Available online at www.sciencedirect.com

SCIENCE DIRECT®

Pattern Recognition 38 (2005) 2472–2482

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Impact of error estimation on feature selection

Chao Sima[a], Sanju Attoor[a], Ulisses Brag-Neto[b], James Lowey[c], Edward Suh[c], Edward R. Dougherty[d,*]

[a]*Department of Electrical Engineering, Texas A&M University, College Station, TX 77843*
[b]*Centro de Pesquisas Aggeu Magalhes - CPqAM/Fiocruz, Cidade Universitria, Recife, Brazil*
[c]*Computational Biology Division, Translational Genomics Research Institute, Phoenix, Arizona*
[d]*Department of Electrical Engineering, Texas A&M University, College Station, TX 77843 and Department of Pathology, University of Texas M.D. Anderson Cancer Center, Houston, TX 77030, USA*

Received 14 December 2004; accepted 28 March 2005

Abstract

Given a large set of potential features, it is usually necessary to find a small subset with which to classify. The task of finding an optimal feature set is inherently combinatoric and therefore suboptimal algorithms are typically used to find feature sets. If feature selection is based directly on classification error, then a feature-selection algorithm must base its decision on error estimates. This paper addresses the impact of error estimation on feature selection using two performance measures: comparison of the true error of the optimal feature set with the true error of the feature set found by a feature-selection algorithm, and the number of features among the truly optimal feature set that appear in the feature set found by the algorithm. The study considers seven error estimators applied to three standard suboptimal feature-selection algorithms and exhaustive search, and it considers three different feature-label model distributions. It draws two conclusions for the cases considered: (1) depending on the sample size and the classification rule, feature-selection algorithms can produce feature sets whose corresponding classifiers possess errors far in excess of the classifier corresponding to the optimal feature set; and (2) for small samples, differences in performances among the feature-selection algorithms are less significant than performance differences among the error estimators used to implement the algorithms. Moreover, keeping in mind that results depend on the particular classifier-distribution pair, for the error estimators considered in this study, bootstrap and bolstered resubstitution usually outperform cross-validation, and bolstered resubstitution usually performs as well as or better than bootstrap.

Keywords: Classification; Feature selection; Error estimation

1. Introduction

Given a large set of potential features for classification, it is necessary to find a small subset with which to classify. The problem is statistically inherent in classification because typically (but not universally), the true error of a designed classifier will fall with use of more features and, after some optimal number of features for a given sample size, begin to rise. For small samples the optimal number can be very small. The task of finding an optimal feature set is inherently combinatoric. According to a classical theorem, to be assured of finding the optimal feature set of a given size, all feature subsets of that size must be checked unless there is distributional knowledge that mitigates the search requirement, a mitigating condition not occurring in practice [1]. There are various methods of choosing feature sets, the

* Corresponding author. Tel.: +1 979 862 8896; fax: +1 979 845 6259.
E-mail address: edward@ee.tamu.edu (E. Dougherty).
URL: http://ee.tamu.edu/∼edward/err_est_fs.

intent being to choose a set of features that provides good classification. When there is a large number of potential features for classification, feature selection is problematic and the best method to use depends on the circumstances. Evaluation of methods is generally comparative and based on simulations [2,3].

If feature selection is based directly on classification error, and not on some auxiliary measure such as correlation, then an algorithm searching for a good feature set must base its decision on estimates of the error. If there is a large data set, then one can obtain good error estimates; however, if the sample is small, then error estimation is problematic and the performance of the feature-selection algorithm will be impacted by the performance of the error estimator. As will be demonstrated in this paper, the lack of optimality with feature selection can be impacted to a greater extent by error estimation than by the choice of feature-selection algorithm, and performance of a particular feature-selection algorithm is affected by the choice of error-estimation rule.

The role of error estimation in the choice of feature sets for small samples has previously been addressed relative to the absolute ranking of feature sets [4,5]. In these studies, based on an exhaustive search, the classifiers corresponding to all feature sets of a given size were found, their true errors and their estimated errors based on various estimation rules were calculated, and the feature sets were ranked based on their true and estimated errors. The key issue was ranking order. It was seen that certain error-estimation rules gave better feature-set ranking, depending on the class-conditional distributions, classification rule, and sample size.

This paper does not concern ranking; rather, it concerns the performance of feature-selection algorithms relative to their purpose of finding good feature sets—in particular, the impact of error estimation in this regard. Thus, we employ two measures of merit: (1) we will compare the true error of the optimal feature set with the true error of the feature set found by a feature-selection algorithm; and (2) we will see how many of the features among the truly optimal feature set appear in the feature set found by the algorithm. In all cases we will average the results over a large collection of samples, and we will categorize the results by feature-selection algorithm, error-estimation rule, classification rule, class-conditional distributions, and sample size. Owing to the large number of simulations and computations, the project has been carried out on a massively parallel Beowulf cluster, and owing to the large number of results, a companion website is provided to augment the results reported in the paper.

To a great extent, this study has been motivated by the large number of papers in recent years dealing with phenotype classification based on expression microarrays. Perhaps the most salient characteristic of expression-based phenotype classification using microarray data is the vast number of potential features (genes) in comparison to the small number of data points (microarrays), and the effect this disparity has on classifier design, error estimation, and feature selection [6]. Whereas there are typically thousands of genes on a microarray, laboratory costs and availability of patient tissue stringently limits the number of microarrays. The following sample sizes for cancer studies are indicative of the commonplace paucity of data points: cutaneous melanoma, 31 [7]; leukemia, 37 [8]; acute leukemia, 38 [9]; breast cancer, 38 [10], follicular lymphoma, 24 [11]; uveal melanoma, 20 [12], glioma, 50 (but only 21 classic tumors used for class prediction) [13]; ovarian carcinoma, 44 [14]; lymphoma, 47 [15]; and glioma, 25 [16]. Even though sample sizes are slowly growing as costs decline, availability of tissue will continue to limit sample sizes. Our simulation analyses reflect this limitation by considering sample sizes of 30, 50, 70, and 90.

2. Experimental set-up

For simulation studies, we consider 3 models. Model 1 is a 2-class Gaussian model, with the classes equally likely and the class-conditional densities being spherical unit variance Gaussians. The class means are located at δa and $-\delta a$, where $\delta > 0$ is a separation parameter and $a = (a_1, a_2, \ldots, a_n)$ is a parameter vector with $\|a\| = 1$. It is well-known that the Bayes classifier is a hyperplane perpendicular to the axis joining the means, with Bayes error $\varepsilon_{BAYES} = 1 - \Phi(\delta)$, where Φ is the standard normal cumulative distribution function. Since $\delta = \Phi^{-1}(1 - \varepsilon_{BAYES})$, one can find δ for a prescribed Bayes error. In our experiments, we choose δ so that the Bayes error is 0.1.

