

The interaction of stability and weakness in AdaBoost

Samuel Kutin Partha Niyogi

October 19, 2001

Abstract

We provide an analysis of AdaBoost within the framework of algorithmic stability. In particular, we show that AdaBoost is a stability-preserving operation: if the “input” (the weak learner) to AdaBoost is stable, then the “output” (the strong learner) is almost-everywhere stable. Because classifier combination schemes such as AdaBoost have greatest effect when the weak learner is weak, we discuss weakness and its implications. We also show that the notion of almost-everywhere stability is sufficient for good bounds on generalization error. These bounds hold even when the weak learner has infinite VC dimension.

1 Introduction

We provide an analysis of AdaBoost within the framework of algorithmic stability. In particular, we show that AdaBoost is a stability-preserving operation: if the “input” (the weak learner) to AdaBoost is stable, then the “output” (the strong learner) is almost-everywhere stable. Because classifier combination schemes such as AdaBoost have greatest effect when the weak learner is weak, we discuss weakness and its implications. We also show that the notion of almost-everywhere stability is sufficient for good bounds on generalization error. These bounds hold even when the weak learner has infinite VC dimension.

Recently, Bousquet and Elisseeff [1] used the notion of algorithmic stability to prove generalization error bounds for learning algorithms. The attraction of such an approach is that it avoids the traditional notion of VC dimension [11], and allows us to focus on a wider class of learning algorithms

than empirical risk minimization. For example, this approach provides generalization error bounds for regularization-based learning algorithms [1] that have been awkward to analyze within the VC framework.

Boosting algorithms have attracted some attention, particularly since the introduction of AdaBoost by Freund and Schapire [5]. Boosting enables one to combine weak hypotheses into a strong hypothesis. Such algorithms fall outside the rubric of empirical risk minimization. However, most attempts to analyze the generalization performance of AdaBoost, including margin analysis [9], have relied on the notion of VC dimension.

There are two problems with such analyses. First, they apply only when the weak learner has finite VC dimension. For example, if the weak learner itself is based on regularization, existing analyses [5, 9] do not apply. Does the boosting process, which forces the weak learner to focus on hard examples, undo the regularization, and overfit the data? Or do we perform some sort of global regularization? Second, in practice, the observed generalization performance is better than that predicted by the VC theoretic analyses.

To address these issues, we turn to the notion of algorithmic stability [4]. An algorithm is stable at a training set S if changing one point in S yields only a small change in the output hypothesis. Breiman [3] argues that unstable weak learners benefit from randomization algorithms such as bagging [2]. He finds that, when the weak learner is stable, bagging does not help, and suggests that AdaBoost is more effective in this case.

Kearns and Ron [6] consider both algorithmic stability and the weaker, related notion of error stability. They prove bounds on the error of cross-validation estimates of generalization error, but their arguments rely on VC theory. Bousquet and Elisseeff [1] prove that an algorithm which is stable everywhere has low generalization error; their proof does not make any reference to VC dimension. They show that regularization networks are stable.

In this paper, we show that the notion of algorithmic stability can also be applied to boosting. We focus on AdaBoost—in fact, our analysis applies only to AdaBoost. This is not a limitation, but rather a strength; we use details about how AdaBoost searches the space of linear combinations. Our analysis does not make use of VC theory. We believe that a similar analysis could be carried out for other boosting techniques.

Our main result is Theorem 5.8: If the base learning algorithm is “weak” (See Section 4) and stable, then the output of AdaBoost has good generalization error. Our proof can be divided into three parts:

1. Lemma 5.4: If the weak learner is stable, then, for “good” training sets, the output of AdaBoost is stable.
2. Theorem 4.15: Under a reasonable assumption of weakness, most training sets are “good.”
3. Theorem 3.4: An algorithm which is almost-everywhere stable has low generalization error.

We prove these points in the reverse of the order above. We first prove Theorem 3.4. Our proof follows the argument of Bousquet and Elisseeff [1], who show that an everywhere-stable algorithm has low generalization error. Their argument uses the method of “independent bounded differences,” developed by McDiarmid [7]. Our Lemma 3.2 is a generalization of McDiarmid’s lemma. This enables us to prove our more general version of Bousquet and Elisseeff’s result.

We next prove Theorem 4.15. We call a training set “good” when no hypothesis has too low an error rate. Obviously it is possible to construct a bad training set (e.g., if all training examples are identical). We show that, as long as the expected error rate of the best hypothesis on a random training set is bounded away from zero, most training sets are good. Our proof again uses McDiarmid’s method of independent bounded differences.

