Australian & New Zealand Journal of Statistics

Aust. N. Z. J. Stat. **52**(4), 2010, 403–421 doi: 10.1111/j.1467-842X.2010.00594.x

SEQUENTIAL, BOTTOM-UP VARIABLE SELECTION FOR HIGH-DIMENSIONAL CLASSIFICATION

PETER HALL AND HUGH MILLER*

University of Melbourne

Summary

Most methods for variable selection work from the top down and steadily remove features until only a small number remain. They often rely on a predictive model, and there are usually significant disconnections in the sequence of methodologies that leads from the training samples to the choice of the predictor, then to variable selection, then to choice of a classifier, and finally to classification of a new data vector. In this paper we suggest a bottom-up approach that brings the choices of variable selector and classifier closer together, by basing the variable selector directly on the classifier, removing the need to involve predictive methods in the classification decision, and enabling the direct and transparent comparison of different classifiers in a given problem. Specifically, we suggest 'wrapper methods', determined by classifier type, for choosing variables that minimize the classification error rate. This approach is particularly useful for exploring relationships among the variables that are chosen for the classifier. It reveals which variables have a high degree of leverage for correct classification using different classifiers; it shows which variables operate in relative isolation, and which are important mainly in conjunction with others; it permits quantification of the authority with which variables are selected; and it generally leads to a reduced number of variables for classification, in comparison with alternative approaches based on prediction.

Key words: bootstrap; centroid-based classifier; classification; dimension reduction; lasso; linear model; median-based classifier; nearest neighbour classifier; regression; sequential variable selection; support vector machine; wrapper methods.

1. Introduction

Projection pursuit (see e.g. Friedman & Tukey 1974) tackles the problem of variable selection by seeking the linear combination of dimensions that is 'most interesting' in some sense, for example because it is the least Gaussian or has greatest entropy, among all projections that are orthogonal to those that have already been chosen. Other classical approaches, for example Asimov's (1985) grand tour and the N-land tool suggested by Ward, Leblanc & Tipnis (1994), also involve searching through many possibilities. This is often feasible when the dimension, p, is smaller than the sample size, n, and also on some occasions when n and p are broadly similar. However, in contemporary problems in which p is much larger than n, approaches of this type are ruled out for several reasons. One is their computational complexity. For example, even if each coefficient in a linear combination can take only two values, searching for the most appropriate one among all possibilities involves $O(2^p)$ calculations if each combination has to be explored. This is infeasible when, as is commonly the case today, p is in the thousands or tens of thousands.

*Author to whom correspondence should be addressed.
Department of Mathematics and Statistics, University of Melbourne, Melbourne, VIC 3010, Australia.
e-mail: h.miller@ms.unimelb.edu.au; halpstat@ms.unimelb.edu.au

These considerations motivate alternative approaches for solving contemporary high-dimensional classification problems. These include methods based on linear prediction, for example the lasso (Tibshirani 1996) or the Dantzig selector (Candes & Tao 2007), distance-based methods such as the support vector machine and centroid classifiers, and also techniques that involve ranking a relevance measure for each of the p components (see e.g. Fan & Lv 2008; Hall & Miller 2009). Algorithms such as these require at most $O(p \log p)$ operations, in terms of their dependence on p, and so are feasible even in many ultra-high-dimensional settings.

Prediction-based approaches are generally top-down, in that they start with the full p-dimensional problem and successively reduce the dimension. They and ranking-based methods usually do not take into account the sort of classifier that will be used. For instance, the set of components that minimize the error for one classifier might be different from that which is optimal for another, but that fact will typically not influence the variable selection step. More generally, the methods discussed above are relatively insensitive to interactions.

An alternative, bottom-up approach involves sequentially and explicitly building a model. Generally this is done by some form of forward stage-wise selection in which variables are sequentially added to the model according to which of the variables best improves an objective function. Such approaches are termed 'wrapper' methods when applied to genetic microarray datasets. There are clear advantages to this recursive approach. In particular, it addresses all potential arrangements (e.g. permutations) of the vector components, but requires only $O(p)$ calculations. It is highly adaptive to the classifier type and places no restriction on the nature of interactions among components that can be permitted. Indeed, in this respect it merely reflects the classifier; if the latter is responsive to highly nonlinear combinations of vector components then the recursive variable selector is too.

In this paper we propose approaches of this type. Their main feature is that they explicitly target the leave-one-out misclassification rate, which leads to more robust variable selection and heightened protection against overfitting. Our theoretical results demonstrate that these methods produce good asymptotic performance, even in very high-dimensional situations. We also investigate bootstrap tools that give insight into the stability of variable set selections. Furthermore, we demonstrate the use of a double layer of cross-validation to produce more reliable accuracy rates. The context of our work is very much that of high-dimensional data, where p can be many orders of magnitude larger than n. In this setting, both the classifier methodology and the technical assumptions made in theoretical work are quite different from their counterparts in conventional fixed-p, large-n problems.

Knowing which components have the greatest influence on correct classification can greatly enhance scientific interpretation of the results of classification. A method that simply assigns new data to different populations is not nearly so useful. The approach that we suggest is inherently of the former type. It has few peers in terms of the explicitness with which it selects variables that have greatest leverage on the successful performance of a particular classifier. It also offers new opportunities for practitioners to compare classifiers, for a particular dataset and on the basis of the variables or features that they select. This includes comparing the emphases that different classifiers give to different variables.

On the basis of our numerical experiments, for simulated data as well as for several real datasets, methodologies based on the sequential application of the centroid-based classifier, or its analogue with authority measured in terms of the median (see Section 2.4 for definitions) give good performance in a wide range of settings. They are not always at the very top in our

comparisons, although in some instances (e.g. for the leukemia dataset studied in Section 3.2) they are; they are seldom far from the top (e.g. for the colon dataset addressed in Section 3.3 they are ranked 3 and 2, respectively, out of 9); and they are relatively fast to compute (see Section 3.6).

The literature on classification, particularly in relation to variable selection, is vast. Methods based on the linear model, in addition to those noted above, include the non-negative garotte (e.g. Breiman 1995; Gao 1998), soft thresholding (e.g. Donoho *et al.* 1995), and related techniques (e.g. Donoho & Huo 2001; Fan & Li 2001; Donoho & Elad 2003; Tropp 2005; Donoho 2006a,b). Duda, Hart & Stork (2001), Hastie, Tibshirani & Friedman (2001) and Shakhnarovich, Darrell & Indyk (2005) provide book-length treatments of classification and related problems. Dudoit, Fridlyand & Speed (2002) discuss the performance of various classifiers. Fields of application of high-dimensional classifiers are as diverse as image analysis (e.g. Cootes *et al.* 1993), forestry (e.g. Franco-Lopez, Ek, & Bauer 2001), speech recognition (e.g. Bilmes & Kirchhoff 2004) and chemometrics (e.g. Schoonover *et al.* 2003), and, of course, genomics (e.g. Moon *et al.* 2006; Clarke *et al.* 2008; Hua, Tembe & Dougherty 2009). Discussion of wrapper methods for microarray data include the review by Saeys, Inza & Larrañaga (2007), as well as the work of Xiong, Fang & Zhao (2001) and Inza *et al.* (2004). Our methodology relies on a highly multivariate form of cross-validation for model selection, and in more conventional problems that approach is addressed by Shao (1993) and references cited therein.

