ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/6956962

Support Vector Machines for Predictive Modeling in Heterogeneous Catalysis: A Comprehensive Introduction and Overfitting Investigation Based on Two Real Applications

ARTICLE *in* JOURNAL OF COMBINATORIAL CHEMISTRY · JULY 2006

Impact Factor: 4.93 · DOI: 10.1021/cc050093m · Source: PubMed

CITATIONS
38

READS
21

4 AUTHORS, INCLUDING:

Laurent A. Baumes
ExxonMobil
62 PUBLICATIONS **895** CITATIONS
SEE PROFILE

Jose Manuel Serra
Universitat Politècnica de València
154 PUBLICATIONS **2,279** CITATIONS
SEE PROFILE

Available from: Laurent A. Baumes
Retrieved on: 20 January 2016

Support Vector Machines for Predictive Modeling in Heterogeneous Catalysis: A Comprehensive Introduction and Overfitting Investigation Based on Two Real Applications

L. A. Baumes, J. M. Serra, P. Serna, and A. Corma*

Instituto de Tecnología Química (UPV-CSIC), av. Naranjos s/n, 46022 Valencia, Spain

Received July 18, 2005

This works provides an introduction to support vector machines (SVMs) for predictive modeling in heterogeneous catalysis, describing step by step the methodology with a highlighting of the points which make such technique an attractive approach. We first investigate linear SVMs, working in detail through a simple example based on experimental data derived from a study aiming at optimizing olefin epoxidation catalysts applying high-throughput experimentation. This case study has been chosen to underline SVM features in a visual manner because of the few catalytic variables investigated. It is shown how SVMs transform original data into another representation space of higher dimensionality. The concepts of Vapnik−Chervonenkis dimension and structural risk minimization are introduced. The SVM methodology is evaluated with a second catalytic application, that is, light paraffin isomerization. Finally, we discuss why SVMs is a strategic method, as compared to other machine learning techniques, such as neural networks or induction trees, and why emphasis is put on the problem of overfitting.

Introduction

Discovery of heterogeneous catalysts for a specific reaction encompasses understanding the different reaction mechanism steps. Especially important is the understanding of the activation/coordination of the reactants over the different active sites driving to specific catalyst activities and selective reaction pathways.[1] Nevertheless, catalyst discovery is mostly an empirical process which applies fundamental knowledge (first principles) for defining/designing the experimental multiparametrical space to be explored/mapped in the search for materials with the desired catalytic properties. Typical catalyst parameters include elemental composition, preparation conditions, and synthetic routes. The application of high-throughput (HT) or combinatorial techniques in the field of materials science and catalysis[2] increases exponentially the number of samples to be processed and, therefore, the number of catalyst variables to be simultaneously explored. As a result, the possibility of finding new catalysts in a shorter time may be strongly increased. The need for refined and automatic search strategies aimed at minimizing the number of samples during the space exploration has enabled statistical approaches and intelligent computation to become a key part in catalyst discovery and optimization.

Different machine learning techniques have been successfully applied for modeling HT experimental data obtained during the exploration of multicomponent heterogeneous catalysts. This kind of models is called structure−activity relationships (SAR), and their implementation in HT experimentation has succeeded as shown in ref 3, in which the statistical practice in this approach is thoroughly reviewed. This type of model allows predicting the catalytic performance of untested materials, taking into account their composition or preparation conditions as input variables. Moreover, these predictions can subsequently be applied to the design of the next catalyst libraries in such a way that only the most promising catalysts are selected (so-called virtual or in silico screening) and included in the final library to be experimentally synthesized and tested for a specific catalytic reaction. Such an approach qualified as a "filter" by the authors has been employed with NN.[4,5] The modeling approaches may also be based on equivalence class search as first shown in ref 6 and then adopted for new studies.[7] Among the different machine learning techniques, neural networks (NNs) have often yielded the best modeling results. NNs have been applied for modeling and prediction of the catalytic performance of libraries of materials for diverse reactions, such as oxidative dehydrogenation of ethane,[8] water gas shift,[9] and methanol synthesis.[10] Nevertheless, NNs may suffer in some cases from overfitting of the training data, reproducibility problems, and lack of information regarding the classification produced.[11,12] There is still, therefore, the need to develop more robust and highly accurate modeling techniques in heterogeneous catalysis. On the other hand, numerous classical direct optimization methods[13] or new ones[14] have been employed. Very recently, a new biased statistical methodology has been proposed,[15] differing significantly from pure modeling or optimization approaches as it aims at selecting "interesting", that is, strategic, future samples to make better use of a modeling technique in a later stage. The recognition of the search space is shown to be tremendously increased.

Here, we focus on modeling as a decision help for a classification problem, that is, the automatic discrimination of good/bad experiments to be conducted. SVMs, which have exhibited outstanding performances in handwritten digit recognition, object recognition, speaker identification, face detection in images, and text categorization, can be a suitable

10.1021/cc050093m CCC: $33.50 © 2006 American Chemical Society
Published on Web 05/13/2006

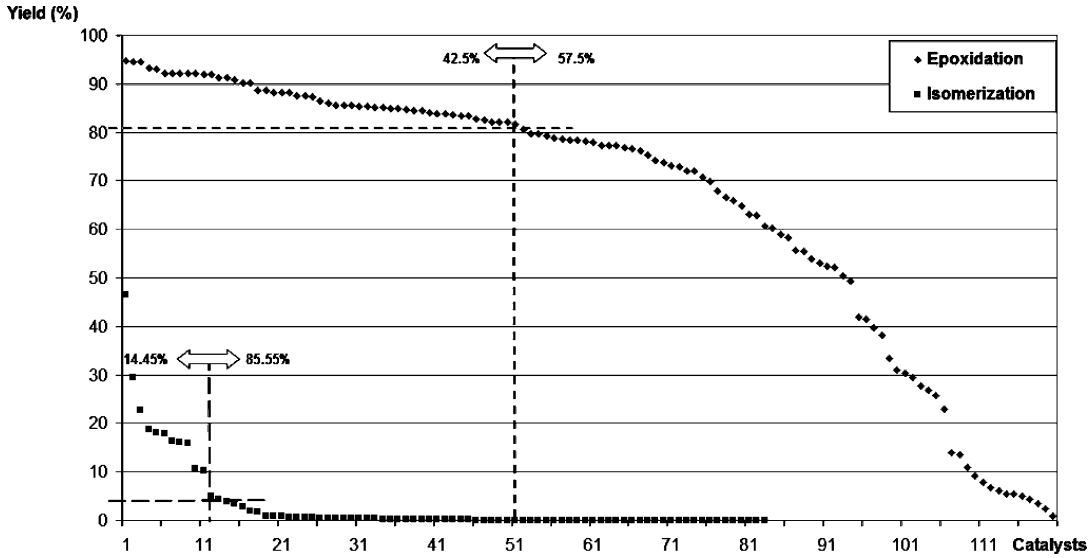


Figure 1. Thresholds used for discretization of the two datasets into a two-class problem taking into account the selected fitness output (specific field or conversion).

method to overcome some of the problems such as overfitting, parameter settings, etc. Although the development of SVMs started in the late 1970s,[16] it has recently received more attention. SVM techniques have been extensively applied in pattern recognition problems[17] and QSPR modeling in drug discovery and medicinal chemistry,[18] process engineering,[19] and physical chemistry,[20] for example, for the prediction of the heat capacity of organic molecules. However, to the best of our knowledge, SVMs have not been applied to the prediction of catalytic performance of heterogeneous catalysts, despite the fact that this methodology may be very helpful, especially when combined with HT techniques at early stages. Here, performances are discretized, and models should be used for selecting/discarding future experiments to be synthesized and tested.

The purpose of this work is, first, to present such a relatively new and unknown methodology to the material science domain, then to provide a comprehensive introduction of the points that differ from other well-known and commonly employed techniques, such as the representation of the data, the selection mode of the optimal model, and the use of the basic ideas behind SVMs for HT heterogeneous catalyst discovery. The SVM methodology is detailed, step-by-step, and based on experimental data derived from a HT study aiming at optimizing olefin epoxidation catalysts. We review a wide range of SVM techniques, including the dual characteristic representation of linear machine learning, feature spaces, learning theory, and generalization theory that are necessary for a comprehensive introduction to SVM. However, the convex optimization used internally is not described due to its popularity and in an effort not to introduce additional difficulty, which can be avoided without losing comprehension. We discuss the theoretical aspects of SVMs, which make them an attractive method for catalytic modeling, as compared to other learning strategies, such as NN or induction graphs. The SVM methodology is evaluated using experimental data derived from exploration/optimization of heterogeneous catalysts in two different industrial fields: oil refining and petrochemistry, the selected reactions for each application being *n*-alkane isomerization in gasoline and olefin epoxidation. We have finally placed special emphasis on the problem of overfitting, and a comparison of generalization is done with classification trees (CTs).

Experimental Datasets

Figure 1 shows the quality distribution of the two experimental datasets and the selected thresholds for discretization of the data, that is, for classifying them into good (A) or bad (B) catalyst classes, or alternatively, active (A) or inactive (B) materials. Since the final application of the SVM models is support of the decision made during the experimental design; that is, helping during the selection of the experiments to be conducted, it appears logical that a "yes" or "no" answer makes the choice straightforward. Although all the SVMs described later are binary classifiers, they can be easily combined to handle the multiclass case. A simple and effective way is to train N one-versus-rest classifiers for the N-class case. Then the class for a test point is defined by the majority or the largest positive distance.[21] It has to be noted that achieving the SVM implementation after reading such a paper is not an objective. All experiments are based on calculations with the free source code called SVMlight.[22] We think the integration of programming code, or pseudocode, would have made the paper blurry without being of additional help for the chemist.

