BIOINFORMATICS ORIGINAL PAPER Vol. 27 no. 21 2011, pages 3056–3064
doi:10.1093/bioinformatics/btr518

*Data and text mining*

Advance Access publication September 13, 2011

# High-dimensional bolstered error estimation

Chao Sima[1], Ulisses M. Braga-Neto[2] and Edward R. Dougherty[1,2,3,*]

[1]Computational Biology Division, Translational Genomics Research Institute, Phoenix, AZ, [2]Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX and [3]Department of Pathology, University of Texas M. D. Anderson Cancer Center, Houston, TX, USA

Associate Editor: Jonathan Wren

**ABSTRACT**

**Motivation:** In small-sample settings, bolstered error estimation has been shown to perform better than cross-validation and competitively with bootstrap with regard to various criteria. The key issue for bolstering performance is the variance setting for the bolstering kernel. Heretofore, this variance has been determined in a non-parametric manner from the data. Although bolstering based on this variance setting works well for small feature sets, results can deteriorate for high-dimensional feature spaces.

**Results:** This article computes an optimal kernel variance depending on the classification rule, sample size, model and feature space, both the original number and the number remaining after feature selection. A key point is that the optimal variance is robust relative to the model. This allows us to develop a method for selecting a suitable variance to use in real-world applications where the model is not known, but the other factors in determining the optimal kernel are known.

**Availability:** Companion website at http://compbio.tgen.org/paper_supp/high_dim_bolstering

**Contact:** edward@mail.ece.tamu.edu

Received on February 10, 2011; revised on September 2, 2011; accepted on September 5, 2011

## 1 INTRODUCTION

Throughout most of the history of pattern recognition, the number of features was much smaller than the numbers currently being generated in high-throughput biology. Less than 15 years ago, in two studies on feature selection most cases considered involved <30 features and the maximum number considered was 65 (Jain and Zongker, 1997; Kudo and Sklansky, 2000). The advent of high-throughput technologies has radically altered the landscape. In conjunction with large numbers of features, bioinformatics is confronted by small sample sizes, often <100, which forces one to train and test on the same data, where bias, variance (Braga-Neto and Dougherty, 2004b) and lack of correlation with the true error (Hanczar *et al.*, 2007, 2010) can severely degrade error estimation. Performance can degrade even further in the presence of feature selection (Molinaro *et al.*, 2005). Recent articles have pointed out the difficulty in establishing performance advantages for proposed classification rules (Boulesteix, 2010; Jelizarow *et al.*, 2010; Rocke *et al.*, 2009). Two statistically grounded sources of overoptimism have been highlighted: (i) applying a classification rule to numerous datasets and then reporting only the results on the dataset for which the designed classifier possesses the lowest estimated error (Yousefi *et al.*, 2010); and (ii) applying multiple classification rules to a dataset and comparing the classification rules according to the estimated errors of the designed classifiers (Boulesteix and Strobl, 2009). In both cases, optimism is a result of inaccurate error estimation.

A good error estimator ideally would have small bias and small variance. This is a difficult trade-off in small-sample settings. In small-sample cases, resubstitution generally has small variance but tends to be quite optimistically biased. Cross-validation has small bias, but tends to display high variance. Bolstered error estimation (Braga-Neto and Dougherty, 2004a) attempts to achieve a compromise to this bias-variance dilemma in small-sample settings. It is based on the idea of modifying ('bolstering') the empirical distribution of the data by placing kernels at each data point and then estimating classifier error by the error on this bolstered empirical distribution in such a way that it reduces bias, while at the same time reducing variance. Bolstered error estimation has shown good performance when compared with popular error estimators in small-sample settings, in particular, for feature-set ranking and when used internally within a feature-selection algorithm (Sima *et al.*, 2005a) and for ranking feature sets (Sima *et al.*, 2005b). Its good performance, including the latter applications, has been demonstrated in the context a small number of features, including feature selection via sequential forward selection (SFS), where it is applied to small potential feature sets in the SFS algorithm. A critical aspect of the method is selecting the right amount of bolstering, which is given by the variance of the bolstering kernels. The original bolstering paper (Braga-Neto and Dougherty, 2004a) proposed a non-parametric estimator for the kernel variance, which was found empirically to perform well in low-dimensional spaces; however, estimation was found to degrade in high dimensions, so that a correction factor can be required (Vu and Braga-Neto, 2008). In fact, it was demonstrated in a preliminary study that a correction factor can also be beneficial for low-dimensional bolstering (Huynh *et al.*, 2007).

This leads us to consider optimal bolstering, specifically, finding an optimal variance for the bolstering kernels. Error estimators like resubstitution and cross-validation (assuming the number of folds is preset) are non-parametric. They contain no free parameters. This is not the case for bootstrap. In general, bootstrap has the form of a convex error estimator, namely,

$$\hat{\varepsilon}^{a}_{\text{boot}} = (1-a)\hat{\varepsilon}_{\text{resub}} + a\hat{\varepsilon}_{\text{zero}}, \tag{1}$$

where $\hat{\varepsilon}_{\text{resub}}$ and $\hat{\varepsilon}_{\text{zero}}$ are the resubstitution and zero-bootstrap estimators and $0 \leq a \leq 1$. The zero-bootstrap utilizes the empirical distribution $\mathbf{F}^*$, which puts mass $\frac{1}{n}$ on each of the $n$ available

*To whom correspondence should be addressed.

data points. A bootstrap sample $S_n^*$ from $\mathbf{F}^*$ consists of $n$ equally-likely draws with replacement from the original data $S_n$. The basic *bootstrap zero estimator* (Efron, 1983) is written in terms of the empirical distribution as

$$\hat{\epsilon}_0 = E_{\mathbf{F}^*}\left(|Y - g(S_n^*, X)| : (X,Y) \in S_n \setminus S_n^*\right). \quad (2)$$

In practice, the expectation $E_{\mathbf{F}^*}$ has to be approximated by a Monte-Carlo estimate based on independent replicates $S_n^{*b}$, for $b = 1, \ldots, B$, in which case the classifier is designed on the bootstrap sample and tested on the original data points left out. An optimal bootstrap estimator results from a value of $a$ that minimizes the mean-square error between $\hat{\varepsilon}_{\text{boot}}^a$ and the true error for a given feature-label distribution (Sima and Dougherty, 2006b). Setting $a = 0.632$, as is commonly done (Efron, 1983), can lead to a far from optimal estimator (optimal weights).

The present article considers optimal bolstering relative to its one free parameter, kernel variance and the manner in which optimal bolstering can be used to arrive at practical implementation of bolstering in high-dimensional feature space. The end product is an implementation protocol in which optimal kernel variances across different models are combined to produce a suitable kernel variance for the problem at hand. Throughout, we will assume feature selection because that would be the standard way to approach classification in the high-dimensional setting we are considering, although this is not a mandatory requirement of the approach.

