Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

PERGAMON

Pattern Recognition 37 (2004) 1267–1281

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Bolstered error estimation

Ulisses Braga-Neto[a,c], Edward Dougherty[b,c,*]

[a] Section of Clinical Cancer Genetics, UT MD Anderson Cancer Center, Houston, TX, USA
[b] Department of Pathology, UT MD Anderson Cancer Center, Houston, TX, USA
[c] Department of Electrical Engineering, Texas A& M University, College Station, TX, USA

Received 14 July 2003; received in revised form 13 August 2003; accepted 13 August 2003

Abstract

We propose a general method for error estimation that displays low variance and generally low bias as well. This method is based on "bolstering" the original empirical distribution of the data. It has a direct geometric interpretation and can be easily applied to any classification rule and any number of classes. This method can be used to improve the performance of any error-counting estimation method, such as resubstitution and all cross-validation estimators, particularly in small-sample settings. We point out some similarities shared by our method with a previously proposed technique, known as smoothed error estimation. In some important cases, such as a linear classification rule with a Gaussian bolstering kernel, the integrals in the bolstered error estimate can be computed exactly. In the general case, the bolstered error estimate may be computed by Monte-Carlo sampling; however, our experiments show that a very small number of Monte-Carlo samples is needed. This results in a fast error estimator, which is in contrast to other resampling techniques, such as the bootstrap. We provide an extensive simulation study comparing the proposed method with resubstitution, cross-validation, and bootstrap error estimation, for three popular classification rules (linear discriminant analysis, k-nearest-neighbor, and decision trees), using several sample sizes, from small to moderate. The results indicate the proposed method vastly improves on resubstitution and cross-validation, especially for small samples, in terms of bias and variance. In that respect, it is competitive with, and in many occasions superior to, bootstrap error estimation, while being tens to hundreds of times faster. We provide a companion web site, which contains: (1) the complete set of tables and plots regarding the simulation study, and (2) C source code used to implement the bolstered error estimators proposed in this paper, as part of a larger library for classification and error estimation, with full documentation and examples. The companion web site can be accessed at the URL http://ee.tamu.edu/~edward/bolster.

Keywords: Bolstering; Error estimation; Classification; Resubstitution; Leave-one-out; Cross-validation; Bootstrap

1. Introduction

Given a classifier designed on an i.i.d. sample, one wants to estimate its error rate with respect to the underlying unknown population in a way that is fast and accurate. This is a major issue in pattern recognition that impacts not only the classification problem at hand, but also model selection (choice of parameters in classifier design) and variable selection (choice of interesting features). Error estimation is especially difficult in practical small-sample settings,where one cannot hold out data from classifier design for testing [1].

Commonly used error estimators include resubstitution, cross-validation methods, and bootstrap methods [2–7]. In this paper, we propose a general extension of an error-estimation technique initially proposed for linear discriminant analysis (LDA) [8]. It has advantages with respect to the aforementioned error estimators in terms of speed and accuracy (bias and variance). The difference

* Corresponding author. Tel.: +1-979-862-8896; fax: +1-979-845-6259.
E-mail addresses: ubraga@mdanderson.org (U. Braga-Neto), e-dougherty@tamu.edu (E. Dougherty).

0031-3203/$30.00 © 2003 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2003.08.017

in performance can be striking, especially with small samples.

The basic idea is to "bolster" the original empirical distribution of the available data by means of suitable bolstering kernels placed at each data-point location. The error can be computed analytically in some cases, such as in the case of linear classifiers, or via Monte-Carlo sampling for more general classifier boundaries; our experiments show that a very small number of Monte-Carlo samples is needed, which results in a fast error estimator. In principle, the technique can be applied in conjunction with any error-counting estimation method, and has the effect of reducing the variance usually associated with such methods, which is especially important in the case of the huge variance associated with cross-validation error estimation. By selecting the parameters of the bolstering kernels appropriately, one can also reduce bias. For simplicity, we focus on two-group classification; however, the proposed method is easily extendable to multiple-group classification.

The proposed error estimation method shares some similarities with the "smoothing" technique for LDA originally proposed in [9] and further explored in [10–13]. In that case, the basic idea is to reduce the variance of error-counting estimators by means of a smoothing function that plays a similar role to the bolstering kernels. Bolstered error estimation can be seen as smoothed estimation in a few special cases; however, the smoothing technique does not have a direct geometrical interpretation in general, can be cumbersome to formulate for multiple classes [10,13], and is not easily extendable to classifiers other than LDA [10,4]. For general classifiers, one has to use some estimate of the posterior probabilities in the computation of the smoothed error estimate. Not only is this problematic in small-sample settings, but it can also lead to bad situations not encountered by bolstered error estimation, such as an identically zero smoothed error estimator for 1-nearest-neighbor classification [4].

We provide an extensive simulation study that compares the proposed method to resubstitution, leave-one-out, 10-fold cross validation with repetition, and the 0.632 bootstrap estimator. These are well-known error estimators often used in practice. We consider three popular classification rules, namely, LDA [14], 3-nearest neighbors (3NN) [4] and decision trees (CART) [5]. The simulation study compares the performance of the error estimators in terms of the deviation distribution (the distributions of the difference between estimated and true errors). Average timings are also computed to assess the speed of the various error estimators.

This paper is organized as follows. Section 2 provides a brief review of error estimators considered in this paper, including smoothed error estimators. Section 3 introduces bolstered error estimation, proposes a general method to choose the amount of bolstering, and discusses a few particular estimators based on Gaussian bolstering kernels. Section 4 presents the results obtained in our simulation study. Finally, Section 5 provides concluding remarks.

2. Error estimation methods

In two-group statistical pattern recognition, there is a *feature vector* $X \in \mathbb{R}^p$ and a *label* $Y \in \{0,1\}$. The pair (X,Y) has a joint probability distribution $\mathbf{F}$, which is unknown in practice. Hence, one has to resort to designing classifiers from *training data*, which consists of a set of n independent observations, $S_n = \{(X_1,Y_1),\ldots,(X_n,Y_n)\}$, drawn from $\mathbf{F}$. A *classification rule* is a mapping $g : \{\mathbb{R}^p \times \{0,1\}\}^n \times \mathbb{R}^p \to \{0,1\}$. A classification rule maps the training data S_n into the *designed classifier* $g(S_n,\cdot) : \mathbb{R}^p \to \{0,1\}$. The *true error* of a designed classifier is its error rate given the training data set:

$$\varepsilon_n[g|S_n] = P(g(S_n,X) \neq Y) = E_{\mathbf{F}}(|Y - g(S_n,X)|), \tag{1}$$

where the notation $E_{\mathbf{F}}$ indicates that the expectation is taken with respect to $\mathbf{F}$; in fact, one can think of (X,Y) in the above equation as a random test point (this interpretation being useful in understanding error estimation). The expected error rate over the data is given by

$$\varepsilon_n[g] = E_{\mathbf{F}_n}(\varepsilon_n[g|S_n]) = E_{\mathbf{F}_n}E_{\mathbf{F}}(|Y - g(S_n,X)|), \tag{2}$$

where $\mathbf{F}_n$ is the joint distribution of the training data S_n. This is sometimes called the *unconditional error* of the classification rule, for sample size n.

Were the underlying feature-label distribution $\mathbf{F}$ known, the true error could be computed exactly, via (1). In practice, one is limited to using an *error estimator*. Ideally, this estimate should be fast to compute and as close as possible to the true error, for the given training data. Most error estimators used in practice implement some form of sample-mean-like approximation using test points. The error estimator is unbiased, with respect to the unconditional error, if the test points come from independent samples not used to design the classifier.

2.1. Resubstitution

The simplest and fastest way to estimate the error of a designed classifier in the absence of test data is to compute its error directly on the sample data itself:

$$\hat{\varepsilon}_{\text{resub}} = \frac{1}{n}\sum_{i=1}^{n} |y_i - g(S_n,x_i)|. \tag{3}$$

This *resubstitution estimator*, attributed to Ref. [2], is very fast, but is usually optimistic (i.e., low-biased) as an estimator of $\varepsilon_n[g]$ [4]. For some classification rules, resubstitution can be severely low-biased, an extreme case being one-nearest-neighbor classification, in which the resubstitution estimator is identically equal to zero. Typically, the more complex is the classifier, the more optimistic is resubstitution, since complex classifiers tend to overfit the data, especially with small samples [15].

