Tissue Classification with Gene Expression Profiles

Amir Ben-Dor * — U. Washington

Laurakay Bruhn — Agilent Laboratories

Nir Friedman — Hebrew University

Iftach Nachman — Hebrew University

Michèl Schummer — U. Washington

Zohar Yakhini — Agilent Laboratories

Abstract

Constantly improving gene expression profiling technologies are expected to provide understanding and insight into cancer related cellular processes. Gene expression data is also expected to significantly aid in the development of efficient cancer diagnosis and classification platforms. In this work we examine two sets of gene expression data measured across sets of tumor and normal clinical samples. One set consists of 2,000 genes, measured in 62 epithelial colon samples [1]. The second consists of $\approx$ 100,000 clones, measured in 32 ovarian samples (unpublished, extension of data set described in [26]).

We examine the use of scoring methods, measuring separation of tumors from normals using individual gene expression levels. These are then coupled with high dimensional classification methods to assess the classification power of complete expression profiles. We present results of performing *leave-one-out cross validation* (LOOCV) experiments on the two data sets, employing SVM [8], AdaBoost [13] and a novel clustering based classification technique. As tumor samples can differ from normal samples in their cell-type composition we also perform LOOCV experiments using appropriately modified sets of genes, attempting to eliminate the resulting bias.

We demonstrate success rate of at least 90% in tumor vs normal classification, using sets of selected genes, with as well as without cellular contamination related members. These results are insensitive to the exact selection mechanism, over a certain range.

1 Introduction

The process by which the approximately 100,000 genes encoded by the human genome are expressed as proteins involves two steps. DNA sequences are initially transcribed into mRNA sequences. These mRNA sequences in turn are translated into the amino acid sequences of the proteins that perform various cellular functions. A crucial aspect of proper cell function is the regulation of gene expression process, so that different cell types express different subsets of genes. Measuring mRNA levels can provide a detailed molecular view of the subset of genes expressed in different cell types under different conditions. Recently developed array-based methods enable simultaneous measurements of the expression levels of thousands of genes. These measurements are made by quantitating the hybridization (detected for example, by fluorescence) of cellular mRNA to an array of defined cDNA or oligonucleotide sequences immobilized on a solid substrate. Array methodologies have led to a tremendous acceleration in the rate at which gene expression pattern information is accumulated [9, 16, 17, 19, 29]. Measuring gene expression levels under different conditions is important for expanding our understanding of gene function, how various gene products interact, and how experimental treatments can affect cellular function.

Gene expression data can help in better understanding of cancer. Normal cells can evolve into malignant cancer cells through a series of mutations in genes that control the cell cycle, apoptosis, and genome integrity, to name only a few. As determination of cancer type and stage is often crucial to the assignment of appropriate treatment [14], a central goal of the analysis of gene expression data is the identification of sets of genes that can serve, via expression profiling assays, as classification or diagnosis platforms.

Another important purpose of gene expression studies is to improve understanding of cellular responses to drug treatment. Expression profiling assays performed before, during and after treatment, are aimed at identifying drug responsive genes, indications of treatment outcomes, and at identifying potential drug targets [7]. More generally, complete profiles can be considered as a potential basis for classification of treatment progression or other trends in the evolution of the treated cells.

Data obtained from cancer related gene expression studies typically consists of expression level measurements of thousands of genes. This complexity calls for data analysis methodologies that will efficiently aid in extracting relevant biological information. Previous gene expression analysis work emphasizes clustering techniques, which aim at partitioning the set of genes into subsets that are expressed similarly across different conditions. Indeed, clustering has been demonstrated to identify functionally related families of genes [2, 9, 6, 11, 15, 29]. Similarly, clustering methods can be used to divide a set of cell samples into clusters based on their expression profile. In [1] this approach was applied to a set of colon samples which was divided into two groups, one containing mostly tumor samples, and the other containing mostly normal tissue samples.

* Contact author. Email: amirbd@cs.washington.edu.

Clustering methods, however, do not use any tissue annotation (e.g., tumor vs.normal) in the partitioning step. This information is only used to asses the success of the method. Such methods are often referred to as *unsupervised*. In contrast, *supervised* methods, attempt to predict the classification of new tissues, based on their gene expression profiles after training on examples that have been classified by an external "supervisor".

The purpose of this work is to rigorously assess the potential of classification approaches based on gene expression data. We present a novel clustering based classification methodology, and apply it together with two other recently developed classification approaches, *Boosting* [23, 13] and *Support Vector Machines* [8, 28] to two data sets. Both sets involve corresponding tissue samples from tumor and normal biopsies. The first is the data set of colon cancer [1], and the other is a data set of ovarian cancer (an extension of the data set reported in [26]). We use established statistical tools, such as *leave one out cross validation* (LOOCV), to evaluate the predictive power of these methods in the data sets.

One of the major challenges of gene expression data is the large number of genes in the data sets. For example, one of our data sets includes almost 100,000 clones. Many of these clones are not relevant to the distinction between cancer and tumor and introduce noise in the classification process. Moreover, for diagnostic purposes it is important to find small sets of genes that are sufficiently informative to distinguish between cells of different types. To this end we suggest a simple combinatorial error rate score for each gene, and use this method to select informative genes. As we show, selecting relatively small subsets of genes can drastically improve the performance. Moreover, this selection process also isolates genes that are potentially intimately related to the tumor makeup and the pathomechanism.

To realistically assess the performance of such methods one needs to address the issue of *sample contamination*. Tumor and normal samples may dramatically differ in terms of their cell-type composition. For example, in the colon cancer data [1], the authors observed that the normal colon biopsy also included smooth muscle tissue from the colon walls. As a result, smooth muscle related genes showed high expression levels in the normal samples compared to the tumor samples. This artifact, if consistent, could contribute to success in classification. To eliminate this effect we remove the muscle specific genes and observe the effect on the success rate of the process.

The rest of the paper is organized as follows. In Section 2, we describe the principle classification methods we use in this study. These include two state of the art methods from machine learning, and a novel approach based on clustering algorithm of [2]. In Section 3, we describe the two data sets, the LOOCV evaluation method, and evaluate the classification methods on the two data sets. In Section 4 we address the problem of gene selection. We propose a simple method for selecting informative genes and evaluate the effect of gene selection on the classification methods. In Section 5, we examine the effect of sample contamination on possible classification. We conclude in Section 6 with a discussion of related works and future directions.

