DOI 10.1007/s10115-003-0116-7
Springer-Verlag London Ltd. © 2004
Knowledge and Information Systems (2004) 6: 645–663

Knowledge and Information Systems

Convex Hull Ensemble Machine for Regression and Classification

Yongdai Kim[1], Jinseog Kim[2]

[1]Department of Statistics, Ewha Womans University, Korea
[2]Department of Statistics, National University, Seoul, Korea

Abstract. We propose a new ensemble algorithm called Convex Hull Ensemble Machine (CHEM). CHEM in Hilbert space is first developed and modified for regression and classification problems. We prove that the ensemble model converges to the optimal model in Hilbert space under regularity conditions. Empirical studies reveal that, for classification problems, CHEM has a prediction accuracy similar to that of boosting, but CHEM is much more robust with respect to output noise and never overfits datasets even when boosting does. For regression problems, CHEM is competitive with other ensemble methods such as gradient boosting and bagging.

Keywords: Bagging; Boosting; Classification; Ensemble; Regression

1. Introduction

Ensemble methods, which construct many classifiers (called "base learners") and combine them to make a final decision, have shown great success in statistics and machine learning areas for their significant improvement in classification accuracy. Examples of ensemble algorithms include bagging (Breiman 1996), boosting (Freund and Schapire 1997), arcing (Breiman 1998), and random forest (Breiman 2001). Of these, bagging and boosting are the two most popular ensemble methods. A number of empirical studies have compared bagging and boosting, including the studies by Quinlan (1996), Bauer and Kohavi (1999), Opitz and Maclin (1999), and Dietterich (2000), to name a few. Their results indicated that while boosting is more accurate than bagging in most cases, boosting can overfit highly noisy dataets, thus decreasing its performance. Ridgeway (2000) gives a simple example in which boosting seriously overfits the data, and Breiman (2001) demonstrates the vulnerability of boosting to output noise.

Received 9 December 2002
Revised 21 February 2003
Accepted 4 June 2003
Published online 15 January 2004

In this paper, we propose a new ensemble algorithm called Convex Hull Ensemble Machine (CHEM), which combines the advantages of bagging and boosting. The classification accuracy of CHEM for low-noise cases compares favorably to that of boosting, but CHEM is more robust to output noise.

We begin by investigating the instability that exists in decision trees. In Sect. 2, we argue that instability arises in decision trees when there are many different decision trees that explain the data similarly. Furthermore, we argue that the existence of many such decision trees occurs when the true model is located not inside the set of decision trees but inside the convex hull of the set of decision trees. Then we devise a simple geometry to explain this situation.

Once we construct the geometry for the instability of decision trees, we develop a hypothetical algorithm for CHEM in Hilbert space, which constructs a sequence of convex combinations of base learners (decision trees) that converges to the optimal model located inside the convex hull of the set of base learners. Although hypothetical, the CHEM in Hilbert space provides useful insight into the aim of CHEM and how CHEM accomplishes it in regression and classification problems.

After establishing CHEM in Hilbert space, CHEM algorithms for regression and classification problems are developed. For regression problems, we develop a CHEM algorithm that simply replaces the inner product used in CHEM in Hilbert space with its empirical counterpart. For classification problems, we embed the problem into a function estimation problem in Hilbert space based on the symmetric logistic regression model and modify the CHEM algorithm in Hilbert space in the same way as for the regression problems.

CHEM unifies regression and classification problems into a function estimation problem in Hilbert space and performs well for both problems. Boosting can also be explained as a way of estimating the optimal function by using the gradient descent method or a Newton-like method (Schapire and Singer 1999; Friedman et al. 2000; Friedman 2001; Mason et al. 2000). However, this interpretation results in overfitting in regression problems (Bühlman and Yu 2000).

Empirical results are given in Sect. 6. Fourteen real datasets used for classification problems from the UC-Irvine machine learning archive are analyzed, and five datasets (two from the UC-Irvine machine learning archive and three synthetic models) are used for regression problems. The empirical results for the classification problems indicate that the classification accuracy of CHEM compare favorably with that of boosting, that CHEM is much more robust to output noise, and that CHEM never overfits. In regression problems, the performance of CHEM is competitive with other ensemble methods such as gradient boosting and bagging.

This paper is organized as follows. Section 2 studies the instability of decision trees. In Sect. 3, the CHEM algorithm in Hilbert space is proposed. CHEM algorithms for regression and classification problems are proposed in Sects. 4 and 5, respectively. Empirical results are presented in Sect. 6, and a discussion follows in Sect. 7.

2. Instability in Decision Trees

Figure 1 shows an example of instability in decision trees. The two regression trees in the figure are constructed from two bootstrap samples of the same dataset. The structures of the two trees are completely different, although the two bootstrap samples are thought to be similar. In particular, the split variables of the two trees are completely different. In this section, we explain why decision trees are unstable.

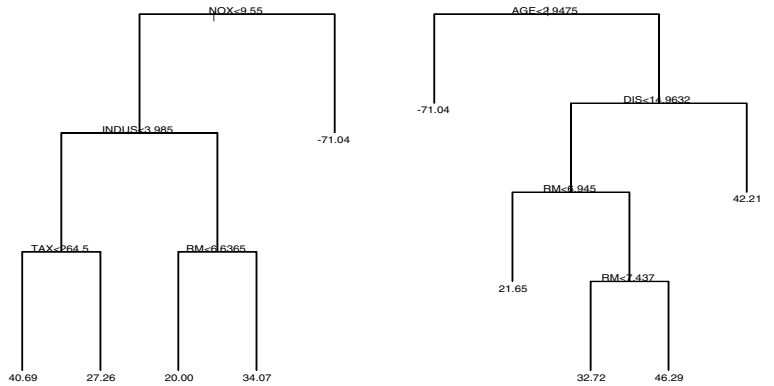

Fig. 1. Two regression trees constructed from two bootstrap samples of the same dataset "Boston Housing".

Based on this explanation, we devise a simple geometry that is used extensively in the following sections.