2.2. Cross-validation

Cross-validation removes the optimism from resubstitution by employing test points not used in classifier design [5]. In *k-fold cross-validation*, the data set S_n is partitioned into k *folds* $S_{(i)}$, for $i = 1,\ldots,k$ (for simplicity, we assume that k divides n). Each fold is left out of the design process and used as a test set, and the estimate is the overall proportion of error committed on all folds:

$$\hat{\varepsilon}_{\mathrm{cvk}} = \frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{n/k} |y_j^{(i)} - g(S_n \backslash S_{(i)}, x_j^{(i)})|, \tag{4}$$

where $(x_j^{(i)}, y_j^{(i)})$ is a sample in the ith fold. The process may be repeated: several cross-validation estimates are computed using different partitions of the data into folds, and the results are averaged. A k-fold cross-validation estimator is unbiased as an estimator of $\varepsilon_{n-n/k}[g]$. The most well-known cross-validation method, usually attributed to Ref. [3], is the *leave-one-out estimator*, whereby a single observation is left out each time:

$$\hat{\varepsilon}_{\mathrm{loo}} = \frac{1}{n}\sum_{i=1}^{n} |y_i - g(S_{n-1}^i, x_i)|, \tag{5}$$

where S_{n-1}^i is the data set resulting from deleting data point i from the original data set S_n. This corresponds to n-fold cross-validation. The leave-one-out estimator is unbiased as an estimator of $\varepsilon_{n-1}[g]$. Cross-validation estimators are often pessimistic, since they use smaller training sets to design the classifier. Their main drawback is their variance [16,4]. They can also be quite slow to compute when the number of folds or samples is large.

2.3. Bootstrap

The bootstrap error estimation technique [6,7] is based on the notion of an "empirical distribution" $\mathbf{F}^*$, which serves as a replacement to the original unknown distribution $\mathbf{F}$. The empirical distribution puts mass $1/n$ on each of the n available data points. A "bootstrap sample" S_n^* from $\mathbf{F}^*$ consists of n equally-likely draws with replacement from the original data S_n. Hence, some of the samples will appear multiple times, whereas others will not appear at all. The actual proportion of times a data point (x_i, y_i) appears in S_n^* can be written as $P_i^* = 1/n\sum_{j=1}^{n} I_{(x_j^*, y_j^*)=(x_i,y_i)}$, where $I_S = 1$ if the statement S is true, zero otherwise. The basic *bootstrap zero estimator* [17] is written in terms of the empirical distribution as $\hat{\varepsilon}_0 = E_{\mathbf{F}^*}(|Y - g(S_n^*, X)| : (X, Y) \in S_n \setminus S_n^*)$. In practice, the expectation $E_{\mathbf{F}^*}$ has to be approximated by a Monte-Carlo estimate based on independent replicates S_n^{*b}, for $b = 1,\ldots,B$ (B between 25 and 200 being recommended [17]):

$$\hat{\varepsilon}_0 = \frac{\sum_{b=1}^{B}\sum_{i=1}^{n} |y_i - g(S_n^{*b}, x_i)|\, I_{P_i^{*b}=0}}{\sum_{b=1}^{B}\sum_{i=1}^{n} I_{P_i^{*b}=0}}. \tag{6}$$

The bootstrap zero estimator works like cross-validation: the classifier is designed on the bootstrap sample and tested on the original data points that are left out. It tends to be high-biased as an estimator of $\varepsilon_n[g]$, since the amount of samples available for designing the classifier is on average only $(1 - e^{-1})n \approx 0.632n$. The estimator

$$\hat{\varepsilon}_{\mathrm{b632}} = (1 - 0.632)\hat{\varepsilon}_{\mathrm{resub}} + 0.632\hat{\varepsilon}_0 \tag{7}$$

tries to correct this bias by doing a weighted average of the bootstrap zero and resubstitution estimators. It is known as the *0.632 bootstrap estimator* [17], and has been perhaps the most popular bootstrap estimator in data mining [18]. It has low variance, but can be extremely slow to compute. In addition, it can fail when resubstitution is too low biased [16].

2.4. Smoothed estimation

The resubstitution estimator can be rewritten as

$$\hat{\varepsilon}_{\mathrm{resub}} = \frac{1}{n}\sum_{i=1}^{n} (g(S_n, x_i)I_{y_i=0} + (1 - g(S_n, x_i))I_{y_i=1}). \tag{8}$$

The function g is a sharp 0-1 step function that can introduce variance by the fact that a point near the decision boundary can change its contribution from 0 to $1/n$ (and vice versa) via a slight change in the training data, even if the corresponding change in the decision boundary is small, and hence so is the change in the true error. In small-sample settings, $1/n$ can be large.

The idea behind *smoothed estimators* [9] is to replace function g in (8) by a suitably chosen "smooth" function taking values in the interval $[0, 1]$, thereby reducing the variance of the original estimator. In Refs. [9,11–13], this idea is applied to LDA classification, which is essentially a one-dimensional problem, since the classifier can be written as

$$g(S_n, x) = \begin{cases} 1 & \text{if } W(x) > 0, \\ 0 & \text{if } W(x) \leqslant 0, \end{cases}$$

where $W : \mathbb{R}^p \to \mathbb{R}$ is Anderson's W statistic [19] (for simplicity, our notation omits the dependence of W on S_n). The W statistic is given by $W(x) = a^T x + m$, where

$$a = \Sigma^{-1}(\mu_1 - \mu_0),$$

$$m = \frac{1}{2}(\mu_0 + \mu_1)^{\mathrm{T}}\Sigma^{-1}(\mu_0 - \mu_1).$$

Here, $\Sigma = \frac{1}{2}(\Sigma_0 + \Sigma_1)$ is the pooled covariance matrix, with μ_i and Σ_i denoting the mean and covariance matrix for class i, respectively, which are obtained via their usual maximum-likelihood estimates. The parameters a and m specify the separating hyperplane produced by LDA: a is a vector normal to the hyperplane, and $m/\|a\|$ is its distance to the origin.

While the sign of the W statistic gives the decision region to which a point belongs, its magnitude measures robustness of that decision. In fact, the *signed* Euclidean distance from a point x to the separating hyperplane is $W_a = W/\|a\|$.

We refer to W_a as the *normalized W statistic*. In addition to measuring robustness of the classification, W is not a step function of the data, but varies in a linear fashion. To achieve smoothing of the error count, the idea then is to use a monotone increasing function $r:\mathbb{R}\to[0,1]$ applied on W in place of the function g in (8). The function r should be such that $r(-u)=1-r(u)$, $\lim_{u\to-\infty}r(u)=0$, and $\lim_{u\to\infty}r(u)=1$. The *smoothed resubstitution* estimator is given by

$$\hat{\varepsilon}^{s}_{\text{resub}}=\frac{1}{n}\sum_{i=1}^{n}(r(W(x_i))I_{y_i=0}+(1-r(W(x_i)))I_{y_i=1}). \quad (9)$$

For instance, one may use the Gaussian function $r(u)=\Phi_\sigma(u)$, where Φ_σ is the cumulative distribution function of a zero-mean Gaussian random variable with variance σ^2. Another example is the windowed linear function $r(u)=0$ on $(-\infty,-b)$, $r(u)=1$ on (b,∞), and $r(u)=(u+b)/2b$ on $[-b,b]$. Generally, a choice of function r depends on tunable parameters, such as σ and b in the previous examples. The choice of parameter is a major issue, which affects the variance and bias of the resulting estimator. A few approaches have been tried, namely, arbitrary choice [9,10], arbitrary function of the separation between classes [10], parametric estimation assuming normal populations [11,12], and simulation-based methods [13]. We will return to this point when we discuss a similar issue in Section 3.

Note that the idea of smoothing can be applied to other error-counting estimators, such as leave-one-out.

Extension of smoothing to classification rules other than LDA is not straightforward, since a suitable replacement to the W statistic must be found, and that is not generally available. This problem has received little attention in the literature. In Ref. [4], and under a different but equivalent guise in [10], it is suggested that one use an estimator η of the posterior probability $P(Y=1|X)$ in such a way that the classification rule can be written as

$$g(S_n,x)=\begin{cases}1 & \text{if } \eta(x)>\frac{1}{2},\\ 0 & \text{if } \eta(x)\leqslant\frac{1}{2}.\end{cases} \quad (10)$$

For example, for a kNN classifier, an estimator of the posterior probability is given by

$$\eta(x)=\frac{1}{k}\sum_{i=1}^{n}y_{(i)}, \quad (11)$$

where $y_{(i)}$ denotes the label of the ith closest observation to the point x. A monotone increasing function $r:[0,1]\to[0,1]$, such that $r(u-1/2)=1-r(u+1/2)$, $r(0)=0$, and $r(1)=1$, can then be applied to η, and the smoothed resubstitution estimator is given as before by (9), with W replaced by η. However, this approach has problems. It is not clear how to choose the estimator η, and it will not have in general an easy geometric interpretation. In [10], it is suggested that one estimate the actual class-conditional probabilities from the data to arrive at η, but this method has serious issues in small-sample settings. To illustrate these difficulties, consider the case of 1NN classification, for which smoothed resubstitution completely fails with the choice of η in (11), as the estimator is zero with probability one [4].