2. Model and methodology

2.1. Estimator of error rate

Assume that a population is a mixture of two sub-populations, Π_0 and Π_1. For a given individual from Π_j we observe a data pair (X, Y), where $X = (X^{(1)}, \ldots, X^{(p)})$ is a p-vector and $Y = j$ denotes the sub-population type. Suppose too that we have training data, in the form of random samples $\mathcal{S}_j = \{(X_{j1}, Y_{j1}), \ldots, (X_{jn_j}, Y_{jn_j})\}$, for $j = 0$ and 1, known to come from Π_j. In particular, $Y_{ji} = j$ for $1 \leqslant i \leqslant n_j$ and for $j = 0, 1$. Let $\mathcal{C}(X \,|\, k_1, \ldots, k_t)$ denote the result of applying a particular classifier $\mathcal{C}$ to a data vector X that has been dimension-reduced to just the components with distinct indices $k_1, \ldots, k_t$. That is, $\mathcal{C}(X \,|\, k_1, \ldots, k_t)$ denotes the sub-population type, either 0 or 1, to which the classifier assigns X. Our estimator of the error rate, computed for these indices and based on the training data, is

$$\widehat{\mathrm{err}}(k_1, \ldots, k_t) = \frac{\pi}{n_0} \sum_{i=1}^{n_0} I\{\mathcal{C}_{-0i}(X_{0i} \,|\, k_1, \ldots, k_t) = 1\} + \frac{1-\pi}{n_1} \sum_{i=1}^{n_1} I\{\mathcal{C}_{-1i}(X_{1i} \,|\, k_1, \ldots, k_t) = 0\}, \tag{1}$$

where π denotes the prior probability of sub-population Π_0, the notation $\mathcal{C}_{-ji}$ means that the classifier is constructed by omitting the ith observation from the training sample $\mathcal{S}_j$, and I denotes the indicator function; $I(\mathcal{E}) = 1$ if the event $\mathcal{E}$ holds, and $I(\mathcal{E}) = 0$ otherwise. Section 2.5 will define $\mathcal{C}(X \,|\, k_1, \ldots, k_t)$ and $\mathcal{C}_{-ji}(X_{ji} \,|\, k_1, \ldots, k_t)$ in the case of centroid-based classifiers, and show how these definitions are altered for other classifier types. In the

case of priors set equal to the observed frequencies in the observed data, (1) precisely equals the leave-one-out cross-validated error rate.

Under mild assumptions on the classifier, the estimator of the error rate at (1) converges to the true error rate,

$$\begin{aligned} \mathrm{err}(k_1, \ldots, k_t) = \pi \; & P\{\mathcal{C}(X_{01} \,|\, k_1, \ldots, k_t) = 1\} \\ & + (1 - \pi) \, P\{\mathcal{C}(X_{11} \,|\, k_1, \ldots, k_t) = 0\}, \end{aligned} \tag{2}$$

as n_0 and n_1 increase. This gives the approach a major advantage over other recursive methods, as it provides automatic protection against overfitting. While most other approaches quickly drive the error to zero in the training set, we shall see that the leave-one-out error plateau is comparable to the true error rate.

2.2. Algorithm

First we describe the initial step of the algorithm, where we select $k = k_1$ from among $1, \ldots, p$ to minimize $\widehat{\mathrm{err}}(k)$. At this point, and in similar situations below, the manner in which we deal with ties is important, as in high-dimensional problems it is often the case that k can take many more distinct values than $\widehat{\mathrm{err}}(k)$. We suggest determining the set $\mathcal{K}$ of values of k for which $\widehat{\mathrm{err}}(k)$ achieves its minimum, and choosing k_1 to be the element of $\mathcal{K}$ for which the classifier produces, in an average sense, the most authoritative classification, over all training data and incorporating prior probabilities where appropriate. For example, many classifiers assign a new data value X to Π_j on the basis of the sign of a score function S, computed from the training data. In particular, in the case of distance-based classifiers we can take $S(X)$ to be a function of the distance from X to $\mathcal{S}_1$, minus the same function of the distance from X to $\mathcal{S}_0$. Section 2.5 gives details in the case of the centroid, as well as other, classifiers. Then, choose k_1 to be the value of $k \in \mathcal{K}$ that maximizes the prior-weighted median, or mean, or a similar measure of 'average', of the values taken by $S(X)$ for variables X in the training data, based on only the kth component. In this paragraph and the next, $X \in \mathcal{S}_0 \cup \mathcal{S}_1$, and the function S itself actually depends on X, because it is computed from the data in $\mathcal{S}_0 \cup \mathcal{S}_1 \setminus \{X\}$. However, to avoid unwieldy notation we do not express this dependence explicitly.

Now we describe how to apply the algorithm recursively. Given distinct integers $k_1, \ldots, k_t$ between 1 and p, we choose k_{t+1} from $\{1, \ldots, p\} \backslash \{k_1, \ldots, k_t\}$ to minimize $\widehat{\mathrm{err}}(k_1, \ldots, k_{t+1})$. We use the procedure suggested above for breaking ties. That is, we choose k_{t+1} to be the value of k that, among those that minimize $\widehat{\mathrm{err}}(k_1, \ldots, k_t, k)$, maximizes the average of the values taken by $S(X)$ for variables X in the training data, when they are stripped of all their components except those indexed by $k_1, \ldots, k_t, k$. However, we terminate the algorithm at the sequence $k_1, \ldots, k_t$ if the operation of adjoining any other component index k_{t+1} would lead to a deterioration in estimated error rate. Here we can define 'deterioration' in a non-strict sense, meaning that $\widehat{\mathrm{err}}(k_1, \ldots, k_{t+1}) \leqslant \widehat{\mathrm{err}}(k_1, \ldots, k_t)$, or in a strict sense, where $\leqslant$ is replaced by $<$. If the algorithm does not terminate itself within a reasonable number of steps, it can be concluded in other ways. The permitted maximum value of t can also be determined differently, for example by terminating when the improvement in error rate,

$$\widehat{\mathrm{err}}(k_1, \ldots, k_t) - \widehat{\mathrm{err}}(k_1, \ldots, k_{t+1}),$$

falls below a given, positive level, or when t reaches a ceiling beyond which scientific interpretation (e.g. in a biological sense, if t denotes the number of genes used in the classifier) is difficult, or when the level of computation reaches a practical limit.

In practice we could determine, in advance, an upper bound t_0 chosen on the basis of computational resources. We would run the algorithm as suggested above, stopping at an empirically determined step $\hat{t}$ if $\hat{t} \leqslant t_0$, and terminating the algorithm at t_0, or at least reconsidering it at that point, if at the t_0th step it was clear that the algorithm would continue. The computational labour required to determine the indices $k_1, \ldots, k_t$ would then be bounded above by a constant multiple of $n^2 p t_0$, where $n = n_0 + n_1$. In practice, n_0 and n_1 are often very much less than p, for example being in the tens whereas p is in the thousands or tens of thousands.

2.3. Extensions of the algorithm

A handicap of the approach described above is that if, for some reason, we produce poor initial choices of k, we must keep them in the index set, potentially disadvantaging the final classifier. Furthermore, different variable choices at some early stage will often result in completely different variable selections later. These problems can be mitigated by using a jittered algorithm. This generates a pool of possibilities for k, for example consisting of all the values that minimize $\widehat{\mathrm{err}}(k)$ and for which the average of the leave-one-out versions of $(-1)^j S(X)$, when classification is based solely on the kth component, is among the ℓ_1 largest, where ℓ_1 is fixed (or, in theoretical terms, does not depend on p). We can also adjoin the set of indices k for which, given k_1 from the aforementioned set, $\widehat{\mathrm{err}}(k_1, k)$ is minimized and the average value of $(-1)^j S(X)$ is among the ℓ_2 largest. Write $\mathcal{L}$ for the set of integers k derived using these or related methods. Once $\mathcal{L}$ has been determined, the algorithm can be re-run so that it starts from any member of $\mathcal{L}$, rather than from k_1; alternatively, it could start from a subset of $\mathcal{L}$.

2.4. Computational labour

The computational cost of computing p values of $\widehat{\mathrm{err}}(k)$ for $k = 1, \ldots, p$ and selecting the largest of those values is bounded above by $C_1\, p$, where C_1 depends polynomially on n_0 and n_1. Both training sample sizes are typically much smaller than p, and therefore C_1 is small relative to p. Note that ranking the p values of $\widehat{\mathrm{err}}(k)$ costs $O(p \log p)$ steps, but the k for which $\widehat{\mathrm{err}}(k)$ is minimized can be found in only $O(p)$ steps, simply by storing and recursively updating both the minimum of $\widehat{\mathrm{err}}(k)$ and the minimizing value for $1 \leqslant k \leqslant k'$ as k' is increased.

Likewise, given $k_1, \ldots, k_{t-1}$, choosing the k that minimizes $\widehat{\mathrm{err}}(k_1, \ldots, k_{t-1}, k)$ requires no more than $C_t\, p$ steps. Therefore, for each t the cost of computing the values of $k_1, \ldots, k_t$ that minimize $\widehat{\mathrm{err}}(k_1, \ldots, k_t)$ sequentially increases only linearly in p.