Let $\mathcal{L} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ be n input-output pairs of the training dataset, which is a random sample of a random vector $(\mathbf{X}, Y)$ whose probability measure is $P(\mathbf{x}, y)$. Here $\mathbf{X} \in R^p$ and $Y \in R$. The true regression model $f^*(\mathbf{x})$ is defined by $f^*(\mathbf{x}) = \mathrm{E}(Y|\mathbf{X} = \mathbf{x})$, and the objective of the regression problem is to estimate f^* based on the training dataset $\mathcal{L}$. For a given class of models $\mathcal{F}$, a typical procedure for estimating f^* is to choose a model $\hat{f}$ in $\mathcal{F}$ that minimizes the square error loss, that is,

$$\hat{f} = \arg\ \min_{f \in \mathcal{F}} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2.$$

Consider the following example. Suppose $X \sim N(0, 1)$ and $Y = I(|X| < c)$, where c is chosen so that $\Pr(|X| < c) = 0.5$. In this setting, $f^*(x) = I(|x| < c)$. Now suppose the class of models $\mathcal{F}$ consists of all stumps (decision trees with only two terminal nodes). That is, $\mathcal{F}$ is given by

$$\left\{ f_\theta(x) = a_L I(x \le \eta) + a_R I(x > \eta) : \theta = (a_L, a_R, \eta) \in R^3 \right\}.$$

It is easy to see that $\hat{\eta}$, the least square estimator of η, is given by

$$\begin{cases} \hat{\eta} = c & \text{if} \quad n_R > n_L \\ \hat{\eta} = -c & \text{if} \quad n_R < n_L, \end{cases}$$

where $n_R = \sum_{i=1}^{n} I(x_i > c)$ and $n_L = \sum_{i=1}^{n} I(x_i < -c)$. Note that $\hat{\eta}$ is a very unstable estimator of η. For example, suppose $n_R = n_L - 1$. Then $\hat{\eta} = c$. However, only two additional observations less than $-c$ make $\hat{\eta}$ move from c to $-c$. Furthermore, this instability persists regardless of the size of n.

One explanation for the instability in this example is that there are two models $f_1(x) = I(x < c)$ and $f_2(x) = I(x > -c)$ in $\mathcal{F}$ that are equally close to the true model $f^*(x)$. In fact, we have

$$\int (y - f_1(x))^2 P(dx, dy) = \int (y - f_2(x))^2 P(dx, dy).$$

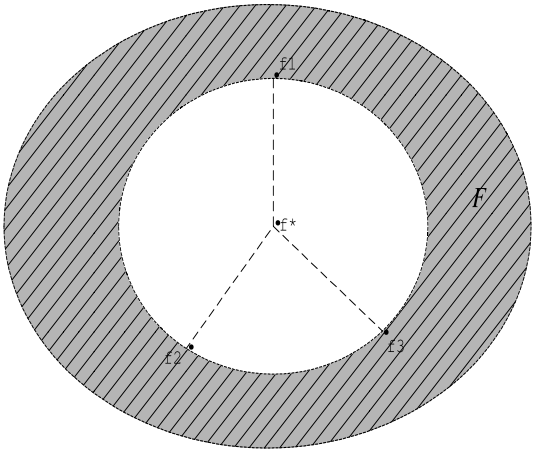

Fig. 2. Assumed geometry.

With finite samples, we compare $\int (y-f_1(x))^2 P_n(dx, dy)$ and $\int (y-f_2(x))^2 P_n(dx, dy)$ and choose the model with the smaller value. Here, $P_n(x, y) = \sum_{i=1}^{n} I((x, y) = (x_i, y_i))/n$. Since P_n is a close approximation of P, $\int (y - f_1(x))^2 P_n(dx, dy)$ and $\int (y - f_2(x))^2 P_n(dx, dy)$ are expected to be close to each other. Hence the choice of the final model depends entirely on the small deviation of P_n from P, which is mainly due to random noise.

Based on these arguments, we conclude that instability arises in decision trees when many models explain a given dataset similarly. The next question is why there are many such models in $\mathcal{F}$. We claim that many such models exist when $\mathcal{F}$ is not convex and the true model is not located inside $\mathcal{F}$. Figure 2 describes this situation. The shaded area in the figure is $\mathcal{F}$, and the distances of f_1, f_2, and f_3 from f^* are all equal. Suppose the data are given as $f^* + \epsilon$, where ϵ is noise. Then the optimal model from the data (i.e., the model closest to $f^* + \epsilon$) depends entirely on ϵ, and thus instability emerges.

In what follows, the geometry depicted in Fig. 2 is always assumed. That is, the class of models $\mathcal{F}$ is a nonconvex set and the true model is not in $\mathcal{F}$. More importantly, the geometry in Fig. 2 assumes that the true model is located inside the convex hull of $\mathcal{F}$. That is, f^* can be represented by

$$f^* = \sum_{i=1}^{\infty} w_i f_i / \sum_{i=1}^{\infty} w_i$$

for some sequences of models $f_1, f_2, \ldots$ in $\mathcal{F}$ and weights $w_1, w_2, \ldots$. Suppose that $\mathcal{F}$ is a subset of Hilbert space equipped with the inner product $<, >$. Then, an assumption equivalent to the geometry in Fig. 2 is:

A1. f^* is located inside the convex hull of $\mathcal{F}$;
A2. for any g in the convex hull of $\mathcal{F}$, $\mathcal{F}(g)$ is a nonempty set where

$$\mathcal{F}(g) = \{f :< g - f^*, f - f^* >= 0, f \in \mathcal{F}\};$$

A3. there exists a positive constant ρ such that $\inf \|f\| < \rho$ on $f \in \mathcal{F}(g)$ and for all g, where $\|f\|^2 =< f, f >$.

A1 is the fundamental assumption. **A2** means that the shaded area in Fig. 2 (i.e., $\mathcal{F}$) encompasses f^* completely. **A3** implies that the distance from f^* to $\mathcal{F}$ at any angle is bounded by ρ. In Sect. 3, we develop an algorithm that constructs a sequence of convex combinations of models in $\mathcal{F}$ that converges to f^* under the assumptions **A1**, **A2**, and **A3**.

Let $\hat{f} = \text{argmin}_{f\in\mathcal{F}} \|f-(f^*+\epsilon)\|$. For a given $\mathcal{F}$ to satisfy the assumed geometry in Fig. 2, $\hat{f}$ should have a small bias (and hence a large variance) since the average of f_1, f_2, and f_3 is exactly the same as f^*. One such class of models is the set of unpruned decision trees, which is used in the empirical study in Sect. 6.

Hereafter a model in $\mathcal{F}$ is called a "base learner" and any convex combination of finite base learners is called an "ensemble model".

3. CHEM in Hilbert Space

In this section, we explain how CHEM constructs a sequence of ensemble models in Hilbert space under the assumed geometry shown in Fig. 2. Suppose the m-th ensemble model H_m is given. Then CHEM updates the ensemble model H_m to H_{m+1} as follows. First, CHEM finds the model f_{m+1} in $\mathcal{F}$ where

$$f_{m+1} = \text{argmin}_{f\in\mathcal{F}_m} \|f - f^*\|$$

and $\mathcal{F}_m = \{f \in \mathcal{F} : f - f^* \perp f - H_m\}$. That is, f_{m+1} is the closest model to f^* satisfying $f - f^* \perp f - H_m$. After constructing f_{m+1}, CHEM updates the ensemble model by

$$H_{m+1} = \frac{u_m H_m + w_{m+1} f_{m+1}}{u_m + w_{m+1}},$$

where u_m and w_{m+1} are chosen so that $\|H_{m+1} - f^*\|$ is minimized.