2 Classification Methods

In this section, we describe the main classification methods that we will be using in this paper. We start by formally defining the classification problem. Assume that we are given a *training set* D, consisting of pairs $\langle x_i, l_i \rangle$, for $i = 1, \ldots, m$. Each *sample* x_i is a vector in $\mathbf{R}^N$ that describes expression values of N genes/clones. The *label* l_i associated with x_i is either -1 or $+1$ (for simplicity, we will discuss two-label classification problems). A classification algorithm is a function f that depends on two arguments, the training set D, and a query $x \in \mathbf{R}^N$, and returns a predicted label $\hat{l} = f_D(x)$. We also allow for no classification to occur if x is either close to none of the classes or when it is too borderline for a decision to be taken. Formally, this is realized by allowing $\hat{l}$ to be -1 , $+1$ or 0, the latter representing an unclassified query. Good classification procedures predict labels that typically match the "true" label of the query. For a precise definition of this notion in the absence of the unclassified option assume that there is some (unknown) joint distribution $P(x, l)$ of expression patterns and labels. The *error* of a classification function $f_D(\cdot)$ is defined as $P(f_D(x) \neq l)$. Of course, since we do not have access to $P(\cdot)$, we cannot precisely evaluate this term and use estimators instead. When unclassified is accepted as a possible output one needs to consider the costs/penalties of the various outcomes in analyzing the value of a classification method. For a comprehensive discussion of classification problems see [3, 10, 22].

2.1 Nearest Neighbor Classifier

One of the simplest classification algorithms is the *nearest neighbor* classifier [10]. The intuition is simple. To classify a query x, find the most similar example in D and predict that x has the same label as that example. To carry out this algorithm we need to define a similarity measure on expression patterns. In our experiments, we use the *Pearson correlation* as a measure of similarity (see, e.g., [11]).

Formally, let

$$k_p(x, y) = \frac{1}{\sqrt{Var[x]\,Var[y]}} \sum_i (x_i - E[x])(y_i - E[y])$$

be the Pearson correlation between two vectors of expression levels. Given a new vector x, the nearest neighbor classification procedure searches for the vector x_i in the training data that maximizes $k_p(x, x_i)$, and returns l_i, the label of x_i.

This simple non-parametric classification method does not take any global properties of the training set into consideration. However, it is surprisingly effective in many types of classification problems. We use it in our analysis as a strawman, to which we compare the more sophisticated classification approaches.

2.2 Using Clustering for Classification

Recall that clustering algorithms, when applied to expression patterns, attempt to partition the set of elements into clusters of patterns, so that all the patterns within a cluster are similar to each other, and different from patterns in other clusters. This suggests that if the labeling of patterns is correlated with the patterns, then unsupervised clustering of the data (labels not taken into account) would cluster patterns with the same label together and separate patterns with different labels. Indeed, such a result is reported by Alon et al. [1] in their analysis of colon cancer. Their study (which we describe in more detail in Section 3) involves gene expression patterns from colon samples that include both tumors and normal tissues. Applying a hierarchical clustering procedure to the data, Alon et al. observe that the topmost division in the dendrogram divides samples into two groups, one predominantly tumor, and the other predominantly normal. This suggests that for some types of classification problems, such as tumor vs. normal, clustering can distinguish between labels. Following this intuition, we build a clustering based classifier. We first describe the underlying clustering algorithm and then present the classifier.

2.2.1 The clustering algorithm

The CAST algorithm, implemented in the BioClust analysis software package [2], takes as input a threshold parameter t, which controls the granularity of the resulting cluster structure, and a similarity measure between the tissues.[1] We say that a tissue v has *high similarity* to a set of tissues $\mathcal{C}$, if the average similarity between v and the tissues in $\mathcal{C}$ is at least t. Otherwise, we say that v has *low similarity* to $\mathcal{C}$. CAST constructs the clusters one at a time, and halts when all tissues are assigned to clusters. Intuitively, the algorithm alternates between adding high similarity tissues to $\mathcal{C}$, and removing low similarity tissues from it. Eventually, all the tissues in $\mathcal{C}$ have high similarity to $\mathcal{C}$, while all the tissues outside of $\mathcal{C}$ have low similarity to $\mathcal{C}$. At this stage the cluster $\mathcal{C}$ is closed, and a new cluster is started (See [2] for complete description of the algorithm).

Clearly, the threshold value t, has great effect on the resulting cluster structure. As t increases, the clusters formed are smaller. At the extreme case, if t is high enough, each tissue would form a singleton cluster. Similarly, as t decreases, the clusters tend to get larger. If t is low enough, all tissues are assigned to the same cluster.

2.2.2 Clustering based classification

As described above, the threshold parameter t determines the cohesiveness and the number of the resulting clusters. A similar situation occurs in other clustering algorithms. For example, in hierarchical clustering algorithms (e.g., [1, 11]), the cutoff "level" of the tree controls the number of clusters. In any clustering algorithm, it is clear that attempting to partition the data into exactly two clusters will not be the optimal choice for predicting labels. For example, if the tumor class consists of several types of tumors, then the most noticeable division into two clusters might separate "extreme" tumors from the milder ones and the normal tissues, and only a further division will separate the normals from the milder tumors.

For the purpose of determining the right parameter to be used in clustering data that contains some labeled samples we propose a measure of cluster structure *compatibility* with a given label assignment. The intuition is simple: on the one hand, we want clusters to be uniformly labeled and therefore penalize pairs of samples that are within the same cluster but have different labels; on the other hand, we do not want to create unnecessary partitions and therefore penalize pairs of samples that have the same label, but are not within the same cluster.

Formally, we define the *compatibility* score of a cluster structure with the training set as the sum of two terms. The first is the number of tissue pairs (v, u) such that v and u have the same label, and are assigned to the same cluster. The second term is the number of (v, u) pairs that have different labels, and are assigned to different clusters. This score is also called the *matching coefficient* in the literature [12]. To handle label assignments defined only on a subset of the data we restrict the comparison to count pairs of examples for which labels are assigned (the matching coefficient for a submatrix is computed).

Using this notion, we can optimize, using a binary search, the choice of clustering parameters to find the most compatible clustering. That is: we consider different threshold values, t; use CAST to cluster the tissues; measure the compatibility $C(t)$ of the resulting cluster structure with the given label assignment; and finally, choose the clustering that has maximal $C(t)$. Thus, although the clustering algorithm is *unsupervised*, in the sense that it does not take into account the labels, we use a supervised procedure for choosing the clustering threshold. We also emphasize that this general idea can be applied to any parameter dependent clustering method, and is not restricted to our particular choice.

To classify a query sample we cluster the training data and the query, maximizing compatibility to the labeling of the training data. We then examine the labels of all elements of the cluster the query belongs to and use a simple majority rule to determine the unknown label. The intuition is that the query's label should agree with the prevailing label in its cluster. Various majority rules, taking into account statistical confidence can be used. When confidence is too low the query is labeled as *unclassified*. The stringency of this test determines the strictness of our classification rule. In the current experiment we use the most liberal rule, i.e. a query is unclassified only if there is an equal number of elements of each label in its cluster. The choice of majority rule depends on the cost of non-classification vs. the cost of misclassification.

2.3 Large-Margin Classifiers

The cluster-based approach we discussed in the previous section attempts to find inherent structure in the data (i.e., clusters of samples) and uses this structure for prediction. We can also use *direct* methods that attempt to learn a *decision surface* that separates the positive labeled samples from the negatively labeled samples.

