Hierarchical Classification using Shrunken Centroids

Mee Young Park * Trevor Hastie †

April 21, 2005

Abstract

There are various types of classifiers that can be trained on gene expression data with class labels. Many of them have an embedded mechanism for feature selection, by which they distinguish a subset of significant genes that are used for future prediction. When dealing with more than two class labels, especially when the number goes up to a dozen or more, people find it useful to know the relative affinity among the classes and different subsets of genes involved in discriminating different groups of classes. It provides them with more information not only when analyzing the relationship among classes, but also when predicting on future instances. We have achieved this by developing a hierarchical adaptation of the nearest shrunken centroid classifier. Here, we demonstrate our new method using a cancer data example.

1 Introduction

DNA microarrays have recently come into the spotlight and have been applied in a wide range of problems. One of the most important issues deals with discriminant and cluster analysis of samples using gene expression. In supervised learning, or discriminant analysis, the classes are predefined, while in unsupervised learning, or cluster analysis, the classes are unknown a priori and need to be discovered

*Ph.D student at Department of Statistics, Stanford University, CA 94305. mypark@stat.stanford.edu, *tel* 16507042581

†Professor at Department of Statistics and Department of Health Research & Policy, Stanford University, CA 94305. hastie@stat.stanford.edu

from the data [Speed, 2003]. Various methods have been explored in building a classifier or defining clusters for gene expression data, which are distinguished from other types of data by the fact that the number of genes (predictors) is usually much larger than the number of samples. More importantly, only a small fraction of predictors are meaningful in classifying the samples, and we want to identify them.

The nearest shrunken centroid (NSC) classifier [Tibshirani et al., 2003] is a simple, efficient, and highly interpretable classification method adapted for gene expression data. It uses a discriminant score defined for each sample, similar to that of the linear discriminant analysis, and soft thresholding is applied so that it successfully selects a subset of genes significant in distinguishing the classes. Also called prediction analysis for microarrays (PAM), it is especially useful when dealing with more than two classes.

When no predefined classes are available, hierarchical clustering methods can be used. Hierarchical clustering uses a measure of dissimilarity between groups of observations to build a nested hierarchy of observations ([Eisen et al., 1998], or Chapter 14 of [Hastie et al., 2001]). It creates a hierarchical representation from the lowest level where each cluster is a single observation, to the highest level where only one cluster with all observations remains. The order in which the clusters are formed is either bottom-up or top-down, depending on the initial stage of clustering. Hierarchical clustering is of interest because different levels of granularity in clustering can be investigated simultaneously due to its nested property.

In this paper, we propose a method that uses the nearest shrunken centroid classifier along with a hierarchical clustering method to come up with a hierarchical classification scheme. In a manner similar to the recursive joining of clusters based on dissimilarity, we recursively apply PAM to groups made up of different numbers of classes at each level of a hierarchy. We start from the bottom where observations of the same class are put into one of K disjoint groups (representing the classes). We then choose the most similar pair of classes and combine them into one larger class. This procedure is repeated with one less class, continuing until we only have one large class consisting of all K classes. The hierarchical nearest shrunken centroid classifier takes advantage of the efficiency and the appealing *feature-selection* property of PAM; moreover, it inherits the interpretability of hierarchical clustering. It gives a hierarchical visualization of multiple classes, and at each level, there is a nearest shrunken

centroid classifier and a corresponding set of genes that characterize the merge. In addition, we design algorithms so that the hierarchical classifier uses fewer genes in prediction than the regular NSC classifier. Through a real microarray data example, we show that this hierarchical method performs as well as the NSC classifier, but provides additional useful information.

In Section 2, we review the discrimination method used in the nearest shrunken centroid classifier and propose a slight modification to be used in the hierarchical classification. In Section 3, we describe different approaches to creating the hierarchical structure of classes: some methods based on cross-validation errors and the others based on a distance measure between the classes. In section 4, these methods are demonstrated with real gene expression data.

2 Nearest Shrunken Centroid Classifier

2.1 Multi-class classification

Here we review the classification method (NSC) proposed in [Tibshirani et al., 2003]. We refer to it as $multi-class$ to distinguish from our new method ($two-class$), which will be covered in 2.2.

Let (x_i, y_i) for $i = 1, \ldots, n$ denote pairs of a gene expression profile ($x_i \in \mathcal{R}^p$) and the corresponding class label ($y_i \in \{1, \ldots, K\}$). Let n_k be the number of samples of class k, with $\sum_k n_k = n$. For a given sample $x \in \mathcal{R}^p$, the posterior probability of class k is (using the Bayes theorem)

$$\Pr(y = k|\mathbf{X} = x) = \frac{f_k(x)\pi_k}{\sum_{g=1}^{K} f_g(x)\pi_g}, \tag{2.1}$$

where $f_g(x)$ is the class conditional density $\Pr(x|Y = g)$, and π_g is the prior probability for class g. The Bayes rule classifies the sample into the class with the highest posterior probability:

$$C(x) = \underset{k}{\operatorname{argmax}} \Pr(y = k|\mathbf{X} = x). \tag{2.2}$$

In the special case of a Gaussian density [McLachlan, 1992] for each class, with a common, diagonal covariance matrix ($\mathbf{\Sigma}_k = diag\{\sigma_1^2, \ldots, \sigma_p^2\},\ k = 1, \ldots, K$), this classification rule reduces to

$$C(x) = \underset{k}{\operatorname{argmin}}\, \delta_k(x), \tag{2.3}$$

where $\delta_k(x) = \sum_{j=1}^{p} \frac{(x_j - \mu_{jk})^2}{\sigma_j^2} - 2\log \pi_k$. When building a classifier, the class means (μ_k) and variances ($\sigma_1^2, \ldots, \sigma_p^2$) need to be estimated from the training data $\{(x_i, y_i)\}$.

