BIOINFORMATICS

BagBoosting for tumor classification with gene expression data

Marcel Dettling

Seminar für Statistik, ETH Zürich, CH-8092 Switzerland

ABSTRACT

Motivation: Microarray experiments are expected to contribute significantly to the progress in cancer treatment by enabling a precise and early diagnosis. They create a need for class prediction tools, which can deal with a large number of highly correlated input variables, perform feature selection and provide class probability estimates that serve as a quantification of the predictive uncertainty. A very promising solution is to combine the two ensemble schemes bagging and boosting to a novel algorithm called BagBoosting.

Results: When bagging is used as a module in boosting, the resulting classifier consistently improves the predictive performance and the probability estimates of both bagging and boosting on real and simulated gene expression data. This quasi-guaranteed improvement can be obtained by simply making a bigger computing effort. The advantageous predictive potential is also confirmed by comparing BagBoosting to several established class prediction tools for microarray data.

Availability: Software for the modified boosting algorithms, for benchmark studies and for the simulation of microarray data are available as an *R* package under GNU public license at http://stat.ethz.ch/~dettling/bagboost.html

Contact: dettling@stat.math.ethz.ch

1 INTRODUCTION

A precise diagnosis of cancerous malignancies is difficult but often crucial for successful treatment. Given the large-scale, high-throughput gene expression technology and accurate statistical methods, biomolecular information could become as, or even more, important for cancer diagnosis than the traditional clinical factors. The challenge is that microarray experiments generate large datasets with expression values for thousands of genes, but usually not more than a few dozens of arrays. The situation with so many more predictor variables than experiments raises new statistical challenges and has leads us to a wealth of research. The task of diagnosing cancer on the basis of microarray data has been termed class prediction in the literature, and encompasses methods ranging from modified versions of traditional discriminant analysis, over penalized regression

approaches and classical non-parametric methods such as the nearest neighbor rule (NNR) to modern tools of machine learning (for an overview and references see Dudoit and Fridlyand, 2003).

Boosting is a flexible class prediction tool from machine learning with remarkable success in a wide variety of applications, especially in those with high-dimensional predictors. It aims at producing an accurate committee classifier from a sequential ensemble of base learners, which are fitted with adaptively reweighted versions of the data. It is attractive to use boosting for class prediction with microarray data due to its natural ability to perform multivariate feature selection, and because it provides class probability estimates that serve as a natural quantification of the predictive uncertainty. This has first been tried by Ben-Dor *et al.* (2000) and Dudoit *et al.* (2002) with moderate success: empirically, boosting could at best keep up with much simpler methods such as the NNR. Later, we suggested a boosting algorithm that was tailored for microarray data, showing a more satisfactory predictive performance (see Dettling and Bühlmann, 2003).

Here, we present a novel type of boosting algorithm and show its promising potential to improve class prediction with microarray data. Our algorithm is called BagBoosting, because it uses bagging as a module in our tailored boosting algorithm for microarray data. The idea is illustrated in Figure 1: in each boosting iteration, we do not just rely on a single base learner, but instead aggregate the output from several base predictors generated from bootstrap samples, each drawn with replacement from the reweighted training data. The rationale for combining the two ensemble schemes bagging and boosting is as follows. Although the boosting committee has clearly lower bias but slightly increased variance than the base learner, bagging of (unstable) base learners leads to an ensemble with lower variance but approximately non-altered bias. Hence BagBoosting might combine the advantages of both methods and results in a prediction tool achieving both lower bias and variance, i.e. lower mean squared error. Even though we cannot present a strict mathematical justification that applies for the microarray setting, simulation studies
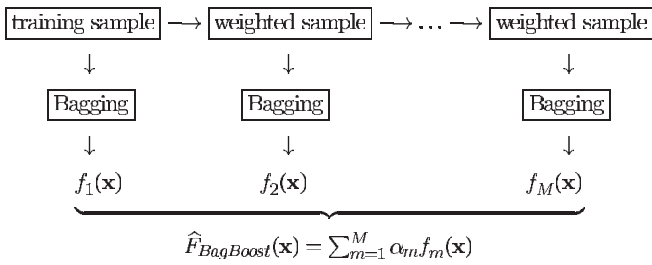
Fig. 1. The fundamental idea of BagBoosting.

clearly reflect these heuristically derived advantages. As a consequence, we also expect BagBoosting to yield superior class prediction results on real microarray data. An elaborate empirical study confirms the improvement compared to both bagging and boosting. Also with respect to established classifiers including discriminant analysis, nearest neighbor methods and modern tools such as support vector machines and random forests, BagBoosting is competitive. Finally, we show how the BagBoosting fit can be rewritten in terms of an additive model. This serves to study the influence of single genes on real world classification problems, and to analyze the ability to recover the true model structure in simulated data.

2 METHODS

2.1 Class prediction with gene expression data

The main goal in class prediction with gene expression data is a precise and early diagnosis of cancerous malignancies that allows to tailor the patients' treatment for maximal efficacy and minimal toxicity. Given microarray experiments and information about outcome of the disease from n former patients, the task of class prediction amounts to learning the relationship between the transcript levels and the outcome. Then, presented with the gene expression profiles of new, independent patients, we can establish a diagnosis of the development of their disease and outcome. In mathematical notion, we assume that we are given a learning sample $\mathcal{L}$ of n training data pairs

$$\mathcal{L} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\},$$

which are independent and identically distributed (iid) realizations of a random vector $(\mathbf{x}, y)$. The feature vector $\mathbf{x} \in \mathbb{R}^p$ is the gene expression profile, which can be either from cDNA microarrays or from Affymetrix oligonucleotide chips, but we assume that it is accurately preprocessed and normalized, such that it can be taken at face-value. In the simplest form, the response variable $y \in \{0, 1\}$ codes for a dichotomous response[1], e.g. tumor subtype or risk category. For class prediction, we choose the approach of estimating

conditional class probabilities

$$\hat{p}(\mathbf{x}) = \hat{P}_{\mathcal{L}}[y = 1 \mid \mathbf{x}]$$

from the learning sample $\mathcal{L}$, based on the gene expression profile $\mathbf{x}$. They are a natural quantification for the uncertainty of class predictions and can in turn be used to predict class labels. In the case of equal misclassification costs, a new patient with gene expression profile $\mathbf{x}_\nu$ is assigned to one of the response classes via

$$\hat{y}(\mathbf{x}_\nu) = \begin{cases} 0, & \text{if } \hat{p}(\mathbf{x}_\nu) < \frac{1}{2} \\ 1, & \text{if } \hat{p}(\mathbf{x}_\nu) \geq \frac{1}{2}. \end{cases} \tag{1}$$

Estimating conditional probabilities and subsequent class prediction is a thoroughly discussed problem in statistics, but microarray data with thousands of predictor variables and just dozens of samples are a new challenge that requires adaption of known and development of novel methodology. A promising tool performing multivariate variable selection, providing probability estimates and having good predictive potential in such high-dimensional problems are modified boosting algorithms.