The step-by-step description of CHEM is as follows. We construct the first base learner f_1 by

$$f_1 = \text{argmin}_{f\in\mathcal{F}} \|f - f^*\|,$$

and we let $H_1 = f_1$. This procedure is depicted in Fig. 3. There, $d_1 = \|f_1 - f^*\|$. The second base learner is the closest one to f^* satisfying $f_2 - f^* \perp H_1 - f^*$. Then, the second ensemble model H_2 is given by

$$H_2 = \frac{u_1 H_1 + w_2 f_2}{u_1 + w_2},$$

choosing u_1 and w_2 to minimize the distance between H_2 and f^*. Simple algebra yields $u_1 = 1/d_1^1$ and $w_2 = 1/d_2^2$, where $d_2 = \|f_2 - f^*\|$. Hence H_2 becomes

$$H_2 = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2},$$

where $w_i = 1/d_i^2$ for $i = 1, 2$. This procedure is summarized in Fig. 4.

The third base learner is constructed similarly, and direct calculation shows that the third ensemble model H_3 is given by

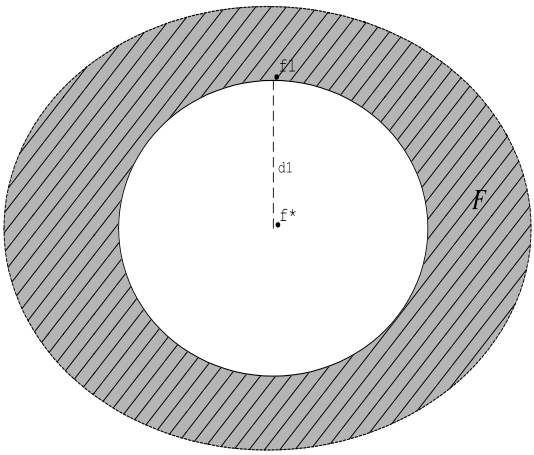

Fig. 3. Construction of the first ensemble model.

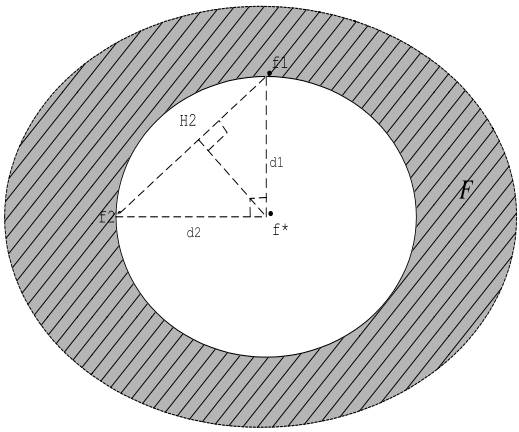

Fig. 4. Construction of the second ensemble model.

$$H_3 = \frac{w_1 f_1 + w_2 f_2 + w_3 f_3}{w_1 + w_2 + w_3},$$

where $w_i = 1/d_i^2$ for $i = 1, 2, 3$. This step is described in Fig. 5.

In this way, we can keep constructing base learners $f_4, f_5, \ldots$ and ensemble models $H_4, H_5, \ldots$ by

$$H_m = \frac{\sum_{i=1}^{m} w_i f_i}{\sum_{i=1}^{m} w_i},$$

where $w_i = 1/d_i^2$ and $d_i = \|f_i - f^*\|$.

The following theorem proves that the sequence of the ensemble models H_m constructed by CHEM in Hilbert space converges to f^* under the assumed geometry.

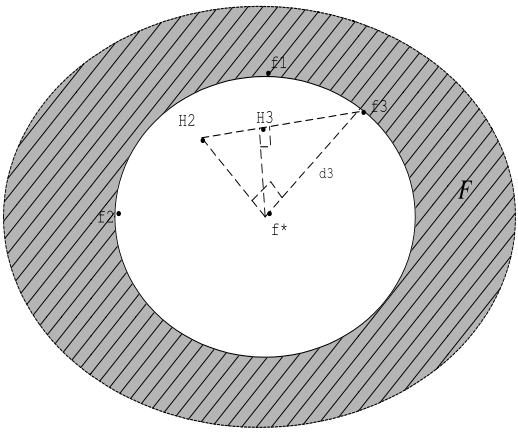

Fig. 5. Construction of the third ensemble model.

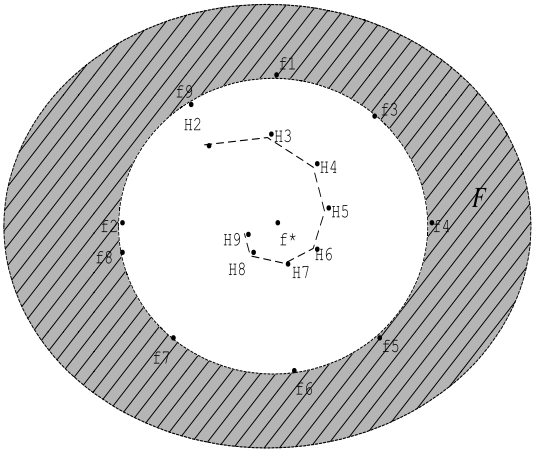

Fig. 6. Construction of the sequence of ensemble models.

Theorem 3.1. Under the assumptions **A1**, **A2**, and **A3**,

$$\|H_m - f^*\| \to 0$$

as $m \to \infty$.

Proof. Suppose that $\liminf_m \|H_m - f^*\| > \epsilon > 0$. For a given H_m, by the definition of f_m we have $\|f_{m+1} - f^*\| \leq \|f - f^*\|$ for all $f \in \mathcal{F}(H_m)$. Hence we have

$$\|H_{m+1} - f^*\| \leq \inf_{0 \leq \alpha \leq 1} \|\alpha H_m + (1-\alpha) f - f^*\|. \tag{1}$$

Since $\|H_{m+1} - f^*\| \leq \|H_m - f^*\|$, $\liminf_m \|H_m - f^*\| > \epsilon$ is equivalent to $\|H_m - f^*\| > \epsilon$ for all m. With $\|f - f^*\| \leq \|f\| + \|f\|^* \leq \rho + \|f^*\|$, simple calculations

yield

$$\inf_{0\le\alpha\le 1}\|\alpha H_m+(1-\alpha)f-f^*\|=\|H_m-f^*\|\sqrt{\frac{\|f-f^*\|^2}{\|H_m-f^*\|^2+\|f-f^*\|^2}}$$
$$\le\|H_m-f^*\|\sqrt{\frac{(\rho+\|f^*\|)^2}{\epsilon^2+(\rho+\|f^*\|)^2}}. \quad (2)$$