2.5. Example: Centroid-based classifier

The standard centroid-based classifier assigns a new data value $X = (X^{(1)}, \ldots, X^{(p)})$ to Π_0 or Π_1 according to whether the statistic

$$S(X) = \sum_{k=1}^{p} \left\{ \left(X^{(k)} - \bar{X}_1^{(k)}\right)^2 - \left(X^{(k)} - \bar{X}_0^{(k)}\right)^2 \right\} \tag{3}$$

is positive or negative, respectively. Here, $\bar{X}_j = (\bar{X}_j^{(1)}, \ldots, \bar{X}_j^{(p)}) = n_j^{-1} \sum_i X_{ji}$ is the average value of the vectors in the training sample $\mathcal{S}_j$. We can equivalently, and more conventionally, interpret this rule as assigning X to Π_0 if X is closer to $\bar{X}_0$ than to $\bar{X}_1$, and assigning it to Π_1 otherwise. That is, X is deemed to be from Π_0 if $\|X - \bar{X}_0\| \leqslant \|X - \bar{X}_1\|$, and from Π_1 otherwise, where $\|\cdot\|$ denotes the conventional Euclidean metric. This approach to classification is popular in genomics; see, for example, Tibshirani *et al.* (2002), Dabney (2005), Dabney & Storey (2005, 2007) and Wang & Zhu (2007).

When constructing the error estimator at (1) we take X to be from one of the training samples $\mathcal{S}_j$, and we delete it from that sample when constructing $S(X)$ at (3). For example, if $X = X_{0i_1}$ is from $\mathcal{S}_0$ then $S(X)$ becomes

$$S_{0i_1}(X_{0i_1}) = \sum_{k=1}^{p} \left\{ \left(X_{0i_1}^{(k)} - \bar{X}_1^{(k)}\right)^2 - \left(X_{0i_1}^{(k)} - \frac{1}{n_0 - 1} \sum_{i\,:\,i \neq i_1} X_{0i}^{(k)} \right)^2 \right\}, \tag{4}$$

and a similar formula applies for $S_{1i_1}(X_{1i_1})$ if $X = X_{1i_1}$ is drawn from $\mathcal{S}_1$. Likewise, the classifier $\mathcal{C}_{-ji}(X_{ji} \mid k_1, \ldots, k_t)$ introduced in Section 2.1 is defined to be the rule that assigns X_{ji} to Π_0 if $S_{ji}(X_{ji} \mid k_1, \ldots, k_t) \geqslant 0$, and assigns it to Π_1 otherwise, where, for example,

$$S_{0i_1}(X_{0i_1} \mid k_1, \ldots, k_t) = \sum_{k = k_1, \ldots, k_t} \left\{ \left(X_{0i_1}^{(k)} - \bar{X}_1^{(k)}\right)^2 - \left(X_{0i_1}^{(k)} - \frac{1}{n_0 - 1} \sum_{i\,:\,i \neq i_1} X_{0i}^{(k)} \right)^2 \right\}. \tag{5}$$

The classifier $\mathcal{C}(X \mid k_1, \ldots, k_t)$ assigns X to Π_0 if $S(X \mid k_1, \ldots, k_t) \geqslant 0$, where $S(X \mid k_1, \ldots, k_t)$ has the definition of $S(X)$ at (3), except that the sum on the right-hand side there is taken only over $k = k_1, \ldots, k_t$.

Next we give an explicit definition of the tie-breaking procedure discussed in Section 2.2. Suppose that $k_1, \ldots, k_t$ have been determined and that we seek k_{t+1}. Compute the set $\mathcal{K}_t$ of values of k for which $\widehat{\mathrm{err}}(k_1, \ldots, k_t, k)$ achieves its minimum. For each $k \in \mathcal{K}_t$, and for each $j = 0, 1$ and each $i \in \{1, \ldots, n_j\}$, calculate $S_{ji}(X_{ji} \mid k_1, \ldots, k_t, k)$, and put

$$T(k) = \frac{\pi}{n_0} \sum_{i=1}^{n_0} S_{0i}(X_{0i} \mid k_1, \ldots, k_t, k) - \frac{1 - \pi}{n_1} \sum_{i=1}^{n_1} S_{1i}(X_{1i} \mid k_1, \ldots, k_t, k), \tag{6}$$

where π denotes the prior probability of Π_0. [We place a minus sign in front of the second term because $S_{1i}(X_{1i} \mid k_1, \ldots, k_t, k)$ is negative if the classifier $\mathcal{C}_{-1i}(\cdot \mid k_1, \ldots, k_t, k)$ correctly assigns X_{1i} to Π_1. Recall from Section 2.2 that the basic classifier has the property: Assign a new data value X to Π_j if $(-1)^j S(X) > 0$.] Choose $k \in \mathcal{K}_t$ to maximize $T(k)$.

The definition at (6) uses the mean to assess the average authority of classification decisions when k is included among components. If using the median we would redefine

$$T(k) = \pi \operatorname*{med}_{1 \leqslant i \leqslant n_0} S_{0i}(X_{0i} \mid k_1, \ldots, k_t, k) - (1 - \pi) \operatorname*{med}_{1 \leqslant i \leqslant n_1} S_{1i}(X_{1i} \mid k_1, \ldots, k_t, k),$$

where $\mathrm{med}_{1 \leqslant i \leqslant n}\, u_i$ denotes the median of values $u_1, \ldots, u_n$.

These constructions have close analogues in other classifiers: for example the support vector machine, where, in high-dimensional settings, $S(X)$ equals the square of the nearest distance from X to the convex hull formed by $\mathcal{S}_1$, minus its counterpart for the convex hull formed by $\mathcal{S}_0$; the nearest-neighbour classifier, where $S(X) = \min_i \|X - X_{1i}\|^2 - \min_i \|X - X_{0i}\|^2$; the average-distance classifier, where $S(X) = n_1^{-1} \sum_i \|X - X_{1i}\|^2 - n_0^{-1} \sum_i \|X -$

$X_{0i}\|^2$; the median-based classifier, an analogue of the centroid-based classifier, where $S(X) = \sum_k (|X^{(k)} - \text{med } X_1^{(k)}| - |X^{(k)} - \text{med } X_0^{(k)}|)$; and the discriminant classifier, where $S(X) = \log(p_0/p_1)$ and p_0, p_1 are the posterior probabilities of being in groups 0, 1, respectively. In each of these cases the definitions of $S_{ji}(X_{ji})$ and $S_{ji}(X_{ji} \mid k_1, \ldots, k_t)$, analogous to those at (4) and (5), follow directly.

3. Numerical properties

3.1. Preliminary discussion of real-data analysis

We study two genetic microarray datasets. The leukemia dataset was first used by Golub *et al.* (1999) to demonstrate the use of microarrays for diagnostic classification. It represents two classes denoting two types of acute leukemia (ALL/AML), with ALL represented by 0 and AML by 1. There are 72 observations in total, with 25 of these AML, and $p = 7129$ gene expression levels for each observation. The colon dataset of Alon *et al.* (1999) contains results of genetic expressions from colon biopsies. Of the 62 observations, 22 are classified as indicating tumours and the remaining 40 are normal. The original authors reduced the number of genes to 2000 from an original 6000, and we used this reduced dataset.

For each dataset we applied the methods listed in Table 1, using the recursive variable selection framework. Most of these methods were introduced in Section 2. The five-nearest-neighbour method classifies on the basis of the majority of the closest five observations. The score function $S(X)$ used is the number of zeros minus the number of ones. This creates the possibility of ties in the measure of authority, T, which must be ranked, for instance at random. However, we found that this was not a major concern on implementation. We used both linear and quadratic discriminant analysis, an introduction to which may be found in chapter 4 of Hastie, Tibshirani & Friedman (2001).

The main set of results compares prediction accuracy for each of these methods as a function of the number of genes in the model. Accuracy was measured using a double layer of cross-validation, with the inner layer leave-one-out and the outer layer 10-fold. Thus, for each classifier we first divided the data into 10 subsets of equal size. Then, for each subset we used the other nine to build a recursive model, which involved looping through the inner layer of cross-validation to select variables using the leave-one-out method. A series of models with increasing numbers of genes resulted. Once the model variables were selected, the resulting fit was applied to the remaining 10% of data, to assess accuracy.