2.2 BagBoosting

Boosting, first proposed by Freund and Schapire (1996), is a powerful ensemble method that consistently estimates conditional class probabilities $\hat{p}(\mathbf{x})$ from a sequence of base classifiers, which are fitted with iteratively reweighted versions of the training data. The initial notion of boosting was that in each iteration, the cases that were misclassified in the previous round get their weights increased, whereas the weights are decreased for cases that were correctly classified. Rather than as a sequential data reweighting scheme, boosting can more fruitfully be seen as a forward stagewise strategy for function estimation. It works by iterative optimization of an empirical risk function

$$R[\mathcal{L}, \hat{p}(\mathbf{x}), L] = \frac{1}{n} \sum_{i=1}^{n} L[y_i, \hat{p}(\mathbf{x}_i)]$$

from the learning set $\mathcal{L}$ via constrained functional gradient descent, where $\hat{p}(\mathbf{x}_i)$ denotes the current boosting probability estimate for the i-th instance and $L[\cdot, \cdot]$ a statistically motivated loss function. If we employ the binomial log-likelihood

$$L[y, \hat{p}(\mathbf{x})] = y \cdot \log[\hat{p}(\mathbf{x})] + (1 - y) \cdot \log[1 - \hat{p}(\mathbf{x})],$$

a continuous surrogate for the 0/1-misclassification loss and a very established criterion for binary classification, it is easy to show that the resulting algorithm, called LogitBoost (Friedman *et al.*, 2000), yields an approximation to half of the

---

[1]Extensions to a polytomous response are discussed later in Section 2.3.

log-odds ratio. That is,

$$\hat{F}_M(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m f_m(\mathbf{x}) \approx \frac{1}{2} \log \left[\frac{p(\mathbf{x})}{1 - p(\mathbf{x})} \right].$$

Hence, LogitBoost is a linear expansion in a set of base learners f_m on the logit scale, obtained by stagewise optimization of the binomial log-likelihood. Estimated conditional class probabilities are obtained from the simple transformation

$$\hat{p}(\mathbf{x}) = \frac{1}{1 + \exp[-2\hat{F}_M(\mathbf{x})]},$$

and can be used for class prediction as in (1). Obviously, the choice of the base learner f_m is crucial for the final boosting estimate $\hat{F}_M$. Originally, decision trees have mainly been used as a base procedure. In the context of microarray data, we observed that stumps, these are the simplest decision trees with only one split and two terminal nodes, yield the best empirical performance (Dettling and Bühlmann, 2003).

Now, we present a novel type of algorithm, which uses Breiman's (1996) bagging as a module in our tailored boosting procedure for microarray data. This amounts to a modification of boostings base learner: instead of relying on a single decision tree/stump, we aggregate multiple versions of them, obtained from bootstrap samples that are drawn with replacement from the reweighted training data of the m-th boosting iteration (Fig. 1). This intuitive idea has been assumed by different researchers, it was briefly sketched by Bühlmann and Yu (2000), but neither there are any publications containing a formal description of the algorithm nor there are any systematical analyses of its performance via empirical studies on real or simulated data.

2.2.1 Algorithmic details Our BagBoosting algorithm for class prediction with microarray data is the LogitBoost algorithm using bagged stumps as base learner. It is defined as follows:

Step 1: Initialization
Set the initial boosting estimate to $\hat{F}_0(\mathbf{x}_i) \equiv 0$ and the initial probabilities to $\hat{p}(\mathbf{x}_i) \equiv \frac{1}{2}$.

Step 2: Boosting iterations
For $m \in \{1, 2, \ldots, M = 100\}$ repeat:

(A) Compute the weights w_i and the working response z_i for $i = 1, \ldots, n$

$$w_i \leftarrow \hat{p}(\mathbf{x}_i) \cdot [1 - \hat{p}(\mathbf{x}_i)],$$
$$z_i \leftarrow \frac{y_i - \hat{p}(\mathbf{x}_i)}{w_i}.$$

(B) Bagging to obtain the base learner
For $b \in \{1, 2, \ldots, B = 50\}$ repeat:
(i) Construct a bootstrap sample $\mathcal{B} = \{(\mathbf{x}_1^*, w_1^*, z_1^*), \ldots, (\mathbf{x}_n^*, w_n^*, z_n^*)\}$ by randomly drawing n triples with replacement from the original data triples $\{(\mathbf{x}_1, w_1, z_1), \ldots, (\mathbf{x}_n, w_n, z_n)\}$.
(ii) Fit a regression stump $g^{(b)}$ by weighted least squares on the bootstrap sample $\mathcal{B}$

$$g^{(b)}(\cdot) = \arg \min_{g(\cdot)} \sum_{i=1}^{n} w_i^* [z_i^* - g(\mathbf{x}_i^*)]^2.$$

Average these B stumps to obtain boostings base learner

$$f_m(\mathbf{x}_i) \leftarrow \frac{1}{B} \sum_{b=1}^{B} g^{(b)}(\mathbf{x}_i), \quad \text{for all } i = 1, \ldots, n.$$

(C) Updating the boosting estimate and the probabilities

$$\hat{F}_m(\mathbf{x}_i) \leftarrow \hat{F}_{m-1}(\mathbf{x}_i) + \frac{1}{2} f_m(\mathbf{x}_i),$$
$$\hat{p}(\mathbf{x}_i) \leftarrow \frac{1}{1 + \exp[-2\hat{F}_m(\mathbf{x}_i)]}.$$

The definition of weights w_i and working response z_i in the Logit(Bag)Boost algorithm is such that the (bagged) base learner is forced to focus on observations close to the decision boundary, i.e. data points where the boosting classifier is in doubt about class membership. The final number of boosting iterations M regulates the complexity of the prediction model, early stopping is a form of shrinkage. In the context of microarray data, we recommend a default value of $M = 100$, which is a reasonable compromise between computing time, predictive accuracy and prevention of overfitting. This choice was shown to be empirically superior to approaches where M was estimated on the training data via cross-validation (Dettling and Bühlmann, 2003). In contrast, the number of bagging iterations B is not a tuning parameter: in theory, the bagged estimator corresponds to the bootstrap expectation and would require infinitely many iterations. In practice, we employ Monte Carlo methods for an approximation; our choice of $B = 50$ has been recognized as sufficiently large by Breiman (1996).