Tibshirani et al. [Tibshirani et al., 2003] define the discriminant score for class k

$$\delta_k(x) = \sum_{j=1}^{p} \frac{(x_j - \bar{x}'_{jk})^2}{(s_j + s_0)^2} - 2\log \pi_k, \tag{2.4}$$

where $\bar{x}'_{jk}$ is the shrunken centroid and $s_j + s_0$ is a regularized estimate of the standard error. Their idea was that the genes whose class specific centroids are close to the overall centroid should not play a role in prediction. The $\bar{x}'_{jk}$ are constructed as follows:

- First let

$$d_{jk} = \frac{\bar{x}_{jk} - \bar{x}_j}{m_k(s_j + s_0)} \tag{2.5}$$

  with $\bar{x}_{jk} = \sum_{i:y_i=k} x_{ji}/n_k$, $\bar{x}_j = \sum_{i=1}^{n} x_{ji}/n$, $s_i^2 = \frac{1}{n-K}\sum_k \sum_{i:y_i=k}(x_{ji} - \bar{x}_{jk})^2$, and $m_k = \sqrt{1/n_k - 1/n}$. $\bar{x}_{jk} - \bar{x}_j$ is standardized, thereby d_{jk} being almost identical to a t-statistic. s_0, a positive constant, is added so that the level of d_{jk} does not depend on the level of gene expressions. s_0 has a default value of the median of all s_j across the genes, and the software allows users to choose different quantiles as well.

- Apply the soft thresholding operator:

$$d'_{jk} = sign(d_{jk})(|d_{jk}| - \Delta)_+. \tag{2.6}$$

- Use these reduced d'_{jk} to reconstruct the centroids:

$$\bar{x}'_{jk} = \bar{x}_j + m_k(s_j + s_0)d'_{jk}. \tag{2.7}$$

Thus, the number of genes that will be used in predicting the classes varies as the shrinkage factor Δ changes. As Δ becomes larger, fewer genes contribute to the classification, leaving only the most significant genes as discriminating features among different classes.

2.2 Two-class classification

We now propose a modification of this classifier for the case of two groups, where each could be a mixture of classes; this alternative version will be used when training a classifier at each non-terminal node of a hierarchical structure. If both groups are composed of only one class, it is equivalent to applying the NSC classification with $K = 2$. In addition to the training set $\{(x_i, y_i) : x_i \in \mathcal{R}^p, y_i \in \{1, \ldots, K\}$ for $i =$

$1, \ldots, n\}$, we have an additional higher-level classification such that a subset of classes $\{1, \ldots, K\}$ belongs to G_1 and the remaining ones belong to G_2.

The class conditional density for G_1 can be written as

$$g_1(x) = \Pr(x|Y \in G_1) = \sum_{k \in G_1} \pi_k f_k(x), \tag{2.8}$$

where $\sum_{k \in G_1} \pi_k = 1$, and f_k are the class conditional densities for the subclasses of group 1. Again we assume that the subclasses have Gaussian densities with different means, but there exists a unique diagonal covariance matrix across all groups and the containing subclasses. Then the posterior probability for group 1 is

$$\Pr(y \in G_1|\mathbf{X} = x) = \frac{\sum_{k \in G_1} \pi_k f_k(x) \Pi_1}{\sum_{k \in G_1} \pi_k f_k(x) \Pi_1 + \sum_{k \in G_2} \pi_k f_k(x)(1 - \Pi_1)} \tag{2.9}$$

$$= \frac{\sum_{k \in G_1} \pi_k \exp(-\frac{1}{2} \frac{\sum_{j=1}^{p} (x_j - \mu_{jk})^2}{\sigma_j^2}) \Pi_1}{\sum_{k \in G_1} \pi_k \exp(-\frac{1}{2} \frac{\sum_{j=1}^{p} (x_j - \mu_{jk})^2}{\sigma_j^2}) \Pi_1 + \sum_{k \in G_2} \pi_k \exp(-\frac{1}{2} \frac{\sum_{j=1}^{p} (x_j - \mu_{jk})^2}{\sigma_j^2})(1 - \Pi_1)}, \tag{2.10}$$

where Π_1 denotes the prior probability for group 1. This formulation could be generalized to K mixtures for any number K, but we confine ourselves to two-class problems for the hierarchical classification.

The classification criterion (2.2) applies here as well, letting us define a mixture version of the discriminant score

$$\delta_1(x) = \sum_{k \in G_1} \pi_k \exp(-\frac{1}{2} \sum_{j=1}^{p} \frac{(x_j - \bar{x}'_{jk})^2}{(s_j + s_0)^2}) \Pi_1, \tag{2.11}$$

and the classification rule

$$C(x) = 1 \text{ if and only if } \delta_1(x) > \delta_2(x). \tag{2.12}$$

The score is similar to (2.4) but cannot be simplified by taking the log transformation.

The shrunken centroids and the estimates of standard errors are defined analogously to the NSC classifier. As Δ, the positive shrinkage parameter, increases, d'_{jk} are set to zero for more genes (as in equation (2.6)). If d'_{jk} are set to zero for all $k \in G_1$, all the subclasses of group 1 have the same centroid for gene j. If it is eliminated for all $k \in G_1 \cup G_2$, the corresponding gene is not used as a discriminant factor in the class prediction. In the hierarchical classification scheme, each step of the hierarchy deals with a two-class classification of groups with different combinations of subclasses. Therefore, each level has a unique set of significant genes contributing to the nearest-centroid formulation.