TABLE 1

Approaches included in computational comparisons

Name	Description
Cent	Centroid-based classifier
Cent.med	Centroid-based classifier with median authority measure
Med	Median-based classifier
Dist	Average-distance classifier
1-NN	Nearest-neighbour classifier
5-NN	Five-nearest-neighbour classifier
LDA	Linear discriminant analysis
QDA	Quadratic discriminant analysis
SVM1	Linear support vector machine

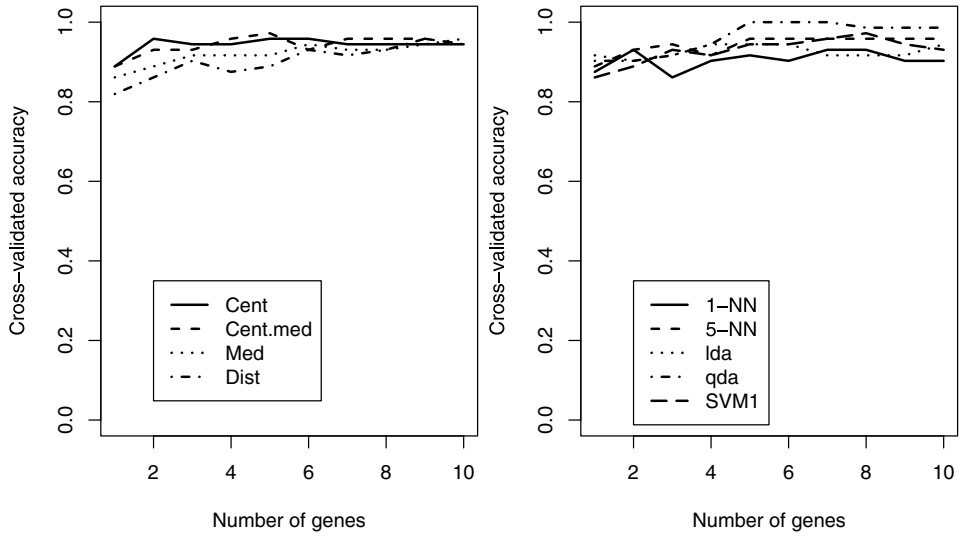

Figure 1. Accuracy curves for leukemia data.

TABLE 2

Accuracy of models using suggested model size for leukemia data

Name	Average model size	Accuracy (10-fold) CV	Final model variables
Cent	2.9	0.931	4847, 804, 6281
Cent.med	2.9	0.917	4847, 807, 6225
Med	2.5	0.889	4847, 804, 4951
Dist	3.5	0.889	3252, 1144, 2111
1-NN	2.8	0.861	3252, 4472, 6041
5-NN	3.4	0.931	4847, 804, 1685
LDA	3.1	0.944	4847, 2295, 1796, 2642
QDA	2.7	0.861	1882, 4342, 4582
SVM1	3.4	0.931	4847, 804, 4680

There is a strong case for the use of a double cross-validation method in such situations, as it avoids over-optimistic accuracy measurements caused by overfitting to the data. The resulting estimates give us a true indication of performance on an unseen dataset, as in each case the 10% of data set aside have not been used to fit the model at all.

3.2. Leukemia data

Figure 1 shows the accuracy curves for the array of methods, measured using double-layered cross-validation. Table 2 shows the obtained accuracy in the experiments, using the stopping rule (failure to decrease $\widehat{\text{err}}$) for model size selection. An average model size is reported, as each outer validation fold requires its own stopping rule, so model size may differ across folds. The variables listed for the 'final model' are those selected when we ran a single layer of cross-validation. We make the following observations.

(i) The accuracy curves as well as the stopping rule results suggest that only a small number of genes are required in the recursive model. In fact, accuracy does not improve beyond a few genes for most of the methods, with QDA and 5-NN the main exceptions. One explanation for this is that the small sample size does not allow reliable detection of effects that give small improvements in prediction accuracy.

(ii) Most approaches show comparable accuracy, here around 90%, despite the significant variation in model structure. The 1-NN method was generally the worst. QDA had equal worst performance in Table 2 when using a small number of genes, but had the best performance when taking a larger number of genes.

(iii) There appears to be a fair amount of stability in the first variable selected across the various methods, with variable 4847 chosen first in six of the nine experiments. Furthermore, given that 4847 was selected first, four of the six then selected 804 second. However, there was no consistency in the selection of third variables, suggesting that there was not a clear signal across all methods at this level of the model.

(iv) One computational consideration is whether an initial feature selection could be effected to reduce the dimensionality, and hence improve computability, while still leaving enough variables so that the final models were unchanged. This would involve discarding all variables that performed poorly in the initial variable selection step. In the case of the leukemia data, it turns out that any significant pruning would impact on some of the models. Variable 4342 in the QDA model was initially ranked 6746th, and variable 1144 in the distance model was initially ranked 1219th. At the other extreme, the 1-NN model needed only the top 50 variables to construct the classifier. In practice, such pre-screening should be avoided where possible. The colon dataset, considered below, illustrates another situation in which significant initial pruning negatively affects the final models.

One way to gain an understanding of the reliability of variable selection is through the bootstrap. As described in Section 2.3, we take resamples of the data and see which variable is selected first in each replication. Similarly, given a first variable we can use bootstrap replications to investigate which variable was selected second, and so on. Results for the Leukemia dataset are presented in Figure 2 , using the centroid version of the recursive method. The leftmost plot shows that there are two main contenders for top selection, variables 4847 and 1834. The second plot shows the range of choices if we choose 4847 as the first variable. We can see that there was much more variability in this second choice, with lower proportions for the strongest variables and a much greater proportion of 'other'. The third plot gives the results when we choose variable 1834 first, again with a fairly large spread of possibilities. There is not a great deal of overlap in the second and third lists, suggesting that variables 1834 and 4847 are different enough for the subsequent pathways to be distinct. Furthermore, variable 4847 does not appear on the third list, even though it is the best individual predictor, and conversely variable 1834 does not appear on the second list. This suggests that 4847 and 1834 contain fairly similar information, and thus better gains in accuracy can be obtained by choosing other variables.

A relevant question is how these classifiers compare with other contemporary approaches. Figure 3 provides an answer in the case of penalty-based methods. The model labelled 'SVM1' denotes the L1 penalized support vector machine (Zhu *et al.* 2004), while 'HHSVM' represents a hybrid Huberized support vector machine with L1 penalty (Wang, Zhu & Zou 2006, 2008).

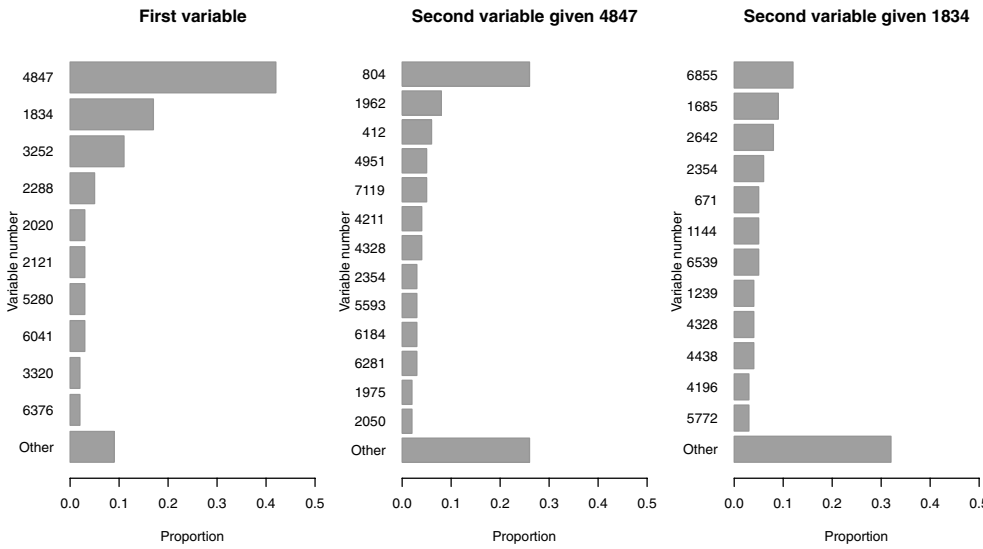

Figure 2. Variable selection frequency under bootstrap resampling.

