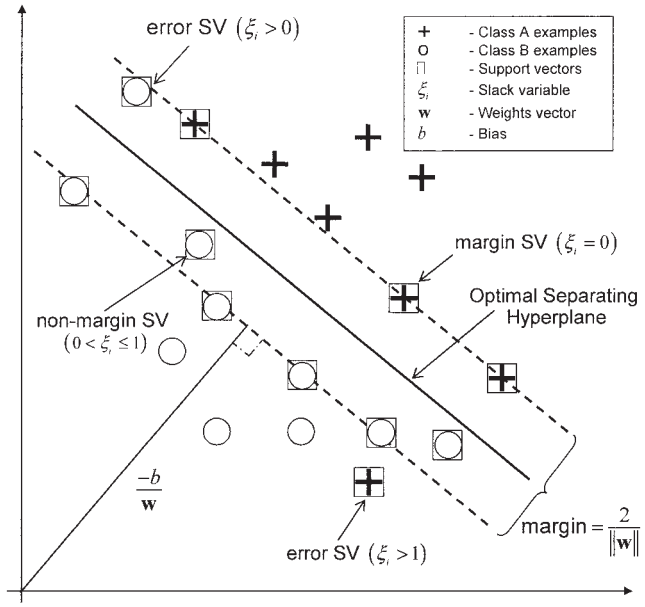

Figure 3. Characterization of the different support vectors for a nonseparable problem.

potential outliers that may compromise the implementation of the methodology.

Nonlinear extensions to the basic SVC formulation

The procedure outlined above can be extended to determining nonlinear function 11 in input space by mapping the input points into a higher-dimensional feature space and solving the optimization in its dual representation in that space. To understand how the foregoing formulations are extended to nonlinear functions, it can be noted that the data points occur as only inner or dot products in both the dual-objective and decision functions. Kernel representation implicitly defines a dot-product feature space corresponding to a nonlinear mapping in the input space.[23] A kernel is a function K, such that for all $\mathbf{x}$, $\mathbf{z}$ in the input space X

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle \tag{12}$$

where ϕ is a mapping from X to a (dot-product) feature space F:

$$F = \{\mathbf{x} \rightarrow \phi(\mathbf{x})\} \qquad \mathbf{x} \in \Re^m, \phi(\mathbf{x}) \in \Re^{n_h} \tag{13}$$

An important consequence of kernel representation is that the underlying map ϕ or dimension of the feature space n_h does not need to be known. As a result, computational difficulties that may have arisen in evaluating the kernel are avoided. For a function to be a valid kernel, it is sufficient to show that it satisfies Mercer's conditions.[24] Examples of commonly used kernel functions include Gaussian radial basis function and polynomial functions.

The kernel representation of the soft-margin optimization problem (Eq. 8) is therefore

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i=1}^{N} \alpha_i$$

$$\text{subject to} \begin{cases} \sum_{i=1}^{N} y_i \alpha_i = 0 \\ 0 \le \alpha_i \le C \qquad i = 1, \ldots, N \end{cases} \tag{14}$$

Given a test pattern $\mathbf{x}$ the decision function is determined as

$$f(\mathbf{x}) = \text{sgn}\left(\sum_{i \in SVs} y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + b \right) \tag{15}$$

Use of kernel mapping (Eq. 12) maps the otherwise nonlinear problem into a higher-dimensional feature space, where the solution can be found using linear methods. Computations in the feature space then correspond to nonlinear functions in the input space.

Many algorithmic implementations of support vector machines are now available (such as www.kernel-machines.org), which can be used as libraries for most applications. For the work reported here, we used a modified object-oriented–programming framework based on the MATLAB SVM toolbox by Cawley.[25]

Multiclass classification using support vector machines

The optimal margin classifier or support vector machine was originally formulated for binary classification problems.[26] Extensions to multiclass problems have been proposed (for example, in Platt et al.[27]), although it is still largely an open research issue. In general, two approaches exist: *one-vs.-all* and *one-vs.-one*.

The one-vs.-all method constructs and combines several binary classifiers, where the ith support vector classifier has all patterns belonging to class i as positive labels and the rest negative. Thus for an n-class problem n support vector classifiers are constructed. The output of the classifiers is the class that corresponds to the support vector classifier with the highest output value.

Alternatively, the one-vs.-one method constructs all possible binary classifiers from the n-class data set. Each classifier is trained on only two of the n classes, resulting in a total of $n(n-1)/2$ classifiers. To classify a pattern, these classifiers must be combined, for which various algorithms have been suggested (such as the Max Wins method), or directed acyclic graph support vector machine (DAGSVM).[27] Each method used has its merits and disadvantages. A comparison of some of the approaches based on numerical performance can also be found in Hsu and Lin.[28] The online methodology proposed later requires good generalization, while simultaneously being efficient. For these reasons it was decided to use the DAGSVM approach.

Identification of Improvement Opportunities with SVMs

Figure 4 illustrates a schematic representation of the continuous process improvement methodology proposed in this paper. The overall methodology closely resembles the proposal in Saraiva and Stephanopoulos,[8] with a few exceptions. To understand the minor difference it must be appreciated that at the center of the original formulation is the pattern-recognition module used in building an active memory of cases that lie close to the boundary of the decision hyperplanes. As discussed earlier, the original approach leaves little room for the process engineer to adjust the resulting suggested improvements.

The modifications proposed use support vector classification in the selection of the memory of examples used in the induction of trees. As indicated, SVC has different elements that must be decided on; the choice of the kernel, associated kernel parameters, and outlier filtering detection. This gives more control to the operator/process engineer on the evolution of the suggested process changes. Because the methodology is similar to the original except for the pattern recognition, the discussion that follows focuses on the proposed innovations and associated advantages. We implemented the other elements principally in a similar fashion to Saraiva and Stephanopoulos.[8]

Description and illustration of methodology

The main differences introduced in our approach are with regard to the pattern-recognition task and subsequent selection of prototypes or memory of exemplars used in the induction of