If a subset L of the original features is selected, then again one has a standard Gaussian model, but now the separation between the classes is a function of which features are selected. The Bayes error is a function of both the separation and the model parameters. To be exact, $\varepsilon^L_{BAYES} = 1 - \Phi(\delta\sqrt{\sum_{k\in L} a_k^2})$. Thus for a given number of selected features, the ones corresponding to parameters with larger amplitude will have smaller ε^L_{BAYES}. The parameter vector $a = (a_1, a_2, \ldots, a_n)$ is picked from a sigmoidal distribution in order to favor a few of the feature sets.

Model 2 is similar to Model 1, but instead of both covariance matrices for the class-conditional densities being $\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix, we let them be $\sigma_1\boldsymbol{I}$ and $\sigma_2\boldsymbol{I}$ for class 1 and class 2, respectively, with $\sigma_1 \neq \sigma_2$. Since there is no closed-form formula for Bayes error in this model, we resort to Monte Carlo methods for computing the separation parameter δ for the desired Bayes errors. We let $\sigma_1 = 1$ and $\sigma_2 = 1.5$ in our experiments, and choose δ so that Bayes error using all the features equals 0.03 or 0.04.

Model 3 is also an equally likely 2-class Gaussian model, with means located at $(\delta, \delta, \ldots, \delta)$ and $(-\delta, -\delta, \ldots, -\delta)$.

Here we employ a block-like covariance matrix structure

$$\mathbf{\Sigma} = \sigma^2 \left[\begin{array}{cccccccccccccccccc}
1 &&&&&&&&&&&&&&&&& \\
& 1 && \rho &&&&&&&&&&&&&& \\
&& \cdot &&&&& 0 &&& \cdot & \cdot & \cdot &&& 0 && \\
& \rho && \cdot &&&&&&&&&&&&&& \\
&&&& 1 &&&&&&&&&&&&& \\
&&&&& 1 &&&&&&&&&&&& \\
&&&&&& 1 && \rho &&&&&&&&& \\
&& 0 &&&&& \cdot &&& \cdot & \cdot & \cdot &&& 0 && \\
&&&&&& \rho && \cdot &&&&&&&&& \\
&&&&&&&&& 1 &&&&&&&& \\
&& \cdot &&&&& \cdot &&& \cdot &&&&& \cdot && \\
&& \cdot &&&&& \cdot &&&& \cdot &&&& \cdot && \\
&& \cdot &&&&& \cdot &&&&& \cdot &&& \cdot && \\
&&&&&&&&&&&&& 1 &&&& \\
&&&&&&&&&&&&&& 1 && \rho & \\
&& 0 &&&&& 0 &&& \cdot & \cdot & \cdot &&& \cdot && \\
&&&&&&&&&&&&&& \rho && \cdot & \\
&&&&&&&&&&&&&&&&& 1
\end{array}\right],$$

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 ρ among each other. In our studies, we let both covariance matrices be $\mathbf{\Sigma}$ with $G = 5$, $\sigma = 1$ and $\rho = 0.5$, and choose δ so that the Bayes error is 0.05 (again by Monte Carlo methods). Notice if G equals the total number of features, then all features are uncorrelated and this model is similar to Model 1.

The experiments are set up in the following manner: 200 independent samples of size S with N features are generated, and we select the K features, using exhaustive search (exhst), sequential forwarding search (SFS) [17], sequential forwarding floating search (SFFS) [17], and the improved branch-and-bound search method (enhBB) [18]. The criterion function is the correct recognition rate, defined as 1 minus the estimated error. We consider three classification rules: linear discriminate analysis (LDA), 3-nearest neighbor (3NN), and classification and regression trees (CART). We apply 7 error estimation methods: true error (true), resubstitution (resub), leave-one-out (loo), 5-fold cross validation (cv5), 0.632 bootstrap (bstrap), bolstered resubstitution (blstr) and semi-bolstered resubstitution (semib). (For a review of these methods, please refer to Appendix A.) By "true error" we mean the computed error for the designed classifier using the known underlying distribution of synthetic data, not the Bayes error, which the "true error" can achieve only when the designed classifier is optimal.

K features are found at the end of the feature search for each sample, and two performance measures, $T1$ and $T2$, are computed. $T1$ is the average true error over the 200 samples. Except in the case of Model 3, $T2$ is defined as the average, over the samples, of the number of common features when we compare the K features found by the feature selection to the K features found by exhaustive search and true error estimation using the same classifier.

The second measure for Model 3, denoted by $\widehat{T2}$, is computed differently. Here features within the same group are equivalent in the sense that, with all other features fixed, choosing any feature in the group should give the same classifying power. Furthermore, the groups are equivalent between each other in the sense that, choosing a feature from group i gives the same classifying power as choosing one from another group j, given the other features are fixed and not coming from group i or j. Thus, the key issue is the number of distinct groups represented by the K features found by feature selection. $\widehat{T2}$ is the average, over the 200 samples, of the number of represented groups.

We consider three (total features, selected features) pairs:$(N, K) = (20, 4), (20, 5)$, and $(25, 4)$. For each pair, we repeat the experiment for $S = 30, 50, 70$ and 90, and the performance measures $T1$ and $T2$ (or $\widehat{T2}$, in the case of Model 3) are computed. The experiments are summarized in Table 1.

3. Experimental results

Selected results from the experiments are shown in Appendix C, Tables 2–4 . A complete description of the experimental results is given on the companion website. To use the tables, suppose we are interested in the performance for the branch-and-bound method for selecting 4 features out of 20, when the sample size is 30, under the LDA rule, in Exp 2. Then we look at Table 3, under column LDA/enhBB and row "size 30". There we find the results for each of the seven error estimation methods.

Table 1
Experiments setup

Model	Exp 1	Exp 2	Exp 3	Exp4
	Model 1	Model 2	Model 2	Model 3
Bayes error	0.10	0.03	0.04	0.05
Classification rule	LDA, 3NN and CART			
Feature selection algorithm	exhst, SFS, SFFS and enhBB			
Error estimation method	true, resub, loo, cv5, bstrap, blstr, and semib			
Sample size	30 , 50 , 70 and 90			
(N, K) pair	(20, 4), (20, 5) and (25, 4)			
Performance measure	$T1$ and $T2$			$T1$ and $\widehat{T2}$

3.1. *Significance of error estimation relative to feature selection*

The most important conclusion we draw from the experiments is that, for small samples, differences in performances among the feature selection algorithms are much less significant than the effects of error estimation. Except for several cases in which branch-and-bound performs very badly (see Appendix B), performances across different feature-selection algorithms are mostly comparable, including exhaustive search. We note three points in this regard.

SFFS generally outperforms SFS, which outperforms enhBB when doing feature selection using the true error, but this is not necessarily the case when using error estimation. For instance, when using 3NN, SFFS outperforms enhBB when true error is used; however, if resubstitution is used, enhBB outperforms SFFS, and if cross-validation or bootstrap are used, the SFFS and enhBB perform essentially the same.