Note that most analyses of boosting require some assumption about the *strength* of the weak learner. Our argument instead requires us to assume that the weak learner is *weak*. Classifier combination schemes such as boosting and bagging are meaningful only when the individual classifiers in the ensemble are weak (in other words, not good). If the weak learner does very well, we do not need to boost its results. We also point out that our analysis carries through in the noisy setting.

Finally, we prove Lemma 5.4. The argument is technical, and the bounds that we get are rather large; our bounds grow as 2^{T^2} , where T is the number of rounds of AdaBoost. In the case where the VC dimension of the weak learner is finite, the VC theoretic bounds depend linearly on T . Of course, our bounds hold even when the weak learner has infinite VC dimension. Our original hope was that algorithmic stability would provide us with a better bound than VC theory. This remains an open question.

2 Preliminaries

We begin with some general definitions of our learning theory framework in Section 2.1. We discuss various notions of algorithmic stability in Section 2.2.

2.1 Generalization error

We first describe the setting for our learning algorithms. There is a space \mathcal{X} of points, with some unknown distribution Δ . A target operator takes as input an element $x \in \mathcal{X}$ and outputs some y in a set \mathcal{Y} of labels, according to a conditional distribution function $F(y | x)$. For simplicity's sake, we assume $\mathcal{Y} = \{0, 1\}$. We write $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, and we write D for the distribution on \mathcal{Z} induced by Δ and F .

A *classifier*, also called a *hypothesis*, is a function $h: \mathcal{X} \rightarrow [0, 1]$. Informally, a learning algorithm is given a finite subset of \mathcal{Z} drawn according to D , and attempts to construct a classifier h such that $h(x)$ is a good approximation to the output of the target operator on input x . The definitions in this section clarify exactly what this means. Note that we allow our classifiers to output values between 0 and 1; this corresponds to a confidence-rated prediction.

One special setting is when there is a *target function* $f: \mathcal{X} \rightarrow \mathcal{Y}$, and a *noise level* η . In this case, the target operator outputs $f(x)$ with probability $1 - \eta$ and $1 - f(x)$ with probability η . We refer to the case $\eta = 0$ as the *noiseless setting*, and the case $\eta > 0$ as the *noisy setting*.

Note 2.1 If D is a distribution on a set X , we use the notation $x \sim D$ to mean that x is chosen from X according to distribution D .

Definition 2.2 The *cost* of a classifier h on a point $z \in \mathcal{Z}$ is a measure of the error h makes on z . We denote this cost by $c(h, z)$, and we require $0 \leq c(h, z) \leq M$ for some constant M .

In sections 3 and 4, we will use this general setting. In Section 5, when we analyze AdaBoost in detail, we will assume

$$c(h, (x, y)) = |h(x) - y|,$$

in which case $M = 1$.

Definition 2.3 The error rate of a classifier $h: \mathcal{X} \rightarrow [0, 1]$, with respect to a distribution D , is

$$\text{Err}_D(h) = \mathbf{E}_{z \sim D}(c(h, z)).$$

The error rate of a collection \mathcal{H} of classifiers is

$$\text{Err}_D(\mathcal{H}) = \min_{h \in \mathcal{H}} \{\text{Err}_D(h)\}.$$

Note that we use this notation even when D has finite support. For $S \in \mathcal{Z}^m$, we use the notation Err_S to mean Err_p where p is the uniform distribution on S .

Definition 2.4 Let p, q be two distributions on \mathcal{Z} with finite support. The *distance* between them, denoted $\|p - q\|$, is the L_1 -norm of $p - q$,

$$\sum_{z \in \mathcal{Z}} |p(z) - q(z)|,$$

where this sum is well-defined because $p(z) = q(z) = 0$ for all but finitely many $z \in \mathcal{Z}$.

Definition 2.5 A *learning algorithm* is a process which takes as input a distribution p on \mathcal{Z} with finite support, and outputs a function $f_p: \mathcal{X} \rightarrow [0, 1]$. For $S \in \mathcal{Z}^m$, we use the notation f_S to mean f_p where p is the uniform distribution on S .

A *symmetric* learning algorithm depends only on the instances given to it, not on the order in which they are presented. Note that, by our definition, any learning algorithm is necessarily symmetric.

Note 2.6 The more usual definition of a learning algorithm is a process which takes as input a finite training set $S \in \cup_m \mathcal{Z}^m$. As noted above, we associate a finite training set with the uniform distribution on that set. We also note that, if we have a collection of points z_1, \dots, z_m , with rational weights w_1, \dots, w_m , we can write $w_i = a_i/K$ over some common denominator K , and then represent our weighted distribution as a collection of K points (where point z_i occurs a_i times). The two definitions are thus equivalent, as long as we restrict our attention to distributions with rational weights, and as long as we require our learning algorithms to be symmetric. We find Definition 2.5 to be simpler for our purposes. Since only distributions with rational weights can arise within AdaBoost, the distinction is purely one of notation.