Hence combining (1) and (2), we have

$$\|H_{m+1}-f^*\|\le\tau\|H_m-f^*\|,$$

where

$$\tau=\sqrt{\frac{(\rho+\|f^*\|)^2}{\epsilon^2+(\rho+\|f^*\|)^2}}.$$

Since $\tau<1$, we have $\|H_m-f^*\|\le\tau^n\|H_1-f^*\|\to 0$, which contradicts the assumption $\liminf_m\|H_m-f^*\|>\epsilon>0$, and the proof is done. □

4. CHEM for Regression

In this section, we present the CHEM algorithm for regression. Recall that the training dataset consists of n I/O pairs $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_n,y_n)\}$, which is a random sample of $(\mathbf{X},Y)$ distributed according to an unknown joint distribution $P(x,y)$. Here $\mathbf{X}\in R^p$ and $Y\in R$. We assume that

$$Y=f^*(\mathbf{X})+\epsilon,$$

where $\mathrm{E}(\epsilon)=0$ and $\mathrm{Var}(\epsilon)=\sigma^2>0$. For regression problems, the objective of CHEM is to estimate f^* based on the sample $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_n,y_n)\}$.

To apply CHEM in Hilbert space to regression problems, we need two devices: (i) to measure the distance of a given base learner to the true model (i.e., $\|f-f^*\|$ for a given $f\in\mathcal{F}$) and (ii) to construct f_{n+1} for a given H_n. For (i), for the square error loss function $l(y,a)=(y-a)^2$, define the deviance of base learner f by

$$d(f)=\sum_{i=1}^{n}l(y_i,f(\mathbf{x}_i))/n.$$

Then we use $d(f)$ as a measure of $\|f-f^*\|^2$ on $L_2(P)$ – the Hilbert space whose inner product is defined by $<f,g>=\int f(\mathbf{x})g(\mathbf{x})P(d\mathbf{x})$. This is a reasonable choice because $d(f)$ converges to $\|f-f^*\|^2+\sigma^2$ under regularity conditions and σ^2 is smaller than $\|f-f^*\|^2$ for unstable base learners.

For (ii), we construct the model ϕ based on the residuals of H_n and project the data onto the direction ϕ. That is, we set

$$f_{n+1}=\mathrm{argmin}_{f\in\mathcal{F}_n}d(f),$$

where $\mathcal{F}_n=\{\eta\phi:\eta\in R\}$. Since the data are considered an approximation of f^*, we have approximately $<f_{n+1}-f^*,H_m-f^*>=0$. That is, we first find the appropriate direction ϕ and construct the optimal model for that direction.

Using these two devices, we propose the following CHEM algorithm for regression problems. Recall that $\mathcal{F}$ is a set of base learners.

Algorithm 1. CHEM algorithm for the regression model.

1. Initialization: Set $z_i = y_i$ for $i = 1, \ldots, n$.
2. Repeat $m = 1, \ldots, M$
   (a) Fit a regression model ϕ in $\mathcal{F}$ with output variables z_i and input variables $\mathbf{x}_i$.
   (b) Calculate the correction factor η by

$$\eta = \text{argmin}_\delta d(\delta\phi).$$

   (c) Set $f_m(\mathbf{x}) = \eta\phi(\mathbf{x})$.
   (d) Update the ensemble model

$$H_m(\mathbf{x}) = \frac{\sum_{i=1}^m w_i f_i(\mathbf{x})}{\sum_{i=1}^m w_i},$$

   where $w_i = 1/d(f_i)$.
   (e) Update the new response $z_i = y_i - H_m(\mathbf{x}_i)$.
3. For a given new data with input $\mathbf{x}$, predict the output as $H_M(\mathbf{x})$.

5. CHEM for Classification

First, we consider a two-class problem (i.e., $Y \in \{-1, 1\}$). We assume the symmetric logistic model

$$\Pr(Y = 1|\mathbf{X} = \mathbf{x}) = \frac{\exp(f(\mathbf{x}))}{\exp(-f(\mathbf{x})) + \exp(f(\mathbf{x}))}$$

and embed the classification problem into the function estimation problem (i.e., estimation of f). With this setup, the CHEM algorithm for regression given in Sect. 4 will be modified. First, we use the negative log-likelihood of the binomial distribution as a loss function instead of the squared error loss. Then, the deviance of a function f is

$$d(f) = \sum_{i=1}^n \log(1 + \exp(-2y_i f(\mathbf{x}_i)))/n.$$

Second, for residuals for the given ensemble model H_m, we use the Pearson residual defined by

$$r_i = \frac{y_i^* - P_m(\mathbf{x}_i)}{\sqrt{P_m(\mathbf{x}_i)(1 - P_m(\mathbf{x}_i))}},$$

where $y_i^* = 2y_i - 1$, $P_m(\mathbf{x}) = \exp(H_m(\mathbf{x}))/(\exp(-H_m(\mathbf{x})) + \exp(H_m(\mathbf{x})))$, and $H_m(\mathbf{x})$ is the m-th ensemble model. However, we do not use the residuals $\{r_i\}$ as a response variable. Instead, we use $|r_i|$ as a weight for the i-th observation and construct ϕ on a weighted bootstrap sample of the original sample $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ with weights $\{|r_1|, \ldots, |r_n|\}$. In summary, the algorithm of CHEM for the two-class problem is given below.

Algorithm 2. CHEM algorithm for two-class classification problem.

1. Initialization: let the weights $v_i = 1/n$ for $i = 1, \ldots, n$.
2. Repeat $m = 1, \ldots, M$
   (a) Make a bootstrap sample $\mathcal{L}^B$ with weights $\{v_i\}$.
   (b) Estimate $p(\mathbf{x}) = \hat{P}(Y = 1|\mathbf{X} = \mathbf{x})$ using $\mathcal{L}^B$ with a given class of base learners.
   (c) Let $\phi(\mathbf{x}) = \frac{1}{2}\log(p(\mathbf{x})/(1 - p(\mathbf{x})))$.
   (d) Calculate the correction factor η by

$$\eta = \text{argmin}_\delta d(\delta\phi).$$

   (e) Let $f_m(\mathbf{x}) = \eta f_m(\mathbf{x})$.
   (f) Update the ensemble model $H_m(\mathbf{x}) = \sum_{i=1}^m w_i f_i(\mathbf{x}) / \sum_{i=1}^m w_i$, where $w_i = 1/d(f_i)$.
   (g) Update the weights $\{v_i\}$ by

$$v_i = \left| \frac{y_i^* - P_m(\mathbf{x}_i)}{\sqrt{P_m(\mathbf{x}_i)(1 - P_m(\mathbf{x}_i))}} \right|,$$

   where $P_m(\mathbf{x}) = \exp(H_m(\mathbf{x}))/(\exp(-H_m(\mathbf{x})) + \exp(H_m(\mathbf{x})))$.
3. For a new input $\mathbf{x}$, assign it to class 1 if $H_M(\mathbf{x}) > 0$ and to class -1 otherwise.

