
CHAPTER 6

Ovidiu Ivanciuc. Applications of Support Vector Machines in Chemistry. In: *Reviews in Computational Chemistry, Volume 23*, Eds.: K. B. Lipkowitz and T. R. Cundari. Wiley-VCH, Weinheim, 2007, pp. 291–400.

Applications of Support Vector Machines in Chemistry

Ovidiu Ivanciuc

*Sealy Center for Structural Biology,
Department of Biochemistry and Molecular Biology,
University of Texas Medical Branch, Galveston, Texas*

INTRODUCTION

Kernel-based techniques (such as support vector machines, Bayes point machines, kernel principal component analysis, and Gaussian processes) represent a major development in machine learning algorithms. Support vector machines (SVM) are a group of supervised learning methods that can be applied to classification or regression. In a short period of time, SVM found numerous applications in chemistry, such as in drug design (discriminating between ligands and nonligands, inhibitors and noninhibitors, etc.), quantitative structure-activity relationships (QSAR, where SVM regression is used to predict various physical, chemical, or biological properties), chemometrics (optimization of chromatographic separation or compound concentration prediction from spectral data as examples), sensors (for qualitative and quantitative prediction from sensor data), chemical engineering (fault detection and modeling of industrial processes), and text mining (automatic recognition of scientific information).

Support vector machines represent an extension to nonlinear models of the generalized portrait algorithm developed by Vapnik and Lerner.¹ The SVM algorithm is based on the statistical learning theory and the Vapnik–Chervonenkis

(VC) dimension.² The statistical learning theory, which describes the properties of learning machines that allow them to give reliable predictions, was reviewed by Vapnik in three books: *Estimation of Dependencies Based on Empirical Data*,³ *The Nature of Statistical Learning Theory*,⁴ and *Statistical Learning Theory*.⁵ In the current formulation, the SVM algorithm was developed at AT&T Bell Laboratories by Vapnik et al.^{6–12}

SVM developed into a very active research area, and numerous books are available for an in-depth overview of the theoretical basis of these algorithms, including *Advances in Kernel Methods: Support Vector Learning* by Schölkopf et al.,¹³ *An Introduction to Support Vector Machines* by Cristianini and Shawe–Taylor,¹⁴ *Advances in Large Margin Classifiers* by Smola et al.,¹⁵ *Learning and Soft Computing* by Kecman,¹⁶ *Learning with Kernels* by Schölkopf and Smola,¹⁷ *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms* by Joachims,¹⁸ *Learning Kernel Classifiers* by Herbrich,¹⁹ *Least Squares Support Vector Machines* by Suykens et al.,²⁰ and *Kernel Methods for Pattern Analysis* by Shawe-Taylor and Cristianini.²¹ Several authoritative reviews and tutorials are highly recommended, namely those authored by Schölkopf et al.,⁷ Smola and Schölkopf,²² Burges,²³ Schölkopf et al.,²⁴ Suykens,²⁵ Schölkopf et al.,²⁶ Campbell,²⁷ Schölkopf and Smola,²⁸ and Sanchez.²⁹

In this chapter, we present an overview of SVM applications in chemistry. We start with a nonmathematical introduction to SVM, which will give a flavor of the basic principles of the method and its possible applications in chemistry. Next we introduce the field of pattern recognition, followed by a brief overview of the statistical learning theory and of the Vapnik–Chervonenkis dimension. A presentation of linear SVM followed by its extension to nonlinear SVM and SVM regression is then provided to give the basic mathematical details of the theory, accompanied by numerous examples. Several detailed examples of SVM classification (SVMC) and SVM regression (SVMR) are then presented, for various structure-activity relationships (SAR) and quantitative structure-activity relationships (QSAR) problems. Chemical applications of SVM are reviewed, with examples from drug design, QSAR, chemometrics, chemical engineering, and automatic recognition of scientific information in text. Finally, SVM resources on the Web and free SVM software are reviewed.

A NONMATHEMATICAL INTRODUCTION TO SVM

The principal characteristics of the SVM models are presented here in a nonmathematical way and examples of SVM applications to classification and regression problems are given in this section. The mathematical basis of SVM will be presented in subsequent sections of this tutorial/review chapter.

SVM models were originally defined for the classification of linearly separable classes of objects. Such an example is presented in Figure 1. For

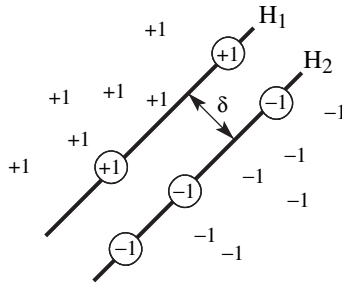


Figure 1 Maximum separation hyperplane.

these two-dimensional objects that belong to two classes (class +1 and class -1), it is easy to find a line that separates them perfectly.

For any particular set of two-class objects, an SVM finds the unique hyperplane having the maximum margin (denoted with δ in Figure 1). The hyperplane H_1 defines the border with class +1 objects, whereas the hyperplane H_2 defines the border with class -1 objects. Two objects from class +1 define the hyperplane H_1 , and three objects from class -1 define the hyperplane H_2 . These objects, represented inside circles in Figure 1, are called support vectors. A special characteristic of SVM is that the solution to a classification problem is represented by the support vectors that determine the maximum margin hyperplane.

SVM can also be used to separate classes that cannot be separated with a linear classifier (Figure 2, left). In such cases, the coordinates of the objects are mapped into a feature space using nonlinear functions called feature functions ϕ . The feature space is a high-dimensional space in which the two classes can be separated with a linear classifier (Figure 2, right).

As presented in Figures 2 and 3, the nonlinear feature function ϕ combines the input space (the original coordinates of the objects) into the feature space, which can even have an infinite dimension. Because the feature space is high dimensional, it is not practical to use directly feature functions ϕ in

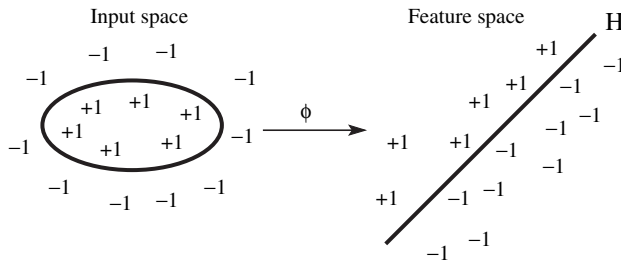


Figure 2 Linear separation in feature space.

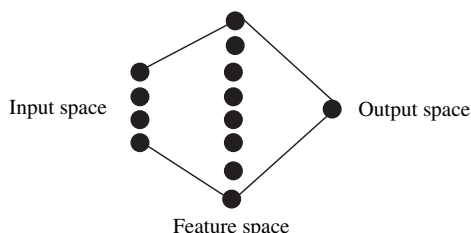


Figure 3 Support vector machines map the input space into a high-dimensional feature space.

computing the classification hyperplane. Instead, the nonlinear mapping induced by the feature functions is computed with special nonlinear functions called kernels. Kernels have the advantage of operating in the input space, where the solution of the classification problem is a weighted sum of kernel functions evaluated at the support vectors.

To illustrate the SVM capability of training nonlinear classifiers, consider the patterns from Table 1. This is a synthetic dataset of two-dimensional patterns, designed to investigate the properties of the SVM classification algorithm. All figures from this chapter presenting SVM models for various datasets were prepared with a slightly modified version of Gunn's MATLAB toolbox, <http://www.isis.ecs.soton.ac.uk/resources/svminfo/>. In all figures, class +1 patterns are represented by +, whereas class -1 patterns are represented by black dots. The SVM hyperplane is drawn with a continuous line, whereas the margins of the SVM hyperplane are represented by dotted lines. Support vectors from the class +1 are represented as + inside a circle, whereas support vectors from the class -1 are represented as a black dot inside a circle.

Table 1 Linearly Nonseparable Patterns Used for the SVM Classification Models in Figures 4–6

Pattern	x_1	x_2	Class
1	2	4.5	1
2	2.5	2.9	1
3	3	1.5	1
4	3.6	0.5	1
5	4.2	2	1
6	3.9	4	1
7	5	1	1
8	0.6	1	-1
9	1	4.2	-1
10	1.5	2.5	-1
11	1.75	0.6	-1
12	3	5.6	-1
13	4.5	5	-1
14	5	4	-1
15	5.5	2	-1

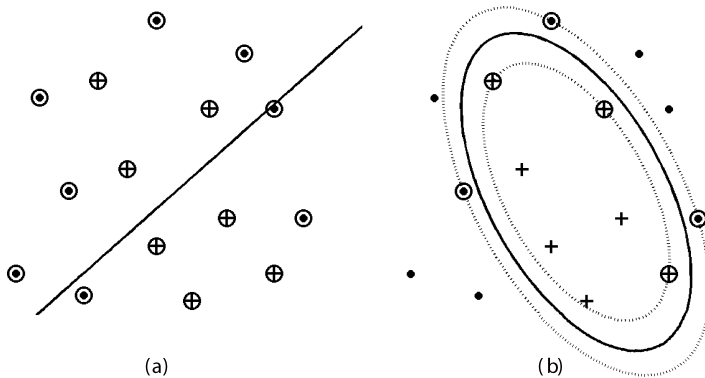


Figure 4 SVM classification models for the dataset from Table 1: (a) dot kernel (linear), Eq. [64]; (b) polynomial kernel, degree 2, Eq. [65].

Partitioning of the dataset from Table 1 with a linear kernel is shown in Figure 4a. It is obvious that a linear function is not adequate for this dataset, because the classifier is not able to discriminate the two types of patterns; all patterns are support vectors. A perfect separation of the two classes can be achieved with a degree 2 polynomial kernel (Figure 4b). This SVM model has six support vectors, namely three from class +1 and three from class -1. These six patterns define the SVM model and can be used to predict the class membership for new patterns. The four patterns from class +1 situated in the space region bordered by the +1 margin and the five patterns from class -1 situated in the space region delimited by the -1 margin are not important in defining the SVM model, and they can be eliminated from the training set without changing the SVM solution.

The use of nonlinear kernels provides the SVM with the ability to model complicated separation hyperplanes in this example. However, because there is no theoretical tool to predict which kernel will give the best results for a given dataset, experimenting with different kernels is the only way to identify the best function. An alternative solution to discriminate the patterns from Table 1 is offered by a degree 3 polynomial kernel (Figure 5a) that has seven support vectors, namely three from class +1 and four from class -1. The separation hyperplane becomes even more convoluted when a degree 10 polynomial kernel is used (Figure 5b). It is clear that this SVM model, with 10 support vectors (4 from class +1 and 6 from class -1), is not an optimal model for the dataset from Table 1.

The next two experiments were performed with the B spline kernel (Figure 6a) and the exponential radial basis function (RBF) kernel (Figure 6b). Both SVM models define elaborate hyperplanes, with a large number of support vectors (11 for spline, 14 for RBF). The SVM models obtained with the exponential RBF kernel acts almost like a look-up table, with all but one

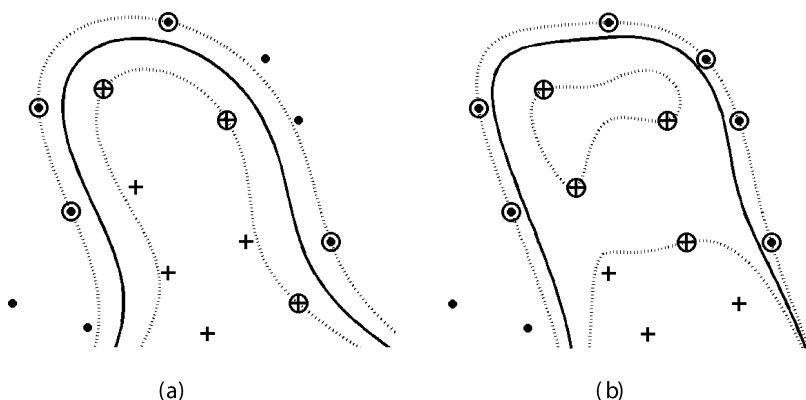


Figure 5 SVM classification models obtained with the polynomial kernel (Eq. [65]) for the dataset from Table 1: (a) polynomial of degree 3; (b) polynomial of degree 10.

pattern used as support vectors. By comparing the SVM models from Figures 4–6, it is clear that the best one is obtained with the degree 2 polynomial kernel, the simplest function that separates the two classes with the lowest number of support vectors. This principle of minimum complexity of the kernel function should serve as a guide for the comparative evaluation and selection of the best kernel. Like all other multivariate algorithms, SVM can overfit the data used in training, a problem that is more likely to happen when complex kernels are used to generate the SVM model.

Support vector machines were extended by Vapnik for regression⁴ by using an ϵ -insensitive loss function (Figure 7). The learning set of patterns is used to obtain a regression model that can be represented as a tube with radius ϵ fitted to the data. In the ideal case, SVM regression finds a function that maps

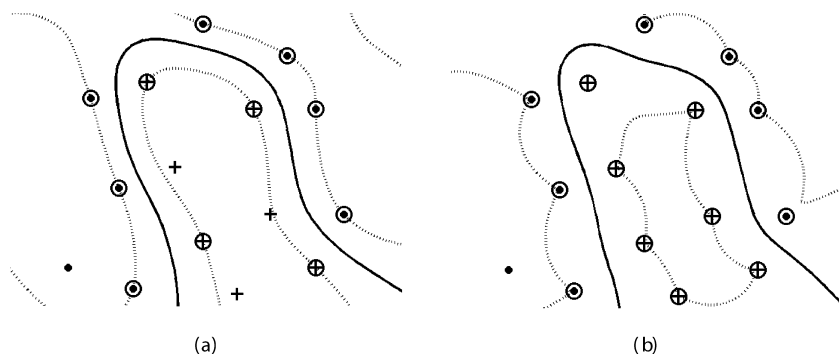


Figure 6 SVM classification models for the dataset from Table 1: (a) B spline kernel, degree 1, Eq. [72]; (b) exponential radial basis function kernel, $\sigma = 1$, Eq. [67].

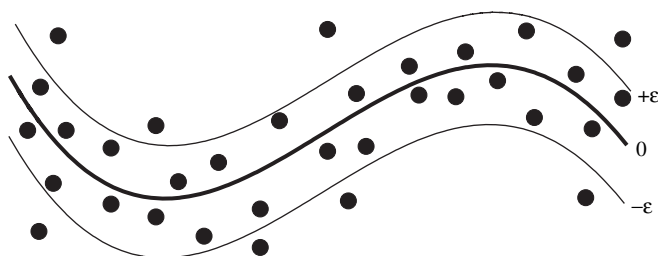


Figure 7 Support vector machines regression determines a tube with radius ε fitted to the data.

all input data with a maximum deviation ε from the target (experimental) values. In this case, all training points are located inside the regression tube. However, for datasets affected by errors, it is not possible to fit all the patterns inside the tube and still have a meaningful model. For the general case, SVM regression considers that the error for patterns inside the tube is zero, whereas patterns situated outside the regression tube have an error that increases when the distance to the tube margin increases (Figure 7).³⁰

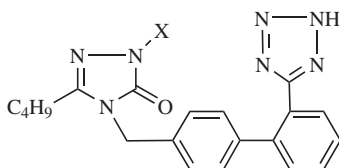
The SVM regression approach is illustrated with a QSAR for angiotensin II antagonists (Table 2) from a review by Hansch et al.³¹ This QSAR, modeling the IC_{50} for angiotensin II determined in rabbit aorta rings, is a nonlinear equation based on the hydrophobicity parameter ClogP:

$$\log 1/IC_{50} = 5.27(\pm 1.0) + 0.50(\pm 0.19)\text{ClogP} - 3.0(\pm 0.83)\log(\beta \times 10^{\text{ClogP}} + 1)$$

$$n = 16 \quad r_{\text{cal}}^2 = 0.849 \quad s_{\text{cal}} = 0.178 \quad q_{\text{LOO}}^2 = 0.793 \quad \text{opt.ClogP} = 6.42$$

We will use this dataset later to demonstrate the kernel influence on the SVM regression, as well as the effect of modifying the tube radius ε . However, we will not present QSAR statistics for the SVM model. Comparative QSAR models are shown in the section on SVM applications in chemistry.

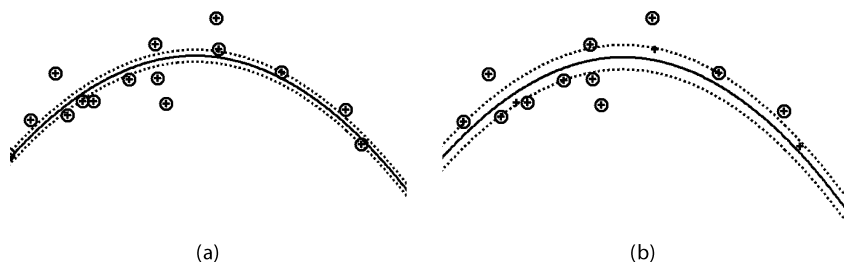
A linear function is clearly inadequate for the dataset from Table 2, so we will not present the SVMR model for the linear kernel. All SVM regression figures were prepared with the Gunn's MATLAB toolbox. Patterns are represented by +, and support vectors are represented as + inside a circle. The SVM hyperplane is drawn with a continuous line, whereas the margins of the SVM regression tube are represented by dotted lines. Several experiments with different kernels showed that the degree 2 polynomial kernel offers a good model for this dataset, and we decided to demonstrate the influence of the tube radius ε for this kernel (Figures 8 and 9). When the ε parameter is too small, the diameter of the tube is also small forcing all patterns to be situated outside the SVMR tube. In this case, all patterns are penalized with a value that increases when the distance from the tube's margin increases. This situation is demonstrated in Figure 8a generated with $\varepsilon = 0.05$, when all patterns are support

Table 2 Data for the Angiotensin II Antagonists QSAR³¹ and for the SVM Regression Models from Figures 8–11

No	Substituent X	ClogP	log 1/IC ₅₀
1	H	4.50	7.38
2	C ₂ H ₅	4.69	7.66
3	(CH ₂) ₂ CH ₃	5.22	7.82
4	(CH ₂) ₃ CH ₃	5.74	8.29
5	(CH ₂) ₄ CH ₃	6.27	8.25
6	(CH ₂) ₅ CH ₃	6.80	8.06
7	(CH ₂) ₇ CH ₃	7.86	6.77
8	CHMe ₂	5.00	7.70
9	CHMeCH ₂ CH ₃	5.52	8.00
10	CH ₂ CHMeCH ₂ CMe ₃	7.47	7.46
11	CH ₂ -cy-C ₃ H ₅	5.13	7.82
12	CH ₂ CH ₂ -cy-C ₆ H ₁₁	7.34	7.75
13	CH ₂ COOCH ₂ CH ₃	4.90	8.05
14	CH ₂ CO ₂ CMe ₃	5.83	7.80
15	(CH ₂) ₅ COOCH ₂ CH ₃	5.76	8.01
16	CH ₂ CH ₂ C ₆ H ₅	6.25	8.51

vectors. As ϵ increases to 0.1, the diameter of the tube increases and the number of support vector decreases to 12 (Figure 8b), whereas the remaining patterns are situated inside the tube and have zero error.

A further increase of ϵ to 0.3 results in a dramatic change in the number of support vectors, which decreases to 4 (Figure 9a), whereas an ϵ of 0.5, with two support vectors, gives an SVMR model with a decreased curvature

**Figure 8** SVM regression models with a degree 2 polynomial kernel (Eq. [65]) for the dataset from Table 2: (a) $\epsilon = 0.05$; (b) $\epsilon = 0.1$.

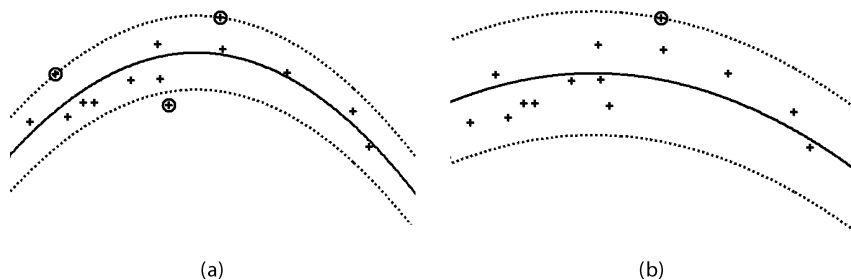


Figure 9 SVM regression models with a degree 2 polynomial kernel (Eq. [65]) for the dataset from Table 2: (a) $\varepsilon = 0.3$; (b) $\varepsilon = 0.5$.

(Figure 9b). These experiments illustrate the importance of the ε parameter on the SVMR model. Selection of the optimum value for ε should be determined by comparing the prediction statistics in cross-validation. The optimum value of ε depends on the experimental errors of the modeled property. A low ε should be used for low levels of noise, whereas higher values for ε are appropriate for large experimental errors. Note that a low ε results in SVMR models with a large number of support vectors, whereas sparse models are obtained with higher values for ε .

We will explore the possibility of overfitting in SVM regression when complex kernels are used to model the data, but first we must consider the limitations of the dataset in Table 2. This is important because those data might prevent us from obtaining a high-quality QSAR. First, the biological data are affected by experimental errors and we want to avoid modeling those errors (overfitting the model). Second, the influence of the substituent X is characterized with only its hydrophobicity parameter ClogP. Although hydrophobicity is important, as demonstrated in the QSAR model, it might be that other structural descriptors (electronic or steric) actually control the biological activity of this series of compounds. However, the small number of compounds and the limited diversity of the substituents in this dataset might not reveal the importance of those structural descriptors. Nonetheless, it follows that a predictive model should capture the nonlinear dependence between ClogP and $\log 1/IC_{50}$, and it should have a low degree of complexity to avoid modeling of the errors. The next two experiments were performed with the degree 10 polynomial kernel (Figure 10a; 12 support vectors) and the exponential RBF kernel with $\sigma = 1$ (Figure 10b; 11 support vectors). Both SVMR models, obtained with $\varepsilon = 0.1$, follow the data too closely and fail to recognize the general relationship between ClogP and $\log 1/IC_{50}$. The overfitting is more pronounced for the exponential RBF kernel, which therefore is not a good choice for this QSAR dataset.

Interesting results are also obtained with the spline kernel (Figure 11a) and the degree 1 B spline kernel (Figure 11b). The spline kernel offers an

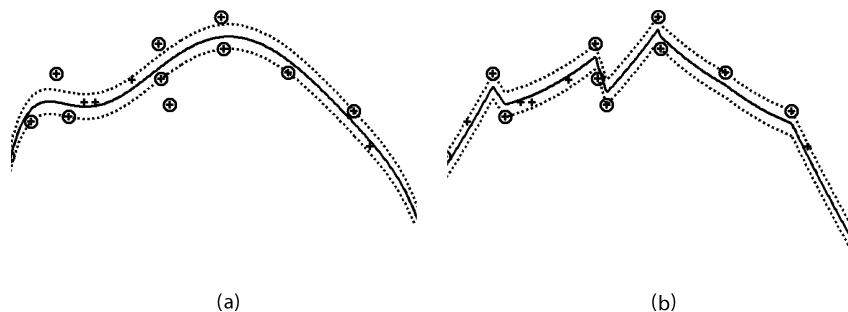


Figure 10 SVM regression models with $\varepsilon = 0.1$ for the dataset of Table 2: (a) polynomial kernel, degree 10, Eq. [65]; (b) exponential radial basis function kernel, $\sigma = 1$, Eq. [67].

interesting alternative to the SVMR model obtained with the degree 2 polynomial kernel. The tube is smooth, with a noticeable asymmetry, which might be supported by the experimental data, as one can deduce after a visual inspection. Together with the degree 2 polynomial kernel model, this spline kernel represents a viable QSAR model for this dataset. Of course, only detailed cross-validation and parameter tuning can decide which kernel is best. In contrast with the spline kernel, the degree 1 B spline kernel displays clear signs of overfitting, indicated by the complex regression tube. The hyperplane closely follows every pattern and is not able to extract a broad and simple relationship between ClogP and $\log 1/IC_{50}$.

The SVMR experiments that we have just carried out using the QSAR dataset from Table 2 offer convincing proof for the SVM ability to model nonlinear relationships but also their overfitting capabilities. This dataset was presented only for demonstrative purposes, and we do not recommend the use of SVM for QSAR models with such a low number of compounds and descriptors.

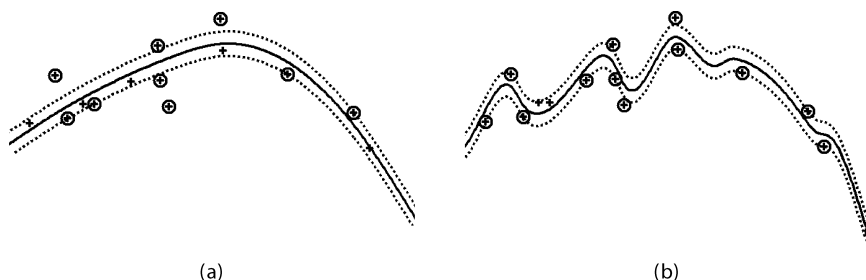


Figure 11 SVM regression models with $\varepsilon = 0.1$ for the dataset of Table 2: (a) spline kernel, Eq. [71]; (b) B spline kernel, degree 1, Eq. [72].

PATTERN CLASSIFICATION

Research in pattern recognition involves development and application of algorithms that can recognize patterns in data.³² These techniques have important applications in character recognition, speech analysis, image analysis, clinical diagnostics, person identification, machine diagnostics, and industrial process supervision as examples. Many chemistry problems can also be solved with pattern recognition techniques, such as recognizing the provenance of agricultural products (olive oil, wine, potatoes, honey, etc.) based on composition or spectra, structural elucidation from spectra, identifying mutagens or carcinogens from molecular structure, classification of aqueous pollutants based on their mechanism of action, discriminating chemical compounds based on their odor, and classification of chemicals in inhibitors and noninhibitors for a certain drug target.

We now introduce some basic notions of pattern recognition. A pattern (object) is any item (chemical compound, material, spectrum, physical object, chemical reaction, industrial process) whose important characteristics form a set of descriptors. A descriptor is a variable (usually numerical) that characterizes an object. Note that in pattern recognition, descriptors are usually called “features”, but in SVM, “features” have another meaning, so we must make a clear distinction here between “descriptors” and “features”. A descriptor can be any experimentally measured or theoretically computed quantity that describes the structure of a pattern, including, for example, spectra and composition for chemicals, agricultural products, materials, biological samples; graph descriptors³³ and topological indices;³⁴ indices derived from the molecular geometry and quantum calculations;^{35,36} industrial process parameters; chemical reaction variables; microarray gene expression data; and mass spectrometry data for proteomics.

Each pattern (object) has associated with it a property value. A property is an attribute of a pattern that is difficult, expensive, or time-consuming to measure, or not even directly measurable. Examples of such properties include concentration of a compound in a biological sample, material, or agricultural product; various physical, chemical, or biological properties of chemical compounds; biological toxicity, mutagenicity, or carcinogenicity; ligand/nonligand for different biological receptors; and fault identification in industrial processes.

The major hypothesis used in pattern recognition is that the descriptors capture some important characteristics of the pattern, and then a mathematical function (e.g., machine learning algorithm) can generate a mapping (relationship) between the descriptor space and the property. Another hypothesis is that similar objects (objects that are close in the descriptor space) have similar properties. A wide range of pattern recognition algorithms are currently being used to solve chemical problems. These methods include linear discriminant analysis, principal component analysis, partial least squares (PLS),³⁷ artificial

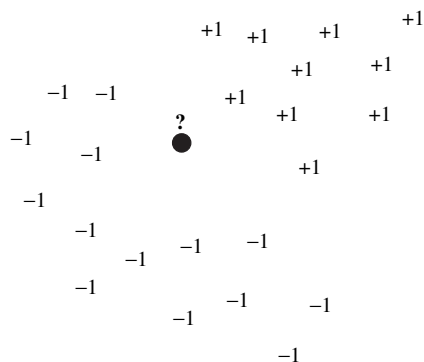


Figure 12 Example of a classification problem.

neural networks,³⁸ multiple linear regression (MLR), principal component regression, k -nearest neighbors (k -NN), evolutionary algorithms embedded into machine learning procedures,³⁹ and large margin classifiers including, of course, support vector machines.

A simple example of a classification problem is presented in Figure 12. The learning set consists of 24 patterns, 10 in class +1 and 14 in class -1. In the learning (training) phase, the algorithm extracts classification rules using the information available in the learning set. In the prediction phase, the classification rules are applied to new patterns, with unknown class membership, and each new pattern is assigned to a class, either +1 or -1. In Figure 12, the prediction pattern is indicated with “?”.

We consider first a k -NN classifier, with $k = 1$. This algorithm computes the distance between the new pattern and all patterns in the training set, and then it identifies the k patterns closest to the new pattern. The new pattern is assigned to the majority class of the k nearest neighbors. Obviously, k should be odd to avoid undecided situations. The k -NN classifier assigns the new pattern to class +1 (Figure 13) because its closest pattern belongs to this class. The predicted class of a new pattern can change by changing the parameter k . The optimal value for k is usually determined by cross-validation.

The second classifier considered here is a hyperplane H that defines two regions, one for patterns +1 and the other for patterns -1. New patterns are assigned to class +1 if they are situated in the space region corresponding to the class +1, but to class -1 if they are situated in the region corresponding to class -1. For example, the hyperplane H in Figure 14 assigns the new pattern to class -1. The approach of these two algorithms is very different: although the k -NN classifier memorizes all patterns, the hyperplane classifier is defined by the equation of a plane in the pattern space. The hyperplane can be used only for linearly separable classes, whereas k -NN is a nonlinear classifier and can be used for classes that cannot be separated with a linear hypersurface.

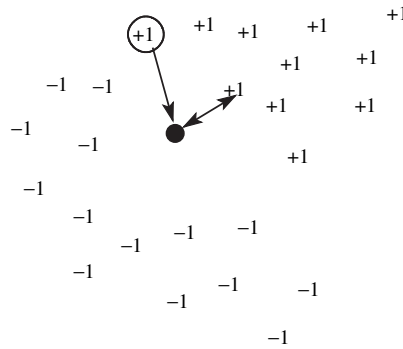


Figure 13 Using the k -NN classifier ($k = 1$), the pattern \bullet is predicted to belong to the class $+1$.

An n -dimensional pattern (object) \mathbf{x} has n coordinates, $\mathbf{x} = (x_1, x_2, \dots, x_n)$, where each x_i is a real number, $x_i \in R$ for $i = 1, 2, \dots, n$. Each pattern \mathbf{x}_i belongs to a class $y_i \in \{-1, +1\}$. Consider a training set T of m patterns together with their classes, $T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$. Consider a dot product space S , in which the patterns \mathbf{x} are embedded, $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m \in S$. Any hyperplane in the space S can be written as

$$\{\mathbf{x} \in S | \mathbf{w} \cdot \mathbf{x} + b = 0\}, \mathbf{w} \in S, b \in R \tag{1}$$

The dot product $\mathbf{w} \cdot \mathbf{x}$ is defined by

$$\mathbf{w} \cdot \mathbf{x} = \sum_{i=1}^n w_i x_i \tag{2}$$

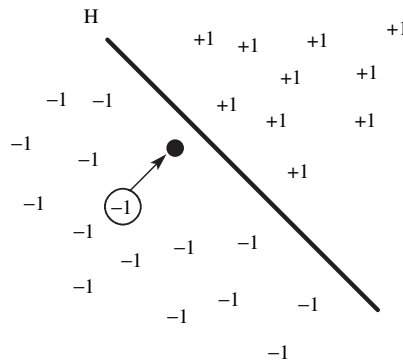


Figure 14 Using the linear classifier defined by the hyperplane H , the pattern \bullet is predicted to belong to the class -1 .

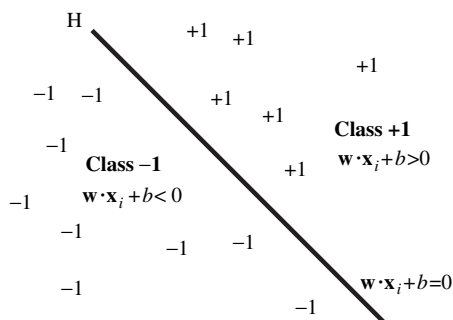


Figure 15 The classification hyperplane defines a region for class +1 and another region for class -1.

A hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$ can be denoted as a pair (\mathbf{w}, b) . A training set of patterns is linearly separable if at least one linear classifier exists defined by the pair (\mathbf{w}, b) , which correctly classifies all training patterns (see Figure 15). All patterns from class +1 are located in the space region defined by $\mathbf{w} \cdot \mathbf{x} + b > 0$, and all patterns from class -1 are located in the space region defined by $\mathbf{w} \cdot \mathbf{x} + b < 0$. Using the linear classifier defined by the pair (\mathbf{w}, b) , the class of a pattern \mathbf{x}_k is determined with

$$\text{class}(\mathbf{x}_k) = \begin{cases} +1 & \text{if } \mathbf{w} \cdot \mathbf{x}_k + b > 0 \\ -1 & \text{if } \mathbf{w} \cdot \mathbf{x}_k + b < 0 \end{cases} \quad [3]$$

The distance from a point \mathbf{x} to the hyperplane defined by (\mathbf{w}, b) is

$$d(\mathbf{x}; \mathbf{w}, b) = \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\|\mathbf{w}\|} \quad [4]$$

where $\|\mathbf{w}\|$ is the norm of the vector \mathbf{w} .

Of all the points on the hyperplane, one has the minimum distance d_{\min} to the origin (Figure 16):

$$d_{\min} = \frac{|b|}{\|\mathbf{w}\|} \quad [5]$$

In Figure 16, we show a linear classifier (hyperplane H defined by $\mathbf{w} \cdot \mathbf{x} + b = 0$), the space region for class +1 patterns (defined by $\mathbf{w} \cdot \mathbf{x} + b > 0$), the space region for class -1 patterns (defined by $\mathbf{w} \cdot \mathbf{x} + b < 0$), and the distance between origin and the hyperplane H ($|b|/\|\mathbf{w}\|$).

Consider a group of linear classifiers (hyperplanes) defined by a set of pairs (\mathbf{w}, b) that satisfy the following inequalities for any pattern \mathbf{x}_i in the training set:

$$\begin{cases} \mathbf{w} \cdot \mathbf{x}_i + b > 0 & \text{if } y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b < 0 & \text{if } y_i = -1 \end{cases} \quad [6]$$

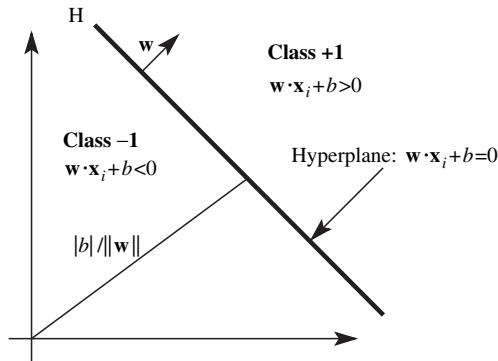


Figure 16 The distance from the hyperplane to the origin.

This group of (\mathbf{w}, b) pairs defines a set of classifiers that are able to make a complete separation between two classes of patterns. This situation is illustrated in Figure 17.

In general, for each linearly separable training set, one can find an infinite number of hyperplanes that discriminate the two classes of patterns. Although all these linear classifiers can perfectly separate the learning patterns, they are not all identical. Indeed, their prediction capabilities are different. A hyperplane situated in the proximity of the border +1 patterns will predict as -1 all new +1 patterns that are situated close to the separation hyperplane but in the -1 region ($\mathbf{w} \cdot \mathbf{x} + b < 0$). Conversely, a hyperplane situated in the proximity of the border -1 patterns will predict as +1 all new -1 patterns situated close to the separation hyperplane but in the +1 region ($\mathbf{w} \cdot \mathbf{x} + b > 0$). It is clear that such classifiers have little prediction success, which led to the idea

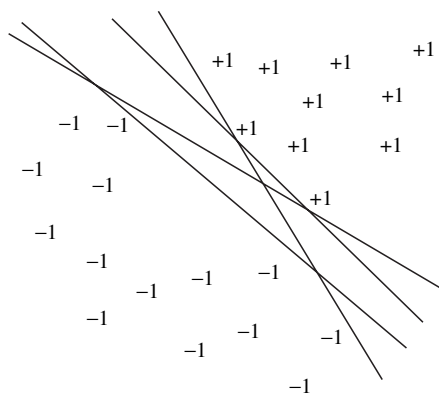


Figure 17 Several hyperplanes that correctly classify the two classes of patterns.

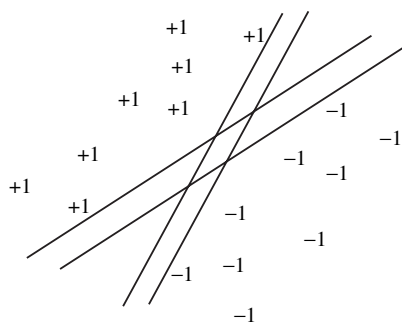


Figure 18 Examples of margin hyperplane classifiers.

of wide margin classifiers, i.e., a hyperplane with a buffer toward the +1 and -1 space regions (Figure 18).

For some linearly separable classification problems having a finite number of patterns, it is generally possible to define a large number of wide margin classifiers (Figure 18). Chemometrics and pattern recognition applications suggest that an optimum prediction could be obtained with a linear classifier that has a maximum margin (separation between the two classes), and with the separation hyperplane being equidistant from the two classes. In the next section, we introduce elements of statistical learning theory that form the basis of support vector machines, followed by a section on linear support vector machines in which the mathematical basis for computing a maximum margin classifier with SVM is presented.

