九州大学学術情報リポジトリ
Kyushu University Institutional Repository

Smooth Boosting Using an Information-Based Criterion

Hatano, Kohei
Department of Informatics, Kyushu University

https://hdl.handle.net/2324/1523957

出版情報：Algorithmic Learning Theory (Lecture Notes in Computer Science). 4264, pp.304-318, 2006-11-13. Springer-Verlag Berlin Heidelberg
バージョン：
権利関係：

Smooth Boosting Using an Information-Based Criterion

Kohei Hatano

Department of Informatics, Kyushu University
hatano@i.kyushu-u.ac.jp

Abstract. Smooth boosting algorithms are variants of boosting methods which handle only smooth distributions on the data. They are proved to be noise-tolerant and can be used in the "boosting by filtering" scheme, which is suitable for learning over huge data. However, current smooth boosting algorithms have rooms for improvements: Among non-smooth boosting algorithms, real AdaBoost or InfoBoost, can perform more efficiently than typical boosting algorithms by using an information-based criterion for choosing hypotheses. In this paper, we propose a new smooth boosting algorithm with another information-based criterion based on Gini index. we show that it inherits the advantages of two approaches, smooth boosting and information-based approaches.

1 Introduction

In recent years, huge data have become available due to the development of computers and the Internet. As size of such huge data can reach hundreds of gigabytes in knowledge discovery and machine learning tasks, it is important to make knowledge discovery or machine learning algorithms scalable. Sampling is one of effective techniques to deal with large data. There are many results on sampling techniques [23, 5] and applications to data mining tasks such as decision tree learning [7], support vector machine [2], and boosting [5, 6].

Especially, boosting is simple and efficient learning method among machine learning algorithms. The basic idea of boosting is to learn many slightly accurate hypotheses (or *weak hypotheses*) with respect to different distributions over the data, and to combine them into a highly accurate one. Originally, boosting was invented under the *boosting by filtering* framework [21, 10] (or the filtering framework), where the booster can sample examples randomly from the whole instance space. On the other hand, in the *boosting by subsampling* framework [21, 10] (or, the subsampling framework), the booster is given a bunch of examples in advance. Of course, the subsampling framework is more suitable when the size of data is relatively small. But, for large data, there are two advantages of the filtering framework over the subsampling framework. First, the space complexity is reduced as the booster "filters" examples and accepts only necessary ones (See, e.g., [10]). The second advantage is that the booster can automatically determine the sufficient sample size. Note that it is not trivial to determine the sufficient sample size a priori in the subsampling framework. So the boosting by filtering framework seems to fit learning over huge data. However, early boosting algorithms [21, 10] which work in the filtering framework were not practical, because they were not "adaptive", i.e., they need the prior knowledge on the accuracy of weak hypotheses.

MadaBoost, a modification of AdaBoost [11], is the first adaptive boosting algorithm which works in the filtering framework [6]. Combining with adaptive sampling methods [5], MadaBoost is shown to be more efficient than AdaBoost over huge data, while keeping the prediction accuracy. By its nature of updating scheme, MadaBoost is categorized as one of "smooth" boosting algorithms [12, 25, 14], where the name, smooth boosting, comes from the fact that these boosting algorithms only deal with smooth distributions over data (In contrast, for example, AdaBoost might construct exponentially skew distributions over data). Smoothness of distributions enables boosting algorithms to sample data efficiently. Also, smooth boosting algorithms have theoretical guarantees for noise tolerance in the various noisy learning settings, such as statistical query model [6] malicious noise model [25] and agnostic boosting [14].

However, there seems still room for improvements on smooth boosting. A non-smooth boosting algorithm, InfoBoost [1] (which is a special form of real AdaBoost [22]), performs more efficiently than other boosting algorithms in the boosting by subsampling framework. More precisely, given hypotheses with error $1/2 - \gamma/2$, typical boosting algorithms take $O((1/\gamma^2)\log(1/\varepsilon))$ iterations to learn a $(1-\varepsilon)$-accurate hypothesis. On the other hand, InfoBoost learns in from $O((1/\gamma)\log(1/\varepsilon))$ to $O((1/\gamma^2)\log(1/\varepsilon))$ iterations by taking advantage of the situation when weak hypotheses have low false positive error [15, 16]. So InfoBoost can be more efficient at most by $O(1/\gamma)$ times.

The main difference between InfoBoost and other boosting algorithms such as AdaBoost or MadaBoost is the criterion for choosing weak hypotheses. Typical boosting algorithms are designed to choose hypotheses whose errors are minimum with respect to given distributions. In contrast, InfoBoost uses an information-based criterion to choose weak hypotheses. The criterion was previously proposed by Kearns and Mansour in the context of decision tree learning [18], and also applied to boosting algorithms using branching programs [19, 26]. But, so far, no smooth algorithm has such the nice property of InfoBoost.

In this paper, we propose a new smooth boosting algorithm, *GiniBoost*, which uses another information-based criterion based on *Gini index* [3]. GiniBoost learns in $O(1/\varepsilon\varDelta)$ iterations, where we call $\varDelta$ the "pseudo gain" of weak hypotheses (that will be defined later). As $\varDelta$ varies from γ^2 to γ, our bound on iterations is potentially smaller than the $O(1/\varepsilon\gamma^2)$ bound which are achieved by previous smooth boosting algorithms [6, 25]. Unfortunately though, we have not given such a refined analysis as done for InfoBoost yet. Then, we propose an adaptive sampling procedure to estimate pseudo gains and apply GiniBoost in the filtering framework. Preliminary experiments show that GiniBoost improves MadaBoost in the filtering framework over large data.

2 Preliminaries

2.1 Learning Model

We adapt the PAC learning model [27]. Let $\mathcal{X}$ be an *instance space* and let $\mathcal{Y} = \{-1, +1\}$ be a set of labels. We assume an unknown *target function* $f : \mathcal{X} \to \mathcal{Y}$. Further we assume that f is contained in a known class $\mathcal{F}$ of functions from $\mathcal{X}$ to $\mathcal{Y}$. Let D be an unknown distribution over $\mathcal{X}$. The learner has an access to the *example oracle* $\mathrm{EX}(f, D)$. When given a call from the learner, $\mathrm{EX}(f, D)$ returns an *example* $(\boldsymbol{x}, f(\boldsymbol{x}))$ where each instance $\boldsymbol{x}$ is drawn randomly according to D. Let $\mathcal{H}$ be a hypothesis space, or a set of functions from $\mathcal{X}$ to $\mathcal{Y}$. We assume that $\mathcal{H} \supset \mathcal{F}$. For any distribution D over $\mathcal{X}$, *error* of hypothesis $h \in \mathcal{H}$ is defined as $\mathrm{err}_D(h) \overset{\text{def}}{=} \Pr_D\{h(\boldsymbol{x}) \neq f(\boldsymbol{x})\}$. Let S be a *sample*, a set of examples $((\boldsymbol{x}_1, f(\boldsymbol{x}_1), \ldots, (\boldsymbol{x}_m, f(\boldsymbol{x}_m)))$. For any sample S, *training error* of hypothesis $h \in \mathcal{H}$ is defined as $\widehat{\mathrm{err}}_S(h) \overset{\text{def}}{=} |\{(\boldsymbol{x}_i, f(\boldsymbol{x}_i) \in S \mid h(\boldsymbol{x}_i) \neq f(\boldsymbol{x}_i)\}|/|S|$.