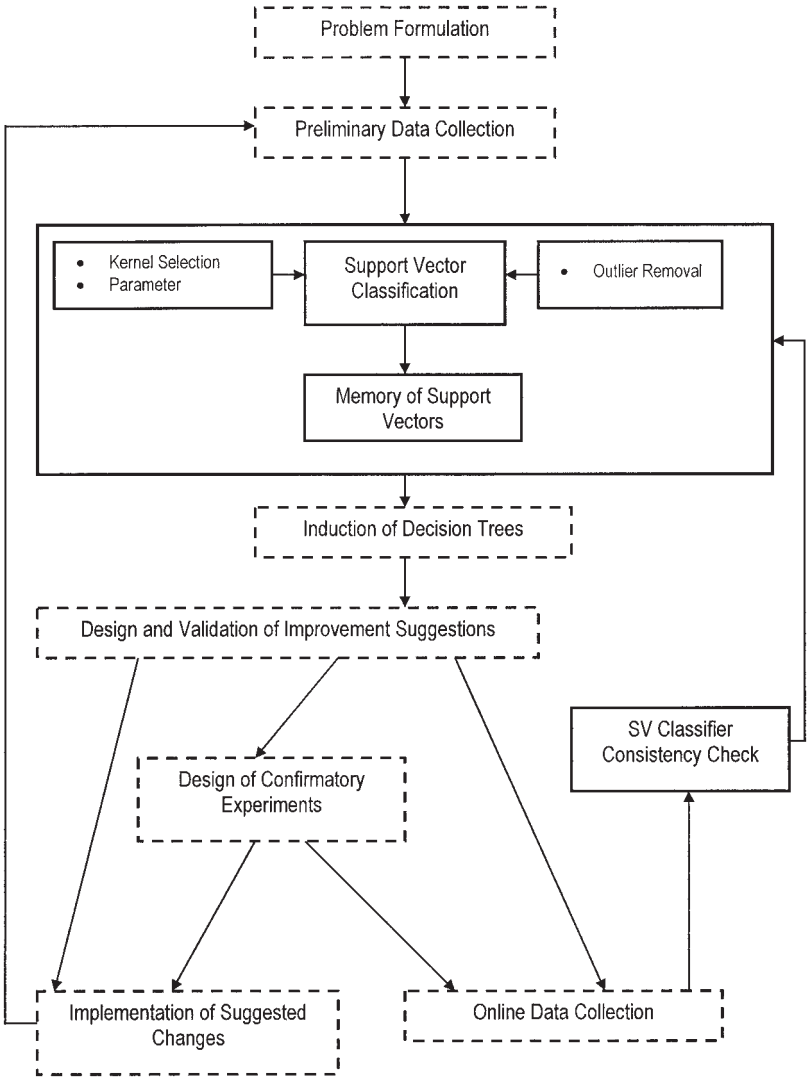

Figure 4. Elements and relationships of the units involved in the process improvement methodology.

classification decision trees. Although the focus of the present contribution is on process improvement by exploiting information in process data, the proposed innovations can also be used as separate modules for other related activities, such as fault detection and diagnosis, pattern-based adaptive control, and so forth.

As in Saraiva and Stephanopoulos[8] we use a simulated first-order irreversible reaction ($A \rightarrow B$) occurring in a continuous stirred tank system to demonstrate the approach. An Arrhenius relationship is assumed for the reaction rate, with activation energy of 99,700 J/mol. Furthermore, the feed stream is assumed to contain reactant A only. The concentration of B in the output stream ($[B]$) is tracked at regular intervals as an index for current process performance. The process performance or $[B]$ is a function of four process variables: the reactor temperature, T (K); concentration of species A in the feed stream, $[A]_i$ (mol/dm^3); the volumetric flow rate of the feed and output stream, Q (m^3/s); and the level of fluid in the reactor, L (m).

To allow for visualization of the decision boundaries, we restricted the input dimensionality to only two variables. Thus, the volumetric flow rate and reactor fluid levels were fixed and process operating data were generated using a Monte Carlo simulator for the following Gaussian distributions, $N(\mu, \sigma)$: $[A]_i \sim N(0.8 \text{ mol/dm}^3, 0.1 \text{ mol/dm}^3)$, $T \sim N(300 \text{ K}, 3.5 \text{ K})$. For illustrating some of the aspects of the modifications, we use training and testing sets of sizes 750 and 250 data points, respectively.

Figure 5 is a scatter plot of the process variables using the simulated data. Also shown are the 95 and 99% confidence intervals for the process under the given conditions. These were derived using established MSPC techniques. Clearly, the process trajectory seems to be evolving as dictated by the imposed control strategy. It may not, however, be immediately obvious how the data can be exploited for improvement opportunities, unless other techniques are used. Next we illustrate various elements of the proposed modifications leading to a descriptive functional representation of the complete methodology.

Problem formulation

For a real-world process, categorizing products and/or processes into classes using predefined criteria is context depen-

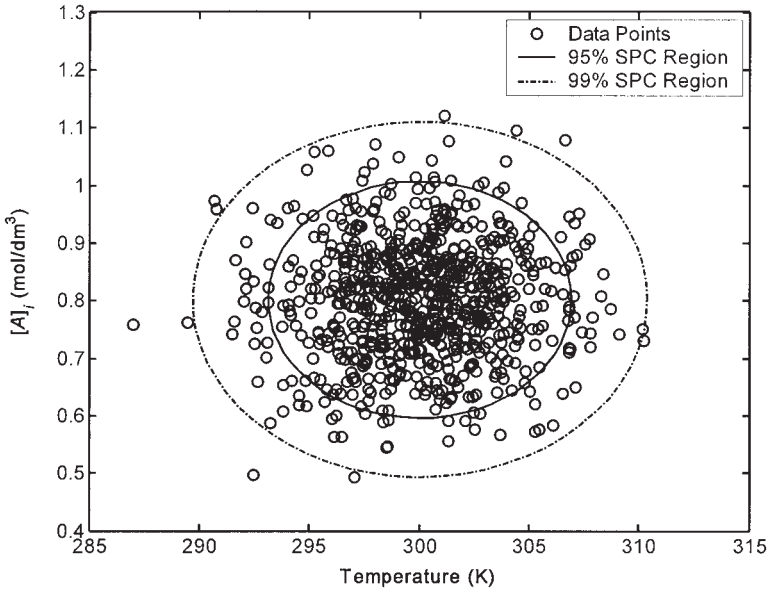

Figure 5. Scatter plot of data representing the simulated behavior of a first-order reaction in a continuous stirred tank reactor (CSTR).

The 95% (solid line) and 99% (broken line) confidence levels enclose most of the data, an indication of a process under statistical control.

dent. There are several approaches that can be used. A reasonable method is based on the distribution statistics of the quality or performance variables. In the illustrative example presented above, we assign the current state of the process into three classes, depending on the range in which $[B]$ lies: Class A if $[B] < \mu_B - \sigma_B$, class B if $\mu_B - \sigma_B \leq [B] \leq \mu_B + \sigma_B$, and class C if $\mu_B + \sigma_B < [B]$, where A, B, and C correspond to "low," "normal," and "high," respectively, and μ_B and σ_B are the mean and standard deviation of the quality index. We are mostly interested in "normal" instances. The resulting class separation is shown in Figure 6a, where we have also plotted the true class boundaries between the different classes. For the two-dimensional problem, one can easily see that a possible improvement would be to restrict the variation of the $[A]_i$ and T in the ranges 0.7–0.9 mol/dm^3 and 296–304 K, respectively. The relevance and importance of the strategy becomes even more significant when the feature space is high dimensional and relationships between the process variables are not as clear-cut.

Using classification decision trees (for inductive symbolic learning), explicit rules for assigning individual cases to the appropriate class can be extracted from the resulting partitioning obtained as shown in Figure 6b. The corresponding decision tree is illustrated in Figure 7, where for simplicity patterns belonging to classes A and B have been grouped into a single set A′. The corresponding rules for assigning a case to class B derived from the tree are summarized in Table 2. Using these rules, the process engineer can search and formulate operational suggestions for process improvement. Although the procedure is simple, it is difficult to design a supervisory control or similar decision support system for online application, arising from the high computational cost and complex decision rules, particularly for high-dimensional systems. Identification of a sparse and informative representation of the operating data can result in an efficient and robust implementation for reasons mentioned earlier.

Identification of sparse informative patterns

Support vector classification (SVC) is a particularly suitable method for this purpose. As explained earlier, SVC identifies a reduced subset of the training data set containing the discriminative information in the data. The decision function is expressed as a linear combination of a few of the training patterns in some feature space (Eq. 12). Kernel mapping (Eq. 9) transforms the linear solution in the feature space to a nonlinear function in the original input space.

Using the same data introduced earlier leads to the results of SVC (Figure 8). Gaussian radial basis kernels with a width of $\sigma = 0.1$ were used to derive the decision boundaries (solid lines) for a regularization constant $C = 10$. Using these support vectors for the induction of decision trees and subsequent refining[8] yielded the results shown in Figure 9 and the corresponding rules for assigning a case to mostly class B are given in Table 3. Also, Figure 9 demonstrates a typical conjunctive rule generated by the system

IF	$0.73 \leq [A]_i < 0.97$
THEN	restrict temperature variation within 295–300 K

The decision rules induced using support vectors result in similar operational objectives as before, even though only fewer samples were used in the inductive learning. This property is particularly appealing for an online approach discussed later.