THE VAPNIK-CHERVONENKIS DIMENSION

Support vector machines are based on the structural risk minimization (SRM), derived from statistical learning theory.^{4,5,10} This theory is the basis for finding bounds for the classification performance of machine learning algorithms. Another important result from statistical learning theory is the performance estimation of finite set classifiers and the convergence of their classification performance toward that of a classifier with an infinite number of learning samples. Consider a learning set of m patterns. Each pattern consists of a vector of characteristics $\mathbf{x}_i \in R^n$ and an associated class membership y_i . The task of the machine learning algorithm is to find the rules of the mapping $\mathbf{x}_i \rightarrow y_i$. The machine model is a possible mapping $\mathbf{x}_i \rightarrow f(\mathbf{x}_i, \pi)$, where each model is defined by a set of parameters π . Training a machine learning algorithm results in finding an optimum set of parameters π . The machine algorithm is considered to be deterministic; i.e., for a given input vector \mathbf{x}_i and a set of parameters π , the output will be always $f(\mathbf{x}_i, \pi)$. The expectation for the test error of a machine trained

with an infinite number of samples is denoted by $\varepsilon(\pi)$ (called expected risk or expected error). The empirical risk $\varepsilon_{\text{emp}}(\pi)$ is the measured error for a finite number of patterns in the training set:

$$\varepsilon_{\text{emp}}(\pi) = \frac{1}{2m} \sum_{i=1}^m |y_i - f(\mathbf{x}_i, \pi)| \tag{7}$$

The quantity $\frac{1}{2}|y_i - f(\mathbf{x}_i, \pi)|$ is called the loss, and for a two-class classification, it can take only the values 0 and 1. Choose a value η such that $0 \leq \eta \leq 1$. For losses taking these values, with probability $1 - \eta$, the following bound exists for the expected risk:

$$\varepsilon(\pi) \leq \varepsilon_{\text{emp}}(\pi) + \sqrt{\frac{d_{\text{VC}}(\log(2m/d_{\text{VC}}) + 1) - \log(\eta/4)}{m}} \tag{8}$$

where d_{VC} is a non-negative integer, called the Vapnik–Chervonenkis (VC) dimension of a classifier, that measures the capacity of a classifier. The right-hand side of this equation defines the risk bound. The second term in the right-hand side of the equation is called VC confidence.

We consider the case of two-class pattern recognition, when the function $f(\mathbf{x}_i, \pi)$ can take only two values, e.g., +1 and -1. Consider a set of m points and all their two-class labelings. If for each of the 2^m labelings one can find a classifier $f(\pi)$ that correctly separates class +1 points from class -1 points, then that set of points is separated by that set of functions. The VC dimension for a set of functions $\{f(\pi)\}$ is defined as the maximum number of points that can be separated by $\{f(\pi)\}$. In two dimensions, three samples can be separated with a line for each of the six possible combinations (Figure 19, top panels). In the case of four training points in a plane, there are two cases that cannot be separated with a line (Figure 19, bottom panels). These two cases require a classifier of higher complexity, with a higher VC dimension. The example

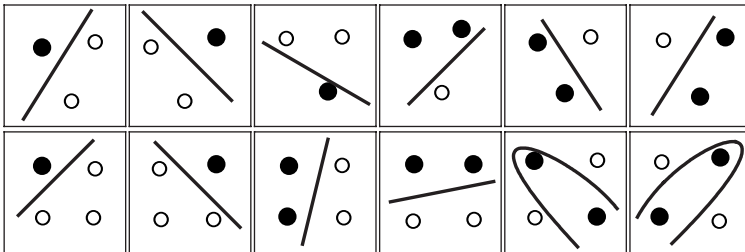


Figure 19 In a plane, all combinations of three points from two classes can be separated with a line. Four points cannot be separated with a linear classifier.

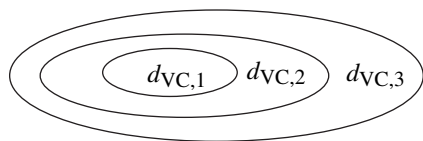


Figure 20 Nested subsets of function, ordered by VC dimension.

from Figure 19 shows that the VC dimension of a set of lines in R^2 is three. A family of classifiers has an infinite VC dimension if it can separate m points, with m being arbitrarily large.

The VC confidence term in Eq. [8] depends on the chosen class of functions, whereas the empirical risk and the actual risk depend on the particular function obtained from the training algorithm.^{2,3} It is important to find a subset of the selected set of functions such that the risk bound for that subset is minimized. A structure is introduced by classifying the whole class of functions into nested subsets (Figure 20), with the property $d_{VC,1} < d_{VC,2} < d_{VC,3}$. For each subset of functions, it is either possible to compute d_{VC} or to get a bound on the VC dimension. Structural risk minimization consists of finding the subset of functions that minimizes the bound on the actual risk. This is done by training for each subset a machine model. For each model the goal is to minimize the empirical risk. Subsequently, one selects the machine model whose sum of empirical risk and VC confidence is minimal.

PATTERN CLASSIFICATION WITH LINEAR SUPPORT VECTOR MACHINES

To apply the results from the statistical learning theory to pattern classification one has to (1) choose a classifier with the smallest empirical risk and (2) choose a classifier from a family that has the smallest VC dimension. For a linearly separable case condition, (1) is satisfied by selecting any classifier that completely separates both classes (for example, any classifier from Figure 17), whereas condition (2) is satisfied for the classifier with the largest margin.

SVM Classification for Linearly Separable Data

The optimum separation hyperplane (OSH) is the hyperplane with the maximum margin for a given finite set of learning patterns. The OSH computation with a linear support vector machine is presented in this section.

The Optimization Problem

Based on the notations from Figure 21, we will now establish the conditions necessary to determine the maximum separation hyperplane. Consider a

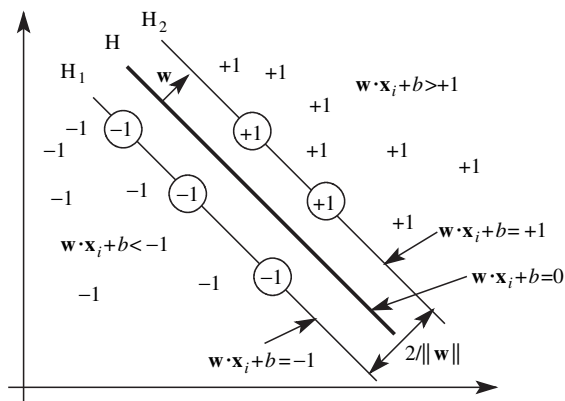


Figure 21 The separating hyperplane.

linear classifier characterized by the set of pairs (\mathbf{w}, b) that satisfy the following inequalities for any pattern \mathbf{x}_i in the training set:

$$\begin{cases} \mathbf{w} \cdot \mathbf{x}_i + b > +1 & \text{if } y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b < -1 & \text{if } y_i = -1 \end{cases} \quad [9]$$

These equations can be expressed in compact form as

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq +1 \quad [10]$$

or

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0 \quad [11]$$

Because we have considered the case of linearly separable classes, each such hyperplane (\mathbf{w}, b) is a classifier that correctly separates all patterns from the training set:

$$\text{class}(\mathbf{x}_i) = \begin{cases} +1 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b > 0 \\ -1 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b < 0 \end{cases} \quad [12]$$

For the hyperplane H that defines the linear classifier (i.e., where $\mathbf{w} \cdot \mathbf{x} + b = 0$), the distance between the origin and the hyperplane H is $|b|/\|\mathbf{w}\|$. We consider the patterns from the class -1 that satisfy the equality $\mathbf{w} \cdot \mathbf{x} + b = -1$ and that determine the hyperplane H_1 ; the distance between the origin and the hyperplane H_1 is equal to $|-1 - b|/\|\mathbf{w}\|$. Similarly, the patterns from the class $+1$ satisfy the equality $\mathbf{w} \cdot \mathbf{x} + b = +1$ and that determine

the hyperplane H_2 ; the distance between the origin and the hyperplane H_2 is equal to $|+1 - b|/\|\mathbf{w}\|$. Of course, hyperplanes H , H_1 , and H_2 are parallel and no training patterns are located between hyperplanes H_1 and H_2 . Based on the above considerations, the margin of the linear classifier H (the distance between hyperplanes H_1 and H_2) is $2/\|\mathbf{w}\|$.

We now present an alternative method to determine the distance between hyperplanes H_1 and H_2 . Consider a point \mathbf{x}_0 located on the hyperplane H and a point \mathbf{x}_1 located on the hyperplane H_1 , selected in such a way that $(\mathbf{x}_0 - \mathbf{x}_1)$ is orthogonal to the two hyperplanes. These points satisfy the following two equalities:

$$\begin{cases} \mathbf{w} \cdot \mathbf{x}_0 + b = 0 \\ \mathbf{w} \cdot \mathbf{x}_1 + b = -1 \end{cases} \quad [13]$$

By subtracting the second equality from the first equality, we obtain

$$\mathbf{w} \cdot (\mathbf{x}_0 - \mathbf{x}_1) = 1 \quad [14]$$

Because $(\mathbf{x}_0 - \mathbf{x}_1)$ is orthogonal to the hyperplane H , and \mathbf{w} is also orthogonal to H , then $(\mathbf{x}_0 - \mathbf{x}_1)$ and \mathbf{w} are parallel, and the dot product satisfies

$$|\mathbf{w} \cdot (\mathbf{x}_0 - \mathbf{x}_1)| = \|\mathbf{w}\| \times \|\mathbf{x}_0 - \mathbf{x}_1\| \quad [15]$$

From Eqs. [14] and [15], we obtain the distance between hyperplanes H and H_1 :

$$\|\mathbf{x}_0 - \mathbf{x}_1\| = \frac{1}{\|\mathbf{w}\|} \quad [16]$$

Similarly, a point \mathbf{x}_0 located on the hyperplane H and a point \mathbf{x}_2 located on the hyperplane H_2 , selected in such a way that $(\mathbf{x}_0 - \mathbf{x}_2)$ is orthogonal to the two hyperplanes, will satisfy the equalities:

$$\begin{cases} \mathbf{w} \cdot \mathbf{x}_0 + b = 0 \\ \mathbf{w} \cdot \mathbf{x}_2 + b = +1 \end{cases} \quad [17]$$

Consequently, the distance between hyperplanes H and H_2 is

$$\|\mathbf{x}_0 - \mathbf{x}_2\| = \frac{1}{\|\mathbf{w}\|} \quad [18]$$

Therefore, the margin of the linear classifier defined by (\mathbf{w}, b) is $2/\|\mathbf{w}\|$. The wider the margin, the smaller is d_{VC} , the VC dimension of the classifier. From

these considerations, it follows that the optimum separation hyperplane is obtained by maximizing $2/\|\mathbf{w}\|$, which is equivalent to minimizing $\|\mathbf{w}\|^2/2$.

The problem of finding the optimum separation hyperplane is represented by the identification of the linear classifier (\mathbf{w}, b) , which satisfies

$$\begin{cases} \mathbf{w} \cdot \mathbf{x}_i + b \geq +1 & \text{if } y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b \leq -1 & \text{if } y_i = -1 \end{cases} \quad [19]$$

for which $\|\mathbf{w}\|$ has the minimum value.

Computing the Optimum Separation Hyperplane

Based on the considerations presented above, the OSH conditions from Eq. [19] can be formulated into the following expression that represents a linear SVM:

$$\text{minimize } f(\mathbf{x}) = \frac{\|\mathbf{w}\|^2}{2} \quad [20]$$

with the constraints $g_i(\mathbf{x}) = y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0, i = 1, \dots, m$

The optimization problem from Eq. [20] represents the minimization of a quadratic function under linear constraints (quadratic programming), a problem studied extensively in optimization theory. Details on quadratic programming can be found in almost any textbook on numerical optimization, and efficient implementations exist in many software libraries. However, Eq. [20] does not represent the actual optimization problem that is solved to determine the OSH. Based on the use of a Lagrange function, Eq. [20] is transformed into its dual formulation. All SVM models (linear and nonlinear, classification and regression) are solved for the dual formulation, which has important advantages over the primal formulation (Eq. [20]). The dual problem can be easily generalized to linearly nonseparable learning data and to nonlinear support vector machines.

A convenient way to solve constrained minimization problems is by using a Lagrangian function of the problem defined in Eq. [20]:

$$\begin{aligned} L_P(\mathbf{w}, b, \Lambda) &= f(\mathbf{x}) + \sum_{i=0}^m \lambda_i g_i(\mathbf{x}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \lambda_i (y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1) \\ &= \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \lambda_i y_i (\mathbf{w} \cdot \mathbf{x}_i + b) + \sum_{i=1}^m \lambda_i = \frac{1}{2} \|\mathbf{w}\|^2 \\ &\quad - \sum_{i=1}^m \lambda_i y_i \mathbf{w} \cdot \mathbf{x}_i - \sum_{i=1}^m \lambda_i y_i b + \sum_{i=1}^m \lambda_i \end{aligned} \quad [21]$$

Here $\Lambda = (\lambda_1, \lambda_2, \dots, \lambda_m)$ is the set of Lagrange multipliers of the training (calibration) patterns with $\lambda_i \geq 0$, and P in L_P indicates the primal

formulation of the problem. The Lagrangian function L_P must be minimized with respect to \mathbf{w} and b , and maximized with respect to λ_i , subject to the constraints $\lambda_i \geq 0$. This is equivalent to solving the Wolfe dual problem,⁴⁰ namely to maximize L_P subject to the constraints that the gradient of L_P with respect to \mathbf{w} and b is zero, and subject to the constraints $\lambda_i \geq 0$.

The Karuch–Kuhn–Tucker (KKT)⁴⁰ conditions for the primal problem are as follows:

Gradient Conditions

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i = 0, \text{ where } \frac{\partial L_P(\mathbf{w}, b, \Lambda)}{\partial \mathbf{w}} = \left(\frac{\partial L}{\partial w_1}, \frac{\partial L}{\partial w_2}, \dots, \frac{\partial L}{\partial w_n} \right) \quad [22]$$

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda)}{\partial b} = \sum_{i=1}^m \lambda_i y_i = 0 \quad [23]$$

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda)}{\partial \lambda_i} = g_i(\mathbf{x}) = 0 \quad [24]$$

Orthogonality Condition

$$\lambda_i g_i(\mathbf{x}) = \lambda_i [y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1] = 0, \quad i = 1, \dots, m \quad [25]$$

Feasibility Condition

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0, \quad i = 1, \dots, m \quad [26]$$

Non-negativity Condition

$$\lambda_i \geq 0, \quad i = 1, \dots, m \quad [27]$$

Solving the SVM problem is equivalent to finding a solution to the KKT conditions. We are now ready to formulate the dual problem L_D :

$$\begin{aligned} \text{maximize } L_D(\mathbf{w}, b, \Lambda) &= \sum_{i=1}^m \lambda_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\ \text{subject to } \lambda_i &\geq 0, \quad i = 1, \dots, m \\ \text{and } \sum_{i=1}^m \lambda_i y_i &= 0 \end{aligned} \quad [28]$$

Both the primal L_P and the dual L_D Lagrangian functions are derived from the same objective functions but with different constraints, and the solution is

found by minimizing L_P or by maximizing L_D . The most popular algorithm for solving the optimization problem is the sequential minimal optimization (SMO) proposed by Platt.⁴¹

When we introduced the Lagrange function we assigned a Lagrange multiplier λ_i to each training pattern via the constraints $g_i(\mathbf{x})$ (see Eq. [20]). The training patterns from the SVM solution that have $\lambda_i > 0$ represent the support vectors. The training patterns that have $\lambda_i = 0$ are not important in obtaining the SVM model, and they can be removed from training without any effect on the SVM solution. As we will see below, any SVM model is completely defined by the set of support vectors and the corresponding Lagrange multipliers.

The vector \mathbf{w} that defines the OSH (Eq. [29]) is obtained by using Eq. [22]:

$$\mathbf{w} = \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \tag{29}$$

To compute the threshold b of the OSH, we consider the KKT condition of Eq. [25] coupled with the expression for \mathbf{w} from Eq. [29] and the condition $\lambda_j > 0$, which leads to

$$\sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x}_j + b = y_j \tag{30}$$

Therefore, the threshold b can be obtained by averaging the b values obtained for all support vector patterns, i.e., the patterns with $\lambda_j > 0$:

$$b = y_j - \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x}_j \tag{31}$$

Prediction for New Patterns

In the previous section, we presented the SVM algorithm for training a linear classifier. The result of this training is an optimum separation hyper-plane defined by (\mathbf{w}, b) (Eqs. [29] and [31]). After training, the classifier is ready to predict the class membership for new patterns, different from those used in training. The class of a pattern \mathbf{x}_k is determined with

$$\text{class}(\mathbf{x}_k) = \begin{cases} +1 & \text{if } \mathbf{w} \cdot \mathbf{x}_k + b > 0 \\ -1 & \text{if } \mathbf{w} \cdot \mathbf{x}_k + b < 0 \end{cases} \tag{32}$$

Therefore, the classification of new patterns depends only on the sign of the expression $\mathbf{w} \cdot \mathbf{x} + b$. However, Eq. [29] offers the possibility to predict new

patterns without computing the vector \mathbf{w} explicitly. In this case, we will use for classification the support vectors from the training set and the corresponding values of the Lagrange multipliers λ_i :

$$\text{class}(\mathbf{x}_k) = \text{sign} \left(\sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x}_k + b \right) \quad [33]$$

Patterns that are not support vectors ($\lambda_i = 0$) do not influence the classification of new patterns. The use of Eq. [33] has an important advantage over using Eq. [32]: to classify a new pattern \mathbf{x}_k , it is only necessary to compute the dot product between \mathbf{x}_k and every support vector. This results in a significant saving of computational time whenever the number of support vectors is small compared with the total number of patterns in the training set. Also, Eq. [33] can be easily adapted for nonlinear classifiers that use kernels, as we will show later.

For a particular SVM problem (training set, kernel, kernel parameters), the optimum separation hyperplane is determined only by the support vectors (Figure 22a). By eliminating from training those patterns that are not support vectors ($\lambda_i = 0$), the SVM solution does not change (Figure 22b). This property suggests a possible approach for accelerating the SVM learning phase, in which patterns that cannot be support vectors are eliminated from learning.

Example of SVM Classification for Linearly Separable Data

We now present several SVM classification experiments for a dataset that is linearly separable (Table 3). This exercise is meant to compare the linear kernel with nonlinear kernels and to compare different topologies for the separating hyperplanes. All models used an infinite value for the capacity parameter C (no tolerance for misclassified patterns; see Eq. [39]).

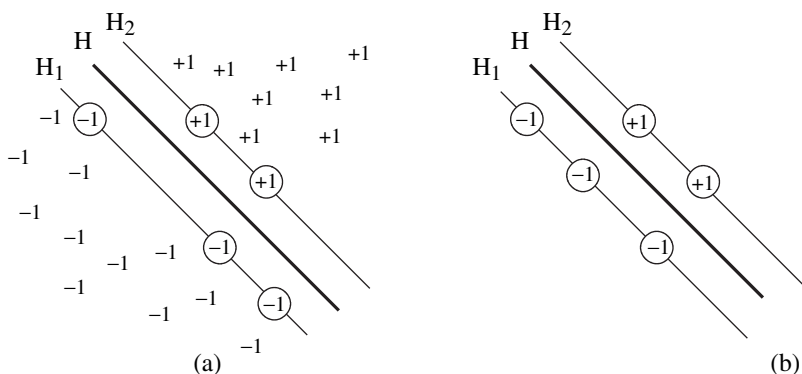


Figure 22 The optimal hyperplane classifier obtained with all training patterns (a) is identical with the one computed with only the support vector patterns (b).

Table 3 Linearly Separable Patterns Used for the SVM Classification Models in Figures 23–25

Pattern	x_1	x_2	Class
1	1	5.5	1
2	2.25	5	1
3	3.25	4.25	1
4	4	5.2	1
5	5.25	2.25	1
6	5.5	4	1
7	0.5	3.5	-1
8	1	2	-1
9	1.5	1	-1
10	2.25	2.7	-1
11	3	0.8	-1
12	3.75	1.25	-1
13	5	0.6	-1

As expected, a linear kernel offers a complete separation of the two classes (Figure 23a), with only three support vectors, namely one from class +1 and two from class -1. The hyperplane has the maximum width and provides both a sparse solution and a good prediction model for new patterns. Note that, according to the constraints imposed in generating this SVMC model, no patterns are allowed inside the margins of the classifier (margins defined by the two bordering hyperplanes represented with dotted lines). To predict the class attribution for new patterns, one uses Eq. [33] applied to the three support vectors. The next experiment uses a degree 2 polynomial kernel (Figure 23b), which gives a solution with five support vectors, namely two from class +1 and three from class -1. The model is not optimal for this dataset, but it still provides an acceptable hyperplane

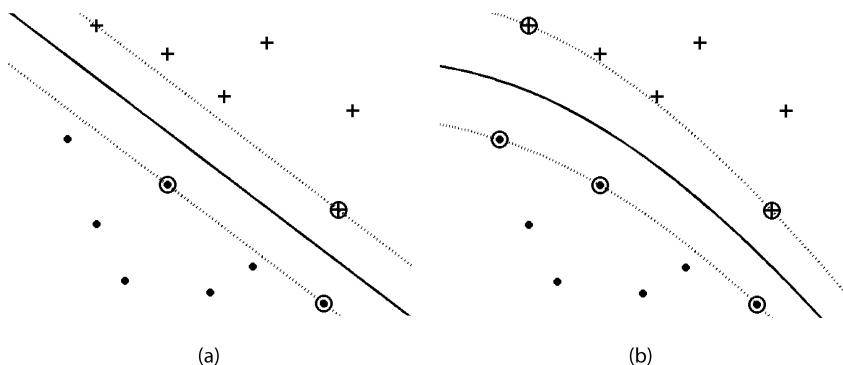


Figure 23 SVM classification models for the dataset from Table 3: (a) dot kernel (linear), Eq. [64]; (b) polynomial kernel, degree 2, Eq. [65].

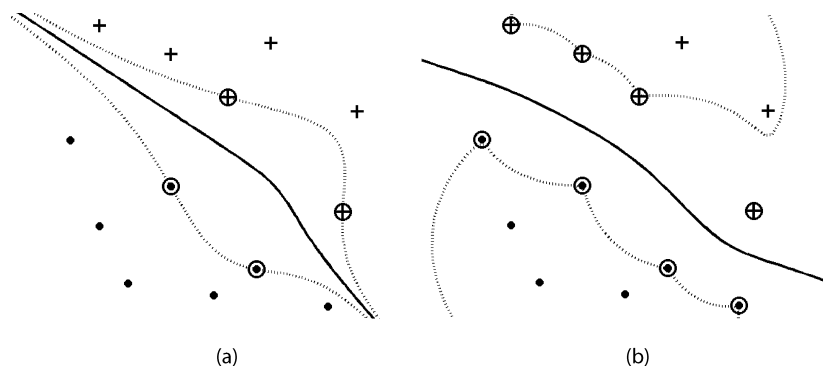


Figure 24 SVM classification models for the dataset from Table 3: (a) polynomial kernel, degree 10, Eq. [65]; (b) exponential radial basis function kernel, $\sigma = 1$, Eq. [67].

topology. We have to notice that the margin width varies, decreasing from left to right.

By increasing the polynomial degree to 10, we obtain an SVM model that has a wide margin in the center of the separating hyperplane and a very small margin toward the two ends (Figure 24a). Four patterns are selected as support vectors, two from each class. This is not a suitable classifier for the dataset from Table 3, mainly because the topology of the separating hypersurface is too complicated. An even more complex discriminating hyperplane is produced by the exponential RBF kernel (Figure 24b).

The last two experiments for the linearly separable dataset are performed with the Gaussian RBF kernel ($\sigma = 1$; Figure 25a) and the B spline kernel (degree 1; Figure 25b). Although not optimal, the classification hyperplane for the Gaussian RBF kernel is much better than those obtained with the exponential RBF kernel and degree 10 polynomial kernel. On the other hand, SVM

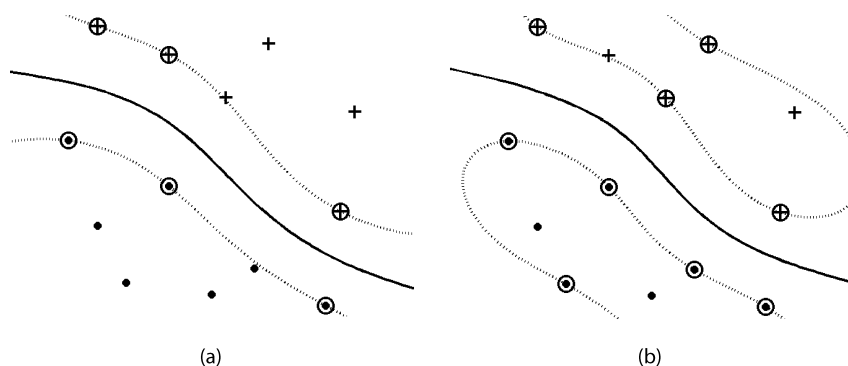


Figure 25 SVM classification models for the dataset from Table 3: (a) Gaussian radial basis function kernel, $\sigma = 1$, Eq. [66]; (b) B spline kernel, degree 1, Eq. [72].

with the B spline kernel is clearly overfitted, with a total of nine support vectors (four from class +1 and five from class -1). The margins of the SVM classifier define two “islands” that surround each cluster of patterns. Noticeable are the support vectors situated far away from the central hyperplane.

The SVM classification models depicted in Figures 23–25 convey an important message for scientists who want to use SVM applications in cheminformatics: SVM models obtained with complex, nonlinear kernels must always be compared with those obtained with a linear kernel. Chances are that the separation hypersurface is almost linear, thus avoiding overfitting the data.

Linear SVM for the Classification of Linearly Non-Separable Data

In the previous section, we presented the SVMC model for the case when the training set is linearly separable, and an optimum separation hyperplane correctly classifies all patterns from that training set. The linear separability of two classes of patterns might not be a valid assumption for real-life applications, however, and in these cases, the algorithm presented earlier will not find a solution. There are many reasons why a training set is linearly nonseparable. The identification of input variables (x_1, x_2, \dots, x_n) that can separate the two classes linearly is not a trivial task. When descriptors are used for SAR models, the selection of those descriptors can be made from thousands of descriptors from the extant literature or they can be computed with available software. Although several procedures have been developed to select the optimum set of structural descriptors, these methods are often time-consuming and may require special algorithms that are not implemented in, e.g., currently available SVM packages. In chemometrics applications, when measured quantities (e.g., spectra, physico-chemical properties, chemical reaction variables, or industrial process variables) are used to separate two classes of patterns, difficulties exist not only for identifying the relevant properties, but also for cost and instrument availability, which may limit the number of possible measurements. Also, all experimental input data are affected by measurement errors and noise, which can make the patterns linearly nonseparable. Finally, the classes might not be separable with a linear classifier, due to the nonlinear mapping between the input space and the two classes.

In Figure 26, we present a classification problem that, for the majority of patterns, can be solved with a linear classifier. However, the region corresponding to the +1 patterns contains two -1 patterns (shown in square boxes), whereas the two +1 patterns are embedded in the region corresponding to the -1 patterns. Of course, no linear classifier can be computed for this learning set, but several hyperplanes can be calculated in such a way as to minimize the number of classification errors, e.g., hyperplane H in Figure 26.

In this section, we consider a training set T of m patterns together with their classes, $T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$ that can be separated

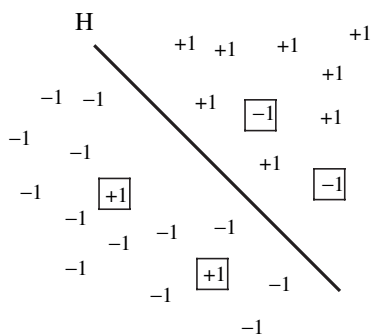


Figure 26 Linearly nonseparable data. The patterns that cannot be linearly separated with a hyperplane are represented inside a square.

linearly, except for a small number of objects. Obviously, computing the optimum separation hyperplane according to Eqs. [21] and [28] will fail to produce any viable solution. We will show below how the SVMC for linearly separable patterns can be adapted to accommodate classification errors in the training set. The resulting SVMC will still be linear, but it will compute an optimum separation hyperplane even for cases like that in Figure 26, which cannot be completely separated with a linear classifier.

In the previous section, we found that the OSH defined by a pair (\mathbf{w}, b) is a buffer between class +1 and class -1 of patterns, with the property that it has the largest margin. The border toward the class +1 is defined by the hyperplane $\mathbf{w} \cdot \mathbf{x} + b = -1$, whereas the border toward the class -1 is defined by the hyperplane $\mathbf{w} \cdot \mathbf{x} + b = +1$. For the OSH, all class +1 patterns satisfy $\mathbf{w} \cdot \mathbf{x} + b \geq +1$, whereas all class -1 patterns satisfy $\mathbf{w} \cdot \mathbf{x} + b \leq -1$, and the learning set is classified without errors.

To obtain an optimum linear classifier for nonseparable data (Figure 27), a penalty is introduced for misclassified data, denoted with ξ and called a slack

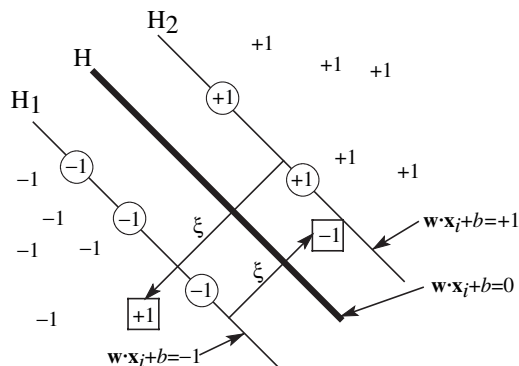


Figure 27 Linear separable hyperplanes for nonseparable data. The patterns that cannot be linearly separated with a hyperplane are represented inside a square.

variable. This penalty associated with any pattern in the training is zero for patterns classified correctly, and has a positive value that increases with the distance from the corresponding hyperplane for patterns that are not situated on the correct side of the classifier.

For a pattern (\mathbf{x}_i, y_i) from the class $+1$, the slack variable is defined as

$$\xi_i(\mathbf{w}, b) = \begin{cases} 0 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \geq +1 \\ 1 - (\mathbf{w} \cdot \mathbf{x}_i + b) & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \leq +1 \end{cases} \quad [34]$$

Similarly, for a pattern (\mathbf{x}_i, y_i) from the class -1 , the slack variable is defined as

$$\xi_i(\mathbf{w}, b) = \begin{cases} 0 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \leq -1 \\ 1 + (\mathbf{w} \cdot \mathbf{x}_i + b) & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \geq -1 \end{cases} \quad [35]$$

From Eqs. [34] and [35] and Figure 27, one can see that the slack variable $\xi_i(\mathbf{w}, b)$ is zero for $+1$ patterns that are classified correctly by hyperplane H_2 ($\mathbf{w} \cdot \mathbf{x} + b \geq +1$) and for -1 patterns that are classified correctly by hyperplane H_1 ($\mathbf{w} \cdot \mathbf{x} + b \leq -1$). Otherwise, the slack variable has a positive value that measures the distance between a pattern \mathbf{x}_i and the corresponding hyperplane $\mathbf{w} \cdot \mathbf{x} + b = y_i$. For $+1$ patterns situated in the buffer zone between H and H_2 , and for -1 patterns situated in the buffer zone between H and H_1 , the slack variable takes values between 0 and 1. Such patterns are not considered to be misclassified, but they have a penalty added to the objective function. If a pattern \mathbf{x}_i is located in the “forbidden” region of the classifier, then $\xi_i(\mathbf{w}, b) > 1$ (see the patterns in square boxes from Figure 27) and the pattern is considered to be misclassified. We can combine Eqs. [34] and [35] for slack variables of $+1$ and -1 patterns into Eq. [36]:

$$\xi_i(\mathbf{w}, b) = \begin{cases} 0 & \text{if } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq +1 \\ 1 - y_i(\mathbf{w} \cdot \mathbf{x}_i + b) & \text{if } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \leq +1 \end{cases} \quad [36]$$

When slack variables are introduced to penalize misclassified patterns or patterns situated in the buffer region between H and the corresponding border hyperplanes (H_1 or H_2), the constraints imposed to the objective function are as follows:

$$\begin{cases} \mathbf{w} \cdot \mathbf{x}_i + b \geq +1 - \xi_i & \text{if } y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b \leq -1 + \xi_i & \text{if } y_i = -1 \\ \xi_i > 0, \forall i \end{cases} \quad [37]$$

The identification of an OSH is much more difficult when slack variables are used, because the optimum classifier is a compromise between two opposing conditions. On the one hand, a good SVMC corresponds to a hyperplane

(\mathbf{w}, b) with a margin as large as possible in order to guarantee good prediction performances, which translates into minimizing $\|\mathbf{w}\|^2/2$. On the other hand, the optimum hyperplane should minimize the number of classification errors and it should also minimize the error of misclassified patterns, which translates in minimizing the number of positive slack variables and simultaneously minimizing the value of each slack variable. The latter condition has the tendency of decreasing the width of the SVMC hyperplane, which is in contradiction with the former condition. A simple way to combine these two conditions and to assign a penalty for classification errors is to change the objective function to be minimized from $\|\mathbf{w}\|^2/2$ to

$$\frac{\|\mathbf{w}\|^2}{2} + C \left(\sum_{i=1}^m \xi_i \right)^k \quad [38]$$

where C is a parameter that can be adjusted by the user, and can either increase or decrease the penalty for classification errors. A large C assigns a higher penalty to classification errors, thus minimizing the number of misclassified patterns. A small C maximizes the margin so that the OSH is less sensitive to the errors from the learning set. Equation [38] is a convex programming problem for any positive integer k , which for $k = 1$ and $k = 2$ is also a quadratic programming problem. The formula with $k = 1$ has the advantage that neither ξ_i nor their Lagrange multipliers appear in the Wolfe dual problem.⁴⁰

Based on the above considerations, we are now ready to state the form of the optimization problem for SVMC with a linear classifier and classification errors:

$$\begin{aligned} &\text{minimize } \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^m \xi_i \\ &\text{with the constraints } \begin{aligned} &y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq +1 - \xi_i, \quad i = 1, \dots, m \\ &\xi_i \geq 0, \quad i = 1, \dots, m \end{aligned} \end{aligned} \quad [39]$$

To solve the above constrained quadratic optimization problem, we follow the approach based on Lagrange multipliers (Eq. [21]). We define the Lagrange multipliers $\mathbf{\Lambda} = (\lambda_1, \lambda_2, \dots, \lambda_m)$ for each constraint $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq +1 - \xi_i$ and the Lagrange multipliers $\mathbf{M} = (\mu_1, \mu_2, \dots, \mu_m)$ for each constraint $\xi_i \geq 0, \forall i = 1, \dots, m$. With these notations, the primal Lagrangian function of this problem is

$$L_P(\mathbf{w}, b, \mathbf{\Lambda}, \mathbf{M}) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i - \sum_{i=1}^m \lambda_i [y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^m \mu_i \xi_i \quad [40]$$

where $\Lambda = (\lambda_1, \lambda_2, \dots, \lambda_m)$ is the set of Lagrange multipliers of the training (calibration) patterns.

The Karuch–Kuhn–Tucker conditions⁴⁰ for the primal problem are as follows:

Gradient Conditions

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i = 0$$

where

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial \mathbf{w}} = \left(\frac{\partial L}{\partial w_1}, \frac{\partial L}{\partial w_2}, \dots, \frac{\partial L}{\partial w_n} \right) \quad [41]$$

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial b} = \sum_{i=1}^m \lambda_i y_i = 0 \quad [42]$$

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial \xi_i} = C - \lambda_i - \mu_i = 0 \quad [43]$$

Orthogonality Condition

$$\lambda_i [y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i] = 0, \quad i = 1, \dots, m \quad [44]$$

Feasibility Condition

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i \geq 0, \quad i = 1, \dots, m \quad [45]$$

Non-negativity Condition

$$\begin{aligned} \xi_i &\geq 0, & i &= 1, \dots, m \\ \lambda_i &\geq 0, & i &= 1, \dots, m \\ \mu_i &\geq 0, & i &= 1, \dots, m \\ \mu_i \xi_i &= 0, & i &= 1, \dots, m \end{aligned} \quad [46]$$

We now substitute Eqs. [41] and [42] into the right side of the Lagrangian function, obtaining the dual problem

$$\begin{aligned} \text{maximize } L_D(\mathbf{w}, b, \Lambda, \mathbf{M}) &= \sum_{i=1}^m \lambda_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\ \text{subject to } 0 &\leq \lambda_i \leq C, \quad i = 1, \dots, m \\ \text{and } \sum_{i=1}^m \lambda_i y_i &= 0 \end{aligned} \quad [47]$$

The solution for the vector \mathbf{w} is obtained from Eq. [41], which represents one of the KKT conditions:

$$\mathbf{w} = \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \quad [48]$$

The value of b can be computed as an average for the b values obtained from all training patterns with the following KKT conditions:

$$\begin{aligned} \lambda_i [y_i (\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i] &= 0 \\ (C - \lambda_i) \xi_i &= 0 \end{aligned} \quad [49]$$

From the above equations, we have also that $\xi_i = 0$ if $\lambda_i < C$. Therefore, b can be averaged only for those patterns that have $0 \leq \lambda_i < C$.