3. Bolstered error estimation

The feature-label empirical distribution $\mathbf{F}^*$ considered in connection with bootstrap error estimation is a distribution for the pair (X,Y) given by the probability mass function $P(X=x_i,Y=y_i)=1/n$, for $i=1,\ldots,n$. It is easy to see that the resubstitution estimator is given by

$$\hat{\varepsilon}_{\text{resub}}=E_{\mathbf{F}^*}(|Y-g(S_n,X)|). \quad (12)$$

The empirical distribution $\mathbf{F}^*$ is confined to the original data points, so that no distinction is made between points near or far from the decision boundary. If one spreads out the probability mass put on each point by the empirical distribution, variation is reduced in (12) because points near the decision boundary will have more mass go to the other side than will points far from the decision boundary. Another way of looking at this is that more confidence is attributed to points far from the decision boundary than points near it. Consider a p-variate probability density function $f_i^{\diamondsuit}$, for each $i=1,\ldots,n$, which we call a *bolstering kernel*. We propose the *bolstered empirical distribution* $\mathbf{F}^{\diamondsuit}$, with probability density function $f^{\diamondsuit}$ given by

$$f^{\diamondsuit}(x,y)=\frac{1}{n}\sum_{i=1}^{n}f_i^{\diamondsuit}(x-x_i)I_{y=y_i}. \quad (13)$$

This is similar to a Parzen-window probability density estimate [5]. However, our sole purpose is error estimation; Eq. (13) is not an attempt to estimate the true feature-label distribution (this would prove futile in small-sample settings, in which lies our main interest).

The *bolstered resubstitution* estimator is obtained by replacing $\mathbf{F}^*$ by $\mathbf{F}^{\diamondsuit}$ in (12):

$$\hat{\varepsilon}^{\diamondsuit}_{\text{resub}}=E_{\mathbf{F}^{\diamondsuit}}(|Y-g(S_n,X)|). \quad (14)$$

The following result gives an alternative computational expression.

Proposition 3.1. *Let $A_j=\{x\in\mathbb{R}^p|g(S_n,x)=j\}$, for $j=0,1$, be the decision regions for the designed classifier. We have that*

$$\hat{\varepsilon}^{\diamondsuit}_{\text{resub}}=\frac{1}{n}\sum_{i=1}^{n}\left(\int_{A_1}f_i^{\diamondsuit}(x-x_i)\,\mathrm{d}x I_{y_i=0}+\int_{A_0}f_i^{\diamondsuit}(x-x_i)\,\mathrm{d}x I_{y_i=1}\right). \quad (15)$$

Proof. From (14), we have that

$$\hat{\varepsilon}^{\diamondsuit}_{\text{resub}}=\int|y-g(S_n,x)|\,\mathrm{d}\mathbf{F}^{\diamondsuit}(x,y)$$

$$=\sum_{y=0}^{1}\int_{\mathbb{R}^p}|y-g(S_n,x)|f^{\diamondsuit}(x,y)\,\mathrm{d}x$$

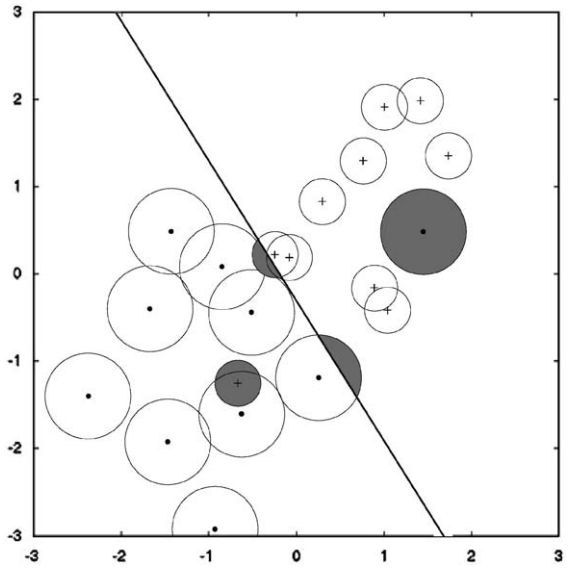

Fig. 1. Bolstered resubstitution for LDA, assuming uniform circular bolstering kernels. The area of each shaded region divided by the area of the associated circle is the error contribution made by a point. The bolstered resubstitution error is the sum of all contributions divided by the number of points (the data set in this example is part of the simulation study in Section 4).

$$= \frac{1}{n}\sum_{y=0}^{1}\sum_{i=1}^{n}\int_{\mathbb{R}^p} |y - g(S_n,x)| f_i^{\diamondsuit}(x - x_i) I_{y=y_i}\,\mathrm{d}x$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left(\int_{\mathbb{R}^p} g(S_n,x)\, f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x\, I_{y_i=0} + \int_{\mathbb{R}^p} (1 - g(S_n,x))\, f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x\, I_{y_i=1}\right).$$

But $g(S_n,x)$ is zero over A_0 and is 1 over A_1, from which (15) follows. □

Eq. (15) extends a similar expression that was proposed in Ref. [8] in the context of LDA. The integrals in (15) are the error contributions made by the data points, according to whether $y_i = 0$ or 1. The bolstered resubstitution is equal to the sum of all error contributions divided by the number of points. See Fig. 1 for an illustration, where the bolstering kernels are given by uniform circular distributions.

When the classifier is linear (the decision boundary is a hyperplane), then it is usually possible to find analytical expressions for the integrals in (15); we present examples of this in conjunction with LDA later. Otherwise, one has to apply Monte-Carlo integration:

$$\hat{\varepsilon}^{\diamondsuit}_{\mathrm{resub}} \approx \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{M} I_{x_{ij}\in A_1} I_{y_i=0} + \sum_{j=1}^{M} I_{x_{ij}\in A_0} I_{y_i=1}\right), \quad (16)$$

where $\{x_{ij}\}_{j=1,\ldots,M}$ are samples drawn from the distribution $f_i^{\diamondsuit}$. The experiments in Section 4 indicate that a small number M of Monte-Carlo samples is needed (in our simulations, a value $M = 10$ was adequate, and increasing M beyond that did not substantially reduce the variance of the estimator).

Bolstering can be applied to any error-counting error estimation method. For example, consider leave-one-out estimation. Recall that S_{n-1}^i denotes the data set resulting from deleting data point i from the original data set S_n. Let $A_j^i = \{x \in \mathbb{R}^p | g(S_{n-1}^i, x) = j\}$, for $j = 0, 1$, be the decision regions for the classifier designed from S_{n-1}^i. The *bolstered leave-one-out* estimator can be computed via

$$\hat{\varepsilon}^{\diamondsuit}_{\mathrm{loo}} = \frac{1}{n}\sum_{i=1}^{n}\left(\int_{A_1^i} f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x I_{y_i=0} + \int_{A_0^i} f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x I_{y_i=1}\right). \quad (17)$$

When the integrals cannot be computed exactly, a Monte-Carlo expression similar to (16) can be employed instead.

3.1. Choosing the amount of bolstering

Although more general bolstering kernels may be considered, in keeping with the principle of not making complicated inferences from a limited amount of data, in this paper we only consider zero-mean, spherical bolstering kernels $f_i^{\diamondsuit}$, with covariance matrices of the form $\sigma_i^2 I_p$. In each case there is a family of bolstered estimators, corresponding to the choices of the standard deviations $\sigma_1, \ldots, \sigma_n$. The choice of these parameters determines the variance and bias properties of the corresponding bolstered estimator. If $\sigma_i = 0$, for $i = 1, \ldots, n$, then there is no bolstering and the bolstered estimator reduces to the original estimator. As a general rule, larger σ_i's, i.e., "wider" bolstering kernels, lead to lower-variance estimators, but after a certain point this advantage becomes offset by increasing bias.

The choice of the standard deviations is a critical issue. We have mentioned in Section 2.4 that several approaches have been attempted to solve a similar problem in smoothed error estimation. In Ref. [8], a simulation-based approach is employed to find a single value $\sigma_i = \sigma$ for bolstered resubstitution. We propose here a simple non-parametric sample-based method to choose these parameters that is applicable in small-sample settings. This method is partly inspired by the distance argument used in Ref. [17], in connection with the 0.632 bootstrap estimator.

When bolstering resubstitution, the aim is to select the parameters so that the bolstered resubstitution estimator is nearly unbiased. As mentioned previously, one can think of (X, Y) in (1) or (2) as a random test point. Given that $Y = y$, this test point is at a "true mean distance" $\delta(y)$ from the data points belonging to class y. This distance is determined by the underlying class-conditional distribution $F(X|Y = y)$.

One reason why plain resubstitution is optimistically biased is that the test points in (3) are all at distance zero from the training data. Since bolstered estimators spread the test points, the task is to find the amount of spreading that makes the test points to be as close as possible to the true mean distance to the training data points.