Olefin Epoxidation. The first experimental dataset is derived from a previous work aimed at the optimization of epoxidation catalysts based on titanium silicate mesoporous materials.[23,5] In that work, the experimental design was ruled by a hybrid optimizer comprising an evolutionary algorithm assisted by a NN, and experimentation was accomplished using HT techniques. Epoxidation of cyclohexene with *tert*-butyl hydroperoxide at 60 °C and atmospheric pressure was used as the probe reaction for the optimization procedure. The objective function was the yield of cyclohexene epoxide, whereas the variables to be optimized were the molar concentrations of some of the components of the starting gel, that is OH^-, titanium, tetramethylammonium (TMA),

Table 1. First Subset of Data: Epoxidation Catalysts Described Using Just Two Synthesis Variables, and Additional Data for Olefin Epoxidation Study

x_i	CTMA/(Ti + Si)	OH/(Ti + Si)	class
	(a) First Subset of Data: Epoxidation Catalysts Described Using Just Two Synthesis Variables		
1	0.3424	0.1615	1
2	0.3678	0.2173	1
3	0.3807	0.2265	1
4	0.3689	0.1558	1
5	0.2674	0.2075	−1
6	0.2515	0.1833	−1
7	0.3082	0.1993	−1
8	0.2889	0.2162	−1
9	0.2800	0.1680	−1
	(b) Additional Data for Olefin Epoxidation Study		
10	0.2259	0.2014	1
11	0.2087	0.1764	1
12	0.2214	0.1508	1
13	0.1873	0.1658	1
14	0.1920	0.2080	1
15	0.1871	0.1479	1
16	0.2439	0.1465	1
17	0.4011	0.2450	1
18	0.1485	0.1450	−1
19	0.4004	0.2567	−1
20	0.2569	0.2394	−1
21	0.4180	0.2060	−1
22	0.4160	0.1800	−1
23	0.2862	0.2472	−1
24	0.3642	0.2538	−1
25	0.1510	0.1641	−1
26	0.1439	0.1488	−1

and cetyltrimethylammonium (CTMA). Here, we first use a subset (Table 1a) of the whole dataset for the introduction of linear classifiers. Table 1a is composed of catalysts in which Ti and TMA ratios are bounded into very little ranges (i.e., nearly constant), and a threshold value of 81% of epoxide yield is used for creating two classes. This value is related to the level of activity, as compared with a reference catalyst, in such a way that class A represents catalysts more active than the reference, whereas class B does not reach this level of activity. Then we consider an increased subset of data composed of more cases (Table 1b) for the nonseparable case with the same separation value. Finally, we consider the full available set of both data and variables (see Figure 1 and Supporting Information). A discretization is done at 87.6% considering the final epoxide yield for defining the two classes.

Isomerization. Isomerization of linear paraffins to produce their branched isomers is one of the possible methods for increasing the octane number in gasoline. The isomerization reaction scheme involves the sequential isomerization of the paraffin into the mono-, di-, and tribranched isomers; however, cracking side reactions occur, reducing the isomerization yield. *n*-Alkane isomerization is a reaction catalyzed by bifunctional catalysts consisting of an acid support impregnated with Pt (acid + hydrogenating metal). Industrially practiced catalysts are, for example, Pt-chlorinated alumina and Pt-mordenite.[24]

The second dataset was taken from a previous work[25] aimed at the discovery bifunctional oxide-based isomerization catalysts. The probe reaction was the isomerization of *n*-pentane under 30 bar of H_2 at temperatures ranging from 200 to 240 °C. The general catalyst formulation includes an oxide support, an acidity enhancer, a metallic promoter, and 0.5 wt % Pt. Catalyst optimization was ruled by a conventional evolutionary algorithm, and three generations of 24 catalysts were run. Objective function was the isopentane yield at 200 °C. Due to the difficulty of this reaction, the creation of the classes A and B has been done considering a threshold value of 5%, since the aim is just to distinguish catalytic and noncatalytic materials.

Step-by-Step SVM Computation

SVMs belong to learning strategies called machine learning. When computers are applied to solve complex problems, situations can arise in which there is no known method for computing the desired output from inputs. An alternative strategy, known as *supervised* learning strategy, attempts to learn the input/output functionality from examples. A learning problem with a finite number of outputs, expressed as categories, is referred as *multiclass classification*, whereas for real-valued outputs, the problem becomes known as *regression*.

A main difference between machine learning methods and classical statistics is that the former does not assume any previous parametric form of the appropriate model to use. Most machine learning techniques may, thus, be classified in the set of *distribution-free* methods, a term used to highlight an a priori independence between the data set and a predefined set of models. Nevertheless, machine learning approaches do not exclude taking advantage of parametric assumptions, if they hold. Thus, instead of starting with assumptions about a problem, machine learning uses a toolbox approach to identify the correct model structure directly from the available data. One of the main consequences is that the machine learning methods typically require larger data sets than parametric statistics. A comparison of statistical models, SVM, and neural networks is done in ref 12. The available dataset is usually split into two parts: the training sample, used to select the model; and test samples, used to evaluate its accuracy (also called capacity or generalization). Most methods are "asymptotically consistent"; i.e., they would produce the really best model, provided that an infinite amount of training data was available. Thus, the focus of the machine learning approach is to optimize performances for finite sample sizes. One should ask what is the quantity that is necessary for such a methodology. One possible solution may be to test the different strategies investigated with numerous benchmarks, that is, virtual data usually in the form of mathematical functions. Such an approach permits one to better understand the algorithm functioning and also to set some of the internal parameters. However, the parameters' setting is independent of the benchmark, and a way to characterize the mostly unknown real search space for comparison with the benchmarks remains to be found. Thus, a criterion of great appeal would be the calculation of a value describing the complexity, for optimization algorithms, of a given benchmark or the "learnability" considering machine learning,[26] thus allowing one later to obtain the criterion value or range for a

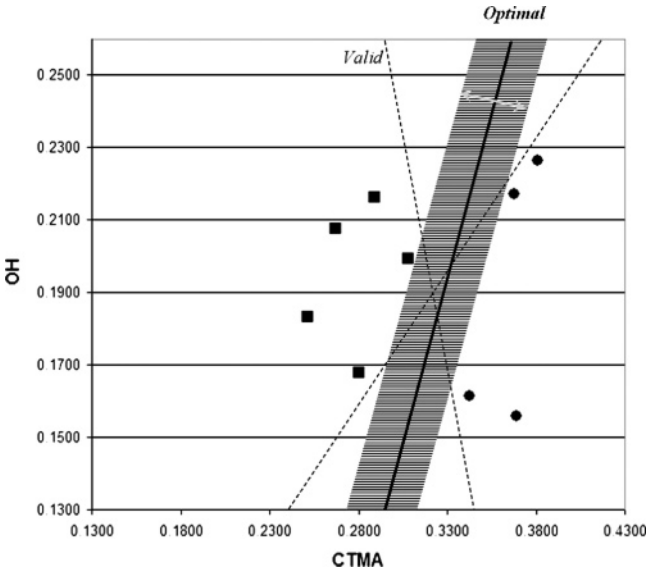


Figure 2. A simple separable case, margin, and optimal hyperplane using the examples of Table 1. Squares represent low-performance catalysts (class B coded −1 for SVMs), and the circles stand for high-performance catalysts (class A coded +1).

real study on the basis of a comparison of performances. However, the suitable sample size required for different applications strongly depends on the complexity of the problem (i.e., the type of relationships between input/output variables) and the number of input variables. Thus, the minimum sample size remains nearly impossible to determine. Moreover, both the distribution (i.e., sampling) of each catalytic experiment (i.e., input values) in the entire search space, and the distribution of the observed outputs influence the performances of the algorithms.[15] Finally, it is important to emphasize that the decision about which input variables must be considered to solve a scientific problem is directly related to the success of the investigation, and thus, the discussion about the complexity of the search space is actually distorted by the fact that the mentioned space has been partially defined by the researcher.[27]

The Linearly Separable Case. Linear classification is first presented, then the dual representation of linear machine learning is introduced. However, this limited computational power leads to the topic of "kernel-induced feature spaces".

The training sample contains l catalysts, called examples, and noted $x_i \in \chi$, where χ[28] represents the whole search space. When each example has to be classified, in agreement with a predefined criterion, a classifier algorithm, that is, SVM, must be able to propose a general way of doing it. It is first considered a *linearly separable* example, that is, the case that two classes can be perfectly discriminated with a hyperplane [w][x] + [b] (Figure 2). In the rest of the text, vectorial or matrix notations are removed. In this case, the mathematical constraints permitting us to verify that each example is assigned correctly to its class are given by eq 1. The type of response required by the algorithm, and permitting the formulation of eq 1, must have the form $y_i = \pm 1$, that is, $y_i \in \{-1, 1\}$. For multiclass problems, the SVM methodology can also be applied by means of merging classes and applying iteratively the two-class strategy. Considering the example drawn in Figure 2 and one particular hyperplane, new examples will be associated with a given class, depending on the estimated y_i (noted $\tilde{y}_i$) value, more precisely, the sign of $\tilde{y}_i$. The main problem deals with the fact that many different hyperplanes can be chosen. It sounds logical that the classifier with the greatest generalization is the one which goes the furthest from all examples while respecting eq 1. Thus, an interesting criterion is to maximize the so-called margin (≡ in Figure 2), that is, twice the distance between the nearest points and the hyperplane. In agreement with eq 1, the margin is equivalent to $2/\|w\|$ in such a way that, if one wants the margin to be maximized, the norm[29] of w ($\|w\|$) must be minimized. Taking into account that a 3D plane is completely fixed (i.e., blocked) by means of a minimum of three points, the margin is just related to such examples that define the plane. These special points are called *support vectors*, and consequently, it will not be necessary to consider the rest of the examples for defining the class of new unknown samples. This is of great appeal because SVM will be preferred to other methodologies when considering a huge amount of data (e.g., even millions of cases) to be treated.

$$w_i x_i + b \geq 1 \quad \text{if } y_i = 1 \text{ and } w_i x_i + b \leq 1$$

$$\text{if } y_i = -1 \Rightarrow y_i(x_i w + b) - 1 \geq 0, \quad \forall i \qquad (1)$$

Not Linearly Separable Case. Real-world applications often require a more expressive hypothesis space than linear functions can provide. When considering the epoxidation catalysts (Table 1b), Figure 3 shows that any linear separation allows one to discriminate perfectly the two classes. For constructing nonlinear decision functions, nonlinear mappings can be used to transform the original data, called "input space", into another high-dimensional feature space in which a linear learning methodology can then be applied (Figure 3, top left to top right). SVM methodology differs from other common strategies on different points; the construction of a separating hyperplane in the "feature space", which leads to a nonlinear decision surface in the original space (Figure 3, bottom right), is the first distinction. Therefore, one still looks for linear separations but into a higher dimensional space. SVM can use a different family of functions, Φ.[30] Increasing the number of dimensions is useful to classify nonseparable data, but it also increases the risk of overfitting the data. For example, considering the dataset shown in Table 1a,b, other transformations could have been applied, such as those depicted in Figure 4. Therefore, one should start with the simplest (or simple) functions (i.e., Φ is restricted to a given family of functions; for example, linear or polynomial) which allows one to minimize the overfitting problem.