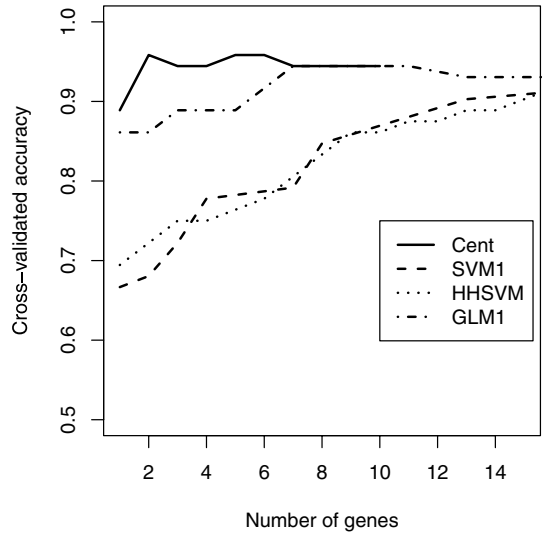

Figure 3. Accuracy on leukemia data compared with that of contemporary penalized methods.

Adjusting the L2 penalty in the latter case did not appear to have a significant impact on the presented results. Finally, 'GLM1' refers to the logistic regression method with L1 penalty. The results here are not conclusive; all approaches produce similar maximum accuracy when optimal gene sets are selected. However, the centroid-based classifier seems to do this the most efficiently, needing only a few genes for good accuracy. Questions of relative performance are pursued further in simulation work below.

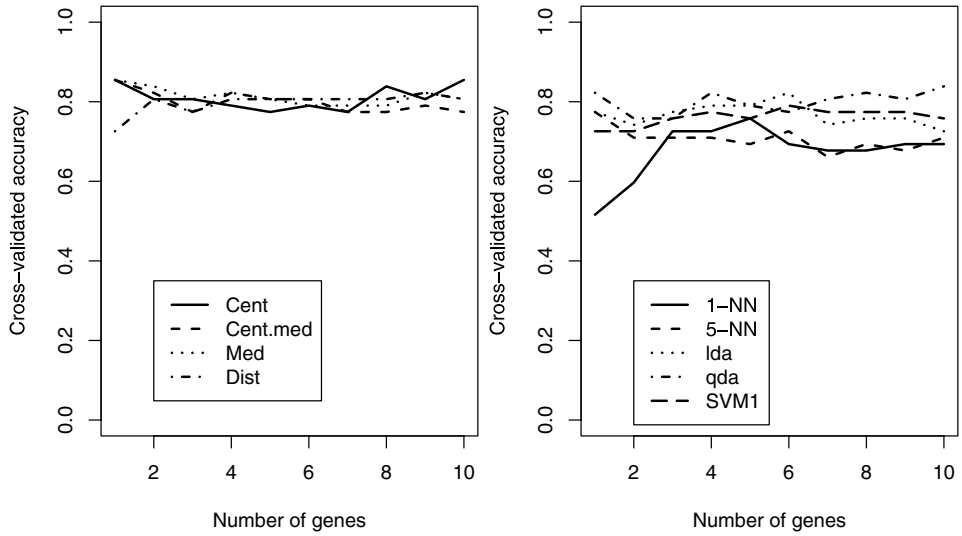

Figure 4. Accuracy curves for colon data.

TABLE 3

Accuracy of models using suggested model size for colon data

Name	Average model size	Accuracy (10-fold) CV	Final model variables
Cent	3.1	0.806	249, 1346, 799
Cent.med	2.9	0.819	249, 1346, 799
Med	2.9	0.861	249, 32
Dist	3.6	0.819	245, 1772, 206
1-NN	4.8	0.792	1042, 883, 1900, 1414
5-NN	3.7	0.792	1671, 1365
LDA	4.4	0.792	1423, 1870, 678, 137, 1769
QDA	4.8	0.792	249, 1757, 377, 1042
SVM1	2.7	0.778	249, 1935, 1976

3.3. Colon data

The accuracy curves for these estimates are plotted in Figure 4 , and Table 3 presents the results using the model size suggested by our stopping rule. The results prompt observations similar to those for the leukemia data. For instance, the selection of the first variable is reasonably stable across methods, with decreasing stability for later additions. Furthermore, the suggested model sizes are still small, although one of the methods (LDA) selected five genes. As before, an initial variable pruning would be inadvisable here, as in many instances a variable initially ranked poorly appears in the model; in the 1-NN model, 1883 is initially ranked 1870th, and in the distance model, variable number 206 is initially ranked 1056th.

3.4. Comparison with alternative approaches under simulation

Here we compare the performance of the cross-validated centroid-based classifier with the L1 penalized Huberized support vector machine and L1 penalized logistic regression.

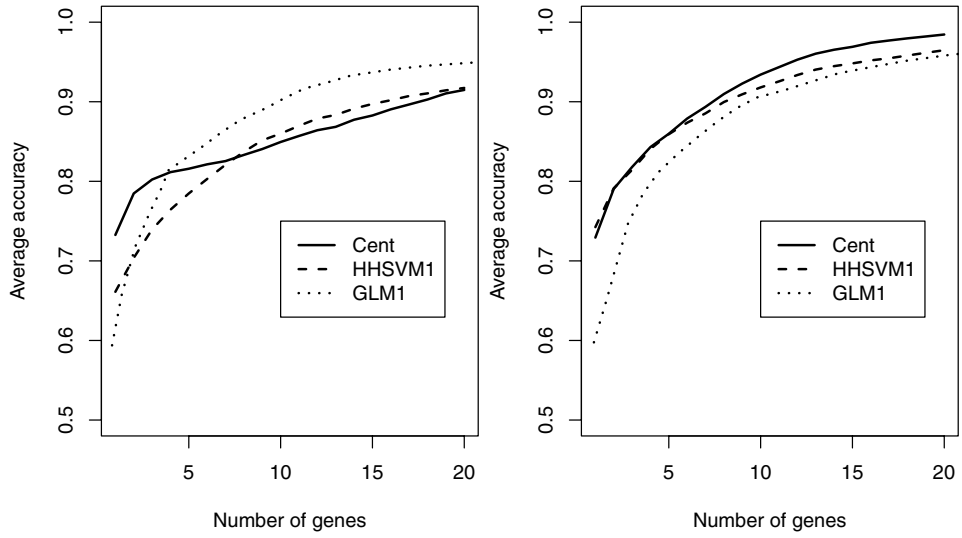

Figure 5. Comparison of accuracy results under simulation.

Two setups were used. In the first, we took $n = 100$ and $p = 5000$. Each variable had zero mean, except for those related to the binary response, Y, and 100 variables had mean μ_j when $Y = 1$ and mean 0 otherwise, with the μ_j randomly generated from a uniform distribution on [0, 2]. Each class had the same number of corresponding observations. The error for each variable was independent and normally distributed with zero mean. The standard deviation of the error was allowed to vary for each variable and each class, and was sampled from a uniform distribution on [1, 3]. The models built were fitted on a separate test set of 500 observations, and the simulation was repeated 50 times and the results averaged.

The results for this simulation are shown in the left panel of Figure 5 . Note that the standard error width for each of these curves is less than 0.01. While the centroid-based classifier strongly outperforms its competitors when the number of genes in the model is low (three or lower), it is generally inferior at larger model sizes. The reason for this appears to be that while the centroid-based classifier correctly targets the variables that maximize accuracy early on, for larger models it regularly has to use the tie-breaking rule to decide on variable choice. This rule makes no distinction between the score on observations classified well and those that are more marginal. By contrast, the loss functions for the competing methods cause them to focus on data that are misclassified. Thus, the discreetness of the cross-validated measure seems to be a factor. One possible means of addressing this is to modify the score function (3), although details are not pursued here.