We will now examine the relationships between the position of a pattern \mathbf{x}_i and the corresponding values for λ_i , ξ_i , and C . The following situations can be distinguished:

1. ($\lambda_i = 0$; $\xi_i = 0$): The pattern is inside the +1 region ($\mathbf{w} \cdot \mathbf{x}_i + b > +1$) if $y_i = +1$ or inside the -1 region ($\mathbf{w} \cdot \mathbf{x}_i + b < -1$) if $y_i = -1$, i.e., it is correctly classified, and its distance from the separating hyperplane is larger than $1/\|\mathbf{w}\|$. Such patterns are not important in defining the SVMC model, and they do not influence the solution. Hence, they can be deleted from the learning set without affecting the model.
2. ($0 < \lambda_i < C$; $\xi_i = 0$): This situation corresponds to correctly classified patterns situated on the hyperplanes that border the SVMC OSH, i.e., patterns +1 are situated on the hyperplane ($\mathbf{w} \cdot \mathbf{x}_i + b = +1$), whereas patterns -1 are situated on the hyperplane ($\mathbf{w} \cdot \mathbf{x}_i + b = -1$). The distance between these patterns and the separating hyperplane is $1/\|\mathbf{w}\|$. Such a pattern is called a margin support vector.
3. ($\lambda_i = C$; $0 < \xi_i \leq 1$): These patterns, correctly classified, are called bound support vectors, and their distance to the separating hyperplane is smaller than $1/\|\mathbf{w}\|$. Patterns from the class +1 are situated in the buffer zone between the separating hyperplane ($\mathbf{w} \cdot \mathbf{x}_i + b = 0$) and the border hyperplane toward the +1 region ($\mathbf{w} \cdot \mathbf{x}_i + b = +1$). Patterns from the class -1 are situated in the buffer zone between the separating hyperplane ($\mathbf{w} \cdot \mathbf{x}_i + b = 0$) and the border hyperplane toward the -1 region ($\mathbf{w} \cdot \mathbf{x}_i + b = -1$).
4. ($\lambda_i = C$; $\xi_i > 1$): These patterns are incorrectly classified. Patterns from the class +1 are situated in the -1 region defined by the separating hyperplane ($\mathbf{w} \cdot \mathbf{x}_i + b < 0$), whereas patterns from the class -1 are situated in the +1 region of the separating hyperplane ($\mathbf{w} \cdot \mathbf{x}_i + b > 0$).

The classification of new patterns uses the optimum values for \mathbf{w} (Eq. [48]) and b (Eq. [49]):

$$\text{class}(\mathbf{x}_k) = \text{sign} \left(\sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x}_k + b \right) \quad [50]$$

Equation [50] depends only on the support vectors and their Lagrange multipliers, and the optimum value for b , showing that one does not need to compute \mathbf{w} explicitly in order to predict the classification of new patterns.

NONLINEAR SUPPORT VECTOR MACHINES

In previous sections, we introduced the linear SVM classification algorithm, which uses the training patterns to generate an optimum separation hyperplane. Such classifiers are not adequate for cases when complex relationships exist between input parameters and the class of a pattern. To discriminate linearly nonseparable classes of patterns, the SVM model can be fitted with nonlinear functions to provide efficient classifiers for hard-to-separate classes of patterns.

Mapping Patterns to a Feature Space

The separation surface may be nonlinear in many classification problems, but support vector machines can be extended to handle nonlinear separation surfaces by using feature functions $\phi(\mathbf{x})$. The SVM extension to nonlinear datasets is based on mapping the input variables into a feature space of a higher dimension (a Hilbert space of finite or infinite dimension) and then performing a linear classification in that higher dimensional space. For example, consider the set of nonlinearly separable patterns in Figure 28, left. It is

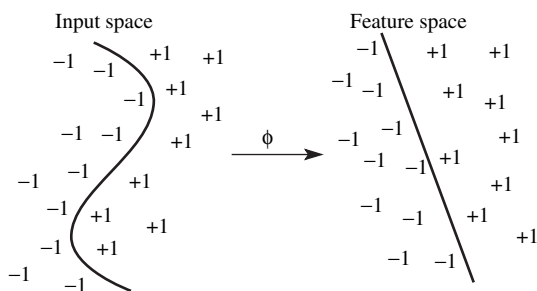


Figure 28 Linear separation of patterns in feature space.

clear that a linear classifier, even with slack variables, is not appropriate for this type of separation surface, which is obviously nonlinear. The nonlinear feature functions ϕ transform and combine the original coordinates of the patterns and perform their mapping into a high-dimensional space (Figure 28, right) where the two classes can be separated with a linear classifier. This property is of value because linear classifiers are easy to compute, and we can use the results obtained for linear SVM classification from the previous sections. The only difficulty is to identify, for a particular dataset, the correct set of nonlinear functions that can perform such mapping.

Consider a training set T of m patterns together with their classes, $T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$, where \mathbf{x} is an n -dimensional pattern, $\mathbf{x} = (x_1, x_2, \dots, x_n)$. Define the set of feature functions as $\phi_1, \phi_2, \dots, \phi_b$. Any pattern \mathbf{x} is mapped to a real vector $\phi(\mathbf{x})$:

$$\mathbf{x} = (x_1, x_2, \dots, x_n) \rightarrow \phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots, \phi_b(\mathbf{x})) \quad [51]$$

After mapping all patterns from the learning set into the feature set, we obtain a set of points in the feature space R^b :

$$\phi(T) = \{(\phi(\mathbf{x}_1), y_1), (\phi(\mathbf{x}_2), y_2), \dots, (\phi(\mathbf{x}_m), y_m)\} \quad [52]$$

The important property of the feature space is that the learning set $\phi(T)$ might be linearly separable in the feature space if the appropriate feature functions are used, even when the learning set is not linearly separable in the original space.

We consider a soft margin SVM in which the variables \mathbf{x} are substituted with the feature vector $\phi(\mathbf{x})$, which represents an optimization problem similar with that from Eq. [39]. Using this nonlinear SVM, the class of a pattern \mathbf{x}_k is determined with Eq. [53].

$$\text{class}(\mathbf{x}_k) = \text{sign}[\mathbf{w} \cdot \phi(\mathbf{x}_k) + b] = \text{sign}\left(\sum_{i=1}^m \lambda_i y_i \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_k) + b\right) \quad [53]$$

The nonlinear classifier defined by Eq. [53] shows that to predict a pattern \mathbf{x}_k , it is necessary to compute the dot product $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_k)$ for all support vectors \mathbf{x}_i . This property of the nonlinear classifier is very important, because it shows that we do not need to know the actual expression of the feature function ϕ . Moreover, a special class of functions, called kernels, allows the computation of the dot product $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_k)$ in the original space defined by the training patterns.

We present now a simple example of linearly nonseparable classes that can become linearly separable in feature space. Consider the dataset from Table 4 and Figure 29. This two-dimensional dataset, with dimensions

Table 4 Linearly Nonseparable Patterns that Can be Separated in a Feature Space

Pattern	x_1	x_2	x_1^2	Class
1	-1	-1	+1	-1
2	-1	0	+1	-1
3	-1	+1	+1	-1
4	0	-1	0	+1
5	0	0	0	+1
6	0	+1	0	+1
7	+1	-1	+1	-1
8	+1	0	+1	-1
9	+1	+1	+1	-1

x_1 and x_2 , consists of three patterns in class +1 and six patterns in class -1. From Figure 29, it is easy to deduce that there is no straight line that can separate these two classes.

On the other hand, one can imagine a higher dimensional feature space in which these classes become linearly separable. The features are combinations of the input data, and for this example, we add x_1^2 as a new dimension (Table 4, column 4). After this transformation, the dataset is represented in a three-dimensional feature space.

The surface $f(x_1, x_2) = x_1^2$ is represented in Figure 30. By adding this simple feature, we have mapped the patterns onto a nonlinear surface. This is easily seen when we plot (Figure 31) the feature space points (x_1, x_2, x_1^2) that are located on the surface from Figure 30.

The feature x_1^2 has an interesting property, as one can see by inspecting Table 4: all patterns from class +1 have $x_1^2 = 0$, whereas all patterns from class -1 have $x_1^2 = +1$. By mapping the patterns in the feature space, we are now able to separate the two classes with a linear classifier, i.e., a plane (Figure 32). Of course, this plane is not unique, and in fact, there is an infinite number of planes that can now discriminate the two classes.

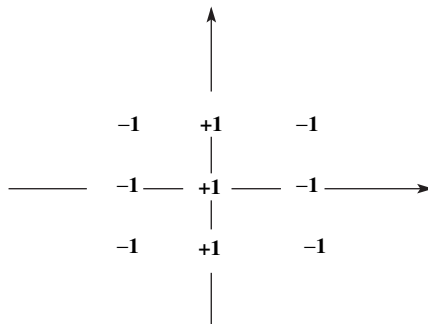


Figure 29 Linearly nonseparable two-dimensional patterns.

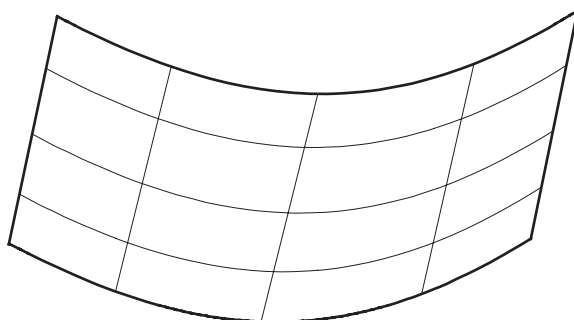


Figure 30 Surface $f(x, y) = x^2$.

The intersection between the feature space and the classifier defines the decision boundaries, which, when projected back onto the original space, look like Figure 33. Thus, transforming the input data into a nonlinear feature space makes the patterns linearly separable. Unfortunately, for a given dataset, one cannot predict which feature functions will make the patterns linearly separable; finding good feature functions is thus a trial-and-error process.

Feature Functions and Kernels

The idea of transforming the input space into a feature space of a higher dimension by using feature functions $\phi(\mathbf{x})$ and then performing a linear

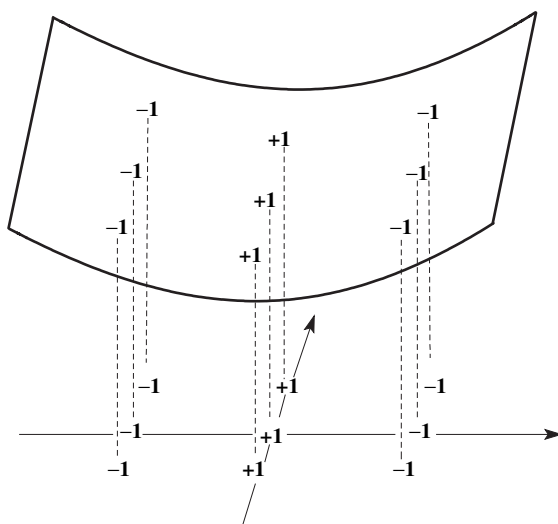


Figure 31 Feature space points (x, y, x^2) .

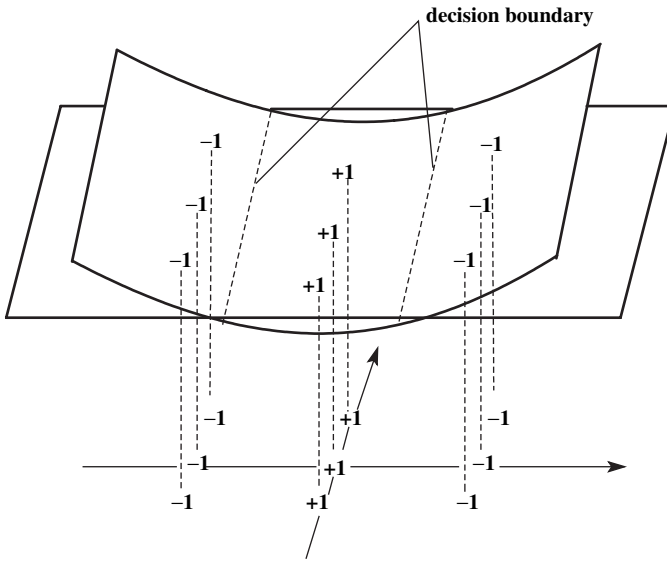


Figure 32 A separation plane for $+1$ patterns (below the plane) and -1 patterns (above the plane).

classification in that higher dimensional space is central to support vector machines. However, the feature space may have a very high dimensionality, even infinite. An obvious consequence is that we want to avoid the inner product of feature functions $\phi(\mathbf{x})$ that appears in Eq. [53]. Fortunately, a method was developed to generate a mapping into a high-dimensional feature space with kernels. The rationale that prompted the use of kernel functions is to

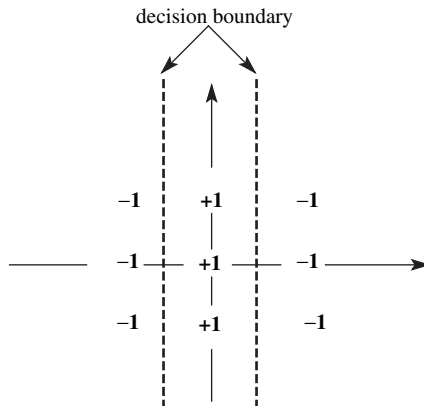


Figure 33 Projection of the separation plane.

enable computations to be performed in the original input space rather than the high-dimensional (even infinite) feature space. Using this approach, the SVM algorithm avoids the evaluation of the inner product of the feature functions.

Under certain conditions, an inner product in feature space has an equivalent kernel in input space:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \quad [54]$$

If the kernel K is a symmetric positive definite function, which satisfies the Mercer's conditions:^{4,42}

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_k^{\infty} a_k \phi_k(\mathbf{x}_i) \phi_k(\mathbf{x}_j), a_k \geq 0 \quad [55]$$

and

$$\iint K(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_i) g(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j > 0 \quad [56]$$

then the kernel represents an inner product in feature space.

Consider the two-dimensional pattern $\mathbf{x} = (x_1, x_2)$ and the feature function defined for a two-dimensional pattern \mathbf{x} :

$$\phi(\mathbf{x}) = \left(1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2\right) \quad [57]$$

From the expression of this feature function, it is easy to obtain the corresponding kernel function

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^2 \quad [58]$$

This example can be easily extended to a three-dimensional pattern $\mathbf{x} = (x_1, x_2, x_3)$, when the feature function has the expression

$$\phi(\mathbf{x}) = \left(1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_3, x_1^2, x_2^2, x_3^2, \sqrt{2}x_1x_2, \sqrt{2}x_1x_3, \sqrt{2}x_2x_3\right) \quad [59]$$

which corresponds to the polynomial of degree two kernel from Eq. [58]. In a similar way, a two-dimensional pattern $\mathbf{x} = (x_1, x_2)$ and a feature function

$$\phi(\mathbf{x}) = \left(1, \sqrt{3}x_1, \sqrt{3}x_2, \sqrt{3}x_1^2, \sqrt{3}x_2^2, \sqrt{6}x_1x_2, \sqrt{3}x_1^2x_2, \sqrt{3}x_1x_2^2, x_1^3, x_2^3\right) \quad [60]$$

is equivalent with a polynomial of degree three kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^3 \quad [61]$$

We will now present an example of infinite dimension feature function with the expression

$$\phi(\mathbf{x}) = \left(\sin(x), \frac{1}{\sqrt{2}}\sin(2x), \frac{1}{\sqrt{3}}\sin(3x), \frac{1}{\sqrt{4}}\sin(4x), \dots, \frac{1}{\sqrt{n}}\sin(nx), \dots \right) \quad [62]$$

where $x \in [1, \pi]$. The kernel corresponding to this infinite series has a very simple expression, which can be easily calculated as follows:

$$\begin{aligned} K(x_i, x_j) &= \phi(x_i) \cdot \phi(x_j) \\ &= \sum_{n=1}^{\infty} \frac{1}{n} \sin(nx_i) \sin(nx_j) = \frac{1}{2} \log \left| \sin\left(\frac{x_i + x_j}{2}\right) / \sin\left(\frac{x_i - x_j}{2}\right) \right| \quad [63] \end{aligned}$$

Kernel Functions for SVM

In this section, we present the most used SVM kernels. As these functions are usually computed in a high-dimensional space and have a nonlinear character, it is not easy to derive an impression on the shape of the classification hyperplane generated by these kernels. Therefore, we will present several plots for SVM models obtained for the dataset shown in Table 5. This dataset is not separable with a linear classifier, but the two clusters can be clearly distinguished.

Linear (Dot) Kernel

The inner product of x and y defines the linear (dot) kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j \quad [64]$$

This is a linear classifier, and it should be used as a test of the nonlinearity in the training set, as well as a reference for the eventual classification improvement obtained with nonlinear kernels.

Table 5 Linearly Nonseparable Patterns Used for the SVM Classification Models in Figures 34–38

Pattern	x_1	x_2	Class	Pattern	x_1	x_2	Class
1	2	4	1	9	0.6	4.5	-1
2	2.5	2.75	1	10	1	3	-1
3	3	5	1	11	1.5	1	-1
4	3.5	2	1	12	2	5.7	-1
5	4.5	4.75	1	13	3.5	5.5	-1
6	5	3.75	1	14	4	0.6	-1
7	3.25	4	1	15	5	1.5	-1
8	4	3.25	1	16	5.3	5.4	-1
				17	5.75	3	-1

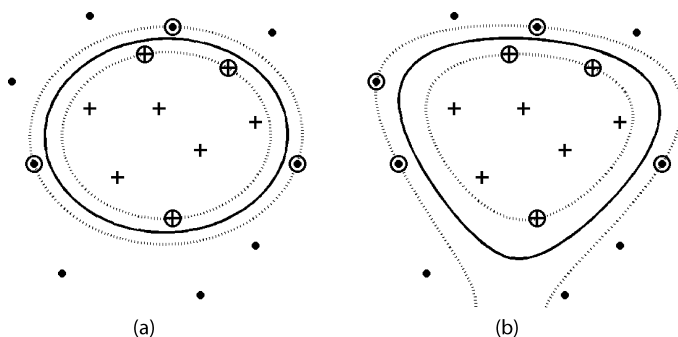


Figure 34 SVM classification models obtained with the polynomial kernel (Eq. [65]) for the dataset from Table 5: (a) polynomial of degree 2; (b) polynomial of degree 3.

Polynomial Kernel

The polynomial kernel is a simple and efficient method for modeling nonlinear relationships:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^d \quad [65]$$

The dataset from Table 5 can be separated easily with a polynomial kernel (Figure 34a, polynomial of degree 2). The downside of using polynomial kernels is the overfitting that might appear when the degree increases (Figure 34b, degree 3; Figure 35a, degree 5; Figure 35b, degree 10). As the degree of the polynomial increases, the classification surface becomes more complex. For the degree 10 polynomial, one can see that the border hypersurface defines two regions for the cluster of +1 patterns.

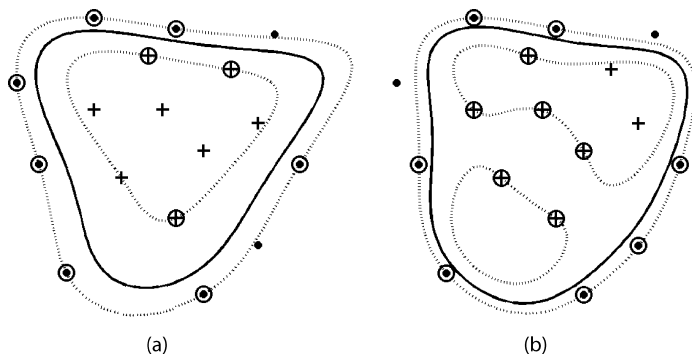


Figure 35 SVM classification models obtained with the polynomial kernel (Eq. [65]) for the dataset from Table 5: (a) polynomial of degree 5; (b) polynomial of degree 10.

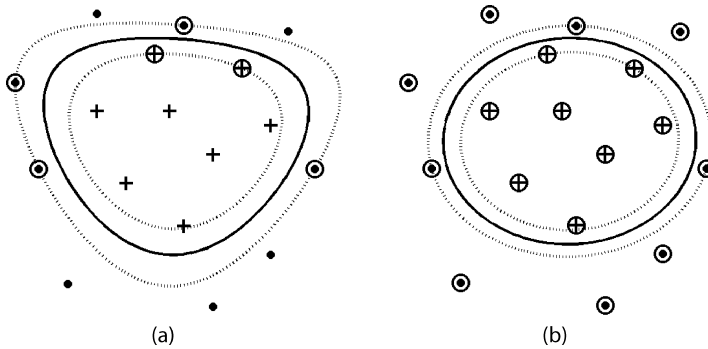


Figure 36 SVM classification models obtained with the Gaussian radial basis function kernel (Eq. [66]) for the dataset from Table 5: (a) $\sigma = 1$; (b) $\sigma = 10$.

Gaussian Radial Basis Function Kernel

Radial basis functions (RBF) are widely used kernels, usually in the Gaussian form:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right) \tag{66}$$

The parameter σ controls the shape of the separating hyperplane, as one can see from the two SVM models in Figure 36, both obtained with a Gaussian RBF kernel (a, $\sigma = 1$; b, $\sigma = 10$). The number of support vectors increases from 6 to 17, showing that the second setting does not generalize well. In practical applications, the parameter σ should be optimized with a suitable cross-validation procedure.

Exponential Radial Basis Function Kernel

If discontinuities in the hyperplane are acceptable, an exponential RBF kernel is worth trying:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|}{2\sigma^2}\right) \tag{67}$$

The form of the OSH obtained for this kernel is apparent in Figure 37, where two values for the parameter σ are exemplified (a, $\sigma = 0.5$; b, $\sigma = 2$). For the particular dataset used here, this kernel is not a good choice, because it requires too many support vectors.

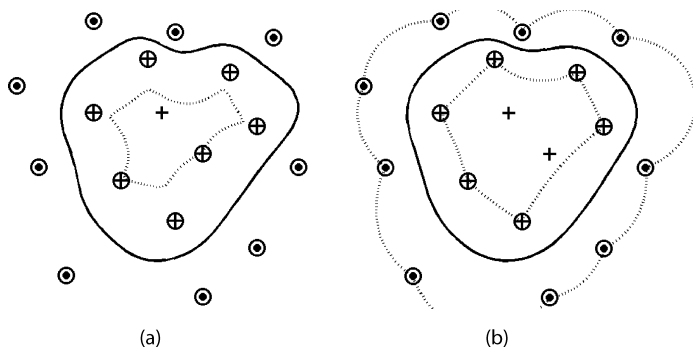


Figure 37 SVM classification models obtained with the exponential radial basis function kernel (Eq. [67]) for the dataset from Table 5: (a) $\sigma = 0.5$; (b) $\sigma = 2$.

Neural (Sigmoid, Tanh) Kernel

The hyperbolic tangent (tanh) function, with a sigmoid shape, is the most used transfer function for artificial neural networks. The corresponding kernel has the formula:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(a\mathbf{x}_i \cdot \mathbf{x}_j + b) \quad [68]$$

Anova Kernel

A useful function is the anova kernel, whose shape is controlled by the parameters γ and d :

$$K(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_i \exp(-\gamma(\mathbf{x}_i - \mathbf{x}_j)) \right)^d \quad [69]$$

Fourier Series Kernel

A Fourier series kernel, on the interval $[-\pi/2, +\pi/2]$, is defined by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sin(N + 1/2)(\mathbf{x}_i - \mathbf{x}_j)}{\sin(1/2(\mathbf{x}_i - \mathbf{x}_j))} \quad [70]$$

Spline Kernel

The spline kernel of order k having N knots located at t_s is defined by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{r=0}^k \mathbf{x}_i^r \mathbf{x}_j^r + \sum_{s=1}^N (\mathbf{x}_i - t_s)_+^k (\mathbf{x}_j - t_s)_+^k \quad [71]$$

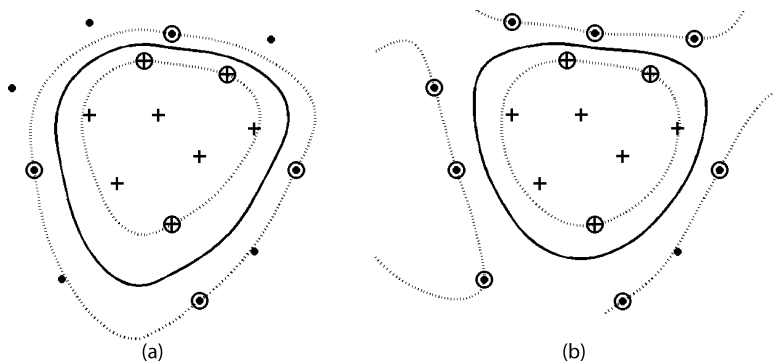


Figure 38 SVM classification models for the dataset from Table 5: (a) spline kernel, Eq. [71]; (b) B spline kernel, degree 1, Eq. [72].

B Spline Kernel

The B spline kernel is defined on the interval $[-1, 1]$ by the formula:

$$K(\mathbf{x}_i, \mathbf{x}_j) = B_{2N+1}(\mathbf{x}_i - \mathbf{x}_j) \tag{72}$$

Both spline kernels have a remarkable flexibility in modeling difficult data. This characteristic is not always useful, especially when the classes can be separated with simple nonlinear functions. The SVM models from Figure 38 (a, spline; b, B spline, degree 1) show that the B spline kernel overfits the data and generates a border hyperplane that has three disjoint regions.

Additive Kernel

An interesting property of kernels is that one can combine several kernels by summing them. The result of this summation is a valid kernel function:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_i K_i(\mathbf{x}_i, \mathbf{x}_j) \tag{73}$$

Tensor Product Kernel

The tensor product of two or more kernels is also a kernel function:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \prod_i K_i(\mathbf{x}_i, \mathbf{x}_j) \tag{74}$$

In many SVM packages, these properties presented in Eqs. [73] and [74] allow the user to combine different kernels in order to generate custom kernels more suitable for particular applications.

Hard Margin Nonlinear SVM Classification

In Figure 39, we present the network structure of a support vector machine classifier. The input layer is represented by the support vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$ and the test (prediction) pattern \mathbf{x}_t , which are transformed by the feature function ϕ and mapped into the feature space. The next layer performs the dot product between the test pattern $\phi(\mathbf{x}_t)$ and each support vector $\phi(\mathbf{x}_i)$. The dot product of feature functions is then multiplied by the Lagrangian multipliers, and the output is the nonlinear classifier from Eq. [53] in which the dot product of feature functions was substituted with a kernel function.

The mathematical formulation of the hard margin nonlinear SVM classification is similar to that presented for the SVM classification for linearly separable datasets, only now input patterns \mathbf{x} are replaced with feature functions, $\mathbf{x} \rightarrow \phi(\mathbf{x})$, and the dot product for two feature functions $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ is replaced with a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, Eq. [64]. Analogously with Eq. [28], the dual problem is

$$\begin{aligned} \text{maximize } L_D(\mathbf{w}, b, \Lambda) &= \sum_{i=1}^m \lambda_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \\ \text{subject to } \lambda_i &\geq 0, \quad i = 1, \dots, m \\ \text{and } \sum_{i=1}^m \lambda_i y_i &= 0 \end{aligned} \quad [75]$$

The vector \mathbf{w} that determines the optimum separation hyperplane is

$$\mathbf{w} = \sum_{i=1}^m \lambda_i y_i \phi(\mathbf{x}_i) \quad [76]$$

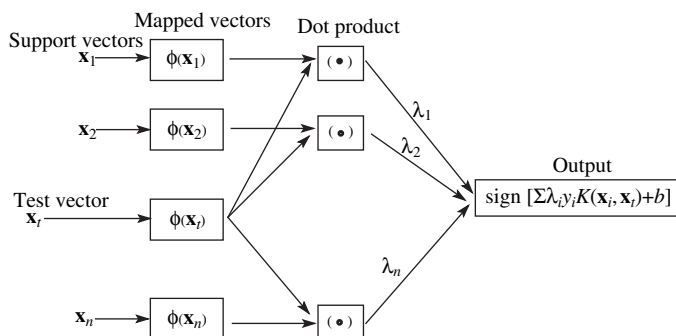


Figure 39 Structure of support vector machines. The test pattern \mathbf{x}_t and the support vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$ are mapped into a feature space with the nonlinear function ϕ , and the dot products are computed.

As with the derivation of b in Eq. [30], we have

$$\sum_{i=1}^m \lambda_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b = y_j \tag{77}$$

Therefore, the threshold b can be obtained by averaging the b values obtained for all support vector patterns, i.e., the patterns with $\lambda_j > 0$:

$$b = y_j - \sum_{i=1}^m \lambda_i y_i K(\mathbf{x}_i, \mathbf{x}_j) \tag{78}$$

The SVM classifier obtained with a kernel K is defined by the support vectors from the training set ($\lambda_i > 0$) and the corresponding values of the Lagrange multipliers λ_i :

$$\text{class}(\mathbf{x}_k) = \text{sign} \left(\sum_{i=1}^m \lambda_i y_i K(\mathbf{x}_i, \mathbf{x}_k) + b \right) \tag{79}$$

Soft Margin Nonlinear SVM Classification

A soft margin nonlinear SVM classifier is obtained by introducing slack variables ξ and the capacity C . As with Eq. [47], the dual problem is

$$\begin{aligned} \text{maximize } L_D(\mathbf{w}, b, \Lambda) &= \sum_{i=1}^m \lambda_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \\ \text{subject to } &0 \leq \lambda_i \leq C, \quad i = 1, \dots, m \\ \text{and } &\sum_{i=1}^m \lambda_i y_i = 0 \end{aligned} \tag{80}$$

which defines a classifier identical with the one from Eq. [79].

The capacity parameter C is very important in balancing the penalty for classification errors. It is usually adjusted by the user, or it can be optimized automatically by some SVM packages. The penalty for classification errors increases when the capacity C increases, with the consequence that the number of erroneously classified patterns decreases when C increases. On the other hand, the margin decreases when C increases, making the classifier more sensitive to noise or errors in the training set. Between these divergent requirements (small C for a large margin classifier; large C for a small number of classification errors), an optimum value should be determined, usually by trying to maximize the cross-validation prediction.

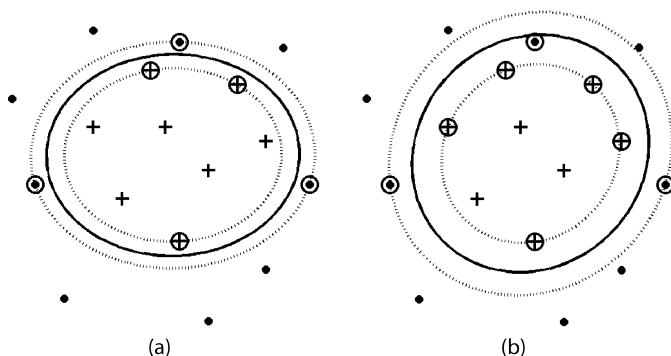


Figure 40 Influence of the C parameter on the class separation. SVM classification models obtained with the polynomial kernel of degree 2 for the dataset from Table 5: (a) $C = 100$; (b) $C = 10$.

To illustrate the influence of the capacity parameter C on the separation hyperplane with the dataset from Table 5 and a polynomial kernel of degree 2, consider Figures 40 (a, $C = 100$; b, $C = 10$) and 41 (a, $C = 1$; b, $C = 0.1$). This example shows that a bad choice for the capacity C can ruin the performance of an otherwise very good classifier. Empirical observations suggest that $C = 100$ is a good value for a wide range of SVM classification problems, but the optimum value should be determined for each particular case.

A similar trend is presented for the SVM models obtained with the spline kernel, presented in Figure 38a (C infinite) and Figure 42 (a, $C = 100$; b, $C = 10$). The classifier from Figure 38a does not allow classification errors, whereas by decreasing the capacity C to 100 (Figure 42a), one -1 pattern is misclassified (indicated with an arrow). A further decrease

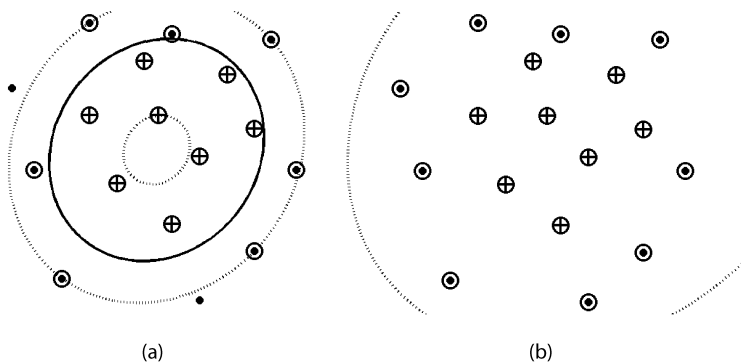


Figure 41 Influence of the C parameter on the class separation. SVM classification models obtained with the polynomial kernel of degree 2 for the dataset from Table 5: (a) $C = 1$; (b) $C = 0.1$.

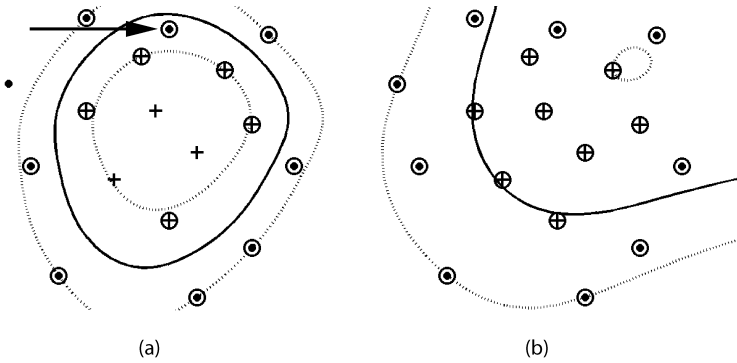


Figure 42 Influence of the C parameter on the class separation. SVM classification models obtained with the spline kernel for the dataset from Table 5: (a) $C = 100$; (b) $C = 10$.

of C to 10 increases the number of classification errors: one for class +1 and three for class -1.

v-SVM Classification

Another formulation of support vector machines is the v-SVM in which the parameter C is replaced by a parameter $v \in [0, 1]$ that is the lower and upper bound on the number of training patterns that are support vectors and are situated on the wrong side of the hyperplane. v-SVM can be used for both classification and regression, as presented in detail in several reviews, by Schölkopf et al.,⁴³ Chang and Lin,^{44,45} Steinwart,⁴⁶ and Chen, Lin, and Schölkopf.⁴⁷

The optimization problem for the v-SVM classification is

$$\begin{aligned} &\text{minimize } \frac{\|\mathbf{w}\|^2}{2} - v\rho + \frac{1}{2} \sum_{i=1}^m \xi_i \\ &\text{with the constraints } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq \rho - \xi_i, \quad i = 1, \dots, m \\ &\quad \quad \quad \xi_i \geq 0, \quad i = 1, \dots, m \end{aligned} \tag{81}$$

With these notations, the primal Lagrangian function of this problem is

$$\begin{aligned} L_P(\mathbf{w}, b, \mathbf{\Lambda}, \xi, \boldsymbol{\beta}, \rho, \delta) &= \frac{1}{2} \|\mathbf{w}\|^2 - v\rho + \frac{1}{m} \sum_{i=1}^m \xi_i \\ &\quad - \sum_{i=1}^m \left\{ \lambda_i [y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - \rho + \xi_i] + \beta_i \xi_i - \delta \rho \right\} \end{aligned} \tag{82}$$

with the Lagrange multipliers $\lambda_i, \beta_i, \delta \geq 0$. This function must be minimized with respect to \mathbf{w}, b, ξ, ρ , and maximized with respect to $\mathbf{\Lambda}, \boldsymbol{\beta}, \delta$. Following

the same derivation as in the case of C-SVM, we compute the corresponding partial derivatives and set them equal to 0, which leads to the following conditions:

$$\mathbf{w} = \sum_{i=1}^m \lambda_i y_i \mathbf{x}_i \quad [83]$$

$$\lambda_i + \beta_i = 1/m \quad [84]$$

$$\sum_{i=1}^m \lambda_i y_i = 0 \quad [85]$$

$$\sum_{i=1}^m \lambda_i - \delta = v \quad [86]$$

We substitute Eqs. [83] and [84] into Eq. [82], using $\lambda_i, \beta_i, \delta \geq 0$, and then we substitute the dot products with kernels, to obtain the following quadratic optimization problem:

$$\begin{aligned} \text{maximize } L_D(\mathbf{\Lambda}) &= -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to:} \\ 0 \leq \lambda_i &\leq 1/m \quad i = 1, \dots, m \\ \sum_{i=1}^m \lambda_i y_i &= 0 \\ \sum_{i=1}^m \lambda_i &\geq v \end{aligned} \quad [87]$$

From these equations, it follows that the v-SVM classifier is

$$\text{class}(\mathbf{x}_k) = \text{sign} \left(\sum_{i=1}^m \lambda_i y_i K(\mathbf{x}_i, \mathbf{x}_k) + b \right) \quad [88]$$

Schölkopf et al. showed that if a v-SVM classifier leads to $\rho > 0$, then the C-SVM classifier with $C = 1/m\rho$ has the same decision function.⁴³

Weighted SVM for Imbalanced Classification

In many practical applications, the ratio between the number of +1 and -1 patterns is very different from 1; i.e., one class is in excess and can dominate the SVM classifier. In other cases the classification error for one class may be more unfavorable or expensive than an error for the other class

(e.g., a clinical diagnostic error). In both classes, it is advantageous to use a variant of the SVM classifier, the weighted SVM, that uses different penalties (C^+ and C^-) for the two classes. The most unfavorable type of error has a higher penalty, which translates into an SVM classifier that minimizes that type of error. By analogy with Eq. [39], the primal problem is the Lagrangian function:

$$\begin{aligned} \text{minimize } & \frac{\|\mathbf{w}\|^2}{2} + C^+ \sum_{\substack{i=1 \\ y_i=+1}}^m \xi_i + C^- \sum_{\substack{i=1 \\ y_i=-1}}^m \xi_i \\ \text{with the constraints } & y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq +1 - \xi_i, \quad i = 1, \dots, m \\ & \xi_i \geq 0, \quad i = 1, \dots, m \end{aligned} \tag{89}$$

which is equivalent with the dual problem

$$\begin{aligned} \text{maximize } L_D(\mathbf{w}, b, \Lambda) &= \sum_{i=1}^m \lambda_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \\ \text{subject to :} & \\ & 0 \leq \lambda_i \leq C^+ \quad i = 1, \dots, m \quad \text{for } y_i = +1 \\ & 0 \leq \lambda_i \leq C^- \quad i = 1, \dots, m \quad \text{for } y_i = -1 \\ & \sum_{i=1}^m \lambda_i y_i = 0 \end{aligned} \tag{90}$$

The final solution is obtained by introducing the feature functions, $\mathbf{x} \rightarrow \phi(\mathbf{x})$, and substituting the dot product $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ with a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$.

