# What is a support vector machine?

William S Noble

**Support vector machines (SVMs) are becoming popular in a wide variety of biological applications. But, what exactly are SVMs and how do they work? And what are their most promising applications in the life sciences?**

A support vector machine (SVM) is a computer algorithm that learns by example to assign labels to objects[1]. For instance, an SVM can learn to recognize fraudulent credit card activity by examining hundreds or thousands of fraudulent and nonfraudulent credit card activity reports. Alternatively, an SVM can learn to recognize handwritten digits by examining a large collection of scanned images of handwritten zeroes, ones and so forth. SVMs have also been successfully applied to an increasingly wide variety of biological applications. A common biomedical application of support vector machines is the automatic classification of microarray gene expression profiles. Theoretically, an SVM can examine the gene expression profile derived from a tumor sample or from peripheral fluid and arrive at a diagnosis or prognosis. Throughout this primer, I will use as a motivating example a seminal study of acute leukemia expression profiles[2]. Other biological applications of SVMs involve classifying objects as diverse as protein and DNA sequences, microarray expression profiles and mass spectra[3].

In essence, an SVM is a mathematical entity, an algorithm (or recipe) for maximizing a particular mathematical function with respect to a given collection of data. The basic ideas behind the SVM algorithm, however, can be explained without ever reading an equation. Indeed, I claim that, to understand the essence of SVM classification, one needs only to grasp four basic concepts: (i) the separating hyperplane, (ii) the maximum-margin hyperplane, (iii) the soft margin and (iv) the kernel function.

Before describing an SVM, let's return to the problem of classifying cancer gene expression profiles. The Affymetrix microarrays employed by Golub *et al.*[2] contained probes for 6,817 human genes. For a given bone marrow sample, the microarray assay returns 6,817 values, each of which represents the mRNA levels corresponding to a given gene. Golub *et al.* performed this assay on 38 bone marrow samples, 27 from individuals with acute lymphoblastic leukemia (ALL) and 11 from individuals with acute myeloid leukemia (AML). These data represent a good start to 'train' an SVM to tell the difference between ALL and AML expression profiles. If the learning is successful, then the SVM will be able to successfully diagnose a new patient as AML or ALL based upon its bone marrow expression profile.

For now, to allow an easy, geometric interpretation of the data, imagine that the microarrays contained probes for only two genes. In this case, our gene expression profiles consist of two numbers, which can be easily plotted (**Fig. 1a**). Based upon results from a previous study[4], I have selected the genes *ZYX* and *MARCKSL1*. In the figure, values are proportional to the intensity of the fluorescence on the microarray, so on either axis, a large value indicates that the gene is highly expressed and vice versa. The expression levels are indicated by a red or green dot, depending upon whether the sample is from a patient with ALL or AML. The SVM must learn to tell the difference between the two groups and, given an unlabeled expression vector, such as the one labeled 'Unknown' in the figure, predict whether it corresponds to a patient with ALL or AML.

## The separating hyperplane

The human eye is very good at pattern recognition. Even a quick glance at **Figure 1a** shows that the AML profiles form a cluster in the upper left region of the plot, and the ALL profiles cluster in the lower right. A simple rule might state that a patient has AML if the expression level of *MARCKSL1* is twice as high as the expression level of *ZYX*, and vice versa for ALL. Geometrically, this rule corresponds to drawing a line between the two clusters (**Fig. 1b**). Subsequently, predicting the label of an unknown expression profile is easy: one simply needs to ask whether the new profile falls on the ALL or the AML side of this separating line.

Now, to define the notion of a separating hyperplane, consider a situation in which the microarray does not contain just two genes. For example, if the microarray contains a single gene, then the 'space' in which the corresponding one-dimensional expression profiles reside is a one-dimensional line. We can divide this line in half by using a single point (**Fig. 1c**). In two dimensions (**Fig. 1b**), a straight line divides the space in half, and in three dimensions, we need a plane to divide the space (**Fig. 1d**). We can extrapolate this procedure mathematically to higher dimensions. The general term for a straight line in a high-dimensional space is a hyperplane, and so the separating hyperplane is, essentially, the line that separates the ALL and AML samples.

## The maximum-margin hyperplane

The concept of treating the objects to be classified as points in a high-dimensional space and finding a line that separates them is not unique to the SVM. The SVM, however, is different from other hyperplane-based classifiers by virtue of how the hyperplane is selected. Consider again the classification problem portrayed in **Figure 1a**. We have now established that the goal of the SVM is to identify a line that separates the ALL from the AML expression profiles in this two-dimensional space. However, many such lines exist (**Fig. 1e**). Which one provides the best classifier?