We say that learning algorithm A is a *strong learner* for $\mathcal{F}$ if and only if, for any $f \in \mathcal{F}$ and any distribution D, given ε, δ $(0 < \varepsilon, \delta < 1)$, a hypothesis space $\mathcal{H}$, and access to the example oracle $\mathrm{EX}(f, D)$ as inputs, A outputs a hypothesis $h \in \mathcal{H}$ such that $\mathrm{err}_D(h) = \Pr_D\{h(x) \neq f(x)\} \leq \varepsilon$ with probability at least $1 - \delta$. We also consider a weaker learner. Specifically, we say that learning algorithm A is a *weak leaner* [1] for $\mathcal{F}$ if and only if, for any $f \in \mathcal{F}$, given a hypothesis space $\mathcal{H}$, and access to the example oracle $\mathrm{EX}(f, D)$ as inputs, A outputs a hypothesis $h \in \mathcal{H}$ such that $\mathrm{err}_D(h) \leq 1/2 - \gamma/2$ for a fixed γ $(0 < \gamma < 1)$. Note that $\mathrm{err}_D(h) = 1/2 - \gamma/2$ if and only if $r = \sum_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})h(\boldsymbol{x})D(\boldsymbol{x})$.

2.2 Boosting Approach

Schapire proved that the strong and weak learnability are equivalent to each other for the first time [21]. Especially the technique to construct a strong learner by using a weak learner is called "boosting". Basic idea of boosting is the following: First, the booster trains a weak learner with respect to different distributions $D_1, \ldots, D_T$ over the domain $\mathcal{X}$, and gets different "weak" hypotheses $h_1, \ldots, h_T$ such that $\mathrm{err}_{D_t}(h_t) \leq 1/2 - \gamma_t/2$ for each $t = 1, \ldots, T$. Then the booster combines weak hypotheses $h_1, \ldots, h_T$ into a final hypotheses h_{final} satisfying $\mathrm{err}_D(h_{final}) \leq \varepsilon$.

In the subsampling framework, the booster calls $EX(f, D)$ for a number of times and obtains a sample $S = ((\boldsymbol{x}_1, f(\boldsymbol{x}_1), \ldots, (\boldsymbol{x}_m, f(\boldsymbol{x}_m)))$ in advance. Then the booster constructs the final hypothesis h_{final} with its training error $\widehat{\mathrm{err}}_S(h_{final}) \leq \varepsilon$ by training the weak learner over the given sample S. The error $\mathrm{err}_D(h_{final})$ can be estimated by using arguments on VC-dimension or margin (E.g., see [11] or [20], respectively). For example, for typical boosting algorithms,

[1] In the original definition of [21], the weak learning algorithm is allowed to output a hypothesis h with $\mathrm{err}_D(h) > 1/2 - \gamma/2$ with probability at most δ as well. But in our definition we omit δ to make our discussion simple. Of course, we can use the original definition, while our analysis becomes slightly more complicated.

$\mathrm{err}_D(h_{final}) \le \widehat{\mathrm{err}}_S(h_{final}) + \tilde{O}(\sqrt{T \log |\mathcal{W}|/m})$ [2] with high probability, where T is the size of the final hypotheses, i.e., the number of weak hypotheses combined in h_{final}. So, assuming that $|\mathcal{W}|$ is finite, the sample and space complexity are $\tilde{O}(1/\gamma^2\varepsilon^2)$, respectively.

In the filtering framework, on the other hand, the booster deal with the whole instance space $\mathcal{X}$ through $\mathrm{EX}(f, D)$. By using statistics obtained from calls to $\mathrm{EX}(f, D)$, the booster tries to minimize $\mathrm{err}_D(h_{final})$ directly. Then, it can be shown that the sample complexity is $\tilde{O}(1/\gamma^4\varepsilon^2)$, but the space complexity is $\tilde{O}(1/\gamma^2)$ (in which the factor $\log(1/\varepsilon)$ is hidden) by using e.g., [6] and [5].

Smooth boosting algorithms generates only such distributions $D_1, \ldots, D_t$ that are "smooth" with respect to the original distribution D. We define the following measure of smoothness.

Definition 1 Let D and D' be any distributions over $\mathcal{X}$. We say that D' is λ-smooth with respect to D if $\max_{\boldsymbol{x} \in \mathcal{X}} D'(\boldsymbol{x})/D(\boldsymbol{x}) \le \lambda$.

The smoothness parameter λ has crucial roles in robustness of boosting algorithms [6, 25, 14]. Also, it affects the efficiency of sampling methods.

2.3 Our Assumption and Technical Goal

In the rest of the paper, we assume that the learner is given a finite set $\mathcal{W}$ of hypotheses such that for any distribution D' over $\mathcal{X}$, there exists a hypothesis $h \in \mathcal{W}$ satisfying $\mathrm{err}_{D'}(h) \le 1/2 - \gamma/2$. Now our technical goal is to construct an efficient smooth boosting algorithm which works in both the subsampling and the filtering framework.

3 Boosting by Subsampling

In this section, we propose our boosting algorithm in the subsampling framework.

3.1 Derivation

First of all, we derive our algorithm. It is well known that many of boosting algorithms can be viewed as greedy minimizers of loss functions [13]. More precisely, it can be viewed that they minimize particular loss functions that bound the training errors. The derivation of our algorithm is also explained simply in terms of its loss function.