Another issue, usually more frequent in nonlinear problems, is the fact that a reasonable level of error/level of generality ratio must be chosen, and thus, it is advisable to assume a certain level of error to decrease the risk of overfitting. In addition to keeping the $\|w\|$ minimization criteria, other variables, called slack variables or "slacks", noted $\xi_i \geq 0$, are defined as a way of penalizing examples that are on the wrong side of the hyperplane (i.e., misclassified). The performance of the classifier can be improved by controlling the importance of both the classifier capacity

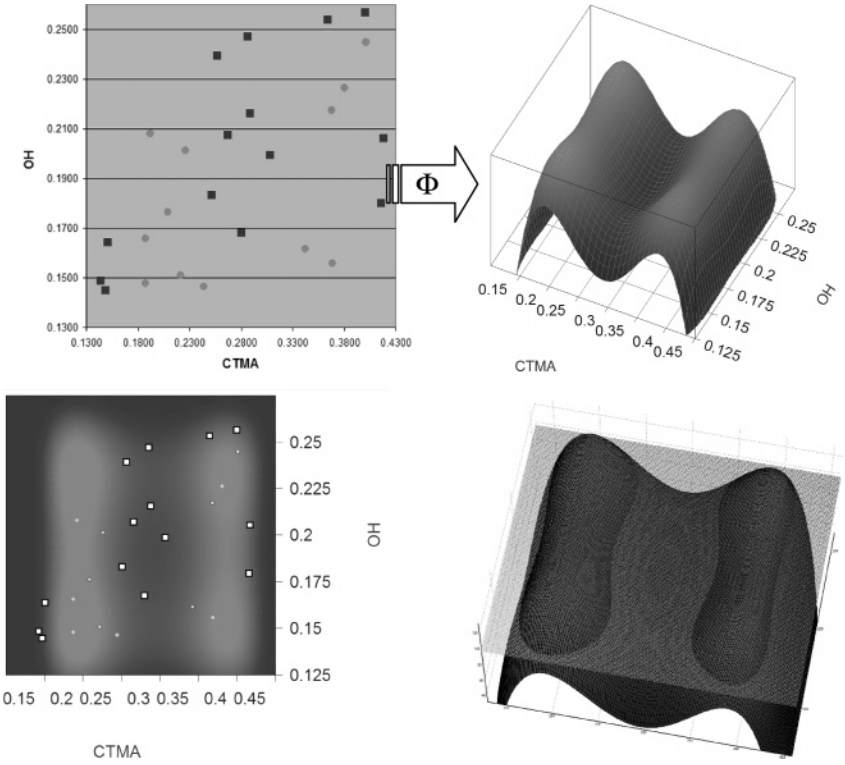
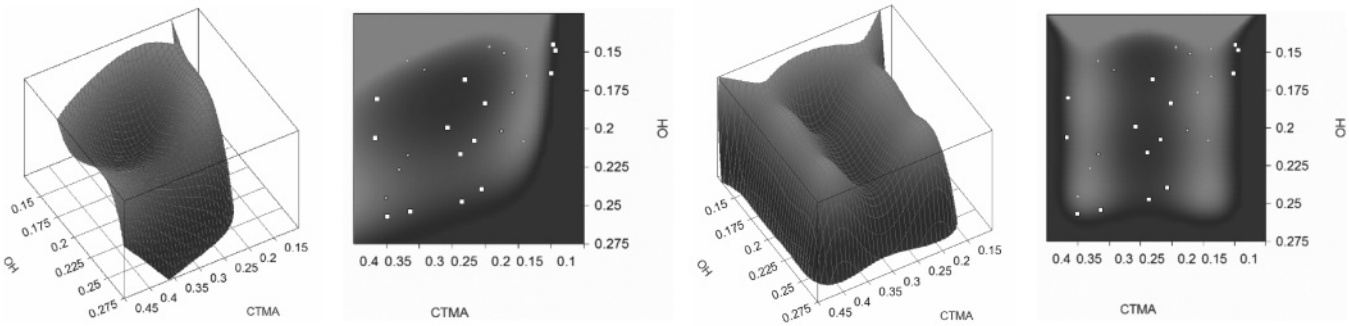


Figure 3. Nonlinear mapping from original 2D space to 3D (dataset of Table 2). $\Phi(x, y) = z = a + bx + cx^2 + dx^3 + ex^4 + fx^5 + gy + hy^2 + iy^3 + jy^4$; $a = -6188.2$, $b = 26\,385.3$, $c = -108\,205.1$, $d = 64\,961.9$, $e = 448\,045.1$, $f = -674\,392.5$, $g = 94\,525.8$, $h = -760\,344.4$, $i = 2\,679\,148.6$, $j = -3\,491\,334.4$.



$$z = a + bx + cx^2 + dx^3 + ex^4 + fx^5 + gy + hy^2 + iy^3 + jy^4 + ky^5$$

$a = 45192.37$ $b = 30599.681$ $c = -136552.3$

$d = 151568.15$ $e = 331645.79$ $f = -623552.19$

$g = -1236282.6$ $h = 12822990$ $i = -65936199$

$j = 1.6809924e+08$ $k = -1.700122e+08$

$$z = a + \frac{b}{x} + cy + \frac{d}{x^2} + ey^2 + \frac{fy}{x} + \frac{g}{x^3} + hy^3 + \frac{iy^2}{x} + \frac{jy}{x^2}$$

$a = 7285.7082$ $b = -1728.5642$ $c = -78255.82$

$d = 212.24972$ $e = 311546.68$ $f = 9272.1273$

$g = -11.181506$ $h = -417444.25$ $i = -16777.998$ $j = -362.67572$

Figure 4. Nonlinear mapping from the same dataset using more complex functions.

(via $||w||$) and the sum of the slacks $C \sum \xi_i$ (Figure 5). C is a parameter to be tuned by the user permitting a tradeoff between classification rate and capacity.

This leads to kernel-induced feature spaces. Kernel functions can implicitly combine the two steps (nonlinear mapping and linear learning) into one step in constructing a nonlinear machine learning.

Kernel Functions. Considering linear functions in a variable vector, given a convex space, we obtain a convex program that can exploit the methods of convex optimization:[31] To find the optimal hyperplane, $\Psi = \{||w||^2\}/\{2\}$ must be minimized, with eq 1 as constraint, which gives the Lagrangian formulation: $L(w, b, \alpha) = \Psi - \sum \{\alpha_i[f(x, w, b) - 1]\}$. The explanation of such a technique is not compulsory for such a quick overview of SVM. Readers are referred to ref 32 for constrained optimization method details. One has simply to be reminded that the search space's being convex allows finding an optimal solution using such optimization tools. The Lagrangian solution is $f(x) = \sum \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b$.

With such a Lagrangian formulation, only the training data (i.e., $\mathbf{x}$) appears in the form of dot products[33] between vectors. This key property allows generalizing SVMs to the most general case. If we make use of kernel functions, it is permissible to replace $(\mathbf{x}_i \cdot \mathbf{x}_j)$ with $k(\mathbf{x}_i \cdot \mathbf{x}_j)$ into the Lagrangian solution. Meanwhile, all the previous considerations hold, since we are doing a linear separation, but in a different space, $\mathcal{F}$. Φ: $\chi \xrightarrow{\chi \to \mathcal{F}} \mathbf{x}$, k: $\chi \times \chi \to$ R. With $k(x_i, x_j) = \mathbf{x}_j \cdot \mathbf{x}_i = \Phi(x_i) \cdot \Phi(x_j)$, Φ does not need to be explicitly known. For

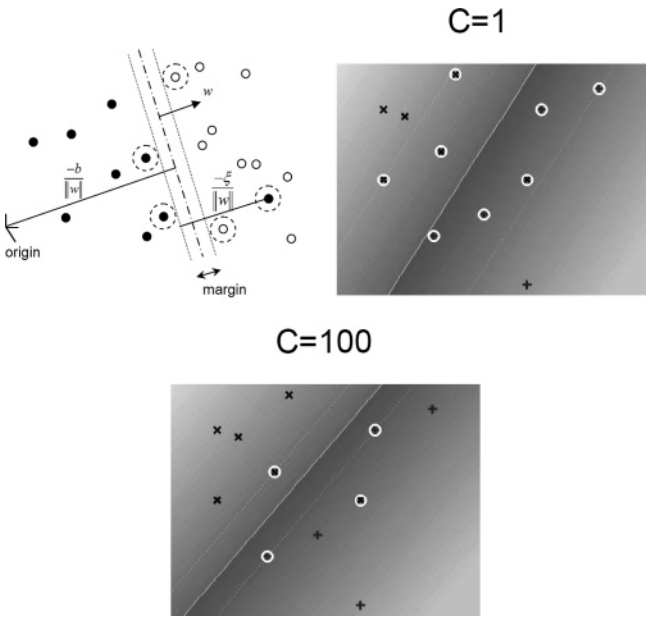


Figure 5. Linear separating hyperplane in a nonseparable case and *C* influence.

example, if $x_i = (x_{i_1}, x_{i_2})$, $\Phi(x_i) = (x_{i_1}^2, \sqrt{2}x_{i_1}x_{i_2}, x_{i_2}^2)$, we obtain[34] $\Phi(x_i)\cdot\Phi(x_j) = (x_i \cdot x_j)^2$.

$$\text{sgn}(f(x)) \text{ with } f(x) = \sum_{i=1}^{N_S} \alpha_i y_i \Phi(s_i)\cdot\Phi(\mathbf{x}) + b = \sum_{i=1}^{N_S} \alpha_i y_i k(s_i, \mathbf{x}) + b \quad (2)$$

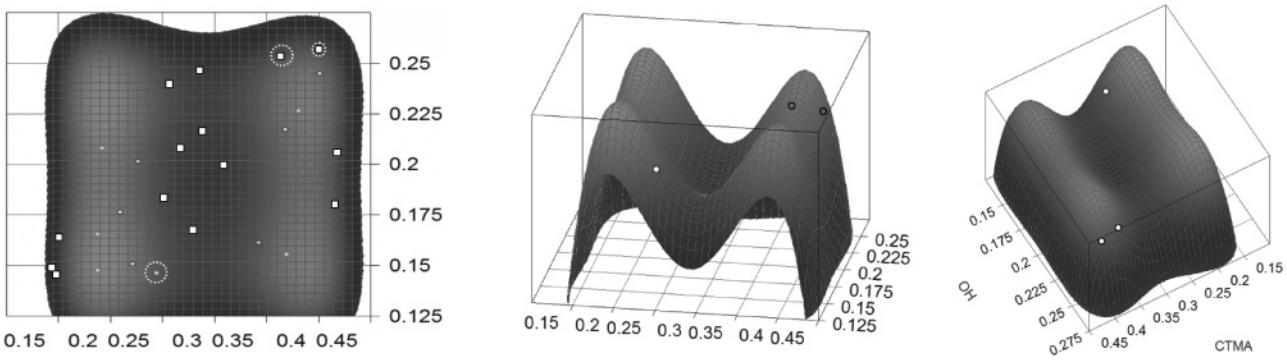
After training, $w \in \not\supset$ is obtained, and during the test phase, the SVM is used by computing the sign of dot products of a given test point x_i, as shown in eq 2. This equation presents the use of trained SVM for a new example "*x*", where *f* is the discriminative function and s_i are the support vectors. Figure 6 shows the support vectors from calculations on the merge dataset from Table 1a,b. Catalysts corresponding to the given support vectors are x_1, x_{19}, and x_{25} in Table 1a,b. The separation rule is given by the indicator function using the dot product between the patterns to be classified (**x**), the support vectors, and a constant threshold, *b*. Note that in eq 2. only the SV are used in the sums. The usual kernels $k(x_i, x_j)$are polynomials $(\gamma x_i^T x_j + r)^d$, $\gamma > 0$; RBF $\exp(-\gamma|x_i - x_j|^2)$, $\gamma > 0$; or sigmoid tanh $(\gamma x_i^T x_j + r)$ with γ, *r*, *d* as the few parameters to be set by the user.

