

For comparison with other error functions, we can divide by  $\ln(2)$  so that the error function passes through the point  $(0, 1)$ . This rescaled error function is also plotted in Figure 7.5 and we see that it has a similar form to the support vector error function. The key difference is that the flat region in  $E_{SV}(yt)$  leads to sparse solutions.

Both the logistic error and the hinge loss can be viewed as continuous approximations to the misclassification error. Another continuous error function that has sometimes been used to solve classification problems is the squared error, which is again plotted in Figure 7.5. It has the property, however, of placing increasing emphasis on data points that are correctly classified but that are a long way from the decision boundary on the correct side. Such points will be strongly weighted at the expense of misclassified points, and so if the objective is to minimize the misclassification rate, then a monotonically decreasing error function would be a better choice.

### 7.1.3 Multiclass SVMs

The support vector machine is fundamentally a two-class classifier. In practice, however, we often have to tackle problems involving  $K > 2$  classes. Various methods have therefore been proposed for combining multiple two-class SVMs in order to build a multiclass classifier.

One commonly used approach (Vapnik, 1998) is to construct  $K$  separate SVMs, in which the  $k^{\text{th}}$  model  $y_k(\mathbf{x})$  is trained using the data from class  $C_k$  as the positive examples and the data from the remaining  $K - 1$  classes as the negative examples. This is known as the *one-versus-the-rest* approach. However, in Figure 4.2 we saw that using the decisions of the individual classifiers can lead to inconsistent results in which an input is assigned to multiple classes simultaneously. This problem is sometimes addressed by making predictions for new inputs  $\mathbf{x}$  using

$$y(\mathbf{x}) = \max_k y_k(\mathbf{x}). \quad (7.49)$$

Unfortunately, this heuristic approach suffers from the problem that the different classifiers were trained on different tasks, and there is no guarantee that the real-valued quantities  $y_k(\mathbf{x})$  for different classifiers will have appropriate scales.

Another problem with the one-versus-the-rest approach is that the training sets are imbalanced. For instance, if we have ten classes each with equal numbers of training data points, then the individual classifiers are trained on data sets comprising 90% negative examples and only 10% positive examples, and the symmetry of the original problem is lost. A variant of the one-versus-the-rest scheme was proposed by Lee *et al.* (2001) who modify the target values so that the positive class has target  $+1$  and the negative class has target  $-1/(K - 1)$ .

Weston and Watkins (1999) define a single objective function for training all  $K$  SVMs simultaneously, based on maximizing the margin from each to remaining classes. However, this can result in much slower training because, instead of solving  $K$  separate optimization problems each over  $N$  data points with an overall cost of  $O(KN^2)$ , a single optimization problem of size  $(K - 1)N$  must be solved giving an overall cost of  $O(K^2N^2)$ .

Another approach is to train  $K(K - 1)/2$  different 2-class SVMs on all possible pairs of classes, and then to classify test points according to which class has the highest number of 'votes', an approach that is sometimes called *one-versus-one*. Again, we saw in Figure 4.2 that this can lead to ambiguities in the resulting classification. Also, for large  $K$  this approach requires significantly more training time than the one-versus-the-rest approach. Similarly, to evaluate test points, significantly more computation is required.

The latter problem can be alleviated by organizing the pairwise classifiers into a directed acyclic graph (not to be confused with a probabilistic graphical model) leading to the *DAGSVM* (Platt *et al.*, 2000). For  $K$  classes, the DAGSVM has a total of  $K(K - 1)/2$  classifiers, and to classify a new test point only  $K - 1$  pairwise classifiers need to be evaluated, with the particular classifiers used depending on which path through the graph is traversed.

A different approach to multiclass classification, based on error-correcting output codes, was developed by Dietterich and Bakiri (1995) and applied to support vector machines by Allwein *et al.* (2000). This can be viewed as a generalization of the voting scheme of the one-versus-one approach in which more general partitions of the classes are used to train the individual classifiers. The  $K$  classes themselves are represented as particular sets of responses from the two-class classifiers chosen, and together with a suitable decoding scheme, this gives robustness to errors and to ambiguity in the outputs of the individual classifiers. Although the application of SVMs to multiclass classification problems remains an open issue, in practice the one-versus-the-rest approach is the most widely used in spite of its ad-hoc formulation and its practical limitations.

There are also *single-class* support vector machines, which solve an unsupervised learning problem related to probability density estimation. Instead of modelling the density of data, however, these methods aim to find a smooth boundary enclosing a region of high density. The boundary is chosen to represent a quantile of the density, that is, the probability that a data point drawn from the distribution will land inside that region is given by a fixed number between 0 and 1 that is specified in advance. This is a more restricted problem than estimating the full density but may be sufficient in specific applications. Two approaches to this problem using support vector machines have been proposed. The algorithm of Schölkopf *et al.* (2001) tries to find a hyperplane that separates all but a fixed fraction  $\nu$  of the training data from the origin while at the same time maximizing the distance (margin) of the hyperplane from the origin, while Tax and Duin (1999) look for the smallest sphere in feature space that contains all but a fraction  $\nu$  of the data points. For kernels  $k(\mathbf{x}, \mathbf{x}')$  that are functions only of  $\mathbf{x} - \mathbf{x}'$ , the two algorithms are equivalent.

### 7.1.4 SVMs for regression

We now extend support vector machines to regression problems while at the same time preserving the property of sparseness. In simple linear regression we