The prior probabilities for the classes and their subclasses are also estimated from the samples. If $k \in G_1$,

$$\hat{\pi}_k = \frac{n_k}{\sum_{g \in G_1} n_g} \text{ and } \hat{\Pi}_1 = \frac{\sum_{g \in G_1} n_g}{n}. \tag{2.13}$$

2.3 Relationship to discriminant analysis

The analogy of the NSC classification method to linear discriminant analysis (LDA) is explored in detail in Tibshirani et al. [Tibshirani et al., 2003]. NSC classification uses a discriminant score similar to that of LDA, but it differs by assuming a diagonal covariance matrix and applying shrinkage to the centroids. A similar analogy can be made between mixture discriminant analysis (MDA: [Hastie and Tibshirani, 1996]) and the two-class classification procedure we introduced in the previous section.

Mixture discriminant analysis also defines the class posterior probabilities as in equation (2.9), but here we make a stronger assumption of having a diagonal covariance matrix in the class conditional densities. In addition, we use the shrunken centroids that are obtained by soft thresholding for each subclass. MDA uses the EM algorithm in order to estimate which subclasses the samples belong to and to estimate the class means and covariances. However, we now know the subclass label for each sample, and we can take the sample means and variances to estimate the centroids for the subclasses and the covariance matrix. It can be said that we have more information than in regular MDA settings.

3 Algorithms for Hierarchical Classification

We have introduced a modification of the NSC classifier for two-class problems where the classes are mixtures of subclasses. We now propose several methods to build the hierarchical structure: we either use the classifiers to train the hierarchy, or build the classifiers after having provided a multi-class layout.

3.1 Bottom-up merge based on the class confusion matrix

First, a K-class problem is decomposed into $\binom{K}{2}$ two-class problems. Instead of training a multi-class classifier at once, we apply NSC classification to each of the $\binom{K}{2}$ possible pairs of classes. They will have different *optimal* shrinkage thresholds (Δ) and thus different sets of significant genes. At their chosen

shrinkage levels, the cross-validation errors are computed as estimates of prediction errors. Then the pair with the largest error is chosen and combined into a new, larger class. This determines the bottom-most join of the hierarchy. Having $(K-1)$ classes, it is now decomposed into $\binom{K-1}{2}$ two-class classifications. We now apply the mixture version of the NSC classifier, introduced in section 2.2, to the new class (of two subclasses) and each of the remaining $(K-2)$ single classes to obtain the cross-validation errors. At this stage, we only use the genes selected to be significant in the previous stage (pairwise merge). For instance, when training the mixture version of the NSC classifier on the mixture of classes K_1 and K_2, and a single class K_3, we begin with the union of the significant genes from the three pairs: (K_1,K_2), (K_2,K_3), and (K_1,K_3). Thus, we continue using the information we have stored from the bottommost series of the classifiers. The next newly-formed class, again with the largest cross-validation error, could be a combination of the previous new group and another single class, or a new combination of two single classes. These steps are repeated $(K-1)$ times, until we have the largest group, that of all K classes put together.

As we have seen, the hierarchical structure is based on the errors that occur when classifying pairs of classes. In other words, we define two classes to be the closest pair when the two are the most confused by the classifier. Along with our procedures to yield the errors, we automatically train many classifiers with one of them chosen to represent each level of hierarchy. Therefore, at the end of the algorithm, we have a two-class classifier trained at each level, and each one uses a unique subset of genes. We not only know the relative affinity of the classes but also have gene groups important for different combinations of the classes.

When we have a new sample to predict, we apply the topmost classifier, which gives it a pair of posterior probabilities. Then, we continue applying the classifiers in order toward the bottom, updating the posterior probabilities (by multiplication) as we go. The series of posterior probabilities are multiplied along every path on the tree, thereby achieving K different products. These values are the final estimates of posterior probabilities for the K classes. The sample is predicted to belong to the class with the largest posterior probability. Since we have the estimated error rate at each level, we know how certain each prediction would be.

3.2 Clustering based on the confusion matrix from the NSC classification

Instead of training all possible pairwise classifiers at each level of hierarchy in order to build the structure, the confusion matrix idea can be developed in a simpler way. The regular NSC classification applied to all K classes yields a confusion matrix, which can be used for a hierarchical clustering of the classes. Once an NSC classifier is trained on all the observations, we can define a distance measure between any two groups of the classes:

$$D(i,j) = 1 - \frac{e_{ij} + e_{ji}}{n_i + n_j}, \tag{3.1}$$

where e_{ij} is the number of cross-validation errors such that the observations in class i are misclassified to class j, and n_i is the number of observations belonging to class i. The bottom-up hierarchical clustering of K classes can be done using this measure as a metric.

Having built the hierarchy, we train a two-class version of the NSC classifier introduced in 2.2, at each non-terminal node. The class prediction for a future sample is done the same way as in the previous algorithm (section 3.1). In fact, these methods of training $(K-1)$ classifiers at non-terminal nodes and predicting on future instances can be applied exactly the same, once a hierarchical structure of the classes is set by some procedure. We present two other ways of building the structure.

3.3 Bottom-up merge based on the inter-class distances

An alternative way to define the closeness between classes is based on the average distance between the samples of the two classes. When euclidean distance is used with standardized samples, it is equivalent to considering the average correlation between the samples. Average euclidean distance between the samples of classes k_1 and k_2 is

$$D(k_1,k_2) = \frac{1}{n_{k_1} n_{k_2}} \sum_{i:y_i=k_1} \sum_{i':y_{i'}=k_2} d(x_i, x_{i'}), \tag{3.2}$$

where $d(x_i, x_{i'}) = \sqrt{\sum_{j=1}^{p} (x_{ji} - x_{ji'})^2}$.