The Theory behind SVM: VC Dimension and SRM. The exploration and formalization of generalization concepts has resulted in one of the shining peaks of the theory of statistical learning on which SVMs are based. The problem related to the estimation of *f* can be achieved with numerous methods.[35] We have only considered here the approach proposed by Vapnik,[30] which is based on obtaining bounds on the risk, *R*. To do so and, thus, to lead to good generalization, SVMs employ the Vapnik and Chervonenkis (VC) dimension[36] (noted *h*), which characterizes Φ and is used to obtain the bounds on *R*, that is, for minimizing the risk of overfitting.

To introduce the VC dimension, let us take an easy example. Thus, considering the two-class problem, 2^l represents the total number of possibilities for *l* examples to be labeled. If for each possibility there exists at least one function belonging to Φ that correctly associates the labels to every example, the whole set of points is said to be "shattered" by Φ (Figure 7a). The VC dimension of Φ, noted *h*, is defined as the maximum number of points that can be shattered by Φ. For a complete presentation, see ref 37. Here, the reader must understand that the VC dimension characterizes the function that will permit one to discriminate between classes. Such criterion indicates, considering all possible cases of labeling, if the function considered can do such a work. With this example, it is easy to understand why linear functions are relatively weak.

Therefore, the learning phase consists of determining the right function that minimizes *R*, considering the whole set of functions in Φ, the so-called risk, *R*. As said earlier, since *P* is unknown, the risk is also unknown; however, the empirical risk (R_{emp}) can be measured on a particular sample. *h* permits one to link the real risk, *R*, to R_{emp}. The principle of risk minimization (SRM) suggests a principled way of choosing the optimal class of functions by choosing the one that minimizes the risk bound over the entire structure.

Taking into account the probability $(1 - \eta)$, eq 3 is true. ϕ is called VC confidence. Φ, which minimizes the right-hand side, gives the lowest upper bound. The VC confidence is a monotonic increasing function of *h*, whatever *l*. It can can be observed that (i) it is independent from $P(x, y)$; (ii)



$$[\alpha] = [\alpha_1 \ \cdots \ \alpha_l] = \left[\underset{\alpha_1}{0.64515} \ 0 \ \cdots \ 0 \ \underset{\alpha_{19}}{-0.02} \ 0 \ \cdots \ 0 \ \underset{\alpha_{24}}{-0.62515} \ 0 \ \cdots \ 0\right]$$

Figure 6. Support vectors considering data from Tables 1and 2. SVs filled with white are visible, unfilled ones are behind the function. Points circled with black are below the plane and the one circled with blue is above it.

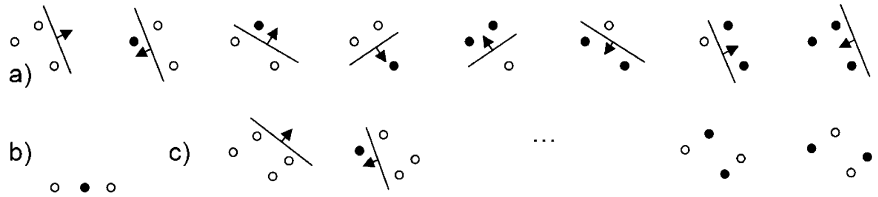


Figure 7. White and black colors represent the two different classes in R^2: (a) This part shows that Φ restricted to linear functions can shatter the given sample; note that the 2l possibilities are drawn. (b) This part presents a set of three points that cannot be shattered by Φ. For a given h, there exists at least one sample of h points that can be shattered by Φ; but assuming that every set of h points can be shattered is generally not true. (c) This part shows that Φ is not able to shatter at least one set of four points in R^2, the VC dimension of the linear Φ in R^2 is three ($h = 3$). The VC dimension of the set of oriented hyperplanes in R^n is $n + 1$.

it is difficult to reach a good generalization from a small sample: the lower l is, the greater the difference between R and R_{emp} can be; and (iii) a classifier that is chosen with a broad class of functions, that is, h is high, can overfit and, thus, gives a risk that is greater than the measured one. Since R cannot be directly minimized, the strategy is to minimize the upper bound. SRM (i) *introduces nested structure* on approximating functions, (ii) *minimizes* R_{emp} on each element producing increasingly complex solutions (models), and (iii) s*elects optimal model* using VC bounds on prediction risk. A structure is a nested set classes of functions S_i such that $S_1 \supset S_2 \supset$ .... Since classes of functions are nested, R_{emp} decreases as the complexity of classes of functions (and hence, h) increases. The risk bound is the sum of ϕ and R_{emp} over the entire structure.

$$R(\alpha) \leq R_{emp}(\alpha) + \phi$$

$$\text{with } \phi = \sqrt{\frac{1}{l}[h(\log(2l/h))] + 1 - \log(\eta/4)} \quad (3)$$

SVMs versus Classification Trees

The SVMs methodology is tested versus classification trees, and generalization is evaluated and discussed for both methods.

Classification Trees Building. Numerous algorithms have been used: ID3, C4.5, a generalized version of ID3 (noted GID3); Assistant-86, Sipina; Improved ChAID (noted Imp. ChAID); a limited search induction tree algorithm (noted Catlett); an improved version of C4.5 (noted Imp. C4.5); and a cost-sensitive version of C4.5.[38] Note that Sipina does not generate trees, but graphs, since it allows the merging of leaves.

CT building proceeds in two complementary and successive steps. First, the training sample is divided into two subsets (growing samples and pruning samples), then the tree is built top-down, starting with an empty tree (composed of one single node, called the top node or root). During this process, the input space is split into subregions, where the output variable remains constant. At each step, a node is selected for splitting; i.e., if the corresponding subset of samples corresponds to a constant output, the node becomes a leaf. Otherwise, the local sample is used to select an optimal split so as to reduce as much as possible the variance (or entropy) of the output variable in the corresponding subsets. To remove the parts of the tree that are not generalizable, the pruning samples that have not been used for tree-growing are used. Together, these two steps allow building a tree that really reflects the relationships among input and output variables, as they may be inferred from the training sample. During the tree-growing, each time a tip node is split, leads to an increase in complexity, and the data fit improves on the growing sample. Once the tree gets large, the speed of improvement slows down because the nodes that are developed are deeper in the tree and, thus, correspond to smaller subsamples. Tree pruning operates in the reverse direction. On the pruning sample, removing test nodes from the tree first increases the data fit, then above a certain level, starts to decrease the quality of the tree. Thus, there is an optimal level of complexity above which the tree would overfit the data, and while below, it would underfit. In practice, the optimal complexity depends on problem features and the size of the available samples.

Sampling, Training Procedure, and Cross-Validation. Different methods can be used to assess the predictive accuracy (e.g., error rate). Both of the two most famous techniques are quickly depicted. The first one, called "split sample" uses a test sample, that is, examples that have not been used at all during the tree-building, neither for growing nor for pruning, as said before. The second one, named cross-validation (CV), is a method for estimating generalization error, based on "resampling".[39] In k-fold CV, the dataset is divided into k subsets of (approximately) equal size. The algorithm is trained k times, each time leaving out one of the subsets from training, and using only the omitted subset to compute whatever error criterion, such as misclassification. If k equals the sample, this is called "leave-one-out" CV. On the other hand, "leave-υ-out" is a more elaborate and expensive version of CV that involves leaving out all possible subsets of υ cases. Leave-one-out CV often works well for estimating generalization error for continuous error functions, such as the mean squared error, but it may perform poorly for discontinuous error functions, such as the number of misclassified cases. k-Fold CV is preferred for the studies in hand.[44] If k gets too small, the error estimate is pessimistically biased because of the difference in training-set size between the full-sample analysis and the CV analyses. Note that CV is quite different from the "split-sample" method that is commonly used. In the split-sample method, only a single subset (the validation set) is used to estimate the generalization error, instead of k different subsets; i.e., there is no "crossing". The distinction between CV and split-sample validation is extremely important[40] because CV is markedly superior for small data sets.

Table 2. Classification Tree Performances on Epoxidation Data[a]

	class		classification trees					
cross-validation	real	predicted	ID3-IV	GID3	ASSISTANT	C4.5	Sipina	Imp. ChAID
5-fold av %	A	A	0.4	0.392	0.325	0.4	0.383	0.392
(nb-std)			(9.6−2.33)	(9.4−3.44)	(7.8−1.6)	(9.6−2.42)	(9.2−1.72)	(9.4−2.73)
	A	B	0.2	0.267	0.183	0.2	0.192	0.25
			(4.8−1.94)	(6.4−2.24)	(4.4−2.33)	(4.8−1.72)	(4.6−1.74)	(6−1.67)
	B	A	0.025	0.033	0.1	0.025	0.042	0.033
			(0.6−0.8)	(0.8−0.75)	(2.4−1.96)	(0.6−1.2)	(1−0.89)	(0.8−0.4)
	B	B	0.375	0.308	0.392	0.375	0.3832	0.325
			(9−1.79)	(7.4−2.42)	(9.4−1.5)	(9−2.83)	(9.2−0.98)	(7.8−1.17)
10-fold av %	A	A	0.4	0.375	0.342	0.392	0.3	0.392
(nb-std)			(4.8−1.89)	(4.5−1.28)	(4.1−1.51)	(4.7−1.1)	(3.6−1.36)	(4.7−1.49)
	A	B	0.233	0.275	0.15	0.225	0.208	0.242
			(2.8−1.54)	(3.3−1.49)	(1.8−1.25)	(2.7−1.55)	(2.5−0.67)	(2.9−0.94)
	B	A	0.025	0.05	0.083	0.033	0.125	0.333
			(0.3−0.64)	(0.6−0.66)	(1−0.89)	(0.4−0.49)	(1.5−1.28)	(0.4−0.49)
	B	B	0.342	0.3	0.425	0.35	0.367	0.333
			(4.1−1.87)	(3.6−1.11)	(5.1−1.3)	(4.2−2.04)	(4.4−1.2)	(4−1.73)
15-fold av %	A	A	0.375	0.391	0.341	0.391	0.334	0.391
(nb-std)			(3−1.21)	(3.13−1.59)	(2.73−1.24)	(3.13−1.15)	(2.67−1.14)	(3.13−1.63)
	A	B	0.191	0.266	0.166	0.234	0.175	0.275
			(1.53−0.96)	(2.13−0.62)	(1.33−1.14)	(1.87−1.02)	(1.4−1.2)	(2.2−1.22)
	B	A	0.05	0.034	0.084	0.034	0.091	0.034
			(0.4−0.8)	(0.27−0.44)	(0.67−0.94)	(0.27−0.44)	(0.73−0.77)	(0.27−0.44)
	B	B	0.384	0.309	0.409	0.341	0.4	0.3
			(3.07−1.18)	(2.47−1.31)	(3.27−1.12)	(2.73−1.53)	(3.2−1.56)	(2.4−1.36)

[a] For each algorithm, three different cross-validations have been run ($k = 5$, 10, and 15). For each combination of k and algorithm, the confusion matrix is given.