2.2.2 Some heuristics about BagBoosting If the underlying base algorithm is a decision stump, this is a univariate indicator function with one split point in a single variable x_j and constant values in the two terminal nodes, BagBoosting yields a model that is additive in the predictor variables. This, since in every boosting iteration an average of B univariate functions is linearly added to the current fit. We can always rearrange the summation and represent the BagBoosting

estimate as an additive combination

$$\hat{F}_M(\mathbf{x}) = \sum_{j=1}^{p} \theta_j h(x_j),　\qquad (2)$$

where $h(\cdot)$ are aggregated indicator functions typically showing a much smoother behavior than stumps, see Section 3.2.3 and Figure 4. The coefficient θ_j is determined by when, how often and with what accuracy the j-th variable was used during the bagging and boosting iterations; it reflects the importance of gene j in the final BagBoosting committee. One of BagBoostings strengths is that it performs multivariate variable selection and adds complexity in a stagewise fashion: some of the genes are never selected at all and have $\theta_j \equiv 0$. The boosting iterations, i.e. the incorporation of new terms in (2) is always conditional on the current fit, such that we select genes that provide supplementary information to the previous ones, rather than picking genes that all re-explain the same phenomenon as with univariate gene selection. The relatively simple, additive model from (2) is focusing on the main effects, but free of interaction terms. This does not mean that we deny them in the 'true' model, but from an empirical viewpoint (data not shown), probably due to the usually small sample size n, it pays off to rely on this simpler auxiliary model. As microarray studies get larger, more complicated models may become appropriate. The biggest advantage of BagBoosting is that we can obtain them, without alterations on the generic algorithm itself, by just using larger decision trees as the base algorithm.

It is well known that stumps are an unstable weak learner, producing highly biased and variable estimates. Here, we give some heuristical arguments as to why BagBoosting, an additive expansion in the set of stumps, improves these poor properties. Bühlmann and Yu (2003) showed that squared error loss boosting with smoothing splines as base learner leads to an ensemble that has strongly reduced bias, but only slightly higher variance than the base learner. Although the mathematical results cannot be directly transferred, the pronounced bias reduction and the slight increase in the variability presumably hold as well when applying LogitBoost with stumps to microarray data. Thus, there is a room to improve boosting with a low variance base learner, which however still needs to have weak learning capacity only, i.e. a considerable amount of bias. Bagging, a smoothing operation that reduces the variance of unstable prediction tools by averaging out hard decisions as from indicator functions, but without having much of an effect on their bias, is predestined to be used as a base learner in boosting. The rationale is that BagBoosting profits from the synergy of baggings variance and boostings bias reduction and achieves lower mean squared error, such that we can expect a better predictive performance. These heuristics are supported by our empirical work on real and simulated data shown in Section 3.

2.2.3 Comparison to other modifications of Boosting
Here, we emphasize that BagBoosting differs from other boosting bagging hybrids that have been proposed in the literature. Friedman (2002), in his stochastic gradient boost, draws a single, random subset of the data points for each boosting iteration and fits a single decision tree learner. The subsampling even increases the variability of the base learner, but Friedman argues that it reduces the correlation among the learners from different stages, which results in a variance reduction of the final boosting committee. He demonstrates superior empirical results on a few simulated regression and (to a much lesser extent) classification problems. Another procedure similar to BagBoosting is Breiman's (2001a) 'Iterated Bagging to Debias Regressions'. As the heading suggests, it is a procedure primarily developed for the regression, but not in the classification context. It is closely related to squared error loss boosting and works by stagewise fitting of unbiasedly estimated residuals from the out-of-bag samples in a bagged base learner. Besides our common goal of simultaneously reducing bias and variance by combining stagewise modeling with bagged learners, our procedures are fundamentally different, since we are boosting small bagged decision trees in a classification problem, without making use of the out-of-bag samples. Finally, Friedman and Popescu (2003) view ensemble learning from the perspective of high-dimensional numerical quadrature. They reveal a connection between sequential ensemble classification and quasi Monte Carlo integration, and they empirically showed that hybrid approaches yield computational advantages. None of their hybrids exactly corresponds to our BagBoosting algorithm, but their methodology could serve as a route for explaining the success of BagBoosting.

2.3 Multiclass problems

Since there are usually no genes that accurately discriminate more than two classes at once, we recommend to run multiple binary comparisons in J-class microarray problems where $y \in \{0, 1, \dots, J-1\}$. The simplest solution is the one-against-all approach, which works by defining the response in the j-th problem as $y^{(j)} = 1$ if $y = j$, and $y^{(j)} = 0$ else. Then, we are running BagBoosting J-times on the modified data $\mathcal{L}^{(j)} = \{(\mathbf{x}_1, y_1^{(j)}), \dots, (\mathbf{x}_n, y_n^{(j)})\}$. The estimated conditional class probabilities are normalized and can in turn be used for maximum-likelihood classification via

$$\hat{p}^{(j)}(\mathbf{x}) = \frac{\hat{P}_{\mathcal{L}^{(j)}}[y^{(j)} = 1|\mathbf{x}]}{\sum_{k=0}^{J-1} \hat{P}_{\mathcal{L}^{(k)}}[y^{(k)} = 1|\mathbf{x}]}.$$

$$\hat{y}(\mathbf{x}) = \operatorname*{arg\,max}_{j \in \{0,\dots,J-1\}} \hat{p}^{(j)}(\mathbf{x}).$$

Dettling and Bühlmann (2003) have shown that this is empirically superior to boosting algorithms where multiclass problems are handled simultaneously. Depending on the structure of the response classes, more complex schemes than

one-against-all may be more accurate for splitting polytomous into multiple binary problems.

2.4 Feature preselection

BagBoosting incorporates multivariate feature selection and hence does not crucially depend on preliminary gene filtering by univariate methods when compared with other class prediction tools. When running our analyses for different numbers (10, 25, 50, 75, 100, 200) of genes filtered by the Wilcoxon test statistic, we observed that the error rates as well as the ranking among the classifiers hardly changed. Moreover, the predictive potential of BagBoosting was only slightly worse without gene preselection, whereas many benchmark classifiers deteriorated considerably. For a fair comparison and due to space constraints, we just display the outcome with 200 genes in Section 3. The complete information is available from the webpage http://stat.ethz.ch/~dettling/bagboost.html.

Numerous alternative methods for gene filtering do exist. These include, e.g. the popular t-test statistic, the TNoM-score of Ben-Dor *et al.* (2000) or a selection based on the Gini-index that is used as the splitting criterion for the stumps. However, we prefer the Wilcoxon statistic due to its theoretical property of being close to optimal over a wide range of data-generating distributions, and due to our empirical evidence that Boosting performed worse with features preselected by the t-test and the TNoM-score (see Dettling and Bühlmann, 2003).

2.5 Other classifiers

We compared BagBoosting to several competitors, using the implementations that are accessible from the statistical software bundle R (R Development Core Team, 2004). The classifiers include (1) Boosting: 100 iterations of LogitBoost with stumps, exactly as described by Dettling and Bühlmann (2003). (2) Random Forests (Breiman, 2001b): a technique based on an ensemble of bagged trees that use random feature selection at each node. We relied on the R function `randomForest()` (Liaw and Wiener, 2002) and tuned the number of candidate variables for each split, as well as the minimum size of terminal nodes by searching the grid $\{1, 2, \ldots, 8\} \times \{1, 2, \ldots, 10\}$ on the training data, as suggested by Meyer *et al.* (2003). (3) Support vector machines (Burges, 1998): a modern machine learning technique that fits hyperplanes with maximum margins to appropriately transformed data. We used the R implementation `svm()` (Meyer, 2001) that is based on the LIBSVM C++ library of Chang and Lin (2001, http://www.csie.ntu.edu.tw/~cjlin/libsvm), and performed C-classification with RBF kernels. The parameter γ and the cost C were tuned by a grid search on $\{2^{-10}, 2^{-9}, \ldots, 2^5\} \times \{2^{-5}, 2^{-4}, \ldots, 2^{10}\}$ by 10-fold cross-validation on each training dataset, similar to Meyer *et al.* (2003). (4) Nearest shrunken centroids (also known as PAM; Tibshirani *et al.*, 2002): this classifier is similar to

diagonal linear discriminant analysis (DLDA), but uses a soft-thresholding scheme to obtain sparse prediction models. We employed the R implementation of the original authors, which we also used for the determination of the shrinkage parameter by 10-fold cross-validation as suggested by Tibshirani *et al.* (2002). As benchmark methods, we employ (5) DLDA and (6) the 1-NNR relying on Euclidean distances.