Suppose that the learner is given a sample $S = \{(\boldsymbol{x}_1, f(\boldsymbol{x}_1), \ldots, (\boldsymbol{x}_m, f(\boldsymbol{x}_m))\}$, a set $\mathcal{W}$ of hypotheses, and the current final hypothesis $H_t(\boldsymbol{x}) = \sum_{j=1}^{t} \alpha_j h_j(\boldsymbol{x})$, where each $h_j \in \mathcal{W}$ and $\alpha_j \in \mathbb{R}$ for $j = 1, \ldots, t$. The training error of $H_t(\boldsymbol{x})$ over S is defined by $\widehat{\mathrm{err}}(\mathrm{sign}(H_t)) = \frac{1}{m}\sum_{i=1}^{m} I(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))$, where $I(a) = 1$ if $a > 1$ and $I(a) = 0$, otherwise. We assume a function $L : \mathbb{R} \to [0, +\infty)$

[2] In the $\tilde{O}(g(n))$ notation, we neglect $poly(\log(n))$ terms.

such that $I(a) \leq L(a)$ for any $a \in \mathbb{R}$. Then, by definition, $\widehat{\mathrm{err}}(\mathrm{sign}(H_t)) \leq \frac{1}{m}\sum_{i=1}^{m} L(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))$. If the function L is convex, the upperbound of the training error have a global minimum. Given a new hypothesis $h \in \mathcal{W}$, a typical boosting algorithm assigns α to h that minimizes a particular loss function. For example, AdaBoost solves the following minimization problem:

$$\min_{\alpha \in \mathbb{R}} \frac{1}{m} \sum_{i=1}^{m} L_{exp}(-f(\boldsymbol{x}_i)\{H_t(\boldsymbol{x}_i) + \alpha h(\boldsymbol{x}_i)\}),$$

where its loss function is given by *exponential loss*, $L_{exp}(x) = e^x$. The solution is given analytically as $\alpha = \frac{1}{2}\ln\frac{1+\gamma}{1-\gamma}$, where $\gamma = \sum_{i=1}^{m} f(\boldsymbol{x}_i)h(\boldsymbol{x}_i)D_t(\boldsymbol{x}_i)$, and $D_t(\boldsymbol{x}_i) = \frac{\exp(-f(\boldsymbol{x}_i)H(\boldsymbol{x}_i))}{\sum_{i=1}^{m}\exp(-f(\boldsymbol{x}_i)H(\boldsymbol{x}_i))}$. InfoBoost is designed to minimize the same loss function L_{exp} as AdaBoost, but it uses a slightly different form of the final hypothesis $H_t(\boldsymbol{x}) = \sum_{j=1}^{r} \alpha_j(h_j(\boldsymbol{x}))h_j(\boldsymbol{x})$, where $\alpha_j(z) = \alpha_j[+1]$ if $z \geq 0$, $\alpha_j(z) = \alpha_j[+1]$, otherwise $(\alpha_j[\pm 1] \in \mathbb{R})$. The main difference is that InfoBoost assigns coefficients for each prediction $+1$ and -1 of a hypothesis. Then, the minimization problem of InfoBoost is given as:

$$\min_{\alpha[+1], \alpha[-1] \in \mathbb{R}} \frac{1}{m} \sum_{i=1}^{m} L_{exp}(-f(\boldsymbol{x})\{H_t(\boldsymbol{x}) + \alpha(h(\boldsymbol{x}))h(\boldsymbol{x})\}).$$

This problem also has the analytical solution: $\alpha[\pm 1] = \frac{1}{2}\ln\frac{1+\gamma[\pm 1]}{1-\gamma[\pm 1]}$, $\gamma[\pm 1] = \frac{\sum_{i:h(\boldsymbol{x}_i)=\pm 1} f(\boldsymbol{x}_i)h(\boldsymbol{x}_i)D_t(\boldsymbol{x}_i)}{\sum_{i:h(\boldsymbol{x}_i)=\pm 1} D(\boldsymbol{x}_i)}$, and $D_t(\boldsymbol{x}_i) = \frac{\exp(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))}{\sum_{i=1}^{m}\exp(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))}$. Curiously, this derivation makes InfoBoost choose a hypothesis that maximizes information gain, where the entropy function is defined not by Shannon's entropy function $E_{Shannon}(p) = -p\log p - (1-p)\log(1-p)$, but by the entropy function $E_{KM}(p) = 2\sqrt{p(1-p)}$ proposed by Kearns and Mansour [18] (See [26] for details). MadaBoost is formulated as the same minimization problem of AdaBoost, except that its loss function is replaced with $L_{mada}(x) = e^x$, if $x \leq 0$, $L_{mada}(x) = x$, otherwise.

Now combining the derivations of InfoBoost and MadaBoost in a straightforward way, our boosting algorithm is given by

$$\min_{\alpha[+1], \alpha[-1] \in \mathbb{R}} \frac{1}{m} \sum_{i=1}^{m} L_{mada}(-f(\boldsymbol{x}_i)\{H_t(\boldsymbol{x}_i) + \alpha(h(\boldsymbol{x}_i))h(\boldsymbol{x}_i)\}). \tag{1}$$

Since the solution cannot be obtained analytically, we minimize an upperbound of (1). The way of our approximation is a modification of the technique used for AdaFlat [14]. By using Taylor expansion (see Lemma 3 in Appendix for a proof) we have $L_{mada}(x+a) \leq L_{mada}(a) + L'_{mada}(a)(x + x^2)$.

Let

$$\ell(x) = L'_{mada}(x) = \begin{cases} 1, & x \geq 0 \\ e^x, & x < 0. \end{cases}$$

Then we get

$$
\begin{aligned}
&\frac{1}{m}\sum_{i=1}^{m} L_{mada}(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i)) - \frac{1}{m}\sum_{i=1}^{m} L_{mada}(-f(\boldsymbol{x}_i)H_{t+1}(\boldsymbol{x}_i)) \\
&\geq \frac{1}{m}\sum_{i=1}^{m}\left\{ f(\boldsymbol{x}_i)h_t(\boldsymbol{x}_i)\alpha[h(\boldsymbol{x}_i)]\ell(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i)) - \alpha[h(\boldsymbol{x}_i)]^2\ell(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))\right\} \\
&\stackrel{\text{def}}{=} \Delta L_t(h).
\end{aligned}
$$

By solving the equations $\partial \Delta L_t(h)/\partial\alpha_t[b] = 0$ for $b = \pm 1$, we see that $\Delta L_t(h)$ is maximized if $\alpha_t[b] = \gamma_t[b](h)/2$, where

$$
\gamma_t[b](h) = \frac{\sum_{i:h(\boldsymbol{x}_i)=b} h(\boldsymbol{x}_i)f(\boldsymbol{x}_i)D_t(\boldsymbol{x}_i)}{\sum_{i:h(\boldsymbol{x}_i)=b} D_t(\boldsymbol{x}_i)}, \text{ and } D_t(\boldsymbol{x}_i) = \frac{\ell(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))}{\sum_{i=1}^{m}\ell(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))}.
$$

By substituting $\alpha_t[b] = \gamma_t[b](h)/2$ for $b = \pm 1$, we get

$$
\Delta L_t(h) = \frac{\mu_t}{4}\left\{p_t(h)\gamma_t[+1](h)^2 + (1-p_t(h))\gamma_t[-1](h)^2\right\} \qquad (2)
$$

where $\mu_t = \frac{\sum_{i=1}^{m}\ell(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))}{m}$, and $p_t(h) = \Pr_{D_t}\{h(\boldsymbol{x}_i) = +1\}$.