We first compute $\binom{K}{2}$ distance measures, one from each pair of single classes. Among them, the least distant pair is chosen and combined into a new class. Then we measure the distances between the new class and the other $(K-2)$ single classes, and compare to the average distances between single classes

already computed. The next chosen pair can be either a combination of the new class and another single class, or that of two single classes. We continue the bottom-up process until it gets to the top, where all K classes are combined. A total of $(K-1)$ merges occur.

3.4 Top-down split based on the inter-class distances

In addition to the bottom-up method, the inter-class distances can be used in a similar manner to build the hierarchy from the top. In order to find the best binary split from the group of all K classes, the inter-class distance needs to be computed for every possible binary split; there are $2^{K-1}-1$ different ways of dividing the K classes into two. This number being prohibitively large when K is in order of ten, we propose building a top-down hierarchy of the classes using the 2-means clustering algorithm.

Applying the 2-means algorithm to the observations in the topmost node separates the observations into two clusters, such that their within cluster point scatter is minimized. However, the regular 2-means algorithm does not necessarily keep the samples of a class in the same node; hence we add one more step so that this property of the hierarchical classification is preserved while training. Once the individual assignments are made, samples of each class take a majority vote to be reassigned to the winning cluster. This scheme is similar to the 2-means on the centroids, but has an advantage of taking into account the class sizes and distributions.

3.5 Comparison

We have suggested various approaches to forming the hierarchical frame for classification. Some of them make use of the confusion matrix, while others define the homogeneity of the classes in terms of distances among the samples. The different criteria produce hierarchical structures that could be dissimilar. Note that when two classes are close in terms of their euclidean distance, the nearest shrunken centroid classifier is not necessarily trained with a high prediction error, and vice versa. We consider the first method (bottom-up merge based on the confusion) to be the most reasonable and biologically meaningful. Also, it will be shown in the next section that the first algorithm yields the most accurate classifier. The followings are remarks that compare different aspects of the algorithms, some of them commenting on why the first method is favored.

- Screening the genes : Using the first algorithm (bottom-up merge based on the confusion), as stated earlier, we may begin with a smaller set of genes when training classifiers at the second level or above. We only keep the genes that have already been selected to be significant from the pairwise merges. This has an effect of eliminating the noisy genes even before the high level (the second or above) classifiers are fitted. We saw from experiments that this pre-screening of the genes brings a decrease in the total number of genes used, not hurting the accuracy.
- Monotonicity of the errors : Building the hierarchical structure based on the errors gives a useful property: monotonicity of the errors. Each of the $(K-1)$ classifiers has an error rate in binary prediction, and using the errors in building the hierarchy ensures that the classifiers trained earlier, thus closer to the bottom, have greater error rates than those closer to the top. In other words, the hierarchical classifier is less likely to cause prediction errors near the top than through the bottom classifiers. Since our software uses the cross-validation to estimate the prediction errors and to choose the optimal shrinkage levels, some randomness is introduced. The error *estimates* could be non-monotonic, although they should fluctuate only by a small amount. It is intuitively appropriate to have smaller error rates when categorizing at higher levels than finer levels. Shedden et al. [Shedden et al., 2003] have used a hierarchical classification method on gene expression data for human cancers as well. Their study concludes that the *error depth* is one of the crucial factors for understanding how various tumors differ from one another.
- Monotonicity of the inter-group distances : When the inter-class distances are used to form the hierarchy, the prediction errors of the classifiers are not necessarily monotonically decreasing as they go bottom-up. Instead, the average distance between the samples of the two classes increases monotonically. Thus this scheme retains the general rule in clustering, that of clustering samples together when their euclidean distances are small.
- Computation issue : Although the bottom-up merge using the confusion carries many appealing properties, it has a drawback of being quite costly. Other algorithms take less computation because they only require training $(K-1)$ classifiers. However, the computations bring in return the desired properties mentioned so far and an improvement in accuracy.
- Mixture modeling vs. non-mixture modeling : Instead of modeling the groups of classes as Gaussian

Label	♯ of samples	Label	♯ of samples	Label	♯ of samples
Bladder	6	Liver	6	Prostate	12
Breast	22	Lung	47	Soft	31
CNS	30	Lymph	8	Stomach	18
Colon	7	Ovary	21	Testis	6
Kidney	40	Pancreas	7		

Table 1: The gene expression dataset of 261 *samples and* 4718 *genes is used. The table shows the number of samples with given labels.*

mixtures, we could simply ignore the fact that the groups are made up of multiple classes. In this case, we would simply fit a regular two-class NSC classifier at each node. We tried using the non-mixture version of classifiers with all the algorithms above. In general, the non-mixture version would fit the entire classifier using fewer predictors (genes), but pay a price of being inaccurate. Assigning a single centroid to a mixture of classes oversimplifies the data structure, resulting in impreciseness.

4 Microarray Data Example

We apply the following classification methods to a microarray data set: NSC classification, hierarchical scheme using errors (bottom-up merge, and clustering using confusion matrix from the NSC), and the hierarchical scheme using inter-class distances (bottom-up merge, and top-down split using the 2-means algorithm). We show the results of fitting the mixture version and the non-mixture version of binary classifiers. The following gene expression dataset has been used by Munagala et al. [Munagala et al., 2004]. It is composed of 261 samples of 4718 genes, each sample labeled with one of 14 different cancer types. *Table* 1 shows the number of samples with each label.