3 RESULTS

3.1 Real data

We report the class prediction performance of BagBoosting on six publicly available datasets. They are as follows:

Dataset	Publication	n	p	J	Response
Leukemia	Golub *et al.* (1999)	72	3571	2	Subtypes of leukemia
Colon	Alon *et al.* (1999)	62	2000	2	Tumor/normal tissue
Prostate	Singh *et al.* (2002)	102	6033	2	Tumor/normal tissue
Lymphoma	Alizadeh *et al.* (2000)	62	4026	3	Subtypes of lymphoma
SRBCT	Khan *et al.* (2001)	63	2308	4	Different tumor types
Brain A	Pomeroy *et al.* (2002)	42	5597	5	Different tumor types

After preprocessing, all gene expression profiles were base 10 log-transformed and, in order to prevent single arrays from dominating the analysis, standardized to zero mean and unit variance. In the absence of genuine test sets for four of the six datasets, we performed our benchmark study by repeated random splitting into learning and test sets exactly as in Dudoit *et al.* (2002). The data were partitioned into a balanced learning set $\mathcal{L}$ comprising two-thirds of the arrays, used for feature preselection, tuning and fitting the classifiers. Then, the class labels of the remaining one-third of the experiments were predicted and the misclassification error was computed as the fraction of predicted class labels that differed from the true one. To reduce the variability, the splitting into learning and test sets was repeated 50 times and the error estimates were averaged. It is important to note that these results are honest in the sense that all gene filtering, classifier tuning and fitting operations were re-done on each of the 50 learning sets to allow for reliable conclusions and to avoid overoptimistic results with downward bias.

In Table 1, we report the misclassification rates of the classifiers over the six datasets. Figure 2 shows a visual illustration of these results, following the suggestions about the presentation of benchmark studies by Hothorn *et al.* (2003). The left panels show boxplots, where the median is highlighted in

Table 1. Misclassification rates for seven classifiers on six microarray data-sets based on 50 random partitions into learning sets (two-thirds of the data) and test sets (one-third of the data)

	Leukemia (%)	Colon (%)	Prostate (%)	Lymphoma (%)	SRBCT (%)	Brain (%)
BagBoost	4.08	16.10	7.53	1.62	1.24	23.86
Boosting	5.67	19.14	8.71	6.29	6.19	27.57
RanFor	1.92	14.86	9.00	1.24	3.71	33.71
SVM	1.83	15.05	7.88	1.62	2.00	28.29
PAM	3.75	11.90	16.53	5.33	2.10	25.29
DLDA	2.92	12.86	14.18	2.19	2.19	28.57
kNN	3.83	16.38	10.59	1.52	1.43	29.71

red. The right panels show density curves, where the vertical red line corresponds to the mean error rate. The performance of the classifiers varies across the different datasets, but in summary, BagBoosting, support vector machines and random forests seem to have an edge. The nearest shrunken centroid classifier (PAM), as well as the simple benchmarks NNR and DLDA do surprisingly well and can almost keep up except on the prostate data, notably the largest dataset in the analysis. This may point out that the success of such methodologically simple tools is limited to gene expression datasets with small sample size. BagBoosting achieves the lowest error rates on the prostate, SRBCT and brain data. It consistently improves upon plain boosting on all six datasets, which already legitimizes the additional computing effort by using bagged stumps as a learner in boosting.

3.2 Simulation studies

Our motivation to run simulation studies is 3-fold. First, due to the scarcity of samples in real datasets, it is hard to detect relevant differences and to assess the clinical relevance for the classifiers. We will thus run a benchmark study on independent realizations of large gene expression datasets. Second, due to the lack of knowledge about the data generating process and the underlying probability distribution on real datasets, we cannot draw any inference about BagBoostings heuristically claimed improvement of bias and variance, i.e. mean squared error. Third, since we know the true response model on the simulated data, we can check how accurately BagBoosting recovers it from data, which is important to get an idea of how well the BagBoosting model can be trusted in real datasets to draw biological conclusions. All these tasks critically hinge on a realistic simulation model for gene expression data as follows:

Step 1: Estimating correlation and means from real data
Use a real gene expression dataset of choice for estimating the $(p \times p)$-covariance matrix Σ, as well as the p-dimensional mean vectors $\mu^{(k)} = (\mu_1^{(k)}, \ldots, \mu_p^{(k)})$ from the samples of class $k \in \{0, 1\}$.

Step 2: Generating new gene expression profiles
For an arbitrary sample size n of choice repeat independently:

(i) generate a random vector by the p-dimensional multivariate standard normal distribution,

$$\mathbf{z} \sim \mathcal{N}_p(0, 1_{p \times p}).$$

(ii) Transform $\mathbf{z}$ into a gene expression profile via

$$\mathbf{x} = B\mathbf{z} + \hat{\mu}^{(k)},$$

where B is a square root of the covariance matrix Σ, determined by singular value decomposition.

Step 3: Response model
Determine conditional probabilities and class labels by one of three response models of different complexity.

(a) Additive model with 10 genes
Use a set of 10 genes, assume w.l.o.g. that these are the first 10 genes. Then,

$$F(\mathbf{x}) = \sum_{j=1}^{10} x_j.$$

(b) Weighted additive combination of 25 genes
Use a set of 25 genes, assume w.l.o.g. that these are the first 25 genes. Fix a set of coefficients β_j, e.g. randomly drawn from a uniform distribution on $[1, 3.5]$. Then,

$$F(\mathbf{x}) = \sum_{j=1}^{25} \beta_j x_j.$$

(c) Complex interaction model with 25 genes
Use a set of 25 genes, assume w.l.o.g that these are the first 25 genes. Fix three sets of coefficients β_j, γ_j and δ_j, e.g. each is randomly drawn from a uniform distribution on $[0, 2]$, $[0, \frac{2}{10}]$ and $[0, \frac{1}{10}]$, respectively. Then,

$$F(\mathbf{x}) = \sum_{j=1}^{25} \beta_j x_j \cdot \left(1 + \sum_{j=1}^{25} \gamma_j x_j\right) \cdot \left(1 + \sum_{j=1}^{25} \delta_j x_j\right).$$