The literature of supervised learning discusses a large number of methods that learn decision surfaces. These methods can be described by two aspects. First, the class of surfaces from which one is selected. This question is often closely related to the *representation* of the learned surface. Examples include linear separation (which we discuss in more detail below), decision-tree representations, and two-layer artificial neural networks. Second, the learning rule that is being used. For example, one of the simplest learning rules attempts to minimize the number of errors on the training set.

Application of direct methods in our domain can suffer from a serious problem. In gene-expression data we expect N, the number of measured genes, to be significantly larger than M, the number of samples. Thus, due to the large number of dimensions there are many simple decision surfaces that can separate the positive examples from the negative ones. This means that counting the number of training set errors is not restrictive enough to distinguish good decision surfaces from bad ones (in terms of their performance on examples not in the training set).

In this paper, we use two methods that received much

[1] In this work we use the Pearson correlation between gene expression profiles as the similarity measure. However, any similarity measure can be used.

recent attention in the machine learning literature. Both methods attempt to follow the intuition that classification of examples depends not only on the region they are in, but also on a notion of *margin*: how close are they to the decision surface. Classification of examples with small margins is not as confident as classification of examples with large margins. (Given slightly different training data, the estimated decision surface moves a bit, thus changing the classification of points which are close to it.) This reasoning suggests that we should select a decision surface that classifies all the training examples correctly with large margin. Following the same argument, given the learned decision surface and an unlabeled sample x, we can set a threshold on the margin of x for classification. If x is closer to the surface then the allowed threshold, we mark is as unclassified. Again, the threshold will depend on the relative costs of the different outcomes.

The basic intuition of large margin classifiers is developed in quite different manners in the following two approaches.

2.3.1 Support Vector Machines

Support vector machines (SVM) were developed in [8, 28]. A tutorial on SVMs can be found in [5]. The intuition for support vector machines is best understood in the example of linear decision rules. A linear decision rule can be represented by a hyperplane in R^N such that all examples on the one side of the hyperplane are labeled positive and all the examples on the other side are labeled negative. Of course, in sufficiently high-dimensional data we can find many linear decision rules that separate the examples. Thus, we want to find a hyperplane that is as far away as possible from all the examples. More precisely, we want to find a hyperplane that separates the positive examples from the negative ones, and also maximizes the minimum distance of the closest points to the hyperplane. This question can be posed as a quadratic program (see Appendix A), and can be solved efficiently. The resulting hyperplane can be written as a weighted sum of the training examples, x_i, and the classification of a new example x can be calculated using inner products with the example vectors, $\langle x, x_i \rangle$. This treatment can be generalized to deal with training sets that are not linearly separable. We refer the reader to [5] for details.

It is clear that linear hyperplanes are a restricted form of decision surfaces. One method of learning more expressive separating surfaces is to project the training examples (and later on queries) into a higher-dimensional space, and learn a linear separator in that space. For example, if our training examples are in R^1, we can project input values x to the vector (x, x^2). A linear separator in the projected space is equivalent to learning an interval in the original representation of the training examples.

Thus, we can fix a projection $\Phi : \mathbf{R}^N \mapsto \mathbf{R}^M$ to higher dimensional space, and get more expressive decision surfaces. In this case, the classification rule for x will be composed of the inner products $\langle \Phi(x), \Phi(x_i) \rangle$. Moreover, for many projections there are *kernel* functions that compute the result of the inner product. A kernel function k for a projection Φ satisfies $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$. Given a legal kernel function, we can use it without knowing or explicitly computing the actual mapping Φ.

To summarize, if we want to learn expressive decision surfaces, we can choose a kernel function, and use it instead of inner-product in the execution of the SVM optimization. This is equivalent to learning a linear hyperplane in the projected space.

In this work we consider two kernel functions:

- The linear kernel $k_1(x, y) = \langle x, y \rangle$.
- The quadratic kernel $k_2(x, y) = (\langle x, y \rangle + 1)^2$.

The rational for using these simple kernels, is that since our input space is high dimensional, we can hope to find a simple separation rule in that space. We therefore test the linear separator, and the next order separator as a comparison to check if higher order kernels can yield better results.

Note that the quadratic kernel is strictly more expressive than the linear one: any decision surface that can be represented with $k_1(\cdot,\cdot)$ can also be represented with $k_2(\cdot,\cdot)$. Nonetheless, it is not obvious that the more expressive representation will always perform better. Given a larger set of decision surfaces to choose from, this procedure is more susceptible to *overfitting*, i.e. learning a decision surface that performs well on the training data but performs badly on test data.

2.3.2 Boosting

Boosting was initially developed as a method for constructing good classifiers by repeated calls to "weak" learning procedure [13, 23]. The assumption is that we have access to a "weak learner" that given a training set D, constructs a function $f_D(x)$ The learner is weak in the sense that the *training set* error better than that of random guess. Formally, we assume that $f_D(x)$ classifies at least $1/2 + 1/poly(m)$ of the input space correctly.

In this paper, we use a fairly simple weak learner, that finds simple rules of the form:

$$g(x : j, t, d) = \begin{cases} d & x[j] > t \\ -d & x[j] < t \end{cases}$$

where x is an expression profile (e.g., a tissue to be classified), j is an index of a gene, $x[j]$ is the expression value of the j'th gene in the vector x, t is a threshold corresponding to gene j, and $d \in \{+1, -1\}$ is a direction parameter. Such a rule is called a *decision stump*. Given a dataset D, we learn decision stumps by exhaustively searching all genes, and for each gene searching over all thresholds and directions, and finally return the combination that has the smallest number of errors.[2]

Boosting uses the weak learning procedure (in our case, the decision stump learner) to construct a sequence of classifiers $f_1, \ldots, f_k$, and then uses a weighted vote among these classifiers. Thus, the prediction made by the boosting algorithm has the form :

$$h(x) = \operatorname{sign}(\sum_j w_j f_j(x)),$$

where w_i are the weights assigned to the classifiers.

The crux of the algorithm is the construction of the sequence of classifiers. The intuition is simple. Suppose that we train the weak learner on the original training data D to get a classifier $f_1(x)$. Then, we can find the examples in D that are classified incorrectly by f_1. We want to force the learning algorithm to give these examples special attention. This is done by constructing a new training data set in which these examples are given more weight. Boosting then invokes the weak learner on the reweighted training set

[2]Note that for each gene, we need to consider only m rules, since the gene takes at most m different values in the training data. Thus, we can limit our attentions to mid-way points between consecutive values attained by the j'th gene in the training data.

Input:

- A data set of m labeled examples $\{(x_1, l_1), \ldots, (x_m, l_m)\}$
- A weak learning algorithm L.