Table 2 summarizes the training errors, 10-fold cross-validation errors, and the total number of genes used in future prediction from various methods when the binary classifiers are trained under mixture modeling. With the hierarchical methods, for every fold of the cross-validation, the remaining 90% of the data were used to build a hierarchical structure with the internal set of cross-validation errors

Method	Training error	CV error (s.e.)	Number of genes
NSC	0.049	0.065 (0.016)	4327/4718
Bottom-up merge using errors	0.042	0.080 (0.018)	3500/4718
Clustering using errors from NSC	0.030	0.114 (0.015)	4256/4718
Bottom-up merge using distances	0.061	0.111 (0.018)	4272/4718
Top-down split using 2-means	0.072	0.092 (0.019)	4718/4718

Table 2: (Mixture modeling) Comparison of training errors and estimates of prediction errors: With the hierarchical methods, for every fold of the cross-validation, the remaining 90% *of the data were used to build a hierarchical structure with the internal set of cross-validation errors (two-layered cross-validation). The bottom-up merge using the errors yields the lowest training and cross-validation errors among all the hierarchical methods. It also selects the smallest number of genes for future prediction*

(two-layered cross-validation). The bottom-up merge using the errors yields the lowest cross-validation error among all the hierarchical methods. This classifier and the regular NSC classifier have the cross-validation errors within one standard error of each other. When the classes are hierarchically clustered using errors from the NSC classifier, it yields an error rate larger than that from the bottom-up merge using errors, which benefits from the pre-screening of the genes. The top-down split using 2-means yields a smaller error rate than the bottom-up merge using distances. As for the number of genes for the future prediction, the bottom-up merge using the errors selects the smallest set of genes; it is mainly due to the noisy gene elimination process described in sections 3.1 and 3.5. Given that this method has the lowest error rate, selecting fewer genes is definitely preferable.

Table 3 compares the hierarchical classification methods based on different algorithms, but with non-mixture modeling used at each split. With a dataset containing so many classes, non-mixture modeling tends to oversimplify the data structure. This method defines a single centroid for each mixture even when the mixture is spread asymmetrically in many directions. Although the non-mixture modeling allows us to reduce the number of genes for prediction, it is less accurate than in mixture modeling. For all four pairs of hierarchical methods with mixture and non-mixture modeling (in *Tables* 2 and 3), training errors and cross-validation errors are smaller in the former, and for some of the methods, the cross-validation errors double in the latter.

Figure 1, from the regular nearest shrunken centroid fit, shows 10-fold cross-validation and training

Method	Training error	CV error (s.e.)	Number of genes
Bottom-up merge using errors	0.099	0.195 (0.022)	1305/4718
Clustering using errors from NSC	0.080	0.222 (0.030)	1910/4718
Bottom-up merge using distances	0.095	0.126 (0.018)	1981/4718
Top-down split using 2-means	0.118	0.179 (0.027)	3264/4718

Table 3: (Non-mixture modeling) Comparison of training errors and estimates of prediction errors: With the hierarchical methods, for every fold of the cross-validation, the remaining 90% *of the data were used to build a hierarchical structure with the internal set of cross-validation errors (two-layered cross-validation). Although the non-mixture modeling allows us to reduce the number of genes for prediction, it is less accurate than in mixture modeling. For all four pairs of hierarchical methods with mixture and non-mixture modeling (in Tables* 2 *and* 3*), training errors and cross-validation errors are smaller in the former, and for some of the methods, the cross-validation errors double in the latter.*

errors at different thresholds, adjusting the number of genes selected for the future prediction. The shrinkage level that yields the lowest cross-validation error leaves 4327 genes. Only a small fraction of d_{jk} are set to zero for all classes; many of the remaining are close to zero. Still, quite a few dimensions of the centroids remain distinct among classes, since there are 14 classes involved. By adopting hierarchical methods, we can now determine which genes are important in distinguishing smaller sets of classes.

Figure 2 gives the hierarchical structures obtained from the two methods using the confusion matrix. The upper tree is from the bottom-up merge; the lower one is from clustering using the errors from the NSC classifier. The next pair of plots (*Figure* 3) is from the methods based on inter-class distances, the bottom-up merge and the top-down split, respectively. The decimals on the tree represent the estimates of prediction errors at those splits; the integers next to the decimals are the number of genes used in prediction. Although the number of genes increases as the level goes up, it is possible to extract the newly selected genes (that have not been used at lower levels) and characterize them to be particularly significant at the level.

When comparing the class layouts in *Figures* 3 and 4, they are not identical since the two use different criteria in defining affinity of the classes. We find that the cross-validation errors are monotonic for the bottom-up merge using errors. As for the bottom-up merge using distances, we have the measure of average distances between any two groups of classes that are used as heights at which the class labels are hung.