Table 3. SVMs Performances on Epoxidation Data[a]

name	kernels	degree (d)	coefficient (r)	gamma (γ)	cost parameter (C)	A−A	A−B	B−A	B−B	recognition rates (A−A, B−B)
Lin1	linear			-	8	0.33	0.38	0.00	0.29	0.63
Poly1	polynomial	3	10	0	7	0.33	0.29	0.00	0.38	0.71
Poly2	polynomial	3	2	0	9	0.25	0.21	0.08	0.46	0.71
Sig1	sigmoid		0	0.25	5	0.38	0.29	0.00	0.33	0.71
Sig2	sigmoid		0	30	2	0.17	0.13	0.17	0.54	0.71
RBF1	RBF			20	5	0.33	0.17	0.00	0.50	0.83
RBF2	RBF			9	9	0.33	0.13	0.00	0.54	0.88
RBF3	RBF	-		15	7	0.33	0.08	0.00	0.58	0.92

[a] 5-Fold and 10-fold cross-validations have been used for defining the best value of C, which controls error penalties (large C means larger penalties).

Results

Olefin Epoxidation. Table 2 shows the classification performances and confusion matrices obtained with six different classification trees using three different k-fold CVs ($k = 5$, 10, and 15) for olefin epoxidation dataset (see Supporting Information). The confusion matrix shows the average classification (real A predicted as A or real B predicted as B) or misclassification (real A predicted as B, i.e., false negative; or real B predicted as A, i.e., false positive). The average percentage of classification or misclassification is given in the first line. The nominal value of cases and the corresponding standard deviation appear in the second line in parentheses. For example, a 5-fold CV is performed on ID3-IV. It means that the 120 cases are randomly separated into 5 groups of 24 catalysts. The algorithm is trained five times, each time a different set of four groups is selected, and the algorithm is evaluated with the fifth one. Therefore, ID3 obtains 40% of A recognized as A (i.e., an average of 9.6 catalysts and a standard deviation of 2.33), 37.5% of B recognized as B, 20% of false negative, and 2.5% of false positive. The sum ($9.6 + 4.8 + 0.6 + 9$) equals 24, as given by the k value. On the other hand, the few parameters of SVMs are investigated and presented in Table 3. For the polynomial kernel function $((\gamma x_i^T x_j + r)^d$, $\gamma > 0)$, the different parameters (d, r, γ) are, respectively, the polynomial degree, a coefficient, and gamma. The selection of an appropriate kernel function is important, since the kernel function defines the feature space in which the training set examples will be classified. The capacity, noted C, is also a parameter of the method (independent from the chosen kernel) and can be seen as a way to control overfitting, since it is a tradeoff between error and margin. C enters into the dual representation of the Lagrangian formulation as a constraint on α_i:$0 \leq \alpha_i \leq C$. In our case, and due to the importance of the capacity setting, we have chosen 5- and a 10-fold CVs for defining/optimizing the best value of C, which controls error penalties (large C means larger penalties). In Figure 9, the recognition performances of the different algorithms for both SVMs and classification trees are compared.

Isomerization Catalysts. Table 4 and Figure 10 show the classification performances and confusion matrices obtained

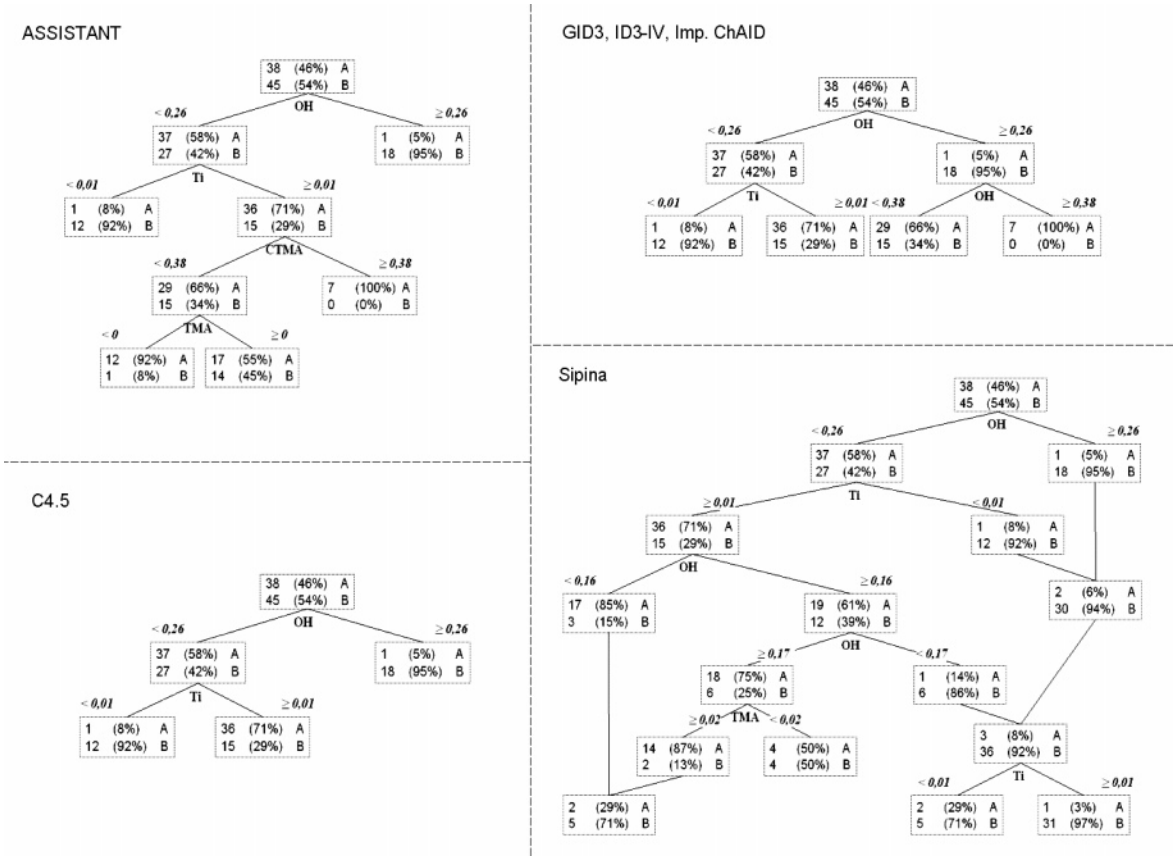


Figure 8. Four different decision graphs on epoxidation data.

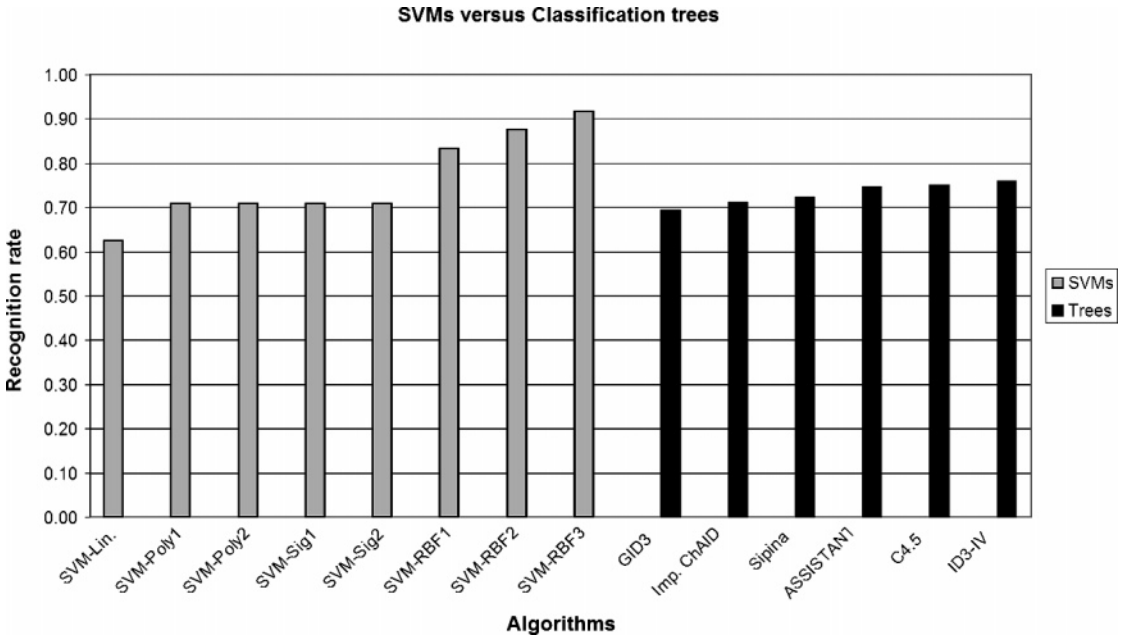


Figure 9. Performance comparison of SVMs and decision tree algorithms, considering only true positives and true negatives, for the epoxidation dataset. The recognition rates for classification trees are the average on each *k*-fold cross-validation with $k = \{5, 10, 15\}$.

with nine different classification trees for *n*-paraffin isomerization data, whereas Table 5 shows the SVM recognition results when different parameters and kernel functions were considered. Ten-fold CV has been used for defining the best value of *C*. Classification results obtained with both techniques are graphically compared in Figure 10. For each CT algorithm, three different *k*-fold CV have been run ($k = 4$, 6, and 8).