Detection and filtering of outliers

An important aspect of the new method is the identification and removal of outliers in the data set. From the discussion on constructing a support vector machine, it will be remembered that it is possible to design a soft-margin classifier with slack variables to allow for possible measurement errors through the

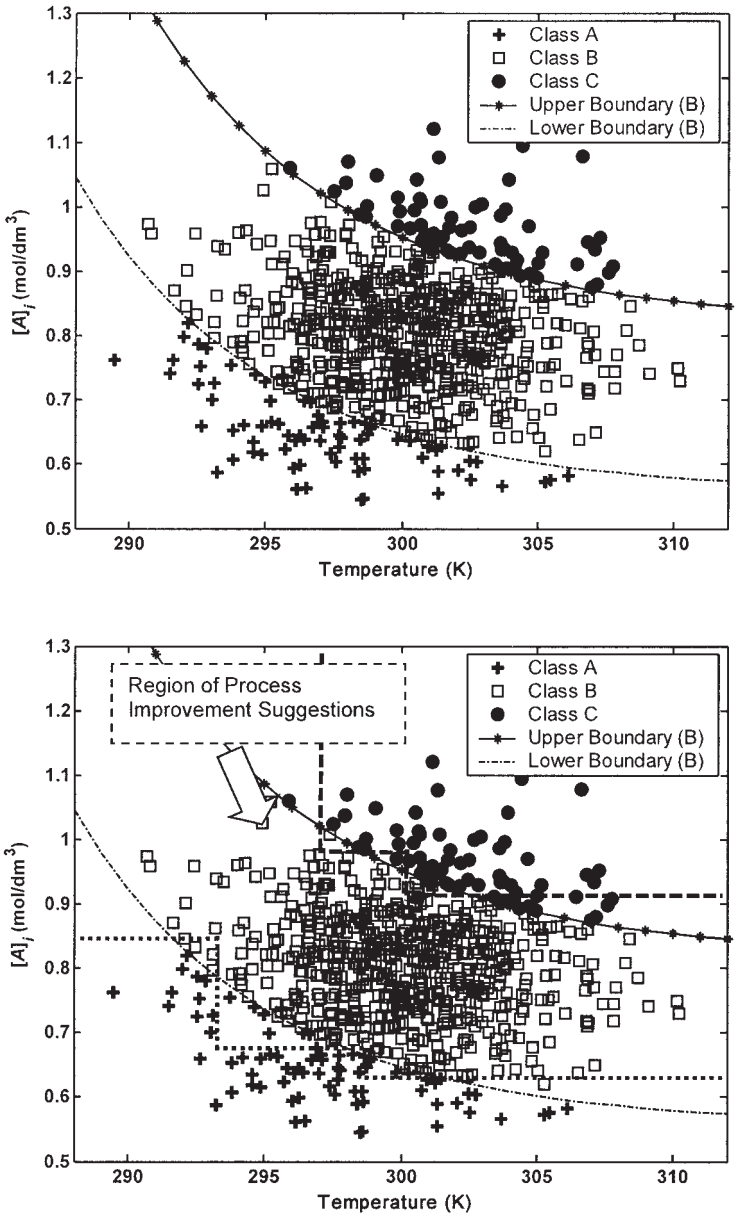

Figure 6. (a) Problem formulation of the simulated first-order CSTR system based on first- and second-order moments estimated from historical process data. (b) Regions of space identified using decision trees for process improvement suggestions (see also Figure 7 and Table 2).

optimization of Eq. 4. As $C \rightarrow \infty$ the solution obtained minimizes the misclassification error width, allowing for as few errors as practically possible. On the other hand, when $C \rightarrow 0$ margin maximization dominates and as many errors as possible are allowed. For support vectors on the margin, the Lagrange multiplier is constrained to $0 < \alpha_i < C$; otherwise, $\alpha_i = C$. This feature can be used advantageously to rank support vectors with respect to how important they are in determining the decision surface.

However, use of this property may not be appropriate in outlier detection because some support vectors, although correctly classified, also have $\alpha_i = C$. Because we are interested in the information provided by these near-misses, another approach is desirable.

A property that has received little attention in support vector machine applications is the information provided by the values of the slack variables (Eq. 7). Patterns with $|\xi_i| \leq 1$ lie within the margin and are correctly classified and, therefore, convey important information on the decision boundary. However, patterns with $1 < |\xi_i| < 2$ are incorrectly classified, but still within the margin. Patterns with $|\xi_i| > 2$ are embedded in a different class and, therefore, are immediate targets for elimination. The threshold value of ξ_i for patterns to be removed can be decided by process experts familiar with the dynamics of the operation, providing an additional control on the decision support system (choice of kernel, kernel parameter, and regularization constant being the others). It should be noted, however, that when there is a shift in the process dynamics, all incoming

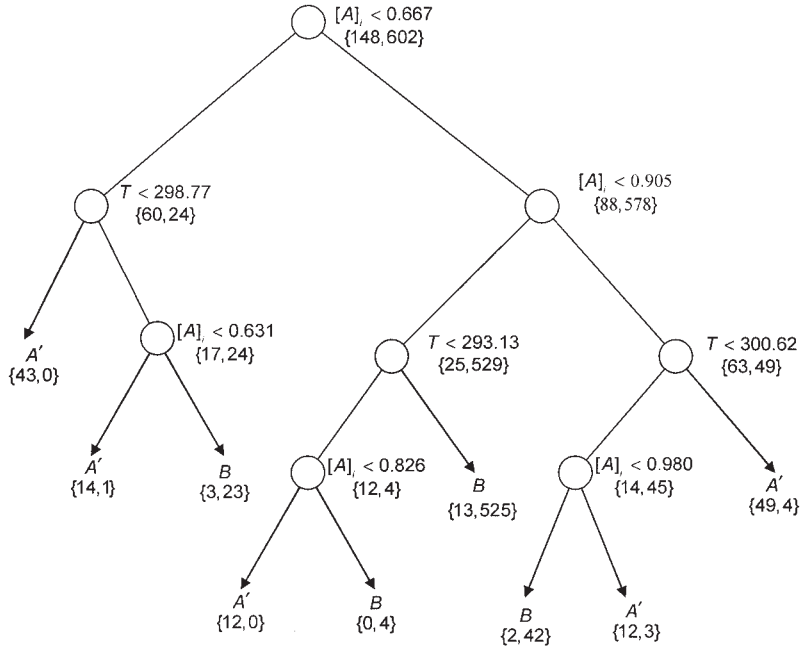

Figure 7. Induction of decision tree using the training data set in Figure 6a.

The tree partitions the data into two minimally homogenous subsets A′ and B, where A′ is the union of classes A and C. The term $\{i, j\}$ below the test node conditions and terminal nodes represents the number of patterns in classes A′ and B, respectively, at that point.

data will be considered outliers and are thus removed. A strategy should thus be devised to identify the occurrence of such a shift, so that these points are not disregarded. An example of how this can be done is discussed below under the online implementation of the methodology.

Figure 10a shows a typical scenario when there are incorrectly measured patterns in the process database used to initialize the online decision support system for process improvement. After training the support vector classifier, these patterns will be included in the memory of patterns used in the symbolic inductive learning phase. Using the values of the slack variables from the optimization algorithm, these samples can be identified and removed from consideration (Figure 10b).