## 2 SYSTEMS AND METHODS

This section will be broken into subsections, with the aim of arriving at the implementation protocol for real-world data. Section 2.1 briefly reviews the necessary essentials of error estimation, mainly bolstering. Section 2.2 defines the scaling factor by which to adjust the bolstering kernel to high dimensions. Section 2.3 discusses optimization of the scaling factor and illustrates the construction of a set of optimal scaling factors across a family of models varying in both structure and classification difficulty. Section 2.4 provides the implementation of high-dimensional bolstered resubstitution based on a family of optimal scaling factors.

### 2.1 Error estimation

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, a classifier is designed from *training data*, which is 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 $\Psi_n : \{\mathbb{R}^p \times \{0,1\}\}^n \times \mathbb{R}^p \to \mathcal{F}$, where $\mathcal{F}$ is the set of mappings from $\mathbb{R}^p$ into $\{0,1\}$. It maps $S_n$ into a *classifier* $\psi_n : \mathbb{R}^p \to \{0,1\}$. The *classification error* $\varepsilon_n$ is the probability of an erroneous classification:

$$\varepsilon_n = P(\psi_n(X) \neq Y | S_n) = E_{\mathbf{F}}(|Y - \psi_n(X)|), \quad (3)$$

where $E_{\mathbf{F}}$ denotes expectation with respect to $\mathbf{F}$. Were $\mathbf{F}$ known, then the error could be found via Equation (3). In practice, one must use an error estimator $\hat{\varepsilon}_n$. An error estimator can suffer from *bias*, $\text{Bias} = E[\hat{\varepsilon}_n - \varepsilon_n]$, and *deviation variance*, $\text{Var}_{\text{dev}} = \text{Var}[\hat{\varepsilon}_n - \varepsilon_n]$. These combine to contribute to the most common measure (used herein) for evaluating the accuracy of an error estimator, the *root-mean-square* (*RMS*):

$$\text{RMS} = \sqrt{E[|\hat{\varepsilon}_n - \varepsilon_n|^2]} = \sqrt{\text{Var}_{\text{dev}} + \text{Bias}^2}. \quad (4)$$

*2.1.1 Classical error estimation* The simplest way to estimate the error in the absence of independent test data is to compute its error directly on the sample data itself. This *resubstitution estimator*, $\hat{\varepsilon}_{\text{resub}}$, is usually optimistic (i.e. biased low), sometimes very much so.

In *k-fold cross-validation*, the dataset $S_n$ is partitioned into $k$ *folds* $S_n^{(i)}$, 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, $\hat{\varepsilon}_{\text{cv}}$, is the overall proportion of error on all folds. A $k$-fold cross-validation estimator is unbiased as an estimator of $\varepsilon_{n-n/k}$. Cross-validation estimators are pessimistic, since they use smaller training sets to design the classifier; however, their bias tends to be small. Their main drawback is their large variance (Braga-Neto and Dougherty, 2004b; Devroye *et al.*, 1996). Sometimes cross-validation is repeated some number of times with different fold partitions and the results averaged. In this article, we use 10-fold cross-validation without repetition.

A recently developed estimation method, called *adjusted bootstrap* ($\hat{\varepsilon}_{\text{abs}}$), which carries out further bootstrap resampling in each fold, has been found to have good RMS performances (Jiang and Simon, 2007). Specifically, $S_n$ is partitioned into $n$ folds and, for each sample left out for testing, $B$ bootstrap sample sets of size $ln$ are drawn from the remaining $n-1$ points, $l = 1, 2, \ldots, L$. For each $l$, the error $e_l$ is the proportion of misclassified samples across $n$ folds and $B$ bootstrap sample sets. Finally, the *adjusted bootstrap error* $\hat{\varepsilon}_{\text{abs}}$ is computed in the form

$$\hat{\varepsilon}_{\text{abs}} = \hat{a} n^{-\hat{c}} + \hat{b},$$

where $\hat{a}, \hat{b}$ and $\hat{c}$ are least squares estimates for the function

$$e_l = a(n \cdot u_l)^{-c} + b,$$

and $u_l$ is the proportion of the expected number of non-repeated samples in a size $ln$ bootstrap sample set.

*2.1.2 Bolstered error estimation* The *empirical feature-label distribution* $F^*$ is a discrete distribution that puts mass $\frac{1}{n}$ on each of the $n$ available data points. The resubstitution estimator can be written in terms of the empirical feature-label distribution as

$$\hat{\varepsilon}_{\text{resub}} = E_{F^*}[|Y - \psi_n(X)|]. \quad (5)$$

Relative to $F^*$, no distinction is made between points near or far from the decision boundary. If one spreads the probability mass of the empirical distribution at each point, then variation is reduced because points near the decision boundary will have more mass on the other side of the boundary than will points far from the decision boundary. Consider a probability density function $f_i^{\diamond}$, for $i = 1, \ldots, n$, called a *bolstering kernel*, and define the *bolstered empirical distribution* $F^{\diamond}$, with probability density function given by

$$f^{\diamond}(X) = \frac{1}{n} \sum_{i=1}^{n} f_i^{\diamond}(X - X_i). \quad (6)$$

The *bolstered resubstitution* estimator (Braga-Neto and Dougherty, 2004a) is obtained by replacing $F^*$ by $F^{\diamond}$ in Equation (5) to obtain

$$\hat{\varepsilon}_{\text{bolst}} = E_{F^{\diamond}}[|Y - \psi(X)|]. \quad (7)$$

Bolstering can be applied to other error estimators; however, we only use bolstered resubstitution, the bolstering method used the most to date.

The bolstered resubstitution estimator is given by

$$\hat{\varepsilon}_{\text{bolst}} = \frac{1}{n} \sum_{i=1}^{n} \left( I_{y_i=0} \int_{A_1} f_i^{\diamond}(x - x_i)\,dx + I_{y_i=1} \int_{A_0} f_i^{\diamond}(x - x_i)\,dx \right), \quad (8)$$

where $A_j = \{x \,|\, \psi(x) = j\}$. The integrals are the error contributions made by the data points, according to whether $y_i = 0$ or $y_i = 1$. If the classifier is linear, then the decision boundary is a hyperplane and it is usually possible to find

analytical expressions for the integrals; otherwise, Monte-Carlo integration can be employed.