The second simulation differed from the first in the following respect. There were 500 features that had different means in the two classes, making the simulation less sparse. These differences in means were sampled from those observed in the Leukemia dataset, and the error had a fixed standard deviation of 1. The results of this simulation are presented in the right-hand panel of Figure 5. In this case the centroid-based classifier dominates across all model sizes, plateauing at a higher accuracy level. This suggests that the modelling approach has superior performance in less sparse situations. In particular, the centroid-based classifier includes fewer redundant variables, improving the accuracy.

TABLE 4

Detection rates of genuine and irrelevant variables plus misclassification rates for data as in Theorem 1. Standard errors are given in brackets

p	u_p	v_p	Average gen.	Average irrel.	Average err.
10	0.126	0.063	3.55 (0.005)	1.36 (0.098)	0.350 (0.085)
20	0.179	0.089	3.54 (0.005)	3.10 (0.096)	0.255 (0.116)
50	0.283	0.141	3.31 (0.004)	3.62 (0.101)	0.166 (0.128)
100	0.400	0.200	3.30 (0.003)	3.08 (0.103)	0.133 (0.105)
200	0.566	0.283	3.45 (0.003)	1.76 (0.123)	0.110 (0.109)
500	0.894	0.447	4.14 (0.004)	0.21 (0.096)	0.062 (0.061)
1000	1.265	0.632	3.89 (0.003)	0.03 (0.073)	0.021 (0.020)
2000	1.789	0.894	3.18 (0.001)	0.00 (0.060)	0.004 (0.000)
5000	2.828	1.414	2.05 (<0.001)	0.00 (0.020)	0.000 (0.000)

3.5. Numerical work supporting theoretical results

Theorem 1 argues that if the classifier is constructed so as to be sensitive to the differences between Π_0 and Π_1, in particular to focus on extreme components if the principal differences are in terms of extrema, then our recursive approach can give particularly good performance and be more effective than, say, a linear classifier that does not acknowledge the context of the data.

To illustrate this we suppose that the data are generated as in the model addressed by Theorem 1. We keep $n_0 = n_1 = 30$ fixed, we take the components of X (when (X, Y) is drawn from Π_0) to be independent normal N(0, 1), we set $r = 5$, and we take $u_p = 2v_p$ and v_p proportional to $p^{1/2}$. Table 4 shows the average results over 100 simulations each for various values of p. The fourth column in the table shows the average number of the five genuine variables detected. As Theorem 1 suggests, we do not consistently detect all five, but this does not impact on asymptotic classification performance. The fifth column of Table 4 shows the average number of variables selected that are not genuine, and that this number decreases to zero as p and u_p grow. Finally, the sixth column shows the misclassification rate, which, in accordance with the theoretical results, is also driven to zero.

To illustrate the results of Theorem 2 we take the components of X to be independent standard normal when drawing from Π_0, and we let $n_0 = n_1 = 30$ and $r = 5$, as before. In the base case we set $p = 20$ and $\mu = 1$, and allow μ to increase at rate $p^{1/8}$. We repeated each simulation 100 times for various p and recorded the number of genuine and redundant variables selected, as well as the error rates for the recursive model and for a model that used all variables. The error rates were calculated using a separate test set of data generated in the same manner. The results are presented in Table 5. As expected we see that as p increases the average number of redundant variables mistakenly selected reduces, and the error for the recursive model also shrinks. However, the error for the full model increases, approaching 0.5, as μ is growing too slowly compared to p.

3.6. Computational considerations

In the past, the large sizes of some datasets meant that comprehensive cross-validation was undesirable owing to the computational labour required. This is not the case today, however. In support of this claim, Table 6 shows the 'raw' computation time, without any attempt made to optimize the cross-validation step, taken to select $t = 10$ genes for a

TABLE 5

Detection rates of genuine and irrelevant variables plus misclassification rates for data as in Theorem 2. Standard errors are given in brackets

p	μ	Average gen.	Average irrel.	Average err. recursive model	Average err. full model
20	1.00	3.27 (0.12)	0.43 (0.071)	0.189 (0.005)	0.157 (0.003)
50	1.12	2.70 (0.11)	0.31 (0.058)	0.187 (0.006)	0.160 (0.003)
100	1.22	2.58 (0.11)	0.23 (0.051)	0.173 (0.006)	0.162 (0.003)
200	1.33	2.47 (0.10)	0.28 (0.051)	0.155 (0.006)	0.176 (0.003)
500	1.50	2.69 (0.12)	0.28 (0.059)	0.124 (0.006)	0.207 (0.003)
1000	1.63	2.64 (0.12)	0.30 (0.052)	0.108 (0.006)	0.231 (0.003)
2000	1.78	2.85 (0.12)	0.23 (0.051)	0.083 (0.005)	0.260 (0.003)
5000	1.99	3.14 (0.15)	0.22 (0.048)	0.056 (0.005)	0.304 (0.003)
10000	2.17	3.22 (0.13)	0.13 (0.037)	0.038 (0.004)	0.323 (0.004)

TABLE 6

Computer time taken to fit recursive models on Leukemia data

Name	Raw fitting time (mins)
Cent	19.9
Cent.med	19.9
Med	121.3
Dist	28.3
1-NN	27.5
5-NN	23.5
LDA	4.7
QDA	5.9
SVM1	399.0

single-layer, leave-one-out cross-validated fit to the Leukemia data ($n = 72$, $p = 7129$) for each of the methods used. The fits were implemented using R running on a typical desktop computer with a 2.66-Ghz processor. The number of classifiers fitted for each method was over five million.

With the possible exception of the median and SVM approaches, the computation times given in Table 6 are very reasonable. In the case of microarrays, many hours of laboratory time are needed to produce the data, so 20 minutes to fit a robust model is not unduly onerous. Of course, part of the reason the modelling times are so reasonable is that we have used relatively simple classifiers, although the good prediction performance above suggests that this is not a significant issue.

Note too that there are often ways to improve model fitting times, taking advantage of the fact that calculations of moments and other similar statistics do not change greatly as individual observations are excluded. For instance, in the case of the centroid-based classifier we can rewrite (4) as

$$S_{0i_1}(X_{0i_1}) = \sum_{k=1}^{p} \left\{ \left(X_{0i_1}^{(k)} - \bar{X}_1^{(k)}\right)^2 - \left(X_{0i_1}^{(k)} - \frac{1}{n_0 - 1} \sum_{i\,:\,i \neq i_1} X_{0i}^{(k)} \right)^2 \right\}$$

$$= \sum_{k=1}^{p} \left\{ \left(X_{0i_1}^{(k)} - \bar{X}_1^{(k)}\right)^2 - \left(1 + \frac{1}{n_0 - 1} \right) \left(X_{0i_1}^{(k)} - \bar{X}_0^{(k)}\right)^2 \right\}$$

Thus, rather than calculating the class means separately when omitting each of the n observations, we can calculate the overall class means and modify these slightly. Because calculation of the mean is the most computer-intensive stage in fitting the centroid model, this simplification offers substantial performance improvements. In the case of the centroid-based classifier applied to the Leukemia dataset, using this approach reduces the fitting time from 19.9 minutes to only 3.8 minutes. A similar gain can be made in the case of the median method, as removing a single observation will only move the overall median up or down half a rank. In this case the computation time was reduced from over two hours to 8.3 minutes. Similarly, the computation time for the distance method can be reduced from 28.3 to 4.8 minutes. The nearest-neighbour methods are not as amenable to this type of optimization. The remaining methods, despite being somewhat more involved, also admit streamlined approaches to computation.

4. Theoretical illustrations

4.1. Example where Π_0 and Π_1 differ in terms of a small number of components taking extreme values

We show here that using a classifier that is tuned to the differences between Π_0 and Π_1, and employing the recursive variable selection algorithm given in Section 2.2, can result in particularly accurate identification of those vector components that have greatest leverage for classification. On the other hand, using a conventional approach to variable selection, in particular one based on a linear model, can produce poor results. Therefore, an attempt at classification using variables chosen by a standard method can be quite ineffective.