Discussion

It is clear from the rules extracted by all classification trees (Figure 8) that $OH^-/(Si + Ti)$ and $Ti/(Si + Ti)$ are always the most discriminative variables. In other words, these features allow one to separate most quickly the samples into the two classes. Actually, the concentration of OH^- in the synthesis gel controls the solution of the silica precursors

Table 4. Classification Tree Performances on Isomerization Data

cross-validation	class real	class predicted	Sipina	ASSISTANT	C4.5	cost-sensitive C4.5	Carlett	Imp ChAID	Imp C4.5	GID3	ID3
4-fold av % (nb-std)	A	A	0.10 (2−1.22)	0.11 (2.25−0.83)	0.13 (2.75−1.09)	0.14 (3−2)	0.12 (2.5−2.5)	0.00 (0−0)	0.00 (0−0)	0.01 (0.25−0.43)	0.13 (2.75−1.48)
	A	B	0.05 (1−1.73)	0.04 (0.75−0.83)	0.01 (0.25−0.43)	0.00 (0−0)	0.02 (0.5−0.5)	0.14 (3−1.87)	0.14 (3−2.12)	0.13 (2.75−0.83)	0.01 (0.25−0.43)
	B	A	0.05 (1−1.22)	0.06 (1.25−0.43)	0.05 (1−0.71)	0.08 (1.75−2.49)	0.05 (1−1)	0.00 (0−0)	0.05 (1−1)	0.04 (0.75−1.3)	0.05 (1−0.71)
	B	B	0.81 (16.75−1.79)	0.80 (16.5−1.12)	0.81 (16.75−1.64)	0.77 (16−1.87)	0.81 (16.75−16.75)	0.86 (17.75−1.92)	0.81 (16.75−1.48)	0.82 (17−1.22)	0.81 (16.75−2.28)
6-fold av % (nb-std)	A	A	0.08 (1.17−1.07)	0.12 (1.67−0.94)	0.10 (1.33−1.37)	0.13 (1.83−1.07)	0.11 (1.5−0.96)	0.01 (0.17−0.37)	0.01 (0.17−0.37)	0.01 (0.17−0.37)	0.13 (1.83−1.77)
	A	B	0.06 (0.83−1.21)	0.02 (0.33−0.47)	0.05 (0.67−0.75)	0.01 (0.17−0.37)	0.04 (0.5−0.5)	0.13 (1.83−1.07)	0.13 (1.83−2.03)	0.13 (1.83−1.34)	0.01 (0.17−0.37)
	B	A	0.06 (0.83−0.69)	0.05 (0.67−0.75)	0.05 (0.67−0.75)	0.08 (1.17−1.07)	0.05 (0.67−1.11)	0.05 (0.67−1.11)	0.02 (0.33−0.75)	0.04 (0.5−1.12)	0.05 (0.67−0.75)
	B	B	0.80 (11−1.73)	0.81 (11.17−1.21)	0.81 (11.17−1.57)	0.77 (10.67−1.37)	0.81 (11.17−1.21)	0.81 (11.17−1.07)	0.83 (11.5−2.22)	0.82 (11.33−1.37)	0.81 (11.17−2.19)
8-fold av % (nb-std)	A	A	0.13 (1.38−0.86)	0.10 (1−0.87)	0.08 (0.88−0.78)	0.10 (1−1)	0.10 (1−0.5)	0.04 (0.38−0.7)	0.04 (0.38−0.48)	0.05 (0.5−0.71)	0.13 (1.38−1.11)
	A	B	0.01 (0.13−0.33)	0.05 (0.5−0.71)	0.06 (0.63−1.32)	0.05 (0.5−0.71)	0.05 (0.5−0.5)	0.11 (1.13−1.17)	0.11 (1.13−1.05)	0.10 (1−1.22)	0.01 (0.13−0.33)
	B	A	0.05 (0.5−0.71)	0.04 (0.38−0.48)	0.02 (0.25−0.66)	0.08 (0.88−0.93)	0.02 (0.25−0.43)	0.02 (0.25−0.43)	0.05 (0.5−0.71)	0.05 (0.5−0.71)	0.07 (0.75−0.97)
	B	B	0.81 (8.38−0.99)	0.82 (8.5−0.87)	0.83 (8.63−0.99)	0.77 (8−1.41)	0.83 (8.63−0.86)	0.83 (8.63−1.32)	0.81 (8.38−1.11)	0.81 (8.38−1.22)	0.78 (8.13−1.27)

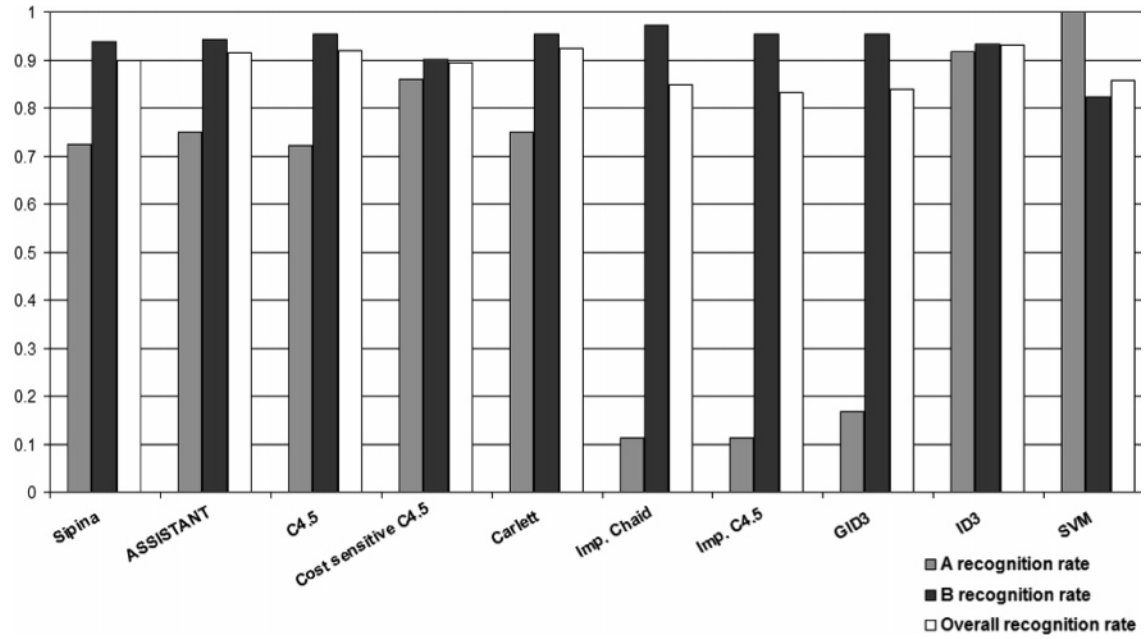


Figure 10. Performance comparison of the best SVM and decision trees, considering the full confusion matrix, for the isomerization dataset. The recognition rates for classification trees are the average on each *k*-fold cross-validation with $k = \{5, 10, 15\}$.

Table 5. SVMs Performances on Isomerization Data

kernels	coefficient (r)	degree (d)	gamma (γ)	cost parameter (C)	nb. of SV	nb. of bounded SV	SV class 1	SV class 2	train (75)	test (25)	overall (100)	10-CV	penalties class 1	penalties class 2	CV k value	capacity boundary
RBF			0.059	0.5	19	14	8	11	87.09	80.952	85.542	87.09	10	1	6	
RBF			0.059	3	22	13	4	18	80.65	85.71	81.93	80.65	10	1	8	
RBF			0.059	2	24	15	5	19	79.03	85.71	80.72	79.03	10	1	6	2
RBF			3	1	30	23	8	22	79.03	85.71	80.72	79.03	3	1	10	
poly	0	3	0.059	2	62	61	8	54	77.42	85.714	79.52	65.33	7	1	10	2

and the crystallization/organization of siliceous species that induce the formation of mesoporous structures. Indeed, no active catalysts were obtained above a limiting concentration of $OH^-/(Si + Ti) \sim 0.27$, since concentrations above this value lead to low surface amorphous materials. With respect to Ti concentration, tetrahedrally coordinated titanium species

are the active epoxidation sites, whereas octahedrally coordinated ones promote secondary reactions and the subsequent catalyst deactivation. Therefore, the amount of titanium should be located in a region where the maximal number of active sites are formed while minimizing other titanium species. The other two variables refer to the concentration of surfactants, and they influence the type of mesoporous structure and its degree of long-distance order, but the variation of these material properties is not as decisive for the final catalytic activity as $OH^-/(Si + Ti)$ and $Ti/(Si + Ti)$. Looking at the influence of CTMA concentration, there can be seen two different regions where good catalysts appear. Two different materials have been found in the high activity areas defined by CTMA. Roughly speaking, lowest OH^- values, independently of CTMA, correspond to a hexagonal poor arrangement (MCM-41), whereas highest OH (always below 0.27 molar ratio) and CTMA values lead to a cubic arrangement (MCM-48), both well-known mesoporous structures produced by self-assembling of CTMA micelles (liquid crystals) and silica. This analysis of the whole epoxidation data by the different classification tree algorithms is fully in agreement with the human-based analysis depicted in ref 17. Conversely, the Sipina algorithm allows obtaining a different type of rule, since it is possible to merge conditions (merging leaves). For example, bad catalysts (class B in Figure 8, Sipina) are obtained with a probability of 94% when $OH \geq 0.26$ *or* when $OH < 0.26$ *and* $Ti < 0.01$. This statement is also in agreement with the human-based observations.

Figure 9 presents the recognition performances of the different algorithms for both SVMs and classification trees. It is clear that simple SVMs have medium performances rather equal to CT performances; however, RBF kernel with different sets of parameters exhibits very high levels of recognition rates, which outperform every classification tree previously tested. The SVM-RBF3 has a 92% recognition rate and 8% false negatives, which is the lowest level among all strategies. Decision trees are generally preferred over other nonparametric techniques because of the readability of their learned hypotheses and the efficiency of training and evaluation. Furthermore, the risk of overfitting the training data is less severe for SVM models, since complexity is considered directly during the learning stage.