The amount of bolstering determines the variance and bias properties (hence, RMS also) of the bolstered estimator. As a general rule, wider bolstering kernels lead to lower variance estimators, but after a certain point this advantage becomes offset by increasing pessimistic bias. In the other direction, insufficiently wide kernels tend to result in optimistic bias. A zero-mean, spherical Gaussian bolstering kernel $f_i^{\diamond}$ with covariance matrix of the form $\kappa_i^2\mathbf{I}$, where $\mathbf{I}$ is the identity matrix, has been proposed (Braga-Neto and Dougherty, 2004a), and has been shown to work well in low-dimensional feature spaces. 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:

$$\hat{d}_y = \frac{\sum_{i=1}^{n} \min_{j\neq i}\{||x_i - x_j||\} : I_{y_i=y}}{\sum_{i=1}^{n} I_{y_i=y}}. \tag{9}$$

The estimate $\hat{d}_y$ is the mean minimum distance between points belonging to class $y$. Next, let $f_i^{\diamond,1}$ be a unit-variance bolstering kernel, $R_i$ be the random variable equal to the distance of a point randomly selected from $f_i^{\diamond,1}$ to the origin and $F_{R_i}(x)$ be the cumulative distribution function of $R_i$. In the case of the bolstering kernel $f_i^{\diamond}$ with covariance matrix $\kappa_i^2\mathbf{I}$, all distances get multiplied by $\kappa_i$. In Braga-Neto and Dougherty (2004a), a single variance $\kappa_y^2$ is estimated for all points from class $y$, such that the median distance of a test point to the origin is equal to the estimated true mean distance $\hat{d}_y$. This implies that half of the ' mass' (i.e. the ' test points' ) of the bolstering kernel will be farther from the center than $\hat{d}_y$ and the other half will be nearer. Hence, $\kappa_y$ is the solution of the equation $\kappa_y F_{R_i}^{-1}(1/2) = \hat{d}_y$. Letting $\alpha_p = F_{R_i}^{-1}(1/2)$, and recognizing that the $R_i$ are identically distributed, the estimated SDs for the bolstering kernels are given by

$$\kappa_i = \frac{\hat{d}_{y_i}}{\alpha_p}, \tag{10}$$

for $i = 1, 2, ..., n$.

## 2.2 High-dimensional bolstered resubstitution

In high-dimensional settings, it is commonplace to perform feature selection and, when performed, feature selection is part of the classification rule, with the entire set of potential features constituting the feature set relative to the classification rule. Feature selection constrains the space of functions from which a classifier might be chosen, but it does not reduce the number of features in the design process. This is why when using cross-validation error estimation, feature selection has to be carried out in each partitioned fold.

If we perform feature selection on a $D$-dimensional dataset $S_n^D$ and arrive at a $d$-dimensional set $S_n^d$ ($d < D$), then the bolstered error estimator can use the previously defined kernel size $\kappa_i$, computed on $S_n^D$, not $S_n^d$. Specifically, the mean minimum distance $\hat{d}_y$ is estimated on $S_n^D$ and $\alpha_p = \alpha_D$. For high dimensions, we replace $\kappa_i$ by

$$\kappa_i^* = k_D \times \frac{\hat{d}_{y_i}^D}{\alpha_D}, \tag{11}$$

where $k_D$ is an additional scaling factor determined by the dimension and where we have indicated the dimension in the mean minimum distance estimate. The idea is to adjust the kernel size by choosing $k_D$ so the bolstered error estimator will be optimal (minimum RMS). $k_D = 1$ yields the previously proposed kernel variance. In essence, $\kappa_i$ is a parameter for the bolstered estimator and Equation (11) sets it free, thereby allowing for optimization. The situation is akin to 0.632 bootstrap as opposed to optimal bootstrap.

Given the kernel sizes, the bolstered resubstitution error estimate is given by Equation (8) in $D$ dimensions. For Gaussian kernels with independent variables, this integral reduces. Let $f_i^{\diamond,d}(x - x_i)$ and $f_i^{\diamond,D-d}(x - x_i)$ denote the Gaussian kernels in $d$- and $(D-d)$-dimensional spaces, respectively, so that the $D$-dimensional Gaussian kernel decomposes as

$$f_i^{\diamond}(x - x_i) = f_i^{\diamond,d}(x - x_i) f_i^{\diamond,D-d}(x - x_i). \tag{12}$$

Denoting $x - x_i$ as $\Delta x_i$, then Equation (8) can be rewritten as

$$\begin{aligned}
\hat{\varepsilon}_{\text{bolst}}^D &= \frac{1}{n}\sum_{i=1}^{n}\left(I_{y_i=0}\int_{A_1} f_i^{\diamond,d}(\Delta x_i) f_i^{\diamond,D-d}(\Delta x_i)\,dx\right. \\
&\quad \left. + I_{y_i=1}\int_{A_0} f_i^{\diamond,d}(\Delta x_i) f_i^{\diamond,D-d}(\Delta x_i)\,dx\right) \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(I_{y_i=0}\int_{A_1^d} f_i^{\diamond,d}(\Delta x_i)\,dx \int_{-\infty}^{\infty} f_i^{\diamond,D-d}(\Delta x_i)\,dx + \right. \\
&\quad \left. I_{y_i=1}\int_{A_0^d} f_i^{\diamond,d}(\Delta x_i)\,dx \int_{-\infty}^{\infty} f_i^{\diamond,D-d}(\Delta x_i)\,dx\right) \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(I_{y_i=0}\int_{A_1^d} f_i^{\diamond,d}(\Delta x_i)\,dx + I_{y_i=1}\int_{A_0^d} f_i^{\diamond,d}(\Delta x_i)\,dx\right),
\end{aligned} \tag{13}$$

where $A_j^d, j = 0, 1$, is the projection of the classifier decision region $A_j$ into $d$-dimensional space, and we added a superscript '$D$' to the bolstered error estimator to indicate it refers to the error in $D$-dimensional space. The previous result indicates that the integrals necessary to find the bolstered error estimate in $D$-dimensional space can be equivalently carried out in $d$-dimensional space. This is akin to resubstitution, where the error count is the same whether it is done in $D$- or $d$-dimensional space. For performance comparison purposes, we will also estimate the kernel size using only the low-dimensional data $S_n^d$, resulting in a bolstered error estimator $\hat{\varepsilon}_{\text{bolst}}^d$, which uses the originally proposed kernel variance (no correction, or $k_D = 1$). For feature selection, we will use sequential forward floating search (SFFS) (Pudil *et al.*, 1994).

## 2.3 Optimization method

To find the optimal kernel scaling factor $k_D$, we utilize the following procedure:

*Protocol* 1

(1) Generate a sample set $S_n^D$ of size $n$ and a total of $D$ features from a specified synthetic model.
(2) Select a size-$d$ feature set $A$ using a feature-selection method $F$ on $S_n^D$, resulting in a reduced dimension sample set $S_n^d$ for the feature set $A$.
(3) Design a classifier $\psi_n$ for $S_n^d$ according to the given classification rule $\Psi_n$.
(4) Compute the true error $\varepsilon_n$ using the underlying distribution of the model.
(5) Compute the 10-fold cross-validation error $\hat{\varepsilon}_{\text{cv}}$ (keeping in mind that feature selection must be repeated for each fold).
(6) Compute the bolstered error $\hat{\varepsilon}_{\text{bolst}}^d$.
(7) Compute the bolstered errors $\hat{\varepsilon}_{\text{bolst}}^{D,i}$ for a list of kernel scaling factors $k_{D,1}, k_{D,2}, ....$
(8) Calculate RMS for each error estimator by repeating Steps 1 through 7 a number $N$ of times.
(9) Repeat Steps 1 through 8 for different models $M$, different levels of model complexities and different classification rules $\Psi_n$.