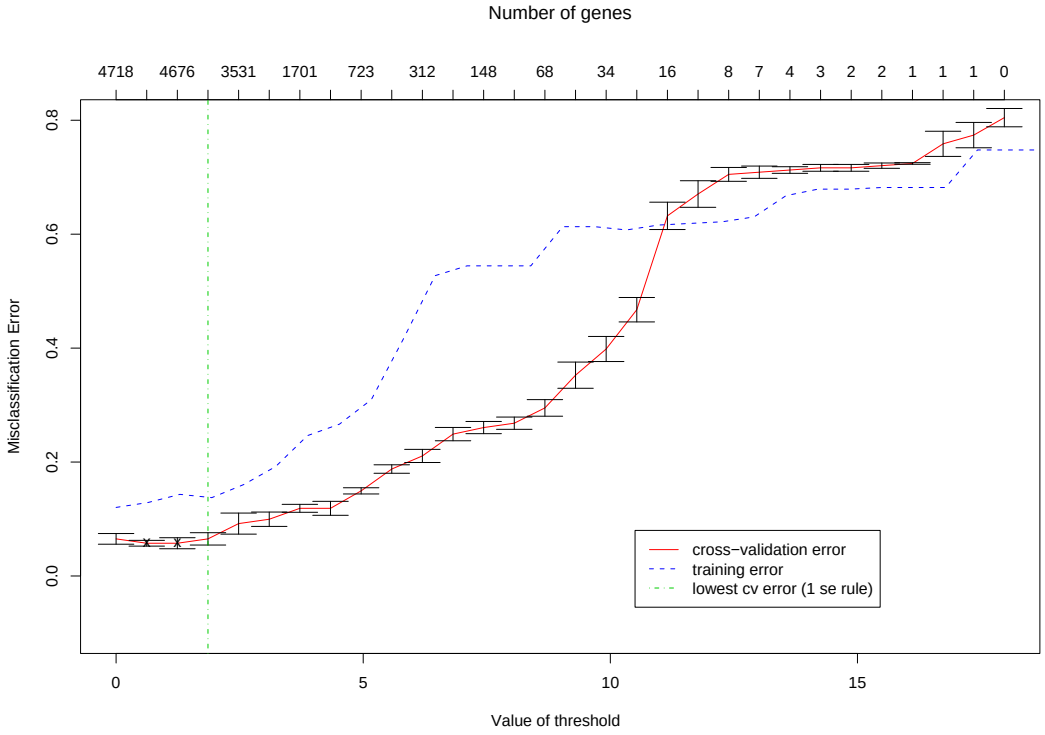

Figure 1: Plot of 10-fold cross-validation and training errors from nearest shrunken centroid fit : 4327 *out of* 4718 *genes are selected at shrinkage parameter* $\Delta = 1.859$.

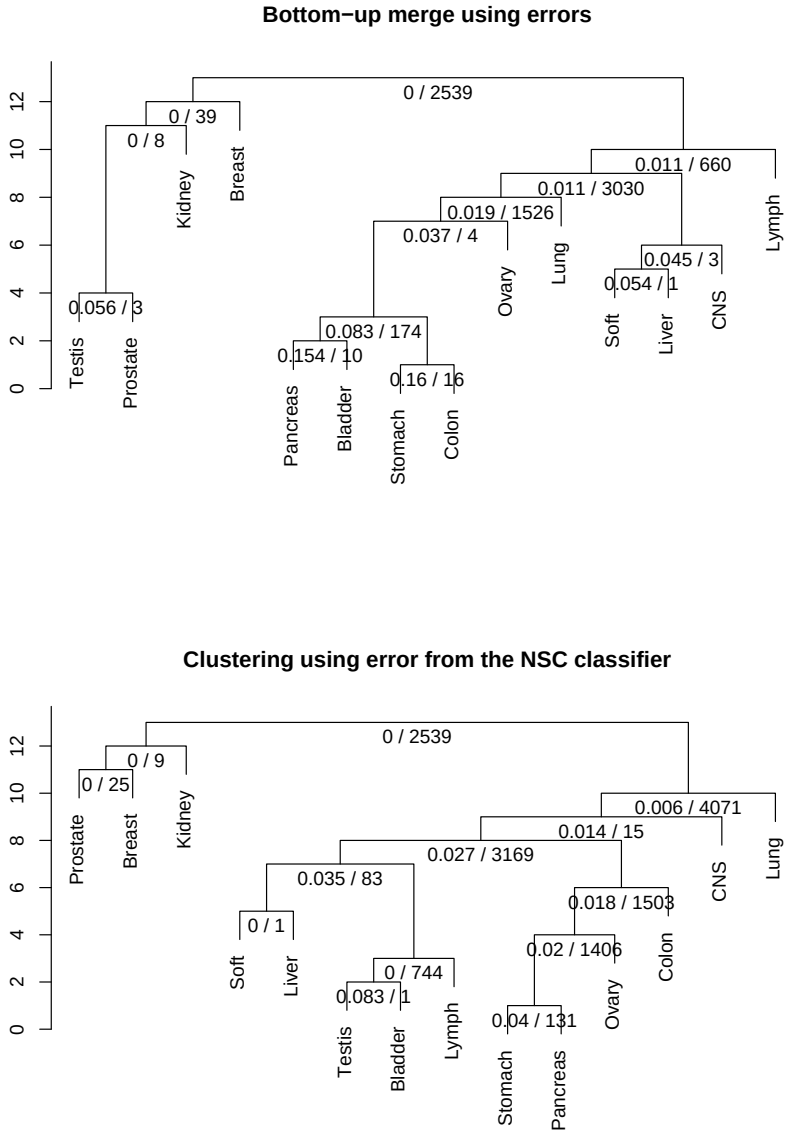

Figure 2: The hierarchical structures obtained from the two methods using the confusion matrix. The upper one is from the bottom-up merge, and the bottom one is from clustering using the errors from the NSC classifier. The decimals on the tree represent the estimates of prediction errors at those splits, and the integers next to the decimals are the number of genes used in prediction. The cross-validation errors are monotonic for the bottom-up merge using errors.