Our derivation implies a new criterion to choose a weak hypothesis. That is, we choose $h \in \mathcal{W}$ that maximizes

$$
\Delta_t(h) = p_t(h)\gamma_t[+1](h)^2 + (1-p_t(h))\gamma_t[-1](h)^2.
$$

We call the quantity *pseudo gain* of hypothesis h with respect to f and D_t. Now we motivate the pseudo gain in the following way. Let $\varepsilon_t[\pm1](h) = \Pr_{D_t}\{f(\boldsymbol{x}_i) = \mp 1 | h(\boldsymbol{x}_i) = \pm 1\}$. Note that $\gamma_t[\pm 1](h) = 1 - 2\varepsilon_t[\pm 1](h)$. Then

$$
\begin{aligned}
&1 - \Delta_t(h) \\
=&p_t(h)\{1-(1-2\varepsilon_t[+1](h))^2\} + (1-p_t)\{1-(1-2\varepsilon_t[-1](h))^2\} \\
=&p_t(h)\cdot 4\varepsilon_t[+1](h)(1-\varepsilon_t[+1](h)) + (1-p_t(h))\cdot 4\varepsilon_t[-1](h)(1-\varepsilon_t[-1](h)),
\end{aligned}
$$

which can be interpreted as the conditional entropy of f given h with respect to D_t, where the entropy is defined by *Gini index* $E_{Gini}(p) = 4p(1-p)$ [3] (See other entropy measures in Figure 1 for comparison). So, maximizing the pseudo gain is equivalent to maximizing the information gain defined with Gini index.

3.2 Our Algorithm

Based on our derivation we propose GiniBoost. The description of our modification is given in Figure 2. To make our notation simple, we denote $p_t(h_t) = p_t$, $\gamma_t[\pm1](h_t) = \gamma_t[\pm1]$, and $\Delta_t(h_t) = \Delta_t$.

First, we show that the smoothness of distributions D_t.

Proposition 1 During the execution of GiniBoost, each distribution D_t $(t \geq 1)$ is $1/\varepsilon$-smooth with respect to D_1, the uniform distribution over S.

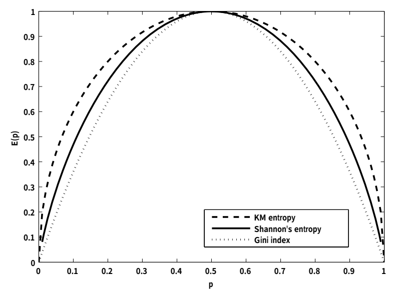

Fig. 1. Plots of three entropy functions, KM entropy (upper) $E_{KM}(p) = 2\sqrt{p(1-p)}$, Shannon's entropy (middle) $E_{Shannon}(p) = -p\log p - (1-p)\log(1-p)$, and Gini index (lower) $E_{Gini}(p) = 4p(1-p)$.

Proof. Note that, during the while-loops, $\mu_t \geq \mathrm{err}_S(h_{final}) > \varepsilon$. Therefore, for any i, $D_t(i)/D_1(i) = \ell(-f(\boldsymbol{x}_i)H_t(\boldsymbol{x}_i))/\mu_t < 1/\varepsilon$. $\square$

It is already shown that smoothness $1/\varepsilon$ is optimal, i.e., there is no boosting algorithm that achieves the smoothness less than $1/\varepsilon$ [25, 14].

Next, we prove the time complexity of GiniBoost.

Theorem 2 Suppose that, during the while-loops, $\mathrm{err}_{D_t}(h_t) \leq 1/2 - \gamma_t/2 \leq 1/2 - \gamma/2$ for some $\gamma > 0$. Then, GiniBoost outputs a final hypothesis h_{final} satisfying $\widehat{\mathrm{err}}_S(h_{final}) \leq \varepsilon$ within $T = O\left(1/\varepsilon\varDelta\right)$ iterations, where $\varDelta = \min_{t=1,\ldots,T} \varDelta_t$ and $\varDelta \geq \gamma^2$.

Proof. By our derivation of GiniBoost, for any $T \geq 1$, the training error $\widehat{\mathrm{err}}(H_T)$ is less than $1 - \sum_{t=1}^{T} \varDelta L_t(h_t)$. As in the proof of Proposition 1, $\mu_t \geq \varepsilon$. So we have $\varDelta L_t(h_t) \geq \varepsilon\varDelta/4$ and thus $\widehat{\mathrm{err}}_S(h_{final}) \leq \varepsilon$ if $T = 4/\varepsilon\varDelta$. Finally, by Jensen's inequality, $\varDelta_t \geq p_t\gamma_t[+1]^2 + (1-p_t)\gamma_t[-1]^2 \geq \gamma_t^2 \geq \gamma^2$, which proves $\varDelta \geq \gamma^2$. $\square$

Remark. We discuss the efficiency of other boosting algorithms and GiniBoost. GiniBoost runs in $O(1/\varepsilon\gamma^2)$ iterations in the worst case. But, since the pseudo gain $\varDelta$ ranges from γ^2 to γ, our bound $O(1/\varepsilon\varDelta)$ is potentially smaller. Smooth boosting algorithms MadaBoost [6] and SmoothBoost [25] run in $O(1/\varepsilon\gamma^2)$ iterations as well. However, the former needs a technical assumption in their analysis that $\gamma_t \geq \gamma_{t+1}$ for each iteration t. Also the latter is not adaptive, i.e., it needs the prior knowledge of $\gamma > 0$. On the other hand, GiniBoost is adaptive and does not need such the technical assumption. AdaFlat [14] is another smooth boosting algorithm which is adaptive, but it takes $O(1/\varepsilon^2\gamma^2)$ iterations. Finally, AdaBoost [11] achieves $O(\log(1/\varepsilon)/\gamma^2)$ bound and the bound is optimal [10]. But AdaBoost might construct exponentially skew distributions. It is shown that a combination of boosting algorithms ("boosting tandems approach" [9, 14]) can achieve $O(\log(1/\varepsilon)/\gamma^2)$ with smoothness $\tilde{O}(1/\varepsilon)$. Yet, it is still open whether a single adaptive boosting algorithm can learn in $O(\log(1/\varepsilon)/\gamma^2)$ iterations while keeping the optimal smoothness $1/\varepsilon$.

```
GiniBoost

  Given: $S = ((\boldsymbol{x}_1, f(\boldsymbol{x}_1)), ..., (\boldsymbol{x}_m, f(\boldsymbol{x}_m)))$, and $\varepsilon$ $(0 < \varepsilon < 1)$
  1. $D_1(i) \leftarrow 1/m$; $(i = 1, ..., m)$ $H_0(\boldsymbol{x}) \leftarrow 0$; $t \leftarrow 1$;
  2. while $\widehat{\mathrm{err}}_S(h_{final}) > \varepsilon$ do
     a) $h_t \leftarrow \arg\max_{h \in \mathcal{W}} \Delta_t(h)$;
     b) $\alpha_t[\pm 1] \leftarrow \gamma_t[\pm 1]/2$; Let $\alpha_t(z) = \alpha_t[+1]$ if $z > 0$, o.w. let $\alpha_t(z) = \alpha_t[-1]$;
     c) $H_{t+1}(\boldsymbol{x}) \leftarrow H_t(\boldsymbol{x}) + \alpha_t(h_t(\boldsymbol{x}))h_t(\boldsymbol{x})$;
     d) Define the next distribution $D_{t+1}$ as
            $D_{t+1}(i) = \frac{\ell(-f(\boldsymbol{x}_i)H_{t+1}(\boldsymbol{x}_i))}{\sum_{i=1}^m \ell(-f(\boldsymbol{x}_i)H_{t+1}(\boldsymbol{x}_i))}$;
     e) $t \leftarrow t + 1$;
     end-while
  3. Output the final hypothesis $h_{final}(\boldsymbol{x}) = \mathrm{sign}\,(H_{t+1}(\boldsymbol{x}))$.
```