Multi-class SVM Classification

Support vector machine classification was originally defined for two-class problems. This is a limitation in some cases when three or more classes of patterns are present in the training set as, for example, classifying chemical compounds as inhibitors for several targets.

Many multiclass SVM classification approaches decompose the training set into several two-class problems. The one-versus-one approach trains a two-class SVM model for any two classes from the training set, which for a k -class problem results in $k(k - 1)/2$ SVM models. In the prediction phase, a voting procedure assigns the class of the prediction pattern to be the class with the maximum number of votes. A variant of the one-versus-one approach is DAGSVM (directed acyclic graph SVM), which has an identical training procedure, but uses for prediction a rooted binary directed acyclic graph in which

each vertex is a two-class SVM model. Debnath, Takahide and Takahashi proposed an optimized one-versus-one multiclass SVM in which only a minimum number of SVM classifiers are trained for each class.⁴⁸

The one-versus-all procedure requires a much smaller number of models, namely for a k -class problem, only k SVM classifiers are needed. The i th SVM classifier is trained with all patterns from the i th class labeled $+1$, and all other patterns labeled -1 . Although it is easier to implement than the one-versus-one approach, the training sets may be imbalanced due to the large number of -1 patterns. In a comparative evaluation of one-versus-one, one-versus-all, and DAGSVM methods for 10 classification problems, Hsu and Lin found that one-versus-all is less suitable than the other methods.⁴⁹ However, not all literature reports agree with this finding. Based on a critical review of the existing literature on multiclass SVM and experiments with many datasets, Rifkin and Klautau concluded that the one-versus-all SVM classification is as accurate as any other multiclass approach.⁵⁰

Angulo, Parra and Català proposed the K -SVCR (K -class support vector classification-regression) for k -class classification.⁵¹ This algorithm has ternary outputs, $\{-1, 0, +1\}$, and in the learning phase evaluates all patterns in a one-versus-one-versus-rest procedure by using a mixed classification and regression SVM. The prediction phase implements a voting scheme that makes the algorithm fault-tolerant. Guermeur applied a new multiclass SVM, called M-SVM, to the prediction of protein secondary structure.^{52,53}

Multiclass SVM classification is particularly relevant for the classification of microarray gene expression data, with particular importance for disease recognition and classification.⁵⁴⁻⁵⁸

SVM REGRESSION

Initially developed for pattern classification, the SVM algorithm was extended by Vapnik⁴ for regression by using an ϵ -insensitive loss function (Figure 7). The goal of SVM regression (SVMR) is to identify a function $f(\mathbf{x})$ that for all training patterns \mathbf{x} has a maximum deviation ϵ from the target (experimental) values y and has a maximum margin. Using the training patterns, SVMR generates a model representing a tube with radius ϵ fitted to the data. For the hard margin SVMR, the error for patterns inside the tube is zero, whereas no patterns are allowed outside the tube. For real-case datasets, this condition cannot account for outliers, an incomplete set of input variables \mathbf{x} or experimental errors in measuring y . Analogously with SVM classification, a soft margin SVMR was introduced by using slack variables. Several reviews on SVM regression should be consulted for more mathematical details, especially those by Mangasarian and Musicant,^{59,60} Gao, Gunn, and Harris,^{61,62} and Smola and Schölkopf.³⁰

Consider a training set T of m patterns together with their target (experimental) values, $T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$, with $\mathbf{x} \in R^n$ and $y \in R$. The linear regression case with a hard margin is represented by the function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$, with $\mathbf{w} \in R^n$ and $b \in R$. For this simple case, the SVMR is represented by

$$\begin{aligned} &\text{minimize } \frac{\|\mathbf{w}\|^2}{2} \\ &\text{with the constraints } \begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b - y_i &\leq \varepsilon, & i = 1, \dots, m \\ y_i - \mathbf{w} \cdot \mathbf{x}_i - b &\leq \varepsilon, & i = 1, \dots, m \end{aligned} \end{aligned} \tag{91}$$

The above conditions can be easily extended for the soft margin SVM regression:

$$\begin{aligned} &\text{minimize } \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^m (\xi_i^+ + \xi_i^-) \\ &\text{with the constraints } \begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b - y_i &\leq \varepsilon + \xi_i^+, & i = 1, \dots, m \\ y_i - \mathbf{w} \cdot \mathbf{x}_i - b &\leq \varepsilon + \xi_i^-, & i = 1, \dots, m \\ \xi_i^+ &\geq 0, & i = 1, \dots, m \\ \xi_i^- &\geq 0, & i = 1, \dots, m \end{aligned} \end{aligned} \tag{92}$$

where ξ_i^+ is the slack variable associated with an overestimate of the calculated response for the input vector \mathbf{x}_i , ξ_i^- is the slack variable associated with an underestimate of the calculated response for the input vector \mathbf{x}_i , ε determines the limits of the approximation tube, and $C > 0$ controls the penalty associated with deviations larger than ε . In the case of the ε -insensitive loss function, the deviations are penalized with a linear function:

$$|\xi|_\varepsilon = \begin{cases} 0 & \text{if } |\xi| \leq \varepsilon \\ |\xi| - \varepsilon & \text{otherwise} \end{cases} \tag{93}$$

The SVM regression is depicted in Figure 43. The regression tube is bordered by the hyperplanes $y = \mathbf{w} \cdot \mathbf{x} + b + \varepsilon$ and $y = \mathbf{w} \cdot \mathbf{x} + b - \varepsilon$. Patterns situated between these hyperplanes have the residual (absolute value for the difference between calculated and experimental y) less than ε , and in SVM regression, the error of these patterns is considered zero; thus, they do not contribute to the penalty. Only patterns situated outside the regression tube have a residual larger than ε and thus a nonzero penalty that, for the ε -insensitive loss function, is proportional to their distance from the SVM regression border (Figure 43, right).

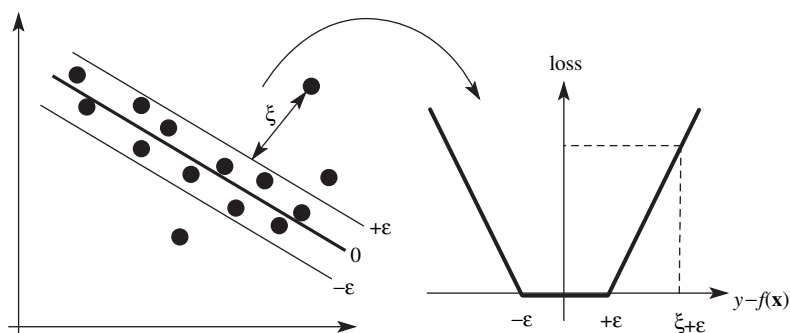


Figure 43 Linear SVM regression case with soft margin and ϵ -insensitive loss function.

The primal objective function is represented by the Lagrange function

$$L_P(\mathbf{w}, b, \Lambda, \mathbf{M}) = \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1}^m (\xi_i^+ + \xi_i^-) - \sum_{i=1}^m (\mu_i^+ \xi_i^+ + \mu_i^- \xi_i^-) \quad [94]$$

$$- \sum_{i=1}^m \lambda_i^+ (\epsilon + \xi_i^+ + y_i - \mathbf{w} \cdot \mathbf{x}_i - b) - \sum_{i=1}^m \lambda_i^- (\epsilon + \xi_i^- - y_i + \mathbf{w} \cdot \mathbf{x}_i + b)$$

where λ_i^+ , λ_i^- , μ_i^+ , and μ_i^- are the Lagrange multipliers. The KKT conditions for the primal problem are as follows:

Gradient Conditions

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial b} = \sum_{i=1}^m (\lambda_i^+ - \lambda_i^-) = 0 \quad [95]$$

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^m (\lambda_i^- - \lambda_i^+) \mathbf{x}_i = 0 \quad [96]$$

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial \xi_i^+} = C - \mu_i^+ - \lambda_i^+ = 0 \quad [97]$$

$$\frac{\partial L_P(\mathbf{w}, b, \Lambda, \mathbf{M})}{\partial \xi_i^-} = C - \mu_i^- - \lambda_i^- = 0 \quad [98]$$

Non-negativity Conditions

$$\begin{aligned} \xi_i^+, \xi_i^- &\geq 0, \quad i = 1, \dots, m \\ \lambda_i^+, \lambda_i^- &\geq 0, \quad i = 1, \dots, m \\ \mu_i^+, \mu_i^- &\geq 0, \quad i = 1, \dots, m \end{aligned} \quad [99]$$

The dual optimization problem is obtained by substituting Eqs. [95]–[98] into Eq. [94]:

$$\begin{aligned}
 L_D(\mathbf{w}, b, \Lambda, \mathbf{M}) &= -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m (\lambda_i^- - \lambda_i^+) (\lambda_j^- - \lambda_j^+) \mathbf{x}_i \cdot \mathbf{x}_j \\
 &\text{maximize} \\
 & - \varepsilon \sum_{i=1}^m (\lambda_i^- + \lambda_i^+) + \sum_{i=1}^m y_i (\lambda_i^- - \lambda_i^+) \tag{100} \\
 &\text{subject to } \sum_{i=1}^m (\lambda_i^- - \lambda_i^+) = 0 \\
 &\text{and } \lambda_i^-, \lambda_i^+ \in [0, C]
 \end{aligned}$$

The vector \mathbf{w} is obtained from Eq. [96]:

$$\mathbf{w} = \sum_{i=1}^m (\lambda_i^- - \lambda_i^+) \mathbf{x}_i \tag{101}$$

which leads to the final expression for $f(\mathbf{x}_k)$, the computed value for a pattern \mathbf{x}_k :

$$f(\mathbf{x}_k) = \sum_{i=1}^m (\lambda_i^- - \lambda_i^+) \mathbf{x}_i \cdot \mathbf{x}_k + b \tag{102}$$

Nonlinear SVM regression is obtained by introducing feature functions ϕ that map the input patterns into a higher dimensional space, $\mathbf{x} \rightarrow \phi(\mathbf{x})$. By replacing the dot product $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ with a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, we obtain from Eq. [100] the following optimization problem:

$$\begin{aligned}
 L_D(\mathbf{w}, b, \Lambda, \mathbf{M}) &= -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m (\lambda_i^- - \lambda_i^+) (\lambda_j^- - \lambda_j^+) K(\mathbf{x}_i, \mathbf{x}_j) \\
 &\text{maximize } - \varepsilon \sum_{i=1}^m (\lambda_i^- + \lambda_i^+) + \sum_{i=1}^m y_i (\lambda_i^- - \lambda_i^+) \tag{103} \\
 &\text{subject to } \sum_{i=1}^m (\lambda_i^- - \lambda_i^+) = 0 \\
 &\text{and } \lambda_i^-, \lambda_i^+ \in [0, C]
 \end{aligned}$$

Similarly with Eq. [101], the kernel SVM regression model has \mathbf{w} given by

$$\mathbf{w} = \sum_{i=1}^m (\lambda_i^- - \lambda_i^+) \phi(\mathbf{x}_i) \tag{104}$$

The modeled property for a pattern \mathbf{x}_k is obtained with the formula:

$$f(\mathbf{x}_k) = \sum_{i=1}^m (\lambda_i^- - \lambda_i^+) K(\mathbf{x}_i, \mathbf{x}_k) + b \tag{105}$$

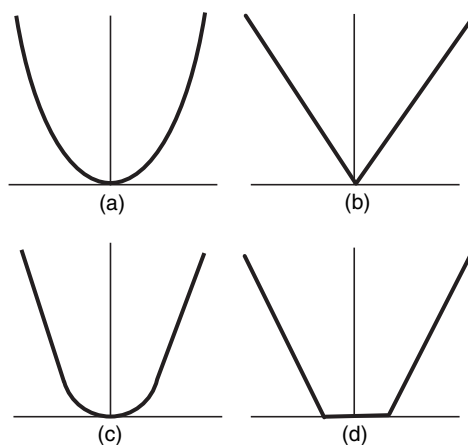


Figure 44 Loss functions for support vector machines regression: (a) quadratic; (b) Laplace; (c) Huber; (d) ε -insensitive.

The ε -insensitive loss function used in the SVM regression adds a new parameter ε that significantly influences the model and its prediction capacity. Besides the ε -insensitive, other loss functions can be used with SVM regression, such as quadratic, Laplace, or Huber loss functions (Figure 44).

We now present an illustrative example of a one-dimensional nonlinear SVM regression using the dataset in Table 6. This dataset has two spikes, which makes it difficult to model with the common kernels.

Table 6 Patterns Used for the SVM Regression Models in Figures 45–48

Pattern	x	y	Pattern	x	y
1	0.2	1.2	18	1.5	1.18
2	0.3	1.22	19	1.6	1.17
3	0.4	1.23	20	1.7	1.16
4	0.5	1.24	21	1.8	1.12
5	0.6	1.25	22	1.85	0.85
6	0.7	1.28	23	1.9	0.65
7	0.8	1.38	24	1.95	0.32
8	0.85	1.6	25	2.0	0.4
9	0.9	1.92	26	2.05	0.5
10	0.95	2.1	27	2.1	0.6
11	1.0	2.3	28	2.15	0.8
12	1.05	2.2	29	2.2	0.95
13	1.1	1.85	30	2.3	1.18
14	1.15	1.6	31	2.4	1.2
15	1.2	1.4	32	2.5	1.21
16	1.3	1.19	33	2.6	1.22
17	1.4	1.18			

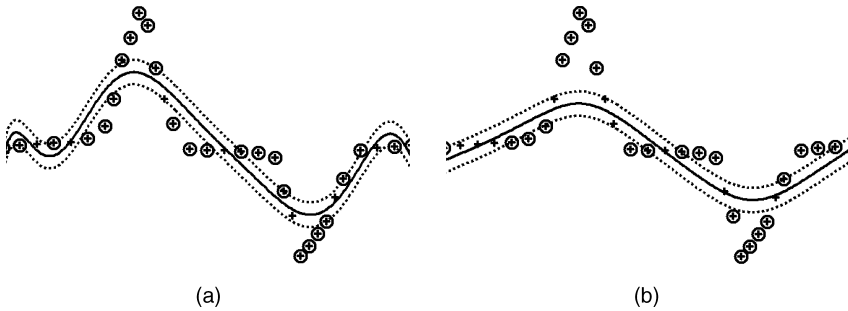


Figure 45 SVM regression models for the dataset from Table 6, with $\varepsilon = 0.1$: (a) degree 10 polynomial kernel; (b) spline kernel.

In Figure 45, we present two SVM regression models, the first one obtained with a degree 10 polynomial kernel and the second one computed with a spline kernel. The polynomial kernel has some oscillations on both ends of the curve, whereas the spline kernel is observed to be inadequate for modeling the two spikes. The RBF kernel was also unable to offer an acceptable solution for this regression dataset (data not shown).

The degree 1 B spline kernel (Figure 46a) with the parameters $C = 100$ and $\varepsilon = 0.1$ gives a surprisingly good SVM regression model, with a regression tube that closely follows the details of the input data. We will use this kernel to explore the influence of the ε -insensitivity and capacity C on the regression tube. By maintaining C to 100 and increasing ε to 0.3, we obtain a less sensitive solution (Figure 46b), that does not model well the three regions having almost constant y values. This is because the diameter of the tube is significantly larger and the patterns inside the tube do not influence the SVMR model (they have zero error).

By further increasing ε to 0.5 (Figure 47a), the shape of the SVM regression model becomes even less similar to the dataset. The regression tube is now

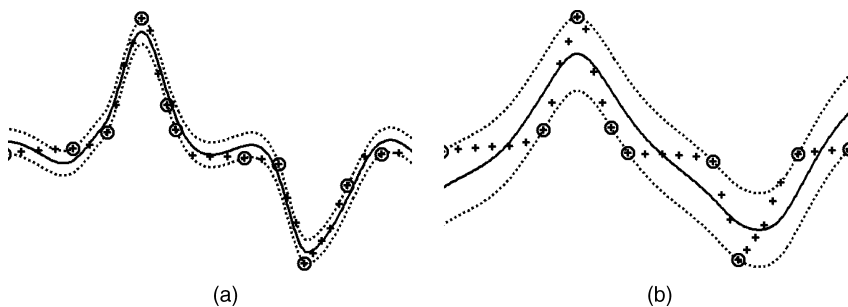


Figure 46 SVM regression models with a B spline kernel, degree 1, for the dataset from Table 6, with $C = 100$: (a) $\varepsilon = 0.1$; (b) $\varepsilon = 0.3$.

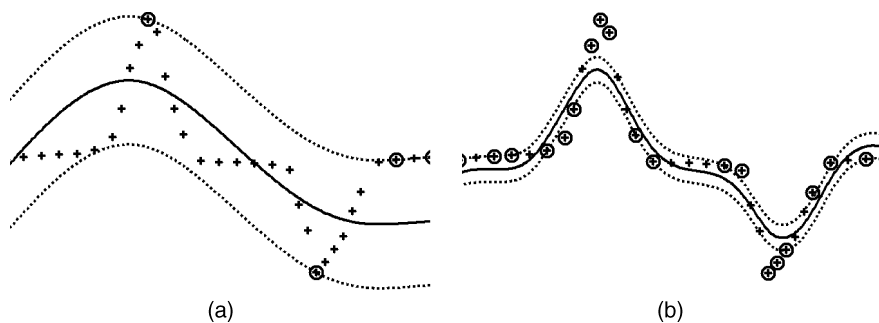


Figure 47 SVM regression models with a B spline kernel, degree 1, for the dataset from Table 6: (a) $\varepsilon = 0.5$, $C = 100$; (b) $\varepsilon = 0.1$, $C = 10$.

defined by a small number of support vectors, but they are not representative of the overall shape of the curve. It is now apparent that the ε -insensitivity parameter should be tailored for each specific problem because small variations in that parameter have significant effects on the regression model. We now consider the influence of the capacity C when ε is held constant to 0.1. The reference here is the SVMR model from Figure 46a obtained for $C = 100$. When C decreases to 10 (Figure 47b), the penalty for errors decreases and the solution is incapable of modeling the points with extreme y values in the two spikes accurately.

By further decreasing the capacity parameter C to 1 (Figure 48a) and then to 0.1 (Figure 48b), the SVMR model further loses the capacity to model the two spikes. The examples shown here for C are not representative for normal experimental values, and they are presented only to illustrate their influence on the shape of the regression hyperplane.

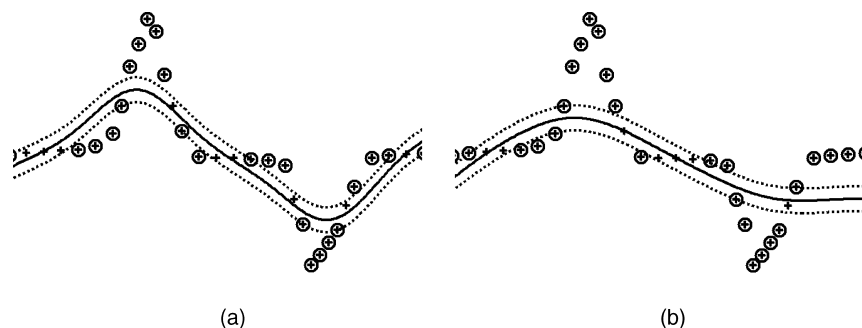


Figure 48 SVM regression models with a B spline kernel, degree 1, for the dataset from Table 6: (a) $\varepsilon = 0.1$, $C = 1$; (b) $\varepsilon = 0.1$, $C = 0.1$.

OPTIMIZING THE SVM MODEL

Finding an SVM model with good prediction statistics is a trial-and-error task. The objective is to maximize the predictions statistics while keeping the model simple in terms of number of input descriptors, number of support vectors, patterns used for training, and kernel complexity. In this section, we present an overview of the techniques used in SVM model optimization.

Descriptor Selection

Selecting relevant input parameters is both important and difficult for any machine learning method. For example, in QSAR, one can compute thousands of structural descriptors with software like CODESSA or Dragon, or with various molecular field methods. Many procedures have been developed in QSAR to identify a set of structural descriptors that retain the important characteristics of the chemical compounds.^{63,64} These methods can be extended to SVM models. Another source of inspiration is represented by the algorithms proposed in the machine learning literature, which can be readily applied to cheminformatics problems. We present here several literature pointers for algorithms on descriptor selection.

A variable selection method via sparse SVM was proposed by Bi et al.⁶⁵ In a first step, this method uses a linear SVM for descriptor selection, followed by a second step when nonlinear kernels are introduced. The recursive saliency analysis for descriptor selection was investigated by Cao et al.⁶⁶ Fung and Mangasarian proposed a feature selection Newton method for SVM.⁶⁷ Kumar et al. introduced a new method for descriptor selection, the locally linear embedding, which can be used for reducing the nonlinear dimensions in QSPR and QSAR.⁶⁸ Xue et al. investigated the application of recursive feature elimination for three classification tests, namely P-glycoprotein substrates, human intestinal absorption, and compounds that cause *torsade de pointes*.⁶⁹ Fröhlich, Wegner, and Zell introduced the incremental regularized risk minimization procedure from SVM classification and regression, and they compared it with recursive feature elimination and with the mutual information procedure.⁷⁰ Five methods of feature selection (information gain, mutual information, χ^2 -test, odds ratio, and GSS coefficient) were compared by Liu for their ability to discriminate between thrombin inhibitors and non-inhibitors.⁷¹ Byvatov and Schneider compared the SVM-based and the Kolmogorov–Smirnov feature selection methods to characterize ligand-receptor interactions in focused compound libraries.⁷² A genetic algorithm for descriptor selection was combined with SVM regression by Nandi et al. to model and optimize the benzene isopropylation on Hbeta catalyst.⁷³ Finally, gene selection from microarray data is a necessary step for disease classification^{55,74–81} with support vector machines.

Support Vectors Selection

The time needed to predict a pattern with an SVM model is proportional to the number of support vectors. This makes prediction slow when the SVM has a large number of support vectors. Downs, Gates, and Masters showed that the SMO algorithm,⁴¹ usually used for SVM training, can produce solutions with more support vectors than are needed for an optimum model.⁸² They found that some support vectors are linearly dependent on other support vectors, and that these linearly dependent support vectors can be identified and then removed from the SVM model with an efficient algorithm. Besides reducing of the number of support vectors, the new solution gives identical predictions with the full SVM model. Their model reduction algorithm was tested for several classification and regression problems, and in most cases lead to a reduction in the number of support vectors, which was as high as 90% in one example. In some cases, the SVM solution did not contain any linearly dependent support vectors so it was not possible to simplify the model.

Zhan and Shen proposed a four-step algorithm to simplify the SVM solution by removing unnecessary support vectors.⁸³ In the first step, the learning set is used to train the SVM and identify the support vectors. In the second step, the support vectors that make the surface convoluted (i.e., their projection of the hypersurface having the largest curvatures) are excluded from the learning set. In the third step, the SVM is retrained with the reduced learning set. In the fourth step, the complexity of the SVM model is further reduced by approximating the separation hypersurface with a subset of the support vectors. The algorithm was tested for tissue classification for 3-D prostate ultrasound images, demonstrating that the number of support vectors can be reduced without degrading the prediction of the SVM model.

Jury SVM

Starting from current machine learning algorithms (e.g., PCA, PLS, ANN, SVM, and k -NN), one can derive new classification or regression systems by combining the predictions of two or more models. Such a prediction meta-algorithm (called jury, committee, or ensemble) can use a wide variety of mathematical procedures to combine the individual predictions into a final prediction. Empirical studies showed that jury methods can increase the prediction performances of the individual models that are aggregated in the ensemble. Their disadvantages include the increased complexity of the model and longer computing time. In practical applications, the use of jury methods is justified if a statistically significant increase in prediction power is obtained. Several examples of using jury SVM follow.

Drug-like compound identification with a jury of k -NN, SVM, and ridge regression was investigated by Merkwirth et al.⁸⁴ Jury predictions with several machine learning methods were compared by Briem and Günther for the

discrimination of kinase inhibitors from noninhibitors.⁸⁵ Yap and Chen compared two jury SVM procedures for classifying inhibitors and substrates of cytochromes P450 3A4, 2D6 and 2C9.⁸⁶ Jerebko et al. used jury SVM classifiers based on bagging (bootstrap aggregation) for polyp detection in CT colonography.⁷⁸ Valentini, Muselli, and Ruffino used bagged jury SVM on DNA microarray gene expression data to classify normal and malignant tissues.⁸⁷ As a final example, we point out the work of Guermeur et al. who used a multi-class SVM to aggregate the best protein secondary structure prediction methods, thus improving their performances.^{52,53}

Kernels for Biosequences

The kernel function measures the similarity between pairs of patterns, typically as a dot product between numerical vectors. The usual numerical encoding for protein sequences is based on a 20-digit vector that encodes (binary) the presence/absence of a certain amino acid in a position. To explore new ways of encoding the structural information from biosequences, various kernels have been proposed for the prediction of biochemical properties directly from a given sequence.

Saigo et al. defined alignment kernels that compute the similarity between two sequences by summing up scores obtained from local alignments with gaps.⁸⁸ The new kernels could recognize SCOP superfamilies and outperform standard methods for remote homology detection. The mismatch kernel introduced by Leslie et al. measures sequence similarity based on shared occurrences of fixed-length patterns in the data, thus allowing for mutations between patterns.⁸⁹ This type of partial string matching kernel predicts successfully protein classification in families and superfamilies. Vert used a tree kernel to measure the similarity between phylogenetic profiles so as to predict the functional class of a gene from its phylogenetic profile.⁹⁰ The tree kernel can predict functional characteristics from evolutionary information. Yang and Chou defined a class of kernels that compute protein sequence similarity based on amino acid similarity matrices such as the Dayhoff matrix.⁹¹ String kernels computed from subsite coupling models of protein sequences were used by Wang, Yang, and Chou to predict the signal peptide cleavage site.⁹² Teramoto et al. showed that the design of small interfering RNA (siRNA) is greatly improved by using string kernels.⁹³ The siRNA sequence was decomposed into 1-, 2-, and 3-mer subsequences that were fed into the string kernel to compute the similarity between two sequences. Leslie and Kuang defined three new classes of k -mer string kernels, namely restricted gappy kernels, substitution kernels, and wildcard kernels, based on feature spaces defined by k -length subsequences from the protein sequence.⁹⁴ The new kernels were used for homology detection and protein classification. Tsuda and Noble used the diffusion kernel to predict protein functional classification from metabolic and protein-protein interaction networks.⁹⁵ The diffusion kernel is a

method of computing pair-wise distances between all nodes in a graph based on the sum of weighted paths between each pair of vertices.

Kernels for Molecular Structures

The common approach in SVM applications for property prediction based on molecular structure involves the computation of various classes of structural descriptors. These descriptors are used with various kernels to compute the structural similarity between two chemical compounds. Obviously, this approach reflects chemical bonding only in an indirect way, through descriptors. The molecular structure can be used directly in computing the pair-wise similarity of chemicals with tree and graph kernels as reviewed below.

Micheli, Portera, and Sperduti used acyclic molecular subgraphs and tree kernels to predict the ligand affinity for the benzodiazepine receptor.⁹⁶ Mahé et al. defined a series of graph kernels that can predict various properties from only the molecular graph and various atomic descriptors.⁹⁷ Jain et al. defined a new graph kernel based on the Schur–Hadamard inner product for a pair of molecular graphs, and they tested it by predicting the mutagenicity of aromatic and hetero-aromatic nitro compounds.⁹⁸ Finally, Lind and Maltseva used molecular fingerprints to compute the Tanimoto similarity kernel, which was incorporated into an SVM regression to predict the aqueous solubility of organic compounds.⁹⁹

PRACTICAL ASPECTS OF SVM CLASSIFICATION

Up to this point we have given mostly a theoretical presentation of SVM classification and regression; it is now appropriate to show some practical applications of support vector machines, together with practical guidelines for their application in cheminformatics and QSAR. In this section, we will present several case studies in SVM classification; the next section is dedicated to applications of SVM regression.

Studies investigating the universal approximation capabilities of support vector machines have demonstrated that SVM with usual kernels (such as polynomial, Gaussian RBF, or dot product kernels) can approximate any measurable or continuous function up to any desired accuracy.^{100,101} Any set of patterns can therefore be modeled perfectly if the appropriate kernel and parameters are used. The ability to approximate any measurable function is indeed required for a good nonlinear multivariate pattern recognition algorithm (artificial neural networks are also universal approximators), but from a practical point of view, more is required. Indeed, good QSAR or cheminformatics models must have an optimum predictivity (limited by the number of data, data distribution, noise, errors, selection of structural descriptors, etc.), not only a good mapping capability. For SVM classification problems, highly nonlinear

kernels can eventually separate perfectly the classes of patterns with intricate hyperplanes. This is what the universal approximation capabilities of SVM guarantees. However, these capabilities cannot promise that the resulting SVM will be optimally predictive. In fact, only empirical comparison with other classification algorithms (kNN, linear discriminant analysis, PLS, artificial neural networks, etc.) can demonstrate, for a particular problem, that SVM is better or worse than other classification methods. Indeed, the literature is replete with comparative studies showing that SVM can often, but not always, predict better than other methods. In many cases, the statistical difference between methods is not significant, and due to the limited number of samples used in those studies, one cannot prefer a method against others.

An instructive example to consider is the HIV-1 protease cleavage site prediction. This problem was investigated with neural networks,¹⁰² self-organizing maps,¹⁰³ and support vector machines.^{91,104} After an in-depth examination of this problem, Rögnavaldsson and You concluded that linear classifiers are at least as good predictors as are nonlinear algorithms.¹⁰⁵ The poor choice of complex, nonlinear classifiers could not deliver any new insight for the HIV-1 protease cleavage site prediction. The message of this story is simple and valuable: always compare nonlinear SVM models with linear models and, if possible, with other pattern recognition algorithms.

A common belief is that because SVM is based on structural risk minimization, its predictions are better than those of other algorithms that are based on empirical risk minimization. Many published examples show, however, that for real applications, such beliefs do not carry much weight and that sometimes other multivariate algorithms can deliver better predictions.

An important question to ask is as follows: Do SVMs overfit? Some reports claim that, due to their derivation from structural risk minimization, SVMs do not overfit. However, in this chapter, we have already presented numerous examples where the SVM solution is overfitted for simple datasets. More examples will follow. In real applications, one must carefully select the nonlinear kernel function needed to generate a classification hyperplane that is topologically appropriate and has optimum predictive power.

It is sometimes claimed that SVMs are better than artificial neural networks. This assertion is because SVMs have a unique solution, whereas artificial neural networks can become stuck in local minima and because the optimum number of hidden neurons of ANN requires time-consuming calculations. Indeed, it is true that multilayer feed-forward neural networks can offer models that represent local minima, but they also give constantly good solutions (although suboptimal), which is not the case with SVM (see examples in this section). Undeniably, for a given kernel and set of parameters, the SVM solution is unique. But, an infinite combination of kernels and SVM parameters exist, resulting in an infinite set of unique SVM models. The unique SVM solution therefore brings little comfort to the researcher because the theory cannot foresee which kernel and set of parameters are optimal for a

particular problem. And yes, artificial neural networks easily overfit the training data, but so do support vector machines.

Frequently the exclusive use of the RBF kernel is rationalized by mentioning that it is the best possible kernel for SVM models. The simple tests presented in this chapter (datasets from Tables 1–6) suggest that other kernels might be more useful for particular problems. For a comparative evaluation, we review below several SVM classification models obtained with five important kernels (linear, polynomial, Gaussian radial basis function, neural, and anova) and show that the SVM prediction capability varies significantly with the kernel type and parameters values used and that, in many cases, a simple linear model is more predictive than nonlinear kernels.

For all SVM classification models described later in this chapter, we have used the following kernels: dot (linear); polynomial (degree $d = 2, 3, 4, 5$); radial basis function, $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$, ($\gamma = 0.5, 1.0, 2.0$); neural (tanh), Eq. [68], ($a = 0.5, 1.0, 2.0$ and $b = 0, 1, 2$); anova, Eq. [69], ($\gamma = 0.5, 1.0, 2.0$ and $d = 1, 2, 3$). All SVM models were computed with mySVM, by Rüping, (<http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/>).

Predicting the Mechanism of Action for Polar and Nonpolar Narcotic Compounds

Because numerous organic chemicals can be environmental pollutants, considerable efforts were directed toward the study of the relationships between the structure of a chemical compound and its toxicity. Significant progress has been made in classifying chemical compounds according to their mechanism of toxicity and to screen them for their environmental risk. Predicting the mechanism of action (MOA) using structural descriptors has major applications in the selection of an appropriate quantitative structure–activity relationships (QSAR) model, to identify chemicals with similar toxicity mechanism, and in extrapolating toxic effects between different species and exposure regimes.^{106–109}

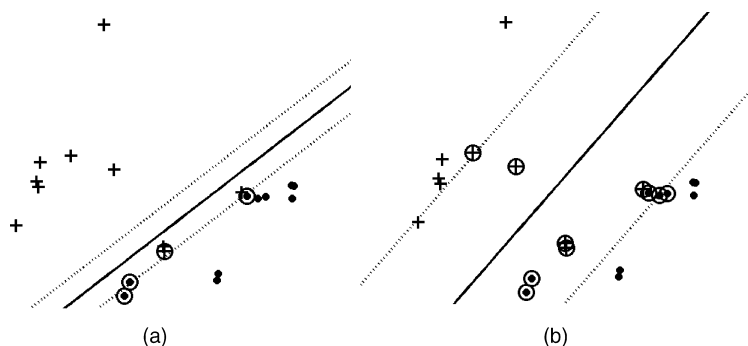
Organic compounds that act as narcotic pollutants are considered to disrupt the functioning of cell membranes. Narcotic pollutants are represented by two classes of compounds, namely nonpolar (MOA 1) and polar (MOA 2) compounds. The toxicity of both polar and nonpolar narcotic pollutants depends on the octanol–water partition coefficient, but the toxicity of polar compounds depends also on the propensity of forming hydrogen bonds. Ren used five structural descriptors to discriminate between 76 polar and 114 nonpolar pollutants.¹⁰⁷ These were the octanol–water partition coefficient $\log K_{ow}$, the energy of the highest occupied molecular orbital E_{HOMO} , the energy of the lowest unoccupied molecular orbital E_{LUMO} , the most negative partial charge on any non-hydrogen atom in the molecule Q^- , and the most positive partial charge on a hydrogen atom Q^+ . All quantum descriptors were computed with the AM1 method.

Table 7 Chemical Compounds, Theoretical Descriptors (E_{HOMO} , E_{LUMO} and Q^-), and Mechanism of Toxic Action (nonpolar, class +1; polar, class -1)

No	Compound	E_{HOMO}	E_{LUMO}	Q^-	MOA	Class
1	tetrachloroethene	-9.902	-0.4367	-0.0372	1	+1
2	1,2-dichloroethane	-11.417	0.6838	-0.1151	1	+1
3	1,3-dichloropropane	-11.372	1.0193	-0.1625	1	+1
4	dichloromethane	-11.390	0.5946	-0.1854	1	+1
5	1,2,4-trimethylbenzene	-8.972	0.5030	-0.2105	1	+1
6	1,1,2,2-tetrachloroethane	-11.655	-0.0738	-0.2785	1	+1
7	2,4-dichloroacetophenone	-9.890	-0.5146	-0.4423	1	+1
8	4-methyl-2-pentanone	-10.493	0.8962	-0.4713	1	+1
9	ethyl acetate	-11.006	1.1370	-0.5045	1	+1
10	cyclohexanone	-10.616	3.3960	-0.5584	1	+1
11	2,4,6-trimethylphenol	-8.691	0.4322	-0.4750	2	-1
12	3-chloronitrobenzene	-10.367	-1.2855	-0.4842	2	-1
13	4-ethylphenol	-8.912	0.4334	-0.4931	2	-1
14	2,4-dimethylphenol	-8.784	0.3979	-0.4980	2	-1
15	4-nitrotoluene	-10.305	-1.0449	-0.5017	2	-1
16	2-chloro-4-nitroaniline	-9.256	-0.9066	-0.6434	2	-1
17	2-chloroaniline	-8.376	0.3928	-0.6743	2	-1
18	pentafluoroaniline	-9.272	-1.0127	-0.8360	2	-1
19	4-methylaniline	-8.356	0.6156	-0.9429	2	-1
20	4-ethylaniline	-8.379	0.6219	-0.9589	2	-1

Using a descriptor selection procedure, we found that only three descriptors (E_{HOMO} , E_{LUMO} , and Q^-) are essential for the SVM model. To exemplify the shape of the classification hyperplane for polar and nonpolar narcotic pollutants, we selected 20 compounds (Table 7) as a test set (nonpolar compounds, class +1; polar compounds, class -1).

The first two experiments were performed with a linear kernel for $C = 100$ (Figure 49a) and $C = 1$ (Figure 49b). The first plot shows that this


Figure 49 SVM classification models with a dot (linear) kernel for the dataset from Table 7: (a) $C = 100$; (b) $C = 1$.