We regard $F(\mathbf{x})$ as the log-odds ratio, which can be converted into a conditional class probability $p(\mathbf{x})$. Finally, the class label $y(\mathbf{x})$ is randomly generated from a Bernoulli experiment, i.e.

$$p(\mathbf{x}) = \frac{1}{1 + \exp[-F(\mathbf{x})]},$$
$$y(\mathbf{x}) \sim \text{Bernoulli}\,[p(\mathbf{x})].$$

Note that the choice of genes and coefficients happens by a random mechanism, but is fixed for all samples in a simulation
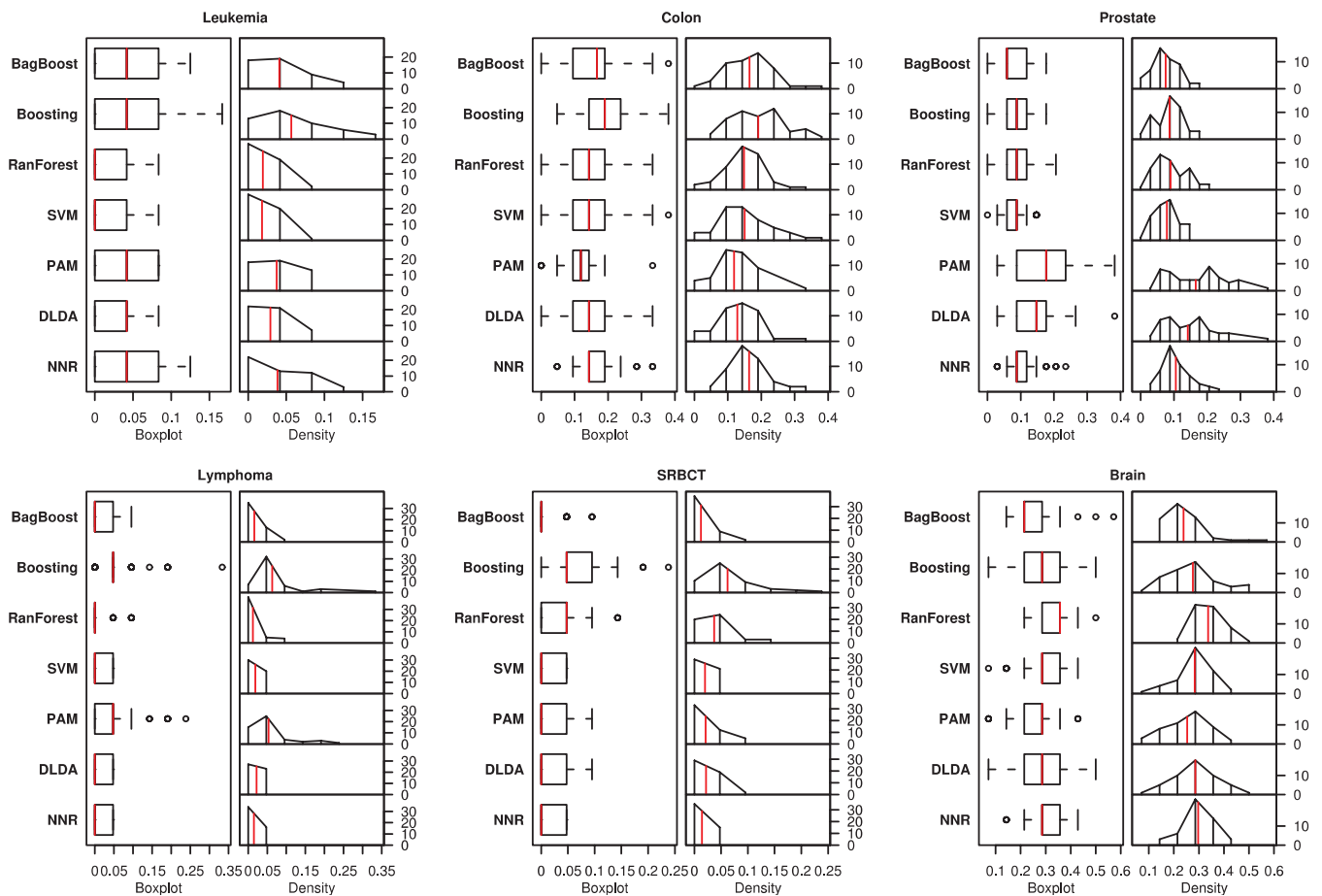
Fig. 2. Boxplots and density curves of the misclassification rates for seven classifiers on six microarray datasets based on 50 random splits into learning and test sets. The vertical red lines highlight the median (boxplots) and the mean value (density curves).

run. Using this recipe, we can generate an arbitrary number of iid gene expression profiles that follow the covariance and differential expression properties of a microarray dataset of choice. For our empirical work, we considered the structure of leukemia, colon and prostate data. Owing to space constraints and since the conclusions drawn from different structures were very similar, we display here the results for leukemia data only (for complete information refer to our Supplementary Webpage http://stat.ethz.ch/~dettling/bagboost.html). The response models yield decision boundaries of various complexity and result in a Bayes error (theoretical misclassification risk) between 5 and 10%. As this is even more than the estimated generalization error on many real gene expression datasets, we conjecture that our response models are sufficiently complicated and realistic.

3.2.1 Classification results For a discussion on the predictive potential, and for testing the null hypothesis of equal performance, we run a benchmark study on simulated gene expression data with several classifiers. In each simulation experiment, we generated a learning set of 200 arrays and