Fig. 2. GiniBoost

4 Boosting by Filtering

In this section, we propose GiniBoost$_{\mathrm{filt}}$ in the filtering framework. Let

$$D_t(\boldsymbol{x}) = \frac{D(\boldsymbol{x})\ell(-f(\boldsymbol{x})H_t(\boldsymbol{x}))}{\sum_{\boldsymbol{x} \in \mathcal{X}} D(\boldsymbol{x})\ell(-f(\boldsymbol{x})H_t(\boldsymbol{x}))}.$$

We define $\mu_t = \sum_{\boldsymbol{x} \in \mathcal{X}} D(\boldsymbol{x})\ell(-f(\boldsymbol{x})H_t(\boldsymbol{x})),$. We denote $\hat{a}$ as the empirical estimate of the parameter a given a sample S_t. The description of GiniBoost$_{\mathrm{filt}}$ is given in Figure 3.

The following property of FiltEX can be immediately verified.

Proposition 3 Fix any iteration t, (i) FiltEX outputs $(\boldsymbol{x}, f(\boldsymbol{x}))$, where $\boldsymbol{x}$ is drawn according to D_t, and (ii) the probability that FiltEX outputs an example is at least $\mu_t \geq \mathrm{err}_D(\mathrm{sign}(H_t))$.

Then, we prove a multiplicative tail bound on the estimate $\hat{\Delta}_t(h)$ of the pseudo gain.

Lemma 1 Fix any $t \geq 1$. Let $\widehat{\Delta}_t(h) = \hat{p}_t(h)\hat{\gamma}_t[+1](h)^2 + (1 - \hat{p}_t(h))\hat{\gamma}_t[-1](h)^2$ be the empirical estimate of $\Delta_t(h)$ given S_t. Then it holds for any ε $(0 < \varepsilon < 1)$ that

$$\Pr_{D^m}\{\widehat{\Delta}_t(h) \geq (1 + \varepsilon)\Delta_t(h)\} \leq b_1 e^{-\frac{\varepsilon^2 \Delta_t m}{c_1}}, \tag{3}$$

and

$$\Pr_{D^m}\{\widehat{\Delta}_t(h) \leq (1 - \varepsilon)\Delta_t(h)\} \leq b_1 e^{-\frac{\varepsilon^2 \Delta_t m}{c_2}}, \tag{4}$$

where $b_1 \leq 8$, $c_1 \leq 600$, and $c_2 \leq 64$.

```
GiniBoost_filt(ε, δ, W)
1.Let $H_1(x) = 0$; $t \leftarrow 1$; $\delta_1 \leftarrow \delta/8$;
  $S'_1 \leftarrow \frac{18\log(1/\delta_1)}{\varepsilon}$ random examples drawn by EX$(f, D)$;
2. while $\widehat{\mathrm{err}}_{S'_t}(\mathrm{sign}(H_t)) \geq \frac{2\varepsilon}{3}$ do
  $(h_t, S_t) \leftarrow$ HSelect$(1/2, \delta_t)$;
  $(\hat{\gamma}_t[+1], \hat{\gamma}_t[-1]) \leftarrow$ empirical estimates over $S_t$;
  $\alpha_t[\pm 1] \leftarrow \hat{\gamma}_t[\pm 1]/2$;
  $H_{t+1}(\boldsymbol{x}) \leftarrow H_t(\boldsymbol{x}) + \alpha_t(h_t(\boldsymbol{x}))h_t(\boldsymbol{x})$;
  $t \leftarrow t+1$; $\delta_t \leftarrow \delta/(4t(t+1))$;
  $S'_t \leftarrow \frac{18\log(1/\delta_t)}{\varepsilon}$ random examples drawn by EX$(f, D)$;
  end-while
3. Output the final hypothesis $h_{final}(\boldsymbol{x}) = \mathrm{sign}\,(H_t(\boldsymbol{x}))$;

FiltEX()
do
  $(\boldsymbol{x}, f(\boldsymbol{x})) \leftarrow$ EX$(f, D)$;
  $r \leftarrow$ uniform random number over $[0, 1]$;
  if $r < \ell(-f(\boldsymbol{x})H_t(\boldsymbol{x}))$ then return $(\boldsymbol{x}, f(\boldsymbol{x}))$;
end-do

HSelect(ε, δ)
  $m \leftarrow 0$; $S \leftarrow \emptyset$; $i \leftarrow 1$; $\varDelta_g \leftarrow 1/2$; $\delta' \leftarrow \delta/(2|\mathcal{W}|)$;
  do
  $(\boldsymbol{x}, f(\boldsymbol{x})) \leftarrow$ FiltEX();
  $S \leftarrow S \cup (\boldsymbol{x}, f(\boldsymbol{x}))$; $m \leftarrow m+1$;
  if $m = \left\lceil \frac{c_1 \ln \frac{b_1}{\delta'}}{\varepsilon^2 \varDelta_g} \right\rceil$ then
      Let $\hat{\varDelta}_t(h)$ be the empirical estimate of $\varDelta_t(h)$ over $S$ for each $h \in \mathcal{W}$;
      if $\exists h \in \mathcal{W}$, $\hat{\varDelta}_t(h) \geq \varDelta_g$ then return $h$ and $S$;
      else $\varDelta_g \leftarrow \varDelta_g/2$; $i \leftarrow i+1$; $\delta \leftarrow \delta/(i(i+1)|\mathcal{W}|)$;
  end-if
  end-do
```

Fig. 3. GiniBoost$_{\mathrm{filt}}$

The proof of Lemma 1 is omitted and given in the technical report version of our paper [17]. Then, we analyze our adaptive sampling procedure HSelect. Let $\varDelta_t^* = \max_{h' \in \mathcal{W}} \varDelta_t(h')$. We prove the following lemma. The proof is also given in [17] .