Previous studies of novelty detection using support vectors have used one-class SVM structures,[20] where the learning problem is formulated as a density estimation problem. Similar to the support vector classification scheme, the solution of the problem ultimately involves solving a convex quadratic programming equation. Because this involves greater computational cost, we did not consider integrating this approach in the framework. In any case, the task at hand involves processes under statistical control and any abnormal event would be identified in complementary modules.

Table 2. Simplified Decision Rules Leading to Mostly Class B in Figures 6b and 7

$[A]_i$ (mol/dm^3)	T (K)	Class
>0.631	≤298.8	B
>0.826	≤293.1	B
≤0.905	>293.1	B
>0.980	≤297.4	B

Adaptive characteristics/evolution of memory of support vectors

Changes in process conditions, such as water quality, ore type, or chemical reagent specifications, may result in process drift inconsistent with prevailing decision boundary definitions. Moreover, it may happen that some of the initially identified support vectors may be inaccurate measurements, which may pass undetected by the outlier filtering step. This is particularly so when these patterns are located at the boundaries of the feature space. As more information becomes available from online data collection, the initial decision boundary surfaces need to be adjusted to reflect an accurate picture of the underlying process behavior. Support vector classification uses all training samples (as a kernel matrix) in finding a solution to the optimization problem. As the number of training samples increases, so do the overhead costs on computation. For batch training, heuristics such as chunking and decomposition, sequential minimization optimization, and the like, have been proposed to simplify the problem. The framework for online support vector learning can easily be constructed[19,20] and the methodology proposed here uses a pseudo-batch online training strategy.

As each new data pattern is collected, it is classified using the present support vector decision boundaries. An SVC update criterion is defined using the rate of misclassification over a user-specified window interval. Too large a window may result in fewer updates than necessary, whereas too small a window may result in more updates than necessary. Therefore, the optimal window size should be chosen by the operators conversant with the process, or by trial and error. Another critical index to monitor is the growth rate of the support vectors. For a process under statistical control there should not be large

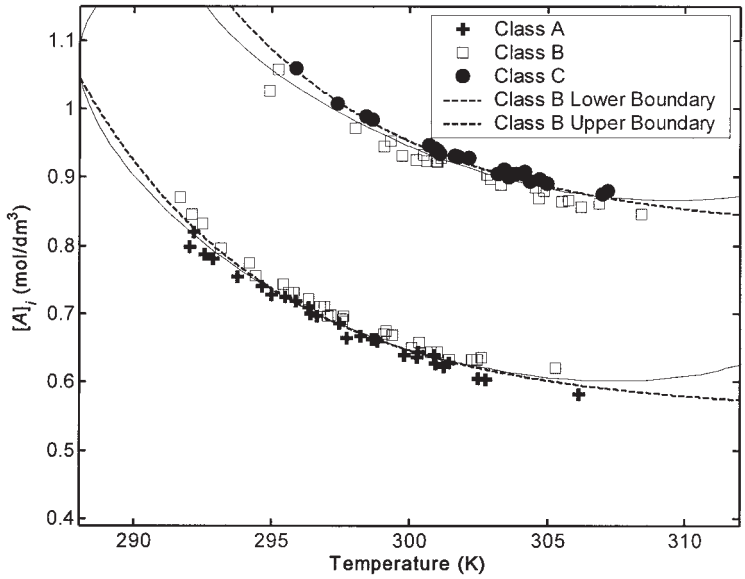

Figure 8. Support vector classification showing the support vectors identified in separating Class B from Classes A and C (bottom and top solid lines, respectively), with the true boundaries indicated by corresponding broken lines.

fluctuations in the number of support vectors necessary to define the system. Thus, an unusually large shift in the number of support vectors could indicate novel information not previously explained. Furthermore, if there is a persistent positive differential in the growth rate of the support vectors, then it may safely be assumed that the underlying process dynamics are changing. In this case, a decision will have to be made on how to update the definition of the decision boundaries. There are two alternatives: (1) Redefine the decision boundaries using all previously seen points, or (2) select an appropriate subset to reinitialize the support vector machine. Besides being computationally expensive, the first option also leads to retention of class patterns that overlap different classes. Selection of a reduced subset of past operating data remains the principled option, but raises other complicated issues such as the size of the time window and dealing with the situation when a time window does not contain patterns of all classes. We propose a user-specified time window and the retention of information in the current set of support vectors of a particular class that may be lacking from the time window. Our experiments have indicated that the exclusion of information from all classes results in a degenerate decision-support system, which collapses when the patterns from the absent class do appear. This degeneracy is related to the method used in building multiclass support vector machines (see discussion on multiclass support vector machines above).

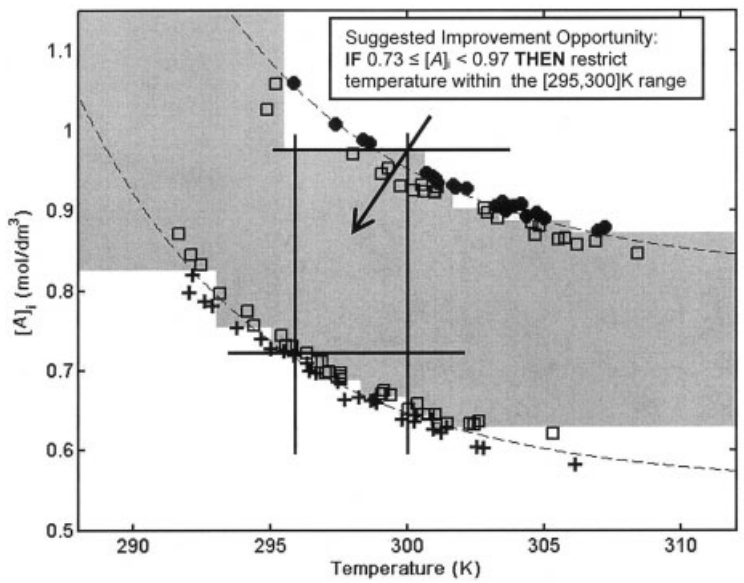

Figure 9. Induction and refinement of classification decision tree using support vectors.

Table 3. Decision List from Symbolic Induction Using Support Vectors Identified in Figure 9

$[A]_i$ (mol/dm^3)	T (K)	Class
>0.826	≤295.5	B
>0.631	>301.0	B
≤0.978	≤300.7	B
>0.978	≤295.5	B
≤0.929	≤303.4	B

An important consideration is that the support vector classifier update module and the outlier filtering module need interlinking because the action of one may negate the action of the other. We avoid filtering when the update module has been invoked *and* a process drift has been detected.

Figure 11 is a detailed summary of the algorithmic implementation of the online decision support system for process improvement opportunities. The symbolic induction module is similar to that proposed by Saraiva and Stephanopoulos[8] and thus not discussed.

Implementation of the online decision-support system for process improvement on a simulated CSTR system

The SVM-based decision-support system for process improvement was run on-line using the same CSTR system in which an irreversible first-order reaction is occurring. The quality variable was selected as the concentration of reactant B in the output stream, classified into any of three classes, "low," "normal," and "high" (or A, B, and C) based on the distribution of the quality variable as before. A Monte Carlo simulator was used to generate 23,000 points. As in Saraiva and Stephanopoulos,[8] the activation energy constant was changed from 99,770 to 101,430 J/mol after 10,000 time units to illustrate the relatively fast adaptive properties of the algorithm. The parameter change introduces an upward shift on the class separating boundaries in the input space, assuming the classification scheme remains unchanged.