We consider four data models, each a two-class Gaussian model with equally likely classes and class-conditional densities having covariance matrices $\Sigma_1$ and $\Sigma_2$. One class mean is located at $-\vec{\mu}$ and the other at $\vec{\mu}$, with the location of $\vec{\mu} = \delta * [a_1\ a_2\ \ldots\ a_D]$ depending on the model. The parameter $\delta$ is chosen to achieve prescribed values for the expected classification error $E[\varepsilon_n]$; different values of $E[\varepsilon_n]$ represent different levels of difficulty at sample size $n$.

**Table 1.** Summary of simulation experiments

| | | |
|---|---|---|
| Data models | $\mathcal{M}$ | **M1**, **M2**, **M3**, **M4** |
| Model difficulty | $E[\varepsilon_n]$ | 0.05, 0.10, 0.15 |
| Classification rules | $\Psi_n$ | LDA,3NN, LSVM NNet,CART |
| Feature-selection methods | $\mathcal{F}$ | SFFS |
| No. of repetitions | $N$ | 500 |
| No. of sample size | $n$ | 50, 100, 150 |
| No. of selected features | $d$ | 5, 10 |
| No. of total features | $D$ | 200, 500 |
| Kernel scaling factors | $k_D$ | 0.2 to 2.0 in 0.2 increment |
| $d_0 = 10, G = 20, c = 2.25, \rho = 0.25$ | | |

LDA, linear discriminant analysis; 3NN, 3-nearest-neighbor; LSVM, linear support vector machine; NNet, neural net; CART, classification and regression tree.

**M1**: A simple linear model in which $\Sigma_1 = \Sigma_2 = \mathbf{I}$, the identity matrix, so that all features are uncorrelated. $a_i = 1$ for $i = 1, 2, \ldots d_0$ and uniformly distributed over $[0, 1]$ for $i = d_0+1, d_0+2, \ldots D$ before all $a_i$'s are randomly permuted.

**M2**: Similar to $M1$ but with $\Sigma_1 = \mathbf{I}$ and $\Sigma_2 = c\mathbf{I}$, where $c$ is a constant and $c \neq 1$. The Bayesian decision boundary is quadratic.

**M3**: This is a *Block Covariance Model* where all features are equally divided into $G$ groups. The features from different groups are uncorrelated and the features from the same group possess the same correlation, $\rho$, among each other. The structure of the covariance matrix is

$$\Sigma_1 = \Sigma_2 = \Sigma_G = \underbrace{\begin{bmatrix} \Sigma_\rho & 0 & \cdots & 0 \\ 0 & \Sigma_\rho & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma_\rho \end{bmatrix}}_{G \text{ blocks}},$$

where $\Sigma_\rho$ has 1 on the diagonal and $\rho$ off the diagonal. Here $a_i = 1$ for $i = 1, 2, \ldots D$.

**M4**: Similar to $M3$, but with $\Sigma_1 = \Sigma_G$ and $\Sigma_2 = c\Sigma_G$.

Table 1 gives a summary of the simulation experiments. Two limiting factors should be noted. First, the maximum total number of features, 500, is smaller than those often considered in microarray studies and, second, the number of selected features is kept to 5 or 10. There are three reasons for this, one pragmatic to our set of simulations and the others having to do with the nature of feature selection. The pragmatic reason is computational: we wish to do a large simulation study and therefore want to limit the computational burden. As for feature selection, given the sample sizes, it is prudent to keep the numbers of total and selected features small to have satisfactory feature selection (Sima and Dougherty, 2006a) and the number of selected features small to avoid the peaking phenomena (Hua *et al.*, 2005, 2009). Regarding the total number of features, limiting the total number of features via prior biological knowledge or requirements on data quality raises the likelihood of finding good feature sets via feature selection (Zhao *et al.*, 2010). Regarding the efficacy of selecting small feature sets, studies have shown that good classification can be achieved with two or three genes when re-examining data from studies that had originally used much larger feature sets (Braga-Neto, 2007; Grate, 2005).

We plot the RMS versus kernel scaling factor $k_D$ for $\hat{\varepsilon}^D_{\text{bolst}}$, using all combinations of simulation parameters displayed in Table 1. Additionally, we compute the RMS for LDA with $D = 200$, $d = 3$ and $n = 50$ for $E[\varepsilon_n] = 0.20, 0.25, 0.30, 0.35, 0.40$ and $0.45$. For comparison, RMS values for $\hat{\varepsilon}^d_{\text{bolst}}$, $\hat{\varepsilon}_{\text{cv}}$ and $\hat{\varepsilon}_{\text{abs}}$ are also plotted, which appear as horizontal lines as they are not related to $k_D$. Here, we present some typical results, the complete set of plots appearing on the companion website. Note that due to the intensive computing in $\hat{\varepsilon}_{\text{abs}}$ we only compute it for LDA with $D = 200$, $d = 3$ and $n = 50$.

Figure 1 shows the result for LDA, $n = 50$, and selecting $d = 3$ out of $D = 200$ features for nine values of $E[\varepsilon_n]$. Letting $k_D^{\min}$ denote the value of $k_D$ achieving minimum RMS, we see that $k_D^{\min}$ increases for increasing expected error, the increase being slight for small expected errors but becoming significant for large expected errors (for $E[\varepsilon_n] = 0.40$ and $0.45$, $k_D^{\min}$ is to the right of where we have stopped the plots at $k_D = 2.0$; see extended plots to $k_D = 3.0$ for these cases on the companion website). We observe that $\hat{\varepsilon}_{\text{cv}}$ and $\hat{\varepsilon}_{\text{abs}}$ typically perform better than $\hat{\varepsilon}^d_{\text{bolst}}$, sometimes by a large margin. However, for an appropriate kernel scaling factor, $\hat{\varepsilon}^D_{\text{bolst}}$ outperforms $\hat{\varepsilon}_{\text{abs}}$ and often outperforms $\hat{\varepsilon}_{\text{cv}}$ by a wide margin. This improvement is achieved by a range of scaling factors and is robust across different models and complexities. Regarding model robustness, for a fixed value of $E[\varepsilon_n]$ the RMS curves are remarkably similar; in particular, the value, $k_D^{\min}$, of $k_D$ achieving minimum RMS is consistent. In three cases, $k_D^{\min}$ remains fixed across the models and in the others it changes by not $> 0.2$. Moreover, in the latter cases, the RMS at the different values of $k_D^{\min}$ is approximately the same. The overall robustness has important practical implications, because in real-world problems we do not know the data model or its level of difficulty, but we do know the sample size $n$, the total and selected numbers of features, $D$ and $d$, and the classification rule. As we will subsequently see, the fact that $k_D^{\min}$ is robust relative to the data model means that, in practice, we can derive a value of $k_D$, albeit not optimal, that can be used in $\hat{\varepsilon}^D_{\text{bolst}}$ for a better error estimator.