First we characterize Π_0 and Π_1. A random vector (X, Y) from Π_1, that is, a vector for which $Y = 1$, is constructed by drawing (X, Y) from Π_0 and replacing r specific components, with indices $k = k_{01}, \ldots, k_{0r}$ say, by random variables all of whose absolute values exceed a given number u_p. We keep the training sample sizes, n_0 and n_1, fixed as p increases, reflecting the high dimension and small sample size of many contemporary problems. The value of u_p is taken to increase with p, and r is held fixed. The vectors $X = (X^{(1)}, \ldots, X^{(p)})$ in the training data are assumed to be independent, but no assumptions are made about dependence among the components of any given X.

The classifier that we shall use reflects characteristics of the data, as would ideally be the case in practice. In particular, a sequence $k_1 < \cdots < k_t$ of distinct integers between 1 and p is chosen empirically using the training data, as suggested in Section 2, and a new data vector X, for which the corresponding Y is not known, is classified as type 1 if $|X^{(k_s)}| > v_p$ for $1 \leqslant s \leqslant t$, where $v_p \in (0, u_p)$ is given, and as type 0 otherwise.

In this example, the actual construction of the classifier does not depend on the training data; for the sake of simplicity we are assuming that u_p is not a function of the data in $\mathcal{S}_0 \cup \mathcal{S}_1$. Therefore, the leave-one-out aspect of the definition (1) of estimated error rate can be ignored, and we can define instead:

$$\widehat{\mathrm{err}}(k_1, \ldots, k_t) = \frac{\pi}{n_0} \sum_{i=1}^{n_0} I\{\mathcal{C}(X_{0i} \,|\, k_1, \ldots, k_t) = 1\} + \frac{1-\pi}{n_1} \sum_{i=1}^{n_1} I\{\mathcal{C}(X_{1i} \,|\, k_1, \ldots, k_t) = 0\}$$

$$= \frac{\pi}{n_0} \sum_{i=1}^{n_0} I\bigl(\bigl|X_{0i}^{(k_s)}\bigr| > v_p \text{ for all } s,\ 1 \leqslant s \leqslant t\bigr)$$

$$+ \frac{1-\pi}{n_1} \sum_{i=1}^{n_1} I\bigl(\bigl|X_{1i}^{(k_s)}\bigr| \leqslant v_p \text{ for some } s,\ 1 \leqslant s \leqslant t\bigr).$$

Moreover, in the asymptotic regime considered in Theorem 1 below, the probability that there is a tie for the minimizing value of k_t, between two indices in the respective sequences $1, \ldots, r$ and $r+1, \ldots, p$, converges to zero as p diverges. Hence, when stating the theorem there is no need to consider the tie-breaking scheme discussed in Section 2.

The theorem shows that, with probability converging to 1 as $p \to \infty$, and provided that p does not increase too rapidly relative to v_p, the recursive variable selector correctly chooses at least a subset of the components where the distributions of the sub-populations Π_0 and Π_1 differ; it does not choose any other components; and it results in zero classification error. We take the prior probability π of the sub-population Π_0 to lie in the interval $(0, 1)$ and to not depend on p. Define $\alpha_p = \max_{k \leqslant p} P(|X^{(k)}| \geqslant v_p \mid Y = 0)$; that is, α_p is the maximum over k of the probability that $|X^{(k)}|$ exceeds v_p when (X, Y) is drawn from Π_0. All proofs are given in a supplementary appendix available online at the *ANZS* website (see Supporting Material, Data S1).

Theorem 1. *Assume that $0 < v_p < u_p$, that $n_0, n_1 \geqslant 1$ are kept fixed as p increases, that p and v_p increase together in such a manner that $p\,\alpha_p^{n_1} \to 0$ as $p \to \infty$, and that the models for Π_0 and Π_1, given above, apply. Then with probability converging to 1 as $p \to \infty$, (i) the algorithm terminates at an integer $t = \hat{t} \leqslant r$, and (ii) the values of $k_1, \ldots, k_{\hat{t}}$ chosen by the recursive algorithm prior to termination are all among the special indices $1, \ldots, r$ for which the distribution of the vector component differs between the sub-populations Π_0 and Π_1. Moreover, (iii) the error rate of the classifier, given by (2) with t replaced by $\hat{t}$, converges to 0 as $p \to \infty$, and with probability converging to 1 the classifier based on the reduced dimensions $k_1, \ldots, k_{\hat{t}}$ gives correct classification for data vectors drawn from either Π_0 or Π_1.*

In many of the cases covered by Theorem 1, conventional linear variable selection can be expected to perform very poorly. For example, if the components of X, when the data come from Π_0, have a symmetric distribution, then Y is uncorrelated with each component of $X = (X^{(1)}, \ldots, X^{(p)})$. This follows from the fact that the conditional distribution of $X^{(j)}$, given Y, is symmetric. Therefore, a model that depended linearly on the variables would have little opportunity for expressing the influence that any $X^{(j)}$ has on Y. This argument was illustrated numerically in Section 3.5.

4.2. Example where Π_0 and Π_1 differ in location

Here we show that even the simple algorithm in Section 2.2 can substantially improve the performance of a conventional classifier. We treat the standard centroid-based method, the performance of which can be quite poor when applied to classification problems in which the distributions of the two sub-populations differ by a relatively large amount in only a small number of components, rather than by a relatively small amount in a large number of components. The latter context is often referred to as having low sparsity, as information is

available in a relatively high proportion of components. Although the centroid-based approach has optimality properties, it demonstrates them only when the degree of sparsity is low, not (as in the examples we give here) in the case of high sparsity; and it shares this feature with related methods, such as the support vector machine. Theorem 2, below, shows that recursive variable selection adapts well to high-sparsity settings, ensuring relatively good performance there. A similar result can be proved in the case of support vector machine classifiers.

We assume that a random vector $X = (X^{(1)}, \ldots, X^{(p)})$, when (X, Y) is drawn from Π_1, is constructed by taking the vector from Π_0 and then adding the constant μ to r specific components of X, in particular those with indices $k_{01}, \ldots, k_{0r}$. The algorithm in Section 2.2 is used to construct the variable selector.

The theorem below shows that, with probability converging to 1 as $p \to \infty$, the recursive classifier correctly assigns a new data value to either Π_0 or Π_1, provided that $|\mu|$ is of larger order then $\log p$; and that, on the other hand, the standard centroid-based classifier fails to give correct classification unless $|\mu|$ is at least as large as $p^{1/4}$. This establishes the extent to which the recursive algorithm can improve performance of the classifier in cases where information is sparse. The theorem also demonstrates that, with probability converging to 1 as $p \to \infty$, the recursive approach correctly chooses at least a subset of the components where the distributions of the sub-populations Π_0 and Π_1 differ, and does not choose any other components.

We suppose that, for some $c > 0$,

$$\sup_{p \geqslant 1} \max_{1 \leqslant k \leqslant p} E\{\exp(c\,|X^{(k)}|) \,|\, Y = 0\} < \infty. \tag{7}$$

That is, we ask that the component-wise moment generating functions of X, when (X, Y) is drawn from Π_0, be uniformly bounded in some neighbourhood of the origin.

Theorem 2. *Assume that (7) holds, that $n_0, n_1 \geqslant 2$, that $|\mu|/\log p \to \infty$, and that the above models for Π_0 and Π_1 apply. Then with probability converging to 1 as $p \to \infty$, results (i) and (ii) from Theorem 1 hold. Moreover, result (iii) from that theorem obtains, and with probability converging to 1 the classifier based on the reduced dimensions $k_1, \ldots, k_{\hat{t}}$ gives correct classification for data vectors drawn from either Π_0 or Π_1. Also, (iv) if the components of X when (X, Y) is drawn from Π_0 are independent and identically distributed; if we employ the standard centroid-based classifier, in which the sign of the statistic at (3) is used to assign X to Π_0 or Π_1, without any dimension reduction; and if $n_0 = n_1$ and $|\mu|/p^{1/4} \to 0$ as $p \to \infty$; then the probability of correct classification converges to $\frac{1}{2}$ as $p \to \infty$.*

The requirement in Theorem 2 that $n_0 \geqslant 2$ and $n_1 \geqslant 2$ ensures that the leave-one-out approach to estimating the error rate is feasible. The assumption of independence in part (iv) of the theorem makes the classification problem relatively difficult, because then the noise can differ markedly from one vector component to another. On the other hand, the condition $n_0 = n_1$, also in part (iv), actually gives a result that is relatively favourable to the standard centroid-based classifier. It permits a critical cancellation at one point in the argument. Without the condition $n_0 = n_1$, and in the case of fixed n_0 and n_1, the value of $|\mu|$ generally has to be as large as $p^{1/2}$, not $p^{1/4}$, before the centroid-based classifier can distinguish between Π_0 and Π_1.