Figures 12a–f are snapshots of the contents of the support vector set constituting the memory used in the induction of the decision trees. The first 40 points of the simulated data were used to initialize the memory base. A numerical increase in the number of support vectors is observed as more data become available. (However, as a fraction of the training data, a decrease actually occurs.) Remarkably, after only about 200 time units the number of support vectors appears to saturate around 80 data patterns. Thereafter, a drastic increase (more than double the saturation level) is observed around 10,000 time units and also an increase in the running error rate. Certainly, such an increase cannot be attributed to measurement errors only. At this point, frequent support vector classifier updates are observed. The decision-support system notes that reconfiguration is required and, accordingly, reinitializes the support vector memory by using patterns within the last 50 time units (or other time interval specified by a human expert). It is essential to ensure that all possible classes are defined in the memory base. Thus, in the absence of any one class within the

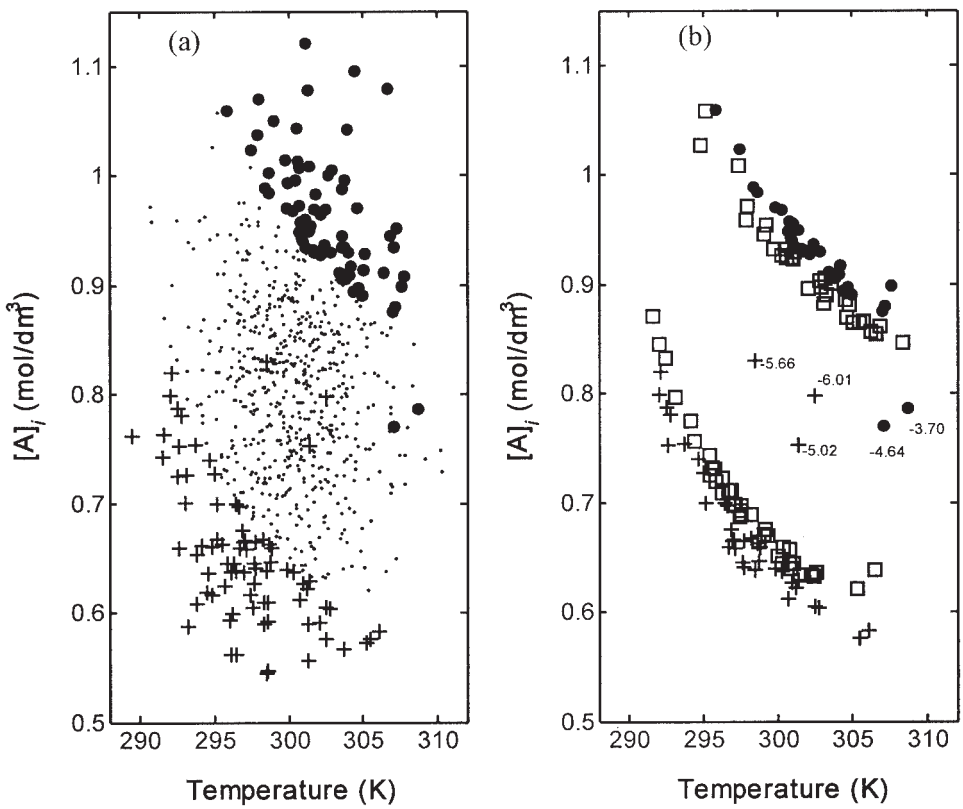

Figure 10. Detection of outliers in simulated CSTR data set using the values of slack variables after SV classification.

(a) The original data with classes A, B, and C, represented by ●, ·, and +, respectively. (b) The SVs where those with negative slack variable values (ξ_i) > 2 are designated as outliers.

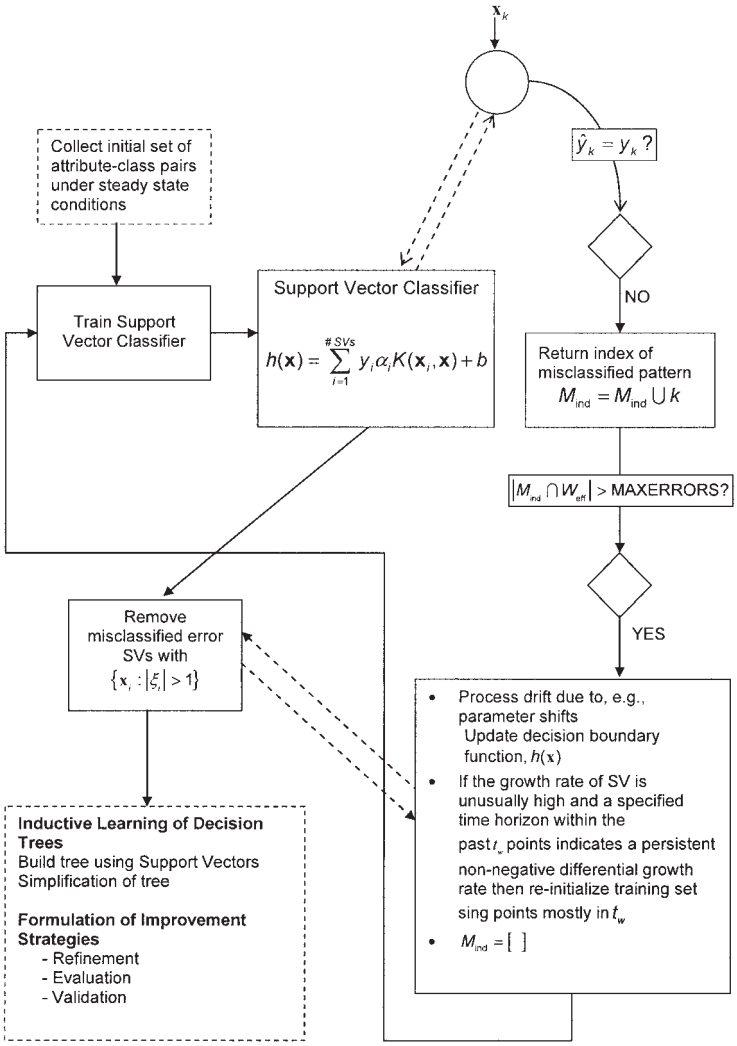

Figure 11. Online decision support system for process improvement.

specified interval, all or some support vectors of the missing class from the last decision boundary definition are retained.

This adaptive strategy ensures an immediate change in the nature of the process improvements suggested. For different parameter specifications, it was observed that the system establishes an accurate reflection of the state of the process within 20 time units. Although this is dependent on the system under analysis, it is substantially more rapid than the original approach used in a nearest neighbor with a Euclidean distance measure or DNN–based system.

We show the corresponding evolution of the active memory of exemplars in Figure 13. Unlike the SVC-based system, the DNN-based system has a long exemplar–memory saturation time. Moreover, although it is able to detect a change in the underlying process dynamics, a long time lag exists between process changes and system stabilization. This may have a negative impact on a production or process plant, resulting in a high rate of rejection of end products.

For the choice of parameters used the SVC implementation yields a slightly worse performance compared to the DNN approach (Figures 13h and 14h). It is also possible to obtain an SVC classifier with improved performance by specifying different model parameters; however, it must be noted that the SVC optimizes the trade-off between the complexity and performance of the model by imposing a maximum bound on the generalization error. Formally, the underlying statistical learning theoretical approach for SVMs seeks to optimize a different criterion (capacity), whereas the empirical risk minimization principle on which the DNN approach is based minimizes the error on a training set. Thus, theoretically the SVM approach is expected to have better generalization properties.

This has useful implications in our approach. Specifically, variations in input and output measurement errors can be established (such as from historical data or sensor specifications) and incorporated into the model parameters, that is, the regularization constant C and kernel parameter(s). As long as the misclassification error is within a certain range we are content that all expected variations are accounted for.