Lemma 2 Fix any $t \geq 1$. Then, with probability at least $1-\delta$, (i) HSelect(ε, δ) outputs a hypothesis $h \in \mathcal{W}$ such that $\varDelta_t(h) > (1-\varepsilon)\varDelta_t^*$, and (ii) the number of calls of $EX(f, D)$ is

$$O\left(\frac{\log\frac{1}{\delta} + \log|\mathcal{W}| + \log\log\frac{1}{\varDelta_t^*}}{\varepsilon^2 \varDelta_t^*}\right).$$

Finally we obtain the following theorem.

Theorem 4 With probability at least $1-\delta$,

(i) GiniBoost$_{\text{filt}}$ outputs the final hypothesis h_{final} such that $\text{err}_D(h_{final}) \le \varepsilon$,
(ii) GiniBoost$_{\text{filt}}$ terminates in $T = O\left(1/\varepsilon\Delta\right)$ iterations,
(iii) the number of calls of EX(f, D) is

$$O\left(\frac{\log\frac{1}{\delta} + \log\frac{1}{\varepsilon\Delta} + \log|\mathcal{W}| + \log\log\frac{1}{\Delta}}{\varepsilon^2\Delta^2} \cdot \left(\log\frac{1}{\delta} + \log\frac{1}{\varepsilon\Delta}\right)\right), \text{ and}$$

(iv) the space complexity is

$$O\left(\frac{\log\frac{1}{\delta} + \log\frac{1}{\varepsilon\Delta} + \log|\mathcal{W}| + \log\log\frac{1}{\Delta}}{\Delta}\right),$$

where $\Delta_t \ge \Delta \ge \gamma^2$.

Proof. We say that GiniBoost fails at iteration t if one of the following event occurs: (a) HSelect fails, i.e., it does not meet the conditions (i) or (ii) in Lemma 2 , (b) FiltEX calls $EX(f, D)$ for more than $(6/\varepsilon)M_t\log(1/\delta_t)$ times at iteration t, where M_t is denoted as the number of calls for FiltEX, (c) $\text{err}_D(\text{sign}(H_t)) > \varepsilon$ and $\widehat{\text{err}}_{S'_t}(\text{sign}(H_t)) < 2\varepsilon/3$, or (d) $\text{err}_D(\text{sign}(H_t)) < \varepsilon/2$ and $\widehat{\text{err}}_{S'_t}(\text{sign}(H_t)) > 2\varepsilon/3$. Note that, by Proposition 3, Lemma 2 and an application of Chernoff bound, the probability of each event (a), ..., (d) is at most δ_t, respectively. So the probability that GiniBoost fails is at most $4\delta_t$ at each iteration t. Then, during T iterations, GiniBoost fails at some iteration is at most $\sum_{t=1}^{T} 4\delta_t = \delta - \delta/(T+1) < \delta$. Now suppose that GiniBoost does not fail during T iterations. Then, we have $\text{err}_D(h_{final}) \le 1 - \sum_{i=t}^{T}(1/8)\Delta_t^*$ by using the similar argument in the proof of Theorem 2, and thus GiniBoost $\text{err}_D(h_{final}) \le \varepsilon/2$ in $T = 16/(\varepsilon\Delta)$ iterations. Then, since GiniBoost does not fail during T iterations, $\widehat{\text{err}}_{S'_t}(\text{sign}(H_t)) < 2\varepsilon/3$ at iteration $T+1$ and GiniBoost outputs h_{final} with $\text{err}_D(h_{final}) \le \varepsilon/2$ and terminates. The total number of calls of $EX(f, D)$ in $T = O(1/\varepsilon\Delta)$ iterations is $O(T \cdot M_T(1/\varepsilon)\log(1/\delta_T))$ with probability $1-\delta$ and by combining with Lemma 2, we complete the proof. □

5 Improvement on Sampling

While Lemma 1 gives a theoretical guarantee without any assumption, the bound has the constant factor $c_1 = 600$, which is too large to apply the lemma in practice. In this section, we derive a practical tail bound on the pseudo gain by using the central limit theorem. We say that a sequence of random variables $\{X_i\}$ is *asymptotically normal* with mean μ_i and variance σ_i^2 (we write X_i is $AN(\mu_i, \sigma_i^2)$ for short) if $(X_i - \mu_i)/\sigma_i$ converges to $N(0, 1)$ in distribution [3]. The central limit

[3] Let $F_1(x), \ldots, F_m(x)$, and $F(x)$ be distribution functions. Let $X_1, \ldots, X_m$, and X be corresponding random variables, respectively. X_m converges to X in distribution if $\lim_{m\to\infty} F_m(x) = F(x)$.

theorem states that, for independent random variables $X_1, \ldots, X_m$ from the same distribution with mean μ and variance σ^2, $\sum_{i=1}^m X_i/m$ is $AN(\mu, \sigma^2/m)$. In particular, we use the multivariate version of the central limit theorem.

Theorem 5 ([24]) Let $\mathbf{X_1}, \ldots, \mathbf{X_m}$ be i.i.d. random vectors with mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{\Sigma}$. Then, $\sum_{i=1}^m \mathbf{X}_i/m$ is $AN(\boldsymbol{\mu}, \mathbf{\Sigma})$.

Fix any hypothesis $h \in \mathcal{W}$, and distribution D_t over $\mathcal{X}$. Let $X \in \{0,1\}$ and $Y \in \{-1,+1\}$ be random variables, induced by an independent random draw of $\boldsymbol{x} \in \mathcal{X}$ under D_t, such that $X = 1$ if $h(\boldsymbol{x}) = +1$, otherwise $X = 0$ and $Y = f(\boldsymbol{x})h_t(\boldsymbol{x})$, respectively. Then the pseudo gain $\Delta_t(h)$ can be written as $E(X) \cdot \{E(XY)/E(X)\}^2 + E(\bar{X}) \cdot \{E(\bar{X}Y)/E(\bar{X})\}^2$, where $\bar{X} = 1 - X$. Our empirical estimate of the pseudo gain is $Z = (\sum_{i=1}^m X_iY_i/m)^2/(\sum_{i=1}^m X_i/m) + (\sum_{i=1}^m \bar{X}_iY_i/m)^2/(\sum_{i=1}^m \bar{X}_i/m)$. The following theorem guarantees that a combination of sequences of asymptotically normal random variables is also asymptotically normal (Theorem 3.3A in [24]).