The true mean distance can be estimated by its sample-based estimate:

$$\hat{d}(y) = \frac{\sum_{i=1}^{n} \min_{j \neq i} \{\|x_i - x_j\|\} I_{y_i = y}}{\sum_{i=1}^{n} I_{y_i = y}}. \tag{18}$$

The estimate $\hat{d}(y)$ is the mean minimum distance between points belonging to class y.

Let $f_i^{\diamondsuit,1}$ be a unit-variance bolstering kernel, and let D_i be the random variable equal to the distance of a point randomly selected from $f_i^{\diamondsuit,1}$ to the origin. Let $F_{D_i}(x)$ be the cdf of D_i. In the case of the bolstering kernel $f_i^{\diamondsuit}$ with variance $\sigma_i^2 I_p$, all distances get multiplied by σ_i. We propose to find the value of σ_y for class y, such that the median distance of a test point to the origin is equal to the estimated true mean distance $\hat{d}(y)$, so that half of the test points will be farther from the center than $\hat{d}(y)$, and the other half will be nearer. Hence, σ_y is the solution of the equation $\sigma_y F_{D_i}^{-1}(1/2) = \hat{d}(y)$. Note that

$$\alpha_{p,i} = F_{D_i}^{-1}(1/2) \tag{19}$$

can be viewed as a constant "correction" factor, which can be computed and stored off-line. The subscript p indicates explicitly that the correction factor is a function of the dimensionality. The estimated standard deviations for the bolstering kernels are thus given by

$$\sigma_i = \frac{\hat{d}(y_i)}{\alpha_{p,i}} \quad \text{for } i = 1, \ldots, n.$$

Clearly, as the number of samples in the training data increases, the standard deviations σ_i decrease, and there is less bias correction introduced by the bolstered resubstitution. This is in accordance with the fact that resubstitution tends to be less optimistically biased as the sample size increases.

In situations where resubstitution is heavily low-biased due to overfitting classification rules, it may not be a good idea to spread incorrectly classified data points because that increases optimism of the error estimate, i.e., low-biasedness. Bias is reduced if one assigns $\sigma_i = 0$ (no bolstering) to incorrectly classified points. This increases variance because there is less bolstering. We call this variation a *semi-bolstered resubstitution* estimator. We have found empirically (see Section 4) that for 3NN and CART, which are overfitting rules, especially with very small samples, the bias-variance trade-off (as measured by the RMS) may be favorable to semi-bolstered resubstitution.

Let us consider now the leave-one-out estimator. In this case, no bias-correction is necessary or desired; the aim is solely reducing the variance of the estimator. Considering the distance argument, we see that each point left out in the design of the classifier g is an independent sample and is already at the right distance to the design data set (this is the reason for the unbiasedness of leave-one-out as estimator of $\varepsilon_{n-1}[g]$). Therefore, we propose to use the minimum distance $d(x_i, S_{n-1}^i)$ of each point to the rest of the data set as the basis for selecting the standard deviation of the corresponding bolstering kernel $f_i^{\diamondsuit}$. As before, we want half of the test points to be farther from the center than $d(x_i, S_{n-1}^i)$, and the other half to be nearer. Therefore, the standard deviations are distinct for each data point, and given by

$$\sigma_i = \frac{d(x_i, S_{n-1}^i)}{\alpha_{p,i}} \quad \text{for } i = 1, \ldots, n,$$

where $\alpha_{p,i}$ is the correction factor in (19).

3.2. Gaussian-bolstered error estimation

In this paper we consider bolstering kernels that are spherical p-variate normal distributions:

$$f_i^{\diamondsuit}(x) = \frac{1}{(2\pi)^{p/2}\sigma_i^{\,p}} \exp\left(-\frac{\|x\|^2}{2\sigma_i^2}\right).$$

For a general classifier, the integrals in (15) and (17) have to be computed by Monte-Carlo sampling. For a linear classifier, however, analytical expressions are possible. The following result illustrates this for the case of resubstitution and LDA.

Proposition 3.2. *For LDA, the Gaussian-bolstered resubstitution error estimator is given by*

$$\hat{\varepsilon}_{\text{resub}}^{\diamondsuit} = \frac{1}{n} \sum_{i=1}^{n} (\Phi_{\sigma_i}(W_a(x_i)) I_{y_i=0} + \Phi_{\sigma_i}(-W_a(x_i)) I_{y_i=1}), \tag{20}$$

where Φ_{σ_i} is the cumulative distribution function of a zero-mean Gaussian random variable with variance σ_i^2, and W_a is the normalized W statistic.

Proof. Suppose that $x_i \in A_0$. By exploiting the symmetry of the problem, we may assume, without loss of generality, the geometry depicted in Fig. 2, where x_i is the origin. We have that

$$\begin{aligned}
&\int_{A_0} f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x \\
&\quad = \int_{A_0} f_i^{\diamondsuit}(x)\,\mathrm{d}x = \int_{A_0} \frac{1}{(2\pi)^{p/2}\sigma_i^{\,p}} \exp\left(-\frac{\|x\|^2}{2\sigma_i^2}\right)\mathrm{d}x \\
&\quad = \int_{-\infty}^{h} \frac{1}{(2\pi)^{1/2}\sigma_i} \exp\left(-\frac{x_{(1)}^2}{2\sigma_i^2}\right)\mathrm{d}x_{(1)} \\
&\quad \times \underbrace{\int_{-\infty}^{\infty} (1/(2\pi)^{1/2}\sigma_i) \exp(-x_{(2)}^2/2\sigma_i^2)\,\mathrm{d}x_{(2)} \cdots}_{1}
\end{aligned}$$

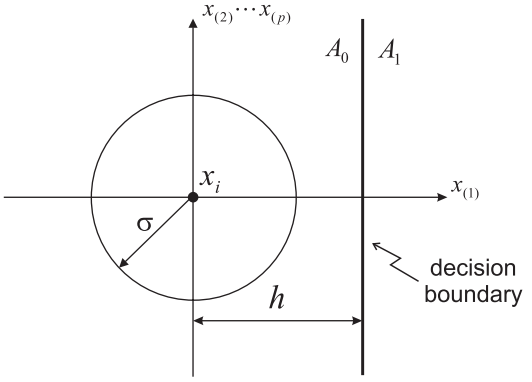

Fig. 2. Diagram for calculation of $\int_{A_0} f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x$ (see proof of Proposition 3.2).

$$\times \underbrace{\int_{-\infty}^{\infty} (1/(2\pi)^{1/2}\sigma_i) \exp(-x_{(p)}^2/2\sigma_i^2)\,\mathrm{d}x_{(p)}}_{1}$$

$$= \Phi_{\sigma_i}(h).$$

Clearly, $\int_{A_1} f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x = 1 - \Phi_{\sigma_i}(h) = \Phi_{\sigma_i}(-h)$. If $x \in A_1$ instead, then the signs of h are interchanged. Now, the normalized W statistic at point x_i is given by $W_a(x_i) = (-1)^{(j+1)}h$ for $x_i \in A_j$, $j = 0, 1$. It follows that $\int_{A_j} f_i^{\diamondsuit}(x - x_i)\,\mathrm{d}x = \Phi_{\sigma_i}((-1)^{(j+1)}W_a(x_i))$, for $x_i \in \mathbb{R}^p$, $j = 0, 1$. By replacing this into (15), one obtains (20). □

A similar expression to (20) applies to the Gaussian-bolstered leave-one-out.

Note that $\Phi_\sigma(0) = 1/2$, which corresponds to the error contribution of a point on the decision boundary. As $\sigma_i \to 0$, for $i = 1, \ldots, n$, then all functions Φ_{σ_i} collapse to indicator step functions and the Gaussian-bolstered error estimator reduces to the original estimator. On the other hand, if $\sigma_i \to \infty$, for $i = 1, \ldots, n$, then the functions Φ_{σ_i} become constant and equal to $\frac{1}{2}$, so that the bolstered estimator is identically equal to $\frac{1}{2}$, regardless of the data. This estimator has zero variance, but is of course not useful.

For LDA and in the case where $\sigma_i = \sigma$, for $i = 1, \ldots, n$, the Gaussian-bolstered resubstitution estimator reduces to the smoothed estimator in (9), with $r(u) = \Phi_\sigma(u/\|a\|)$.

The actual values of σ_i in a practical situation are computed according to the distance-based scheme outlined in the previous subsection. In the present Gaussian case, the distance variables D_i are distributed as a *chi* random variable D with p degrees of freedom. The density function of D is given by [20]:

$$f_D(x) = \frac{2^{1-p/2}x^{p-1}\mathrm{e}^{-x^2/2}}{\Gamma(p/2)}, \tag{21}$$

where Γ is the gamma function. For $p = 2$, this becomes the well-known Rayleigh density. The cdf F_D can be computed by numerical integration of (21), and the inverse at point 1/2 can be found by a simple binary search procedure (using the fact that F_D is monotonically increasing), which yields the correction factor α_p. For instance, the values of the correction factor up to five dimensions are: $\alpha_1 = 0.674$, $\alpha_2 = 1.177$, $\alpha_3 = 1.538$, $\alpha_4 = 1.832$, $\alpha_5 = 2.086$.