On the other hand, the composition of isomerization catalysts was optimized by choosing different supports and promoters and by tuning simultaneously their loading (wt %). As a result of the optimization, the best catalysts showed combinations of zirconia, with sulfur as an acidity enhancer and diverse metallic promoters. Isomerization data shows a low amount of catalysts (14.45%, Figure 1) with an acceptable activity; therefore, in this case,it is the correct prediction of class A catalysts of major importance. In other words, false negatives should be minimized. The minimization of false negatives is very important when dealing with HT experimentation programs for catalyst or new materials discovery, especially when exploring multicomponent catalysts. Typically, in such discovery programs, most of the screened materials show undetectable catalytic activity, and very few exhibit some activity (so-called *hits*). The appropriate recognition (and prediction) of these hits is crucial for the subsequent optimization steps in the next experimentation rounds. In fact, false negatives imply that promising/interesting catalyst composition will be neglected, and the further optimization will not take into account such important catalytic compositions; thus, the convergence is strongly prejudiced.

Classification trees (Figure 10) show rather good recognition rates for isomerization data; however, some algorithms fail totally when considering the learning on the interesting class (i.e., A), such as Imp. C4.5, GID3, and Imp. ChAID. Nevertheless, SVM models (Figure 10) allow one to obtain 100% of recognition of active catalyst, while the number of false positives is very low (i.e., more than 82%). The confusion matrix for all SVMs is the following: 100% of A predicted as A and 82.35% of B recognized as B. RBF-4 is kept as the best model (Table 5). Linear penalty functions (Table 5) have been used to penalize classification violations in SVM classifier design. Note that it might be argued that it is difficult to calculate the cost of classification errors in many real-world applications. Thus, it is possible that an algorithm that ignores the cost of classification errors may be more robust and useful than an algorithm that is sensitive to classification errors. However, in our case, the results are much better than without penalties, since not only SVM outperforms (Figure 10) CT, but also because C can be kept rather low and the classification rates remain stable for train, test, and CV.

Previous knowledge is applied for the design of the catalytic parameter space to be explored. However, it is usually very difficult to have an a priori knowledge about the topology of the output space, especially when a high number of synthesis parameters are varied simultaneously. SVM will operate correctly even if the designer does not know exactly which features of the training data are being used in the kernel-induced feature space, but integrating a priori knowledge is possible through SVM kernel functions. The kernel expresses prior knowledge about the phenomenon being modeled, encoded as a similarity measure between two vectors.

Overfitting is traditionally defined as training some flexible representation so that it memorizes the data but fails to predict well. There are many kinds of overfitting, and a nonexhaustive list is given in Table 6. The first case (the most current) is handled in SVMs through the definition of the kernels. One should start with the simplest one: that is, linear; then polynomials; and finally, sigmoid and RBF, as is done in this study. The second and third causes of overfitting have been handled, for each study that has been conducted here, through k-fold CV with different values of k. The fourth cause is common in combinatorial studies because clustering methods are often used to define groups of catalysts on the basis of their independent features. When a given clustering method is applied before a learning algorithm, the performance of this learning strategy cannot be directly compared to another that only takes into account the raw values. Moreover, one has to be careful not to integrate the test set into the clustering method, since in reality, one should assign the group label to new catalysts

Table 6. Possible Overfitting Sources

name	how	why	remedy
traditional overfitting	train a complex predictor on too few examples		use a simpler predictor, get more training examples (if possible) or integrate over many predictors
parameter tweak overfitting	use a learning algorithm with many parameters; choose the parameters based on the test set performance	for example, choosing the features so as to optimize test set performance can achieve this kind of overfitting	use a simpler predictor, get more training examples (if possible) or integrate over many predictors
weak measure	use a measure of performance which is especially weak to overfitting	for example leave-one-out cross-validation is weak	prefer stronger measures of performance
human-loop overfitting	use a human as part of a learning algorithm and do not take into account overfitting by the entire human/computer interaction	this is subtle and comes in many forms; one example is a human using a clustering algorithm (on training and test examples) to guide learning algorithm choice	make sure test examples are not available to the human
data set selection	chose to report results on some subset of datasets where the given algorithm performs well	data set selection subverts this and is very difficult to detect	use comparisons on standard datasets; select datasets without using the test set
reformulation of the problem	alter the problem so that the performance increases	for example, ignore asymmetric false positive/false negative costs	make sure problem specifications are clear
old datasets	create an algorithm for the purpose of improving performance on old datasets	after a dataset has been released, algorithms can be made to perform well on the dataset using a process of feedback design, indicating better performance	prefer simplicity in algorithm design, weight newer datasets higher in consideration

on the basis of supervised learning methods. The fifth cause has been illustrated through the introduction of SVM. We start with a subset of both variables and cases (Table 1a,b) and progressively show that the selection of catalysts allows one to find a good solution for the given set but that these results are not generalizable (Figures 2, 3). The sixth cause is quite common when comparing classification algorithm and regressions for classes defined through a threshold setting. The last cause is very common in pure computer science domain, since some benchmarks have been available through the Web for a long time. Therefore, understanding what makes the benchmark difficult for a given method can permit one to adapt the given algorithm. Such strategy produces very specific algorithms. One should always test its methodology on benchmarks, since different levels of complexity can be artificially created, and the cost of experiments is null. However, as said before, the correlation between benchmarks and real data is not straightforward, since surface similarity should be handled properly.

Conclusions

SVM methodology has been evaluated using data derived from two different industrial fields: oil refining and petrochemistry, and the selected reactions for each application are gasoline isomerization and olefin epoxidation, respectively. It has been shown that SVMs outperform other algorithms by improving the recognition performance (generalization) results. More specifically, the performance of SVMs and different classification trees has been compared. SVM methodology allows one to obtain the optimal solution for the given set of training data and parameters. This is due to the fact that the space search is convex; the method is based on Lagrangian calculations with a unique optimal solution, avoiding the local minima, which are one of the pitfalls of perceptrons.

Acknowledgment. We thank the referees for their comments, which permitted us to make the paper clearer. Financial support by EU Project TOPCOMBI and Grant FPU AP2003-4635 of the Spanish Government is gratefully acknowledged.

Supporting Information Available. Two tables showing experimental data; SVM details. This material is available free of charge via the Internet at http://pubs.acs.org.

References and Notes

(1) Senkan, S. *Angew. Chem., Int. Ed.* **2001**, *40*, 312−329.
(2) (a) Hagemeyer, A.; Jandeleit, B.; Liu, Y. M.; Poojary, D. M.; Turner, H. W.; Volpe, A. F.; Weinberg, W. H. *Appl. Catal.* **2001**, *221*, 23−43. (b) Burello, E.; Marion, P.; Galland, J. C.; Chamard, A.; Rothenberg, G. *Adv. Synth. Catal.* **2005**, *347*, 803. (c) Corma, A. *J. Catal*. **2003**, *216*, 298−312.
(3) Malo, N.; Hanley, J.; Cerquozzi, S.; Pelletier, J.; Nadon, R. *Nat. Biotechnol.* **2006**, *24*, 167−175.
(4) Baumes, L. A.; Farruseng, D.; Lengliz, M.; Mirodatos, C. *QSAR Comb. Sci.* **2004**, *29*, 767−778. (b) Baumes, L. A.; Corma, A.; Lengliz, M.; Serra, J. M.; Moliner, M. *Int. Conf. EuropaCat-VII*; Sophia, Bulgaria, Aug. 28−Sept. 1, 2005; Book of abstracts OF1-09.
(5) Corma, A.; Serra, J. M.; Serna, P.; Argente, E.; Valero, S.; Botti, V. *J. Catal.* **2005**, *25*, 459−469.
(6) (a) Baumes, L. A.; Jouve, P.; Farrusseng, D.; Lengliz, M.; Nicoloyannis, N.; Mirodatos, C. 7th International Conference on Knowledge-Based Intelligent Information & Engineering

Systems (KES '2003), University of Oxford, U.K., Sept. 3−5, 2003; In *Lecture Notes in AI*; LNCS/LNAI series; Palade, V., Howlett, R. J., Jain, L. C., Eds.; Springer-Verlag: New York. (b) Serra, J. M.; Corma, A.; Farrusseng, D.; Baumes, L. A.; Mirodatos, C.; Flego, C.; Perego, C. *Catal. Today* **2003**, *81*, 425−436. (c) Serra, J. M.; Corma, A.; Farrusseng, D.; Baumes, L. A.; Mirodatos, C.; Flego, C.; Perego, C. *Catal. Today* **2003**, *81*, 425−436.

(7) (a) Pereira, S. R. M.; Clerc, F.; Farrusseng, D.; van der Waal, J. C.; Maschmeyer, T.; Mirodatos, C. *QSAR Comb. Sci.* **2005**, *24*, 45−57.

(8) Corma, A.; Serra, J. M.; Argente, E.; Valero, S.; Botti, V. *ChemPhysChem* **2002**, *3*, 939−945.

(9) Farruseng, D.; Klanner, K.; Baumes, L. A.; Lengliz, M.; Mirodatos, C.; Schüth, F. *QSAR Comb. Sci.* **2004**, *23* (9), 767−778.

(10) (a) Omata, K.; Watanabe, Y.; Hashimoto, M.; Umegaki, T.; Yamada M. *Ind. Eng. Chem. Res.* **2004**, *43*, 3282−3288. (b) Umegaki, T.; Watanabe, N.; Nukui, N.; Omata, K.; Yamada, M.; *Energy Fuels* **2003**, *17*, 850−856.

(11) Manallack, D. T.; Livinstone, D. J. *Eur. J. Med. Chem.* **1999**, *34*, 195−208.

(12) Reference deleted in proof.

(13) (a) Wolf, D.; Buyevskaya, O. V.; Baerns, M. *Appl. Catal., A***2000**, 63−77. (b) Buyevskaya, O. V.; Wolf, D.; Baerns, M. *Catal. Today*, **2000**, *62*, 91−99. (c) Buyevskaya, O. V.; Bruckner, A.; Kondratenko, E. V.; Wolf, D.; Baerns, M. *Catal. Today* **2001**, *67*, 369−378. (d) Corma, A.; Serra, J. M.; Chica, A. In *Principles and methods for accelerated catalyst design and testing*; Derouane, E. G., Parmon, V., Lemos, F., Ribeiro, F. R., Eds.; Kluver Academic Publishers: Dordrecht, The Netherlands, 2002; pp 153−172. (e) Wolf, D. In *Principles and methods for accelerated catalyst design and testing*; Derouane, E. G., Parmon, V., Lemos, F., Ribeiro, F. R., Eds.; Kluver Academic Publishers: Dordrecht, The Netherlands, 2002; pp 125−133. (f) Vauthey, I.; Baumes, L. A.; Hayaud, C.; Farrusseng, D.; Mirodatos, C.; Grubert, G.; Kolf, S.; Cholinska, L.; Baerns, M.; Pels, J. R. Eurocombicat 2002: European Workshop on Combinatorial Catalysis; Ischia, Italy, June 2−5, 2002; abstract available at http://www.ec-combicat.org/dowmloads/abstracts/Vauthey.pdf (Website accessed March 15, 2006). (g) Holena, M.; Baerns, M. *Catal. Today* **2003**, *81*, 485−494. (h) Serra, J. M.; Chica, A.; Corma, A. *Appl. Catal., A* **2003**, *239*, 35−42. (i) Grubert, G.; Kondratenko, E. V.; Kolf, S.; Baerns, M.; van Geem, P.; Parton, R. *Catal. Today* **2003**, *81*, 337−345. (j) Maier, W. F.; Frantzen, A.; Frenzer, G.; Kirsten, G.; Saalfrank, J.; Scheidtmann, J.; Stutz, B.; Thome, C.; Weiss, P.-A.; Wolter, T. *Polym. Mater. Sci. Eng.* **2004**, *90*, 652−653.