For LDA, SFFS and SFS perform almost equivalently to exhaustive search when the true error is used, but they degrade relative to exhaustive search when error estimation is employed, SFS doing worse than SFFS, and the latter degrading little in relation to exhaustive search when using bootstrap or bolstering.

The choice of error estimator for feature selection can make more of a difference than choice of feature selection algorithm in terms of the true error of the designed classifier. Consider the following observations. Referring to Table 2 (Exp 1), for LDA and $S = 50$, if leave-one-out is used along with a full search, then the error of the designed classifier is 0.2241, but if bolstered resubstitution is used, then the worst result occurs with SFS, and this classifier has error 0.2172, better than an exhaustive search with leave-one-out (and better than an exhaustive search with 5-fold cross-validation). This is for selecting 4 features out of 20. When selecting 5 features out of 20 for LDA and $S = 50$ (see companion website), if leave-one-out is used along with an exhaustive search, then the error of the designed classifier is 0.2104, but if bolstered resubstitution is used, then the worst result occurs with SFS, and this classifier has error 0.1962, again better than an exhaustive search with leave-one-out (and better than an exhaustive search with both 5-fold cross-validation and bootstrap). Similar phenomena occur throughout the results. In particular, there are many cases where bolstered resubstitution and bootstrap yield better feature sets using SFFS than the feature sets obtained by cross-validation (both loo and cv5) using an exhaustive search. For instance, for all cases in Tables 2 and 3, bolstered resubstitution and bootstrap yield better $T1$ values using SFFS than cross-validation using an exhaustive search, with bolstered resubstitution outperforming bootstrap for LDA and CART in all cases in both tables. Moreover, bolstered resubstitution yields better $T2$ values using SFFS than cross-validation using an exhaustive search for all cases in Tables 2 and 3.

3.2. *Some general trends*

Besides observations regarding the prominence of error-estimation choices relative to feature-selection choices, some general trends can be discerned. As would be expected, throughout the experimental results larger samples yield better performances of $T1$ and $T2$ ($\widehat{T2}$ in Exp 4). No matter which error estimation procedure is adopted, the results are much worse than using the true error for all feature selection methods, both for $T1$ and $T2$ ($\widehat{T2}$). The feature-selection algorithms perform better for the blocked covariance structure of Model 3 (Exp 4) than for Models 1 and 2. All feature-selection algorithms perform the worst for CART, and this is especially true fore small sample size ($S = 30$), no matter the error estimation method, including using the true underlying distribution. This suggests that one should avoid feature selection for complicated classification rules when only small samples are available.

3.3. *Comparison of error-estimation methods*

Consistent with the results reported in straight feature ranking [5], for feature selection, bootstrap and bolstered resubstitution usually outperform cross-validation, with bolstering usually performing as well as or better than bootstrap; however, we must take care and consider individual results,

Table 2
Selected performance measures results for Exp 1

		LDA				3NN				CART			
		exhst	SFS	SFFS	enhBB	exhst	SFS	SFFS	enhBB	exhst	SFS	SFFS	enhBB
T1 for $N = 20,\ K = 4,\ S = 30, 50, 70$													
Size	True	0.1667	0.1757	0.1737	0.2272	0.1867	0.1889	0.1882	0.2014	0.2421	0.2515	0.2503	0.2922
30	resub	0.2494	0.2594	0.2551	0.2492	0.3094	0.3015	0.3018	0.2993	0.3741	0.3658	0.3664	0.3468
	loo	0.2452	0.2670	0.2569	0.2794	0.2770	0.2753	0.2755	0.2768	0.3404	0.3353	0.3359	0.3384
	cv5	0.2490	0.2636	0.2558	0.2933	0.2682	0.2754	0.2851	0.2841	0.3275	0.3354	0.3289	0.3541
	bstrap	0.2418	0.2517	0.2347	0.2935	0.2608	0.2599	0.2613	0.2586	0.3174	0.3218	0.3214	0.3195
	blstr	0.2161	0.2325	0.2184	0.2488	0.2570	0.2595	0.2672	0.2696	0.3023	0.3042	0.3036	0.3408
	semib	0.2214	0.2367	0.2243	0.2421	0.2635	0.2668	0.2654	0.2822	0.2958	0.3016	0.3013	0.3344
Size	True	0.1683	0.1706	0.1699	0.1826	0.1954	0.1981	0.1972	0.2091	0.2390	0.2485	0.2466	0.2828
50	resub	0.2261	0.2440	0.2339	0.2211	0.2969	0.2951	0.2949	0.2877	0.3746	0.3674	0.3668	0.3245
	loo	0.2241	0.2431	0.2320	0.2397	0.2645	0.2614	0.2641	0.2669	0.3222	0.3320	0.3315	0.3286
	cv5	0.2240	0.2459	0.2365	0.2588	0.2585	0.2732	0.2652	0.2754	0.3160	0.3218	0.3186	0.3453
	bstrap	0.2164	0.2292	0.2178	0.2267	0.2511	0.2532	0.2573	0.2527	0.3085	0.3059	0.3055	0.3108
	blstr	0.1956	0.2172	0.1957	0.1981	0.2532	0.2531	0.2500	0.2533	0.2802	0.2893	0.2888	0.3329
	semib	0.2019	0.2199	0.2038	0.2031	0.2522	0.2565	0.2587	0.2678	0.2836	0.2891	0.2890	0.3258
Size	True	0.1654	0.1667	0.1660	0.1704	0.1929	0.1955	0.1945	0.2039	0.2321	0.2407	0.2394	0.2705
70	resub	0.2033	0.2251	0.2127	0.1975	0.2756	0.2743	0.2738	0.2764	0.3564	0.3481	0.3490	0.3122
	loo	0.2018	0.2263	0.2122	0.2103	0.2447	0.2487	0.2500	0.2552	0.3127	0.3261	0.3252	0.3120
	cv5	0.2040	0.2226	0.2099	0.2286	0.2467	0.2546	0.2519	0.2631	0.3037	0.3049	0.3087	0.3374
	bstrap	0.1941	0.2149	0.1963	0.1998	0.2379	0.2416	0.2415	0.2394	0.2872	0.2889	0.2905	0.2956
	blstr	0.1806	0.2081	0.1826	0.1795	0.2340	0.2326	0.2357	0.2403	0.2657	0.2724	0.2728	0.3232
	semib	0.1868	0.2115	0.1900	0.1866	0.2342	0.2350	0.2400	0.2543	0.2669	0.2707	0.2736	0.3150
T2 for $N = 20,\ K = 4,\ S = 30, 50, 70$													
Size	True	4.0000	3.0900	3.2900	1.9550	4.0000	3.4650	3.5300	3.0800	4.0000	1.9400	1.9100	1.3700
30	resub	1.5700	1.4400	1.4850	1.6700	1.4700	1.5500	1.5350	1.5950	0.9100	0.8500	0.8250	1.1300
	loo	1.6650	1.3650	1.4900	1.3050	1.8300	1.8400	1.8250	1.8450	0.9850	1.0350	1.0300	1.0200
	cv5	1.6000	1.3450	1.5100	1.0800	1.9050	1.8200	1.7450	1.7150	1.0650	1.1450	1.1100	0.8800
	bstrap	1.7850	1.5650	1.8650	1.1000	2.0700	2.0850	2.0500	2.0900	1.1550	1.1900	1.1450	1.1600
	blstr	2.1000	1.7850	2.0900	1.6750	2.1000	2.0500	2.0200	1.8950	1.2050	1.2700	1.2050	1.0950
	semib	2.0350	1.7250	1.9550	1.7400	2.0450	1.9600	1.9450	1.7700	1.3450	1.2500	1.2600	0.9600
Size	True	4.0000	3.2250	3.3500	2.6250	4.0000	3.3850	3.4850	3.0150	4.0000	2.2650	2.2800	1.6150
50	resub	1.7750	1.4700	1.6300	1.8450	1.6600	1.6800	1.6550	1.7000	0.9300	0.9500	0.9600	1.3950
	loo	1.7850	1.4650	1.6800	1.5950	1.9800	2.0700	2.0350	1.9750	1.2650	1.1550	1.1450	1.2500
	cv5	1.7950	1.3850	1.5950	1.2700	2.1800	1.9350	2.0150	1.9400	1.2750	1.1650	1.2550	1.0250
	bstrap	1.9200	1.6600	1.8900	1.7750	2.2900	2.2650	2.1400	2.2300	1.3800	1.4250	1.4100	1.3400
	blstr	2.3000	1.7950	2.2850	2.2300	2.2650	2.1850	2.2900	2.2200	1.6350	1.4950	1.4150	1.1650
	semib	2.1700	1.7700	2.0650	2.1850	2.2450	2.2000	2.1650	2.0450	1.6700	1.4750	1.5650	1.1500
Size	True	4.0000	3.2800	3.4400	2.8450	4.0000	3.3250	3.3350	3.0350	4.0000	2.5950	2.6150	1.9500
70	resub	2.0550	1.6350	1.8500	2.1400	1.8550	1.8050	1.8100	1.7750	1.1200	1.0950	1.1000	1.5350
	loo	2.0100	1.5750	1.8300	1.9350	2.2850	2.1100	2.1100	2.0450	1.4900	1.2150	1.2150	1.5250
	cv5	1.9450	1.5850	1.9400	1.6250	2.2250	2.0000	2.1000	1.9600	1.5000	1.5200	1.3700	1.2050
	bstrap	2.1750	1.7350	2.1400	2.1050	2.3900	2.3250	2.3450	2.3000	1.8650	1.7850	1.7200	1.6300
	blstr	2.4800	1.8550	2.4450	2.5400	2.4300	2.4300	2.3550	2.2700	1.9950	1.8300	1.8400	1.3150
	semib	2.3100	1.7800	2.2150	2.4200	2.4500	2.4100	2.2850	2.0500	2.0300	1.9200	1.8950	1.3850