Theorem 6 ([24]) Suppose that $\mathbf{X} = (X^{(1)}, \ldots, X^{(k)})$ is $AN(\boldsymbol{\mu}, b\mathbf{\Sigma})$, with $\mathbf{\Sigma}$ a covariance matrix and $b \to 0$. Let $g(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_n(\mathbf{x}))$, $\mathbf{x} = (x_1, \ldots, x_k)$, be a vector-valued function for which each component function $g_i(\mathbf{x})$ is real-valued and has a nonzero differential at $\mathbf{x} = \boldsymbol{\mu}$. Then, $g(\mathbf{X})$ is $AN(g(\boldsymbol{\mu}), b^2\mathbf{D\Sigma D'})$, where

$$\mathbf{D} = \left[\frac{\partial g_i}{\partial x_j}\Big|_{\mathbf{x}=\boldsymbol{\mu}} \right]_{n \times k}.$$

By using Theorem 5 and 6 for $\mathbf{X}_m = (\sum_{i=1}^m X_i/m, \sum_{i=1}^m X_iY_i/m, \sum_{i=1}^m \bar{X}_iY_i/m)$ and $g(u,v,w) = v^2/u + w^2/(1-u)$, we get the following result.

Corollary 7 $Z = \frac{\left(\sum_{i=1}^m X_iY_i/m\right)^2}{\sum_{i=1}^m X_i/m} + \frac{\left(\sum_{i=1}^m \bar{X}_iY_i/m\right)^2}{\sum_{i=1}^m \bar{X}_i/m}$ is $AN(\mu_z, \sigma_z^2)$, where $\mu_z = \frac{E(XY)^2}{E(X)} + \frac{E(\bar{X}Y)^2}{E(\bar{X})}$, and $\sigma_z^2 \le 4\mu_z/m$.

The proof is given in [17]. When the given sample is large enough, we may be able to use the central limit theorem. Then

$$\Pr\left\{ \frac{Z - \mu_z}{\sigma_z} \le \varepsilon \right\} \approx \Phi(\varepsilon),$$

where $\Phi(x) = \int_{-\infty}^{x} (1/\sqrt{2\pi}) e^{-\frac{1}{2}y^2} dy$. Since $1 - \Phi(x) \le 1/(x\sqrt{2\pi}) e^{-\frac{1}{2}x^2}$ (see, e.g.,[8]),

$$\begin{aligned}\Pr\{Z - \mu_z > \varepsilon\mu_z\} = \Pr\left\{ \frac{Z - \mu_z}{\sigma_z} > \frac{\varepsilon\mu_z}{\sigma_z} \right\} &\lessapprox \frac{\sigma_z}{\varepsilon\mu_z\sqrt{2\pi}} e^{-\frac{\varepsilon^2\mu_z^2}{2\sigma_z^2}} \\ &< \frac{2}{\sqrt{2\pi\varepsilon^2\mu_z m}} e^{-\frac{\varepsilon^2\mu_z m}{8}}. \qquad (5)\end{aligned}$$

Substituting

$$m = \frac{8\left(\ln\frac{1}{\delta\sqrt{2\pi}} - \frac{1}{2}\ln\ln\frac{1}{\delta\sqrt{2\pi}}\right)}{\varepsilon^2\mu_z}$$

to inequality (5), we obtain $\Pr\{Z - \mu_z > \varepsilon\mu_z\} < \delta$. Note that the same argument holds for $\Pr\{Z \leq (1-\varepsilon)\mu_z\}$. Therefore, we can replace the estimate of sample size $m = \frac{c_1 \ln(b_1/\delta)}{\varepsilon^2 \Delta_g}$ in HSelect with $m = \frac{8\left(\ln\frac{1}{\delta\sqrt{2\pi}} - \frac{1}{2}\ln\ln\frac{1}{\delta\sqrt{2\pi}}\right)}{\varepsilon^2\Delta_g}$ and this modification makes HSelect more practical.

6 Experimental Results

In this section, we show our preliminary experimental results in the filtering framework. We apply GiniBoost and MadaBoost for text categorization tasks on a collection of Reuters news (Reuters-21578 [4]). We use the modified Apte split which contains about $10,000$ news documents labeled with topics. We choose five major topics and for each topics, we let boosting algorithms classify whether a news document belongs to the topic or not. As weak hypotheses, we prepare about $30,000$ decision stumps corresponding to words.

We evaluate algorithms using cross validation in a random fashion, as done in [4]. For each topic, we split the data randomly into a training data with probability 0.7 and a test data with probability 0.3. We prepare 10 pairs of such training and test data. We train algorithms over the training data until they sample $1,000,000$ examples in total, and then we evaluate them over the test data. The results are averaged over 10 trials and 5 topics. We conduct our experiments on a computer with a CPU Xeon 3.8GHz using 8 Gb of memory under Linux.

We consider two versions of GiniBoost in our experiments. The first version is the original one which we described in Section 3. The second version is a slight modification of the original one, in which we use $\alpha_t[\pm 1] = \gamma_t[\pm 1]$. We call this version GiniBoost2.

We run GiniBoost with HSelect(ε, δ), where parameter $\varepsilon = 0.75$ and $\delta = 0.1$ are fixed. Also, we run MadaBoost with geometric AdaSelect [5] whose parameters are $s = 2$, $\varepsilon = 0.5$ and $\delta = 0.1$. Note that, in this setting, we demand both HSelect and AdaSelect to output a weak hypothesis h_t with $\gamma_t^2 \geq (1/4)\max_{h' \in \mathcal{W}} \gamma_t(h')^2$. In the following experiments, we use the approximation based on the central limit theorem, described in Section 5.

The results are shown in Table 1 and Figure 4, As indicated, GiniBoost and GiniBoost2 improve the performance of MadaBoost. We also run AdaBoost (without sampling) for 100 iterations, where AdaBoost processes about 1,000,000 examples. Then, GiniBoost is about three times faster than AdaBoost, while improving the accuracy. The main reason why filtering-based algorithms save time would be that they use rejection sampling. By using rejection sampling,

[4] http://www.daviddlewis.com/resources/testcollections/reuters21578.