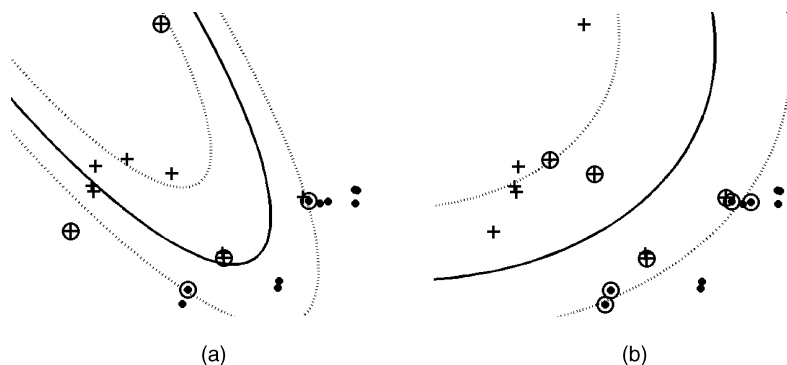


Figure 50 SVM classification models with a degree 2 polynomial kernel for the dataset from Table 7: (a) $C = 100$; (b) $C = 1$.

dataset can be separated with a linear classifier if some errors are accepted. Note that several +1 compounds cannot be classified correctly. A decrease of the capacity C shows a larger margin, with a border close to the bulk of class +1 compounds.

A similar analysis was performed for the degree 2 polynomial kernel with $C = 100$ (Figure 50a) and $C = 1$ (Figure 50b). The classification hyperplane is significantly different from that of the linear classifier, but with little success because three +1 compounds cannot be classified correctly. By decreasing the penalty for classification errors (Figure 50b), the margin increases and major changes appear in the shape of the classification hyperplane.

We will now show two SVMC models that are clearly overfitted. The first one is obtained with a degree 10 polynomial kernel (Figure 51a), whereas for the second, we used a B spline kernel (Figure 51b). The two classification

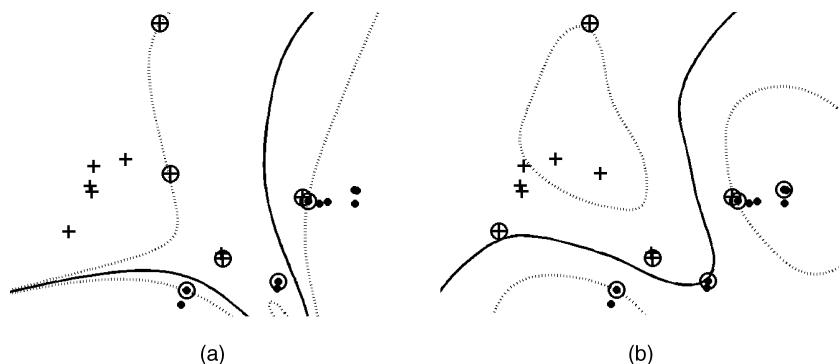


Figure 51 SVM classification models for the dataset from Table 7, with $C = 100$: (a) polynomial kernel, degree 10; (b) B spline kernel, degree 1.

hyperplanes are very complex, with a topology that clearly does not resemble that of the real data.

The statistics for all SVM models that were considered for this example are presented in Table 8. The calibration of the SVM models was performed with the whole set of 190 compounds, whereas the prediction was tested with a leave-20%-out cross-validation method. All notations are explained in the footnote of Table 8.

Table 8 shows that the SVMs with a linear kernel give very good results. The prediction accuracy from experiment 3 ($AC_p = 0.97$) is used to compare the performances of other kernels. The polynomial kernel (experiments 4-15) has AC_p between 0.93 and 0.96, which are results that do not equal those of the linear kernel. The overfitting of SVM models is clearly detected in several cases. For example, as the degree of the polynomial kernel increases from 2 to 5, AC_c increases from 0.97 to 1, whereas AC_p decreases from 0.96 to 0.93. The SVM models with perfect classification in training have the lowest prediction statistics.

The RBF kernel (experiments 16-24), with AC_p between 0.96 and 0.97, has better calibration statistics than the linear kernel, but its performance in prediction only equals that of the linear SVM. Although many tests were performed for the neural kernel (experiments 25-51), the prediction statistics are low, with AC_p between 0.64 and 0.88. This result is surprising, because the tanh function gives very good results in neural networks. Even the training statistics are low for the neural kernel, with AC_c between 0.68 and 0.89.

The last set of SVM models were obtained with the anova kernel (experiments 52-78), with AC_p between 0.94 and 0.98. In fact, only experiment 58 has a better prediction accuracy ($AC_p = 0.98$) than the linear SVM model from experiment 3. The linear SVM has six errors in prediction (all nonpolar compounds predicted to be polar), whereas the anova SVM has four prediction errors, also for nonpolar compounds.

Our experiments with various kernels show that the performance of the SVM classifier is strongly dependent on the kernel shape. Considering the results of the linear SVM as a reference, many nonlinear SVM models have lower prediction statistics. It is also true that the linear classifier does a good job and there is not much room for improvement. Out of the 75 nonlinear SVM models, only one, with the anova kernel, has slightly higher prediction statistics than the linear SVM.

Predicting the Mechanism of Action for Narcotic and Reactive Compounds

The second experiment we present in this tutorial for classifying compounds according to their mechanism of action involves the classifications of 88 chemicals. The chemicals are either narcotics (nonpolar and polar narcotics) and reactive compounds (respiratory uncouplers, soft electrophiles, and proelectrophiles).¹¹⁰ The dataset, consisting of 48 narcotic compounds

Table 8 Results for SVM Classification of Polar and Nonpolar Pollutants Using E_{HOMO} , E_{LUMO} , and Q^{-a}

Exp	C	K	TP _c	FN _c	TN _c	FP _c	SV _c	AC _c	TP _p	FN _p	TN _p	FP _p	SV _p	AC _p
1	10	L	105	9	76	0	27	0.95	104	10	76	0	22.2	0.95
2	100		106	8	76	0	25	0.96	104	10	76	0	20.2	0.95
3	1000		106	8	76	0	25	0.96	108	6	76	0	19.6	0.97
<i>d</i>														
4	10	P	109	5	75	1	21	0.97	108	6	75	1	18.0	0.96
5	100		109	5	76	0	20	0.97	108	6	74	2	15.2	0.96
6	1000		109	5	76	0	19	0.97	108	6	72	4	14.8	0.95
7	10		112	2	76	0	21	0.99	108	6	73	3	15.2	0.95
8	100		113	1	76	0	19	0.99	107	7	73	3	15.2	0.95
9	1000		114	0	76	0	18	1.00	106	8	73	3	14.4	0.94
10	10		112	2	76	0	22	0.99	106	8	73	3	17.0	0.94
11	100		114	0	76	0	20	1.00	106	8	72	4	15.8	0.94
12	1000		114	0	76	0	20	1.00	106	8	72	4	15.8	0.94
13	10		114	0	76	0	19	1.00	107	7	70	6	15.0	0.93
14	100		114	0	76	0	20	1.00	107	7	70	6	15.0	0.93
15	1000		114	0	76	0	20	1.00	107	7	70	6	15.0	0.93
<i>γ</i>														
16	10	R	109	5	76	0	26	0.97	107	7	75	1	23.6	0.96
17	100		112	2	76	0	20	0.99	108	6	74	2	17.0	0.96
18	1000		113	1	76	0	19	0.99	108	6	74	2	15.8	0.96
19	10		112	2	76	0	35	0.99	109	5	75	1	34.0	0.97
20	100		113	1	76	0	28	0.99	109	5	75	1	26.4	0.97
21	1000		114	0	76	0	21	1.00	109	5	75	1	21.8	0.97
22	10		113	1	76	0	45	0.99	109	5	74	2	44.8	0.96
23	100		114	0	76	0	43	1.00	109	5	75	1	40.8	0.97
24	1000		114	0	76	0	43	1.00	109	5	75	1	40.8	0.97

25	10	N	0.5	0.0	102	12	68	8	26	0.89	102	12	66	10	24.2	0.88
26	100		0.5	0.0	102	12	64	12	28	0.87	104	10	63	13	23.4	0.88
27	1000		0.5	0.0	102	12	64	12	28	0.87	103	11	62	14	22.0	0.87
28	10		1.0	0.0	98	16	60	16	34	0.83	95	19	61	15	30.6	0.82
29	100		1.0	0.0	98	16	60	16	34	0.83	100	14	56	20	31.4	0.82
30	1000		1.0	0.0	98	16	60	16	34	0.83	95	19	60	16	29.6	0.82
31	10		2.0	0.0	85	29	48	28	60	0.70	80	34	55	21	45.2	0.71
32	100		2.0	0.0	87	27	48	28	58	0.71	80	34	55	21	45.2	0.71
33	1000		2.0	0.0	85	29	47	29	60	0.69	86	28	48	28	47.6	0.71
34	10		0.5	1.0	95	19	53	23	53	0.78	92	22	52	24	41.4	0.76
35	100		0.5	1.0	92	22	53	23	49	0.76	89	25	51	25	39.4	0.74
36	1000		0.5	1.0	92	22	53	23	49	0.76	89	25	50	26	39.2	0.73
37	10		1.0	1.0	85	29	47	29	61	0.69	87	27	50	26	44.6	0.72
38	100		1.0	1.0	98	16	59	17	35	0.83	83	31	52	24	43.8	0.71
39	1000		1.0	1.0	98	16	59	17	35	0.83	84	30	46	30	48.0	0.68
40	10		2.0	1.0	86	28	43	33	64	0.68	86	28	50	26	35.6	0.72
41	100		2.0	1.0	86	28	43	33	64	0.68	94	20	55	21	26.6	0.78
42	1000		2.0	1.0	86	28	43	33	64	0.68	97	17	46	30	34.0	0.75
43	10		0.5	2.0	87	27	46	30	67	0.70	90	24	44	32	54.2	0.71
44	100		0.5	2.0	84	30	46	30	63	0.68	85	29	44	32	51.0	0.68
45	1000		0.5	2.0	84	30	46	30	62	0.68	84	30	44	32	50.2	0.67
46	10		1.0	2.0	83	31	45	31	64	0.67	71	43	50	26	52.0	0.64
47	100		1.0	2.0	83	31	45	31	64	0.67	82	32	45	31	51.6	0.67
48	1000		1.0	2.0	83	31	45	31	64	0.67	82	32	45	31	51.6	0.67
49	10		2.0	2.0	85	29	46	30	63	0.69	75	39	65	11	46.0	0.74
50	100		2.0	2.0	97	17	58	18	37	0.82	79	35	68	8	42.0	0.77
51	1000		2.0	2.0	97	17	58	18	37	0.82	82	32	65	11	38.2	0.77

	γ	A	1	110	4	76	0	26	0.98	106	8	75	1	22.0	0.95	
52	10	A	0.5	1	110	4	76	0	26	0.98	106	8	75	1	22.0	0.95
53	100		0.5	1	111	3	76	0	17	0.98	108	6	74	2	15.4	0.96

(continued)

Table 8 (Continued)

Exp	C	K	TP _c	FN _c	TN _c	FP _c	SV _c	AC _c	TP _p	FN _p	TN _p	FP _p	SV _p	AC _p
54	1000	0.5	112	2	76	0	14	0.99	109	5	73	3	13.2	0.96
55	10	1.0	111	3	76	0	26	0.98	109	5	75	1	20.4	0.97
56	100	1.0	111	3	76	0	18	0.98	110	4	74	2	16.0	0.97
57	1000	1.0	113	1	76	0	17	0.99	110	4	72	4	14.6	0.96
58	10	2.0	111	3	76	0	24	0.98	110	4	76	0	20.6	0.98
59	100	2.0	113	1	76	0	18	0.99	109	5	73	3	17.8	0.96
60	1000	2.0	114	0	76	0	14	1.00	109	5	70	6	15.2	0.94
61	10	0.5	112	2	76	0	24	0.99	107	7	75	1	18.4	0.96
62	100	0.5	112	2	76	0	20	0.99	108	6	74	2	16.8	0.96
63	1000	0.5	114	0	76	0	15	1.00	107	7	74	2	14.2	0.95
64	10	1.0	112	2	76	0	21	0.99	108	6	75	1	18.8	0.96
65	100	1.0	114	0	76	0	20	1.00	107	7	73	3	16.6	0.95
66	1000	1.0	114	0	76	0	20	1.00	107	7	73	3	16.6	0.95
67	10	2.0	114	0	76	0	24	1.00	108	6	73	3	24.6	0.95
68	100	2.0	114	0	76	0	22	1.00	108	6	73	3	23.0	0.95
69	1000	2.0	114	0	76	0	22	1.00	108	6	73	3	23.0	0.95
70	10	0.5	112	2	76	0	21	0.99	108	6	74	2	17.0	0.96
71	100	0.5	114	0	76	0	17	1.00	107	7	73	3	15.4	0.95
72	1000	0.5	114	0	76	0	17	1.00	107	7	73	3	15.4	0.95
73	10	1.0	114	0	76	0	20	1.00	107	7	74	2	20.4	0.95
74	100	1.0	114	0	76	0	20	1.00	107	7	74	2	20.4	0.95
75	1000	1.0	114	0	76	0	20	1.00	107	7	74	2	20.4	0.95
76	10	2.0	114	0	76	0	38	1.00	108	6	74	2	37.2	0.96
77	100	2.0	114	0	76	0	38	1.00	108	6	74	2	37.2	0.96
78	1000	2.0	114	0	76	0	38	1.00	108	6	74	2	37.2	0.96

^a The table reports the experiment number Exp, capacity parameter C, kernel type K (linear L; polynomial P; radial basis function R; neural N; anova A), and corresponding parameters, calibration results (TP_c, true positive in calibration; FN_c, false negative in calibration; TN_c, true negative in calibration; FP_c, false positive in calibration; SV_c, number of support vectors in calibration; AC_c, calibration accuracy), and L20%O prediction results (TP_p, true positive in prediction; FN_p, false negative in prediction; TN_p, true negative in prediction; FP_p, false positive in prediction; SV_p, average number of support vectors in prediction; AC_p, prediction accuracy).

(class +1) and 40 reactive compounds (class -1), was taken from two recent studies.^{108,111} Four theoretical descriptors are used to discriminate between their mechanism of action, namely the octanol–water partition coefficient $\log K_{ow}$, the energy of the highest occupied molecular orbital E_{HOMO} , the energy of the lowest unoccupied molecular orbital E_{LUMO} , and the average acceptor superdelocalizability S_{av}^N . The prediction power of each SVM model was evaluated with a leave-10%-out cross-validation procedure.

The best prediction statistics for each kernel type are presented here: linear, $C = 1000$, $AC_p = 0.86$; polynomial, degree 2, $C = 10$, $AC_p = 0.92$; RBF, $C = 100$, $\gamma = 0.5$, $AC_p = 0.83$; neural, $C = 10$, $a = 0.5$, $b = 0$, $AC_p = 0.78$; and anova, $C = 10$, $\gamma = 0.5$, $d = 1$, $AC_p = 0.87$. These results indicate that a degree 2 polynomial is a good separation hyperplane between narcotic and reactive compounds. The neural and RBF kernels have worse predictions than does the linear SVM model, whereas the anova kernel has similar AC_p with the linear model.

Predicting the Mechanism of Action from Hydrophobicity and Experimental Toxicity

This exercise for classifying compounds according to their mechanism of action uses as input data the molecule's hydrophobicity and experimental toxicity against *Pimephales promelas* and *Tetrahymena pyriformis*.¹¹² SVM classification was applied for a set of 337 organic compounds from eight MOA classes (126 nonpolar narcotics, 79 polar narcotics, 23 ester narcotics, 13 amine narcotics, 13 weak acid respiratory uncouplers, 69 electrophiles, 8 proelectrophiles, and 6 nucleophiles).¹¹³ The MOA classification was based on three indices taken from a QSAR study by Ren, Frymier, and Schultz¹¹³ namely: $\log K_{ow}$, the octanol–water partition coefficient; $\log 1/IGC_{50}$, the 50% inhibitory growth concentration against *Tetrahymena pyriformis*; and $\log 1/LC_{50}$, the 50% lethal concentration against *Pimephales promelas*. The prediction power of each SVM model was evaluated with a leave-5%-out (L5%O) cross-validation procedure.

In the first test we used SVM models to discriminate between nonpolar narcotic compounds (chemicals that have baseline toxicity) and other compounds having excess toxicity (representing the following MOAs: polar narcotics, ester narcotics, amine narcotics, weak acid respiratory uncouplers, electrophiles, proelectrophiles, and nucleophiles). From the total set of 337 compounds, 126 represent the SVM class +1 (nonpolar narcotic) and 211 represent the SVM class -1 (all other MOA classes).

The best cross-validation results for each kernel type are presented in Table 9. The linear, polynomial, RBF, and anova kernels have similar results that are of reasonably quality, whereas the neural kernel has very bad statistics; the slight classification improvement obtained for the RBF and anova kernels is not statistically significant.

Table 9 SVM Classification of Nonpolar Narcotic Compounds (SVM class +1) From Other Compounds (SVM class -1) Using as Descriptors $\log K_{ow}$, $\log 1/IGC_{50}$ and $\log 1/LC_{50}$

Kernel	TP _c	FN _c	TN _c	FP _c	SV _c	AC _c	TP _p	FN _p	TN _p	FP _p	SV _p	AC _p
L	78	48	186	25	195	0.78	79	47	186	25	185.8	0.79
P, $d = 2$	81	45	185	26	176	0.79	82	44	184	27	165.8	0.79
R, $\gamma = 1.0$	97	29	190	21	172	0.85	89	37	180	31	165.1	0.80
N, $a = 0.5, b = 0$	75	51	98	113	158	0.51	49	77	130	81	152.1	0.53
A, $\gamma = 0.5, d = 2$	95	31	190	21	169	0.85	87	39	182	29	119.2	0.80

The chemicals exhibiting excess toxicity belong to seven MOA classes, and their toxicity has a wide range of variation. For these molecules, it is useful to further separate them as being less-reactive and more-reactive compounds. In the second test, we have developed SVM models that discriminate between less-reactive compounds (SVM class +1, formed by polar narcotics, ester narcotics, amine narcotics) and more-reactive compounds (SVM class -1, formed by weak acid respiratory uncouplers, electrophiles, proelectrophiles, and nucleophiles). From the total of 211 compounds with excess toxicity, 115 are less-reactive and 96 are more-reactive compounds.

In Table 10, we show the best cross-validation results for each kernel type. The radial kernel has the best predictions, followed by the linear SVM model. The remaining kernels have worse predictions than does the linear model.

Classifying the Carcinogenic Activity of Polycyclic Aromatic Hydrocarbons

Structure-activity relationships are valuable statistical models that can be used for predicting the carcinogenic potential of new chemicals and for the interpretation of the short-term tests of genotoxicity, long-term tests of carcinogenicity in rodents, and epidemiological data. We show here an SVM application for identifying the carcinogenic activity of a group of methylated and nonmethylated polycyclic aromatic hydrocarbons (PAHs).¹¹⁴ The dataset

Table 10 SVM Classification of Less-Reactive Compounds and More-Reactive Compounds

Kernel	TP _c	FN _c	TN _c	FP _c	SV _c	AC _c	TP _p	FN _p	TN _p	FP _p	SV _p	AC _p
L	97	18	50	46	151	0.70	97	18	46	50	144.2	0.68
P, $d = 2$	101	14	38	58	154	0.66	97	18	38	58	144.0	0.64
R, $\gamma = 2.0$	107	8	77	19	141	0.87	94	21	55	41	133.2	0.71
N, $a = 2, b = 1$	71	44	51	45	91	0.58	64	51	59	37	90.5	0.58
A, $\gamma = 0.5, d = 2$	109	6	77	19	112	0.88	85	30	57	39	105.8	0.67

consists of 32 PAHs and 46 methylated PAHs taken from literature.^{115–118} From this set of 78 aromatic hydrocarbons, 34 are carcinogenic and 44 are noncarcinogenic. The carcinogenic activity was predicted by using the following four theoretical descriptors computed with the PM3 semiempirical method: energy of the highest occupied molecular orbital E_{HOMO} ; energy of the lowest unoccupied molecular orbital E_{LUMO} ; hardness HD, where $\text{HD} = (E_{\text{LUMO}} - E_{\text{HOMO}})/2$; and difference between E_{HOMO} and $E_{\text{HOMO}-1}$ denoted ΔH .¹¹⁷ The prediction power of each SVM model was evaluated with a leave-10%-out cross-validation procedure.

The best prediction statistics for each kernel type are presented here: linear, $\text{AC}_p = 0.76$; polynomial, degree 2, $\text{AC}_p = 0.82$; RBF, $\gamma = 0.5$, $\text{AC}_p = 0.86$; neural, $a = 2$, $b = 0$, $\text{AC}_p = 0.66$; and anova, $\gamma = 0.5$, $d = 1$, $\text{AC}_p = 0.84$ ($C = 10$ for these SVM models). The relationship between the quantum indices and PAH carcinogenicity is nonlinear, as evidenced by the increase in prediction power when going from a linear to an RBF kernel.

Structure-Odor Relationships for Pyrazines

Various techniques of molecular design can significantly help fragrance researchers to find relationships between the chemical structure and the odor of organic compounds.^{119–124} A wide variety of structural descriptors (molecular fragments, topological indices, geometric descriptors, or quantum indices) and a broad selection of qualitative or quantitative statistical equations have been used to model and predict the aroma (and its intensity) for various classes of organic compounds. Besides providing an important guide for the synthesis of new fragrances, structure-odor relationships (SOR) offer a better understanding of the mechanism of odor perception.

We illustrate the application of support vector machines for aroma classification using as our example 98 tetra-substituted pyrazines (Figure 52) representing three odor classes, namely 32 green, 23 nutty, and 43 bell-pepper.¹²⁵ The prediction power of each SVM model was evaluated with a leave-10%-out cross-validation procedure.¹²⁶ This multiclass dataset was modeled with an one-versus-all approach.

In the first classification test, class +1 contained green aroma compounds and class -1 contained compounds with nutty or bell-pepper aroma. The best prediction statistics for each kernel type are linear, $C = 10$, $\text{AC}_p = 0.80$; polynomial, degree 2, $C = 1000$, $\text{AC}_p = 0.86$; RBF, $C = 10$, $\gamma = 0.5$, $\text{AC}_p = 0.79$; neural, $C = 10$, $a = 0.5$, $b = 0$, $\text{AC}_p = 0.73$; and anova, $C = 100$, $\gamma = 0.5$,

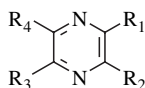


Figure 52 General structure of pyrazines.

$d = 1$, $AC_p = 0.84$. A degree 2 polynomial kernel has the best prediction, followed by the anova kernel and the linear model.

In the second test, class +1 contained compounds with nutty aroma, whereas the remaining pyrazines formed the class -1. The prediction statistics show a slight advantage for the anova kernel, whereas the linear, polynomial, and RBF kernels have identical results: linear, $C = 10$, $AC_p = 0.89$; polynomial, degree 2, $C = 10$, $AC_p = 0.89$; RBF, $C = 10$, $\gamma = 0.5$, $AC_p = 0.89$; neural, $C = 100$, $a = 0.5$, $b = 0$, $AC_p = 0.79$; and anova, $C = 100$, $\gamma = 0.5$, $d = 1$, $AC_p = 0.92$.

Finally, compounds with bell-pepper aroma were considered to be in class +1, whereas green and nutty pyrazines formed the class -1. Three kernels (RBF, polynomial, and anova) give much better predictions than does the linear SVM classifier: linear, $C = 10$, $AC_p = 0.74$; polynomial, degree 2, $C = 10$, $AC_p = 0.88$; RBF, $C = 10$, $\gamma = 0.5$, $AC_p = 0.89$; neural, $C = 100$, $a = 2$, $b = 1$, $AC_p = 0.68$; and anova, $C = 10$, $\gamma = 0.5$, $d = 1$, $AC_p = 0.87$. We have to notice that the number of support vectors depends on the kernel type (linear, $SV = 27$; RBF, $SV = 43$; anova, $SV = 31$; all for training with all compounds), so for this structure-odor model, one might prefer the SVM model with a polynomial kernel that is more compact, i.e., contains a lower number of support vectors.

In this section, we compared the prediction capabilities of five kernels, namely linear, polynomial, Gaussian radial basis function, neural, and anova. Several guidelines that might help the modeler obtain a predictive SVM model can be extracted from these results: (1) It is important to compare the predictions of a large number of kernels and combinations of parameters; (2) the linear kernel should be used as a reference to compare the results from nonlinear kernels; (3) some datasets can be separated with a linear hyperplane; in such instances, the use of a nonlinear kernel should be avoided; and (4) when the relationships between input data and class attribution are nonlinear, RBF kernels do not necessarily give the optimum SVM classifier.

PRACTICAL ASPECTS OF SVM REGRESSION

Support vector machines were initially developed for class discrimination, and most of their applications have been for pattern classification. SVM classification is especially relevant for important cheminformatics problems, such as recognizing drug-like compounds, or discriminating between toxic and nontoxic compounds, and many such applications have been published. The QSAR applications of SVM regression, however, are rare, and this is unfortunate because it represents a viable alternative to multiple linear regression, PLS, or neural networks. In this section, we present several SVMR applications to QSAR datasets, and we compare the performance of several kernels.

The SVM regression models we present below implement the following kernels: linear; polynomial (degree $d = 2, 3, 4, 5$); radial basis function, $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$, ($\gamma = 0.25, 0.5, 1.0, 1.5, 2.0$); neural (tanh), Eq. [68], ($a = 0.5, 1.0, 2.0$ and $b = 0, 1, 2$); and anova, Eq. [69], ($\gamma = 0.25,$

0.5, 1.0, 1.5, 2.0 and $d = 1, 2, 3$). All SVM models were computed with mySVM, by Rüping, (<http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/>).

SVM Regression QSAR for the Phenol Toxicity to *Tetrahymena pyriformis*

Aptula et al. used multiple linear regression to investigate the toxicity of 200 phenols to the ciliated protozoan *Tetrahymena pyriformis*.¹²⁷ Using their MLR model, they then predicted the toxicity of another 50 phenols. Here we present a comparative study for the entire set of 250 phenols, using multiple linear regression, artificial neural networks, and SVM regression methods.¹²⁸ Before computing the SVM model, the input vectors were scaled to zero mean and unit variance. The prediction power of the QSAR models was tested with complete cross-validation: leave-5%-out (L5%O), leave-10%-out (L10%O), leave-20%-out (L20%O), and leave-25%-out (L25%O). The capacity parameter C was optimized for each SVM model.

Seven structural descriptors were used to model the 50% growth inhibition concentration, IGC_{50} . These descriptors are $\log D$, where D is the dissociation constant (i.e., the octanol-water partition coefficient corrected for ionization); E_{LUMO} , the energy of the lowest unoccupied molecular orbital; MW, the molecular weight; P_{NEG} , the negatively charged molecular surface area; ABSQon, the sum of absolute charges on nitrogen and oxygen atoms; MaxHp, the largest positive charge on a hydrogen atom; and SsOH, the electrotopological state index for the hydroxy group. The MLR model has a calibration correlation coefficient of 0.806 and is stable to cross-validation experiments:

$$\begin{aligned} pIGC_{50} = & -0.154(\pm 0.080) + 0.296(\pm 0.154)\log D - 0.352(\pm 0.183)E_{LUMO} \\ & + 0.00361(\pm 0.00188)MW - 0.0218(\pm 0.0113)P_{NEG} - 0.446(\pm 0.232) \\ & ABSQon + 1.993(\pm 1.037)MaxHp + 0.0265(\pm 0.0138)SsOH \end{aligned} \quad [106]$$

$$\begin{aligned} n = 250 \quad r_{cal} = 0.806 \quad RMSE_{cal} = 0.49 \quad s_{cal} = 0.50 \quad F_{cal} = 64 \\ r_{LOO} = 0.789 \quad q_{LOO}^2 = 0.622 \quad RMSE_{LOO} = 0.51 \\ r_{L5\%O} = 0.785 \quad q_{L5\%O}^2 = 0.615 \quad RMSE_{L5\%O} = 0.51 \\ r_{L10\%O} = 0.786 \quad q_{L10\%O}^2 = 0.617 \quad RMSE_{L10\%O} = 0.51 \\ r_{L20\%O} = 0.775 \quad q_{L20\%O}^2 = 0.596 \quad RMSE_{L20\%O} = 0.53 \\ r_{L25\%O} = 0.788 \quad q_{L25\%O}^2 = 0.620 \quad RMSE_{L25\%O} = 0.51 \end{aligned}$$

Based on the cross-validation statistics, the best ANN model has tanh functions for both hidden and output neurons, and it has only one hidden neuron. The statistics for this ANN are $r_{cal} = 0.824$, $RMSE_{cal} = 0.47$; $r_{L5\%O} = 0.804$, $q_{L5\%O}^2 = 0.645$, $RMSE_{L5\%O} = 0.49$; $r_{L10\%O} = 0.805$, $q_{L10\%O}^2 = 0.647$, $RMSE_{L10\%O} = 0.49$, $r_{L20\%O} = 0.802$, $q_{L20\%O}^2 = 0.642$, $RMSE_{L20\%O} = 0.50$;

and $r_{L25\%O} = 0.811$, $q_{L25\%O}^2 = 0.657$, $RMSE_{L25\%O} = 0.48$. On the one hand, the ANN statistics are better than those obtained with MLR, indicating that there is some nonlinearity between $pIGC_{50}$ and the seven structural descriptors. On the other hand, the predictions statistics for ANN models with two or more hidden neurons decrease, indicating that the dataset has a high level of noise or error.

The SVM regression results for the prediction of phenol toxicity to *Tetrahymena pyriformis* are presented in Tables 11 and 12. In calibration

Table 11 Kernel Type and Corresponding Parameters for Each SVM Model^a

Exp	Kernel		C_{opt}	SV_{cal}	r_{cal}	$RMSE_{cal}$	$r_{L5\%O}$	$q_{L5\%O}^2$	$RMSE_{L5\%O}$
1	L		64.593	250	0.803	0.51	0.789	0.593	0.53
2	P	2	88.198	250	0.853	0.44	0.787	0.591	0.53
3	P	3	64.593	243	0.853	0.45	0.326	-3.921	1.83
4	P	4	73.609	248	0.993	0.09	0.047	<-100	>10
5	P	5	88.198	250	0.999	0.04	0.137	<-100	>10
6	R	0.25	88.198	250	0.983	0.15	0.694	0.330	0.68
7	R	0.5	88.198	250	0.996	0.08	0.660	0.303	0.69
8	R	1.0	88.198	250	1.000	0.01	0.668	0.428	0.63
9	R	1.5	88.198	250	1.000	0.00	0.659	0.433	0.62
10	R	2.0	64.593	250	1.000	0.00	0.636	0.400	0.64
11	N	0.5 0.0	0.024	250	0.748	0.56	0.743	0.536	0.56
12	N	1.0 0.0	0.016	250	0.714	0.59	0.722	0.506	0.58
13	N	2.0 0.0	0.016	250	0.673	0.61	0.696	0.474	0.60
14	N	0.5 1.0	0.020	250	0.691	0.60	0.709	0.483	0.59
15	N	1.0 1.0	0.012	250	0.723	0.61	0.706	0.468	0.60
16	N	2.0 1.0	0.015	248	0.688	0.61	0.678	0.440	0.62
17	N	0.5 2.0	0.020	250	0.642	0.64	0.614	0.374	0.65
18	N	1.0 2.0	0.015	250	0.703	0.62	0.670	0.429	0.62
19	N	2.0 2.0	0.012	250	0.695	0.62	0.586	0.343	0.67
20	A	0.25 1	88.198	250	0.842	0.46	0.718	0.433	0.62
21	A	0.5 1	88.198	250	0.857	0.43	0.708	0.414	0.63
22	A	1.0 1	88.198	250	0.868	0.42	0.680	0.348	0.67
23	A	1.5 1	88.198	250	0.880	0.40	0.674	0.323	0.68
24	A	2.0 1	88.198	250	0.884	0.40	0.688	0.360	0.66
25	A	0.25 2	88.198	250	0.977	0.18	0.531	-0.760	1.10
26	A	0.5 2	88.198	250	0.994	0.09	0.406	-1.595	1.33
27	A	1.0 2	88.198	250	1.000	0.01	0.436	-1.182	1.22
28	A	1.5 2	88.198	250	1.000	0.00	0.492	-0.512	1.02
29	A	2.0 2	64.593	250	1.000	0.00	0.542	-0.141	0.88
30	A	0.25 3	88.198	250	0.999	0.04	0.312	-4.199	1.89
31	A	0.5 3	64.593	250	1.000	0.00	0.506	-0.781	1.10
32	A	1.0 3	64.593	250	1.000	0.00	0.625	0.134	0.77
33	A	1.5 3	64.593	250	1.000	0.00	0.682	0.377	0.65
34	A	2.0 3	64.593	250	1.000	0.00	0.708	0.461	0.61

^a Notations: Exp, experiment number; r_{cal} , calibration correlation coefficient; $RMSE_{cal}$, calibration root mean square error; $r_{L5\%O}$, leave-5%-out correlation coefficient; $q_{L5\%O}^2$, leave-5%-out q^2 ; $RMSE_{L5\%O}$, leave-5%-out root-mean-square error; L, linear kernel; P, polynomial kernel (parameter: degree d); R, radial basis function kernel (parameter: γ); N, neural kernel (parameters: a and b); and A, anova kernel (parameters: γ and d).

Table 12 Support Vector Regression Statistics for Leave-10%-out, Leave-20%-out, and Leave-25%-out Cross-validation Tests^a

Exp	Kernel	RMS-			RMS-			RMS-		
		$r_{L10\%O}$	$q_{L10\%O}^2$	$E_{L10\%O}$	$r_{L20\%O}$	$q_{L20\%O}^2$	$E_{L20\%O}$	$r_{L25\%O}$	$q_{L25\%O}^2$	$E_{L25\%O}$
1	L	0.789	0.593	0.53	0.786	0.588	0.53	0.790	0.589	0.53
2	P	0.784	0.586	0.53	0.746	0.495	0.59	0.762	0.501	0.58
3	P	0.316	-3.915	1.83	0.142	-14.254	3.23	0.116	-11.734	2.95
4	P	-0.008	<-100	>10	-0.055	<-100	>10	0.059	<-100	>10
5	P	0.035	<-100	>10	0.196	<-100	>10	0.069	<-100	>10
6	R	0.676	0.307	0.69	0.684	0.291	0.70	0.647	0.238	0.72
7	R	0.663	0.339	0.67	0.650	0.288	0.70	0.629	0.301	0.69
8	R	0.662	0.424	0.63	0.673	0.440	0.62	0.626	0.381	0.65
9	R	0.650	0.422	0.63	0.662	0.438	0.62	0.595	0.353	0.67
10	R	0.628	0.390	0.65	0.640	0.405	0.64	0.561	0.312	0.69
11	N	0.737	0.530	0.57	0.744	0.542	0.56	0.743	0.534	0.56
12	N	0.719	0.503	0.58	0.715	0.497	0.59	0.716	0.497	0.59
13	N	0.685	0.460	0.61	0.689	0.464	0.61	0.700	0.478	0.60
14	N	0.714	0.491	0.59	0.704	0.470	0.60	0.701	0.474	0.60
15	N	0.689	0.451	0.61	0.705	0.452	0.61	0.709	0.470	0.60
16	N	0.684	0.443	0.62	0.661	0.430	0.62	0.624	0.381	0.65
17	N	0.610	0.369	0.66	0.629	0.393	0.64	0.630	0.394	0.64
18	N	0.678	0.436	0.62	0.683	0.443	0.62	0.678	0.436	0.62
19	N	0.682	0.430	0.62	0.683	0.430	0.62	0.528	0.255	0.71
20	A	0.725	0.457	0.61	0.724	0.465	0.60	0.634	0.241	0.72
21	A	0.723	0.458	0.61	0.730	0.480	0.60	0.601	0.148	0.76
22	A	0.684	0.367	0.66	0.655	0.300	0.69	0.624	0.230	0.73
23	A	0.694	0.373	0.65	0.670	0.333	0.68	0.613	0.152	0.76
24	A	0.703	0.397	0.64	0.675	0.351	0.67	0.621	0.158	0.76
25	A	0.493	-0.877	1.13	0.423	-1.871	1.40	0.378	-1.626	1.34
26	A	0.351	-1.850	1.40	0.335	-2.174	1.47	0.366	-1.465	1.30
27	A	0.349	-1.390	1.28	0.404	-1.103	1.20	0.454	-0.798	1.11
28	A	0.471	-0.439	0.99	0.516	-0.285	0.94	0.523	-0.294	0.94
29	A	0.549	-0.057	0.85	0.577	0.023	0.82	0.569	-0.043	0.84
30	A	0.282	-4.289	1.90	0.354	-3.835	1.82	0.360	-3.149	1.68
31	A	0.449	-1.050	1.18	0.462	-1.040	1.18	0.528	-0.601	1.05
32	A	0.597	0.087	0.79	0.609	0.136	0.77	0.633	0.171	0.75
33	A	0.671	0.365	0.66	0.678	0.384	0.65	0.671	0.347	0.67
34	A	0.703	0.457	0.61	0.707	0.468	0.60	0.686	0.412	0.63

^a Notations for the cross-validation statistical indices: $r_{L10\%O}$, $q_{L10\%O}^2$, and $RMSE_{L10\%O}$ for leave-10%-out; $r_{L20\%O}$, $q_{L20\%O}^2$, and $RMSE_{L20\%O}$ for leave-20%-out; and $r_{L25\%O}$, $q_{L25\%O}^2$, and $RMSE_{L25\%O}$ for leave-25%-out.

(fitting), the results are much better than those obtained with either MLR or ANN. In fact, several SVMR models have a perfect correlation, with $r_{cal} = 1$. These include experiments 8–10 with RBF and experiments 27–29 and 31–34 with the anova kernel. As expected, the SVMR with a linear kernel has predictions slightly higher than those of the MLR QSAR. However, the SVM regression models with nonlinear kernels have worse predictions. We compare here only $r_{L25\%O}$ for all QSAR models (for each kernel type, only the best prediction statistics are given): MLR, 0.788; ANN, 0.811; SVMR linear, 0.790; SVMR

polynomial, 0.762; SVMR RBF, 0.647; SVMR neural, 0.743; and SVMR anova, 0.671. The results presented here seem to indicate that SVM regression can fit the data, but the prediction is not reliable.