Initialize the distribution over the data set:

$$D_1(x_i) = 1/m$$

For $t = 1, 2, ..., T$

- Call L with distribution D_t; Get back a hypothesis h_t.
- Calculate the error of h_t:

$$\epsilon_t = \sum_{i=1}^{m} D(x_i) 1\{l_i \neq h(x_i)\}$$

- Set $w_t = \frac{1}{2} log \frac{1-\epsilon_t}{\epsilon_t}$
- Set the new distribution to be:

$$D_{t+1}(x_i) \propto D_t(x_i) e^{w_t l_i h_t(x_i)}$$

such that D_{t+1} will sum to 1.

Output: The final hypothesis

$$h(x) = sign(\sum_{t=1}^{T} w_t h_t(x))$$

Figure 1: The AdaBoost algorithm.

and obtains a new classifier. Examples are then reweighted again, and the process is iterated. Thus, boosting adaptively reweights training examples to focus on the "hard" ones.[3] In this paper, we use the AdaBoost algorithm of Freund and Schapire [13]. This algorithm is described in Figure 1.

In practice boosting is an efficient learning procedure that usually has small number of errors on test sets. The theoretical understanding of this phenomenon uses a notion of margin that is quite similar to the one defined for SVMs. Recall, that boosting classification is made by averaging the "votes" of many classifiers. Define the margin of example x_i to be

$$m_i = l_i \sum_j w_j f_j(x_i).$$

By definition, we have that if $m_i > 0$, then $h(x_i) = l_i$, and thus x_i is classified correctly. However, if m_i is close to 0, then this classification is "barely" made. On the other hand, if m_i is close to 1, then a large majority of the classifiers make the right prediction on x_i. The analysis of Schapire et al. [20, 24] shows that the generalization error of boosting (and other voting schemes) depends on the distribution of margins of training examples. Schapire et al. also show that repeated iterations of AdaBoost continually increase the smallest margin of training examples. This is contrasted with other voting schemes that are not necessarily increasing the margin for the training set examples.

[3] More precisely, boosting distorts the distribution of the input samples. For some weak learners, like the stump classifier, this can be simulated by simply reweighting the samples.

3 Evaluation

In the previous section we discussed several approaches for classification. In this section we examine their performance, on experimental data.

3.1 Data Sets

Descriptions of the two datasets studied follow. Both of these data sets involve comparing tumor and normal samples of the same tissue.

Colon cancer data set. This data set is a collection of expression measurements from colon biopsy samples reported by Alon et al. [1]. The data set consists of 62 samples of colon epithelial cells. These samples were collected from colon-cancer patients. The "tumor" biopsies were collected from tumors, and the "normal" biopsies were collected from healthy parts of the colons of the same patients. The final assignments of the status of biopsy samples were made by pathological examination.

Gene expression levels in these 62 samples were measured using high density oligonucleotide microarrays. Of the $\approx$ 6000 genes detected in these microarray, 2000 genes were selected based on the confidence in the measured expression levels. The data, 62 samples over 2000 genes is available at `http://www.molbio.princeton.edu/colondata`.

Ovarian cancer data set. This data set is a collection of expression measurements from 32 samples: 15 ovary biopsies of ovarian carcinomas, 13 biopsies of normal ovaries, and 4 samples of other tissues. Thus, the data set consists of 28 samples labeled as tumor or normal. Gene expression levels in these 32 samples were measured using a membrane-based array with radioactive probes. The array consisted of cDNAs representing approximately 100,000 clones from ovarian clone libraries. For some of the samples, there are two or three repeated hybridizations for error assessments. In these cases, we collapsed the repeated experiments into one experiment, represented by the average.

3.2 Estimating Prediction Errors

When evaluating the prediction accuracy of the classification methods we described above, it is important not to use the *training error*. Most classification methods will perform well on examples they have seen during training. To get a realistic estimate of performance of the classifier, we must test it on examples that did not appear in the training set. Unfortunately, since we have a small number of examples, we cannot remove a portion of the examples from the training set, and use them for testing.

A common method to test accuracy in such situations is *cross-validation*. To apply this method, we partition the data into k sets of samples, $C_1, \ldots, C_k$ (typically, these will be of roughly the same size). Then, we construct a dataset $D_i = D - C_i$, and test the accuracy of $f_{D_i}()$ on the samples in C_i. Having done this for all $1 \leq i \leq k$ we estimate the accuracy of the method by averaging the accuracy in each one of the cross-validation trials.

Cross-validation has several important properties . First, the training set and the test set in each trial are disjoint. Second, the classifier is tested on each sample exactly once. Finally, the training set for each trial is $(k-1)/k$ of the original data set. Thus, for large k, we get a relatively unbiased estimate of the classifier behavior given a training set of size m.

There are several possible choices of k. A common approach is to set $k = m$. In this case, every trial removes a

Method		Precent	
	correct	incorrect	unclassified
Colon			
Clustering	88.7	11.3	0.0
Nearest Neighbor	80.6	19.4	0.0
SVM, linear kernel	77.4	12.9	9.7
SVM, quad. kernel	74.2	14.5	11.3
Boosting, 100 iter.	72.6	17.7	9.7
Boosting, 1000 iter.	72.6	17.7	9.7
Boosting, 10,000 iter.	71.0	19.4	9.7
Ovarian			
Clustering	42.9	17.9	39.3
Nearest Neighbor	71.4	28.6	0.0
SVM, linear kernel	67.9	3.6	28.6
SVM, quad. kernel	64.3	3.6	32.1
Boosting, 100 iter.	89.3	10.7	0.0
Boosting, 1000 iter.	85.7	10.7	3.6
Boosting, 10,000 iter.	85.7	14.3	0.0

Table 1: Summary of classification performance of the different methods on the two training sets. The tables shows the precent of samples that were correctly classified, incorrectly classfied, and unclassfied by each method in the LOOCV evaluation. Unsupervised labels for margin based classifier were decided by a fixed threshold on classification margin: in SVM, 0.25, and in Adaboost, 0.05.

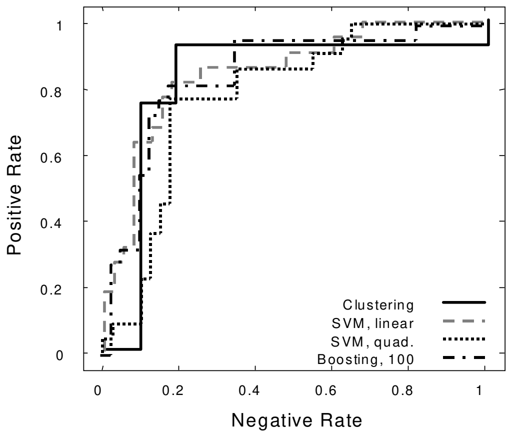

Figure 2: ROC curves for methods applied to colon cancer data set. The x-axis shows percentage of negative examples classified as positives, and y-axis shows percentage of positive examples classified as positive. Each point along the curve corresponds to the percentages achieved by a particular confidence threshold value by the corresponding classification method. Error estimates are based on LOOCV trials.

single sample and trains on the rest. This method is known as *leave one out cross validation* (LOOCV). Other common choices are $k = 10$ or $k = 5$. LOOCV has been in use since early days of pattern recognition (e.g., [10]). In some situations, using larger partitions reduces the variance of the estimators (see [18]). In this work, since the number of samples is small, we use LOOCV.