For multiclass problems (i.e., $Y \in \{1, \ldots, J\}$, $J > 2$), we assume the symmetric logistic model:

$$\Pr(Y = k|\mathbf{X} = \mathbf{x}) = \frac{\exp(f_k(\mathbf{x}))}{\sum_{j=1}^J \exp(f_j(\mathbf{x}))}.$$

To extend the CHEM algorithm for two-class problems to multiclass problems, we mimic the algorithm of the multiclass LogitBoost (Friedman et al. 2000). Consider J many two-class classification problems. The j-th base learner $f_j(\mathbf{x})$ is constructed from the j-th two-class problems in which new response variables $\{y_{ij}^* = I(y_i = j), i = 1, \ldots, n\}$ are used. Then, f_j's are centered by

$$f_j(\mathbf{x}) = f_j(\mathbf{x}) - \sum_{k=1}^J f_k(\mathbf{x})/J. \tag{3}$$

Then the correction factor η is obtained by using the negative log-likelihood of the multinomial distribution as a loss function, and $\underline{f} = (f_1, \ldots, f_J)$ is updated accordingly. In this setup, the deviance of $\underline{f}$ is given by

$$d(\underline{f}) = -\sum_{i=1}^n \left[f_{y_i}(\mathbf{x}_i) - \log\left(\sum_{k=1}^J \exp(f_k(\mathbf{x}_i)) \right) \right].$$

To summarize, we obtained the following CHEM algorithm for multiclass problems.

Algorithm 3. CHEM algorithm for J class classification problem.

1. Initialization
   (a) Set weights $\{v_{ij}\}$ by $v_{ij} = 1/n$ for $i = 1, \ldots, n$ and $j = 1, \ldots, J$.
   (b) Set $y^*_{ij} = I(y_i = j)$ for $i = 1, \ldots, n$ and $j = 1, \ldots, J$.
2. Repeat $m = 1, \ldots, M$
   (a) Repeat $j = 1, \ldots, J$
      i. Make a bootstrap sample $\mathcal{L}^B_j$ from $\{(y^*_{1j}, \mathbf{x}_1), \ldots, (y^*_{nj}, \mathbf{x}_n)\}$ with weights $\{v_{1j}, \ldots, v_{nj}\}$.
      ii. Estimate $p_j(\mathbf{x}) = \hat{P}(Y^*_j = 1 | \mathbf{X} = \mathbf{x})$ using $\mathcal{L}^B_j$ with a given class of base learners.
      iii. Set $\phi_j(\mathbf{x}) = \frac{1}{2}\log(p_j(\mathbf{x})/(1 - p_j(\mathbf{x})))$.
   (b) Set $\phi_j(\mathbf{x}) = \phi_j(\mathbf{x}) - \sum_{k=1}^{J} \phi_k(\mathbf{x})/J$ for $j = 1, \ldots, J$.
   (c) Calculate the correction factor η by

$$\eta = \text{argmin}_\delta d(\delta \underline{\phi}),$$

   where $\underline{\phi} = (\phi_1, \ldots, \phi_J)$.
   (d) Let $\underline{f}_m(\mathbf{x}) = \eta \underline{\phi}(\mathbf{x})$.
   (e) Update the ensemble model $\underline{H}_m(\mathbf{x}) = \sum_{i=1}^{m} w_i \underline{f}_i(\mathbf{x}) / \sum_{i=1}^{m} w_i$, where $w_i = 1/d(\underline{f}_i)$.
   (f) For $j = 1, \ldots, J$, update the weights $\{v_{ij}\}$ by

$$v_{ij} = \left| \frac{y^*_{ij} - P_{mj}(\mathbf{x}_i)}{\sqrt{P_{mj}(\mathbf{x}_i)(1 - P_{mj}(\mathbf{x}_i))}} \right|,$$

   where $P_{mj}(\mathbf{x}) = \exp(H_{mj}(\mathbf{x}))/(\sum_{k=1}^{J} \exp(H_{mk}(\mathbf{x})))$.
3. Assign a new datum with input variable $\mathbf{x}$ to class $\text{argmax}_j H_{Mj}(\mathbf{x})$.

Remark. Friedman et al. (2000) proposed using

$$f_j(\mathbf{x}) = \frac{J-1}{J}\left(f_j(\mathbf{x}) - \sum_{k=1}^{J} f_k(\mathbf{x})/J\right) \tag{4}$$

instead of (3). The only difference between (3) and (4) is the constant term $(J-1)/J$. In CHEM, this constant term is replaced by the correction factor η.

6. Empirical Studies

In this section, we present empirical results for comparing various aspects of CHEM with boosting and bagging. We focus mainly on classification problems and consider regression problems briefly in the last subsection.

Table 1. Datasets used for classification. CV = cross validation.

ID	Dataset	Training	Test	Class
1	Breast cancer	699	CV	2
2	Pime-Indian-Diabetes	768	CV	2
3	German	1000	CV	2
4	Glass	214	CV	7
5	House-vote-84	435	CV	2
6	Image	210	2100	7
7	Ionosphere	351	CV	2
8	kr-vs-kp	3196	CV	2
9	Letter	16000	4000	26
10	Satimage	4435	2000	6
11	Sonar	210	CV	2
12	Vehicle	846	CV	2
13	Vowel	528	462	11
14	Waveform	300	5000	3

6.1. Setup for Classification

For base learners, CHEM and bagging use unpruned trees (the largest of the trees whose terminal] nodes have no less than five instances), while boosting uses best-first trees (Friedman et al. 2000) with eight terminal nodes. For the final ensemble model, 50 base learners are combined in bagging and 500 base learners are combined in CHEM and boosting. We analyzed 14 benchmark datasets from the UC-Irvine machine learning archive. Table 1 summarizes the characteristics of the datasets. For datasets without test samples, the generalization errors (test set misclassification errors) are calculated using ten repetitions of ten-fold cross validation.

6.2. Generalization Error

Table 2 presents the generalization errors. The generalization errors of CHEM compare favorably with those of boosting. For exactly half of the datasets (7 out of 14 datasets), CHEM has lower generalization errors than boosting, and vice versa. In comparison with bagging, CHEM has lower generalization errors in most cases (10 out of 14 datasets). These results are summarized in Fig. 7, which compares the improvement rates of CHEM and boosting over bagging. The improvement rate of CHEM over bagging (x-axis) is defined by the difference of generalization errors of bagging and CHEM divided by the generalization error of bagging. The improvement rate of boosting over bagging (y-axis) is defined similarly. Most of the datasets locate in the first and third quadrants, which means that CHEM and boosting either improve or are poorer than the performance of bagging simultaneously. However, note that most of the datasets in the third quadrant locate under the 45° line, which implies that when the prediction accuracy of CHEM and boosting is inferior to that of bagging, boosting loses more accuracy than CHEM does. That is, the performance of CHEM is more stable than boosting.