Supporting information

Additional Supporting Information may be found in the online version of this article at http://wileyonlinelibrary.com/journal/anzs

Data S1. Supplementary proofs

Please note: Wiley-Blackwell are not responsible for the content or functionality of any supporting materials supplied by the authors. Any queries (other than missing material) should be directed to the corresponding author for the article.

References

ALON, U., BARKAI, N., NOTTERMAN, D.A., GISH, K., YBARRA, S., MACK, D. & LEVINE, A.J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci.* **96**, 6745–6750.

ASIMOV, D. (1985). The grand tour: A tool for viewing multidimensional data. *SIAM J. Sci. Statist. Comput.* **6**, 128–143.

BILMES, J.A. & KIRCHHOFF, K. (2004). Generalized rules for combination and joint training of classifiers. *Patt. Anal. Appl.* **6**, 201–211.

BREIMAN, L. (1995). Better subset regression using the nonnegative garrote. *Technometrics* **37**, 373–384.

CANDES, E. & TAO, T. (2007). The Dantzig selector: statistical estimation when p is much larger than n. *Ann. Statist.* **35**, 2313–2351.

CLARKE, R., RESSOM, H.W., WANG, A., XUAN, J., LIU, M.C., GEHAN, E.A. & WANG, Y. (2008). The properties of high-dimensional data spaces: implications for exploring gene and protein expression data. *Nature Reviews Cancer* **8**, 37–49.

COOTES, T.F., HILL, A., TAYLOR, C.J. & HASLAM, J. (1993). The use of active shape models for locating structures in medical images. In *Information Processing in Medical Imaging, Lecture Notes in Computer Science 687*, eds H. H. BARRET and A. F. Gmitro, pp. 33–47. Berlin: Springer.

DABNEY, A.R. (2005). Classification of microarrays to nearest centroids. *Bioinformatics* **21**, 4148–4154.

DABNEY, A.R. & STOREY, J.D. (2005). Optimal feature selection for nearest centroid classifiers, with applications to gene expression microarrays. Working Paper No. 267, University of Washington Biostatistics Working Paper Series.

DABNEY, A.R. & STOREY, J.D. (2007). Optimality driven nearest centroid classification from genomic data. *PLoS One* **2**, e1002. doi: 10.1371/journal.pone.0001002.

DONOHO, D.L. (2006a). For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution. *Comm. Pure Appl. Math* **59**, 797–829.

DONOHO, D.L. (2006b). For most large underdetermined systems of equations, the minimal ℓ_1-norm near-solution approximates the sparsest near-solution. *Comm. Pure Appl. Math* **59**, 907–934.

DONOHO, D.L. & ELAD, M. (2003). Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization. *Proc. Nat. Acad. Sci. USA* **100**, 2197–2202.

DONOHO, D.L. & HUO, X. (2001). Uncertainty principles and ideal atomic decomposition. *IEEE Trans. Inform. Th.* **47**, 2845–2862.

DONOHO, D.L., JOHNSTONE, I.M., KERKYACHARIAN, G. & PICARD, D. (1995). Wavelet shrinkage: asymptopia? With discussion and a reply by the authors. *J. Roy. Statist. Soc. Ser. B* **57**, 301–369.

DUDA, R.O., HART, P.E. & STORK, D.G. (2001). *Pattern Classification*, 2nd edn. New York: Wiley.

DUDOIT, S., FRIDLYAND, J. & SPEED, T. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *J. Amer. Statist. Assoc.* **97**, 77–87.

FAN, J. & LI, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *J. Amer. Statist. Assoc.* **96**, 1348–1360.

FAN, J. & LV, J. (2008). Sure independence screening for ultra-high dimensional feature space. *J. Roy. Statist. Soc. Ser. B* **70**, 849–911.

FRANCO-LOPEZ, H., EK, A.R. & BAUER, M.E. (2001). Estimation and mapping of forest stand density, volume, and cover type using the k-nearest neighbors method. *Remote Sensing of Environment* **77**, 251–274.

FRIEDMAN, J.H. & TUKEY, J.W. (1974). A projection pursuit algorithm for exploratory data analysis. *IEEE Trans. Comput.* **C-23**, 881–890.

GAO, H.-Y. (1998). Wavelet shrinkage denoising using the non-negative garrote. *J. Comput. Graph. Statist.* **7**, 469–488.

GOLUB, T.R., SLONIM, D.K., TAMAYO, P., HUARD, C., GAASENBEEK, M., MESIROV, J.P., COLLER, H., LOH, M.L., DOWNING, J.R., CALIGIURI, M.A., BLOOMFIELD, C.D. & LANDER, E.S. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* **286**, 531–537.

HALL, P. & MILLER, H. (2009). Using generalized correlation to effect variable selection in very high dimensional problems. *J. Comput. Graph. Statist.* **18**, 533–550.

HASTIE, T., TIBSHIRANI, R. & FRIEDMAN, J. (2001). *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. New York: Springer.

HUA, J.P., TEMBE, W.D. & DOUGHERTY, E.R. (2009). Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recognition* **42**, 409–424.

INZA, I., LARRAÑAGA, P., BLANCO, R. & CERROLAZA, A. J. (2004). Filter versus gene wrapper approaches in DNA microarray domains. *Artif. Intell. Med.* **31**, 91–103.

MOON, H., AHN, H., KODELL, R.L., LIN, C.-J., BAEK, S. & CHEN, J.J. (2006). Classification methods for the development of genomic signatures from high-dimensional data. *Genome Biology* **7**, R121.1–R121.7.

SAEYS, Y., INZA, I. & LARRAÑAGA, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics* **7**, 2507–2517.

SCHOONOVER, J.R., MARX, R. & ZHANG, S.L. (2003). Multivariate curve resolution in the analysis of vibrational spectroscopy data files. *Applied Spectroscopy* **57**, 483–490.

SHAKHNAROVICH, G., DARRELL, T. & INDYK, P. (2005). *Nearest-Neighbor Methods in Learning and Vision. Theory and Practice*. Cambridge, MA: MIT Press.

SHAO, J. (1993). Linear model selection by cross-validation. *J. Amer. Statist. Assoc.* **88**, 486–494.

TIBSHIRANI, R. (1996). Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B* **58**, 267–288.

TIBSHIRANI, R., HASTIE, T., NARASIMHAN, B. & CHU, G. (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc. Natl. Acad. Sci.* **99**, 6567–6572.

TROPP, J.A. (2005). Recovery of short, complex linear combinations via ℓ_1 minimization. *IEEE Trans. Inform. Th.* **51**, 1568–1570.

WANG, L., ZHU, J. & ZOU, H. (2006). The doubly regularized support vector machine. *Statistica Sinica* **16**, 589–615.

WANG, L., ZHU, J. & ZOU, H. (2008). Hybrid Huberized support vector machines for microarray classification and gene selection. *Bioinformatics* **24**, 412–419.

WANG, S. & ZHU, J.I. (2007). Improved centroids estimation for the nearest shrunken centroid classifier. *Bioinformatics* **23**, 972–979.

WARD, M.O., LEBLANC, J.T. & TIPNIS, R. (1994). N-Land: a graphical tool for exploring N-dimensional data. In *Proc. Computer Graphics International Conference*; Jun 1994, Melbourne, Australia. Available from URL: http://davis.wpi.edu/∼matt/docs/cgi94.ps.

XIONG, M., FANG, X. & ZHAO, J. (2001). Biomarker identification by feature wrappers. *Genome Res.* **11**, 1878–1887.

ZHU, J., ROSSET, S., HASTIE, T. & TIBSHIRANI, R. (2004). 1-norm support vector machines. In *Advances in Neural Information Processing Systems 16*, eds S. Thrun, L.K. Saul and B. Schölkopf, pp. 49–56. Boston, MA: MIT Press.