Table 1 lists the accuracy estimates for the different methods applied to the two datasets. As we can see, the clustering approach performs significantly better than the other approaches on the colon cancer data set, but not so on the ovarian data set. We can also see that quadratic SVM does not perform as well as the linear SVM, probably because it overfits the training data. The same phenomenon occurs in Adaboost, where the classifiers are more accurate after 100 iterations than after 10000 iterations.

3.3 ROC Curves

Estimates of classification accuracy give only a partial insight on the performance of a method. In our evaluation, we treated all errors as having equal penalty. In many applications, however, errors have asymmetric weights. For a general discussion of risk and loss considerations in classification see, e.g, [22]. To set terminology for our particular case, we distinguish *false positive* errors - normal tissues classified as tumor, and *false negative* errors - tumor tissues are classified as normal. In diagnostic applications, false negative errors can be detrimental, while false positives may be tolerated (since additional tests will be performed on the patient).

To deal with asymmetric weights for errors, we introduce the *confidence parameter*, β. In clustering approaches, the modified procedure labels a query sample as tumor if the cluster containing it has at least a fraction β of tumors. In a similar manner, we can introduce confidence parameters for SVM and boosting approaches by changing the threshold margin needed for positive classification.

ROC curves are used to evaluate the "power" of a classification method for different asymmetric weights (see, for example, [27]). A ROC curve plots the tradeoff between the two types of errors as the confidence parameter varies. Each point on the two dimensional curve corresponds to a particular value of the confidence parameter. The (x, y) coordinates of a point represent the fractions of negative and positive samples that are classified as positive with this particular confidence parameter. The extreme ends of the curves are the most strict and most permissive confidence values: with the strictest confidence value nothing is classified as positive, putting $(0, 0)$ on the curve; with the most permissive confidence value everything is classified as positive, putting $(1, 1)$ on the curve. The path between these two extremes shows how flexible the procedure is with respect to trading-off error rates. The best case scenario is that the path goes through the point $(0, 1)$. This implies that for some confidence parameter, all positives are classified as positive, and all negatives are classified as negative. That is - the procedure can be made very strict with respect to false positive error, with no false negative price to pay. ROC curves with large areas underneath mean that high false positive stringencies can be obtained without much of a false negative price.

In Figure 2 we plot the ROC curves for clustering, SVM and boosting on the colon cancer data set. As we can see, there is no clear domination among the methods. (The only exception is SVM with quadratic kernel that is consistently worse than the other methods.) The clustering procedure is dominant in the region where misclassification errors of both types are roughly of the same importance. However, SVM with linear kernel and boosting are preferable in regions of highly asymmetric error cost (both ends of the spectrum). This may be due to the fact that the matching coefficient score (see Section 2.2), which determines the cluster granularity, treats both types of errors as having equal costs.

4 Gene Selection

It is clear that the expression levels of many of the genes that are measured in our data sets are irrelevant to the distinc-

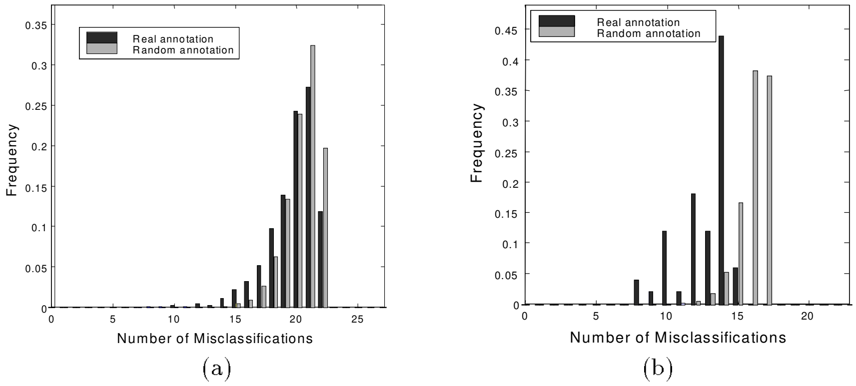

Figure 3: (a) The distribution of gene TNoM scores for the colon cancer data set compared to the distribution of scores in randomly labeled data. (b) the same, for the 50 best scoring genes.

tion between tumor and normal tissues. Taking such genes into account during classification increases the dimensionality of the classification problem, presents computational difficulties, and introduces unnecessary noise to the process. Another issue with a large number of genes is the *interpretability* of the results. If the "signal" that allows our methods to distinguish tumor from normal tissues is encoded in the expression levels of few genes, then we might be able to understand the biological significance of these genes. Moreover, a major goal for diagnostic research is to develop diagnostic procedures based on inexpensive microarrays that have enough probes to detect diseases. Thus, it is crucial to recognize whether a small number of genes can suffice for good classification.

The problem of *feature selection* received a thorough treatment in pattern recognition and machine learning. The gene expression data sets are problematic in that they contain a large number of genes (features) and thus methods that search over subsets of features can be prohibitively expensive. Moreover, these data sets contain only a small number of samples, so the detection of irrelevant genes can suffer from statistical instabilities.

To address these issues, we utilize measures of "relevance" of each gene. In particular, we focus on a quantity we call the *threshold number of misclassification* or *TNoM score* of a gene. The intuition is that an informative gene has quite different values in the two classes (normal and tumor), and thus we should be able to separate these by a threshold value. Formally, we seek the best decision stump for that gene (as defined in Section 2.3.2), and then count the classification errors this decision stump makes on the training examples. Also see Appendix B for a formal definition.

An immediate question to ask is whether genes with low TNoM scores are indeed indicative of the classification of expression. In other words, we want to test the statistical significance of the scores of the best scoring genes in our data set. We can measure significance by analyzing the distribution of scores for random labeling of samples, independent of gene expression data. To estimate this distribution we we use simulations. As we can see from Figure 3, the better TNoM scores observed in the real data are extremely unlikely in random data.

To evaluate the biological meaning of the high scoring genes we have ordered the genes in both data sets, according to their TNoM scores, and examined the genes at the top of the list (those with better TNoM scores). Among the top 100 genes in the colon cancer data set there are a number of genes that are interesting from the perspective of a potential involvement in tumorigenesis including, for example, genes involved in cell cycle regulation and angiogenesis. There were also genes, for example (D63874) HMG-1 (human) and (T55840) tumor-associated antigen L6 (human), that have previously been found to have a particular association with colorectal carcinomas [25, 30].