6.3. Robustness to Output Noise

Another important advantage of CHEM over boosting is that CHEM is much more robust to output noise than boosting. To see this, the class labels of a random 10%

Table 2. Generalization errors.

Dataset ID	CHEM	Boosting	Bagging
1	0.0333	0.0310	0.0315
2	0.2411	0.2764	0.2358
3	0.2299	0.2650	0.2361
4	0.2238	0.2104	0.2386
5	0.0572	0.0649	0.0469
6	0.0671	0.0881	0.0667
7	0.0665	0.0662	0.0782
8	0.0036	0.0050	0.0123
9	0.0485	0.0290	0.0975
10	0.0880	0.0885	0.1075
11	0.1515	0.1195	0.1798
12	0.2420	0.2194	0.2540
13	0.4372	0.4805	0.5562
14	0.1664	0.1592	0.1866

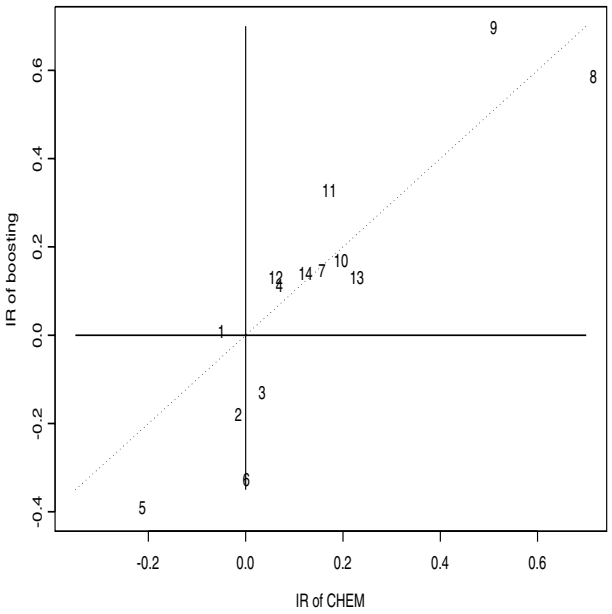

Fig. 7. Improvement rates (IR) of CHEM and boosting over bagging. The numbers plotted are the ID numbers of the datasets in Table 1.

of the training samples were changed at random and the three ensemble methods compared. Table 3 presents the generalization errors and increases in error rates due to noise (%). In most cases, increases in error rates of boosting due to noise are much larger than those of CHEM while bagging is least affected by noise. Many researchers have noted the vulnerability of boosting to output noise, including Breiman (2001) and Rätsch et al. (2001). They explain that the main source of this vulnerability to output noise is the way in which boosting updates the weights. Boosting keeps increasing the weights on most frequently misclassified observations, and instances having incorrect class labels tend to persist in being misclassified. Hence

Table 3. Generalization errors with 10% output noise (increases in error rates).

Dataset ID	CHEM	Boosting	Bagging
1	0.0438 (31.53)	0.0945 (204.83)	0.0373 (18.41)
2	0.2570 (6.59)	0.3135 (13.42)	0.2470 (4.74)
3	0.2452 (6.65)	0.3098 (16.90)	0.2479 (4.99)
4	0.2566 (14.65)	0.2542 (20.81)	0.2586 (8.38)
5	0.0615 (7.51)	0.1236 (90.44)	0.0458 (−2.34)
6	0.0776 (15.64)	0.0900 (2.15)	0.0790 (18.44)
7	0.0808 (21.50)	0.1034 (56.19)	0.0777 (−0.63)
8	0.0313 (769.44)	0.0808 (1516.00)	0.0084 (−31.70)
9	0.0830 (71.13)	0.1057 (264.48)	0.0947 (−2.87)
10	0.0960 (9.09)	0.1065 (20.33)	0.1110 (3.25)
11	0.1820 (20.13)	0.1650 (38.07)	0.1851 (2.94)
12	0.2432 (0.49)	0.2338 (6.56)	0.2598 (2.28)
13	0.4956 (13.35)	0.5303 (10.36)	0.5346 (−3.88)
14	0.1598 (−3.96)	0.1650 (3.64)	0.1956 (4.82)

boosting concentrates the weights mistakenly on these noisy instances. In contrast, the weights of CHEM ($|r_i|$ in Algorithm 2) are not dominated by a few larger ones. This is partly because the weights are adjusted by using the normalized ensemble model (i.e., $\sum_{i=1}^{m} w_i f_i(\mathbf{x}) / \sum_{i=1}^{m} w_i$ in (f) of Algorithm 2). This is explained further in Sect. 7.

6.4. Overfitting

In the late 1990s, it was found that one interesting property of boosting was that it seldom overfits the data, no matter how many base learners are combined. In particular, the generalization error keeps decreasing even after the training error reaches 0. Schapire et al. (1998) explained this phenomenon using the margin. They provided the upper bound of the generalization error, which is proportional to the margin. Then they showed that boosting keeps increasing the margin as more base learners are combined even after the training error becomes 0.

In recent studies (Quinlan 1996; Ridgeway 2000; Rätsch et al. 2001; Jiang 2002), however, much empirical evidence that contradicts the resistance of boosting to ovefitting is reported. In particular, Ridgeway (2000) provides a simple synthetic example that demonstrates that it is possible for boosting to result in serious overfitting when the decision trees used as base learners are too large. Rätsch et al. (2001) argue that overfitting in boosting arises when outliers exist, and they propose various regularized boosting algorithms resistant to outliers and hence to overfitting.

In contrast to boosting, CHEM never overfits the data. To see this, we first repeated the experiment of Ridgeway (2000). Generate $n = 1000$ observations as $x \sim N(0, 1)$, $F(x) = -x^2/2 + 1$, and $y|x \sim \text{Bernoulli}(p(x))$, where

$$p(x) = \frac{\exp(2F(x))}{1 + \exp(2F(x))}.$$

The ensemble models of CHEM and boosting are constructed in this simulated dataset, and the generalization errors are calculated with an additional 10,000 generated samples. The results are given in Fig. 8. Boosting results in very serious overfitting. The generalization error keeps growing as more base learners are added. Conversely, the generalization error of CHEM does not appear unusual.