Table 3
Selected performance measures results for Exp 2

		LDA				3NN				CART			
		exhst	SFS	SFFS	enhBB	exhst	SFS	SFFS	enhBB	exhst	SFS	SFFS	enhBB
T1 for $N = 20,\ K = 4,\ S = 30, 50, 70$													
Size	True	0.1440	0.1508	0.1494	0.2054	0.1525	0.1559	0.1549	0.1759	0.1848	0.2089	0.2046	0.2518
30	resub	0.2256	0.2387	0.2345	0.2262	0.2620	0.2667	0.2678	0.2543	0.3110	0.3101	0.3100	0.3047
	loo	0.2224	0.2403	0.2294	0.2534	0.2301	0.2351	0.2364	0.2444	0.2923	0.2888	0.2898	0.2908
	cv5	0.2289	0.2367	0.2304	0.2755	0.2298	0.2314	0.2375	0.2524	0.2823	0.2875	0.2846	0.2967
	bstrap	0.2190	0.2235	0.2129	0.2632	0.2216	0.2192	0.2201	0.2226	0.2731	0.2810	0.2779	0.2799
	blstr	0.1923	0.2053	0.1918	0.2236	0.2140	0.2241	0.2270	0.2347	0.2486	0.2600	0.2626	0.2857
	semib	0.1955	0.2151	0.2016	0.2210	0.2195	0.2228	0.2230	0.2451	0.2474	0.2658	0.2621	0.2920
Size	True	0.1407	0.1431	0.1425	0.1540	0.1484	0.1515	0.1504	0.1660	0.1749	0.1904	0.1878	0.2289
50	resub	0.1896	0.2069	0.1981	0.1849	0.2353	0.2326	0.2307	0.2367	0.3080	0.2966	0.2963	0.2648
	loo	0.1865	0.2083	0.1972	0.2034	0.2063	0.2147	0.2152	0.2222	0.2627	0.2712	0.2719	0.2672
	cv5	0.1907	0.2126	0.1991	0.2269	0.2026	0.2106	0.2124	0.2314	0.2555	0.2595	0.2632	0.2792
	bstrap	0.1802	0.1928	0.1825	0.1970	0.1956	0.1995	0.2026	0.2027	0.2434	0.2478	0.2500	0.2581
	blstr	0.1625	0.1830	0.1638	0.1663	0.1929	0.1970	0.1990	0.2103	0.2223	0.2295	0.2308	0.2722
	semib	0.1693	0.1893	0.1713	0.1707	0.1985	0.2024	0.2057	0.2193	0.2230	0.2296	0.2289	0.2699
Size	true	0.1388	0.1404	0.1397	0.1437	0.1457	0.1482	0.1471	0.1590	0.1710	0.1832	0.1815	0.2156
70	resub	0.1754	0.1992	0.1892	0.1752	0.2152	0.2210	0.2222	0.2140	0.2849	0.2792	0.2802	0.2494
	loo	0.1756	0.1951	0.1824	0.1847	0.1960	0.2007	0.2017	0.2052	0.2413	0.2463	0.2476	0.2511
	cv5	0.1750	0.1970	0.1881	0.1994	0.1961	0.1964	0.2001	0.2130	0.2333	0.2427	0.2444	0.2736
	bstrap	0.1680	0.1864	0.1692	0.1693	0.1847	0.1874	0.1891	0.1858	0.2271	0.2319	0.2310	0.2354
	blstr	0.1536	0.1769	0.1531	0.1495	0.1835	0.1854	0.1903	0.1947	0.2060	0.2161	0.2136	0.2632
	semib	0.1583	0.1812	0.1591	0.1549	0.1843	0.1880	0.1865	0.1999	0.2074	0.2151	0.2169	0.2573
T2 for $N = 20,\ K = 4,\ S = 30, 50, 70$													
Size	True	4.0000	3.1500	3.2650	1.9550	4.0000	3.3300	3.4400	2.7800	4.0000	1.7050	1.8350	1.1750
30	resub	1.5300	1.3600	1.4800	1.5650	1.4550	1.3400	1.3500	1.5450	0.8500	0.8800	0.9000	1.0050
	loo	1.6150	1.3000	1.4700	1.2400	1.7600	1.7000	1.6850	1.6200	0.9000	0.8750	0.8800	0.9650
	cv5	1.5100	1.3750	1.5100	0.9700	1.7700	1.7100	1.6600	1.5100	0.9350	0.8800	0.8950	0.9350
	bstrap	1.7150	1.5500	1.7650	1.1500	1.9350	1.9100	1.9700	1.9800	1.0200	0.8850	0.9200	0.9000
	blstr	2.1550	1.7900	2.1250	1.7300	1.9350	1.7950	1.7900	1.7150	1.2650	1.0200	1.0400	1.0300
	semib	2.0550	1.6250	1.9650	1.7250	1.9300	1.8400	1.8550	1.6650	1.2400	0.9950	1.0350	0.9750
Size	True	4.0000	3.2350	3.3200	2.7300	4.0000	3.3000	3.3600	2.8600	4.0000	2.1950	2.2850	1.6050
50	resub	1.9500	1.6100	1.7700	2.0250	1.6650	1.6550	1.6650	1.6050	1.0150	1.0000	1.0050	1.3450
	loo	1.9950	1.5500	1.7750	1.7700	1.9900	1.7750	1.7650	1.7000	1.2600	1.0800	1.0750	1.2300
	cv5	1.9350	1.5350	1.7450	1.3600	2.0700	1.9300	1.8850	1.6500	1.3100	1.2000	1.2150	1.0100
	bstrap	2.1050	1.7750	1.9850	1.8900	2.1550	2.1150	2.1050	2.0500	1.3900	1.3750	1.2800	1.2400
	blstr	2.3950	1.9650	2.3500	2.4350	2.1500	2.2150	2.1500	2.0000	1.5250	1.5250	1.5000	1.2300
	semib	2.2700	1.8450	2.2500	2.3150	2.1700	2.0150	1.9150	1.8350	1.5300	1.4300	1.4100	1.1600
Size	True	4.0000	3.3150	3.4350	2.9450	4.0000	3.2500	3.3550	2.8150	4.0000	2.3850	2.4500	1.7350
70	resub	2.0850	1.6900	1.8650	2.0650	1.7200	1.7050	1.6700	1.8350	1.0350	1.1450	1.1200	1.4000
	loo	2.0700	1.7750	1.9900	1.9650	2.0500	2.0400	2.0150	2.0250	1.4350	1.3700	1.3400	1.4200
	cv5	2.0800	1.6850	1.8800	1.6500	2.0650	2.0400	2.0200	1.9100	1.4700	1.4050	1.3850	1.0600
	bstrap	2.2250	1.8700	2.1900	2.2400	2.2600	2.2550	2.2500	2.2800	1.5400	1.5650	1.4650	1.5650
	blstr	2.5400	2.0300	2.5800	2.7000	2.3250	2.2900	2.1800	2.1800	1.9250	1.7300	1.7100	1.2700
	semib	2.4350	2.0300	2.4600	2.5150	2.3400	2.2200	2.2400	2.0100	1.8050	1.7200	1.7300	1.3250