4. Experimental results

In this section, we report results obtained from a large simulation study based on synthetic data, which measures the performance of resubstitution (resub), leave-one-out (loo), 10-fold cross-validation with 10 repetitions (cv10r) and the 0.632 bootstrap (b632) against a few Gaussian-bolstered error estimators, namely, bolstered resubstitution (bresub), semi-bolstered resubstitution (sresub), and bolstered leave-one-out (bloo), for three popular classification rules, LDA, 3NN, and CART. For the computation of cv10r, we use *stratified* cross-validation, whereby the classes are represented in each fold in the same proportion as in the original data (there is evidence that this improves the estimator [18]). For computation of the bootstrap estimator, we use a variance-reducing technique called *balanced* bootstrap resampling [21], where each sample is made to appear exactly B times in the computation (e.g., if it appears twice in one bootstrap sample, it has to be absent in some other bootstrap sample). The number of bootstrap samples is $B = 100$, which makes the number of designed classifiers be the same as for cv10r. For LDA, the bolstered estimators are computed using the analytical formulas developed in Section 3; for 3NN and CART, Monte-Carlo sampling is used. We have found that only $M = 10$ Monte-Carlo samples per bolstering kernel is adequate, and increasing M to a much larger value, $M = 200$, reduces the variance of the estimators only slightly. To improve the performance and minimize overfit in CART, the tree is not fully grown, but splitting stops when there are six points or fewer in a node. The simulations were performed on a 2.5 GHz Pentium 4 computer, running Windows 2000 and Cygwin (a UNIX environment for Windows). The C code developed to implement all the error estimators and classification rules, with full documentation and examples, can be downloaded from the companion website.

Our experiments assess the empirical distribution of $\varepsilon_n[g|S_n] - \hat{\varepsilon}$, for each error estimator $\hat{\varepsilon}$. This *deviation distribution* (see also Ref. [16]) indicates how far from the actual error value the estimated value is; in our study, it is derived from 1000 independent training data sets drawn from several models. The use of synthetic data is central to the analysis because it allows us to the compute true errors needed to construct the deviation distributions. We also present average timings for computation of the several error estimators considered in the simulation.

We consider here a catalog of 72 experimental conditions, each involving a thousand replications using different sample training data drawn from an underlying

Table 1
Twelve basic experiments used in the simulation study

Exp.	Rule	p	δ	σ_1	σ_2	Bayes err.
1	LDA	2	0.59	1.00	1.00	0.202
2	LDA	2	0.59	1.00	4.00	0.103
3	LDA	5	0.37	1.00	1.00	0.204
4	LDA	5	0.37	1.00	2.16	0.103
5	3NN	2	1.20	1.00	1.00	0.204
6	3NN	2	1.20	1.00	5.20	0.103
7	3NN	5	0.77	1.00	1.00	0.204
8	3NN	5	0.77	1.00	2.35	0.105
9	CART	2	1.20	1.00	1.00	0.204
10	CART	2	1.20	1.00	5.20	0.103
11	CART	5	0.77	1.00	1.00	0.204
12	CART	5	0.77	1.00	2.35	0.105

model. The model assumed for LDA consists of Gaussian class-conditional densities, with spherical covariances and means located at $(\delta,\ldots,\delta)$ and $(-\delta,\ldots,-\delta)$, where $\delta > 0$ is a separation parameter that controls the Bayes error. The model used for 3NN and CART corresponds to class-conditional densities given by a mixture of Gaussians, with spherical covariances and means at opposing vertices of a hypercube centered at the origin and side 2δ; e.g., in five dimensions, the class-conditional density for class 1 has means at $(\delta,\delta,\delta,\delta,\delta)$ and $(-\delta,-\delta,-\delta,\delta,-\delta)$, whereas the class-conditional density for class 2 has means at $(\delta,-\delta,\delta,-\delta,\delta)$ and $(-\delta,\delta,-\delta,\delta,-\delta)$. In all cases, we assume equal prior probabilities for each class.

We consider 12 experiments and six sample sizes, varying from 20 to 120 in increments of 20, which make up the total of 72 experimental conditions. The twelve experiments correspond to a choice among the three classification rules, using low or moderate dimensionality, and equal or distinct covariance matrices; see Table 1. Here, p is the dimensionality, and the separation parameter δ and the standard deviations σ_1 and σ_2 are adjusted so that the optimal Bayes error is about 0.1 in half of the cases, and about 0.2 in the other half. We point out that these are difficult models, with considerable overlapping between the classes (even in the cases where the Bayes error is 0.1, owing to a large discrepancy in variance between the classes, not linear separation). Due to space constraints, we present here a selection of results covering representative experiments and sample sizes. Please see the companion website for the full results of the complete set of experiments.

Tables 2 and 3 display the statistics of the empirical distribution of $\varepsilon_n[g|S_n] - \hat{\varepsilon}$, for each error estimator $\hat{\varepsilon}$, derived from the 1000 independent draws of the observations S_n, for three representative experiments and sample sizes $n = 20$ and $n = 80$. Also displayed are the sample mean and sample standard deviation of the true error $\varepsilon_n[g|S_n]$, which is obtained exactly in the LDA case, and by Monte-Carlo computation in the 3NN and CART cases.

Figs. 3 and 4 display the box plots of the empirical deviation distribution, for $n = 20$ and $n = 80$, respectively. For each column (estimator), the box has its bottom at the lower quartile and the top at the upper quartile. The location of the median is indicated by a line across the box. There are also dashed lines ("whiskers") extending from each end of the box to show the extent of the rest of the data. The ends of the whiskers on each side of the box lie at 1.5 times the interquartile range. Crosses mark the position of outliers, i.e., data with values beyond the ends of the whiskers.

Figs. 5 and 6 display plots of the empirical deviation distribution, for $n = 20$ and 80, respectively, obtained by fitting a beta density to the raw data. For clarity of presentation, we leave out the cross-validation estimators (the complete plots, using color to facilitate visualization, can be accessed on the companion website).

The simulation results lead to several conclusions. Resubstitution, leave-one-out, and 10-fold cross-validation with repetition are as a general rule outperformed by the 0.632 bootstrap and the bolstered estimators. The latter are very competitive with the bootstrap. For LDA, the best estimator overall is without question the bolstered resubstitution. For 3NN and CART, classifiers known to overfit in small-sample settings (note that $n = 20$ with $p = 5$ is an extreme small-sample situation), the situation is not so clear. For 3NN, we can see that bolstered resubstitution fails in correcting the bias of resubstitution for $p = 5$, despite having small variance (however, it is still the best overall estimator for 3NN in Experiment 5). In that case, the semi-bolstered resubstitution is a very competitive alternative. For CART, the bootstrap and bolstered resubstitution estimators are affected by the extreme low-biasedness of resubstitution. In this case, bolstered resubstitution performs better than in the 3NN case, but the best overall estimator is the semi-bolstered

Table 2
Statistics of empirical deviation distribution for three representative experiments, in the case $n = 20$

	Resub	Loo	cv10r	b632	Bresub	Sresub	Bloo
Experiment 1							
LDA, $p = 2$, $\delta = 0.59$, $\sigma_1 = 1.00$, $\sigma_2 = 1.00$							
Mean$[\varepsilon_n[g\|S_n]] = 0.224$, Var$[\varepsilon_n[g\|S_n]] = 0.001$							
Mean	−0.046	0.001	0.000	−0.002	−0.008	0.036	0.025
Variance	0.008	0.010	0.010	0.008	0.005	0.008	0.008
RMS	0.101	0.101	0.98	0.092	0.074	0.098	0.090
Experiment 7							
3NN, $p = 5$, $\delta = 0.77$, $\sigma_1 = 1.00$, $\sigma_2 = 1.00$							
Mean$[\varepsilon_n[g\|S_n]] = 0.331$, Var$[\varepsilon_n[g\|S_n]] = 0.002$							
Mean	−0.156	0.070	0.035	0.013	−0.083	−0.004	0.105
Variance	0.007	0.016	0.013	0.005	0.003	0.006	0.007
RMS	0.176	0.145	0.120	0.072	0.099	0.080	0.134
Experiment 12							
CART, $p = 5$, $\delta = 0.77$, $\sigma_1 = 1.00$, $\sigma_2 = 2.35$							
Mean$[\varepsilon_n[g\|S_n]] = 0.373$, Var$[\varepsilon_n[g\|S_n]] = 0.003$							
Mean	−0.321	0.042	0.025	−0.069	−0.079	−0.067	0.036
Variance	0.003	0.026	0.018	0.005	0.003	0.004	0.009
RMS	0.325	0.168	0.138	0.099	0.098	0.090	0.102

Table 3
Statistics of empirical deviation distribution for three representative experiments, in the case $n = 80$