There is also robustness with respect to the classification rule and number of features. Figure 2a and b show robustness curves for 3NN and CART, respectively, for complexity $E[\varepsilon_n] = 0.10$ (more on the companion website) for $n = 50$, $d = 3$ and $D = 200$. The curves bear a strong resemblance to the corresponding curves for LDA in Figure 1 and for all models $k_D^{\min} = 0.8$, as with LDA. We again have LDA in Figure 2c for $E[\varepsilon_n] = 0.10$ (more on the companion website), but now with $D = 500$. Again there is resemblance to the corresponding case in Figure 1 and again $k_D^{\min} = 0.8$ for all models.

The preceding observations are mostly constrained to small samples. When $n$ is large, the benefits of using $\hat{\varepsilon}^{D,\text{opt}}_{\text{bolst}}$ tend to diminish. Figure 3a and b show RMS curves for LDA for $n = 100$ and $n = 150$, respectively, $E[\varepsilon_n] = 0.10$ (more on the companion website), $d = 3$ and $D = 200$. If the model is known, an optimal $\hat{\varepsilon}^D_{\text{bolst}}$ is achievable, but robustness diminishes. For $n = 100$, there is still some robustness, but for $n = 150$, even a small deviation from $k_D^{\min}$ can result a worse performance than $\hat{\varepsilon}_{\text{cv}}$. Hence, for $n = 150$, choosing an appropriate $\hat{\varepsilon}^{D,\text{opt}}_{\text{bolst}}$ is not feasible in practice; however, since our interest is using bolstered error estimation for very small samples, this is not a significant drawback.

## 2.4 Implementation for real data

For practical application, based on the sample size, the total and selected numbers of features, and the classification rule, we will perform a model-based analysis like the ones we have performed, thereby resulting in a look-up table of pairs $(E[\varepsilon_n], k_D^{\min})$ as in Figure 1. To illustrate, by averaging across the four models, we obtain the following table for $(E[\varepsilon_n], k_D^{\min})$: $(0.05, 0.8)$, $(0.10, 0.8)$, $(0.15, 0.9)$, $(0, 20, 1.0)$, $(0.25, 1.2)$, $(0.30, 1.3)$, $(0, 35, 1.6)$. Upon designing a classifier from the data, we will obtain the 10-fold cross-validation error estimate, $\varepsilon_0$, and then, in the fashion of the method of moments, set $E[\varepsilon_n] = \varepsilon_0$ and choose the corresponding value of $k_D^{\min}$ to serve as the scaling factor for bolstering. Since the look-up table is discrete, $E[\varepsilon_n] = \varepsilon_0$ must be solved approximately by interpolation. Corresponding to the seven values of $k_D^{\min}$ in the look-up table for Figure 1, we have: $k_D^{\min} = 0.8$ for $\varepsilon_0 < 0.125$, $k_D^{\min} = 0.9$ for $0.125 \leq \varepsilon_0 < 0.175$, $k_D^{\min} = 1.0$ for $0.175 \leq \varepsilon_0 < 0.225$, $k_D^{\min} = 1.2$ for $0.225 \leq \varepsilon_0 < 0.275$, $k_D^{\min} = 1.3$ for $0.275 \leq \varepsilon_0 < 0.325$ and $k_D^{\min} = 1.6$ for $0.325 \leq \varepsilon_0$. If one so desires, then a finer

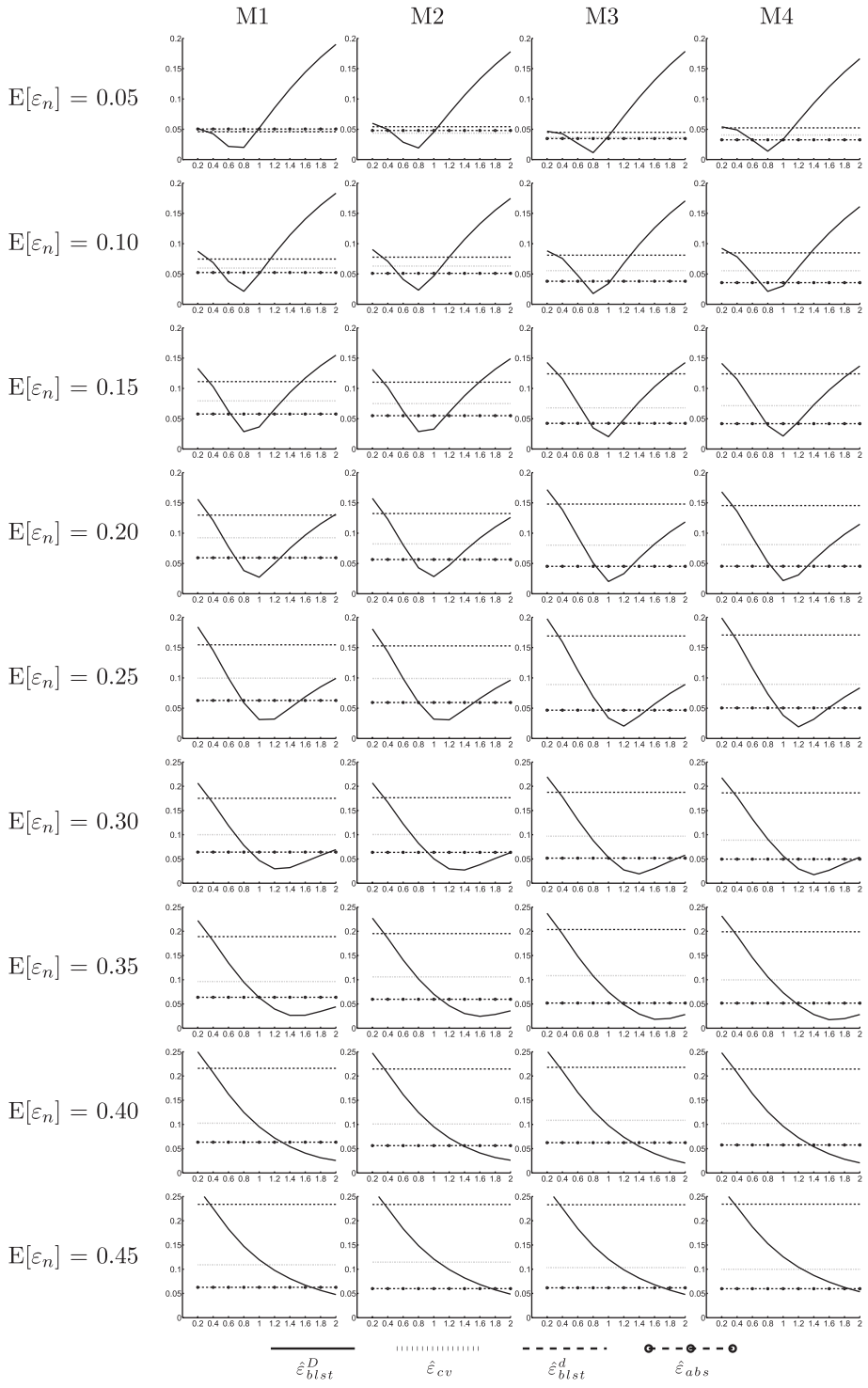


**Fig. 1.** RMS versus scaling factor $k_D$ for LDA with sample size $n=50$, total feature size $D=200$ and selected feature size $d=3$.

selection of expected errors and interpolation can be obtained. One might also use a coarser interpolation for computational purposes, with some loss of performance. In fact, that is precisely what we do here because we will subsequently perform a computationally intensive robustness analysis. Here we use: $k_D^{\min}=0.8$ for $\varepsilon_0<0.125$; $k_D^{\min}=1$ for $0.125\leq\varepsilon_0<0.225$; $k_D^{\min}=1.2$ for $0.225\leq\varepsilon_0\leq0.275$; $k_D^{\min}=1.4$ for $0.275\leq\varepsilon_0\leq0.325$; and $k_D^{\min}=1.6$ for $\varepsilon_0>0.325$.