Table 4
Selected performance measures results for Exp 4

		LDA				3NN				CART			
		exhst	SFS	SFFS	enhBB	exhst	SFS	SFFS	enhBB	exhst	SFS	SFFS	enhBB
T1 for $N = 20,\ K = 4,\ S = 30, 50, 70$													
Size	True	0.1211	0.1289	0.1266	0.1590	0.1322	0.1384	0.1376	0.1377	0.1987	0.2190	0.2160	0.2339
30	resub	0.1740	0.1741	0.1729	0.1648	0.2009	0.1900	0.1897	0.1850	0.2638	0.2695	0.2692	0.2594
	loo	0.1611	0.1731	0.1722	0.1825	0.1719	0.1735	0.1728	0.1781	0.2473	0.2583	0.2583	0.2574
	cv5	0.1643	0.1708	0.1685	0.1717	0.1731	0.1742	0.1776	0.1782	0.2487	0.2551	0.2551	0.2542
	bstrap	0.1513	0.1656	0.1508	0.1723	0.1621	0.1671	0.1685	0.1727	0.2470	0.2525	0.2516	0.2552
	blstr	0.1458	0.1634	0.1483	0.1691	0.1669	0.1737	0.1721	0.1732	0.2390	0.2465	0.2461	0.2542
	semib	0.1470	0.1628	0.1518	0.1656	0.1667	0.1737	0.1727	0.1712	0.2393	0.2470	0.2462	0.2515
Size	True	0.1205	0.1260	0.1249	0.1297	0.1335	0.1397	0.1381	0.1357	0.1872	0.2061	0.2044	0.2191
50	resub	0.1430	0.1553	0.1514	0.1439	0.1764	0.1786	0.1778	0.1767	0.2528	0.2503	0.2511	0.2364
	loo	0.1404	0.1560	0.1522	0.1559	0.1632	0.1680	0.1674	0.1681	0.2314	0.2392	0.2394	0.2388
	cv5	0.1409	0.1549	0.1493	0.1604	0.1630	0.1680	0.1654	0.1709	0.2329	0.2352	0.2371	0.2433
	bstrap	0.1339	0.1491	0.1399	0.1555	0.1574	0.1616	0.1618	0.1650	0.2292	0.2333	0.2340	0.2403
	blstr	0.1391	0.1492	0.1404	0.1456	0.1611	0.1649	0.1636	0.1638	0.2227	0.2258	0.2271	0.2390
	semib	0.1386	0.1521	0.1416	0.1453	0.1622	0.1647	0.1650	0.1687	0.2223	0.2281	0.2262	0.2382
Size	True	0.1199	0.1240	0.1236	0.1219	0.1340	0.1407	0.1395	0.1358	0.1809	0.1976	0.1966	0.2114
70	resub	0.1370	0.1446	0.1440	0.1355	0.1691	0.1755	0.1752	0.1741	0.2415	0.2402	0.2408	0.2298
	loo	0.1347	0.1454	0.1413	0.1442	0.1589	0.1680	0.1668	0.1651	0.2228	0.2302	0.2301	0.2357
	cv5	0.1381	0.1454	0.1431	0.1476	0.1593	0.1680	0.1644	0.1655	0.2240	0.2312	0.2299	0.2376
	bstrap	0.1346	0.1434	0.1353	0.1455	0.1552	0.1607	0.1610	0.1632	0.2192	0.2227	0.2229	0.2280
	blstr	0.1325	0.1432	0.1354	0.1407	0.1563	0.1603	0.1609	0.1604	0.2115	0.2159	0.2160	0.2346
	semib	0.1354	0.1431	0.1389	0.1414	0.1590	0.1619	0.1625	0.1652	0.2101	0.2179	0.2152	0.2355
T2 for $N = 20,\ K = 4,\ S = 30, 50, 70$													
Size	True	4.0000	3.9950	3.9900	3.3200	4.0000	3.9950	4.0000	3.9550	3.7350	3.3700	3.4300	3.1750
30	resub	2.9500	3.2250	2.9950	3.2100	2.7000	2.9800	2.9900	3.0400	2.1100	2.4600	2.4400	2.6150
	loo	3.3500	3.3200	3.1600	2.9550	3.2850	3.1750	3.1750	3.1200	2.8700	2.6300	2.6250	2.4450
	cv5	3.2650	3.3350	3.2400	3.1650	3.3000	3.2050	3.1450	3.1650	3.1050	2.9950	3.0000	3.1300
	bstrap	3.6600	3.5550	3.6200	3.1350	3.4700	3.3350	3.3050	3.1850	3.1100	3.1650	3.1150	3.1850
	blstr	3.4250	3.4550	3.3950	3.2250	3.3350	3.1600	3.2450	3.2250	3.1300	3.0900	3.0500	3.1500
	semib	3.4300	3.4300	3.2950	3.2850	3.4000	3.1500	3.2200	3.2750	3.1900	3.0250	3.0000	3.0900
Size	True	4.0000	4.0000	4.0000	3.8800	4.0000	4.0000	4.0000	3.9800	3.9150	3.5650	3.6350	3.3100
50	resub	3.5950	3.4450	3.4200	3.6000	3.3250	3.2100	3.2300	3.2100	2.5850	2.8550	2.8300	3.1100
	loo	3.6750	3.5150	3.4300	3.2800	3.5100	3.3400	3.3350	3.3050	3.3000	3.0350	3.0250	2.8450
	cv5	3.6250	3.5100	3.4750	3.2000	3.5150	3.3500	3.4250	3.2950	3.2950	3.1900	3.1850	3.1300
	bstrap	3.8350	3.6550	3.7000	3.2800	3.6450	3.5000	3.4900	3.3500	3.3350	3.3050	3.2700	3.1450
	blstr	3.5700	3.6550	3.5550	3.4050	3.4700	3.3950	3.4650	3.4300	3.1900	3.2200	3.2000	3.1700
	semib	3.6150	3.5350	3.5400	3.4450	3.4900	3.4300	3.3900	3.2700	3.2100	3.1050	3.1700	3.1700
Size	True	4.0000	4.0000	4.0000	3.9700	4.0000	4.0000	4.0000	3.9900	3.9650	3.7650	3.7250	3.4800
70	resub	3.7250	3.6250	3.5000	3.7550	3.5650	3.2900	3.3150	3.3250	2.8900	3.0300	3.0300	3.1650
	loo	3.7800	3.6100	3.6100	3.5100	3.6150	3.3300	3.3850	3.3800	3.3500	3.1800	3.1800	3.0700
	cv5	3.7350	3.5800	3.5150	3.4050	3.6700	3.3850	3.4700	3.4000	3.4450	3.2900	3.2800	3.1400
	bstrap	3.8450	3.7150	3.7700	3.4700	3.7500	3.5800	3.6150	3.4150	3.4500	3.3800	3.3600	3.2750
	blstr	3.7350	3.7200	3.6850	3.4550	3.6800	3.5700	3.6050	3.5250	3.3650	3.2100	3.2600	3.1050
	semib	3.7150	3.6800	3.5650	3.4350	3.5550	3.5200	3.5450	3.3850	3.4000	3.2500	3.2700	3.1750