Control of Manganese in a Solution Preparation Circuit

Real data from industrial plants occasionally exhibit peculiarities that may require system-specific modifications to the

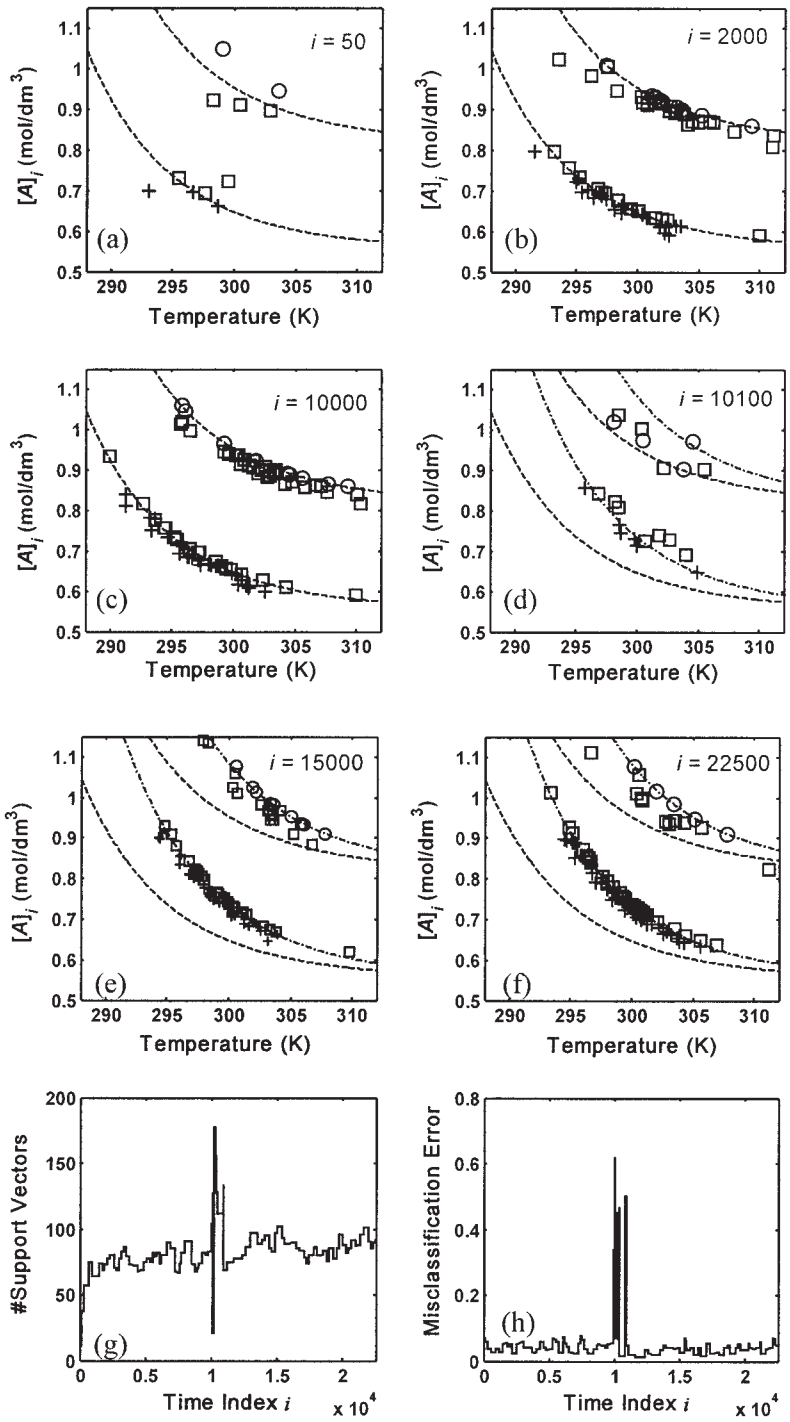

Figure 12. Snapshots of the online evolution of the support vectors before (a–c) and after (d–f) effecting a change in the activation energy at the indicated times.

The evolution of the number of support vectors is shown in (g) and the evolution of the misclassification error rate on a test set of the "next" 200 points in the data series is shown in (h).

direct implementation of the proposed method. For example, plant data are occasionally collected at sampling rates that do not allow the capture of high-frequency process dynamics or, in some cases, sampling may be subject to biased methods, sensors, and so forth. In this section we discuss the implementation of the proposed approach to real data collected from a manganese producing plant.

As shown in Figure 14, the manganese is produced in a sequence of stages including leaching, thickening, and electroplating. Calcined ore containing manganese is fed into a leaching circuit in which dissolution of the metallic species into solution is promoted by the addition of sulfuric acid. Because all metallic impurities have a higher electro-affinity than the manganese, it is important to purify the solution after leaching

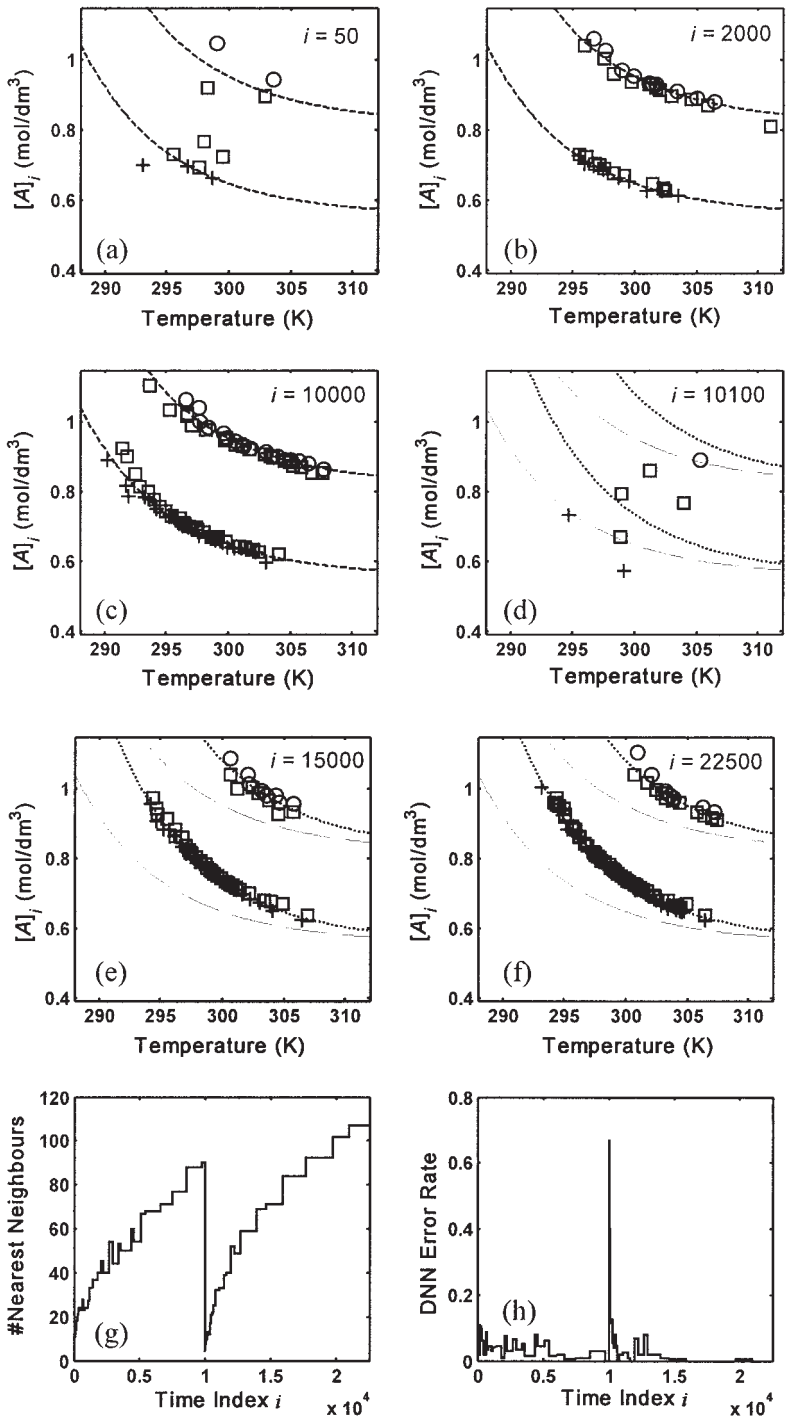

Figure 13. Snapshots of the online evolution of the active memory of exemplars in the DNN approach before (a–c) and after (d–f) effecting a change in the activation energy at the indicated times.