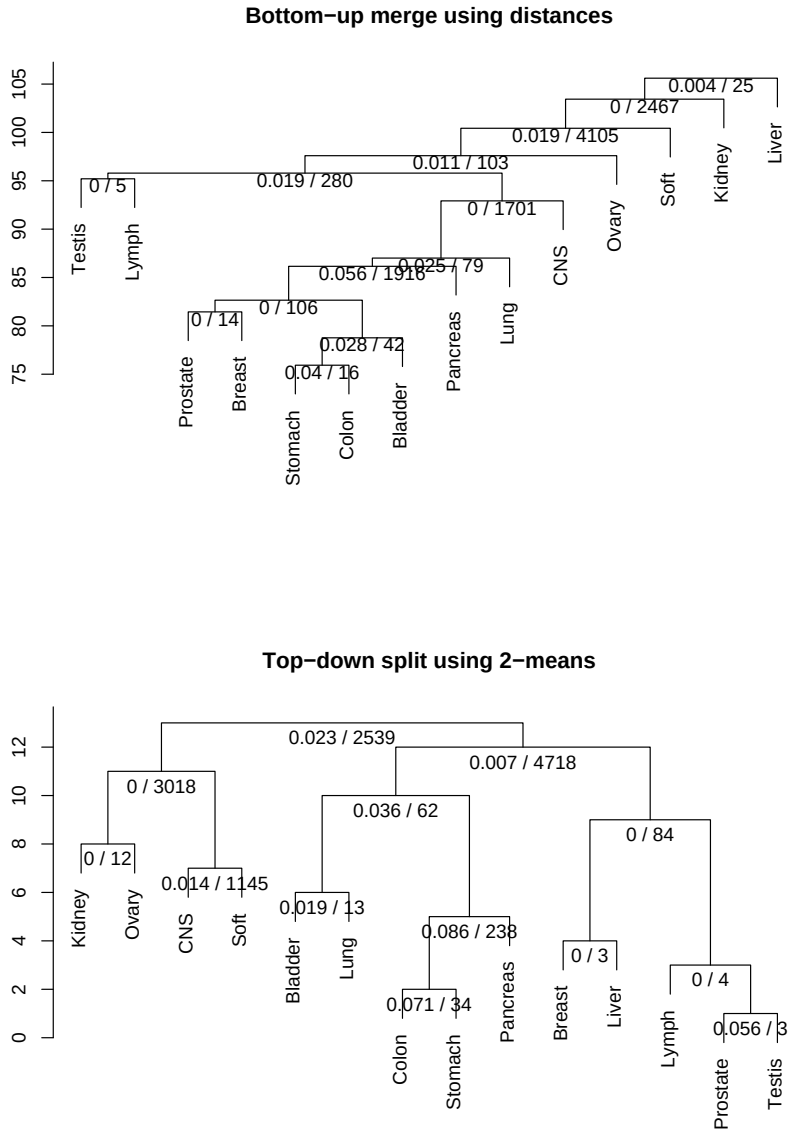

Figure 3: The hierarchical structures obtained from the two methods using the inter-class distances. The upper one is from the bottom-up merge, and the lower one is from the top-down split. The decimals on the tree represent the estimates of prediction errors at those splits, and the integers next to the decimals are the number of genes used in prediction. The average distances between the groups are used as heights for the bottom-up merge method.

5 Related work

The hierarchical classification method we proposed share several important components with Shedden et al. [Shedden et al., 2003]; they build a pathological tree, having different kinds of human cancers at the terminal nodes. Most importantly, Shedden et al. emphasize that gene selection is an essential part in learning with gene expression data. In the training process, they first fix the hierarchical structure of the classes based on their biological relationship, then pre-select a set of significant genes for each node. When predicting on future data, the nearest neighbor decisions are made at sequential nodes, using the chosen gene lists. Instead of this multi-step procedure, we have suggested a method to build the hierarchical frame, learn the classification rule, and select the significant genes at the same time.

Just in the way we used the NSC classifier, there have been various attempts to feed preexisting classifiers to different hierarchical schemes. Jordan et al. [Jordan and Jacobs, 1994], quoting the phrase divide-and-conquer, propose the hierarchical mixtures of experts (HME) model that uses the generalized linear models (GLM) for gating networks (non-terminal nodes) and expert networks (terminal nodes). Similarly, Titsias et al. [Titsias and Likas, 2002] also propose a mixture of experts classification using a hierarchical bayes classifier, in order to model the data efficiently when they are drawn from more than one cluster. Part of their purpose is dividing the model space into pieces and training different multi-class classifiers at the terminal nodes; the performance of each classifier is expected to improve from the single overall classifier. On the other hand, the hierarchical scheme we propose eventually has K terminal nodes, each one named with a class label. The internal nodes of the hierarchy not only facilitate the classification, but also show the class organization. Although the HME methods are similar to ours in making parametric assumptions on the data structure and estimating the parameters when training the data, they have a different optimizing method. They use the EM algorithm to simultaneously find the parameters for all levels of the hierarchy. We estimate the parameters (including the tuning parameters for gene filtering) characterizing the data structure at each node of the hierarchy separately. Due to this independent optimization, having as many levels as needed is not computationally prohibited, in constrast to other methods, which apply a global fit. Also, finding a value for the shrinkage parameter independently at each level is meaningful for our purpose of selecting the most significant set of genes for a particular set of classes.