because, specific results, and sometimes even trends in the results, must be examined for each particular classification-distribution combination. Considering Exp 1 in some detail, we note several phenomena.

For LDA and $S = 50$, with exhaustive search, SFS, or SFFS, resubstitution and cross-validation estimators perform about the same with respect to error. Bootstrap does better and bolstering does even better. However, for branch and bound, resubstitution and bootstrap both outperform cross-validation and are comparable. Moreover, the advantage of bolstering over bootstrap is even greater. Most of these observations are mirrored in the $T2$ statistic.

Now look at LDA and $S = 30$. The overall situation is different. For exhaustive search, resubstitution, cross-validation, and bootstrap all perform about the same, with bolstering substantially better. For SFS, there is a slight ordering, cross-validation being the worst, resubstitution being slightly better, bootstrap being still slightly better, and bolstering having a more substantial advantage over bootstrap. For SFFS, the results are similar to $S = 50$. For enhBB, there is generally worse performance, especially for the computationally intensive cv5 and bootstrap, with loo being slightly better. The striking difference is that resubstitution and bolstering perform about the same, with both being much better than bootstrap.

For 3NN and all sample sizes, there appears to be a more consistent trend based on both $T1$ and $T2$ than for LDA: bootstrap and bolstering perform about the same and are better than cross-validation, and resubstitution is by far the worst.

For CART and all sample sizes, we again witness the main trend from best to worst: bolstering, bootstrap, cross-validation, and finally resubstitution, which is far worse than any of the others.

4. Conclusion

Feature selection is unavoidable when there is a large number of features from which to choose. Our experiments indicate SFS and SFFS (and even branch and bound) can perform close to optimal (full search with true error) when the true error is employed in feature selection, but in practice knowledge of the true error is impossible. With large samples, most error estimation procedures work quite well so that one has good estimates of the true error; however, this is not the case with small samples, as are common in situations where data are expensive or difficult to obtain owing to a limitation on their availability, as is often the case with patient samples. Depending on the sample size and the classification rule, in particular its complexity, feature-selection algorithms can produce feature sets whose corresponding classifiers possess errors far in excess of the classifier corresponding to the optimal feature set. Moreover, and most importantly in application since one may have no alternative to a small sample, our experiments show that, for small samples, differences in performances among the feature selection algorithms are less significant than performance differences among the error estimators used to implement the feature-selection algorithm. Keeping in mind that specific results, and sometimes even trends in the results, depend on the particular classifier-distribution pair, for the error estimators considered in this study, bootstrap and bolstered resubstitution usually outperform cross-validation. Moreover, bolstered resubstitution usually performs as well as or better than bootstrap, and with much less computation time.

Appendix A

We shall present a short review of the error estimation methods we used in this paper.

A.1. Classifier error

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 $g : \{\mathbb{R}^p \times \{0, 1\}\}^n \times \mathbb{R}^p \to \{0, 1\}$. It maps S_n into the *designed classifier* $g(S_n, \cdot) : \mathbb{R}^p \to \{0, 1\}$. In fact, a classification rule is actually a collection of mappings, one for each n; however, we follow the usual practice of using a single operator notation g to represent all of the individual mappings. The *true error* of a designed classifier is its error rate given the training data set:

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

where $E_{\mathbf{F}}$ denotes expectation with respect to $\mathbf{F}$. The expected error rate over the data is given by $\varepsilon_n[g] = E_{\mathbf{F}_n} E_{\mathbf{F}}(|Y - g(S_n, X)|)$, where $\mathbf{F}_n$ is the joint distribution of the training data S_n. Were the underlying feature-label distribution $\mathbf{F}$ known, the true error could be computed exactly via (1). In practice, one must use 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.