Definition 2.7 The *training error* of a learning algorithm on an input S is the error rate of f_S on the set S , or $\text{Err}_S(f_S)$. The *true error* of a learning algorithm on an input S is the error rate of f_S on a randomly chosen example, or $\text{Err}_D(f_S)$. The *generalization error* of a learning algorithm on an input S is the difference between the observed error rate and the true error rate, or

$$|\text{Err}_D(f_S) - \text{Err}_S(f_S)|.$$

This quantity is sometimes called the *estimation error*.

Given a finite set of examples, it is easy to construct a rule which gets all of them right; the challenge is to generate hypotheses with a low error rate on new data drawn from the same distribution. Intuitively, we say that a learning algorithm is good if the training error is small and if the generalization error is small, since then the true error will also be small.

Our goal in this paper is to bound the probability that the generalization error of a learning algorithm A is large. Note that this is not sufficient to imply that A is good; we would also need to know that the training error is small.¹ We focus on generalization error bounds because, for AdaBoost, they have been more elusive than training error bounds.

2.2 Definitions of stability

For $S \in \mathcal{Z}^m$, $S = (z_1, \dots, z_m)$, we let S^i denote $S \setminus z_i$, the training set with the i th instance removed. So, for each i , $S^i \in \mathcal{Z}^{m-1}$. For $u \in \mathcal{Z}$, we let $S^{i,u}$ denote $S^i \cup \{u\}$, the training set with the i th instance replaced by u . So $S^{i,u} \in \mathcal{Z}^m$.

Definition 2.8 (Devroye and Wagner [4]) We say that a learning algorithm has *leave-one-out stability* β if the following holds:

$$\forall S \in \mathcal{Z}^m, \quad \forall i, \quad \forall z \in \mathcal{Z}, \quad |c(f_S, z) - c(f_{S^i}, z)| \leq \beta.$$

Definition 2.9 (Bousquet and Elisseeff [1]) We say that a learning algorithm has *change-one stability* β if the following holds:

$$\forall S \in \mathcal{Z}^m, \quad \forall i, \quad \forall u, z \in \mathcal{Z}, \quad |c(f_S, z) - c(f_{S^{i,u}}, z)| \leq \beta.$$

¹Ideally, we would like to prove that, as the size of the training set grows, the classifiers generated by A approach the optimal classifier minimizing $\text{Err}_D(h)$ over the set of classifiers available to A .

Note that, in both of these definitions, we view β as a function of m . We are most interested in the case where $\beta = \lambda/m$ for a constant λ .

Note 2.10 We include both change-one stability and leave-one-out stability for the sake of completeness, but the distinction between them is minor. It is clear that, if a learning algorithm has leave-one-out stability β , then it has change-one stability 2β . The converse is not true (for example, our algorithm might have radically different behavior when the size of the training set is even or odd). However, we show in Lemma 2.12 that, for any constant λ , change-one stability λ/m implies leave-one-out stability λ/m .

We will be considering learning algorithms which take as input not merely a set of points, but a weighted set of points. We will thus require the following definition of stability:

Definition 2.11 We say that a learning algorithm has L_1 -stability λ , or is λ - L_1 -stable, if for any two distributions p, q on \mathcal{X} with finite support,

$$\forall z \in \mathcal{Z}, \quad |c(f_p, z) - c(f_q, z)| \leq \lambda \|p - q\|.$$

We say that the algorithm is L_1 -stable if it has L_1 -stability λ for some constant λ .

We now justify the interchangeability of these notions of stability.

Lemma 2.12 *A learning algorithm has L_1 -stability λ if and only if it has change-one stability $2\lambda/m$. Furthermore, if a learning algorithm has L_1 -stability λ , it has leave-one-out stability $2\lambda/m$.*

Proof: First, let A be a learning algorithm with L_1 -stability λ . Choose any $S \in \mathcal{Z}^m$, and any i ; let p be the uniform distribution on S , and q the uniform distribution on S^i . Then

$$\|p - q\| = \frac{1}{m} + (m-1) \left(\frac{1}{m-1} - \frac{1}{m} \right) = \frac{2}{m},$$

and hence, for any $z \in \mathcal{Z}$,

$$|c(f_S, z) - c(f_{S^i}, z)| \leq \lambda \|p - q\| = \frac{2\lambda}{m}.$$

So, A has leave-one-out stability $2\lambda/m$.