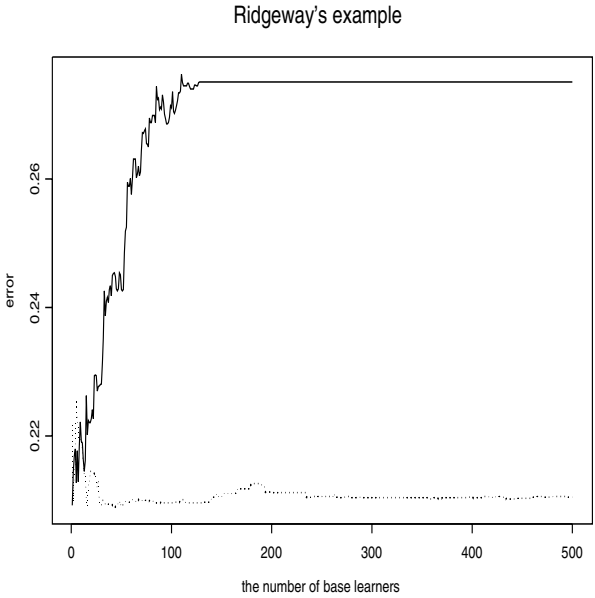

Fig. 8. Generalization errors for Ridgeway's example. The *bold line* is for boosting and the *dotted line* is for CHEM.

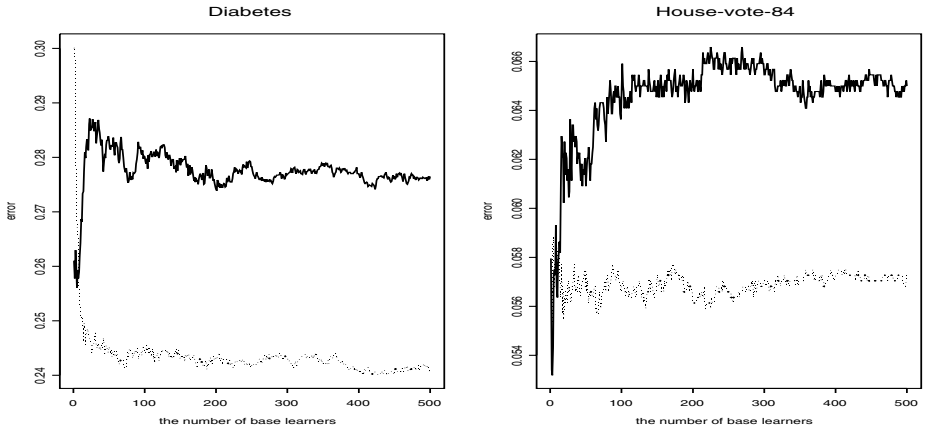

Fig. 9. Examples of overfitting of boosting: the *bold line* is for boosting and the *dotted line* is for CHEM.

In the empirical studies, overfitting of boosting is observed in the two datasets "Pima-Indian-Diabetes" and "House-vote-84", which are presented in Fig. 9. Note that overfitting never happens in CHEM.

6.5. Regression Problems

This section presents the empirical results for regression problems. Two datasets from the UC-Irvine machine learning archive and three synthetic datasets were analyzed.

Table 4. Summary of datasets used for regression problems. CV = cross validation.

Dataset	Training	Test	Inputs
Boston Housing	506	CV	12
Servo	330	CV	8
Friedman 1	200	2000	10
Friedman 2	200	2000	4
Friedman 3	200	2000	4

Table 5. Mean-squared test set errors for the regression problems.

Dataset	CHEM	Gradient boosting	Bagging
Boston housing	6.1553	5.6529	7.5654
Servo	0.1801	0.1978	0.3788
Friedman 1	5.9930	4.3621	6.7511
Friedman 2	549.5285	546.3978	467.0340
Friedman 3	0.0414	0.0401	0.0411

Detailed descriptions of the three synthetic datasets can be found in Friedman (1991). Table 4 summarizes the characteristics of the datasets.

Three ensemble methods are compared: CHEM, gradient boosting (Friedman 2001), and bagging. For gradient boosting, the squared error loss is used and the regularization procedure through shrinkage with the shrinkage parameter 0.1 is applied. As in the classification problems, decision trees with eight terminal nodes are used as base learners in gradient boosting and unpruned tress are used in CHEM and bagging. Also, the generalization errors (test sample mean squared errors) are calculated using the averages of ten repetitions of tenfold cross validation errors when test samples are not available.

Table 5 presents the generalization errors. The performance of the three ensemble methods for the regression problems is data dependent. For the two real datasets and Friedman 1, both CHEM and gradient boosting improve on bagging while bagging beats the other two ensemble methods significantly for Friedman 2. Note that the regularization procedure through shrinkage is used in gradient boosting. The performance of CHEM may be improved further by similar regularization.

7. Discussion

In this paper, we proposed a new ensemble algorithm called CHEM. CHEM has several advantages over bagging and boosting. It has a prediction accuracy similar to that of boosting and at the same time is as robust to output noise as bagging. Moreover, it never overfits and has lower generalization errors for regression problems, too.

In CHEM, unpruned decision trees are used as base learners. The performance of CHEM with smaller trees tends to deteriorate. This phenomenon can be partially explained as follows. CHEM constructs a sequence of ensemble models that converges to the true model inside the convex hull of the set of base learners. Hence the size of the convex hull of the set of base learners is an important ingredient in the success of CHEM, and the convex hull of the set of unpruned decision trees is the largest.

A comparison of the algorithms of CHEM and boosting gives interesting insights. The boosting algorithm (Real Adaboost, Schapire and Singer 1999) is given in Al-

gorithm 4. Boosting is similar to CHEM in the sense that it constructs a base learner based on residuals. In fact,

$$\begin{aligned} w_i &= w_i \exp(-y_i f_m(\mathbf{x}_i)) \\ &= \exp\left(-y_i \sum_{k=1}^{m} f_k(\mathbf{x}_i)\right) \\ &= \left|\frac{y_i^* - P_m(\mathbf{x}_i)}{\sqrt{P_m(\mathbf{x}_i)(1 - P_m(\mathbf{x}_i))}}\right|, \end{aligned}$$

where $P_m(\mathbf{x}) = \exp(H_m(\mathbf{x}))/(\exp(-H_m(\mathbf{x})) + \exp(H_m(\mathbf{x}))$ and $H_m(\mathbf{x}) = \sum_{k=1}^{m} f_k(\mathbf{x})$. However, there is a fundamental difference between CHEM and boosting. CHEM uses the normalized ensemble model (i.e., $H_m(\mathbf{x}) = \sum_{k=1}^{m} w_k f_k(\mathbf{x}) / \sum_{k=1}^{m} w_k$) to obtain the residuals, while boosting uses the unnormalized ensemble model (i.e., $H_m(\mathbf{x}) = \sum_{k=1}^{m} f_k(\mathbf{x})$). This seemingly minor difference results in qualitative differences in their performance. First, the ensemble model H_m in CHEM is a good estimator of the probability, while that in boosting is not. Figure 10 compares the deviances of CHEM and boosting based on the negative binomial log-likelihood loss for two arbitrarily chosen datasets. The results with the other datasets are similar. The results presented in Fig. 10 show that the ensemble models in CHEM becomes stabilized as more base learners are combined while the values of the ensemble models in boosting keep growing their values and consequently give more and more masses on extreme data points as more base learners are combined. This partially explains why CHEM is resistant to output noise and boosting is not.