(14) (a) Tompos, A.; Margitfalvi, J. L.; Tfirst, E.; Veqvari, L.; Jaloull, M. A.; Khalfalla, H. A.; Elgarni, M. A. *Appl. Catal., A* **2005**, *285*, 65−78. (b) Baumes, L. A. Unpublished results.

(15) (a) Baumes, L. A. *J. Comb. Chem.* **2006**, *8*, 304−314. (b) Baumes, L. A.; Serra, J. M.; Millet, I.; Serna, P.; Corma, A. *Int. Conf. EuropaCat-VII*, Sophia, Bulgaria, Aug. 28−Sept. 01, 2005; Book of abstracts OF1-08.

(16) Vapnik, V.; Chervonenkis, A. *Theory of Pattern Tecognition* (in Russian); Nauka: Moscow, 1974; (German translation) *Theorie der Zeichenerkennung*; Akademic-Verlag: Berlin, 1979.

(17) (a) Liu, H. X.; Zhang, R. S.; Luan, F.; Yao, X. J.; Liu, M. C.; Hu, Z. D.; Fan B. T. *J. Chem. Inf. Comput. Sci.* **2003**. 43, 900−907. (b) Zhan, Y.; Shen, D. *Pattern Recognit.* **2005**, *38*, 157−161.

(18) (a) Burbidge, R.; Trotter, M.; Buxton, B.; Holden, S. *Comput. Chem.* **2001**, *26*, 5−14. (b) Zhang, S. W.; Pan, Q.; Zhang, H. C.; Zhang, Y. L.; Wang, H. Y. *Bioinformatics* **2003,** *19*, 2390−2396.

(19) Jemwa, G. T.; Aldrich, C. *AIChE J.* **2005**, *51*, 526−543.

(20) (a) Xue, C. X.; Zhang, R. S.; Liu, H. X.; Liu, M. C.; Hu, Z. D.; Fan, B. T. *J. Chem. Inf. Comput. Sci*. **2004**, *44*, 1267−1274. (b) Liu, H. X.; Zhang, R. S.; Yao, X. J.; Liu, M. C.; Hu, Z. D.; Fan, B. T. *J. Chem. Inf. Comput. Sci.* **2004**, *43*, 161−167. (c) Liu, H. X.; Zhang, R. S.; Yao, X. J.; Liu, M. C.; Hu, Z. D.; Fan, B. T. *Anal. Chim. Acta* **2004**, *525*, 31−41.

(21) Chang, C.; Lin, C. http://www.csie.ntu.edu.tw/∼cjlin/libsvm/ (accessed March 15, 2006).

(22) (a) http://svmlight.joachims.org/ (accessed March 15, 2006). (b) Joachims, T. *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*; Kluwer: Norwell, MA, 2002. (c) Joachims, T. Advances in Kernel Methods—Support Vector Learning. In *Making Large-Scale SVM Learning Practical*; Schölkopf, B., Burges, C., Smola, A., Eds.; MIT Press: Cambridge, MA, 1999.

(23) Corma A.; Jordá, J. L.; Navarro, M. T.; Rey, F. *Chem. Commun.* **1998**, 1899−1900.

(24) Corma, A.; Frontela, J.; Lázaro, J. *Process for the Obtainment of a Light Paraffin Isomerization Catalyst*; U.S. Patent 5,-057,471, 1991; patent granted to CEPSA.

(25) Serra, J. M.; Chica, A.; Corma, A. *Appl. Catal., A* **2003**, *239*, 35−42.

(26) (a) Zighed, D.; Lallich, S.; Muhlenbach, F. *Appl. Stochastic Models Business Ind.* **2005**, *21* (2), 187−197. (b) Zighed, D.; Lallich, S.; Muhlenbach, F. *Principles of Data Mining and Knowledge Discovery*; 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD '02), Helsinki, Finland, August 2002; LNAI; Vol. 2431; pp 475−487.

(27) (a) Klanner, C.; Farrusseng, D.; Baumes, L. A.; Mirodatos, C.; Schüth, F. *QSAR Comb. Sci.* **2003**, *22*, 729−736. (b) Klanner, C.; Farrusseng, D.; Baumes, L. A.; Lengliz, M.; Mirodatos, C.; Schüth, F. *Angew. Chem., Int. Ed.* **2004**, *43* 5347−5349. (c) Farrusseng, D.; Klanner, C.; Baumes, L. A.; Lengliz, M.; Mirodatos, C.; Schüth, F. *QSAR Comb. Sci.* **2005**, *24*, 78−93.

(28) $\vec{x}_i = [x_{i_1} \ldots x_{i_m} \ldots x_{i_M}] = x_i$ is a vector with $i \in [1 \ldots l]$, and M is the number of variables. Let us suppose for the moment that $\chi = \mathrm{R}^M$, i.e., $\forall m \in [1 \ldots M]$, $x_{i_m} \in \mathrm{R}$.

(29) The unqualified term "norm" refers the flavor of vector norm technically known as the L2-norm. This norm is denoted $||w||_2$, $||w||$, or $|w|$ and gives the length of an n-vector, x. It can be computed as $||w|| = \sqrt{\sum_{i=1}^{i=n} x_i^2}$.

(30) Vapnik, V. *The Nature of Statistical Learning Theory*; Springer-Verlag: New York, 1995.

(31) Cortes, C.; Vapnik, V. *Mach. Learn.* **1995**, *20* (3), 273−297.

(32) (a) Fletcher, R. *Practical Methods of Optimization*, 2nd ed.; Wiley and Sons, Inc.: New York, 1987. (b) McCormick, G. P. *Non Linear Programming: Theory, Algorithms and Applications*; Wiley and Sons, Inc.: New York, 1983.

(33) $x_i, x_j) = \sum_{m=1}^{m=M} x_{i_m} x_{j_m} = x_i^{\mathrm{T}} x_j$

(34) $x_{i1}^2, \sqrt{2}x_{i_1}x_{i_2}, x_{i_2}^2)(x_{j_1}^2, \sqrt{2}x_{j_1}x_{j_2}, x_{j_2}^2)^T = x_{i_1}^2 x_{j_1}^2 + 2x_{i_1}x_{i_2}x_{j_1}x_{j_2} + x_{i_2}^2 x_{j_2}^2 = (x_{i_1}x_{j_1} + x_{i_2}x_{j_2})^2 = (x_i, x_j)^2$

(35) (a) Bishop, C. *Neural Networks for Pattern Recognition*; Oxford University Press: New York, 1995. (b) MacKay, D. J. *Neural Comput.* **1992**, *4* (3), 448−472. (c) Friedman, J. In *From Statistics to Neural Networks, Theory and Pattern Recognition Applications*; Cherkassky, V., Friedman, J., Wechsler, H., Eds.; NATO ASI Series F; Springer Verlag: New York, 1994; pp 1−61. (d) Geman, S.; Bienenstock, E.; Doursat, R. *Neural Comput.* **1992**, *4*, 1−58.

(36) Vapnik, V. Springer-Verlag: New York, 1982.

(37) Burges, C. J. C. *Data Min. Knowledge Discovery* **1997**, *2* (2), 121−167.

(38) One reference is given for each algorithm respecting the order of appearance in the text. (a) Quinlan, J. R. *Mach. Learn.* **1986**, *1*, 81−106. (b) Quinlan, J. R. *C4.5: Programs for*

Machine Learning; Morgan Kaufmann: San Mateo, CA, 1993. (c) Cheng, J.; Fayyad, U. M.; Irani, K. B.; Qian Z. *5th International Conference of Machine Learning*, Ann Arbor, MI; Morgan Kaufman: San Mateo, CA, pp 100−108. (d) Cestnik, B.; Kononenko, I.; Bratko, I. In *Progress in Machine Learning*; Bratko, I., Lavrac, N., Eds.; Sigma Press: Wilmslow, England, 1987, pp 31−45. (e) Zighed, D. A.; Auray, J. P.; Duru, G. *SIPINA: Méthode et Logiciel*, A ed.; Lacassagne: Lyon, France, 1992. (f) Rakotomalala, R.; Zighed, D. In *Proc. AIDRI '97*; 1997. (g) Catlett, J. Ph.D. Dissertation, University of Sydney, Australia, 1991. (h) Rakotomalala, R.; Lallich, S. In *Proc. Int. Conf. Comput. Sci. Informat.* **1998**, 25−27. (i) Chauchat, J.−H.; Rakotomalala, R.; Carloz, M.; Pelletier, C. In *Data Mining for Marketing Applications (Working Notes)*, PKDD '2001; Freiburg, Germany, September 2001, pp 1−13; http://www.uhasselt.be/iteo/articles/chauchat_workshop.pdf (Website accessed March 15, 2006). (j) Weiss, S. M.; Kulikowski, C. A. *Computer Systems That Learn*; Morgan Kaufmann: San Mateo, CA, 1991. (k) Efron, B.; Tibshirani, R. J. *An Introduction to the Bootstrap*; Chapman & Hall: London, 1993..

(39) (a) Hjorth, J. S. U. *Computer Intensive Statistical Methods Validation, Model Selection, and Bootstrap*; Chapman & Hall: London, 1994. (b) Plutowski, M.; Sakata, S.; White, H. In *Advances in Neural Information Processing Systems 6*; Cowan, J. D.; Tesauro, G.; Alspector, J., Eds.; Morgan Kaufman: San Mateo, CA, 1994; pp 391−398. (c) Shao, J.; Tu, D. *The Jackknife and Bootstrap*; Springer-Verlag: New York, 1995. (d) For an insightful discussion of the limitations of CV choice among several learning methods, see, for instance: Goutte, C. *Neural Comput.* **1997**, *9*, 1211−1215.

(40) Stone, M. *Biometrika* **1977**, *64*, 29−35.

CC050093M