Our scheme differs from some other hierarchical classifiers in that each split means dividing a group of classes into two groups, each containing non-overlapping class labels. Thus the training samples of the same class label are always put in a node. For example, our method differs from CART (Classification and Regression Trees), Breiman et al. [Breiman et al., 1984], in this respect. When CART is used for classification, a split does not necessarily separate samples into two distinct groups of classes. Also, the training samples of the same class could follow different paths on the tree while training, and one obvious reason for this is the top-down recursive partitioning. However, since we use a bottom-up method starting with entire classes as terminal nodes, we avoid them being scattered into different nodes. Regardless of this, CART is a flexible, interpretable, and fast classification method, and one of our main foundations for this study. Zhang et al. [Zhang et al., 2001] attempt to classify tumor types by recursive partitioning on genes, in a way similar to CART. Random forests, introduced by Breiman [Breiman, 2001], are trained by combining multiple trees based on randomly selected values from the training data. It has been applied with some modifications for microarray data, for example in Zhang et al. [Zhang et al., 2003]. These methods exhibit high accuracy on classification, but our method is better suited in extracting and grouping the important features and displaying the results for further interpretation. When CART is used on microarray data, having a relatively small number of samples, it runs out of data before using many genes, and thus only discovers a few important variables. Using random forests can lead to a more diverse variable selection, but having multiple trees makes it hard to grasp the relationship among classes simultaneously.

6 Discussion

We have introduced a way to learn a classifier in a hierarchical fashion. The classifier is built through intermediate steps where several nearest shrunken centroid classifiers are trained. Although this process requires more work than a single classifier, the computations are worthwhile. Each split of the hierarchical structure is used to separate a group of classes into two, thereby revealing how multiple classes are organized in terms of their relative similarity. In addition to clarifying the hierarchical relationship among classes, the prediction on future instances offers more information than in the case of a flat

classifier. We know the expected error rate at each step, and believing that the top classifiers give reliable predictions, we can limit the possible misclassifications. As pointed out previously, embedded feature selection takes place when training the data, which enables different sets of important genes to be involved in prediction at different levels.

Our use of hierarchical PAM allows us to go one step further in several directions. To extend its relationship to MDA, we can further divide each class into multiple clusters within itself and apply our procedure at a lower level. We, in general, expect that the clusters from the same classes will merge earlier than those from two different classes, although it might not always be true.

The steps of building a hierarchical structure shown in section 3 can be applied with all other classifiers. For instance, we can fit a logistic regression as a binary classifier at each non-terminal node; the hierarchical structure can be built based on either the confusion matrix or the inter-class distances. Since the logistic regression estimates the posterior probabilities for the daughter nodes, the series of probabilities can be multiplied to yield the final estimates for all K classes. In case of using the support vector machine (SVM) for binary classifications, we need a way to convert the SVM fit in order to estimate the posterior probabilities, since the generic fit yields the margins rather than the probabilities.

When a series of generic classifiers are used in a hierarchical scheme, each binary classification can be more accurate than an overall multi-class classification, thereby having a relatively small bias at each level. However, building multiple classifiers, and that using smaller data, can increase the variance. This phenomenon can be alleviated by applying appropriate smoothing or feature selection at different levels. In other words, each binary classifier needs to be accurate enough to find a reasonable boundary between any groups of classes, but it has to accompany a tuning process for variance reduction. The shrinkage technique is playing the rule in our method, and it is even more useful for interpretation since it creates the significant gene lists at the same time.

References

[Breiman, 2001] Breiman, L. (2001). Random forests. Technical report, University of California Berkeley.

[Breiman et al., 1984] Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth, Belmont.

[Eisen et al., 1998] Eisen, M., Spellman, P., Brown, P., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, pages 95:14863–14868.

[Hastie and Tibshirani, 1996] Hastie, T. and Tibshirani, R. (1996). Discriminant analysis by gaussian mixtures. *Journal of Royal Statistical Society, series B*, pages 58:155–176.

[Hastie et al., 2001] Hastie, T., Tibshirani, R., and Friedman, J. (2001). *Elements of Statistical Learning; Data Mining, Inference, and Prediction*. Springer-Verlag, New York.

[Jordan and Jacobs, 1994] Jordan, M. and Jacobs, R. (1994). Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, pages 6:181–214.

[McLachlan, 1992] McLachlan, G. (1992). *Discriminant Analysis and Statistical Pattern Recognition*. Wiley, New York.

[Munagala et al., 2004] Munagala, K., Tibshirani, R., and Brown, P. (2004). Cancer characterization and feature set extraction by discriminative margin clustering. *BMC Bioinformatics*, pages 5:21–35.

[Shedden et al., 2003] Shedden, K., Taylor, J., Giordano, T., Kuick, R., Misek, D., Rennert, G., Schwartz, D., Gruber, S., Logsdon, C., Simeone, D., Kardia, S., Greenson, J., Cho, K., Beer, D., Fearon, E., and Hanash, S. (2003). Accurate molecular classification of human cancers based on gene expression using a simple classifier with a pathological tree-based framework. *American Journal of Pathology*, pages 163(5):1985–1995.

[Speed, 2003] Speed, T. (2003). *Statistical Analysis of Gene Expression Microarray Data*. Chapman & HallCRC, London.

[Tibshirani et al., 2003] Tibshirani, R., Hastie, T., Narasimhan, B., and Chu, G. (2003). Class prediction by nearest shrunken centroids, with application to dna microarrays. *Statistical Science*, pages 18(1):104–117.

[Titsias and Likas, 2002] Titsias, M. and Likas, A. (2002). Mixture of experts classification using a hierarchical mixture model. *Neural Computation*, pages 14(9):2221–2244.

[Zhang et al., 2003] Zhang, H., Yu, C.-Y., and Singer, B. (2003). Cell and tumor classification using gene expression data: Construction of forests. *Proceedings of the National Academy of Sciences*, pages 100:4168–4172.

[Zhang et al., 2001] Zhang, H., Yu, C.-Y., Singer, B., and Xiong, M. (2001). Recursive partitioning for tumor classification with gene expression microarray data. *Proceedings of the National Academy of Sciences*, pages 98:6730–6735.