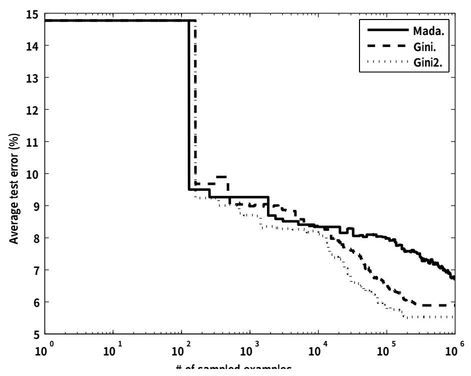

Fig. 4. Test errors (%) of boosting algorithms for Reuters-21578 data. The test errors are averaged over topics.

filtering-based algorithms keep only accepted examples in hand. Since the number of accepted example is much smaller than that of the whole given sample, we can find weak hypotheses faster over accepted examples than over the given sample.

In particular, GiniBoost uses fewer accepted examples than MadaBoost. mainly because they use different criteria. Roughly speaking, MadaBoost takes $\tilde{O}(1/\gamma_t^2)$ accepted examples in order to estimate γ_t. On the other hand, in order to estimate Δ_t, GiniBoost takes $\tilde{O}(1/\Delta_t)$ accepted examples, which is smaller than $\tilde{O}(1/\gamma_t^2)$. This consideration would explain why GiniBoost is faster than MadaBoost.

7 Summary and Future Work

In this paper, we propose a smooth boosting algorithm that uses an information-based criterion based on Gini index for choosing hypotheses. Our preliminary experiments show that our algorithm performs well in the filtering framework. As future work, we further investigate the connections between boosting and information-based criteria. Also, we will conduct experiments over much huge data in the filtering framework.

Acknowledgments

I would like to thank Prof. Masayuki Takeda of Kyushu University for his various support. I thank Prof. Osamu Wannabe and Prof. Eiji Takimoto for their discussion. I also thank anonymous referees for their helpful comments. This work is supported in part by the 21st century COE program at Graduate School of Information Science and Electrical Engineering in Kyushu University.

	# of sampled examples	# of accepted examples	time (sec.)	test error (%)
Ada.	N/A	N/A	1349	5.6
Mada.	1,032,219	157,320	493	6.7
Gini.	1,039,943	156,856	408	5.8
Gini2.	1,027,874	140,916	359	5.5

Table 1. Summary of experiments over Reuters-2158.

References

1. J. A. Aslam. Improving algorithms for boosting. In *Proc. 13th Annu. Conference on Comput. Learning Theory*, pages 200–207, 2000.
2. Jose L. Balcazar, Yang Dai, and Osamu Watanabe. Provably fast training algorithms for support vector machines. In *Proceedings of IEEE International Conference on Data Mining (ICDM'01)*, pages 43–50, 2001.
3. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth International Group, 1984.
4. Sanjoy Dasgupta and Philip M. Long. Boosting with diverse base classifers. In *Proceedings of the 16th Annual Conference on Learning Theory and 7th Kernel Workshop*, pages 273–287, 2003.
5. C. Domingo, R. Gavaldà, and O. Watanabe. Adaptive sampling methods for scaling up knowledge discovery algorithms. *Data Mining and Knowledge Discovery*, 6(2):131–152, 2002.
6. C. Domingo and O. Watanabe. MadaBoost: A modification of AdaBoost. In *Proceedings of 13th Annual Conference on Computational Learning Theory*, pages 180–189, 2000.
7. P. Domingos and G. Hulten. Mining high-speed data streams. In *Proceedings of the Sixth ACM International Conference on Knowledge Discovery and Data Mining*, pages 71–80, 2000.
8. W. Feller. *An introduction to probability theory and its applications.* Wiley, 1950.
9. Y. Freund. An improved boosting algorithm and its implications on learning complexity. In *Proc. 5th Annual ACM Workshop on Computational Learning Theory*, pages 391–398. ACM Press, New York, NY, 1992.
10. Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.
11. Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.
12. Yoav Freund. An adaptive version of the boost by majority algorithm. In *COLT '99: Proceedings of the twelfth annual conference on Computational learning theory*, pages 102–113, 1999.
13. J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statisitics*, 2:337–374, 2000.
14. D. Gavinsky. Optimally-smooth adaptive boosting and application to agnostic learning. *Journal of Machine Learning Research*, 2003.
15. K. Hatano and M. K. Warmuth. Boosting versus covering. In *Advances in Neural Information Processing Systems 16*, 2003.

16. K. Hatano and O. Watanabe. Learning r-of-k functions by boosting. In *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, pages 114–126, 2004.
17. Kohei Hatano. Smooth boosting using an information-based criterion. Technical Report DOI-TR-225, Department of Informatics, Kyushu University, 2006.
18. M. Kearns and Y. Mansour. On the boosting ability of top-down decision tree learning algorithms. *Journal of Computer and System Sciences*, 58(1):109–128, 1999.
19. Yishay Mansour and David A. McAllester. Boosting using branching programs. *Journal of Computer and System Sciences*, 64(1):103–112, 2002.
20. R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.
21. Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.
22. Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
23. Tobias Scheffer and Stefan Wrobel. Finding the most interesting patterns in a database quickly by using sequential sampling. *Journal of Machine Learning Research*, 3:833–862, 2003.
24. R. J. Serfling. *Approximation theorems of mathematical statistics*. Wiley, 1980.
25. R. A. Servedio. Smooth boosting and learning with malicious noise. In *14th Annual Conference on Computational Learning Theory*, pages 473–489, 2001.
26. Eiji Takimoto, Syuhei Koya, and Akira Maruoka. Boosting based on divide and merge. In *Proceedings of the 15th International Conference on Algorithmic Learning Theory*, pages 127–141, 2004.
27. L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Appendix

Lemma 3 Let $L(x) = x + 1$, if $x > 0$ and e^x, otherwise. Then it holds for any $a \in \mathbb{R}$ and any $x \in [-1, +1]$ that

$$L(x + a) \leq L(a) + L'(a)x + L'(a)x^2.$$

Proof. For any $x \in [-1, 1]$, let $g_x(a) = L(a)+L'(a)(x+x^2)-L(x+a)$. We consider the following cases. (Case 1: $x+a, a \leq 0$) We have $g_x(a) = e^a(1+x+x^2-e^x) \geq 0$, as $e^x \leq 1 + x + x^2$ for $x \in [-1, 1]$. (Case 2: $x + a, a \geq 0$) It is immediate to see that $g_x(a) = x^2 \geq 0$. (Case 3: $x + a < 0$, and $a > 0$) It holds that $g_x(a) = 1+a+x+x^2-e^{x+a} \geq 0$ since $g'_x(a) = 1-e^{x+a} > 0$ and $g_x(0) = 1+x+x^2-e^x \geq 0$. (Case 4: $x+a > 0$, and $a < 0$) By using the fact that $1+x+x^2 \geq e^x$ for $x \in [-1, 1]$, we have $g_x(a) = e^a(1 + x + x^2) - (x + a + 1) \geq e^{x+a} - (1 + x + a) \geq 0$. □