	Resub	Loo	cv10r	b632	Bresub	Sresub	Bloo
Experiment 1							
LDA, $p = 2$, $\delta = 0.59$, $\sigma_1 = 1.00$, $\sigma_2 = 1.00$							
Mean$[\varepsilon_n[g\|S_n]] = 0.207$, Var$[\varepsilon_n[g\|S_n]] = 0.000$							
Mean	−0.010	−0.000	0.001	−0.001	0.000	0.029	0.006
Variance	0.002	0.002	0.002	0.002	0.001	0.002	0.002
RMS	0.045	0.045	0.044	0.042	0.039	0.053	0.042
Experiment 7							
3NN, $p = 5$, $\delta = 0.77$, $\sigma_1 = 1.00$, $\sigma_2 = 1.00$							
Mean$[\varepsilon_n[g\|S_n]] = 0.288$, Var$[\varepsilon_n[g\|S_n]] = 0.000$							
Mean	−0.140	0.009	0.006	−0.022	−0.069	−0.002	0.039
Variance	0.002	0.003	0.003	0.001	0.001	0.002	0.001
RMS	0.145	0.060	0.055	0.044	0.074	0.039	0.053
Experiment 12							
CART, $p = 5$, $\delta = 0.77$, $\sigma_1 = 1.00$, $\sigma_2 = 2.35$							
Mean$[\varepsilon_n[g\|S_n]] = 0.277$, Var$[\varepsilon_n[g\|S_n]] = 0.001$							
Mean	−0.226	0.009	0.011	−0.056	−0.031	−0.016	0.025
Variance	0.001	0.005	0.003	0.001	0.001	0.001	0.002
RMS	0.229	0.071	0.057	0.068	0.043	0.035	0.050

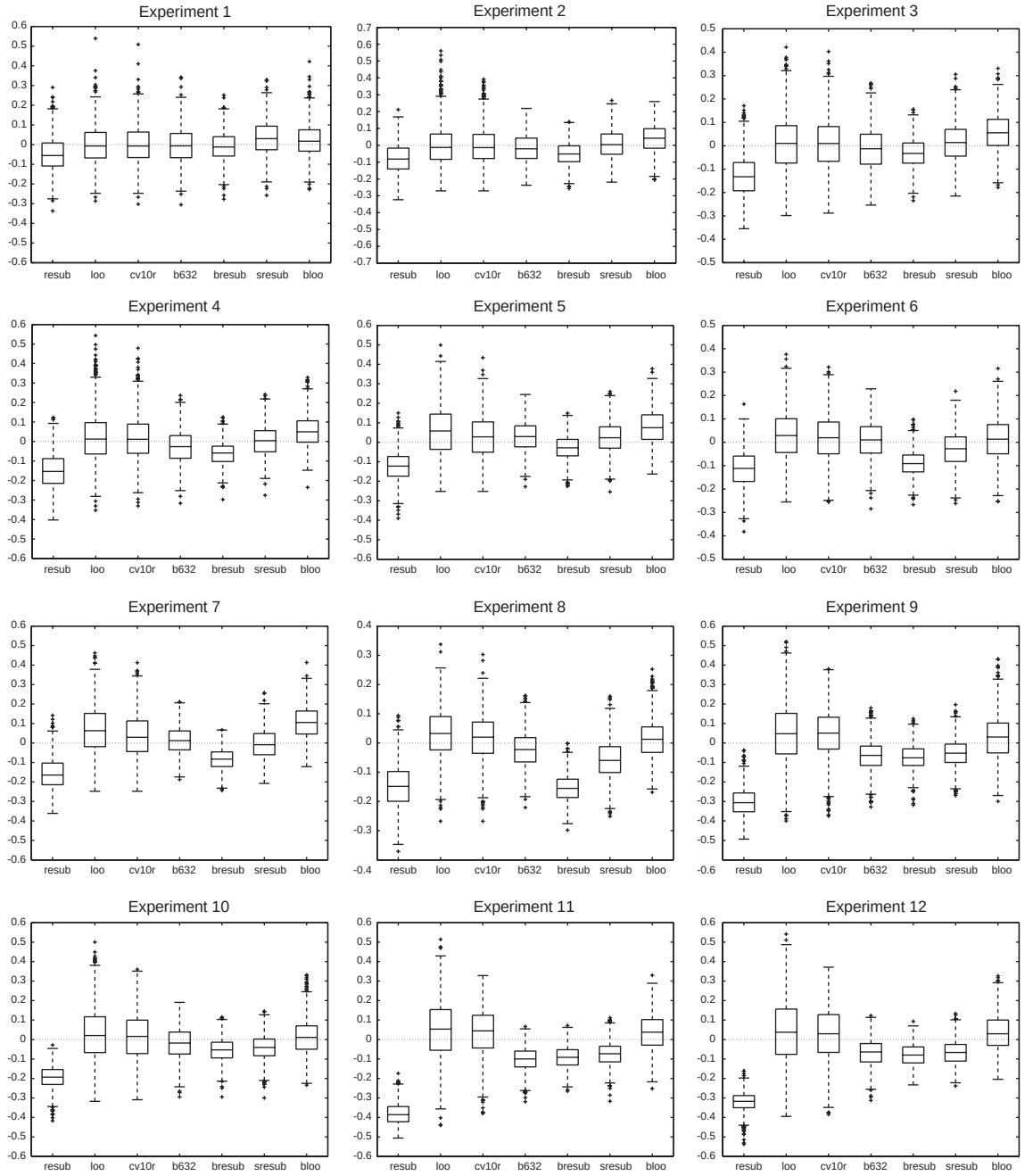

Fig. 3. Box plots of empirical deviation distribution for $n = 20$.

resubstitution. The bolstered leave-one-out is generally more variable than the bolstered resubstitution estimators, but it displays less bias. It can be seen that increasing the sample size improves the performance of all the estimators considerably, but the general relative trends discussed above still hold.

Among the bolstered error estimators considered, we recommend bolstered resubstitution as the most effective. It has the least variance among all estimators considered, including the bootstrap. For LDA or other linear classifiers, bolstered resubstitution should undoubtedly be the error estimator of choice. For more overfitting rules, and in extremely

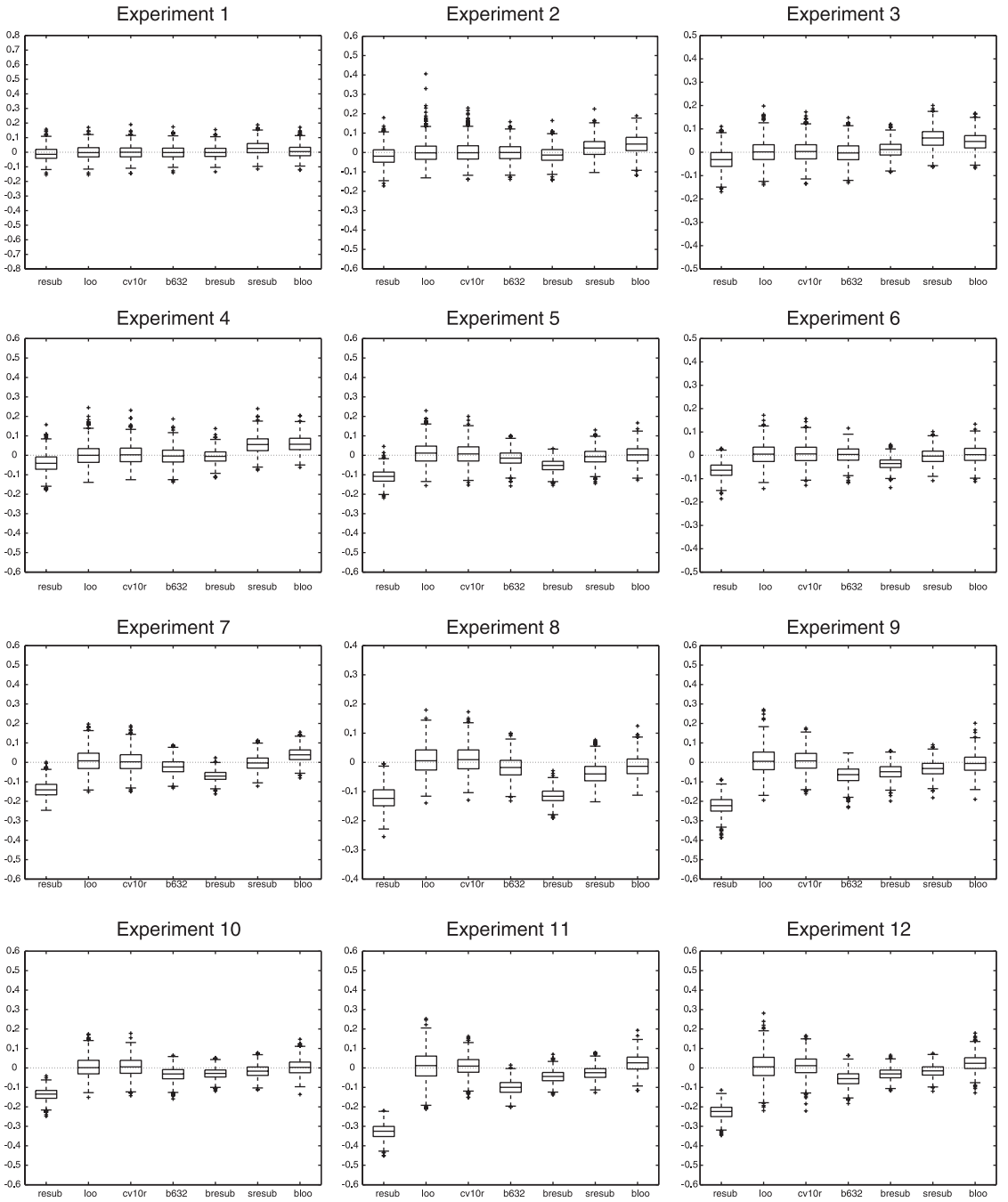

Fig. 4. Box plots of empirical deviation distribution for $n = 80$.