A.2. Classical error estimation

The simplest way to estimate the error of a designed classifier 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 very fast, but is usually optimistic (i.e., biased low) as an estimator of $\varepsilon_n[g]$, sometimes very much so. Typically, the more complex the classifier is, the more optimistic resubstitution is, since complex classifiers tend to overfit the data, especially with small samples [19].

Cross-validation removes the optimism from resubstitution by employing test points not used in the design of the classifier. 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, $\hat{\varepsilon}_{\text{cvk}}$, is the overall proportion of error on all folds. 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 *leave-one-out estimator*, $\hat{\varepsilon}_{\text{loo}}$, in which a single observation is left out each time, corresponds to n-fold cross-validation. It 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 [20,21]. They can also be quite slow to compute when the number of folds or samples is large.

The bootstrap error estimation technique [22,23] 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. The basic *bootstrap zero estimator* [23] is written in terms of the empirical distribution as $\hat{\varepsilon}_0 = E_{\mathbf{F}^*}(|Y - g(S_n^*, X)|: (X, Y) \in S_n \backslash S_n^*)$. In practice, the expectation $E_{\mathbf{F}^*}$ has to be approximated by a Monte-Carlo estimate based on independent replicates S_n^{*b}, for $b=1, \ldots, B$. 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 *0.632 bootstrap estimator* [23], $\hat{\varepsilon}_{\text{b632}} = (1 - 0.632)\hat{\varepsilon}_{\text{resub}} + 0.632\hat{\varepsilon}_0$, tries to correct this bias by doing a weighted average of the bootstrap zero and resubstitution estimators. It has low variance, but can be extremely slow to compute. In addition, it can fail when resubstitution is too low-biased [20].

A.3. Bolstered error estimation

The resubstitution estimator is defined in terms of the empirical feature-label distribution F^* by $\hat{\varepsilon}_n^R = E_{F^*}[|Y - g(S_n, \mathbf{X})|]$. Relative to F^*, no distinction is made between points near or far from the decision boundary. If one spreads the probability mass at each point of the empirical distribution, 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. To take advantage of this observation, 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 $f^\diamond(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^n f_i^\diamond(\mathbf{x} - \mathbf{x}_i)$. The *bolstered resubstitution* estimator [24] is obtained by replacing F^* by $F^\diamond$ in the definition of

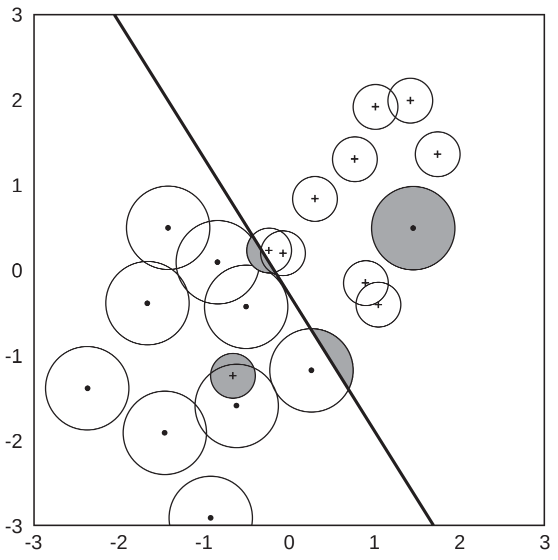

Fig. 1. Bolstered resubstitution for a linear classifier, assuming uniform circular bolstering kernels. The area of each shaded region divided by the area of the associated circle is the error contribution made by a point. The bolstered resubstitution error is the sum of all contributions divided by the number of points.

$\hat{\varepsilon}_n^R$ to obtain

$$\hat{\varepsilon}_n^{\diamond R} = E_{F^\diamond}[|Y - g(S_n, \mathbf{X})|]. \quad (2)$$

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

A computational expression for the bolstered resubstitution estimator is given by

$$\hat{\varepsilon}_n^{\diamond R} = \frac{1}{n}\sum_{i=1}^n \left(I_{y_i=0} \int_{A_1} f_i^\diamond(x - x_i)\,\mathrm{d}x + I_{y_i=1} \times \int_{A_0} f_i^\diamond(x - x_i)\,\mathrm{d}x \right), \quad (3)$$

where $A_j = \{x \mid g(S_n, x) = j\}$. The integrals are the error contributions made by the data points, according to whether $y_i = 0$ or $y_i = 1$. The bolstered resubstitution error estimate is equal to the sum of all error contributions divided by the number of points. 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. Experimentation indicates that a small number of Monte-Carlo samples is needed. Our simulations employ 10, the number used in [24]. Fig. 1 illustrates the situation where the bolstering kernels are given by uniform circular distributions and the classifier is linear. In this case, no Monte-Carlo computation is needed; the bolstered resubstitution error estimate is given in terms of the areas of the shaded regions.

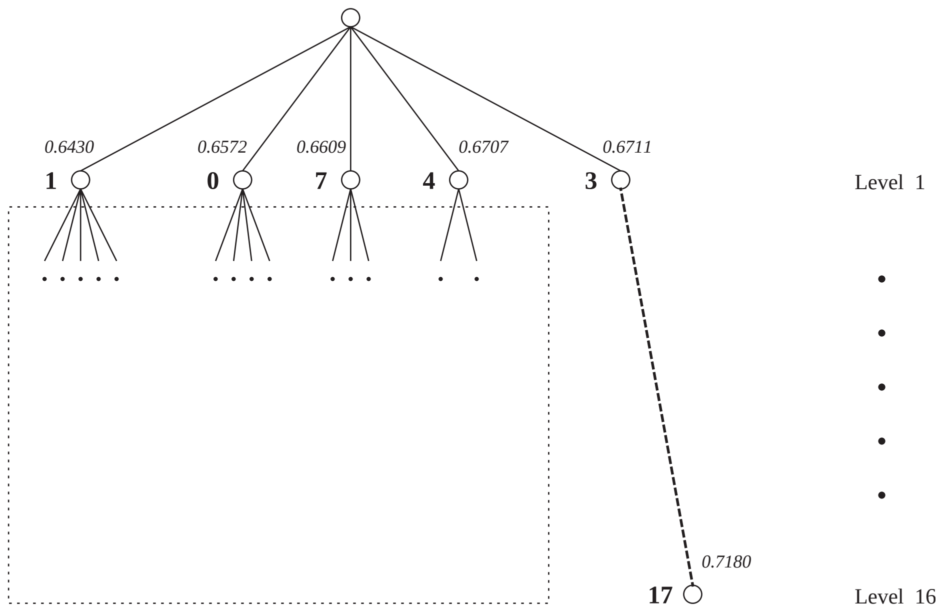

Fig. 2. A typical branch-and-bound tree searching path for $N = 20$, $K = 4$, $S = 30$ using "true error" estimation for LDA rule. (Taken from a simulation from Exp 1.)