a large test set with 1000 observations, both with balanced class distributions. Exactly as on the real data, the preselection of 200 genes, as well as tuning and fitting of the classifiers was performed on the learning data, before the misclassification risk was estimated by the fraction of predicted test set class labels differing from the true one. This simulation experiment was independently repeated 100 times. The error rates are illustrated in Figure 3, where again the vertical red lines correspond to the median and mean value in the boxplots and density curves, respectively. The iid property of the error estimates also allows to formally test the null hypothesis that classifiers perform equally. We focus on pairwise comparisons of BagBoosting against its competitors. Table 2 reports the error rates for each classifier, averaged over the 100 simulation runs, as well as the number of runs where it was less accurate than BagBoosting. In a third column, we report the *p*-value of the two-sided sign test for the null hypothesis. In our simulation study with correlation and differential expression structure from the leukemia dataset, the relationship among the classifiers does not vary much over the three response models. Support vector machines is clearly the best
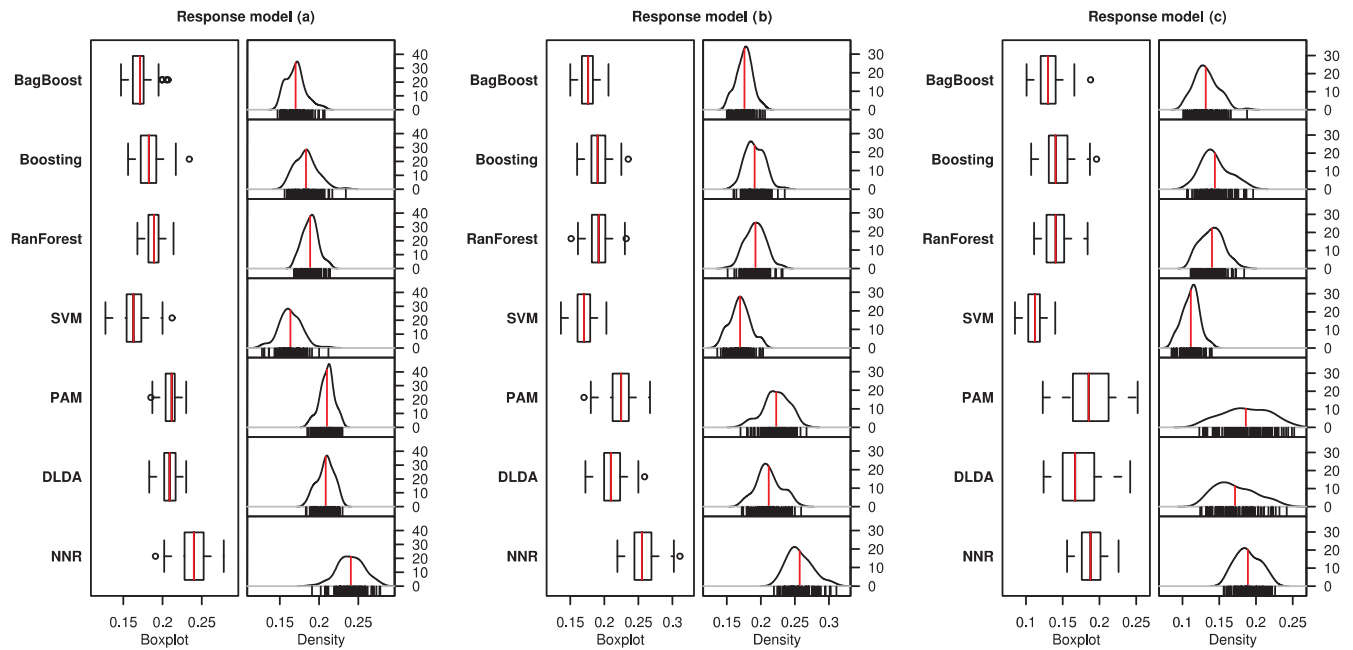
Fig. 3. Misclassification rates for outsample classification on simulated gene expression data with various classifiers and three different response models. The left panels show boxplots where the median is highlighted in red, the right panels show density curves where the red vertical line corresponds to the mean error rate.

classifier, showing a highly significant advantage in the error rates. BagBoosting is the second best and yields significantly better predictions than the remaining competitors: the hardest challenger is random forests, but even when using response model (c) with interactions of third-order, BagBoosting has an edge. This is quite surprising, since random forests are built from mid-sized trees that encompass such interaction terms, whereas BagBoosting with stumps only fits a main effect model. Although boosting shows a much better performance on simulated data when compared with real data, it remains clearly behind BagBoosting. The lag is strongly significant and increases with the complexity of the response model. The difference between these top four methods and the benchmarks PAM/DLDA/NNR is much bigger than that of the real data, yielding further evidence that the success of benchmarks is limited to small datasets.

3.2.2 Improvement of bias and variance The knowledge about the probability distribution $P[y = 1|\mathbf{x}]$ in simulation experiments allows to check whether BagBoosting has the heuristically claimed property to lower bias and variance in comparison with single, bagged and boosted stumps. There is no bias-variance decomposition of the 0/1-misclassification error, we are thus running such an analysis on the basis of the predicted conditional class probabilities $\hat{p}(\mathbf{x})$. Since we are not focusing on a particular input value $\mathbf{x}$, but want to learn about the precision of an estimated probability structure over the whole input space $\mathcal{X}$, the measure of interest is the

integrated mean squared error (IMSE)

$$\text{IMSE}(\hat{p}) = \int_{\mathcal{X}} \text{Var}(\hat{p}(\mathbf{x})) + \mathcal{E}[\hat{p}(\mathbf{x}) - p(\mathbf{x})]^2 d\mathcal{F}(\mathbf{x}),$$

where $\mathcal{F}$ is the probability distribution on the input space. The IMSE is decomposed into variance and bias and can be estimated via approximating the outer integral by randomly drawing a sufficiently large number of input values $\mathbf{x}_i$ from $\mathcal{F}$. Variance and expectation are approximated by averaging over a sufficiently large number of simulations: an estimation of integrated variance and integrated squared bias of a probability function are given by:

$$\text{Var}(\hat{p}) = \frac{1}{T} \sum_{i=1}^{T} \left[\frac{1}{S} \sum_{k=1}^{S} [\hat{p}_k(\mathbf{x}_i) - \bar{p}(\mathbf{x}_i)]^2 \right],$$

$$\text{Bias}(\hat{p})^2 = \frac{1}{T} \sum_{i=1}^{T} \left[\frac{1}{S} \sum_{k=1}^{S} \hat{p}_k(\mathbf{x}_i) - p_k(\mathbf{x}_i) \right]^2,$$

where $\hat{p}_k(\mathbf{x}_i)$ and $p_k(\mathbf{x}_i)$ denote the estimated and true probabilities in the k-th simulation run, and where $\bar{p}(\mathbf{x}_i) = \sum_{k=1}^{S} \hat{p}_k(\mathbf{x}_i)$. For checking the bias-variance properties of BagBoosting versus single stumps, bagged stumps and boosted stumps, we performed $S = 100$ simulation runs and generated learning sets of 200 observations each. Gene preselection and fitting the four predictors was re-done on each learning set, before out-of-sample probability estimates

Table 2. Misclassification rates for outsample classification on 100 iid simulation experiments with various classifiers and three different response models, as well as the number of simulations where each of the classifiers was worse than BagBoosting, and the *P*-value for the two-sided sign test that the performance is equal

	Response (a) Error (%)	Worse	*P*-value	Response (b) Error (%)	Worse	*P*-value	Response (c) Error (%)	Worse	*P*-value
BagBoost	17.02	—	—	17.57	—	—	13.19	—	—
Boosting	18.32	89	0.0000	19.09	96	0.0000	14.42	97	0.0000
RanFor	18.87	93	0.0000	19.19	87	0.0000	14.00	68	0.0004
SVM	16.33	36	0.0066	16.94	31	0.0002	11.13	7	0.0000
PAM	20.99	100	0.0000	22.25	99	0.0000	18.61	96	0.0000
DLDA	20.83	99	0.0000	21.15	98	0.0000	17.18	91	0.0000
*k*NN	24.04	100	0.0000	25.67	100	0.0000	18.90	100	0.0000

Table 3. Estimates of IMSE, variance and squared bias for conditional class probabilities, obtained from four different prediction methods on simulated gene expression data with three different response models

	Response (a) IMSE	Var	Bias^2	Response (b) IMSE	Var	Bias^2	Response (c) IMSE	Vari	Bias^2
Stumps	0.102	0.041	0.061	0.116	0.040	0.075	0.128	0.037	0.091
Bagging	0.069	0.010	0.059	0.083	0.010	0.073	0.096	0.008	0.088
Boosting	0.076	0.048	0.028	0.086	0.056	0.030	0.090	0.047	0.043
BagBoost	0.045	0.020	0.026	0.050	0.023	0.027	0.056	0.019	0.037

$\hat{p}(\mathbf{x}_i)$ for $T = 1000$ fixed test points were computed. The results are summarized in Table 3.