Among the top scoring 137 clones in the ovarian cancer data, there are 85 clones that match 8 cancer related genes (potential markers or expressed in cancer cells) and one gene that is related to increased metabolic rate (mitochondrial gene). The 8 genes are keratin 18 (breast cancer), pyruvate kinase muscle 2 (hepatoma), thymopoietin (cell proliferation), HE4 (ovarian cancer), SLPI (many different cancers, among them lung, breast, oropharyngeal, bladder, endometrial, ovarian and colorectal carcinoma), ferritin H (ovarian cancer), collagen 1A1 (ovarian cancer, osteosarcoma, cervical carcinoma), and GAPDH (cancers of lung, cervix and prostate). In addition, 2 clones with no homology to a known gene are found in this selection. Given the high number of cancer related genes in the top 137, it is likely that these novel genes exhibit a similar cancer-related behavior. We conducted expression validation for GAPDH, SLPI, HE4 and keratin 18 which confirmed the elevated expression in some ovarian carcinomas compared to normal ovarian tissues.

4.1 Classifying with Selected Subsets

When using gene selection, we need to pre-process the training data to select genes. Then, the classification procedure is applied using the training data restricted to the subset of selected genes. The gene selection stage is given a parameter k, which determines the largest error-score allowed. It then selects all genes that have a smaller or equal error score on the training data. alternatively, a p-value approach can be taken: all genes with scores which are very rare in random data are selected.

To evaluate performance with gene selection, we have to be careful to jointly evaluate both stages of the process: gene selection and classification. Thus, in each cross-validation trial, gene selection is applied based on the training examples in that trial. Note, that since the training examples are different in different cross validation trials, we expect the number of selected genes to depend on the trial.

Figure 4 describes the performance of some of the methods we discussed above when we vary the stringency of the selection process.

In the colon data set, gene selection leads to mixed results. Some methods, such as clustering, perform slightly worse with fewer genes, while others, such as SVM, perform better with smaller set of genes. On the other hand, in the ovarian data set, gene selection leads to impressive improve-

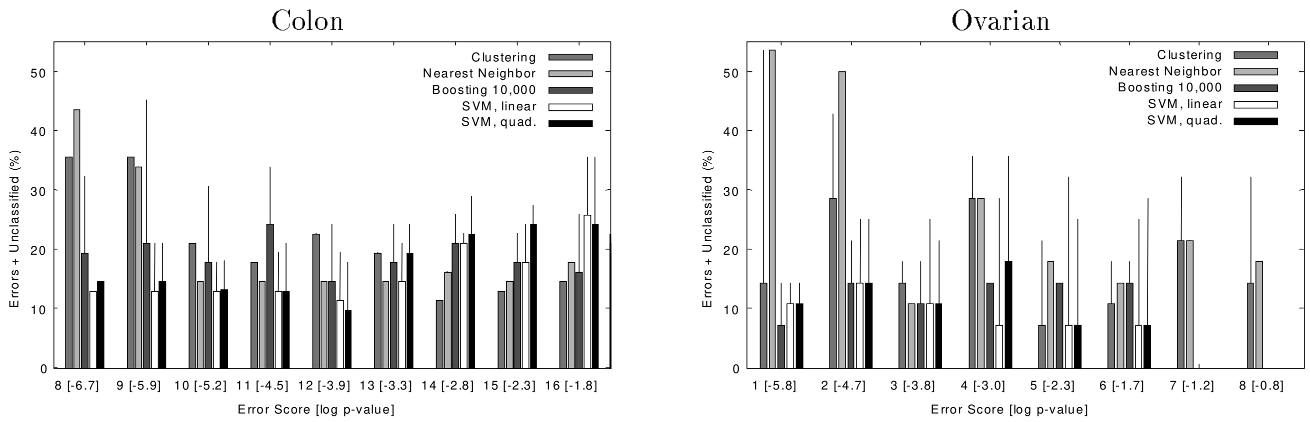

Figure 4: Classification performance, as it depends on the threshold used for selecting genes. The x-axis shows the TNoM score threshold used and the base 10 logarithm of the associated p-value as described in Appendix B . The results are based on performing LOOCV for the whole process of selection and classification, as explained in the text. For each method, the solid bar represents the fraction of the training data that was mis-classified. The thin line extensions represent the fraction which was unclassified.

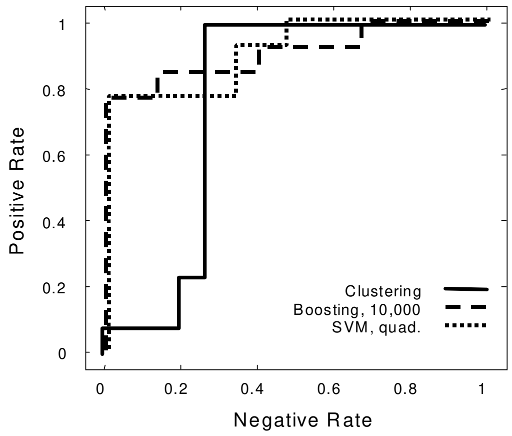

Figure 5: ROC curves for three methods that are applied to the ovarian data set with TNoM score threshold set to 3

ment in all methods. All methods perform well in the region between threshold 3 (avg. 173 clones) to 6 (avg. 4375 clones). Note that both Boosting and SVM perform well even with fewer clones.

Figure 5 shows an ROC curve for Clustering approach, Boosting, and quadratic SVM with threshold of 3 (linear SVM has similar curve to quadratic SVM, and thus was not plotted). As we can see, although all methods have roughly the same accuracy with this subset of genes, their ROC profile is strikingly different. These curves clearly show that the Clustering approach makes false positive errors, while all the other approaches make false negative errors.

5 Sample Contamination

Cancer classification based on array-based gene expression profiling may be complicated by the fact that clinical samples, e.g. tumor vs. normal, will likely contain a mixture of different cell types. In addition, the genomic instability inherent in tumor samples may lead to a large degree of random fluctuations in gene expression patterns. Although both the biological and genetic variability in tumor samples have the potential to lead to confusing and difficult to interpret expression profiles, gene expression profiling does allow us to efficiently distinguish tumor and normal samples, as we have seen in the previous sections. However, the presence of different cell types within and between samples could lead to identification of genes that strongly affect cluster formation but which may have little to do with the process being studied, in this case tumorigenesis. For example, in the case of the colon cancer data set presented above, a large number of muscle-specific genes were identified as being characteristic of normal colon samples both in our clustering results and in the results of Alon et al. [1]. This is most likely due to a higher degree of smooth muscle contamination in the normal versus tumor samples.

This raises the concern that our classification may be biased by the presence of muscle specific genes. To test this hypothesis, we attempted to construct data sets that avoid genes that are suspected in introducing bias. We listed the top 200 error-score ranking genes in the colon cancer data set, and identified muscle-specific genes. These include (J02854) myosin regulatory light chain 2, smooth muscle isoform (human); (T60155) actin, aortic smooth muscle (human); and (X12369) tropomyosin alpha chain, smooth muscle (human)that are designated as smooth muscle-specific by Alon et al.'s analysis, and (M63391) desmin (human), complete cds; (D31885) muscle-specific EST (human); and (X7429) alpha 7B integrin (human) which are suspected to be expressed in smooth muscle based on literature searches.