The final bolstered error estimate is computed from the data using this scaling factor. The success of the procedure depends on robustness in choosing a scaling factor because (i) the estimated model will be inaccurate

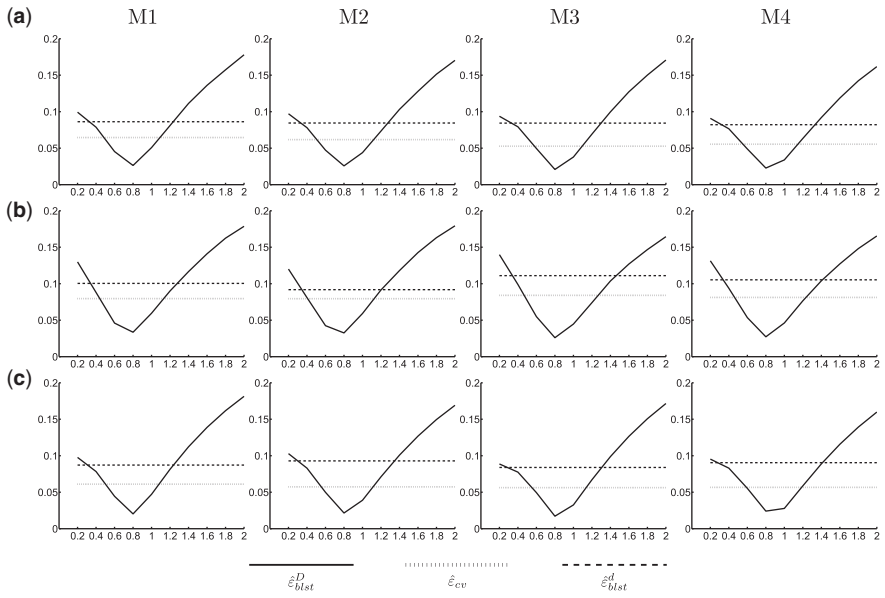


**Fig. 2.** RMS versus scaling factor $k_D$ for sample size $n = 50$ and selected feature size $d = 3$, for (**a**) 3NN with total feature size $D = 200$, (**b**) CART with $D = 200$ and (**c**) LDA with $D = 500$.

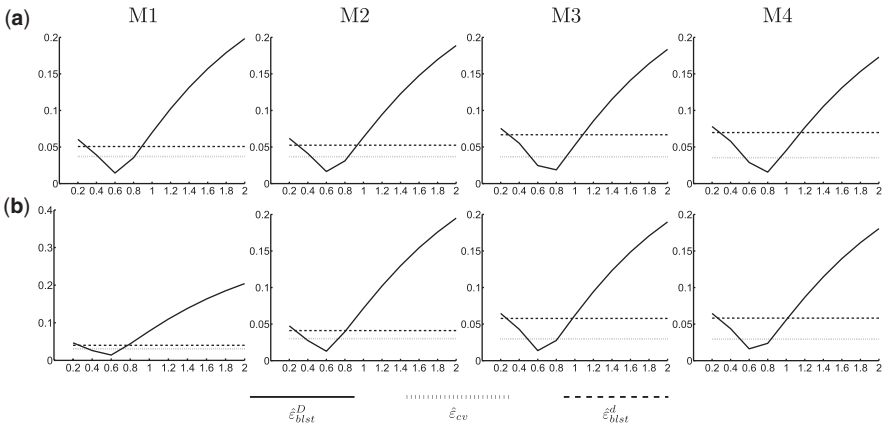


**Fig. 3.** RMS versus scaling factor $k_D$ for LDA with total feature size $D = 200$ and selected feature size $d = 3$, for (**a**) sample size $n = 100$ and (**b**) $n = 150$.

owing to small sample size, (ii) cross-validation has significant variance for small samples, (iii) the estimated model will differ to some extent from the models involved in creating the look-up table and (iv) the method of moments is not optimal. The following protocol is used to obtain the bolstered resubstitution error estimate:

*Protocol* 2

(1) Given a sample set $S_n^D$ with size $n$ and dimension $D$, select a size-$d$ feature set $A$ using a feature-selection method $F$ on $S_n^D$, resulting in a reduced dimension sample set $S_n^d$ for the feature set $A$.

(2) Design a classifier $\psi_n$ for $S_n^d$ according to the given classification rule $\Psi_n$, and compute the 10-fold cross-validation error estimate $\varepsilon_0$.

(3) From the look-up table $(E[\varepsilon_n], k_D^{\min})$ choose the kernel scaling factor $k_D^{\min}$ by setting $E[\varepsilon_n] = \varepsilon_0$.

(4) Compute the bolstered error estimate $\hat{\varepsilon}^{D,\text{data}}_{\text{bolst}}$ using the selected scaling factor.

## 3 RESULTS AND DISCUSSION

To illustrate application, we have applied the method to two gene expression datasets:

- *Myeloma dataset*: data are downloaded from the NIH Gene Expression Omnibus (GEO) under accession numbers GSE5900 and GSE2658, which contain 54 613 probe sets and 559 multiple myeloma (MM) samples, as well as 3 other subtypes [monoclonal gammopathy of undetermined significance (MGUS)], 44 samples; smoldering MM (SMM), 12 samples; healthy donors with normal plasma cell (NPC), 22 samples (Zhan *et al.*, 2006). Samples are labeled into two classes, one for MGUS/SMM/NPC and the other for MM. Due to the significant unbalance of the samples between the two classes, only 156 samples are randomly selected from the 559 MM samples. The number 156 has been chosen as

a compromise to take as many samples as possible from MM without significant unbalance between the two classes. Furthermore, only $D=200$ features with the largest variances across samples are selected from the total 54 613 probe sets. It is advantageous to limit ourselves to the 200 features with the largest variances, because these are more likely to reveal class discrimination and feature selection tends to perform poorly for very large numbers of features when samples are small (Sima and Dougherty, 2006a). Here we must put in a word of caution concerning the methodology. We are using feature variance to produce a set of 200 features to be taken as the full feature set for our performance analysis and will apply feature selection, classifier design and error estimation based on this set, including cross-validation. In practice, this approach would be unacceptable, because the actual dataset to which we are applying data-dependent feature selection is the full 54 613 probe sets. For instance, cross-validation would have to use the variance-based feature reduction from the full 54 613 on each fold, else it would be optimistically biased. But that is not our goal here. We are *a priori* assuming that there are only 200 features to which we will apply data analysis. In practice, such a scenario would occur if the reduction to 200 were based on prior biological knowledge.