With some thought, one may come up with the simple idea of selecting the line that is, roughly speaking, 'in the middle'. In other words, one would select the line that separates the two classes but adopts the maximal distance from any one of the given expression profiles (**Fig. 1f**). It turns out that a theorem from the

*William S. Noble is in the Departments of Genome Sciences and of Computer Science and Engineering, University of Washington, 1705 NE Pacific Street, Seattle, Washington 98195, USA. e-mail: noble@gs.washington.edu*

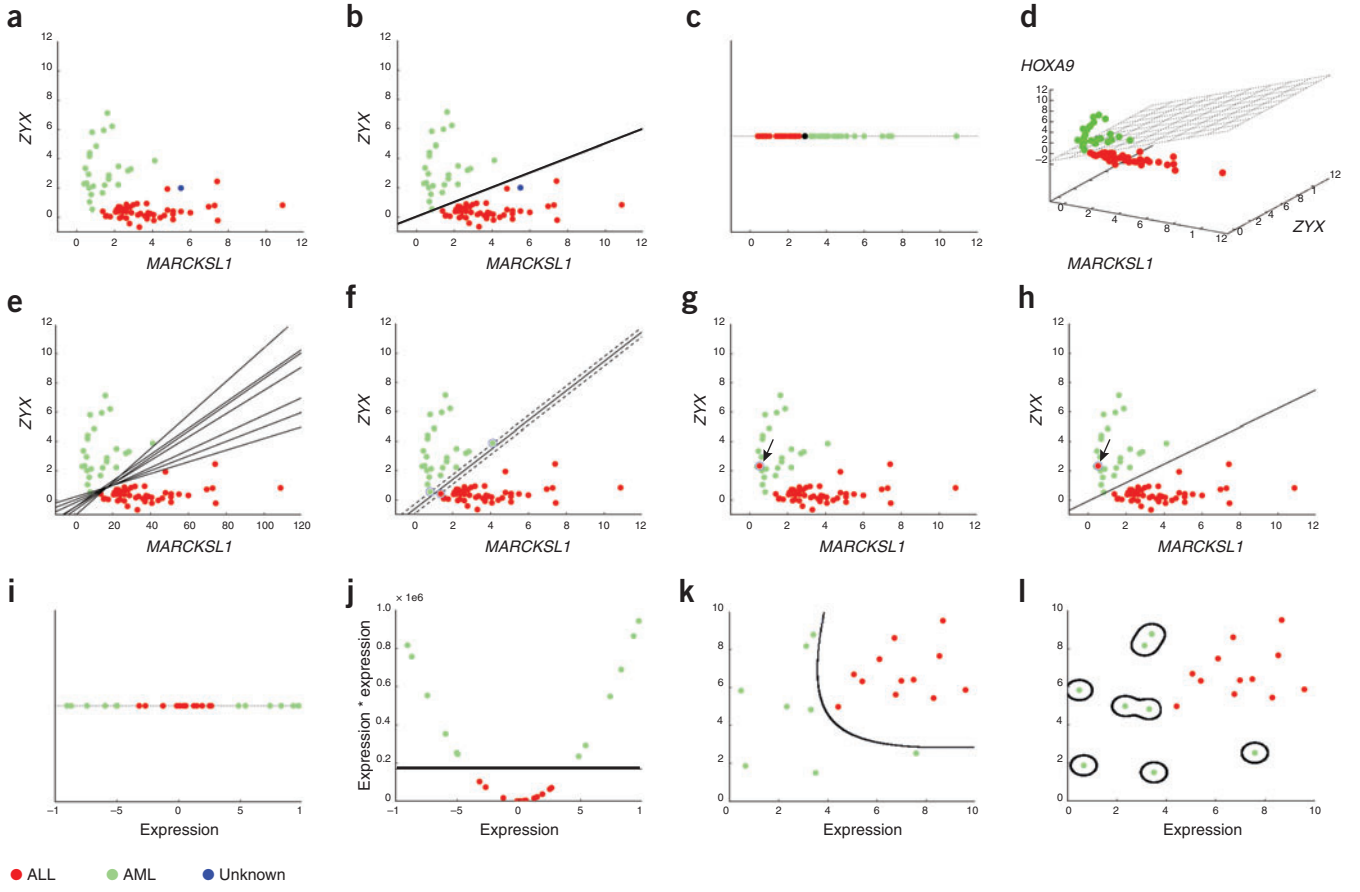


**Figure 1** Support vector machines (SVMs) at work. (**a**) Two-dimensional expression profiles of lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML) samples. Each dimension corresponds to the measured mRNA expression level of a given gene. The SVM's task is to assign a label to the gene expression profile labeled 'Unknown'. (**b**) A separating hyperplane. Based upon this hyperplane, the inferred label of the 'Unknown' expression profile is 'ALL'. (**c**) A hyperplane in one dimension. The hyperplane is shown as a single black point. (**d**) A hyperplane in three dimensions. (**e**) Many possible separating hyperplanes. (**f**) The maximum-margin hyperplane. The three support vectors are circled. (**g**) A data set containing one error, indicated by arrow. (**h**) A separating hyperplane with a soft margin. Error is indicated by arrow. (**i**) A nonseparable one-dimensional data set. (**j**) Separating previously nonseparable data. (**k**) A linearly nonseparable two-dimensional data set, which is linearly separable in four dimensions. (**l**) An SVM that has overfit a two-dimensional data set. In **a**, **b**, **d**–**h**, the expression values are divided by 1,000.