SVM Regression QSAR for Benzodiazepine Receptor Ligands

Benzodiazepine receptor (BzR) ligands (either benzodiazepines or structurally unrelated chemical compounds) act as modulators of γ -aminobutyric acid (GABA) binding to its receptor, by altering the transmembrane chloride ion conductance.^{129–131} The interest for developing new BzR ligands is stimulated by their ability to induce a wide spectrum of central nervous system effects, from full agonism through antagonism to inverse agonism.

In this exercise, we compare MLR and SVMR QSAR models for the benzodiazepine receptor affinity of 52 2-aryl(heteroaryl)-2,5-dihydropyrazolo[4,3-*c*]quinolin-3-(3*H*)-ones (Figure 53).¹³² Both models were developed with five structural descriptors, namely the Hammett electronic parameter $\sigma_{R'}$, the molar refractivity MR_{R8} , the Sterimol parameter $L_{R'4'}$, an indicator variable I (1 or 0) for 7-substituted compounds, and the Sterimol parameter B_{5R} .¹³⁰ The MLR model has a calibration correlation coefficient of 0.798 and fairly good prediction ability:

$$\log 1/IC_{50} = 11.538 (\pm 2.869) - 2.320 (\pm 0.577) \sigma_{R'} - 0.294 (\pm 0.073) MR_{R8} - 0.326 (\pm 0.081) L_{R'4'} - 0.560 (\pm 0.139) I - 1.795 (\pm 0.446) B_{5R} \quad [107]$$

$$n = 52 \quad r_{\text{cal}} = 0.798 \quad RMSE_{\text{cal}} = 0.69 \quad s_{\text{cal}} = 0.73 \quad F_{\text{cal}} = 16.18$$

$$r_{\text{LOO}} = 0.721 \quad q_{\text{LOO}}^2 = 0.481 \quad RMSE_{\text{LOO}} = 0.82$$

$$r_{\text{L5\%O}} = 0.716 \quad q_{\text{L5\%O}}^2 = 0.458 \quad RMSE_{\text{L5\%O}} = 0.84$$

$$r_{\text{L10\%O}} = 0.711 \quad q_{\text{L10\%O}}^2 = 0.448 \quad RMSE_{\text{L10\%O}} = 0.85$$

$$r_{\text{L20\%O}} = 0.733 \quad q_{\text{L20\%O}}^2 = 0.502 \quad RMSE_{\text{L20\%O}} = 0.81$$

$$r_{\text{L25\%O}} = 0.712 \quad q_{\text{L25\%O}}^2 = 0.470 \quad RMSE_{\text{L25\%O}} = 0.83$$

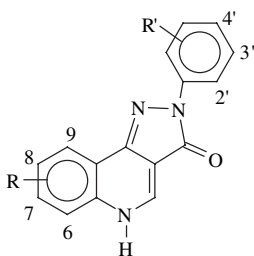


Figure 53 General formula for the pyrazolo[4,3-*c*]quinolin-3-ones.

Table 13 Best SVMR Predictions for the Benzodiazepine Receptor Ligands QSAR^a

Exp	Kernel	$r_{L5\%O}$	$q_{L5\%O}^2$	RMSE _{L5%O}	$r_{L10\%O}$	$q_{L10\%O}^2$	RMSE _{L10%O}
1	L	0.667	0.261	0.98	0.672	0.273	0.97
2	P	-0.270	<-100	>10	-0.265	<-100	>10
6	R	0.665	0.368	0.91	0.676	0.370	0.91
12	N	0.665	0.416	0.87	0.659	0.411	0.87
24	A	0.641	0.293	0.96	0.653	0.339	0.93
Exp	Kernel	$r_{L20\%O}$	$q_{L20\%O}^2$	RMSE _{L20%O}	$r_{L25\%O}$	$q_{L25\%O}^2$	RMSE _{L25%O}
1	L	0.674	0.228	1.00	0.667	0.189	1.03
2	P	-0.277	<-100	>10	0.332	<-100	>10
6	R	0.633	0.317	0.94	0.680	0.344	0.92
12	N	0.670	0.418	0.87	0.691	0.432	0.86
24	A	0.632	0.155	1.05	0.675	0.376	0.90

^a See Table 11 for the parameters of each kernel.

In Table 13, we present the best regression predictions for each kernel. Despite the large number of SVMR experiments we carried out for this QSAR (34 total), the cross-validation statistics of the SVM models are well below those obtained with MLR.

SVM Regression QSAR for the Toxicity of Aromatic Compounds to *Chlorella vulgaris*

Toxicity testing on model systems, short-term assays, and predictions from quantitative structure-toxicity models are inexpensive and fast methods to screen and prioritize chemicals for more elaborate, more expensive, and time-consuming toxicity evaluations. A novel short-term toxicity assay using the unicellular green alga *Chlorella vulgaris* was proposed recently.^{133–136} That assay used fluorescein diacetate, which is metabolized to fluorescein. The appearance of fluorescein, after 15 minutes of exposure to the toxicant, was measured fluorimetrically. The concentration causing a 50% decrease in fluorescence, EC₅₀ (mM), as compared with a control, was determined for 65 aromatic compounds.¹³³ These experimental data were used to develop a quantitative structure-toxicity model with four structural descriptors: log K_{ow} , octanol–water partition coefficient; E_{LUMO} , AM1 energy of the lowest unoccupied molecular orbital; A_{max} , maximum acceptor superdelocalizability; and ${}^0\chi^v$, Kier–Hall connectivity index. QSAR models for predicting the toxicity of aromatic compounds to *Chlorella vulgaris*, using MLR, ANN, and SVMR have been evaluated.¹³⁷ The MLR results, presented below, show that these four descriptors are useful in predicting log 1/EC₅₀:

$$\begin{aligned} \log 1/EC_{50} = & -4.483 (\pm 0.461) + 0.5196 (\pm 0.0534) \log K_{ow} \\ & - 0.3425 (\pm 0.0352) E_{LUMO} + 7.260 (\pm 0.746) A_{max} \\ & + 0.1375 (\pm 0.0141) {}^0\chi^v \end{aligned} \quad [108]$$

$$\begin{aligned}
 n = 65 \quad r_{\text{cal}} = 0.929 \quad \text{RMSE}_{\text{cal}} = 0.39 \quad S_{\text{cal}} = 0.40 \quad F_{\text{cal}} = 94.76 \\
 r_{\text{LOO}} = 0.913 \quad q_{\text{LOO}}^2 = 0.834 \quad \text{RMSE}_{\text{LOO}} = 0.43 \\
 r_{\text{L5\%O}} = 0.910 \quad q_{\text{L5\%O}}^2 = 0.826 \quad \text{RMSE}_{\text{L5\%O}} = 0.44 \\
 r_{\text{L10\%O}} = 0.909 \quad q_{\text{L10\%O}}^2 = 0.826 \quad \text{RMSE}_{\text{L10\%O}} = 0.44 \\
 r_{\text{L20\%O}} = 0.910 \quad q_{\text{L20\%O}}^2 = 0.828 \quad \text{RMSE}_{\text{L20\%O}} = 0.43 \\
 r_{\text{L25\%O}} = 0.919 \quad q_{\text{L25\%O}}^2 = 0.844 \quad \text{RMSE}_{\text{L25\%O}} = 0.41
 \end{aligned}$$

The ANN cross-validation results show that using more than one hidden neuron results in decreased predictive power. Because it is very time consuming, the LOO procedure was not tested with ANN. Therefore, as the best model for the neural network QSAR, we selected the one with a single hidden neuron and with tanh functions for both hidden and output neurons: $r_{\text{cal}}=0.934$, $\text{RMSE}_{\text{cal}}=0.38$; $r_{\text{L5\%O}}=0.906$, $q_{\text{L5\%O}}^2=0.820$, $\text{RMSE}_{\text{L5\%O}}=0.44$; and $r_{\text{L10\%O}}=0.910$, $q_{\text{L10\%O}}^2=0.828$, $\text{RMSE}_{\text{L10\%O}}=0.43$; $r_{\text{L20\%O}}=0.909$, $q_{\text{L20\%O}}^2=0.824$, $\text{RMSE}_{\text{L20\%O}}=0.44$; and $r_{\text{L25\%O}}=0.917$, $q_{\text{L25\%O}}^2=0.840$, $\text{RMSE}_{\text{L25\%O}}=0.42$. The neural network does not improve the prediction of $\log 1/\text{EC}_{50}$ compared with the MLR model.

The best SVM regression results for each kernel are given in Table 14. By comparing the results from MLR, ANN, and SVMR, we find that no clear

Table 14 Best SVMR Predictions for the Toxicity QSAR of Aromatic Compounds to *Chlorella vulgaris*^a

Exp	Kernel	C_{opt}	SV_{cal}	r_{cal}	RMSE_{cal}	r_{LOO}	q_{LOO}^2	RMSE_{LOO}		
1	L	39.925	65	0.927	0.40	0.920	0.841	0.42		
2	P	2	88.198	65	0.929	0.39	0.873	0.52		
6	R	0.25	88.198	65	0.990	0.15	0.766	0.70		
11	N	0.5	0.0	0.057	65	0.915	0.46	0.912	0.800	0.47
20	A	0.25	1	88.198	65	0.953	0.32	0.921	0.846	0.41
Exp	Kernel	$r_{\text{L5\%O}}$	$q_{\text{L5\%O}}^2$	$\text{RMSE}_{\text{L5\%O}}$	$r_{\text{L10\%O}}$	$q_{\text{L10\%O}}^2$	$\text{RMSE}_{\text{L10\%O}}$			
1	L	0.920	0.843	0.41	0.915	0.833	0.43			
2	P	0.832	0.641	0.63	0.835	0.635	0.63			
6	R	0.818	0.645	0.62	0.788	0.567	0.69			
11	N	0.909	0.796	0.47	0.907	0.793	0.48			
20	A	0.894	0.797	0.47	0.909	0.826	0.44			
Exp	Kernel	$r_{\text{L20\%O}}$	$q_{\text{L20\%O}}^2$	$\text{RMSE}_{\text{L20\%O}}$	$r_{\text{L25\%O}}$	$q_{\text{L25\%O}}^2$	$\text{RMSE}_{\text{L25\%O}}$			
1	L	0.916	0.837	0.42	0.917	0.837	0.42			
2	P	0.850	0.674	0.60	0.781	0.442	0.78			
6	R	0.775	0.530	0.72	0.807	0.587	0.67			
11	N	0.883	0.748	0.52	0.899	0.783	0.49			
20	A	0.921	0.840	0.42	0.873	0.745	0.53			

^a See Table 11 for the parameters of each kernel.

trend exists when the prediction difficulty increases. For each prediction test, the top three models are, in decreasing order, (1) LOO: SVM anova, SVM linear, MLR; (2) L5%O: SVM linear, MLR, SVM neural; (3) L10%O: SVM linear, ANN, MLR, and SVM anova; (4) L20%O: SVM anova, SVM linear, MLR; and (5) L25%O: MLR, ANN, SVM linear. The trends in the prediction statistics are sometimes counterintuitive. In the L20%O test, SVM anova has $r = 0.921$ and MLR has $r = 0.910$, whereas in the L25%O test, SVM anova decreases to $r = 0.873$ and MLR increases to $r = 0.919$. The prediction behavior identified here might explain the success of ensemble methods, which usually surpass the performances of individual models. Note that by using a single prediction (cross-validation) test, one might end up with a misleading ranking of QSAR models; using multiple cross-validation tests is thus advised.

SVM Regression QSAR for Bioconcentration Factors

During the bioconcentration process, chemical compounds accumulate in, for example, fish, by absorption through skin or respiratory surface. The steady-state ratio between the concentration of a compound in an aquatic organism and its concentration in the aquatic environment defines the bioconcentration factor (BCF). To determine the environmental fate of chemicals released from industrial, agricultural, or residential sources, it is essential to determine or predict their BCF. Because the experimental determination of BCF is time consuming and expensive, various QSAR models have been developed for the BCF prediction using structural descriptors.¹³⁸⁻¹⁴³ Gramatica and Papa proposed a BCF QSAR model for 238 diverse chemicals, based on five theoretical descriptors, namely: $V_{D,deg}^M$, mean information content of the distance degree magnitude; MATS2m, Moran autocorrelation of a topological structure; GATS2e, Geary autocorrelation; H6p, H autocorrelation; and nHAcc, number of H-bond acceptors.¹⁴⁴ We used this dataset of 238 compounds and those five descriptors to compare MLR and SVMR models.¹⁴⁵ The cross-validation statistics of the MLR model are close to the calibration statistics, indicating that the model is stable and gives good predictions:

$$\begin{aligned} \text{BCF} = & -18.291 (\pm 4.727) + 1.867 (\pm 0.483) V_{D,deg}^M \\ & + 15.813 (\pm 4.087) \text{MATS2m} - 0.356 (\pm 0.092) \text{GATS2e} \\ & - 2.204 (\pm 0.570) \text{H6p} - 0.488 (\pm 0.126) \text{nHAcc} \end{aligned} \quad [109]$$

$$n = 238 \quad r_{\text{cal}} = 0.910 \quad \text{RMSE}_{\text{cal}} = 0.57 \quad s_{\text{cal}} = 0.57 \quad F_{\text{cal}} = 223.04$$

$$\begin{array}{lll} r_{\text{LOO}} = 0.904 & q_{\text{LOO}}^2 = 0.818 & \text{RMSE}_{\text{LOO}} = 0.58 \\ r_{\text{L5\%O}} = 0.906 & q_{\text{L5\%O}}^2 = 0.820 & \text{RMSE}_{\text{L5\%O}} = 0.58 \\ r_{\text{L10\%O}} = 0.905 & q_{\text{L10\%O}}^2 = 0.820 & \text{RMSE}_{\text{L10\%O}} = 0.58 \\ r_{\text{L20\%O}} = 0.904 & q_{\text{L20\%O}}^2 = 0.816 & \text{RMSE}_{\text{L20\%O}} = 0.59 \\ r_{\text{L25\%O}} = 0.906 & q_{\text{L25\%O}}^2 = 0.821 & \text{RMSE}_{\text{L25\%O}} = 0.58 \end{array}$$

Table 15 Best SVMR Predictions for the Bioconcentration Factors of 238 Diverse Organic Compounds^a

Exp	Kernel				RMS-			RMS-		
		C _{opt}	SV _{cal}	r _{cal}	E _{cal}	r _{L5%O}	q _{L5%O} ²	E _{L5%O}		
1	L		88.198	238	0.909	0.57	0.907	0.822	0.58	
2	P	2	73.609	238	0.921	0.54	0.891	0.782	0.64	
6	R	0.25	88.198	238	0.976	0.30	0.866	0.714	0.73	
11	N	0.5	0.0	0.026	238	0.886	0.69	0.883	0.750	0.68
20	A	0.25	1	88.198	238	0.923	0.53	0.842	0.664	0.79

Exp	Kernel	RMS-		RMS-		RMS-				
		r _{L10%O}	q _{L10%O} ²	r _{L20%O}	q _{L20%O} ²	r _{L25%O}	q _{L25%O} ²	E _{L25%O}		
1	L	0.907	0.822	0.58	0.907	0.821	0.58	0.906	0.819	0.58
2	P	0.896	0.791	0.62	0.887	0.775	0.65	0.881	0.758	0.67
6	R	0.868	0.718	0.73	0.856	0.674	0.78	0.860	0.704	0.74
11	N	0.886	0.750	0.68	0.891	0.763	0.67	0.876	0.751	0.68
20	A	0.857	0.700	0.75	0.851	0.682	0.77	0.827	0.626	0.84

^a See Table 11 for the parameters of each kernel.

Table 15 contains the best SVM regression results for each kernel. The cross-validation results show that the correlation coefficient decreases in the following order of kernels: linear > degree 2 polynomial > neural > RBF > anova. The MLR and SVMR linear models are very similar, and both are significantly better than the SVM models obtained with nonlinear kernels. The inability of nonlinear models to outperform the linear ones can be attributed to the large experimental errors in determining BCF.

SVM regression is a relatively novel addition to the field of QSAR, but its potential has not yet been sufficiently explored. In this pedagogically driven chapter, we have presented four QSAR applications in which we compared the performances of five kernels with models obtained with MLR and ANN. In general, the SVM regression cannot surpass the predictive ability of either MLR or ANN, and the prediction of nonlinear kernels is lower than that obtained with the linear kernel. Several levels of cross-validation are necessary to confirm the prediction stability; in particular, the QSAR for *Chlorella vulgaris* toxicity shows a different ranking for these methods, depending on the cross-validation test. The statistics of the QSAR models are dependent on the kernel type and parameters, and SVM regression gives in some cases unexpectedly low prediction statistics. Another problem with nonlinear kernels is overfitting, which was found in all four QSAR experiments. For this tutorial we also experimented with ANNs having different output transfer functions (linear, symlog, and tanh; data not shown). When the number of hidden neurons was kept low, all ANN results were consistently good, unlike SVM regression, which shows a wide and unpredictable variation with the kernel type and parameters.

The fact that the linear kernel gives better results than nonlinear kernels for certain QSAR problems is documented in the literature. Yang et al. compared linear, polynomial, and RBF kernels for the following properties of alkyl benzenes: boiling point, enthalpy of vaporization at the boiling point, critical temperature, critical pressure, and critical volume.¹⁴⁶ A LOO test showed that the first four properties were predicted best with a linear kernel, whereas critical volume was predicted best with a polynomial kernel.

REVIEW OF SVM APPLICATIONS IN CHEMISTRY

A rich literature exists on the topic of chemical applications of support vector machines. These publications are usually for classification, but some interesting results have also been obtained with SVM regression. An important issue is the evaluation of the SVM capabilities. Accordingly, many papers contain comparisons with other pattern recognition algorithms. Equally important is the assessment of various parameters and kernels that can give rise to the best SVM model for a particular problem. In this section, we present a selection of published SVM applications in chemistry that focus on drug design and classification of chemical compounds, SAR and QSAR, genotoxicity of chemical compounds, chemometrics, sensors, chemical engineering, and text mining for scientific information.

Recognition of Chemical Classes and Drug Design

A test in which kinase inhibitors were discriminated from noninhibitors was used by Briem and Günther to compare the prediction performances of several machine learning methods.⁸⁵ The learning set consisted of 565 kinase inhibitors and 7194 inactive compounds, and the validation was performed with a test set consisting of 204 kinase inhibitors and 300 inactive compounds. The structure of the chemical compounds was encoded into a numerical form by using Ghose–Crippen atom type counts. Four classification methods were used: SVM with a Gaussian RBF kernel, artificial neural networks, *k*-nearest neighbors with genetic algorithm parameter optimization, and recursive partitioning (RP). All four methods were able to classify kinase inhibitors from noninhibitors, but with slight differences in the predictive power of the models. The average test accuracy for 13 experiments indicates that SVMs give the best predictions for the test set: SVM 0.88, *k*-NN 0.84, ANN 0.80, RP 0.79. The results for a majority vote of a jury of 13 experiments show that SVM and ANN had identical test accuracy: SVM 0.88, *k*-NN 0.87, ANN 0.88, and RP 0.83.

Müller et al. investigated several machine learning algorithms for their ability to identify drug-like compounds based on a set of atom type counts.¹⁴⁷ Five machine learning procedures were investigated: SVM with polynomial

and Gaussian RBF kernels, linear programming machines, linear discriminant analysis, bagged k -nearest neighbors, and bagged decision trees C4.5. Drug-like compounds were selected from the World Drug Index (WDI), whereas non-drug compounds were selected from the Available Chemicals Directory (ACD), giving a total of 207,001 compounds. The chemical structure was represented with the counts of Ghose–Crippen atom types. The test error for discriminating drug-like from non-drug compounds shows that the two SVM models give the best results: SVM RBF 6.8% error, SVM linear 7.0% error, C4.5 8.2% error, and k -NN 8.2% error.

Jorissen and Gilson applied SVM to in silico screening of molecular databases for compounds possessing a desired activity.¹⁴⁸ Structural descriptors were computed with Dragon, and the parameters of the SVM (with a Gaussian RBF kernel) were optimized through a cross-validation procedure. Five sets of 50 diverse inhibitors were collected from the literature. The active compounds are antagonists of the α_{1A} adrenoceptor and reversible inhibitors of cyclin-dependent kinase, cyclooxygenase-2, factor Xa, and phosphodiesterase-5. The nonactive group of compounds was selected from the National Cancer Institute diversity set of chemical compounds. Based on the enrichment factors computed for the five sets of active compounds, it was found that SVM can successfully identify active compounds and discriminate them from nonactive chemicals.

Yap and Chen developed a jury SVM method for the classification of inhibitors and substrates of cytochromes P450 3A4 (CYP3A4, 241 inhibitors and 368 substrates), 2D6 (CYP2D6, 180 inhibitors and 198 substrates), and 2C9 (CYP2C9, 167 inhibitors and 144 substrates).⁸⁶ Structural descriptors computed with Dragon were selected with a genetic algorithm procedure and a L10%O or L20%O SVM cross-validation. Two jury SVM algorithms were applied. The first is the positive majority consensus SVM (PM-CSVM), and the second is the positive probability consensus SVM (PP-CSVM). PM-CSVM classifies a compound based on the vote of the majority of its SVM models, whereas PP-CSVM explicitly computes the probability for a compound being in a certain class. Several tests performed by Yap and Chen showed that at least 81 SVM models are necessary in each ensemble. Both PM-CSVM and PP-CSVM were shown to be superior to a single SVM model (Matthews correlation coefficient for CYP2D6, MCC = 0.742 for single SVM, MCC = 0.802 for PM-CSVM, and MCC = 0.821 for PP-CSVM). Because PP-CSVM appears to outperform PM-CSVM, the final classification results were generated with PP-CSVM: MCC = 0.899 for CYP3A4, MCC = 0.884 for CYP2D6, and MCC = 0.872 for CYP2C9.

Arimoto, Prasad, and Gifford compared five machine learning methods (recursive partitioning, naïve Bayesian classifier, logistic regression, k -nearest neighbors, and SVM) for their ability to discriminate between inhibitors ($IC_{50} < 3 \mu\text{M}$) and noninhibitors ($IC_{50} > 3 \mu\text{M}$) of cytochrome P450 3A4.¹⁴⁹ The dataset of 4470 compounds was characterized with four sets of

descriptors: MAKEBITS BCI fingerprints (4096 descriptors), MACCS fingerprints (166 descriptors), MOE TGT (typed graph triangle) fingerprints (13608 descriptors), and MolconnZ topological indices (156 descriptors). The models were evaluated with L10%O cross-validation and with a test set of 470 compounds (179 inhibitors and 291 noninhibitors). The most predictive models are the BCI fingerprints/SVM, which correctly classified 135 inhibitors and 249 noninhibitors; the MACCS fingerprints/SVM, which correctly classified 137 inhibitors and 248 noninhibitors; and topological indices/recursive partitioning, which correctly classified 147 inhibitors and 236 noninhibitors. A consensus of these three models slightly increased the accuracy to 83% compared to individual classification models.

Svetnik et al. performed a large-scale evaluation of the stochastic gradient boosting method (SGB), which implements a jury of classification and regression trees.¹⁵⁰ SGB was compared with a single decision tree, a random forest, partial least squares, *k*-nearest neighbors, naïve Bayes, and SVM with linear and RBF kernels. For the 10 QSAR datasets that were used for these tests we indicate here the best two methods for each QSAR, as determined by the prediction statistics (mean for 10 cross-validation experiments): blood-brain barrier (180 active compounds, 145 non-active compounds), random forest AC = 0.806 and SGB AC = 0.789; estrogen receptor binding activity (131 binding compounds, 101 non-binding compounds) random forest AC = 0.827 and SGB AC = 0.824; P-glycoprotein (P-gp) transport activity (108 P-gp substrates, 78 P-gp non-substrates) random forest AC = 0.804 and PLS AC = 0.798; multidrug resistance reversal agents (298 active compounds, 230 non-active compounds) random forest AC = 0.831 and SGB AC = 0.826; cyclin-dependent kinase 2 antagonists (361 active compounds, 10579 inactive compounds) random forest AC = 0.747 and SVM RBF AC = 0.723; binding affinity for the dopamine D₂ receptor (116 disubstituted piperidines) random forest $q^2 = 0.477$ and PLS $q^2 = 0.454$; log D (11260 compounds) SGB $q^2 = 0.860$ and SVM linear $q^2 = 0.841$; binding to unspecified channel protein (23102 compounds) SVM linear $q^2 = 0.843$ and SVM RBF $q^2 = 0.525$; cyclooxygenase-2 inhibitors (314 compounds for regression; and 153 active compounds and 161 non-active compounds for classification), regression random forest $q^2 = 0.434$ and SGB $q^2 = 0.390$, and classification SGB AC = 0.789 and SVM linear AC = 0.774. The study shows that jury methods are generally superior to single models.

An important adverse drug reaction is the *torsade de pointes* (TdP) induction. TdP accounts for almost one third of all drug failures during drug development and has resulted in several drugs being withdrawn from the market. Yap et al. developed an SVM classification model to predict the TdP potential of drug candidates.¹⁵¹ The drugs that induce TdP were collected from the literature (204 for training and 39 for prediction), whereas drugs with no reported cases of TdP in humans were selected as the non-active compounds (204 for training and 39 for prediction). The molecular structure for

each molecule was characterized with the linear solvation energy relationship, (LSER) descriptors. The prediction accuracy for each method is 91.0% accuracy for SVM with Gaussian RBF kernel, 88.5% accuracy for k -NN, 78.2% accuracy for probabilistic neural networks, and 65.4% accuracy for the C4.5 decision tree, thus illustrating the good results of support vector machines classification.

HERG (human ether-a-go-go) potassium channel inhibitors can lead to a prolongation of the QT interval that can trigger TdP, an atypical ventricular tachycardia. Tobita, Nishikawa, and Nagashima developed an SVM classifier that can discriminate between high and low HERG potassium channel inhibitors.¹⁵² The IC_{50} values for 73 drugs were collected from the literature, and two thresholds were used by those authors to separate high and low inhibitors, namely $pIC_{50} = 4.4$ (58 active and 15 non-active compounds) and $pIC_{50} = 6.0$ (28 active and 45 non-active compounds). The chemical structure of each molecule was represented by 57 2D MOE descriptors and 51 molecular fragments representing a subset of the public 166-bit MACCS key set. The classification accuracy for L10%O cross-validation was 95% for $pIC_{50} = 4.4$ and 90% for $pIC_{50} = 6.0$, again showing the utility of SVM for classification.

Xue et al. investigated the application of recursive feature elimination for the three following classification tests: P-glycoprotein substrates (116 substrates and 85 non-substrates), human intestinal absorption (131 absorbable compounds and 65 non-absorbable compounds), and compounds that cause *torsade de pointes* (85 TdP inducing compounds and 276 non-TdP inducing compounds).⁶⁹ With the exception of TdP compounds, the recursive feature elimination increases significantly the prediction power of SVM classifiers with a Gaussian RBF kernel. The accuracy (AC) and Matthews correlation coefficient (MCC) for SVM alone and for SVM plus recursive feature elimination (SVM + RFE) using a L20%O cross-validation test demonstrates the importance of eliminating ineffective descriptors: P-glycoprotein substrates, SVM AC = 68.3% and MCC = 0.37, SVM + RFE AC = 79.4% and MCC = 0.59; human intestinal absorption, SVM AC = 77.0% and MCC = 0.48, SVM + RFE AC = 86.7% and MCC = 0.70; *torsade de pointes* inducing compounds, SVM AC = 82.0% and MCC = 0.48, and SVM + RFE AC = 83.9% and MCC = 0.56.

Selecting an optimum group of descriptors is both an important and time-consuming phase in developing a predictive QSAR model. Fröhlich, Wegner, and Zell introduced the incremental regularized risk minimization procedure for SVM classification and regression models, and they compared it with recursive feature elimination and with the mutual information procedure.⁷⁰ Their first experiment considered 164 compounds that had been tested for their human intestinal absorption, whereas the second experiment modeled the aqueous solubility prediction for 1297 compounds. Structural descriptors were computed by those authors with JOELib and MOE, and full cross-validation was performed to compare the descriptor selection methods. The incremental

regularized risk minimization procedure gave slightly better results than did the recursive feature elimination.

Sorich et al. proposed *in silico* models to predict chemical glucuronidation based on three global descriptors (equalized electronegativity, molecular hardness, and molecular softness) and three local descriptors (atomic charge, Fukui function, and atomic softness), all based on the electronegativity equalization method (EEM).¹⁵³ The metabolism of chemical compounds by 12 human UDP-glucuronosyltransferase (UGT) isoforms was modeled with a combined approach referred to as cluster-GA-PLSDA (cluster analysis-genetic algorithm-partial least-squares discriminant analysis) and with *v*-SVM with an RBF kernel. Groups containing between 50 and 250 substrates and nonsubstrates for each of the 12 UGT isoforms were used to generate the classification models. The average percentage of correctly predicted chemicals for all isoforms is 78% for SVM and 73% for cluster-GA-PLSDA. By combining the EEM descriptors with 2-D descriptors, the SVM average percentage of correctly predicted chemicals increases to 84%.

Jury methods can increase the prediction performances of the individual models that are aggregated in the ensemble. Merkwirth et al. investigated the use of *k*-NN, SVM, and ridge regression for drug-like compound identification.⁸⁴ The first test of their jury approach involved 902 compounds from high-throughput screening experiments that were classified as “frequent hitters” (479 compounds) and “non-frequent hitters” (423 compounds), each of which was characterized by 1814 structural descriptors. The second test consisted of inhibitors of the cytochrome P450 3A4 (CYP3A4), which were divided into a group of low inhibitors (186 compounds with $IC_{50} < 1 \mu M$) and another group of high inhibitors (224 compounds with $IC_{50} > 1 \mu M$). Their cross-validation statistics show that SVM models (single and in a jury of 15 models) are the best classifiers, as can be seen from the values of the Matthews correlation coefficient: for frequent hitters, SVM 0.91 and jury-SVM 0.92; for CYP3A4, SVM and jury-SVM 0.88. Both SVM and jury-SVM classifiers were obtained by using all structural descriptors, which gave better results than models obtained when using only selected input descriptors. Overall, this approach to jury prediction does not provide any significant advantage over single SVM classifiers.

Five methods of feature selection (information gain, mutual information, χ^2 -test, odds ratio, and GSS coefficient) were compared by Liu for their ability to discriminate between thrombin inhibitors and noninhibitors.⁷¹ The chemical compounds were provided by DuPont Pharmaceutical Research Laboratories as a learning set of 1909 compounds contained 42 inhibitors and 1867 noninhibitors, and a test set of 634 compounds contained 150 inhibitors and 484 noninhibitors. Each compound was characterized by 139,351 binary features describing their 3-D structure. In this comparison of naïve Bayesian and SVM classifiers, all compounds were considered together, and a L10%O cross-validation procedure was applied. Based on information gain descriptor selection,

SVM was robust to a 99% reduction of the descriptor space, with a small decrease in sensitivity (from 58.7% to 52.5%) and specificity (from 98.4% to 97.2%).

Byvatov and Schneider compared the SVM-based and the Kolmogorov–Smirnov feature selection methods to characterize ligand–receptor interactions in focused compound libraries.⁷² Three datasets were used to compare the feature selection algorithms: 226 kinase inhibitors and 4479 noninhibitors; 227 factor Xa inhibitors and 4478 noninhibitors; and 227 factor Xa inhibitors and 195 thrombin inhibitors. SVM classifiers with a degree 5 polynomial kernel were used for all computations, and the molecular structure was encoded into 182 MOE descriptors and 225 topological pharmacophores. In one test, both feature selection algorithms produced comparable results, whereas in all other cases, SVM-based feature selection had better predictions.

Finally, we highlight the work of Zernov et al. who tested the SVM ability to discriminate between active–inactive compounds from three libraries.¹⁵⁴ The first test evaluated the discrimination between drug-like and non-drug compounds. The learning set contained 15,000 compounds (7465 drugs and 7535 non-drugs), and the test set had 7500 compounds (3751 drugs and 3749 non-drugs). The test set accuracy for SVM with an RBF kernel (75.15%) was slightly lower in percentage prediction than that of ANN (75.52%). The second experiment evaluated the discrimination between agrochemical and non-agrochemical compounds, and the third evaluated the discrimination between low and high carbonic anhydrase II inhibitors. In both of these tests, SVM classifiers had the lowest number of errors.

QSAR

SVM classification and regression were used to model the potency of diverse drug-like compounds to inhibit the human cytochromes P450 3A4 (CYP3A4) and 2D6 (CYP2D6).¹⁵⁵ The dataset consisted of 1345 CYP3A4 and 1187 CYP2D6 compounds tested for the 50% inhibition (IC_{50}) of the corresponding enzyme. The SVM models were trained with the Gaussian RBF kernel, and the one-versus-one technique was used for multiclass classification. For SVM classification, the datasets were partitioned into three groups: strong inhibitors, consisting of compounds with $IC_{50} < 2 \mu\text{M}$ (243 CYP3A4 inhibitors and 182 CYP2D6 inhibitors); medium inhibitors, consisting of those compounds with IC_{50} between 2 and 20 μM (559 CYP3A4 inhibitors and 397 CYP2D6 inhibitors); and weak inhibitors, consisting of compounds with $IC_{50} > 20 \mu\text{M}$ (543 CYP3A4 inhibitors and 608 CYP2D6 inhibitors). Four sets of structural descriptors were used to train the SVM models: in-house 2-D descriptors, such as atom and ring counts; MOE 2-D descriptors, such as topological indices and pharmacophores; VolSurf descriptors, based on molecular interaction fields; and a set of 68 AM1 quantum indices. Leave-10%–out was used to cross-validate the SVM models. The best SVM

classification predictions were obtained with the MOE 2-D descriptors. For CYP3A4, the test set accuracy is 72% (76% for strong inhibitors, 67% for medium inhibitors, and 77% for weak inhibitors), whereas for CYP2D6, the test set accuracy is 69% (84% for strong inhibitors, 53% for medium inhibitors, and 74% for weak inhibitors). The same group of descriptors gave the best SVM regression predictions: CYP3A4, $q^2 = 0.51$ vs. $q^2 = 0.39$ for PLS, and CYP2D6, $q^2 = 0.52$ vs. $q^2 = 0.30$ for PLS. In these QSAR models, SVM regression gave much better predictions than did PLS.

Aires-de-Sousa and Gasteiger used four regression techniques [multiple linear regression, perceptron (a MLF ANN with no hidden layer), MLF ANN, and v-SVM regression] to obtain a quantitative structure-enantioselectivity relationship (QSER).¹⁵⁶ The QSER models the enantiomeric excess in the addition of diethyl zinc to benzaldehyde in the presence of a racemic catalyst and an enantiopure chiral additive. A total of 65 reactions constituted the dataset. Using 11 chiral codes as model input and a three-fold cross-validation procedure, a neural network with two hidden neurons gave the best predictions: ANN 2 hidden neurons, $R^2_{\text{pred}} = 0.923$; ANN 1 hidden neurons, $R^2_{\text{pred}} = 0.906$; perceptron, $R^2_{\text{pred}} = 0.845$; MLR, $R^2_{\text{pred}} = 0.776$; and v-SVM regression with RBF kernel, $R^2_{\text{pred}} = 0.748$.

A molecular similarity kernel, the Tanimoto similarity kernel, was used by Lind and Maltseva in SVM regression to predict the aqueous solubility of three sets of organic compounds.⁹⁹ The Tanimoto similarity kernel was computed from molecular fingerprints. The RMSE and q^2 cross-validation statistics for the three sets show a good performance of SVMR with the Tanimoto kernel: set 1 (883 compounds), RMSE = 0.62 and $q^2 = 0.88$; set 2 (412 compounds), RMSE = 0.77 and $q^2 = 0.86$; and set 3 (411 compounds), RMSE = 0.57 and $q^2 = 0.88$. An SVMR model was trained on set 1 and then tested on set 2 with good results, i.e., RMSE = 0.68 and $q^2 = 0.89$.

Yang et al. developed quantitative structure-property relationships (QSPR) for several properties of 47 alkyl benzenes. These properties included boiling point, enthalpy of vaporization at the boiling point, critical temperature, critical pressure, and critical volume.¹⁴⁶ The molecular structure of alkyl benzenes was encoded with Randić–Kier–Hall connectivity indices, electrotopological state indices, and Kappa indices. For each property, the optimum set of topological indices, kernel (linear, polynomial, or Gaussian RBF), C , and ϵ were determined with successive LOO cross-validations. The LOO RMSE statistics for SVM regression, PLS, and ANN (three hidden neurons) show that the SVM model is the best: boiling point, SVMR 2.108, PLS 2.436, ANN 5.063; enthalpy of vaporization at the boiling point, SVMR 0.758, PLS 0.817, ANN 1.046; critical temperature, SVMR 5.523, PLS 7.163, ANN 9.704; critical pressure, SVMR 0.075, PLS 0.075, ANN 0.114; and critical volume, SVMR 4.692, PLS 5.914, ANN 9.452. The first four properties were best predicted with a linear kernel, whereas a polynomial kernel was used to model the critical volume.

Kumar et al. introduced a new method for descriptor selection, the locally linear embedding, which can be used for reducing the nonlinear dimensions in QSPR and QSAR.⁶⁸ SVM regression (Gaussian RBF kernel) was used to test the new descriptor selection algorithm, using three datasets: boiling points of 150 alkanes characterized by 12 topological indices; the Selwood dataset with 31 chemical compounds characterized by 53 descriptors; and the Steroids dataset consisting of 31 steroids with 1248 descriptors (autocorrelation of molecular surface indices). The statistics obtained with locally linear embedding were better than those obtained with all descriptors or by PCA descriptor reduction.