- *Breast cancer data set*: data are from a microarray-based classification study that analyzes breast tumor samples from 295 patients (van de Vijver *et al.*, 2002). Using a previously established $D=70$-gene prognosis profile (van't Veer *et al.*, 2002), a prognosis signature based on gene expression is proposed in van de Vijver *et al.* (2002) that correlates well with patient survival data and other clinical measures. Of the 295 microarrays, 115 belong to the 'good-prognosis' class and 180 belong to the 'poor-prognosis' class. Referring to our cautionary comment regarding the multiple myeloma data, we note here that feature selection was used originally to obtain the 70 genes, but, again, from our performance perspective, that is not important for our analysis.

We consider sample size $n=50$ and $d=3$ features selected from the $D=200$ and $D=70$ features in the myeloma and breast cancer datasets, respectively, and LDA for classification. We repeatedly draw (stratified) $n=50$-point samples with replacement from the empirical distribution (full dataset) as training data with the remaining sample points held out for true error estimation in computing the RMS ($\varepsilon_0$ is still computed from the training data). The total number of repetitions is 200. The average true error and SD for the myeloma dataset are 0.2170 and 0.0309, respectively. For the breast cancer dataset, the average true error and SD are 0.2340 and 0.0362, respectively. Figure 4 shows the RMS for the two patient datasets. In both cases, $\hat{\varepsilon}_{\text{bolst}}^{D,\text{data}}$ performs significantly better. Owing to robustness of the optimal scaling factor, a coarse selection of expected errors and interpolation has proven sufficient.

To further demonstrate the effectiveness of Protocol 2, we have applied it to four models in Figure 1: models M1 and M2 with expected errors 0.20 and 0.35. The performance graphs corresponding to Figure 4 are provided in Figure 5. Of particular interest are the scaling factors produced by the protocol. The average scaling factors for the four models are given by: M1, $E[\varepsilon_n]=0.20$ – average scaling factor 1.10; M1, $E[\varepsilon_n]=0.35$ – average scaling factor 1.39; M2 $E[\varepsilon_n]=0.20$ – average scaling factor 1.09; and M2,

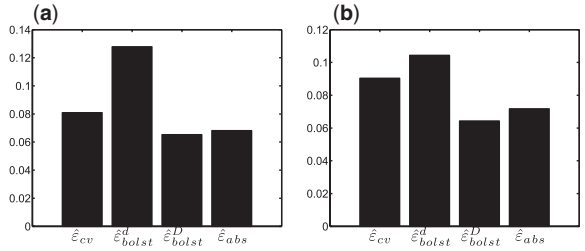


**Fig. 4.** RMS using LDA for (**a**) myeloma dataset, total feature size $D=200$ and (**b**) breast cancer dataset, total feature size $D=70$. For both datasets: sample size $n=50$ and selected feature size $d=3$.

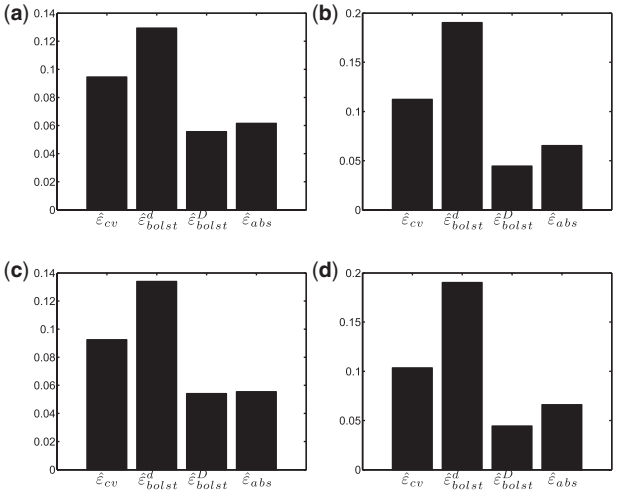


**Fig. 5.** RMS using LDA and protocol 2 for (**a**) M1, $E[\varepsilon_n]=0.20$, (**b**) M1, $E[\varepsilon_n]=0.35$, (**c**) M2, $E[\varepsilon_n]=0.20$, (**d**) M2, $E[\varepsilon_n]=0.35$. All with sample size $n=50$ and selected feature size $d=3$.

$E[\varepsilon_n]=0.35$ – average scaling factor 1.43. Referring to Figure 1, we see that all these averages are centered within the range of scaling factors where optimal bolstering outperforms $\hat{\varepsilon}_{\text{abs}}$.

### 3.1 Robustness to non-Gaussian data

Although $k_D^{\min}$ is derived with Gaussian models, it is robust enough for models where this assumption is violated, as with the patient data, where the underlying distribution is almost certainly not Gaussian. To further investigate this issue, we take the model $M2$ in Section 2.3, but perturb the skewness and kurtosis of the class at the origin to obtain a Pearson system (Elderton and Johnson, 1969). Figure 6 shows the eight different distributions in the Pearson system with varying skewness and kurtosis. For the resulting model $\mathcal{M}^p$ and each skewness and kurtosis combination, where valid, we do the following:

(1) Generate a sample set $S_n^D$ of size $n=50$ and a total of $D=200$ features from the model $\mathcal{M}^p$.
(2) Feature select a size-$d=3$ feature set $A$, resulting in a reduced dimension sample set $S_n^d$.
(3) Design a classifier $\psi_n$ for $S_n^d$ using LDA.

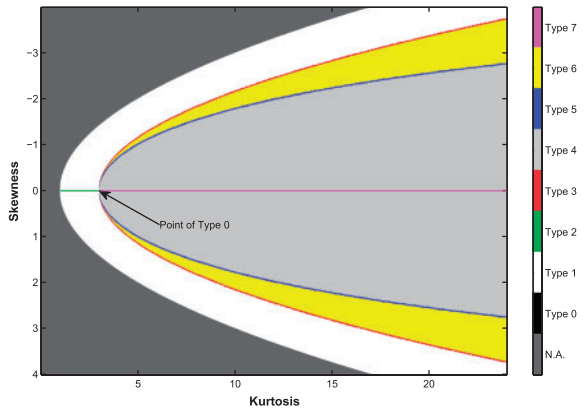


**Fig. 6.** The plane of (skewness, kurtosis) pairs and their corresponding probability distributions in a Pearson system. In particular, Type 0 (Gaussian distribution) has a skewness of 0 and kurtosis of 3, which is represented by a single point on the plane. There are eight different types of distributions in a Pearson system.

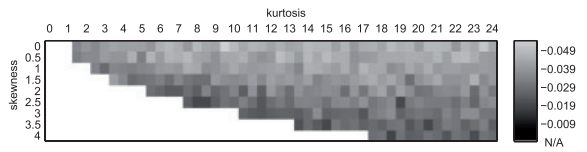


**Fig. 7.** Heatmap for RMS of $\hat{\varepsilon}^{D,\text{data}}_{\text{bolst}}$ minus RMS of $\hat{\varepsilon}_{cv}$ for different skewness and kurtosis. All values are negative.