field of statistical learning theory supports exactly this choice[5]. If we define the distance from the separating hyperplane to the nearest expression vector as the margin of the hyperplane, then the SVM selects the maximum-margin separating hyperplane. Selecting this particular hyperplane maximizes the SVM's ability to predict the correct classification of previously unseen examples.

This theorem is, in many ways, the key to an SVM's success. Let's take a minute, therefore, to consider some caveats that come with it. First, the theorem assumes that the data on which the SVM is trained are drawn from the same distribution as the data on which it is tested. This is reasonable, since we cannot expect, for example, an SVM trained on microarray data to be able to classify mass spectrometry data. More relevantly, we cannot expect the SVM to perform well if the bone marrow samples for the training data set were prepared using a different protocol than the samples for the test data set. On the other hand, the theorem does not assume that the two data sets were drawn from a particular class of distributions. For example, an SVM does not assume that the training data values follow a normal distribution.

## The soft margin

So far, we have assumed that the data can be separated using a straight line. Of course, many real data sets cannot be separated as cleanly; instead, they look like the one in **Figure 1g**, where the data set contains an 'error', the circled gene expression profile. Intuitively, we would like the SVM to be able to deal with errors in the data by allowing a few anomalous expression profiles to fall on the 'wrong side' of the separating hyperplane. To handle cases like these, the SVM algorithm has to be modified by adding a 'soft margin'. Essentially, this allows some data points to push their way through the margin of the separating hyperplane without affecting the final result. **Figure 1h** shows the soft margin solution to the problem in **Figure 1g**. The one outlier now resides on the same side of the line with members of the opposite class.

Of course, we don't want the SVM to allow for too many misclassifications. Hence, introducing the soft margin necessitates introducing a user-specified parameter that controls, roughly, how many examples are allowed to violate the separating hyperplane and how far across the line they are allowed to go. Setting this parameter is complicated by the fact that we still want to try to achieve a large margin with respect to the correctly classified examples. Hence, the soft margin parameter specifies a trade-off between hyperplane violations and the size of the margin.

## The kernel function

To understand the notion of a kernel function, we are going to simplify our example set even

further. Rather than a microarray containing two genes, let's assume that we now have only a single gene expression measurement (**Fig. 1c**). In that case, the maximum-margin-separating hyperplane is a single point. **Figure 1i** illustrates the analogous case of a nonseparable data set. Here, the AML values are grouped near zero, and the ALL examples have large absolute values. The problem is that no single point can separate the two classes, and introducing a soft margin would not help.

The kernel function provides a solution to this problem by adding an additional dimension to the data (**Fig. 1j**). In this case, to get the new dimension, the original expression values were simply squared. Fortuitously, as shown in the figure, this simple step separates the ALL and AML examples with a straight line in the two-dimensional space. In essence, the kernel function is a mathematical trick that allows the SVM to perform a 'two-dimensional' classification of a set of originally one-dimensional data. In general, a kernel function projects data from a low-dimensional space to a space of higher dimension. If one is lucky (or smart) and chooses a good kernel function, the data will become separable in the resulting higher dimensional space.

To understand kernels a bit better, consider the two-dimensional data set shown in **Figure 1k**. These data cannot be separated using a straight line, but a relatively simple kernel function that projects the data from the two-dimensional space up to four dimensions (corresponding to the products of all pairs of features) allows the data to be linearly separated. We cannot draw the data in a four-dimensional space, but we can project the SVM hyperplane in that space back down to the original two-dimensional space. The result is shown as the curved line in **Figure 1k**.

It is possible to prove that, for any given data set with consistent labels (where consistent simply means that the data set does not contain two identical objects with opposite labels) there exists a kernel function that will allow the data to be linearly separated. This observation begs the question of why not always project the data into a very high-dimensional space, to be sure of finding a separating hyperplane. This would take care of the need for soft margins, whereas the original theorem would still apply.

This is a reasonable suggestion—however, projecting into very high-dimensional spaces can be problematic, due to the so-called curse of dimensionality: as the number of variables under consideration increases, the number of possible solutions also increases, but exponentially. Consequently, it becomes harder for any algorithm to select a correct solution. **Figure 1l** shows what happens when a data set is projected into a space with too many dimensions. The figure contains the same data as **Figure 1k**, but the projected hyperplane comes from an SVM that uses a very high-dimensional kernel function. The result is that the boundary between the classes is very specific to the examples in the training data set. The SVM is said to have overfit the data. Clearly, such an SVM will not generalize well when presented with new gene expression profiles.