The evolution of the number of nearest neighbors is shown in (g) and the evolution of the misclassification error rate on a test set of the "next" 200 points in the data series is shown in (h).

before electroplating. Impurities are reduced to very low levels by the action of ammonium sulfide during thickening. Finally, the manganese is plated out in the cell house and the residual solution recycled back into the circuit.

A preliminary analysis of the data indicated significant correlation between species concentrations and cell efficiencies, suggesting that solution control in the leaching and thickening stages was critical to efficient plant operation. However, the same data indicated somewhat arbitrary dosage rates for the sulfuric acid, ammonium hydroxide, and alum. It was thus decidedly difficult to develop decision rules based on the species concentration. A possible solution would be develop-

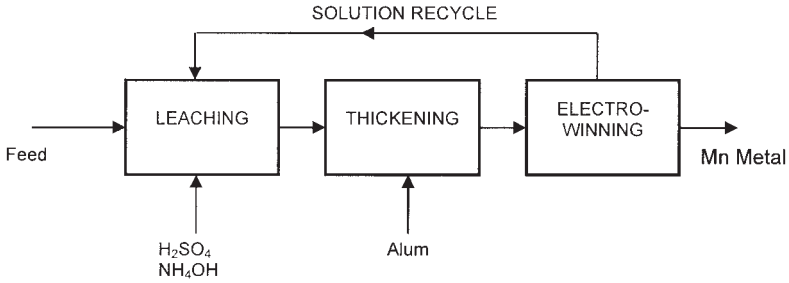

Figure 14. Manganese metal solution preparation plant.

ment of models for the prediction of the variation of species concentration in the cell house as a function of time-delayed concentrations of species concentrations upstream. Below we consider the variation of manganese concentration in the cell house as a function of manganese concentrations in the cell house and leach tanks a unit time earlier

$$[\mathrm{Mn}(t+1)]_{\mathrm{Cell}} = f([\mathrm{Mn}(t)_{\mathrm{Cell}}], [\mathrm{Mn}(t)]_{\mathrm{Leach}}) \qquad (16)$$

To search out and formulate possible improvement strategies, we identify $[\mathrm{Mn}(t+1)]_{\mathrm{Cell}}$ as the quality variable, as a function of $[\mathrm{Mn}(t)]_{\mathrm{Cell}}$ and $[\mathrm{Mn}(t)]_{\mathrm{Leach}}$. Three classes of the quality variable are specified according to the statistical distribution of the available plant data, as shown in Figure 15.

As can be seen, there seems to be a linear relationship between the two product variables. A region of desirable species concentration in solution for improved cell efficiencies can be discerned, although the separation is not as clear-cut. It was found that direct implementation of the proposed improvement framework, although instructive, did not give results suitable for delineating the problem space. This could be attributed to inadequate data for better decision boundary definition. Observing that the crucial data points are those on the boundaries of the desirable class, B in this case, we propose to use only those samples in defining a hyper-rectangular zone from which improvement opportunities can be formulated, as indicated in Figure 16. Integrating the suggested routes by which the evolution of the process and fundamental chemistry involved in the underlying reactions can be influenced, optimal dosage rates for various reagents can be estimated to improve control of the circuit.

Conclusions

In this paper we have introduced innovative modifications to an online methodology for process improvement opportunities previously proposed by Saraiva and Stephanopoulos.[8] The modifications substitute the central pattern-recognition module with one inspired by developments in statistical learning theory, that is, support vector classification. It was shown that use of support vector classification in defining the memory of exemplars provides for a number of advantages over the original strategy, including control of the number of data patterns in memory, effective outlier detection and filtering, rapid and flexible adaptive properties, and an ability to handle systems whose decision boundaries are appropriately defined by nonlinear functions.

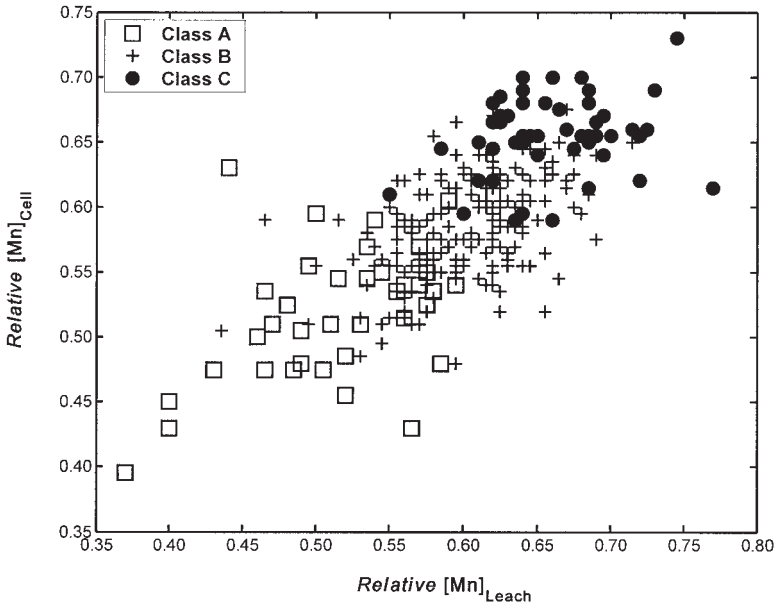

Figure 15. Problem formulation for manganese control.

Classes A, B, and C, respectively, represent low, normal, and high manganese concentration (scaled values) in the electrowinning circuit.