small-sample settings, semi-bolstered resubstitution may be a better alternative, in terms of bias (even though it is more variable); this proved to be the case especially with 3NN and $p = 5$. If bias is the most important criterion in a given application, then bolstered leave-one-out is a very efficient estimator, being often less biased than plain leave-one-out and 10-fold cross validation with repetition, and in all cases being less variable than those cross-validation estimators.

Our results show that bolstered resubstitution is very competitive with the 0.632 bootstrap estimator in terms of variance and overall RMS error, with the added benefit that its computational cost is much lower than for the bootstrap.

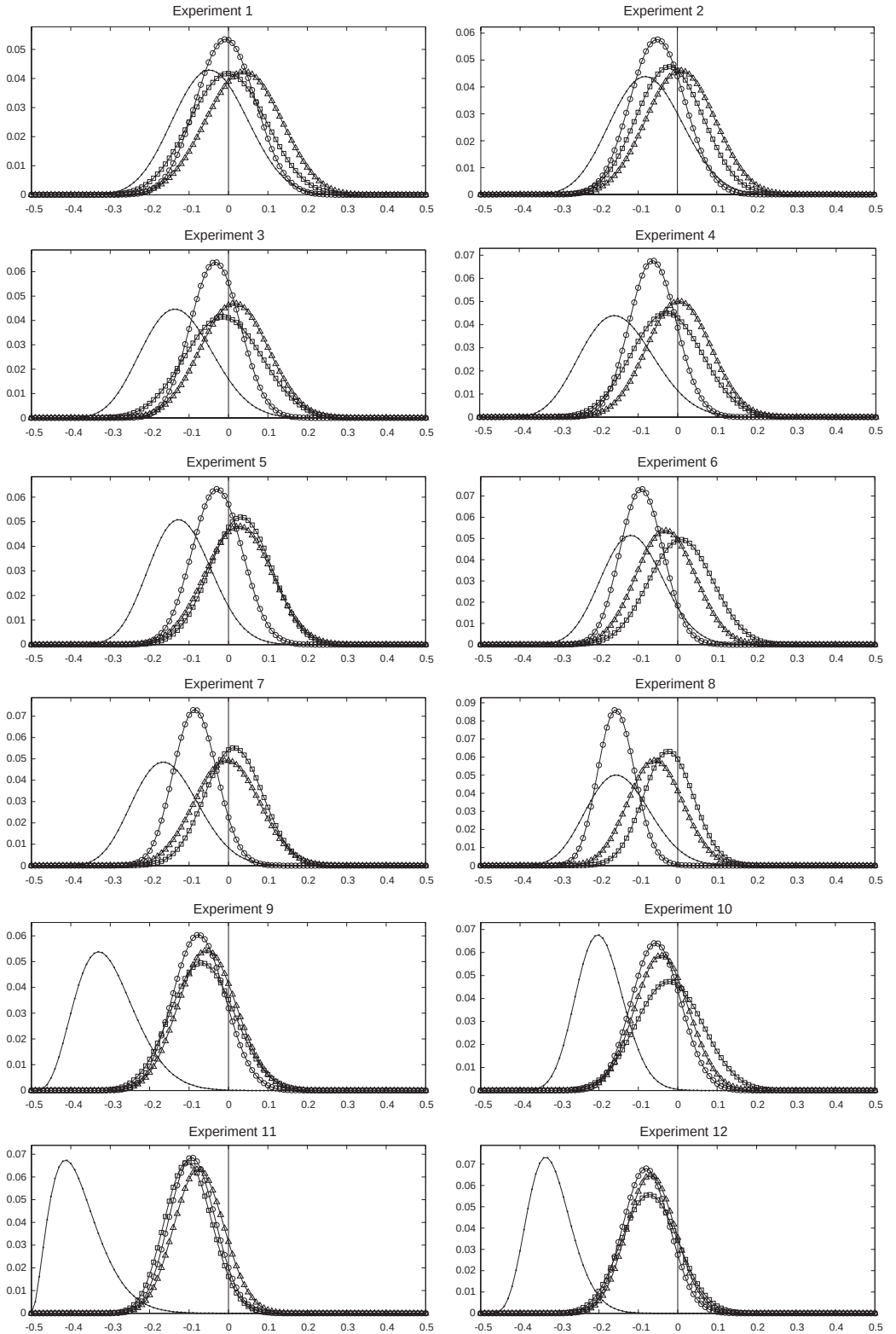

Fig. 5. Empirical deviation distribution for $n = 20$. Key: $\bullet$ = resubstitution, $\square$ = bootstrap, $\circ$ = bolstered resubstitution, $\triangle$ = semi-bolstered resubstitution.

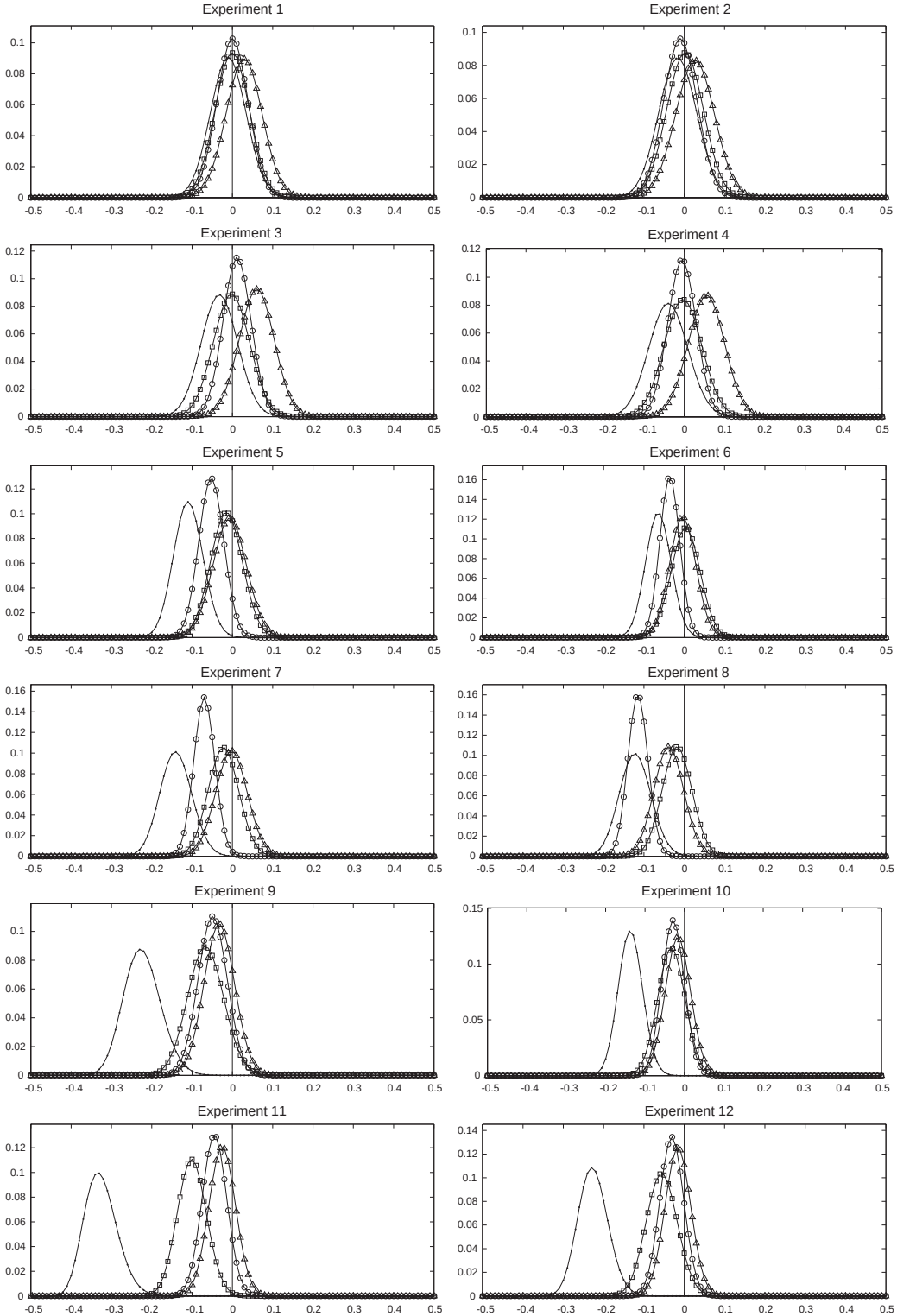

Fig. 6. Empirical deviation distribution for $n = 80$. Key: $\bullet$ = resubstitution, $\square$ = bootstrap, $\circ$ = bolstered resubstitution, $\triangle$ = semi-bolstered resubstitution.