An additional form of "contamination" is due to the high metabolic rate of the tumors. This results in high expression values for ribosomal genes. Although such high expression levels can be indicative of tumors, such a finding does not necessarily provide novel biological insight into the process, nor provide a diagnostic tool since ribosomal activity is present in virtually all tissues. Thus, we also identified ribosomal genes in the top 200 scoring genes.

Figure 6 shows the performance of the clustering approach on three data sets: the full 2000 gene data set, a data set without muscle specific genes, and a data set with-

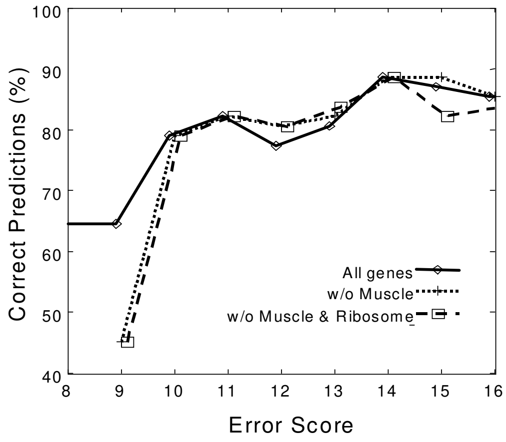

Figure 6: Curves showing the predictive performance of clustering methods in the original Alon et al. data set, and data sets where muscle specific, and ribosomal genes were removed. All estimates are based on LOOCV evaluation. These results show that even without the obvious contaminations, our methods are successful in reliably predicting tissue type.

out both muscle specific and ribosomal genes. As the learning curves show, the removal of genes affects the results only in cases using the smallest sets of genes. From error score threshold of 10 (avg. 9.1 genes) and higher, there is no significant change in performance for the procedure. Thus, although muscle specific genes can be highly indicative, the classification procedure performs well even without relying on these genes.

Although the muscle contamination did not necessarily alter the ability of this gene set to be used to classify tumor vs. normal samples in this case, it will continue to be important to account for possible affects of tissue contamination on clustering and classification results. Experimental designs that include gene expression profiles of tissue and/or cell culture samples representative of types of tissue contaminants known to be isolated along with different types of tumor samples (for example see Perou et al. [21]), can be utilized to help distinguish contaminant gene expression profiles from those actually associated with specific types of tumor cells.

6 Conclusions

In this paper we examined the question of tissue classification based on expression data. Our contribution is four-fold. First, we introduced a new cluster-based approach for classification. This approach builds on clustering algorithms that are suitable for gene expression data. Second, we performed rigorous evaluation of this method, and of known methods from the machine learning literature. These include large margin classification methods (SVM and AdaBoost) and the nearest-neighbor method. Third, we highlighted the issue of sample contamination and estimated the sensitivity of our approach to sample variability. Differences in tissue biopsies could theoretically affect the quality of any given classification method. Studying this issue, we observed no significant contaminating tissue bias in the colon cancer data set. Finally, we investigated the issue of gene selection in expression data. As our results for the ovarian data set show, a large number of clones can have a negative impact on predictive performance. We showed that a fairly simple selection procedure can lead to significant improvements in prediction accuracy.

The work reported here is closely related to two recent papers. First, Lander et al. [14] examined gene expression profile differences in AML and ALL (two types of leukemia) biopsies. They employ a scoring rule to select informative genes and perform LOOCV experiments to test a voting based classification approach. Although their score for gene selection and their classification method are different than ours, their main conclusions are quite similar in that they get good classification accuracy with relatively small number of genes. Second, Brown et al. [4] use support vector machines in the context of gene expression data. In contrast to our approach, they attempt to classify the genes rather then samples. Thus, they deal with the dual classification problem. The characteristics of their classification problem are quite different: many examples (i.e., thousands of genes), and few attributes (i.e., expression in different samples). We note that some of the approaches we used in this work (e.g., clustering based classification) might be applicable to this dual classification problem as well.

As noted above, the gene selection process we explored in this paper is quite simplistic. In particular, it was based on scoring single genes for relevance. Thus, the process might select several genes that convey the same information, and might ignore genes that add independent information. We are currently studying more direct approaches to the selection of informative *sets* of genes. Identifying sets of genes that give rise to efficient learned classifiers might reveal previously unknown disease related genes and guide further biological research.

Acknowledgments

The authors are grateful to Yoram Singer for help with SVM. Amir Ben-Dor was supported by the *Program for Mathematics and Molecular Biology (PMMB)*. Nir Friedman and Iftach Nachman were supported through the generosity of the Michael Sacher Trust.

References

[1] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Nat. Acad. Sci. USA*, 96:6745–6750, 1999.

[2] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6:281–297, 1999.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, U.K., 1995.

[4] M.P.S. Brown, W.N. Grundy, D. Lin, N. Cristianini, C. Sugnet, T.S. Furey, M. Ares Jr., and D. Haussler. Knowledge-based analysis of microarray gene expression data using support vector machines. Technical Report UCSC-CRL-99-09, U. C. Santa Cruz, 1999.

[5] C. J. C. Burges. A tutorial on Support Vector Machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[6] S. Chu, J. DeRisi, M. Eisen, J. Mullholland, D. Botstein, P. Brown, and I. Herskowitz. The transcriptional program of sporulation in budding yeast. *Science*, 282:699–705, 1998.

[7] P. A. Clarke, M. George, D. Cunningham, I. Swift, and P. Workman. Ananlysis of tumor gene expression following chemotherapeutic treatment of patients with bowel cancer. In *Proc. Nature Genetics Microarray Meeting 99*, page 39, Scottsdale, Arizona, 1999.

[8] C. Cortes and V. Vapnik. Support vector machines. *Machine Learning*, 20:273–297, 1995.

[9] J. DeRisi., V. Iyer, and P. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 282:699–705, 1997.

[10] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, 1973.

[11] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Nat. Acad. Sci. USA*, 95:14863–14868, 1998.

[12] B. Everitt. *Cluster Analysis*. Edward Arnold, London, third edition, 1993.

[13] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Computer and System Sciences*, 55:119–139, 1997.

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

[15] V.R. Iyer, M.B. Eisen, D.T. Ross, G. Schuler, T. Moore, J.C.F. Lee, J.M. Trent, L.M. Staudt, J. Hudson, M.S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P.O. Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283:83–87, 1999.

[16] Kim lab home page. `http://cmgm.stanford.edu/~kimlab/`.

[17] J. Khan, R. Simon, M. Bittner, Y. Chen, S. B. Leighton, T. Pohida, P. D. Smith, Y. Jiang, G. C. Gooden, J. M. Trent, and P. S. Meltzer. Gene expression profiling of *Alveolar rhabdomyosarcoma* with cDNA microarrays. *Cancer Reasearch*, 1998.