All our heuristical claims are confirmed by the simulation results that are consistent over the three response models. In terms of IMSE, BagBoosting is best, ranking before the about equally good bagging and boosting, whereas single stumps are clearly worse. This means that BagBoosting not only yields the best 0/1-classification, but also the most precise estimates of the conditional class probabilities. As expected from theory, bagged stumps have considerably lower variance but similar bias as single stumps. On the other hand, boosted stumps have slightly higher variance than single stumps, but they compensate by a much bigger gain in bias. Finally, as heuristically derived, BagBoosting has both lower bias and variance than stumps. The use of a bagged stump as base learner in boosting clearly pays off: while the variance is ∼60% less than for plain boosting, the bias does not increase and is even ∼10% lower. Hence, we conjecture that BagBoosting really exploits the synergy of bagging and boosting; the mechanisms for improving bias and variance work in the microarray setting. These results confirm our previous evidence that BagBoosting improves upon bagging or boosting, and that it is worth the additional computing effort.

3.2.3 Model recovery The purpose of this section is to check how well BagBoosting recovers the true response model in such a difficult setting as microarray data with thousands of highly correlated predictor variables. The inference is based on a single (but representative) learning set $\mathcal{L}$ from our simulation model with response type (a) and correlation/differential expression structure of the leukemia data. As a slight modification, we standardized all genes to unit variance to facilitate the inference. Then, we select the 200 most discriminative genes according to the Wilcoxon statistic, run BagBoosting and rewrite the fit in its componentwise additive representation from Equation (2). For each variable x_j, the smoothed step functions, obtained by combining numerous stumps, are centered to zero mean and scaled to unit variance across the 200 fitted values; this yields the univariate functions $h(x_j)$. The scaling factor is then defined as the coefficient $\hat{\theta}_j$. It reflects the importance of the j-th variable in the final BagBoosting estimate.

Figure 4 shows the true function values x_j (black dots on the diagonal), along with the smoothed step functions and their fitted values (gray circles) from the final BagBoosting estimate. Displayed are the three most important predictor variables, i.e. the genes with the biggest coefficients $\hat{\theta}_j$. Note that the estimated functions $h(x_j)$ are accurately centered, but their shape is smooth and linear in just a restricted interval around zero. This is not too surprising, since Logit(Bag)Boost mainly focuses on accurate estimation in a region around the decision boundary, and data points that are classified with higher confidence are regarded as less interesting. This argument is supported by the fact that a log-odds ratio of
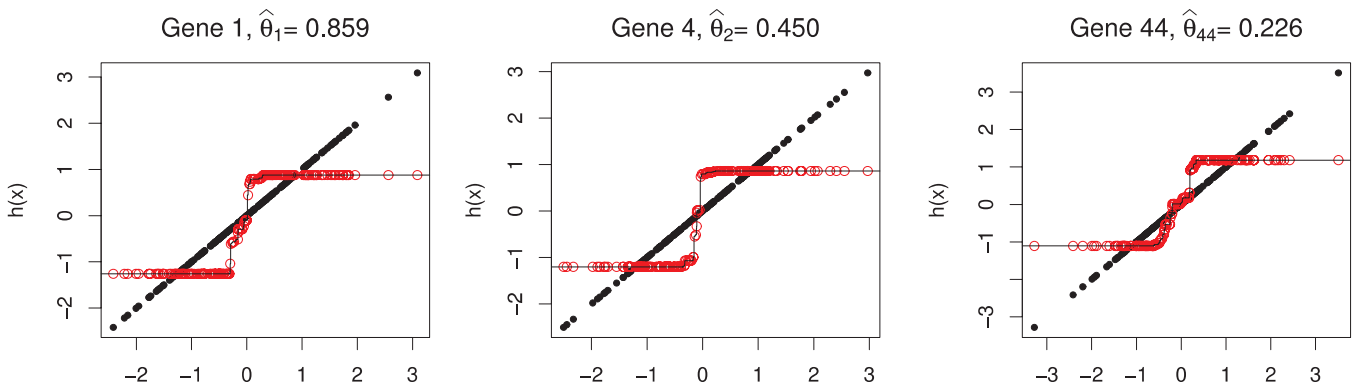
Fig. 4. BagBoosting model fit and true predictors: the black dots represent the linear univariate functions for three genes in the simulation model. Superimposed are the smoothed univariate step functions and their fitted values (grey circles), obtained by BagBoosting with stumps.

Table 4. Comparison of the 10 true and the 10 most important BagBoosting genes: given are their estimated model coefficients $\hat{\theta}_j$, the ranking $R(\hat{\theta}_j)$ of the coefficients according to their magnitude and the maximal correlation of each gene to 1 of the 10 genes from the other group

Important genes				True genes			
Gene	$R(\hat{\theta}_j)$	$\hat{\theta}_j$	Correlation	Gene	$R(\hat{\theta}_j)$	$\hat{\theta}_j$	Correlation
1	1	0.859	1.000	1	1	0.859	1.000
2	2	0.450	1.000	2	2	0.450	1.000
44	3	0.226	0.494	3	19	0.129	0.810
582	4	0.220	0.646	4	61	0.047	0.752
1026	5	0.217	0.623	5	66	0.041	0.864
1072	6	0.212	0.482	6	74	0.033	0.846
520	7	0.204	0.710	7	88	0.026	0.832
930	8	0.197	0.582	8	137	0.012	0.779
1894	9	0.188	0.661	9	149	0.009	0.655
261	10	0.183	0.859	10	155	0.008	0.694

$F(\mathbf{x}) = 1$, where the step functions level out, corresponds to an estimated probability of $p(\mathbf{x}) = 0.88$.

Table 4 reports the estimated model coefficients $\hat{\theta}_j$ for the 10 true genes from the simulation model, as well as for the 10 most important features in the BagBoosting fit. The importance ranking $R(\hat{\theta}_j)$ is determined by the magnitude of $\hat{\theta}_j$. We observe a small overlap of only two genes between the true and the most important variables. Moreover, many of the true genes have low importance $\hat{\theta}_j$, far from the theoretical value $\theta_j = 1$. On the other hand, the 10 most important genes have limited influence, too. BagBoosting spreads the responsibility on the shoulders of a larger gene set: 6 out of 200 available genes were not selected at all, 141 have a negligible influence with $\hat{\theta}_j < 0.05$, whereas the remaining 53 genes form the core for class prediction. The discrepancy between the true and the fitted model may seem disappointing, but is explained as follows. Although only 10 genes determine the hidden target function $F(\mathbf{x})$ and the exact course of the

decision boundary, many more are closely associated with the class labels y on the learning set that BagBoosting is presented with, due to the high correlation and the (often strong) differential expression for the genes. This is confirmed with our empirical analysis in Table 4, where we showed the maximal correlation of each gene from the true and the important feature set to a member of the other group. These correlations are >0.5 throughout and indicate that the true genes are substituted by very similarly regulated genes. The discrepancy between the true and the fitted model is a flaw and not only BagBoosting suffers from this flaw. When analyzing the fitted models of the competing classifiers (where possible, data not shown), we observe a similar situation, which is inherent to the highly collinear gene expression data. This emphasizes that the prediction models, despite their merits in cancer diagnosis, must be treated with enormous care for drawing biological conclusions.