Genotoxicity of Chemical Compounds

During the process of drug discovery, the genotoxicity of drug candidates must be monitored closely. Genotoxicity mechanisms include DNA methylation, DNA intercalation, unscheduled DNA synthesis, DNA adduct formation, and strand breaking. Li et al. compared the ability of several machine learning algorithms to classify a set of 860 compounds that were tested for genotoxicity (229 genotoxic and 631 non-genotoxic).¹⁵⁷ Four methods were compared: SVM, probabilistic neural networks, k -nearest neighbors, and the C4.5 decision tree. An initial set of 199 structural descriptors (143 topological indices, 31 quantum indices, and 25 geometrical descriptors) was reduced to 39 descriptors using an SVM descriptor selection procedure. A L20%O cross-validation test showed that SVM has the highest prediction accuracy: 89.4% SVM with RBF kernel, 82.9% k -NN, 78.9% probabilistic neural networks, and 70.7% C4.5.

Typically, an SVM application that predicts properties from the molecular structure uses structural descriptors as input to the SVM model. These descriptors are used in nonlinear functions, such as the polynomial or RBF kernels, to compute the SVM solution. Mahé et al. defined a series of graph kernels that can predict various properties from only the molecular graph.⁹⁷ Atoms (graph vertices) are characterized by their chemical nature or by their connectivity through the Morgan index. Their first test of the molecular graph kernels considered the classification of 230 aromatic and hetero-aromatic nitro compounds that were tested for mutagenicity on *Salmonella typhimurium*. This dataset was further divided into a regression-easy set of 188 compounds (125 mutagens and 63 nonmutagens) and a regression-difficult set of 42 compounds. In a comparative test of leave-10%-out cross-validation accuracy, the molecular graph kernel ranked third: feature construction 95.7%, stochastic matching 93.3%, graph kernel 91.2%, neural network 89.4%, linear regression 89.3%, and decision tree 88.3%. For the group of 42 compounds, the literature has fewer comparative tests. In a LOO test, the accuracy of the new kernel was higher than that of other methods: graph kernel 88.1%, inductive logic programming 85.7%, and decision tree 83.3%. Their second test of the

graph kernel used a dataset of 684 non-congeneric compounds classified as mutagens (341 compounds) or nonmutagens (343 compounds) in a *Salmonella* microsome assay. Previous models for this dataset, based on molecular fragments, have a L10%O accuracy of 78.5%, whereas the graph kernel has an accuracy between 76% and 79%.

The mutagenicity dataset of 230 aromatic and hetero-aromatic nitro compounds was also used as a test case for a molecular graph kernel by Jain, Geibel, and Wysotzki.⁹⁸ Their kernel is based on the Schur–Hadamard inner product for a pair of molecular graphs. The leave–10%–out cross-validation accuracy is 92% for the set of 188 compounds and 90% for the set of 42 compounds. The problem of computing the Schur–Hadamard inner product for a pair of graphs is NP complete, and in this paper, it was approximated with a recurrent neural network. However, these approximations are not, in general, a kernel. Moreover, for some values of the parameters that control the kernel, the calculation of an SVM solution was not possible.

Helma et al. used the MOLFEA program for generating molecular substructures to discriminate between mutagenic and nonmutagenic compounds.¹⁵⁸ A group of 684 compounds (341 mutagenic and 343 nonmutagenic) evaluated with the Ames test (*Salmonella*/microsome assay) was used to compare the C4.5 decision tree algorithm, the PART rule learning algorithm, and SVM with linear and degree 2 polynomial kernels. The L10%O accuracy of 78.5% for the SVM classifier is higher than that of the C4.5 (75.0% accuracy) and PART (74.7% accuracy) algorithms.

Chemometrics

Several forms of transmissible spongiform encephalopathies are known today, such as scrapie, fatal familial insomnia, kuru, chronic wasting disease, feline spongiform encephalopathy, Creutzfeldt–Jacob disease in humans, or bovine spongiform encephalopathy (BSE). The main pathological characteristic of these diseases is a sponge-like modification of brain tissue. Martin et al. developed a serum-based diagnostic pattern recognition method for BSE diagnosis.¹⁵⁹ Mid-infrared spectroscopy of 641 serum samples was performed, and four classification algorithms (linear discriminant analysis, LDA; robust linear discriminant analysis, RLDA; ANN; SVM) were used to characterize the samples as BSE-positive or BSE-negative. The four classifiers were tested for a supplementary set of 160 samples (84 BSE-positive and 76 BSE-negative). For the test set, ANN had the highest sensitivity (ANN 93%, SVM 88%, LDA 82%, RLDA 80%), whereas SVM had the highest specificity (SVM 99%, LDA 93%, ANN 93%, RLDA 88%).

After the emergence of the mad cow disease, the European Union regulatory agencies banned processed animal proteins (meat and bone meal, MBM) in feedstuffs destined to farm animals that are kept, fattened, or bred for the production of food. A Near IR–SVM (NIR–SVM) system based

on plane array near-infrared imaging spectroscopy was proposed by Pierna et al. to detect MBM.¹⁶⁰ Learning was based on NIR spectra from 26 pure animal meals and 59 pure vegetal meals, with a total of 5521 spectra collected (2233 animal and 3288 vegetal). An LOO comparative evaluation of PLS, ANN, and SVM shows that support vector machines have the lowest RMSE: SVM RBF kernel 0.102, ANN 0.139, and PLS 0.397.

Wet granulation and direct compression are two methods used to manufacture tablets in the pharmaceutical industry. Zomer et al. used pyrolysis-gas chromatography-mass-spectrometry coupled with SVM classification to discriminate between the two tablet production methods.¹⁶¹ Mass spectra data were submitted to a PCA analysis, and the first principal components were used as input for SVM models having linear, polynomial, and Gaussian RBF kernels. SVM classifiers with polynomial and RBF kernels performed better in prediction than discriminant analysis.

The pathological condition induced by exposing an organism to a toxic substance depends on the mode of admission, the quantity, and the type of dosage (acute or chronic). Urine profiling by β -cyclodextrin-modified micellar electrokinetic capillary electrophoresis was used by Zomer et al. to identify the type of cadmium intoxication (acute or chronic).¹⁶² Their dataset of 96 samples was split into a learning set of 60 samples and a test set of 36 samples. Discriminant analysis applied to the first six principal components had better results on the test set (96.97% correctly classified) than did SVM trained on the original measured data (75.76% correctly classified).

NIR spectroscopy is often used for nondestructive measurement of chemicals in various materials. The application of least-squares SVM regression (LS-SVMR) in predicting mixture composition from NIR spectra was investigated by Thissen et al.¹⁶³ NIR spectra for ternary mixtures of ethanol, water, and 2-propanol were measured at 30°C, 40°C, 50°C, 60°C, and 70°C. The learning set consisted of 13 mixtures per temperature, whereas the test set consisted of 6 mixtures per temperature. For the test set, the least-squares SVM approach had an RMSE 2.6 times lower than that from a PLS analysis.

Chauchard et al. investigated the ability of least-squares SVM regression to predict the acidity of different grape varieties from NIR spectra.¹⁶⁴ NIR scans between 680 and 1100 nm for 371 grape samples were collected for three varieties: carignan (188 samples), mourverdre (84 samples), and ugni-blanc (99 samples). The total acidity (malic and tartaric acid concentrations) was measured with an HPLC assay. The PLS model selected 68 wavelengths from the NIR spectra, and with eight principal factors gave a prediction $q^2 = 0.76$ and a test correlation coefficient $R^2 = 0.77$. Using 10 principal factors, LS-SVMR models were more predictive than was PLS, with $q^2 = 0.83$ and $R^2 = 0.86$. A comparison between an MLR with eight wavelengths ($q^2 = 0.69$ and $R^2 = 0.68$) and an LS-SVMR obtained for the same wavelengths ($q^2 = 0.77$ and $R^2 = 0.78$) showed a significant improvement for the support vector machines model.

PLS and SVM regression were compared in their ability to predict, from Raman spectra, the monomer masses for the copolymerization of methyl methacrylate and butyl acrylate in toluene.¹⁶⁵ The high- and low-resolution Raman spectra of 37 training samples were used to compute the regression models, which were subsequently tested for 41 test samples. For the high-resolution spectra, the mean relative errors were 3.9% for SVMR and 10.1% for PLS. For the low-resolution spectra, these errors were 22.8% for SVMR and 68.0% for PLS. In general, SVMR with a degree 1 polynomial kernel gave the best predictions, which shows that a linear SVM model predicts better than the linear PLS model for this type of analysis.

Active learning support vector machines (AL-SVMs) was used by Zomer et al. to identify beach sand polluted with either gasoline or crude oil.¹⁶⁶ A total of 220 samples were split into 106 learning samples and 114 test samples. Each sample was analyzed using HS-MS (head-space sampler coupled to a mass-spectrometer) and with the mass spectra recorded in the range m/z 49–160. The results obtained by Zomer et al. show that the active learning procedure is effective in selecting a small subset of training samples, thus greatly reducing the number of experiments necessary to obtain a predictive model.

Chemometrics techniques are usually applied in capillary electrophoresis to obtain an optimum resolution of the peaks, lower detection limits, shorter migration times, good peak shapes, higher precision, and better signal-to-noise ratio. Optimum separation conditions in capillary electrophoresis were determined by Zhai et al. by combining a genetic algorithm with least-squares support vector machines.¹⁶⁷ The optimization target of the genetic algorithm was to increase the peak resolution, symmetry, and height, and to decrease the migration time. The study involved the identification of four compounds with anti-tumor activity. The optimizable parameters are the voltage and electrophoresis buffer composition, whereas the output measured parameters were the migration time, height, and width for each of the four peaks. The correlation coefficient for LS-SVM LOO cross-validation was 0.978. By combining the simulation results of LS-SVM and a fitness function, the genetic algorithm finds an optimum combination of experimental conditions for capillary electrophoresis separation.

Sensors

Heat treatment of milk ensures the microbial safety of milk and increases its shelf life. Different heat treatments (UHT, pasteurized, sterilized) can be distinguished by analyzing the volatile compounds with an electronic nose. A hybrid system that uses an electronic nose combined with an SVM classification method was tested by Brudzewski, Osowski, and Markiewicz for milk recognition and classification.¹⁶⁸ The electronic nose was composed of seven tin oxide-based gas sensors, and the SVM model was tested with linear and RBF kernels. In the first experiment, four brands (classes) of milk were

discriminated, with each class containing 180 samples. In the second experiment, the UHT milk from one producer was classified according to the fat content, again with 180 samples for each of the four brands. For each brand, 90 samples were used for learning and 90 samples for testing the SVM classifier. The prediction was perfect for both experiments and all brands of milk.

Measurements collected from an electronic nose were used by Sadik et al. in an SVM classification system to identify several organophosphates.¹⁶⁹ The following organophosphates were tested: parathion, malathion, dichlorvos, trichlorfon, paraoxon, and diazinon. The electronic nose contained 32 conducting polymer sensors whose output signal was processed and fed into the SVM classifier for one-*versus*-one and one-*versus*-all classification. A total of 250 measurements were recorded for each of the six organophosphates, and a L20%O cross-validation procedure was implemented. Four kernels were tested, namely linear, Gaussian RBF, polynomial, and the S2000 kernel, $K(\mathbf{x}_1, \mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|^2$. In all experiments, the SVM performed better than a neural network.

An electronic nose and an SVM classifier were evaluated by Distanto, Ancona, and Siciliano for the recognition of pentanone, hexanal, water, acetone, and three mixtures of pentanone and hexanal in different concentrations.¹⁷⁰ In a LOO test, the SVM classifier with a degree 2 polynomial kernel gave the best predictions: SVM 4.5% error, RBF neural network 15% error, and multilayer feed-forward ANN 40% error.

Seven types of espresso coffee were classified by Pardo and Sberveglieri with a system composed of an electronic nose and an SVM with polynomial and Gaussian RBF kernels.¹⁷¹ For each coffee type, 36 measurements were performed with an electronic nose equipped with five thin-film semiconductor sensors based on SnO₂ and Ti-Fe. The output signal from sensors was submitted to a PCA analysis whose principal components (between 2 and 5) represented the input data for the SVM classifier. The error surface corresponding to various kernel parameters and number of input principal components was investigated.

Gasoline supplemented with alcohol or ethers has an enhanced octane number. Adding 10 vol% ethanol, for example, increases the octane number by 2.5 or more units. The most popular ether additive is methyl tertiary butyl ether (MTBE), followed by ethyl tertiary butyl ether (ETBE) and tertiary amyl methyl ether. MTBE adds 2.5–3.0 octane numbers to gasoline and is used in 25% of all U.S. gasoline. Brudzewski et al. used an electronic nose and support vector machines to identify gasoline supplemented with ethanol, MTBE, ETBE, and benzene.¹⁷² The electronic nose was composed of seven tin oxide-based gas sensors. Twelve gasoline blend types were prepared, and a total of 432 measurements were performed with the electronic nose. In a six-fold cross-validation experiment, it was found that SVM with linear, degree 2 polynomial, and Gaussian RBF kernels achieved a perfect classification.

Bicego used the similarity-based representation of electronic nose measurements for odor classification with the SVM method.¹⁷³ In the similarity-based representation, the raw data from sensors are transformed into pairwise (dis)similarities, i.e., distances between objects in the dataset. The electronic nose is an array of eight carbon black-polymer detectors. The system was tested for the recognition of 2-propanol, acetone, and ethanol, with 34 experiments for each compound. Two series of 102 experiments were performed, the first one with data recorded after 10 minutes of exposure, whereas in the second group of experiments, the data were recorded after 1 second of exposure. The one-versus-one cross-validation accuracy of the first group of experiments was 99% for similarity computed using the Euclidean metric. For the second group of experiments, the accuracy was 79% for the Euclidean metric and 80% for the Manhattan metric.

Chemical Engineering

Hybrid systems (ANN-GA and SVMR-GA) were compared by Nandi et al. for their ability to model and optimize the isopropylation of benzene on Hbeta catalyst.⁷³ The input parameters used to model the reaction were temperature, pressure, benzene-to-isopropyl alcohol ratio, and weight hourly space velocity. The output parameters were the yield of isopropylbenzene and the selectivity S , where $S = 100 \times (\text{weight of isopropylbenzene formed per unit time}) / (\text{weight of total aromatics formed per unit time})$. Based on 42 experiments, the genetic algorithm component was used to select the optimum set of input parameters that maximize both yield and selectivity. The GA-optimized solutions were then verified experimentally, showing that the two hybrid methods can be used to optimize industrial processes.

The SVM classification of vertical and horizontal two-phase flow regimes in pipes was investigated by Trafalis, Oladunni, and Papavassiliou.¹⁷⁴ The vertical flow dataset, with 424 cases, had three classes, whereas the horizontal flow dataset, with 2272 cases, had five classes. One-versus-one multiclass SVM models were developed with polynomial kernels (degrees 1 to 4). The transition region is determined with respect to pipe diameter, superficial gas velocity, and superficial liquid velocity. Compared with experimental observations, the predictions of the SVM model were, in most cases, superior to those obtained from other types of theoretical models.

The locally weighted regression was extended by Lee et al. to support vector machines and tested on the synthesis of polyvinyl butyrate (PVB).¹⁷⁵ Weighted SVM regression has a variable capacity C , which depends on a weight computed for each data point. The weighted SVM regression was computed with $\varepsilon = 0.001$. Three kernels were tested: polynomial, Gaussian RBF, and neural (tanh). A dataset of 120 patterns was divided into 80 training patterns, 20 validation patterns, and 20 test patterns. Each pattern consisted of 12 measurements of controlled variables (such as viscosity and

concentration of PVB, quantities of the first and second catalyst, reaction time, temperature) and one product property, PVB viscosity. A comparative test showed that the weighted SVM regression has the lowest error with $RMSE = 23.9$, compared with SVM regression $RMSE = 34.9$ and neural network $RMSE = 109.4$.

Chu, Qin, and Han applied an SVM classification model for the fault detection and identification of the operation mode in processes with multi-mode operations.¹⁷⁶ They studied the rapid thermal annealing, which is a critical semiconductor process used to stabilize the structure of silicon wafers and to make uniform the physical properties of the whole wafer after ion implantation. A dataset of 1848 batch data was divided into 1000 learning data and 848 test data. Input data for the SVM model were selected with an entropy-based algorithm, and 62 input parameters were used to train three SVM classification models. The system based on SVM is superior to the conventional PCA fault detection method.

The melt index of thermoplastic polymers like polypropylene (PP) and polystyrene is defined as the mass rate of extrusion flow through a specified capillary under prescribed conditions of temperature and pressure. The melt index of polypropylene and styrene-acrylonitrile (SAN) polymerization were modeled by Han, Han, and Chung with PLS, ANN, and SVM regression having a Gaussian RBF kernel.¹⁷⁷ For the SAN polymerization, 33 process variables were measured for 1024 training data and 100 testing data. The test set RMSE shows that the best predictions were obtained with the SVM regression: SVMR 0.97, ANN 1.09, and PLS 3.15. For the PP synthesis, 78 process variables were measured for 467 training data and 50 testing data. The melt index of PP is best predicted by SVMR, as shown by the corresponding RMSE values: SVMR 1.51, PLS 2.08, and ANN 3.07.

Text Mining for Scientific Information

Automatic text datamining is an important source of knowledge, with many applications in generating databases from scientific literature, such as protein–disease associations, gene expression patterns, subcellular localization, and protein–protein interactions.

The NLPProt system developed by Mika and Rost combines four support vector machines, trained individually for distinct tasks.^{178,179} The first SVM is trained to recognize protein names, whereas the second learns the environment in which a protein name appears. The third SVM is trained on both protein names and their environments. The output of these three SVMs and a score from a protein dictionary are fed into the fourth SVM, which provides as output the protein whose name was identified in the text. A dictionary of protein names was generated from SwissProt and TrEMBL, whereas the Merriam-Webster Dictionary was used as a source of common words. Other terms were added to the dictionary, such as medical terms, species names, and tissue

types. The system has a 75% accuracy, and in a test on recent abstracts from *Cell* and *EMBO Journal*, NLProt reached 70% accuracy.

An SVM approach to name recognition in text was used by Shi and Campagne to develop a protein dictionary.¹⁸⁰ A database of 80,528 full text articles from *Journal of Biological Chemistry*, *EMBO Journal*, and *Proceedings of the National Academy of Sciences* were used as input to the SVM system. A dictionary of 59,990 protein names was produced. Three support vector machines were trained to discriminate among protein names and cell names, process names, and interaction keywords, respectively. The processing time is half a second for a new full-text paper. The method can recognize name variants not found in SwissProt.

Using PubMed abstracts, the PreBIND system can identify protein-protein interactions with an SVM system.¹⁸¹ The protein-protein interactions identified by the automated PreBIND system are then combined and scrutinized manually to produce the BIND database (<http://bind.ca>). Based on a L10%O cross-validation of a dataset of 1094 abstracts, the SVM approach had a precision and recall of 92%, whereas a naïve Bayes classifier had a precision and recall of 87%.

Bio-medical terms can be recognized and annotated with SVM-based automated systems, as shown by Takeuchi and Collier.¹⁸² The training was performed with 100 Medline abstracts where bio-medical terms were marked-up manually in XML by an expert. The SVM system recognized approximately 3400 terms and showed good prediction capability for each class of terms (proteins, DNA, RNA, source, etc.).

Bunescu et al. compared the ability of several machine learning systems to extract information regarding protein names and their interactions from Medline abstracts.¹⁸³ The text recognition systems compared are dictionary based, the rule learning system Rapier, boosted wrapper induction, SVM, maximum entropy, k -nearest neighbors, and two systems for protein name identification, KEX and Abgene. Based on the F -measure (harmonic mean of precision and recall) in L10%O cross-validation, the best systems for protein name recognition are the maximum entropy with dictionary ($F = 57.86\%$) followed by SVM with dictionary ($F = 54.42\%$).

SVM RESOURCES ON THE WEB

The Internet is a vast source of information on support vector machines. The interested reader can find tutorials, reviews, theoretical and application papers, as well as a wide range of SVM software. In this section, we present several starting points for retrieving relevant SVM information from the Web.

<http://support-vector-machines.org/>. This Web portal is dedicated to support vector machines and their applications. It provides exhaustive lists of books, tutorials, publications (with special sections for applications in cheminformatics, bioinformatics, and computational biology), software for various

platforms, and links to datasets that can be used for SVM classification and regression. Very useful are the links to open-access SVM papers. The site offers also a list of SVM-related conferences.

<http://www.kernel-machines.org/>. This portal contains links to websites related to kernel methods. Included are tutorials, publications, books, software, datasets used to compare algorithms, and conference announcements. A list of major scientists in kernel methods is also available from this site.

<http://www.support-vector.net/>. This website is a companion to the book *An Introduction to Support Vector Machines* by Cristianini and Shawe-Taylor,¹⁴ and it has a useful list of SVM software.

<http://www.kernel-methods.net/>. This website is a companion to the book *Kernel Methods for Pattern Analysis* by Shawe-Taylor and Cristianini.²¹ The MatLab scripts from the book can be downloaded from this site. A tutorial on kernel methods is also available.

<http://www.learning-with-kernels.org/>. Several chapters on SVM from the book *Learning with Kernels* by Schölkopf and Smola¹⁷ are available from this site.

<http://www.boosting.org/>. This is a portal for boosting and related ensemble learning methods, such as arcing and bagging, with application to model selection and connections to mathematical programming and large margin classifiers. The site provides links to software, papers, datasets, and upcoming events.

Journal of Machine Learning Research, <http://jmlr.csail.mit.edu/>. The *Journal of Machine Learning Research* is an open-access journal that contains many papers on SVM, including new algorithms and SVM model optimization. All papers can be downloaded and printed for free. In the current context of widespread progress toward an open access to scientific publications, this journal has a remarkable story and is an undisputed success.

<http://citeseer.ist.psu.edu/burges98tutorial.html>. This is an online reprint of Burges's SVM tutorial "A Tutorial on Support Vector Machines for Pattern Recognition."²³ The citeseer repository has many useful SVM manuscripts.

PubMed, <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed>. This is a comprehensive database of abstracts for chemistry, biochemistry, biology, and medicine-related literature. PubMed is a free service of the National Library of Medicine and is a great place to start your search for SVM-related papers. PubMed has direct links for many online journals, which are particularly useful for open-access journals, such as *Bioinformatics* or *Nucleic Acids Research*. All SVM applications in cheminformatics from major journals are indexed here, but unfortunately, the relevant chemistry journals are not open access. On the other hand, PubMed is the main hub for open access to important SVM applications, such as gene arrays, proteomics, or toxicogenomics.

PubMed Central, <http://www.pubmedcentral.nih.gov/>. PubMed Central (PMC) is the U.S. National Institutes of Health (NIH) free digital archive of biomedical and life sciences journal literature. It represents the main public repository for journals that publish open-access papers. The site contains information regarding the NIH initiative for open-access publication of NIH-funded research. Numerous papers can be found on SVM applications in bioinformatics and computational biology.

SVM SOFTWARE

Fortunately, scientists interested in SVM applications in cheminformatics and computational chemistry can choose from a wide variety of free software, available for download from the Internet. The selection criteria for a useful package are problem type (classification or regression); platform (Windows, Linux/UNIX, Java, MATLAB, R); available kernels (the more the better); flexibility in adding new kernels; possibility to perform cross-validation or descriptor selection. Collected here is relevant information for the most popular SVM packages. All are free for nonprofit use, but they come with little or no support. On the other hand, they are straightforward to use, are accompanied by extensive documentation, and almost all are available as source code. For users wanting to avoid compilation-related problems, many packages are available as Windows binaries. A popular option is the use of SVM scripts in computing environments such as MATLAB, R, Scilab, Torch, YaLE, or Weka (the last five are free). For small problems, the Gist server is a viable option. The list of SVM software presented below is ordered in an approximate decreasing frequency of use.

SVM^{light}, <http://svmlight.joachims.org/>. SVM^{light}, by Joachims,¹⁸⁴ is one of the most widely used SVM classification and regression packages. It has a fast optimization algorithm, can be applied to very large datasets, and has a very efficient implementation of the leave-one-out cross-validation. It is distributed as C++ source and binaries for Linux, Windows, Cygwin, and Solaris. Kernels available include polynomial, radial basis function, and neural (tanh).

SVM^{struct}, http://svmlight.joachims.org/svm_struct.html. SVM^{struct}, by Joachims, is an SVM implementation that can model complex (multivariate) output data y , such as trees, sequences, or sets. These complex output SVM models can be applied to natural language parsing, sequence alignment in protein homology detection, and Markov models for part-of-speech tagging. Several implementations exist: SVM^{multiclass}, for multiclass classification; SVM^{cfg}, which learns a weighted context free grammar from examples; SVM^{align}, which learns to align protein sequences from training alignments; and SVM^{hmm}, which learns a Markov model from examples. These modules have straightforward applications in bioinformatics, but one can imagine

significant implementations for cheminformatics, especially when the chemical structure is represented as trees or sequences.

mySVM, <http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/index.html>. mySVM, by Rüping, is a C++ implementation of SVM classification and regression. It is available as C++ source code and Windows binaries. Kernels available include linear, polynomial, radial basis function, neural (tanh), and anova. All SVM models presented in this chapter were computed with mySVM.

JmySVM. A Java version of mySVM is part of the YaLE (Yet Another Learning Environment, <http://www-ai.cs.uni-dortmund.de/SOFTWARE/YALE/index.html>) learning environment under the name JmySVM

mySVM/db, <http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVMDB/index.html>. mySVM/db is an efficient extension of mySVM, which is designed to run directly inside a relational database using an internal JAVA engine. It was tested with an Oracle database, but with small modifications, it should also run on any database offering a JDBC interface. It is especially useful for large datasets available as relational databases.

LIBSVM, <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>. LIBSVM (Library for Support Vector Machines) was developed by Chang and Lin and contains C-classification, ν -classification, ϵ -regression, and ν -regression. Developed in C++ and Java, it also supports multiclass classification, weighted SVMs for unbalanced data, cross-validation, and automatic model selection. It has interfaces for Python, R, Splus, MATLAB, Perl, Ruby, and LabVIEW. Kernels available include linear, polynomial, radial basis function, and neural (tanh).

looms, <http://www.csie.ntu.edu.tw/~cjlin/looms/>. looms, by Lee and Lin, is a very efficient leave-one-out model selection for SVM two-class classification. Although LOO cross-validation is usually too time consuming to be performed for large datasets, looms implements numerical procedures that make LOO accessible. Given a range of parameters, looms automatically returns the parameter and model with the best LOO statistics. It is available as C source code and Windows binaries.

BSVM, <http://www.csie.ntu.edu.tw/~cjlin/bsvm/>. BSVM, authored by Hsu and Lin, provides two implementations of multiclass classification, together with SVM regression. It is available as source code for UNIX/Linux and as binaries for Windows.

OSU SVM Classifier Matlab Toolbox, http://www.ece.osu.edu/~maj/osu_svm/. This MATLAB toolbox is based on LIBSVM.

SVM Torch, <http://www.idiap.ch/learning/SVMTorch.html>. SVM Torch, by Collobert and Bengio,¹⁸⁵ is part of the Torch machine learning library (<http://www.torch.ch/>) and implements SVM classification and regression. It is distributed as C++ source code or binaries for Linux and Solaris.

Weka, <http://www.cs.waikato.ac.nz/ml/weka/>. Weka is a collection of machine learning algorithms for datamining tasks. The algorithms can either

be applied directly to a dataset or called from a Java code. It contains an SVM implementation.

SVM in R, <http://cran.r-project.org/src/contrib/Descriptions/e1071.html>. This SVM implementation in R (<http://www.r-project.org/>) contains C-classification, v -classification, ε -regression, and v -regression. Kernels available include linear, polynomial, radial basis, and neural (tanh).

M-SVM, <http://www.loria.fr/~guermeur/>. This is a multi-class SVM implementation in C by Guermeur.^{52,53}

Gist, <http://microarray.cpmc.columbia.edu/gist/>. Gist is a C implementation of support vector machine classification and kernel principal components analysis. The SVM part of Gist is available as an interactive Web server at <http://svm.sdsc.edu>. It is a very convenient server for users who want to experiment with small datasets (hundreds of patterns). Kernels available include linear, polynomial, and radial.

MATLAB SVM Toolbox, <http://www.isis.ecs.soton.ac.uk/resources/svminfo/>. This SVM toolbox, by Gunn, implements SVM classification and regression with various kernels, including linear, polynomial, Gaussian radial basis function, exponential radial basis function, neural (tanh), Fourier series, spline, and B spline. All figures from this chapter presenting SVM models for various datasets were prepared with a slightly modified version of this MATLAB toolbox.

TinySVM, <http://chasen.org/~taku/software/TinySVM/>. TinySVM is a C++ implementation of C-classification and C-regression that uses sparse vector representation. It can handle several thousand training examples and feature dimensions. TinySVM is distributed as binary/source for Linux and binary for Windows.

SmartLab, <http://www.smartlab.dibe.unige.it/>. SmartLab provides several support vector machines implementations, including cSVM, a Windows and Linux implementation of two-class classification; mcSVM, a Windows and Linux implementation of multiclass classification; rSVM, a Windows and Linux implementation of regression; and javaSVM1 and javaSVM2, which are Java applets for SVM classification.

Gini-SVM, <http://bach.ece.jhu.edu/svm/ginisvm/>. Gini-SVM, by Chakrabartty and Cauwenberghs, is a multiclass probability regression engine that generates conditional probability distributions as a solution. It is available as source code.

GPDT, <http://dm.unife.it/gpdt/>. GPDT, by Serafini, et al., is a C++ implementation for large-scale SVM classification in both scalar and distributed memory parallel environments. It is available as C++ source code and Windows binaries.

HeroSvm, <http://www.cenparmi.concordia.ca/~people/jdong/HeroSvm.html>. HeroSvm, by Dong, is developed in C++, implements SVM classification, and is distributed as a dynamic link library for Windows. Kernels available include linear, polynomial, and radial basis function.

Spider, <http://www.kyb.tuebingen.mpg.de/bs/people/spider/>. Spider is an object-orientated environment for machine learning in MATLAB. It performs unsupervised, supervised, or semi-supervised machine learning problems and includes training, testing, model selection, cross-validation, and statistical tests. Spider implements SVM multiclass classification and regression.

Java applets, <http://svm.dcs.rhbnc.ac.uk/>. These SVM classification and regression Java applets were developed by members of Royal Holloway, University of London, and the AT&T Speech and Image Processing Services Research Laboratory. SVM classification is available from <http://svm.dcs.rhbnc.ac.uk/pagesnew/GPat.shtml>. SVM regression is available at <http://svm.dcs.rhbnc.ac.uk/pagesnew/1D-Reg.shtml>.

LEARNSC, <http://www.support-vector.ws/html/downloads.html>. This site contains MATLAB scripts for the book *Learning and Soft Computing* by Kecman.¹⁶ LEARNSC implements SVM classification and regression.

Tree Kernels, <http://ai-nlp.info.uniroma2.it/moschitti/Tree-Kernel.htm>. Tree Kernels, by Moschitti, is an extension of SVM^{light}, and was obtained by encoding tree kernels. It is available as binaries for Windows, Linux, Mac-OSx, and Solaris. Tree kernels are suitable for encoding chemical structures, and thus this package brings significant capabilities for cheminformatics applications.

LS-SVMlab, <http://www.esat.kuleuven.ac.be/sista/lssvmlab/>. LS-SVMlab, by Suykens, is a MATLAB implementation of least-squares support vector machines (LS-SVMs), a reformulation of the standard SVM that leads to solving linear KKT systems. LS-SVM primal-dual formulations have been formulated for kernel PCA, kernel CCA, and kernel PLS, thereby extending the class of primal-dual kernel machines. Links between kernel versions of classic pattern recognition algorithms such as kernel Fisher discriminant analysis and extensions to unsupervised learning, recurrent networks, and control are available.

MATLAB SVM Toolbox, <http://www.igi.tugraz.at/aschwaig/software.html>. This is a MATLAB SVM classification implementation that can handle 1-norm and 2-norm SVM (linear or quadratic loss function) problems.

SVM/LOO, <http://bach.ece.jhu.edu/pub/gert/svm/incremental/>. SVM/LOO, by Cauwenberghs, has a very efficient MATLAB implementation of the leave-one-out cross-validation.

SVMsequel, <http://www.isi.edu/~hdaume/SVMsequel/>. SVMsequel, by Daume III, is an SVM multiclass classification package, distributed as C source or as binaries for Linux or Solaris. Kernels available include linear, polynomial, radial basis function, sigmoid, string, tree, and information diffusion on discrete manifolds.

LSVM, <http://www.cs.wisc.edu/dmi/lsvm/>. LSVM (Lagrangian Support Vector Machine) is a very fast SVM implementation in MATLAB by Mangasarian and Musicant. It can classify datasets containing several million patterns.

ASVM, <http://www.cs.wisc.edu/dmi/asvm/>. ASVM (Active Support Vector Machine) is a very fast linear SVM script for MATLAB, by Musicant and Mangasarian, developed for large datasets.

PSVM, <http://www.cs.wisc.edu/dmi/svm/psvm/>. PSVM (Proximal Support Vector Machine) is a MATLAB script by Fung and Mangasarian that classifies patterns by assigning them to the closest of two parallel planes.

SimpleSVM Toolbox, <http://asi.insa-rouen.fr/~gloosli/simpleSVM.html>. SimpleSVM Toolbox is a MATLAB implementation of the SimpleSVM algorithm.

SVM Toolbox, <http://asi.insa-rouen.fr/%7Earakotom/toolbox/index>. This fairly complex MATLAB toolbox contains many algorithms, including classification using linear and quadratic penalization, multiclass classification, ε -regression, v -regression, wavelet kernel, and SVM feature selection.

MATLAB SVM Toolbox, <http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox/>. Developed by Cawley, this software has standard SVM features, together with multiclass classification and leave-one-out cross-validation.

R-SVM, <http://www.biostat.harvard.edu/~xzhang/R-SVM/R-SVM.html>. R-SVM, by Zhang and Wong, is based on SVM_Torch and is designed especially for the classification of microarray gene expression data. R-SVM uses SVM for classification and for selecting a subset of relevant genes according to their relative contribution in the classification. This process is done recursively in such a way that a series of gene subsets and classification models can be obtained in a recursive manner, at different levels of gene selection. The performance of the classification can be evaluated either on an independent test dataset or by cross validation on the same dataset. R-SVM is distributed as Linux binary.

JSVM, <http://www-cad.eecs.berkeley.edu/~hwawen/research/projects/jsvm/doc/manual/index.html>. JSVM is a Java wrapper for SVM^{light}.

SvmFu, <http://five-percent-nation.mit.edu/SvmFu/>. SvmFu, by Rifkin, is a C++ package for SVM classification. Kernels available include linear, polynomial, and Gaussian radial basis function.

CONCLUSIONS

Kernel learning algorithms have received considerable attention in data modeling and prediction because kernels can straightforwardly perform a nonlinear mapping of the data into a high-dimensional feature space. As a consequence, linear models can be transformed easily into nonlinear algorithms that in turn can explore complex relationships between input data and predicted property. Kernel algorithms have applications in classification, clustering, and regression. From the diversity of kernel methods (support vector machines, Gaussian processes, kernel recursive least squares, kernel principal component analysis, kernel perceptron learning, relevance vector machines, kernel Fisher discriminants, Bayes point machines, and kernel Gram-Schmidt), only SVM was readily adopted for QSAR and cheminformatics applications.

Support vector machines represent the most important development in chemometrics after (chronologically) partial least-squares and artificial neural networks. We have presented numerous SAR and QSAR examples in this chapter that demonstrate the SVM capabilities for both classification and

regression. These examples showed that the nonlinear features of SVM should be used with caution, because this added flexibility in modeling the data brings with it the danger of overfitting. The literature results reviewed here show that support vector machines already have numerous applications in computational chemistry and cheminformatics. Future developments are expected to improve the performance of SVM regression and to explore the SVM use in jury ensembles as an effective way to increase their prediction power.

REFERENCES

1. V. Vapnik and A. Lerner, *Automat. Remote Contr.*, **24**, 774–780 (1963). Pattern Recognition Using Generalized Portrait Method.
2. V. Vapnik and A. Chervonenkis, *Theory of Pattern Recognition*, Nauka, Moscow, Russia, 1974.
3. V. Vapnik, *Estimation of Dependencies Based on Empirical Data*, Nauka, Moscow, Russia, 1979.
4. V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, 1995.
5. V. Vapnik, *Statistical Learning Theory*, Wiley-Interscience, New York, 1998.
6. C. Cortes and V. Vapnik, *Mach. Learn.*, **20**, 273–297 (1995). Support-Vector Networks.
7. B. Schölkopf, K. K. Sung, C. J. C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, *IEEE Trans. Signal Process.*, **45**, 2758–2765 (1997). Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers.
8. O. Chapelle, P. Haffner, and V. N. Vapnik, *IEEE Trans. Neural Netw.*, **10**, 1055–1064 (1999). Support Vector Machines for Histogram-based Image Classification.
9. H. Drucker, D. H. Wu, and V. N. Vapnik, *IEEE Trans. Neural Netw.*, **10**, 1048–1054 (1999). Support Vector Machines for Spam Categorization.
10. V. N. Vapnik, *IEEE Trans. Neural Netw.*, **10**, 988–999 (1999). An Overview of Statistical Learning Theory.
11. V. Vapnik and O. Chapelle, *Neural Comput.*, **12**, 2013–2036 (2000). Bounds on Error Expectation for Support Vector Machines.
12. I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, *Mach. Learn.*, **46**, 389–422 (2002). Gene Selection for Cancer Classification Using Support Vector Machines.
13. B. Schölkopf, C. J. C. Burges, and A. J. Smola, *Advances in Kernel Methods: Support Vector Learning*, MIT Press, Cambridge, Massachusetts, 1999.
14. N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, Cambridge, United Kingdom, 2000.
15. A. J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, *Advances in Large Margin Classifiers*, MIT Press, Cambridge, Massachusetts, 2000.
16. V. Kecman, *Learning and Soft Computing*, MIT Press, Cambridge, Massachusetts, 2001.
17. B. Schölkopf and A. J. Smola, *Learning with Kernels*, MIT Press, Cambridge, Massachusetts, 2002.
18. T. Joachims, *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*, Kluwer, Norwell, Massachusetts, 2002.
19. R. Herbrich, *Learning Kernel Classifiers*, MIT Press, Cambridge, Massachusetts, 2002.
20. J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*, World Scientific, Singapore, 2002.
21. J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, Cambridge, United Kingdom, 2004.
22. A. J. Smola and B. Schölkopf, *Algorithmica*, **22**, 211–231 (1998). On a Kernel-based Method for Pattern Recognition, Regression, Approximation, and Operator Inversion.