[18] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc. Fourteenth International Joint Conference on Artificial Intelligence (IJCAI '95)*, pages 1137–1143. Morgan Kaufmann, San Francisco, Calif., 1995.

[19] D. J. Lockhart, H. Dong, M. C. Byrne, M. T. Follettie, M. V. Gallo, M. S. Chee, M. Mittmann, C. Want, M. Kobayashi, H. Horton, and E. L. Brown. DNA expression monitoring by hybridization of high density oligonucleotide arrays. *Nature Biotechnology*, 14:1675–1680, 1996.

[20] L. Mason, P. Bartlett, and J. Baxter. Direct optimization of margins improves generalization in combined classifiers. In *Advances in Neural Information Processing Systems 11*. MIT Press, Cambridge, Mass., 1999.

[21] C. M. Perou, S. S. Jeffrey, M. v de Rijn, C. A. Rees, M. B. Eisen, D. T. Ross, A. Pergamenschikov, C. F. Williams, S. X. Zhu, J. C. F. Lee, D. Lashkari, D. Shalon, P. O. Brown, and Botstein D. Distinctive gene expression patterns in human mammary epithelial cells and breast cancers. *Proc. Nat. Acad. Sci. USA*, 96:9212–9217, 1999.

[22] B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, 1996.

[23] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

[24] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26:1651–1686, 1998.

[25] T.H. Schiedeck, S. Christoph, M. Duchrow, and H.P. Bruch. Detection of hl6-mrna: new possibilities in serologic tumor diagnosis of colorectal carcinomas. *Zentralbl Chir*, 123(2):159–162, 1998.

[26] M. Schummer, W. NG, R. Bumgarner, P. Nelson, B. Schummer, L. Hassell, L. R. Baldwin., B. Karlan, and L. Hood. Comperative hybridization of an array of 21,500 overian cDNAs for the discovery of genes overexpressed in overian carcinomas. *Gene*, 238:375–385, 1999.

[27] J. Swets. Measuring the accuracy of diagnostic systems. *Science*, 240:1285–1293, 1988.

[28] V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, New York, 1999.

[29] X. Wen, S. Furhmann, G. S. Micheals, D. B. Carr, S. Smith, J. L. Barker, and R. Somogyi. Large-scale temporal gene expression mapping of central nervous system development. *Proc. Nat. Acad. Sci. USA*, 95:334–339, 1998.

[30] Y.Y. Xiang, DY Wang, M. Tanaka, M. Suzuki, E. Kiyokawa, H. Igarashi, Y. Naito, Q. Shen, and H. Sugimura. Expression of high-mobility group-1 mrna in human gastrointestinal adenocarcinoma and corresponding non-cancerous mucosa. *Int J. Cancer*, 74(1):1–6, Feb 1997.

A Support Vector Machines

A linear decision rule can be represented by a hyperplane in R^N such that all examples on the one side of the hyperplane are labeled positive and all the examples on the other side are labeled as negative. Such a rule can be represented by a vector $w \in R^N$ and a scalar b that together specify the hyperplane $w \cdot x + b = 0$. Classification for a new example x is performed by computing $\mathrm{sign}(w \cdot x + b)$. Recall that $|\frac{x \cdot w + b}{||w||}|$ is the distance from x to the line $x \cdot w + b = 0$. Thus, if all points in the training data satisfy

$$l_i(x_i \cdot w + b) \geq 1 \tag{1}$$

than all points correctly classified, and all of them have a distance of at least $1/||w||$ from the hyperplane. We can find the hyperplane that maximizes the margin of error by solving the following quadratic program:

Minimize $||w||^2$
Subject to $l_i(x_i \cdot w + b) \geq 1$ for $i = 1, \ldots, m$.

Such quadratic programs can be solved in the dual form. This dual form is posed in terms of auxiliary variables α_i. The solution has the property that

$$w = \sum_i \alpha_i l_i x_i,$$

and thus, we can classify a new example x by evaluating

$$\mathrm{sign}(\sum_i \alpha_i l_i \langle x_i, x \rangle + b) \tag{2}$$

In practice, there is a range of optimization methods that can be used for solving the dual optimization problem. See [5] for more details.

The SVM dual optimization problem and its solution have several attractive properties. First, only a subset of the training examples determine the position of the hyperplane. Intuitively, these are exactly these samples that are at a distance $1/||w||$ from the hyperplane. It turns out that the dual problem solution assigns $\alpha_i = 0$ to all examples that are not "supporting" the hyperplane. Thus, we only need to store the *support vectors* x_i for which $\alpha_i > 0$. (Hence the name of the technique.)

Second, the dual form of the quadratic optimization problem involves only inner products of vectors in R^N. In other words, vectors x_i do not appear outside the scope of an inner product operation. Similarly, the classification rule (2) only examines vectors in R^N inside the inner product operation. Thus, if we want to consider any projection $\Phi : \mathbf{R}^N \mapsto \mathbf{R}^M$, then we can find an optimal separating hyperplane in the projected space, by solving the quadratic problem with inner products $\langle \Phi(x_i), \Phi(x_j) \rangle$.

In many cases, we can perform the optimization in high-dimensional spaces, by efficient computation of the inner product in these spaces. A function $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$ is called a *kernel function*. For many projections, the kernel function can be computed in time that is linear in N, regardless of the dimension M.

B TNoM Score

The *TNoM score* is the relevance measure, for single genes, that was employed in this work. We begin by the formal definition. Given a vector $v \in \{+, -\}^m$ and $1 \leq i \leq m$, let

$$\oplus(i) = \text{the number of +'s in } v_1, ..., v_i,$$

and

$$\ominus(i) = \text{the number of −'s in } v_1, ..., v_i.$$

Let $m_+ = \oplus(m)$ and $m_- = \ominus(m)$ be the number $+$ and $-$ in v. Then let

$$M(i) = \min\left(\ominus(i) + m_+ - \oplus(i)\ ,\ \oplus(i) + m_- - \ominus(i)\right), \tag{3}$$

and

$$TNoM(v) = \min_{1 \leq i \leq m} M(i). \tag{4}$$

The index i that attains this minimum is, in fact, the best decision stump for this vector, as described in Section 2.3.2.

To assess the relevance of gene expression patterns in actual data, to the classification problem studied we want to compare the TNoM score associated with each such pattern to TNoM scores expected in random data. Let $U \in \{+, -\}^m$ be a random vector drawn uniformly over the set

$$\left\{ u \in \{+,-\}^m : \begin{array}{c} \text{the number of +'s in } u = m_+, \\ \text{and} \\ \text{the number of −'s in } u = m_-. \end{array} \right\}$$

To perform the stated comparison task we would like to estimate

$$Prob(TNoM(U) \leq s),$$

for all $0 \leq s \leq \min(m_+, m_-)$. We currently employ simulations for doing this. We have also developed a rigorous recursive procedure that exactly computes these probabilities. This will be discussed in future work.