(4) Compute the true error $\varepsilon_n$ using the underlying distribution of the model $\mathcal{M}^p$.

(5) Compute the 10-fold cross-validation error $\hat{\varepsilon}_{\text{cv}}$.

(6) Compute $\hat{\varepsilon}^{D,\text{data}}_{\text{bolst}}$ using $k_D^{\min}$ from the previous section.

(7) Calculate RMS for $\hat{\varepsilon}_{\text{cv}}$ and $\hat{\varepsilon}^{D,\text{data}}_{\text{bolst}}$ by repeating Steps 1 through 6 a number $N = 400$ of times.

Figure 7 shows the values of RMS for $\hat{\varepsilon}^{D,\text{data}}_{\text{bolst}}$ minus the RMS for $\hat{\varepsilon}_{cv}$ for different skewness and kurtosis in a heatmap. Due to symmetry, only positive skewness is shown. In all cases, $\hat{\varepsilon}^{D,\text{data}}_{\text{bolst}}$ is superior to $\hat{\varepsilon}_{cv}$.

### 3.2 Concluding remarks

We have derived an optimal kernel scaling factor that can be used for bolstered error estimation in high feature dimensions. This bolstered error estimator achieves a significant RMS improvement over cross-validation when samples are small, with continued, albeit smaller, performance improvement over the adjusted bootstrap. This superior performance is robust over a wide range of models. Hence, we have been able to incorporate optimality criteria from across a collection of families to arrive at suitable bolstering kernels for practical situations, thereby facilitating its use in applications like classification of genomic data when samples are small.

## ACKNOWLEDGEMENTS

We would also like to thank the High-Performance Biocomputing Center of TGen for providing the clustered computing resources used in this study; this includes the Saguaro-2 cluster supercomputer, a collaborative effort between TGen and the ASU Fulton High Performance Computing Initiative.

*Funding*: National Science Foundation (CCF-0634794 and CCF-0845407); National Institutes of Health grant (1S10RR025056-01) to Saguaro-2 cluster, in part.

*Conflict of Interest*: none declared.

## REFERENCES

Boulesteix,A.-L. (2010) Over-optimism in bioinformatics research. *Bioinformatics*, **26**, 437–439.

Boulesteix,A.-L. and Strobl,C. (2009) Optimal classifier selection and negative bias in error rate estimation: an empirical study on high-dimensional prediction. *BMC Med. Rese. Methodol.*, **9**, 85.

Braga-Neto,U. (2007) Fads and fallacies in the name of small-sample microarray classification. *IEEE Sig. Proc. Mag.*, **24**, 91–99.

Braga-Neto,U. and Dougherty,E. (2004a) Bolstered error estimation. *Pattern Recognit.*, **37**, 1267–1281.

Braga-Neto,U.M. and Dougherty,E.R. (2004b) Is cross-validation valid for small-sample microarray classification? *Bioinformatics*, **20**, 374–380.

Devroye,L. *et al.* (1996) *A Probabilistic Theory of Pattern Recognition*. Springer, New York.

Efron,B. (1983) Estimating the error rate of a prediction rule: Improvement on cross-validation. *J. Am. Stat. Assoc.*, **78**, 316–331.

Elderton,S.W. and Johnson,N. (1969) *Systems of Frequency Curves*. Cambridge University Press, Cambridge.

Grate,L. (2005) Many accurate small-discriminatory feature subsets exist in microarray transcript data: biomarker discovery. *BMC Bioinformatics*, **6**, 97.

Hanczar,B. *et al.* (2007) Decorrelation of the true and estimated classifier errors in high-dimensional settings. *EURASIP J. Bioinformatics Syst. Biol.*, **2007**, 1–12.

Hanczar,B. *et al.* (2010) Small-sample precision of roc-related estimates. *Bioinformatics*, **26**, 822–830.

Hua,J. *et al.* (2005) Optimal number of features as a function of sample size for various classification rules. *Bioinformatics*, **21**, 1509–1515.

Hua,J. *et al.* (2009) Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recognit.*, **42**, 409–424.

Huynh,K. *et al.* (2007) Improved bolstering error estimation for gene ranking. In *Proceedings of the IEEE EMBS*. Lyon, France.

Jain,A. and Zongker,D. (1997) Feature selection: evaluation, application, and small sample performance. *IEEE Trans. Pattern Anal. Mach. Intell.*, **19**, 153–158.

Jelizarow,M. *et al.* (2010) Over-optimism in bioinformatics: an illustration. *Bioinformatics*, **26**, 1990–1998.

Jiang,W. and Simon,R. (2007) A comparison of bootstrap methods and an adjusted bootstrap approach for estimating the prediction error in microarray classification. *Stat. Med.*, **26**, 5320–5334.

Kudo,M. and Sklansky,J. (2000) Comparison of algorithms that select features for pattern classifiers. *Pattern Recognit.*, **33**, 25–41.

Molinaro,A.M. *et al.* (2005) Prediction error estimation: a comparison of resampling methods. *Bioinformatics*, **21**, 3301–3307.

Pudil,P. *et al.* (1994) Floating search methods in feature-selection. *Pattern Recognit. Lett.*, **15**, 1119–1125.

Rocke,D.M. *et al.* (2009) Papers on normalization, variable selection, classification or clustering of microarray data. *Bioinformatics*, **25**, 701–702.

Sima,C. and Dougherty,E. (2006a) What should be expected from feature selection in small-sample settings. *Bioinformatics*, **22**, 2430–2436.

Sima,C. and Dougherty,E.R. (2006b) Optimal convex error estimators for classification. *Pattern Recognit.*, **39**, 1763–1780.

Sima,C. *et al.* (2005a) Impact of error estimation on feature selection. *Pattern Recognit.*, **38**, 2472–2482.

Sima,C. *et al.* (2005b) Superior feature-set ranking for small samples using bolstered error estimation. *Bioinformatics*, **21**, 1046–1054.

van de Vijver,M.J. *et al.* (2002) A gene-expression signature as a predictor of survival in breast cancer. *N. Engl. J. Med.*, **347**, 1999–2009.

van't Veer,L.J. *et al.* (2002) Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, **415**, 530–536.

Vu,T. and Braga-Neto,U. (2008) Preliminary study on bolstered error estimation in high-dimensional spaces. In *Proceedings of the IEEE GENSIPS*. Phoenix, AZ.

Yousefi,M.R. *et al.* (2010) Reporting bias when using real data sets to analyze classification performance. *Bioinformatics*, **26**, 68–76.

Zhan,F. *et al.* (2006) The molecular classification of multiple myeloma. *Blood*, **108**, 2020–2028.

Zhao,C. *et al.* (2010) Characterization of the effectiveness of reporting lists of small feature sets relative to the accuracy of the prior biological knowledge. *Cancer Inform.*, **9**, 49–60.