This observation brings us to the largest practical difficulty encountered when applying an SVM classifier to a new data set. One would like to use a kernel function that is likely to allow the data to be separated but without introducing too many irrelevant dimensions. How is such a function best chosen? Unfortunately, in most cases, the only realistic answer is trial and error. Investigators typically begin with a simple SVM, and then experiment with a variety of 'standard' kernel functions. An optimal kernel function can be selected from a fixed set of kernels in a statistically rigorous fashion by using cross-validation. However, this approach is time-consuming and does not guarantee that a kernel function that was not considered in the cross-validation procedure would not perform better.

In addition to allowing SVMs to handle nonlinearly separable data sets and to incorporate prior knowledge, the kernel function yields at least two additional benefits. First, kernels can be defined on inputs that are not vectors. This ability to handle nonvector data is critical in biological applications, allowing the SVM to classify DNA and protein sequences, nodes in metabolic, regulatory and protein-protein interaction networks and microscopy images. Second, kernels provide a mathematical formalism for combining different types of data. Imagine, for example, that we are doing biomarker discovery for the ALL/AML distinction, and we have the Golub *et al.* data plus a corresponding collection of mass spectrometry profiles from the same set of patients. It turns out that we can use simple algebra to combine a kernel on microarray data with a kernel on mass spectrometry data. The resulting joint kernel would allow us to train a single SVM to perform classification on both types of data simultaneously.

## Extensions of the SVM algorithm

The most obvious drawback to the SVM algorithm, as described thus far, is that it apparently only handles binary classification problems. We can discriminate between ALL and AML, but how do we discriminate among a large variety of cancer classes? Generalizing to multiclass classification is straightforward and can be accomplished by using any of a variety of methods. Perhaps the simplest approach is to train multiple, one-versus-all classifiers. Essentially, to recognize three classes, A, B and C, we simply have to train three separate SVMs to answer the binary questions, "Is it A?," "Is it B?" and "Is it C?" This simple approach actually works quite well for cancer classification[6]. More sophisticated approaches also exist, which generalize the SVM optimization algorithm to account for multiple classes.

For data sets of thousands of examples, solving the SVM optimization problem is quite fast. Empirically, running times of state-of-the-art SVM learning algorithms scale approximately quadratically, which means that when you give the SVM twice as much data, it requires four times as long to run. SVMs have been successfully trained on data sets containing approximately one million examples, and fast approximation algorithms exist that scale almost linearly and perform nearly as well as the SVM[7].

## Conclusion

Using all 6,817 gene expression measurements, an SVM can achieve near-perfect classification accuracy on the ALL/AML data set[8]. Furthermore, an even larger data set has been used to demonstrate that SVMs also perform better than a variety of alternative methods for cancer classification from microarray expression profiles[6]. Although this primer has focused on cancer classification from gene expression profiles, SVM analysis can be applied to a wide variety of biological data. As we have seen, the SVM boasts a strong theoretical underpinning, coupled with remarkable empirical results across a growing spectrum of applications. Thus, SVMs will probably continue to yield valuable insights into the growing quantity and variety of molecular biology data.

ACKNOWLEDGEMENTS

The author thanks the support from NSF award IIS-0093302

1. Boser, B.E., Guyon, I.M. & Vapnik, V.N. A training algorithm for optimal margin classifiers. in *5th Annual ACM Workshop on COLT* (ed. Haussler, D.) 144–152 (ACM Press, Pittsburgh, PA, 1992).
2. Golub, T.R. *et al.* Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* **286**, 531–537 (1999).
3. Noble, W.S. Support vector machine applications in computational biology. in *Kernel Methods in Computational Biology* (eds. Schoelkopf, B., Tsuda, K. & Vert, J.-P.) 71–92 (MIT Press, Cambridge, MA, 2004).
4. Guyon, I., Weston, J., Barnhill, S. & Vapnik, V. Gene selection for cancer classification using support vector machines. *Mach. Learn.* **46**, 389–422 (2002).
5. Vapnik, V. & Lerner, A. Pattern recognition using generalized portrait method. *Autom. Remote Control* **24**, 774–780, 1963.
6. Ramaswamy, S. *et al.* Multiclass cancer diagnosis using tumor gene expression signatures. *Proc. Natl. Acad. Sci. USA* **98**, 15149–15154 (2001).
7. Bordes, A., Ertekin, S., Weston, J. & Bottou, L. Fast kernel classifiers with online and active learning. *J. Mach. Learn. Res.* **6**, 1579–1619 (2005).
8. Furey, T.S. *et al.* Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics* **16**, 906–914 (2000).