4 CONCLUSIONS

The goal in class prediction with microarray data is a precise classification of cancerous malignancies at an early stage, allowing for directed and more successful therapies. Important for this task are classification algorithms that can deal with the high dimensionality of gene expression data, and that exploit as much of the available information as possible. We propose a hybrid approach of ensemble methods, where bagged stumps are employed as the base learner in boosting. For both real and simulated gene expression data, we showed that BagBoosting consistently lowers the misclassification error of plain boosting and bagging, and that it is competitive in comparison with modern prediction tools such as random forests and support vector machines. On simulated data, we furthermore provide sound empirical evidence that BagBoosting results in more precise conditional class probability estimates by reducing both the bias and variance of the base algorithm.

It has been recognized by several researchers that the introduction of randomness into ensemble schemes can improve their predictive potential. Owing to the Monte Carlo approximation of the bootstrap expectation, BagBoosting is a non-deterministic rule, too. However, we do not think that this is the principal explanation for its success, which is rather caused by a variance reduction achieved by averaging over a set of stumps, a very crude and unstable learner for microarray data. In simulation studies, this variance reduction is shown to propagate within the boosting algorithm, which however still lowers the bias as desired. BagBoosting thus really combines the advantages of the two ensemble methods it is built from.

Critical voices are often concerned that the time-consuming effort of fitting very complex prediction tools, such as (Bag)Boosting, random forests and so on, is not legitimated by the small empirical improvement over much simpler methods on microarray data. Although at present, the advantage of BagBoosting versus the NNR or DLDA on gene expression data with limited number of samples is not very big in terms of the 0/1-misclassification error. Whereas the difference grows to significant size on larger microarray problems and becomes very pronounced on simulated datasets with more than 1000 expression profiles. While already a small improvement today can save an additional patients life, sophisticated class prediction tools such as BagBoosting are expected to display their full benefit only in future studies with larger sample size.

REFERENCES

Alizadeh,A.A., Eisen,M.B., Davis,R.E., Ma,C., Lossos,I.S., Rosenwald,A., Boldrick,J.C., Sabet,H., Tran,T., Yu,X. *et al.* (2000) Distinct types of diffuse large B-Cell-lymphoma identified by gene expression profiling. *Nature*, **403**, 503–511.

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

Ben-Dor,A., Bruhn,L., Friedman,N., Nachman,I., Schummer,M. and Yakhini,Z. (2000) Tissue classification with gene expression profiles. *J. Comput. Biol.*, **7**, 559–583.

Breiman,L. (1996) Bagging predictors. *Mach. Learn.*, **24**, 123–140.

Breiman,L. (2001a) Using iterated bagging to debias regressions. *Mach. Learn.*, **45**, 261–277.

Breiman,L. (2001b) Random forests. *Mach. Learn.*, **45**, 5–32.

Bühlmann,P. and Yu,B. (2000) Invited discussion on additive logistic regression: a statistical view of boosting. *Ann. Stat.*, **28**, 377–386.

Bühlmann,P. and Yu,B. (2003) Boosting with the L_2-loss: regression and classification. *J. Am. Stat. Assoc.*, **98**, 324–339.

Burges,C. (1998) A tutorial on support vector machines for pattern recognition. *Knowl. Discov. Data Min.*, **2**, 121–167.

Chang,C. and Lin,C. (2001) LIBSVM: a library for support vector machines.

Dettling,M. and Bühlmann,P. (2003) Boosting for tumor classification with gene expression data. *Bioinformatics*, **19**, 1061–1069.

Dudoit,S. and Fridlyand,J. (2003) Classification in microarray experiments. In Terry Speed (ed.) *Statistical Analysis of Gene Expression Data*. Chapman and Hall, NY, pp. 93–158.

Dudoit,S., Fridlyand,J. and Speed,T. (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. *J. Am. Stat. Assoc.*, **97**, 77–87.

Freund,Y. and Schapire,R. (1996) Experiments with a new boosting algorithm. *Machine Learning: Proceedings to the Thirteenth International Conference*. Morgan Kauffman, San Francisco, CA, pp. 148–156.

Friedman,J. (2002) Stochastic gradient boosting. *Comp. Stat. Data Anal.*, **38**, 367–368.

Friedman,J. and Popescu,B. (2003) Importance sampled learning ensembles. *Technical Report*, Department of Statistics, Stanford University, CA.

Friedman,J., Hastie,T. and Tibshirani,R. (2000) Additive logistic regression: a statistical view of boosting. *Ann. Stat.*, **28**, 337–407 (with Discussion).

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

Hothorn,T., Leisch,F., Zeileis,A. and Hornik,K. (2003) The design and analysis of benchmark experiments. *Technical Report Number 82*. Institut für Statistik, Wirtschaftsuniversität Wien.

Khan,J., Wei,J., Ringner,M., Saal,L., Ladanyi,M., Westermann,F., Berthold,F., Schwab,M., Antonescu,C. *et al.* (2001) Classification and diagnostic prediction of cancer using gene expression profiling and artificial neural networks. *Nat. Med.*, **6**, 673–679.

Liaw,A. and Wiener,M. (2002) Classification and regression by `randomForest`. *R News*, **2**, 18–22.

Meyer,D. (2001) Support vector machines. *R News*, **1**, 23–26.

Meyer,D., Leisch,F. and Hornik,K. (2003) The support vector machine under test. *Neurocomputing*, **55**, 169–186.

Pomeroy,S., Tamayo,P., Gaasenbeek,M., Sturla,L., Angelo,M., McLaughlin,M., Kim,J., Goumnerova,L., Black,P., Lau,C. *et al.* (2002) Prediction of central nervous system embryonal tumor outcome based on gene expression. *Nature*, **415**, 436–442.

R Development Core Team (2004) R: A Language and Environment for Statistical Computing, Vienna, Austria, ISBN 3-900051-00-3.

Singh,D., Febbo,P., Ross,K., Jackson,D., Manola,J., Ladd,C., Tamayo,P., Renshaw,A., D'Amico,A., Richie,J. *et al.* (2002) Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, **1**, 203–209.

Tibshirani,R., Hastie,T., Narasimhan,B. and Chu,G. (2002) Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc. Natl Acad. Sci. USA*, **99**, 6567–6572.