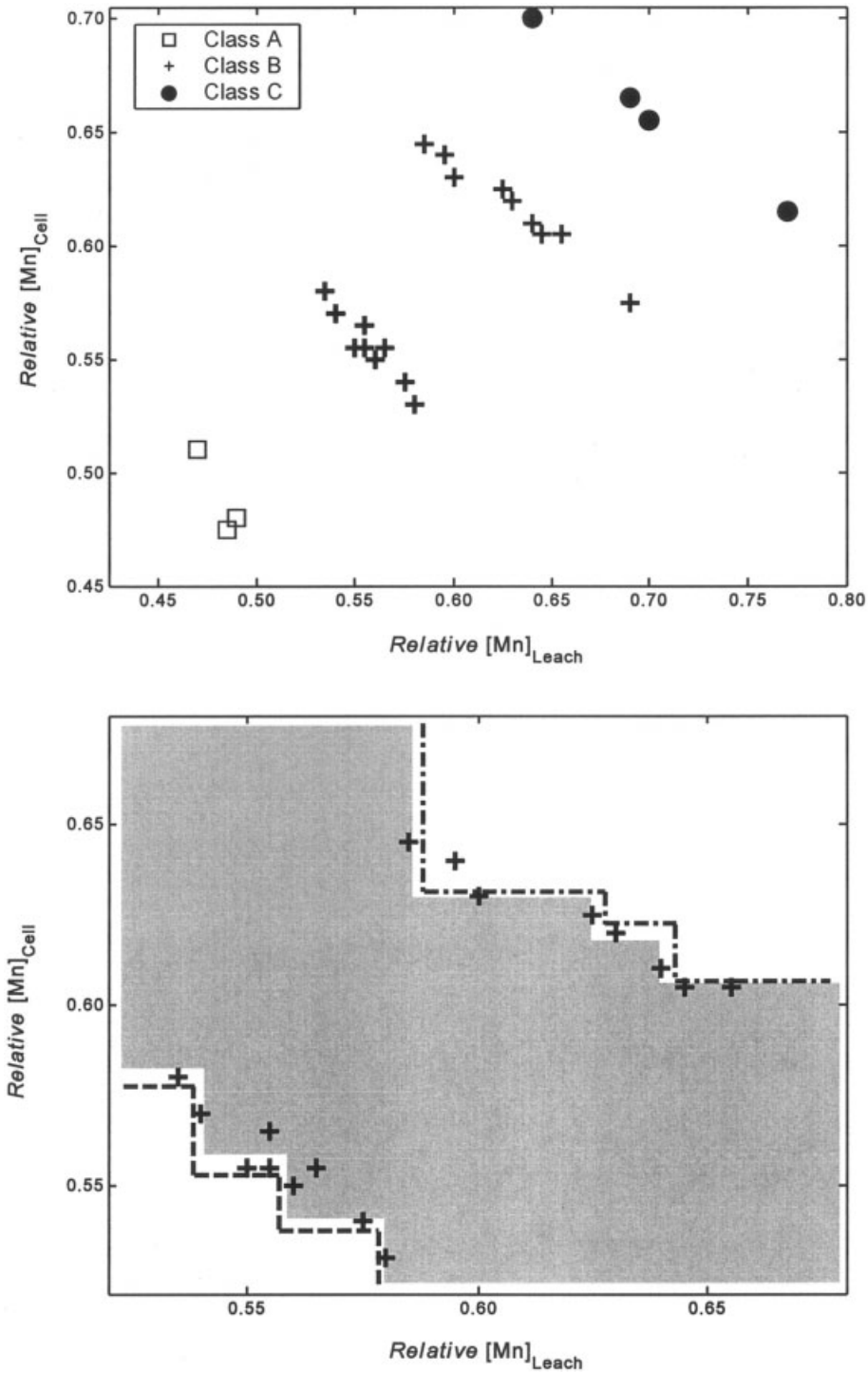

Figure 16. Formulating decision improvements for species control in a manganese solution preparation plant.

The plot on the left shows the support vectors generated by the SVC. The "+" markers denote the boundaries of the normal operating region, whereas the circles and squares denote the outer boundaries of classes A and C, respectively. The shaded area in the plot on the right shows the operating region supporting opportunities for process improvement.

Integration of support vector classification and a symbolic component (classification decision tree) provides an improvement in online management of product and/or process quality. A salient feature of support vector classifiers is their ability to capture pivotal relationships in a higher-dimensional feature space, which may not be possible in the input space. Thus, although we may measure correlated variables online, implicit mapping into a high-dimensional feature space unmasks these relationships.

Using a simulated CSTR system, various advantages of these modifications were illustrated. We compared the online performance of the SVC-based and the original DNN-based

systems in which the comparative strengths of the former were exposed. To illustrate the use of the methodology in practical systems, process improvement opportunities were formulated for an industrial manganese extraction plant, where useful approximations could still be formulated, despite sparse and unreliable plant data.

Acknowledgments

The authors thank Professor Saraiva for clarifying an issue on the first-order reaction used in the simulated CSTR example, and the anonymous reviewers for their helpful comments.

Literature Cited

1. Shewart WA. *Economic Control of Quality of Manufactured Product*. Princeton, NJ: Van Nostrand; 1931.
2. Woodward RH, Goldsmith PL. *Cumulative Sum Techniques*. London: Oliver & Boyd; 1964.
3. Hunter JS. Exponentially weighted moving average. *Journal of Quality Technology.* 1986;18:203-210.
4. Kourti T, Nomikos P, MacGregor JF. Analysis, monitoring and fault diagnosis of batch processes using multiblock and multiway PLS. *Journal of Process Control.* 1995;5:277-284.
5. Kresta JV, MacGregor JF, Marlin TE. Multivariate statistical monitoring of process operating performance. *Canadian Journal of Chemical Engineering.* 1991;69:35-47.
6. Raich A, Çinar A. Statistical process monitoring and disturbance diagnosis in multivariable continuous processes. *AIChE Journal.* 1996;42:995-1009.
7. Pearson RK. Exploring process data. *Journal of Process Control.* 2001;11:179-194.
8. Saraiva PM, Stephanopoulous G. Continuous process improvement through inductive and analogical learning. *AIChE Journal.* 1992;38: 161-183.
9. Fukunaga K. *Introduction to Statistical Pattern Recognition*. 2nd ed. San Diego, CA: Academic Press; 1990.
10. Jain AK, Duin PW, Mao J. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2000;22:4-37.
11. Quinlan JR. Induction of decision trees. *Machine Learning.* 1986;1: 81-106.
12. Quinlan JR. Decision trees and decisionmaking. *IEEE Transactions on Systems, Man, and Cybernetics*. 1990;20:339-346.
13. Breiman L, Friedman JH, Olshen RA, Stone CJ. *Classification and Regression Trees*. New York, NY: Chapman & Hall; 1993.
14. Utgoff PE. Incremental induction of decision trees. *Machine Learning.* 1989;4:161-186.
15. Bakshi BR, Stephanopoulos G. Representation of process trends—IV. Induction of real-time patterns from operating data for diagnosis and supervisory control. *Computers & Chemical Engineering.* 1994;18: 303-332.
16. Vapnik VN. *The Nature of Statistical Learning*. New York, NY: Springer-Verlag; 1995.
17. Vapnik VN. *Statistical Learning Theory*. New York, NY: Wiley; 1998.
18. Burges CJC. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery.* 1998;2:121-167.
19. Cristianini N, Shawe-Taylor J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge, UK: Cambridge University Press; 2000.
20. Schölkopf B, Smola AJ. *Learning with Kernels*. Cambridge, MA: MIT Press; 2002.
21. Gunn SR, Brown M, Bossely KM. Network performance assessment for neurofuzzy data modelling. In: Liu X, Cohen P, Berthold M, eds. *Proceedings of Lecture Notes in Computer Science: Intelligent Data Analysis*, Vol. 1208; 1997:313-323.
22. Fletcher R. *Practical Methods of Optimization*. New York, NY: Wiley; 1989.
23. Müller KR, Mika S, Rätsch G, Tsuda K, Schölkopf B. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks.* 2001;12:181-202.
24. Haykin S. *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, NJ: Prentice Hall; 1999.
25. Cawley GC. MATLAB support vector machine toolbox (v0.54), 2000. Software available at http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox. (Last accessed 28 March 2004.)
26. Cortes C, Vapnik V. Support vector networks. *Machine Learning.* 1995;20:273-297.
27. Platt JC, Cristianini N, Shawe-Taylor J. Large margin DAGs for multiclass classification. In: Solla SA, Leen TK, Müller K-R, eds. *Advances in Neural Information Processing Systems 12*. Cambridge, MA: MIT Press; 2000:547-553.
28. Hsu C-W, Lin C-J. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks.* 2002;13: 415-425.

Manuscript received Jan. 21, 2004, and revision received May 28, 2004.