Best Classification Estimator for Small Samples in Pattern Recogniton

Bruce R. Linnell, PhD

2005

In PR, you need "samples" of each kind of class as "examples" for the classifier, so that it can learn how to differentiate one class from another. Here, $\mathbf{N_C}$ represents the number of examples per class. In addition, every example has so many "features" that describe it - for example, features that describe people could include their height, weight, and eye, skin, and hair color. The number of features is also called the number of dimensions (**D**).

The "sample ratio" (SR) represents the number of examples of each class you have per dimension ($SR = \mathbf{N_C} / \mathbf{D}$). In PR, things work very well as long as the SR is ≥5, not so well for SR=2-5, poorly for SR=1-2, and very badly for SR<1. The problem is, sometimes it is very time-consuming and/or expensive to get a single sample, but very easy to extract lots of data (features, dimensions) about each sample, so you can often end up with small-SR situations.

There are two issues when it comes to figuring out how well a classifier is going to work when you only have a few examples of each class (SR < 2) : what classifier works best, and what estimator works best. What I discovered is that of all the "traditional" classifiers (I did not test neural networks (NN) nor the support vector machine), **the linear classifier works best**. If the SR is less than one, use the standard matrix pseudo-inverse to calculate the discriminant function.

Of many different estimators tested, I found that the **average of the resubstitution and leave-one-out classifiers provided the best estimate of future classification success when the SR is small**. In a nutshell, this is because resubstitution is optimistic, and gets more and more optimistic as the SR decreases. On the other hand, leave-one-out gets more and more pessimistic as the SR decreases, and amazingly, together their biases pretty much cancel each other out. *Even more amazing, a 50/50 weighting (in other words, the arithmetic mean of the two) turns out to be the best possible weighting!*

As a by-the-way, I have always used the good old-fashioned quadratic classifier whenever I can in all my PR work. It has consistently worked well, even if the data isn't truly Gaussian (as long as the SR is ≥5). I once had the opportunity to have someone else try to classify some e-nose data using the latest, fanciest neural network algorithms, and their NN had the same classification success rate as the quadratic classifier! Why is this such a big deal? Because NN's are very "finicky" – they have many parameters that need to be set (initial neuron values, number of hidden neurons, etc.) that can significantly affect the final classification success, and they are iterative – it takes many cycles for a NN to "settle down" to its final state. What that means is that it takes a very long time in order to get any kind of accurate estimation of the future classification success rate (which is itself an iterative process). In contrast, the quadratic classifier has no parameters, and compared to NN's, calculating the classifier takes an instant. While it is well-proven that NN's and the SVM work better for small-SR conditions, and/or when the data has an unusual distribution, the point is : don't use a new technology just because it's a new technology.