Algorithm 4. Real boosting.

1. Start with weights $w_i = 1/n, \quad i = 1, \ldots, n$.
2. Repeat for $m = 1, \ldots, M$
   (a) Fit the classifier to obtain a class probability estimate $p_m(\mathbf{x}) = \hat{P}_w(y = 1|\mathbf{x})$ using weights w_i on the training data.
   (b) Set $f_m(\mathbf{x}) = \frac{1}{2} \log p_m(\mathbf{x})/(1 - p_m(\mathbf{x}))$.
   (c) Update $w_i = w_i \exp(-y_i f_m(\mathbf{x}_i)), \quad i = 1, \ldots, n$ and renormalize so that $\sum_i w_i = 1$.
3. Output the classifier $\text{sign}(\sum_{m=1}^{M} f_m(\mathbf{x}))$.

The role of base learners in CHEM and boosting differs. To explain the role of base learners in boosting, Friedman et al. (2000) proposed that the complexity of base learners determines the level of the dominant interaction in the final ensemble model. They argued that boosting is an additive model, and only the final ensemble model has the meaning of approximating the decision boundary and indicates that the complexity of base learners controls the level of dominant interactions. Hence if the base learners are too complicated, overfitting may result. The role of base learners in CHEM is different. In CHEM, each base learner is the best model (minimizing the deviance) for a given direction (based on residuals). This means that base learners in CHEM are not merely weak learners as in boosting but are strong learners from various directions. Hence each base learner has useful information about the data from a different angle, and we can use this information to understand the final decision. In this paper, we used unpruned decision trees. A better approach might be to let the tree size of the base learners vary.

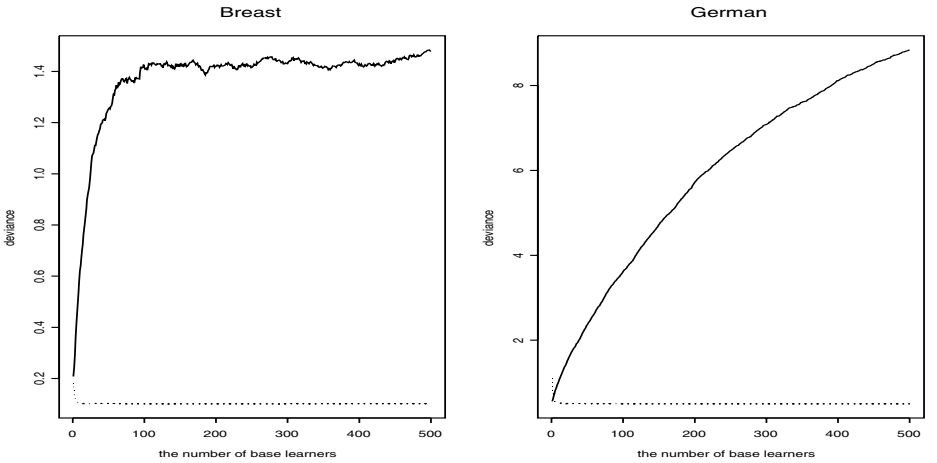

Fig. 10. Deviances in boostring and CHEM: the *bold line* is for boosting and the *dotted line* is for CHEM.

Acknowledgements. We thank the anonymous reviewers for their very useful comments and suggestions. This research is supported in part by U.S. Air Force Research Grant F62562-02-P-0547 and in part by KOSEF through the Statistical Research Center for Complex Systems at Seoul National University.

References

Bauer E, Kohavi R (1999) An empirical comparison of voting classification algorithms: bagging, boosting and variants. Mach Learn 36:105–139

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

Breiman L (1998) Arcing classifiers. Mach Learn 26:801–846

Breiman L (2001) Random forests. Mach Learn 45:5–32

Bühlmann P, Yu B (2000) Contribution to the discussion of paper by Friedman, Hastie and Tibshirani. Ann Stat 28:377–386

Dietterich TG (2000) An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting and randomization. Mach Learn 40:139–157

Friedman JH (1991) Multivariate adaptive regression splines (with discussion). Ann Stat 19:1–141

Friedman JH, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting. Ann Stat 38:337–374

Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Stat 29:1189–1232

Freund Y, Schapire R (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Sys Sci 55:119–139

Jiang W (2002) On weak base hypotheses and their implications for boosting regression and classification. Ann Stat 30:51–73

Mason L, Baxter J, Bartlett PL, Frean M (2000) Functional gradient techniques for combining hypotheses. In: Smola AJ, Bartlett P, Schölkopf B, Shuurmans C (eds) Advances in large margin classifiers. MIT Press, Cambridge, MA

Opitz D, Maclin R (1999) Popular ensemble methods: an empirical study. J Artif Intell Res 11:169–198

Quinlan J (1996) Boosting first-order learning. In: Arikawa S, Sharma (eds) Proceedings of the 7th international workshop on algorithmic learning theory. Lecture notes in artificial intelligence, vol 1160. Springer, Berlin Heidelberg New York, pp 143–155

Rätsch G, Onoda T, Müller KR (2001) Soft margins for AdaBoost. Mach Learn 42:287–320

Ridgeway G (2000) Contribution to the discussion of paper by Friedman, Hastie and Tibshirani. Ann Stat 28:393–400

Schapire R, Freund Y, Bartlett P, Lee W (1998) Boosting the margin: a new explanation for the effectiveness of voting methods. Ann Stat 26:1651–1686

Schapire R, Singer Y (1999) Improved boosting algorithms using confidence-rated predictions. Mach Learn 37:297–336

Author Biographies

Yongdai Kim received his B.A. from Seoul National University, Seoul, Korea, in 1991 and his M.S. from Seoul National University, Seoul, Korea. He earned his Ph.D. in statistics at The Ohio State University in Columbus, Ohio. Currently, he is an assistant professor at Ewha Womans University, Seoul, Korea. His major research interests are data mining, machine learning, and statistical learning.

Jinseog Kim is currently a postdoc in the Department of Statistics at Seoul National University, Seoul, Korea. He got his Ph.D. in statistics at Seoul National University in 2002. He worked at System Business Co. as the director of the Data Mining Research Lab. His main research interests are machine learning, data mining, and statistical learning.

Correspondence and offprint requests to: Yongdai Kim, Department of Statistics, Ewha Womans University, Seoul, 120-750, South Korea. Email: ydkim@mm.ewha.ac.kr