When resubstitution is strongly low-biased, it may not be good to spread incorrectly classified data points, as that increases optimism of the error estimator. Bias is reduced by using no bolstering for incorrectly classified points. The result is the *semi-bolstered resubstitution* estimator [24].

Although more general bolstering kernels may be considered, in keeping with the principle of not making complicated inferences from a limited amount of data, we only consider zero-mean, spherical bolstering kernels $f_i^{\diamond}$, 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 employ a nonparametric sample-based method to choose these parameters that is applicable in small-sample settings [24]. The method is somewhat involved, so we leave its description to the cited paper. In this paper, a Gaussian kernel bolstering is used.

Appendix B

B.1. Branch-and-bound performance

We have seen that the branch-and-bound algorithm can perform much worse than SFS and SFFS for LDA with very small samples. To appreciate the source of this problem, we refer to a typical branch-and-bound search in Fig. 2. The $N = 20$ features are labeled $0, 1, \ldots, 19$. Marked at each node explored is the label number of the feature discarded at that point, along with the criterion function value evaluated [25,18]. Notice that the criterion function value at node 17 is higher than that at node 4. Thus, the search stops after merely one branch exploration. This gives us the best features as $0, 1, 4$, and 7, whereas the best features found by exhaustive search are $0, 1, 3$, and 15. The monotonicity assumption for branch and bound is severely violated here. The poor performance of enhBB is largely due to designing a classifier on a very small sample. At level 1 in Fig. 2, a 19-dimensional LDA classifier must be designed with only 30 data points, and the designed LDA classifier is likely to possess a large error.

Appendix C

C.1. Tables. Selected experimental results

Tables 2–4.

References

[1] T. Cover, J. Van Campenhout, On the possible orderings in the measurement selection problem, IEEE Trans. Syst. Man Cybernet. 7 (1977) 657–661.

[2] A.K. Jain, D. Zongker, Feature selection—evaluation application and small sample performance, IEEE Trans. Pattern Anal. Mach. Intell. 19 (1997) 153–158.

[3] M. Kudo, J. Sklansky, Comparison of algorithms that select features for pattern classifiers, Pattern Recognition 33 (2000) 25–41.

[4] U.M. Braga-Neto, R. Hashimoto, E.R. Dougherty, D.V. Nguyen, R.J. Carroll, Is cross-validation better than resubstitution for ranking genes?, Bioinformatics 20 (2004) 253–258.

[5] C. Sima, U.M. Braga-Neto, E.R. Dougherty, Superior feature-set ranking for small samples using bolstered error estimation, Bioinformatics 21 (2005) 1046–1054.

[6] E.R. Dougherty, Small sample issues for microarray-based classification, Comp. Funct. Genom. 2 (2001) 28–34.

[7] M. Bittner, P. Meltzer, Y. Chen, Y. Jiang, E. Seftor, M. Hendrix, M. Radmacher, R. Simon, Z. Yakhini, A. Ben-Dor, N. Sampas, E.R. Dougherty, E. Wang, F. Marincola, C. Gooden, J. Lueders, A. Glatfelter, P. Pollock, J. Carpten, E. Gillanders, D. Leja, K. Dietrich, C. Beaudry, M. Berens, D. Alberts, V. Sondak, N. Hayward, J. Trent, Molecular classification of cutaneous malignant melanoma by gene expression profiling, Nature 406 (2000) 536–540.

[8] S.A. Armstrong, J.E. Staunton, L.B. Silverman, R. Pieters, M.L. den Boer, M.D. Minden, S.E. Sallan, E.S. Lander, T.R. Golub, S.J. Korsmeyer, MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia, Nat. Genet. 30 (2002) 41–47.

[9] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, E.S. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 15 (286) (1999) 531–537.

[10] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson, J.R. Marks Jr., J.R. Nevins, Predicting the clinical status of human breast cancer by using gene expression profiles, Proc. Natl. Acad. Sci. 98 (2001) 11462–11467.

[11] S.P. Bohen, O.G. Troyanskaya, O. Alter, R. Warnke, D. Botstein, P.O. Brown, R. Levy, Variation in gene expression patterns in follicular lymphoma and the response to rituximab, Proc. Natl. Acad. Sci. 100 (4) (2003) 1926–1930.

[12] F. Tschentscher, J. Hüsing, T. Hölter, E. Kruse, I.G. Dresen, K.H. Jöckel, G. Anastassiou, H. Schilling, N. Bornfeld, B. Horsthemke, D.R. Lohmann, M. Zeschnigk, Tumor classification based on gene expression profiling shows that uveal melanomas with and without monosomy 3 represent two distinct entities, Cancer Res. 63 (2003) 2578–2584.

[13] C.L. Nutt, D.R. Mani, R.A. Betensky, P. Tamayo, J.G. Cairncross, C. Ladd, U. Pohl, C. Hartmann, M.E. McLaughlin, T.T. Batchelor, P.M. Black, A. von Deimling, S.L. Pomeroy, T.R. Golub, D.N. Louis, Gene expression-based classification of malignant gliomas correlates better with survival than histological classification, Cancer Res. 63 (2003) 1602–1607.

[14] M.E. Schaner, et al., Gene expression patterns in ovarian carcinomas, Mol. Biol. Cell 14 (11) (2003) 4376–4386.

[15] L. Li, C.R. Weinberg, T.A. Darden, L.G. Pedersen, Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method, Bioinformatics 17 (2001) 1131–1142.

[16] S. Kim, E.R. Dougherty, I. Shmulevich, K.R. Hess, S.R. Hamilton, J.M. Trent, G.N. Fuller, W. Zhang, Identification of combination gene sets for glioma classification, Mol. Cancer Ther. 1 (13) (2002) 1229–1236.

[17] P. Pudil, J. Novovičová, J. Kittler, Floating search methods in feature selection, Pattern Recognition Lett. 15 (1994) 1119–1125.

[18] K. Fukunaga, Introduction to Statistical Pattern Recognition, second ed., Academic Press, New York, 1990.

[19] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, NY, 1998.

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

[21] L. Devroye, L. Gyorfi, G. Lugosi, A Probabilistic Theory of Pattern Recognition, Springer, Berlin, New York, 1996.

[22] B. Efron, Bootstrap methods: another look at the jacknife, Ann. Statist. 7 (1979) 1–26.

[23] B. Efron, Estimating the error rate of a prediction rule: improvement on cross-validation, J. Am. Statist. Soc. 78 (1983) 316–331.

[24] U.M. Braga-Neto, E.R. Dougherty, Bolstered error estimation, Pattern Recognition 37 (2004) 1267–1281.

[25] P.M. Narendra, K. Fukunaga, A branch and bound algorithm for feature subset selection, IEEE Trans. Comput. 26 (1977) 917–922.