Table 4
Average timings in milliseconds for three representative experiments

n	Resub	Loo	cv10r	b632	Bresub	Sresub	Bloo
Experiment 1							
LDA, $p=2$, $\delta=0.59$, $\sigma_1=1.00$, $\sigma_2=1.00$							
20	0.0	0.4	2.2	3.6	0.0	0.0	0.4
40	0.1	1.2	3.7	6.3	0.0	0.0	1.3
60	0.1	2.7	5.1	8.9	0.1	0.0	2.7
80	0.0	4.6	6.5	11.7	0.1	0.1	4.7
100	0.1	7.0	7.7	14.4	0.1	0.2	7.1
120	0.0	9.9	9.4	17.2	0.2	0.2	9.9
Experiment 7							
3NN, $p=5$, $\delta=0.77$, $\sigma_1=1.00$, $\sigma_2=1.00$							
20	0.0	0.0	0.6	5.2	0.7	0.7	0.6
40	0.1	0.1	1.6	13.5	2.0	1.5	1.7
60	0.2	0.3	2.7	23.8	3.1	2.7	3.3
80	0.4	0.5	5.0	40.3	4.7	4.8	4.7
100	0.6	0.8	7.0	56.0	6.7	6.1	7.0
120	0.6	0.8	8.6	76.8	8.7	8.5	9.4
Experiment 12							
CART, $p=5$, $\delta=0.77$, $\sigma_1=1.00$, $\sigma_2=2.35$							
20	0.0	1.2	5.5	6.0	0.3	0.2	1.5
40	0.0	9.4	20.2	19.7	0.5	0.5	9.7
60	0.0	31.8	45.1	44.4	0.6	0.7	32.7
80	0.0	77.3	80.7	81.7	1.0	0.9	78.8
100	0.0	154.4	127.0	131.5	1.2	1.1	156.1
120	0.0	274.8	187.2	197.0	1.5	1.3	278.1

Table 4 reports the average computation timings in milliseconds of a single run of the various error estimators, for three representative experiments. Resubstitution is the fastest estimator. Leave-one-out is fast for a small number of samples, but its performance quickly degrades with increasing numbers of samples. The 10-fold cross-validation with repetition and the bootstrap 0.632 estimator are the slowest estimators. Bolstered resubstitution is tens to hundreds of times faster than the bootstrap estimator. In a large experiment with thousands of variables, where hundreds of thousands of variable subsets may have to be considered for feature extraction, repeated cross-validation and bootstrap estimation, as well as leave-one-out if many samples are considered at a time, are problematic (perhaps prohibitive) in terms of computation time.

5. Conclusions

We have proposed in this paper an error estimation technique that is fast and accurate, and is particularly useful in small-sample settings. It improves error estimation by bolstering the empirical distribution of the data, which is accomplished by bolstering kernels applied on the training data set. This leads to error estimators that have low variance, and generally low bias as well. The amount of bolstering is automatically selected by means of a simple nonparametric technique. Results from an extensive simulation study show that bolstered resubstitution is a very attractive choice as an error estimator in small-sample settings. In our simulations, bolstered resubstitution was vastly superior to plain resubstitution and cross-validation, and was very competitive with the 0.632 bootstrap estimator, while being tens to hundreds of times faster. For overfitting classification rules, particularly 3NN in our study, the semi-bolstered resubstitution may be a better alternative in terms of lower bias. If bias is the most important criterion, then bolstered leave-one-out becomes a good alternative, especially with overfitting rules such as 3NN and CART. Forthcoming studies will elucidate further properties of the proposed bolstered error estimators.

References

[1] E. Dougherty, Small sample issues for microarray-based classification, Comp. Functional Genomics 2 (2001) 28–34.

[2] C. Smith, Some examples of discrimination, Ann. Eugenics 18 (1947) 272–282.
[3] P. Lachenbruch, M. Mickey, Estimation of error rates in discriminant analysis, Technometrics 10 (1968) 1–11.
[4] L. DeVroye, L. Gyorfi, G. Lugosi, A Probabilistic Theory of Pattern Recognition, Springer, New York, NY, 1996.
[5] R. Duda, P. Hart, Pattern Classification, 2nd Edition, Wiley, New York, NY, 2001.
[6] B. Efron, Bootstrap methods: another look at the jacknife, Ann. Stat. 7 (1979) 1–26.
[7] B. Efron, The Jacknife, The bootstrap, and other resampling plans, SIAM Monograph #38, NSF-CBMS, 1982.
[8] S. Kim, E. Dougherty, J. Barrera, Y. Chen, M. Bittner, J. Trent, Strong feature sets from small samples, Computational Biol. 9 (2002) 127–146.
[9] N. Glick, Additive estimators for probabilities of correct classification, Pattern Recognition 10 (1978) 211–222.
[10] G. Tutz, Smoothed additive estimators for non-error rates in multiple discriminant analysis, Pattern Recognition 18 (2) (1985) 151–159.
[11] S. Snapinn, J. Knoke, An evaluation of smoothed classification error-rate estimators, Technometrics 27 (2) (1985) 199–206.
[12] S. Snapinn, J. Knoke, Estimation of error rates in discriminant analysis with selection of variables, Biometrics 45 (1989) 289–299.
[13] D. Hirst, Error-rate estimation in multiple-group linear discriminant analysis, Technometrics 38 (4) (1996) 389–399.
[14] G. McLachlan, Discriminant Analysis and Statistical Pattern Recognition, Wiley, New York, NY, 1992.
[15] V. Vapnik, Statistical Learning Theory, Wiley, New York, NY, 1998.
[16] U. Braga-Neto, E. Dougherty, Is cross-validation valid for microarray classification?, Bioinformatics, 2003, in press.
[17] B. Efron, Estimating the error rate of a prediction rule: improvement on cross-validation, J. Am. Stat. Assoc. 78 (382) (1983) 316–331.
[18] I. Witten, E. Frank, Data Mining, Academic Press, San Diego, CA, 2000.
[19] T. Anderson, Classification of multivariate analysis, Psychometrika 16 (1951) 31–52.
[20] M. Evans, N. Hastings, B. Peacock, Statistical Distributions, 3rd Edition, Wiley, New York, NY, 2000.
[21] M. Chernick, Bootstrap Methods: A Practitioner's Guide, Wiley, New York, NY, 1999.

About the Author—ULISSES M. BRAGA-NETO received the Baccalaureate degree in Electrical Engineering from the Universidade Federal de Pernambuco (UFPE), Brazil, in 1992, the Master's degree in Electrical Engineering from the Universidade Estadual de Campinas, Brazil, in 1994, the M.S.E. degree in Electrical and Computer Engineering and the M.S.E. degree in Mathematical Sciences, both from The Johns Hopkins University, in 1998, and the Ph.D. degree in Electrical and Computer Engineering, from The Johns Hopkins University, in 2001. He is currently a post-doctoral fellow at the Section of Clinical Cancer Genetics of the University of Texas MD Anderson Cancer Center, and visiting scholar at the Department of Electrical Engineering of Texas A& M University. His current research interests include bioinformatics and small-sample error estimation.

About the Author—EDWARD DOUGHERTY is a professor in the Department of Electrical Engineering at Texas A& M University in College Station. He holds a Ph.D. in mathematics from Rutgers University and an M.S. in Computer Science from Stevens Institute of Technology. He is author of eleven books and editor of four others. He has published more than one hundred journal papers, is an SPIE fellow, is currently Chair of the SIAM Activity Group on Imaging Science, and has served as editor of the Journal of Electronic Imaging for six years. Prof. Dougherty has contributed extensively to the statistical design of nonlinear operator for image processing and the consequent application of pattern recognition theory to nonlinear image processing. His current research is focused in genomic signal processing, with the central goal being to model genomic regulatory mechanisms. He is head of the Genomic Signal Processing Laboratory at Texas A& M University.