23. C. J. C. Burges, *Data Min. Knowl. Discov.*, **2**, 121–167 (1998). A Tutorial on Support Vector Machines for Pattern Recognition.
24. B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola, *IEEE Trans. Neural Netw.*, **10**, 1000–1017 (1999). Input Space Versus Feature Space in Kernel-based Methods.
25. J. A. K. Suykens, *Eur. J. Control*, **7**, 311–327 (2001). Support Vector Machines: A Nonlinear Modelling and Control Perspective.
26. K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, *IEEE Trans. Neural Netw.*, **12**, 181–201 (2001). An Introduction to Kernel-based Learning Algorithms.
27. C. Campbell, *Neurocomputing*, **48**, 63–84 (2002). Kernel Methods: A Survey of Current Techniques.
28. B. Schölkopf and A. J. Smola, in *Advanced Lectures on Machine Learning*, Vol. 2600, Springer, New York, 2002, pp. 41–64. A Short Introduction to Learning with Kernels.
29. V. D. Sanchez, *Neurocomputing*, **55**, 5–20 (2003). Advanced Support Vector Machines and Kernel Methods.
30. A. J. Smola and B. Schölkopf, *Stat. Comput.*, **14**, 199–222 (2004). A Tutorial on Support Vector Regression.
31. A. Kurup, R. Garg, D. J. Carini, and C. Hansch, *Chem. Rev.*, **101**, 2727–2750 (2001). Comparative QSAR: Angiotensin II Antagonists.
32. K. Varmuza, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 3, Wiley-VCH, Weinheim, Germany, 2003, pp. 1098–1133. Multivariate Data Analysis in Chemistry.
33. O. Ivanciuc, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 1, Wiley-VCH, Weinheim, Germany, 2003, pp. 103–138. Graph Theory in Chemistry.
34. O. Ivanciuc, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 3, Wiley-VCH, Weinheim, Germany, 2003, pp. 981–1003. Topological Indices.
35. R. Todeschini and V. Consonni, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 3, Wiley-VCH, Weinheim, Germany, 2003, pp. 1004–1033. Descriptors from Molecular Geometry.
36. P. Jurs, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 3, Wiley-VCH, Weinheim, Germany, 2003, pp. 1314–1335. Quantitative Structure-Property Relationships.
37. L. Eriksson, H. Antti, E. Holmes, E. Johansson, T. Lundstedt, J. Shockcor, and S. Wold, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 3, Wiley-VCH, Weinheim, Germany, 2003, pp. 1134–1166. Partial Least Squares (PLS) in Cheminformatics.
38. J. Zupan, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 3, Wiley-VCH, Weinheim, Germany, 2003, pp. 1167–1215. Neural Networks.
39. A. von Homeyer, in *Handbook of Chemoinformatics*, J. Gasteiger, Ed., Vol. 3, Wiley-VCH, Weinheim, Germany, 2003, pp. 1239–1280. Evolutionary Algorithms and Their Applications in Chemistry.
40. R. Fletcher, *Practical Methods of Optimization*, 2 ed., John Wiley and Sons, New York, 1987.
41. J. Platt, in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds., MIT Press, Cambridge, Massachusetts, 1999, pp. 185–208. Fast Training of Support Vector Machines Using Sequential Minimal Optimization.
42. J. Mercer, *Phil. Trans. Roy. Soc. London A*, **209**, 415–446 (1909). Functions of Positive and Negative Type and Their Connection with the Theory of Integral Equations.
43. B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, *Neural Comput.*, **12**, 1207–1245 (2000). New Support Vector Algorithms.
44. C. C. Chang and C. J. Lin, *Neural Comput.*, **13**, 2119–2147 (2001). Training v-Support Vector Classifiers: Theory and Algorithms.
45. C. C. Chang and C. J. Lin, *Neural Comput.*, **14**, 1959–1977 (2002). Training v-Support Vector Regression: Theory and Algorithms.

46. I. Steinwart, *IEEE Trans. Pattern Anal. Mach. Intell.*, **25**, 1274–1284 (2003). On the Optimal Parameter Choice for v-Support Vector Machines.
47. P. H. Chen, C. J. Lin, and B. Schölkopf, *Appl. Stoch. Models. Bus. Ind.*, **21**, 111–136 (2005). A Tutorial on v-Support Vector Machines.
48. R. Debnath, N. Takahide, and H. Takahashi, *Pattern Anal. Appl.*, **7**, 164–175 (2004). A Decision-based One-against-one Method for Multi-class Support Vector Machine.
49. C. W. Hsu and C. J. Lin, *IEEE Trans. Neural Netw.*, **13**, 415–425 (2002). A Comparison of Methods for Multiclass Support Vector Machines.
50. R. Rifkin and A. Klautau, *J. Mach. Learn. Res.*, **5**, 101–141 (2004). In Defense of One-vs-all Classification.
51. C. Angulo, X. Parra, and A. Català, *Neurocomputing*, **55**, 57–77 (2003). K-SVCR. A Support Vector Machine for Multi-class Classification.
52. Y. Guermeur, *Pattern Anal. Appl.*, **5**, 168–179 (2002). Combining Discriminant Models with New Multi-class SVMs.
53. Y. Guermeur, G. Pollastri, A. Elisseeff, D. Zelus, H. Paugam-Moisy, and P. Baldi, *Neurocomputing*, **56**, 305–327 (2004). Combining Protein Secondary Structure Prediction Models with Ensemble Methods of Optimal Complexity.
54. A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy, *Bioinformatics*, **21**, 631–643 (2005). A Comprehensive Evaluation of Multicategory Classification Methods for Microarray Gene Expression Cancer Diagnosis.
55. T. Li, C. L. Zhang, and M. Ogihara, *Bioinformatics*, **20**, 2429–2437 (2004). A Comparative Study of Feature Selection and Multiclass Classification Methods for Tissue Classification Based on Gene Expression.
56. Y. Lee and C. K. Lee, *Bioinformatics*, **19**, 1132–1139 (2003). Classification of Multiple Cancer Types by Tip Multicategory Support Vector Machines Using Gene Expression Data.
57. S. H. Peng, Q. H. Xu, X. B. Ling, X. N. Peng, W. Du, and L. B. Chen, *FEBS Lett.*, **555**, 358–362 (2003). Molecular Classification of Cancer Types from Microarray Data Using the Combination of Genetic Algorithms and Support Vector Machines.
58. S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov, T. Poggio, W. Gerald, M. Loda, E. S. Lander, and T. R. Golub, *Proc. Natl. Acad. Sci. U. S. A.*, **98**, 15149–15154 (2001). Multiclass Cancer Diagnosis Using Tumor Gene Expression Signatures.
59. O. L. Mangasarian and D. R. Musicant, *IEEE Trans. Pattern Analysis Mach. Intell.*, **22**, 950–955 (2000). Robust Linear and Support Vector Regression.
60. O. L. Mangasarian and D. R. Musicant, *Mach. Learn.*, **46**, 255–269 (2002). Large Scale Kernel Regression via Linear Programming.
61. J. B. Gao, S. R. Gunn, and C. J. Harris, *Neurocomputing*, **55**, 151–167 (2003). SVM Regression Through Variational Methods and its Sequential Implementation.
62. J. B. Gao, S. R. Gunn, and C. J. Harris, *Neurocomputing*, **50**, 391–405 (2003). Mean Field Method for the Support Vector Machine Regression.
63. W. P. Walters and B. B. Goldman, *Curr. Opin. Drug Discov. Dev.*, **8**, 329–333 (2005). Feature Selection in Quantitative Structure-Activity Relationships.
64. D. J. Livingstone and D. W. Salt, in *Reviews in Computational Chemistry*, K. B. Lipkowitz, R. Larter, and T. R. Cundari, Eds., Vol. **21**, Wiley-VCH, New York, 2005, pp. 287–348. Variable Selection - Spoil for Choice?
65. J. Bi, K. P. Bennett, M. Embrechts, C. M. Breneman, and M. Song, *J. Mach. Learn. Res.*, **3**, 1229–1243 (2003). Dimensionality Reduction via Sparse Support Vector Machines.
66. L. Cao, C. K. Seng, Q. Gu, and H. P. Lee, *Neural Comput. Appl.*, **11**, 244–249 (2003). Saliency Analysis of Support Vector Machines for Gene Selection in Tissue Classification.
67. G. M. Fung and O. L. Mangasarian, *Comput. Optim. Appl.*, **28**, 185–202 (2004). A Feature Selection Newton Method for Support Vector Machine Classification.

68. R. Kumar, A. Kulkarni, V. K. Jayaraman, and B. D. Kulkarni, *Internet Electron. J. Mol. Des.*, **3**, 118–133 (2004). Structure–Activity Relationships Using Locally Linear Embedding Assisted by Support Vector and Lazy Learning Regressors.
69. Y. Xue, Z. R. Li, C. W. Yap, L. Z. Sun, X. Chen, and Y. Z. Chen, *J. Chem. Inf. Comput. Sci.*, **44**, 1630–1638 (2004). Effect of Molecular Descriptor Feature Selection in Support Vector Machine Classification of Pharmacokinetic and Toxicological Properties of Chemical Agents.
70. H. Fröhlich, J. K. Wegner, and A. Zell, *QSAR Comb. Sci.*, **23**, 311–318 (2004). Towards Optimal Descriptor Subset Selection with Support Vector Machines in Classification and Regression.
71. Y. Liu, *J. Chem. Inf. Comput. Sci.*, **44**, 1823–1828 (2004). A Comparative Study on Feature Selection Methods for Drug Discovery.
72. E. Byvatov and G. Schneider, *J. Chem. Inf. Comput. Sci.*, **44**, 993–999 (2004). SVM-based Feature Selection for Characterization of Focused Compound Collections.
73. S. Nandi, Y. Badhe, J. Lonari, U. Sridevi, B. S. Rao, S. S. Tambe, and B. D. Kulkarni, *Chem. Eng. J.*, **97**, 115–129 (2004). Hybrid Process Modeling and Optimization Strategies Integrating Neural Networks/Support Vector Regression and Genetic Algorithms: Study of Benzene Isopropylation on Hbeta Catalyst.
74. Y. Wang, I. V. Tetko, M. A. Hall, E. Frank, A. Facius, K. F. X. Mayer, and H. W. Mewes, *Comput. Biol. Chem.*, **29**, 37–46 (2005). Gene Selection from Microarray Data for Cancer Classification - A Machine Learning Approach.
75. N. Pochet, F. De Smet, J. A. K. Suykens, and B. L. R. De Moor, *Bioinformatics*, **20**, 3185–3195 (2004). Systematic Benchmarking of Microarray Data Classification: Assessing the Role of Non-linearity and Dimensionality Reduction.
76. G. Natsoulis, L. El Ghaoui, G. R. G. Lanckriet, A. M. Tolley, F. Leroy, S. Dunlea, B. P. Eynon, C. I. Pearson, S. Tugendreich, and K. Jarnagin, *Genome Res.*, **15**, 724–736 (2005). Classification of a Large Microarray Data Set: Algorithm Comparison and Analysis of Drug Signatures.
77. X. Zhou and K. Z. Mao, *Bioinformatics*, **21**, 1559–1564 (2005). LS Bound Based Gene Selection for DNA Microarray Data.
78. A. K. Jerebko, J. D. Malley, M. Franaszek, and R. M. Summers, *Acad. Radiol.*, **12**, 479–486 (2005). Support Vector Machines Committee Classification Method for Computer-aided Polyp Detection in CT Colonography.
79. K. Faceli, A. de Carvalho, and W. A. Silva, *Genet. Mol. Biol.*, **27**, 651–657 (2004). Evaluation of Gene Selection Metrics for Tumor Cell Classification.
80. L. B. Li, W. Jiang, X. Li, K. L. Moser, Z. Guo, L. Du, Q. J. Wang, E. J. Topol, Q. Wang, and S. Rao, *Genomics*, **85**, 16–23 (2005). A Robust Hybrid between Genetic Algorithm and Support Vector Machine for Extracting an Optimal Feature Gene Subset.
81. C. A. Tsai, C. H. Chen, T. C. Lee, I. C. Ho, U. C. Yang, and J. J. Chen, *DNA Cell Biol.*, **23**, 607–614 (2004). Gene Selection for Sample Classifications in Microarray Experiments.
82. T. Downs, K. E. Gates, and A. Masters, *J. Mach. Learn. Res.*, **2**, 293–297 (2001). Exact Simplification of Support Vector Solutions.
83. Y. Q. Zhan and D. G. Shen, *Pattern Recognit.*, **38**, 157–161 (2005). Design Efficient Support Vector Machine for Fast Classification.
84. C. Merkwirth, H. A. Mauser, T. Schulz-Gasch, O. Roche, M. Stahl, and T. Lengauer, *J. Chem. Inf. Comput. Sci.*, **44**, 1971–1978 (2004). Ensemble Methods for Classification in Cheminformatics.
85. H. Briem and J. Günther, *ChemBioChem*, **6**, 558–566 (2005). Classifying "Kinase Inhibitor-likeness" by Using Machine-learning Methods.
86. C. W. Yap and Y. Z. Chen, *J. Chem Inf. Model.*, **45**, 982–992 (2005). Prediction of Cytochrome P450 3A4, 2D6, and 2C9 Inhibitors and Substrates by Using Support Vector Machines.
87. G. Valentini, M. Muselli, and F. Ruffino, *Neurocomputing*, **56**, 461–466 (2004). Cancer Recognition with Bagged Ensembles of Support Vector Machines.
88. H. Saigo, J.-P. Vert, N. Ueda, and T. Akutsu, *Bioinformatics*, **20**, 1682–1689 (2004). Protein Homology Detection Using String Alignment Kernels.

89. C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, *Bioinformatics*, **20**, 467–476 (2004). Mismatch String Kernels for Discriminative Protein Classification.
90. J.-P. Vert, *Bioinformatics*, **18**, S276–S284 (2002). A Tree Kernel to Analyse Phylogenetic Profiles.
91. Z. R. Yang and K. C. Chou, *Bioinformatics*, **20**, 735–741 (2004). Bio-support Vector Machines for Computational Proteomics.
92. M. Wang, J. Yang, and K. C. Chou, *Amino Acids*, **28**, 395–402 (2005). Using String Kernel to Predict Peptide Cleavage Site Based on Subsite Coupling Model.
93. R. Teramoto, M. Aoki, T. Kimura, and M. Kanaoka, *FEBS Lett.*, **579**, 2878–2882 (2005). Prediction of siRNA Functionality Using Generalized String Kernel and Support Vector Machine.
94. C. Leslie and R. Kuang, *J. Mach. Learn. Res.*, **5**, 1435–1455 (2004). Fast String Kernels Using Inexact Matching for Protein Sequences.
95. K. Tsuda and W. S. Noble, *Bioinformatics*, **20**, i326–i333 (2004). Learning Kernels from Biological Networks by Maximizing Entropy.
96. A. Micheli, F. Portera, and A. Sperduti, *Neurocomputing*, **64**, 73–92 (2005). A Preliminary Empirical Comparison of Recursive Neural Networks and Tree Kernel Methods on Regression Tasks for Tree Structured Domains.
97. P. Mahé, N. Ueda, T. Akutsu, J.-L. Perret, and J.-P. Vert, *J. Chem Inf. Model.*, **45**, 939–951 (2005). Graph Kernels for Molecular Structure-Activity Relationship Analysis with Support Vector Machines.
98. B. J. Jain, P. Geibel, and F. Wysotzki, *Neurocomputing*, **64**, 93–105 (2005). SVM Learning with the Schur-Hadamard Inner Product for Graphs.
99. P. Lind and T. Maltseva, *J. Chem. Inf. Comput. Sci.*, **43**, 1855–1859 (2003). Support Vector Machines for the Estimation of Aqueous Solubility.
100. B. Hammer and K. Gersmann, *Neural Process. Lett.*, **17**, 43–53 (2003). A Note on the Universal Approximation Capability of Support Vector Machines.
101. J. P. Wang, Q. S. Chen, and Y. Chen, in *Advances in Neural Networks*, F. Yin, J. Wang, and C. Guo, Eds., Vol. 3173, Springer, New York, 2004, pp. 512–517. RBF Kernel Based Support Vector Machine with Universal Approximation and its Application.
102. T. B. Thompson, K. C. Chou, and C. Zheng, *J. Theor. Biol.*, **177**, 369–379 (1995). Neural Network Prediction of the HIV-1 Protease Cleavage Sites.
103. Z. R. Yang and K. C. Chou, *J. Chem. Inf. Comput. Sci.*, **43**, 1748–1753 (2003). Mining Biological Data Using Self-organizing Map.
104. Y. D. Cai, X. J. Liu, X. B. Xu, and K. C. Chou, *J. Comput. Chem.*, **23**, 267–274 (2002). Support Vector Machines for Predicting HIV Protease Cleavage Sites in Protein.
105. T. Rögnvaldsson and L. W. You, *Bioinformatics*, **20**, 1702–1709 (2004). Why Neural Networks Should not be Used for HIV-1 Protease Cleavage Site Prediction.
106. E. Urrestarazu Ramos, W. H. J. Vaes, H. J. M. Verhaar, and J. L. M. Hermens, *J. Chem. Inf. Comput. Sci.*, **38**, 845–852 (1998). Quantitative Structure-Activity Relationships for the Aquatic Toxicity of Polar and Nonpolar Narcotic Pollutants.
107. S. Ren, *Environ. Toxicol.*, **17**, 415–423 (2002). Classifying Class I and Class II Compounds by Hydrophobicity and Hydrogen Bonding Descriptors.
108. S. Ren and T. W. Schultz, *Toxicol. Lett.*, **129**, 151–160 (2002). Identifying the Mechanism of Aquatic Toxicity of Selected Compounds by Hydrophobicity and Electrophilicity Descriptors.
109. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **2**, 195–208 (2003). Aquatic Toxicity Prediction for Polar and Nonpolar Narcotic Pollutants with Support Vector Machines.
110. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **1**, 157–172 (2002). Support Vector Machine Identification of the Aquatic Toxicity Mechanism of Organic Compounds.
111. A. P. Bearden and T. W. Schultz, *Environ. Toxicol. Chem.*, **16**, 1311–1317 (1997). Structure-Activity Relationships for *Pimephales* and *Tetrahymena*: A Mechanism of Action Approach.

112. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **3**, 802–821 (2004). Support Vector Machines Prediction of the Mechanism of Toxic Action from Hydrophobicity and Experimental Toxicity Against *Pimephales promelas* and *Tetrahymena pyriformis*.
113. S. Ren, P. D. Frymier, and T. W. Schultz, *Ecotox. Environ. Safety*, **55**, 86–97 (2003). An Exploratory Study of the use of Multivariate Techniques to Determine Mechanisms of Toxic Action.
114. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **1**, 203–218 (2002). Support Vector Machine Classification of the Carcinogenic Activity of Polycyclic Aromatic Hydrocarbons.
115. R. S. Braga, P. M. V. B. Barone, and D. S. Galvão, *J. Mol. Struct. (THEOCHEM)*, **464**, 257–266 (1999). Identifying Carcinogenic Activity of Methylated Polycyclic Aromatic Hydrocarbons (PAHs).
116. P. M. V. B. Barone, R. S. Braga, A. Camilo Jr., and D. S. Galvão, *J. Mol. Struct. (THEOCHEM)*, **505**, 55–66 (2000). Electronic Indices from Semi-empirical Calculations to Identify Carcinogenic Activity of Polycyclic Aromatic Hydrocarbons.
117. R. Vendrame, R. S. Braga, Y. Takahata, and D. S. Galvão, *J. Mol. Struct. (THEOCHEM)*, **539**, 253–265 (2001). Structure-Carcinogenic Activity Relationship Studies of Polycyclic Aromatic Hydrocarbons (PAHs) with Pattern-Recognition Methods.
118. D. J. G. Marino, P. J. Peruzzo, E. A. Castro, and A. A. Toropov, *Internet Electron. J. Mol. Des.*, **1**, 115–133 (2002). QSAR Carcinogenic Study of Methylated Polycyclic Aromatic Hydrocarbons Based on Topological Descriptors Derived from Distance Matrices and Correlation Weights of Local Graph Invariants.
119. M. Chastrette and J. Y. D. Laumer, *Eur. J. Med. Chem.*, **26**, 829–833 (1991). Structure Odor Relationships Using Neural Networks.
120. M. Chastrette, C. El Aïdi, and J. F. Peyraud, *Eur. J. Med. Chem.*, **30**, 679–686 (1995). Tetralin, Indan and Nitrobenzene Compound Structure-musk Odor Relationship Using Neural Networks.
121. K. J. Rossiter, *Chem. Rev.*, **96**, 3201–3240 (1996). Structure-Odor Relationships.
122. D. Zakarya, M. Chastrette, M. Tollabi, and S. Fkih-Tetouani, *Chemometrics Intell. Lab. Syst.*, **48**, 35–46 (1999). Structure-Camphor Odour Relationships using the Generation and Selection of Pertinent Descriptors Approach.
123. R. D. M. C. Amboni, B. S. Junkes, R. A. Yunes, and V. E. F. Heinzen, *J. Agric. Food Chem.*, **48**, 3517–3521 (2000). Quantitative Structure-Odor Relationships of Aliphatic Esters Using Topological Indices.
124. G. Buchbauer, C. T. Klein, B. Wailzer, and P. Wolschann, *J. Agric. Food Chem.*, **48**, 4273–4278 (2000). Threshold-Based Structure-Activity Relationships of Pyrazines with Bell-Pepper Flavor.
125. B. Wailzer, J. Klocker, G. Buchbauer, G. Ecker, and P. Wolschann, *J. Med. Chem.*, **44**, 2805–2813 (2001). Prediction of the Aroma Quality and the Threshold Values of Some Pyrazines Using Artificial Neural Networks.
126. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **1**, 269–284 (2002). Structure–Odor Relationships for Pyrazines with Support Vector Machines.
127. A. O. Aptula, N. G. Jeliaskova, T. W. Schultz, and M. T. D. Cronin, *QSAR Comb. Sci.*, **24**, 385–396 (2005). The Better Predictive Model: High q^2 for the Training Set or Low Root Mean Square Error of Prediction for the Test Set?
128. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **4**, 928–947 (2005). QSAR for Phenols Toxicity to *Tetrahymena pyriformis* with Support Vector Regression and Artificial Neural Networks.
129. A. Carotti, C. Altornare, L. Savini, L. Chlasserini, C. Pellerano, M. P. Mascia, E. Maciocco, F. Busonero, M. Mameli, G. Biggio, and E. Sanna, *Bioorg. Med. Chem.*, **11**, 5259–5272 (2003). High Affinity Central Benzodiazepine Receptor Ligands. Part 3: Insights into the Pharmacophore and Pattern Recognition Study of Intrinsic Activities of Pyrazolo[4,3-*c*]quinolin-3-ones.

130. D. Hadjipavlou-Litina, R. Garg, and C. Hansch, *Chem. Rev.*, **104**, 3751–3793 (2004). Comparative Quantitative Structure-Activity Relationship Studies (QSAR) on Non-benzodiazepine Compounds Binding to Benzodiazepine Receptor (BzR).
131. L. Savini, P. Massarelli, C. Nencini, C. Pellerano, G. Biggio, A. Maciocco, G. Tuligi, A. Carrieri, N. Cinone, and A. Carotti, *Bioorg. Med. Chem.*, **6**, 389–399 (1998). High Affinity Central Benzodiazepine Receptor Ligands: Synthesis and Structure-Activity Relationship Studies of a New Series of Pyrazolo[4,3-*c*]quinolin-3-ones.
132. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **4**, 181–193 (2005). Support Vector Regression Quantitative Structure-Activity Relationships (QSAR) for Benzodiazepine Receptor Ligands.
133. T. I. Netzeva, J. C. Dearden, R. Edwards, A. D. P. Worgan, and M. T. D. Cronin, *J. Chem. Inf. Comput. Sci.*, **44**, 258–265 (2004). QSAR Analysis of the Toxicity of Aromatic Compounds to *Chlorella vulgaris* in a Novel Short-term Assay.
134. T. I. Netzeva, J. C. Dearden, R. Edwards, A. D. P. Worgan, and M. T. D. Cronin, *Bull. Environ. Contam. Toxicol.*, **73**, 385–391 (2004). Toxicological Evaluation and QSAR Modelling of Aromatic Amines to *Chlorella vulgaris*.
135. M. T. D. Cronin, T. I. Netzeva, J. C. Dearden, R. Edwards, and A. D. P. Worgan, *Chem. Res. Toxicol.*, **17**, 545–554 (2004). Assessment and Modeling of the Toxicity of Organic Chemicals to *Chlorella vulgaris*: Development of a Novel Database.
136. A. D. P. Worgan, J. C. Dearden, R. Edwards, T. I. Netzeva, and M. T. D. Cronin, *QSAR Comb. Sci.*, **22**, 204–209 (2003). Evaluation of a Novel Short-term Algal Toxicity Assay by the Development of QSARs and Inter-species Relationships for Narcotic Chemicals.
137. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **4**, 911–927 (2005). Artificial Neural Networks and Support Vector Regression Quantitative Structure-Activity Relationships (QSAR) for the Toxicity of Aromatic Compounds to *Chlorella vulgaris*.
138. O. Ivanciuc, *Rev. Roum. Chim.*, **43**, 347–354 (1998). Artificial Neural Networks Applications. Part 7 - Estimation of Bioconcentration Factors in Fish Using Solvatochromic Parameters.
139. X. X. Lu, S. Tao, J. Cao, and R. W. Dawson, *Chemosphere*, **39**, 987–999 (1999). Prediction of Fish Bioconcentration Factors of Nonpolar Organic Pollutants Based on Molecular Connectivity Indices.
140. S. Tao, H. Y. Hu, X. X. Lu, R. W. Dawson, and F. L. Xu, *Chemosphere*, **41**, 1563–1568 (2000). Fragment Constant Method for Prediction of Fish Bioconcentration Factors of Non-polar Chemicals.
141. S. D. Dimitrov, N. C. Dimitrova, J. D. Walker, G. D. Veith, and O. G. Mekenyan, *Pure Appl. Chem.*, **74**, 1823–1830 (2002). Predicting Bioconcentration Factors of Highly Hydrophobic Chemicals. Effects of Molecular Size.
142. S. D. Dimitrov, N. C. Dimitrova, J. D. Walker, G. D. Veith, and O. G. Mekenyan, *QSAR Comb. Sci.*, **22**, 58–68 (2003). Bioconcentration Potential Predictions Based on Molecular Attributes - An Early Warning Approach for Chemicals Found in Humans, Birds, Fish and Wildlife.
143. M. H. Fatemi, M. Jalali-Heravi, and E. Konuze, *Anal. Chim. Acta*, **486**, 101–108 (2003). Prediction of Bioconcentration Factor Using Genetic Algorithm and Artificial Neural Network.
144. P. Gramatica and E. Papa, *QSAR Comb. Sci.*, **22**, 374–385 (2003). QSAR Modeling of Bioconcentration Factor by Theoretical Molecular Descriptors.
145. O. Ivanciuc, *Internet Electron. J. Mol. Des.*, **4**, 813–834 (2005). Bioconcentration Factor QSAR with Support Vector Regression and Artificial Neural Networks.
146. S. S. Yang, W. C. Lu, N. Y. Chen, and Q. N. Hu, *J. Mol. Struct. (THEOCHEM)*, **719**, 119–127 (2005). Support Vector Regression Based QSPR for the Prediction of Some Physico-chemical Properties of Alkyl Benzenes.
147. K.-R. Müller, G. Rätsch, S. Sonnenburg, S. Mika, M. Grimm, and N. Heinrich, *J. Chem. Inf. Model.*, **45**, 249–253 (2005). Classifying ‘Drug-likeness’ with Kernel-based Learning Methods.

148. R. N. Jorissen and M. K. Gilson, *J. Chem Inf. Model.*, **45**, 549–561 (2005). Virtual Screening of Molecular Databases Using a Support Vector Machine.
149. R. Arimoto, M. A. Prasad, and E. M. Gifford, *J. Biomol. Screen*, **10**, 197–205 (2005). Development of CYP3A4 Inhibition Models: Comparisons of Machine-learning Techniques and Molecular Descriptors.
150. V. Svetnik, T. Wang, C. Tong, A. Liaw, R. P. Sheridan, and Q. Song, *J. Chem Inf. Model.*, **45**, 786–799 (2005). Boosting: An Ensemble Learning Tool for Compound Classification and QSAR Modeling.
151. C. W. Yap, C. Z. Cai, Y. Xue, and Y. Z. Chen, *Toxicol. Sci.*, **79**, 170–177 (2004). Prediction of Torsade-causing Potential of Drugs by Support Vector Machine Approach.
152. M. Tobita, T. Nishikawa, and R. Nagashima, *Bioorg. Med. Chem. Lett.*, **15**, 2886–2890 (2005). A Discriminant Model Constructed by the Support Vector Machine Method for HERG Potassium Channel Inhibitors.
153. M. J. Soricich, R. A. McKinnon, J. O. Miners, D. A. Winkler, and P. A. Smith, *J. Med. Chem.*, **47**, 5311–5317 (2004). Rapid Prediction of Chemical Metabolism by Human UDP-glucuronosyltransferase Isoforms Using Quantum Chemical Descriptors Derived with the Electro-negativity Equalization Method.
154. V. V. Zernov, K. V. Balakin, A. A. Ivaschenko, N. P. Savchuk, and I. V. Pletnev, *J. Chem. Inf. Comput. Sci.*, **43**, 2048–2056 (2003). Drug Discovery Using Support Vector Machines. The Case Studies of Drug-likeness, Agrochemical-likeness, and Enzyme Inhibition Predictions.
155. J. M. Kriegl, T. Arnhold, B. Beck, and T. Fox, *QSAR Comb. Sci.*, **24**, 491–502 (2005). Prediction of Human Cytochrome P450 Inhibition Using Support Vector Machines.
156. J. Aires-de-Sousa and J. Gasteiger, *J. Comb. Chem.*, **7**, 298–301 (2005). Prediction of Enantiomeric Excess in a Combinatorial Library of Catalytic Enantioselective Reactions.
157. H. Li, C. Y. Ung, C. W. Yap, Y. Xue, Z. R. Li, Z. W. Cao, and Y. Z. Chen, *Chem. Res. Toxicol.*, **18**, 1071–1080 (2005). Prediction of Genotoxicity of Chemical Compounds by Statistical Learning Methods.
158. C. Helma, T. Cramer, S. Kramer, and L. De Raedt, *J. Chem. Inf. Comput. Sci.*, **44**, 1402–1411 (2004). Data Mining and Machine Learning Techniques for the Identification of Mutagenicity Inducing Substructures and Structure Activity Relationships of Noncongeneric Compounds.
159. T. C. Martin, J. Moecks, A. Belousov, S. Cawthraw, B. Dolenko, M. Eiden, J. von Frese, W. Köhler, J. Schmitt, R. Somorjai, T. Udelhoven, S. Verzakov, and W. Petrich, *Analyst*, **129**, 897–901 (2004). Classification of Signatures of Bovine Spongiform Encephalopathy in Serum Using Infrared Spectroscopy.
160. J. A. F. Pierna, V. Baeten, A. M. Renier, R. P. Cogdill, and P. Dardenne, *J. Chemometr.*, **18**, 341–349 (2004). Combination of Support Vector Machines (SVM) and Near-infrared (NIR) Imaging Spectroscopy for the Detection of Meat and Bone Meal (MBM) in Compound Feeds.
161. S. Zomer, R. G. Brereton, J. F. Carter, and C. Eckers, *Analyst*, **129**, 175–181 (2004). Support Vector Machines for the Discrimination of Analytical Chemical Data: Application to the Determination of Tablet Production by Pyrolysis-gas Chromatography-mass Spectrometry.
162. S. Zomer, C. Guillo, R. G. Brereton, and M. Hanna-Brown, *Anal. Bioanal. Chem.*, **378**, 2008–2020 (2004). Toxicological Classification of Urine Samples Using Pattern Recognition Techniques and Capillary Electrophoresis.
163. U. Thissen, B. Üstün, W. J. Melssen, and L. M. C. Buydens, *Anal. Chem.*, **76**, 3099–3105 (2004). Multivariate Calibration with Least-Squares Support Vector Machines.
164. F. Chauchard, R. Cogdill, S. Roussel, J. M. Roger, and V. Bellon-Maurel, *Chemometrics Intell. Lab. Syst.*, **71**, 141–150 (2004). Application of LS-SVM to Non-linear Phenomena in NIR Spectroscopy: Development of a Robust and Portable Sensor for Acidity Prediction in Grapes.
165. U. Thissen, M. Pepers, B. Üstün, W. J. Melssen, and L. M. C. Buydens, *Chemometrics Intell. Lab. Syst.*, **73**, 169–179 (2004). Comparing Support Vector Machines to PLS for Spectral Regression Applications.

166. S. Zomer, M. D. N. Sánchez, R. G. Brereton, and J. L. P. Pavón, *J. Chemometr.*, **18**, 294–305 (2004). Active Learning Support Vector Machines for Optimal Sample Selection in Classification.
167. H. L. Zhai, H. Gao, X. G. Chen, and Z. D. Hu, *Anal. Chim. Acta*, **546**, 112–118 (2005). An Assisted Approach of the Global Optimization for the Experimental Conditions in Capillary Electrophoresis.
168. K. Brudzewski, S. Osowski, and T. Markiewicz, *Sens. Actuators B*, **98**, 291–298 (2004). Classification of Milk by Means of an Electronic Nose and SVM Neural Network.
169. O. Sadik, W. H. Land, A. K. Wanekaya, M. Uematsu, M. J. Embrechts, L. Wong, D. Leibensperger, and A. Volykin, *J. Chem. Inf. Comput. Sci.*, **44**, 499–507 (2004). Detection and Classification of Organophosphate Nerve Agent Simulants Using Support Vector Machines with Multiarray Sensors.
170. C. Distante, N. Ancona, and P. Siciliano, *Sens. Actuators B*, **88**, 30–39 (2003). Support Vector Machines for Olfactory Signals Recognition.
171. M. Pardo and G. Sberveglieri, *Sens. Actuators B*, **107**, 730–737 (2005). Classification of Electronic Nose Data with Support Vector Machines.
172. K. Brudzewski, S. Osowski, T. Markiewicz, and J. Ulaczyk, *Sens. Actuators B*, **113**, 135–141 (2006). Classification of Gasoline with Supplement of Bio-products by Means of an Electronic Nose and SVM Neural Network.
173. M. Bicego, *Sens. Actuators B*, **110**, 225–230 (2005). Odor Classification Using Similarity-based Representation.
174. T. B. Trafalis, O. Oladunni, and D. V. Papavassiliou, *Ind. Eng. Chem. Res.*, **44**, 4414–4426 (2005). Two-phase Flow Regime Identification with a Multiclassification Support Vector Machine (SVM) Model.
175. D. E. Lee, J. H. Song, S. O. Song, and E. S. Yoon, *Ind. Eng. Chem. Res.*, **44**, 2101–2105 (2005). Weighted Support Vector Machine for Quality Estimation in the Polymerization Process.
176. Y. H. Chu, S. J. Qin, and C. H. Han, *Ind. Eng. Chem. Res.*, **43**, 1701–1710 (2004). Fault Detection and Operation Mode Identification Based on Pattern Classification with Variable Selection.
177. I. S. Han, C. H. Han, and C. B. Chung, *J. Appl. Polym. Sci.*, **95**, 967–974 (2005). Melt Index Modeling with Support Vector Machines, Partial Least Squares, and Artificial Neural Networks.
178. S. Mika and B. Rost, *Nucleic Acids Res.*, **32**, W634–W637 (2004). NLProt: Extracting Protein Names and Sequences from Papers.
179. S. Mika and B. Rost, *Bioinformatics*, **20**, i241–i247 (2004). Protein Names Precisely Peeled off Free Text.
180. L. Shi and F. Campagne, *BMC Bioinformatics*, **6**, 88 (2005). Building a Protein Name Dictionary from Full Text: A Machine Learning Term Extraction Approach.
181. I. Donaldson, J. Martin, B. de Bruijn, C. Wolting, V. Lay, B. Tuekam, S. D. Zhang, B. Baskin, G. D. Bader, K. Michalickova, T. Pawson, and C. W. V. Hogue, *BMC Bioinformatics*, **4**, (2003). PreBIND and Textomy - Mining the Biomedical Literature for Protein-protein Interactions Using a Support Vector Machine.
182. K. Takeuchi and N. Collier, *Artif. Intell. Med.*, **33**, 125–137 (2005). Bio-medical Entity Extraction Using Support Vector Machines.
183. R. Bunescu, R. F. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani, and Y. W. Wong, *Artif. Intell. Med.*, **33**, 139–155 (2005). Comparative Experiments on Learning Information Extractors for Proteins and Their Interactions.
184. T. Joachims, in *Advances in Kernel Methods — Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds., MIT Press, Cambridge, Massachusetts, 1999. Making Large-scale SVM Learning Practical.
185. R. Collobert and S. Bengio, *J. Mach. Learn. Res.*, **1**, 143–160 (2001). SVMtorch: Support Vector Machines for Large-scale Regression Problems.