If we instead choose some $u \in \mathcal{Z}$, and let p be the uniform distribution on S and q the uniform distribution on $S^{i,u}$, then, again, $\|p - q\| = 2/m$, and A thus has change-one stability $2\lambda/m$ as well.

Now, let B be a learning algorithm with change-one stability $2\lambda/m$. Choose $S \in \mathcal{Z}^m$, and let p, q be two distributions on S . We assume that all values $p(z), q(z)$ are rational, so we can write all weights over a common denominator K . So p can be associated with an element $P \in \mathcal{Z}^K$, where a point with weight a/K is repeated a times in P . Similarly q can be associated with $Q \in \mathcal{Z}^K$. Note that $f_p = f_P$ and $f_q = f_Q$.

We now perform the following sequence of operations: start with $P_0 = P$, and replace some point which occurs more often in P than in Q with a point which occurs more often in Q than in P . This gives us $P_1 \in \mathcal{Z}^K$. Since algorithm B has change-one stability $2\lambda/m$,

$$\forall z \in \mathcal{Z}, \quad |c(f_{P_0}, z) - c(f_{P_1}, z)| \leq \frac{2\lambda}{K}.$$

We now repeat this process, constructing a sequence of K -tuples $P_t \in \mathcal{Z}^K$, until, at some time, $P_T = Q$. (Technically, we get $P_T = Q$ up to rearrangement of terms, but our learning algorithm is symmetric, so $f_{P_T} = f_Q$.) Thus,

$$\forall z \in \mathcal{Z}, \quad |c(f_p, z) - c(f_q, z)| \leq \frac{2T\lambda}{K}.$$

Let p_t denote the distribution on S arising from the number of occurrences of each element in P_t . Then $\|p_t - p_{t+1}\| = 2/K$, and we conclude that $\|p - q\| = 2T/K$. Hence we have shown that B is λL_1 -stable. \blacksquare

Note 2.13 We prove Lemma 2.12 only in the case where the weights $p(z), q(z)$ are rational. As we remark in Note 2.6, this is the only case in which we are interested.

We now introduce the notion of stability at a point. Kearns and Ron [6] use leave-one-out stability, but for consistency we phrase our definition in terms of change-one stability.

Definition 2.14 (Kearns and Ron [6]) We say that a learning algorithm A is β -stable at S , if

$$\forall i \in \{1, \dots, m\}, \quad \forall u, z \in \mathcal{Z}, \quad |c(f_S, z) - c(f_{S^{i,u}}, z)| \leq \beta.$$

We say that A is (β, δ) -stable if

$$\Pr_{S \sim D^m} (A \text{ is } \beta\text{-stable at } S) \geq 1 - \delta.$$

We view β, δ as functions of m . We will be interested in the case where $\beta = \lambda/m$ for some constant λ and $\delta = e^{-\Omega(m)}$.

Note that A is $(\beta, 0)$ -stable if and only if A has change-one stability β .

2.2.1 Error Stability

Kearns and Ron [6] refer to the notion of (β, δ) -stability as *hypothesis stability*. They also work with *error stability*:

Definition 2.15 (Kearns and Ron [6]) We say that a learning algorithm has *error stability* (β, δ) if, for any i ,

$$\Pr_{S \sim D^m} (|\text{Err}_D(f_S) - \text{Err}_D(f_{S^i})| \geq \beta) \leq \delta.$$

(Since our learning algorithms are symmetric, it does not matter whether we do this for all i or we consider a probability distribution on i .)

This is similar to the notion of (β, δ) -stability; instead of requiring the hypotheses to be close for all inputs, we simply require the overall error rates to be close.

Kearns and Ron [6] prove that a learning algorithm with good error stability, and with a hypothesis class of bounded VC dimension, has low generalization error. They point out that the error-stability condition is significantly weaker than hypothesis stability, and that results involving error stability are thus more widely applicable. In particular, they show that any algorithm performing empirical risk minimization over a set of bounded VC dimension has good error stability.

In Theorem 3.4, we will prove a relationship between (β, δ) -stability and low generalization error. It seems reasonable to ask whether we could replace (β, δ) -stability with error stability in Theorem 3.4. In fact, we cannot.

Example 2.16 Consider the learning algorithm which generates the following classifier: get all training instances right, and return $1/2$ on any unknown point. (If we require our classifiers to have output in $\{0, 1\}$, we could simply guess randomly.) For such an algorithm, $\text{Err}_D(f_S) = 1/2$ and $\text{Err}_S(f_S) = 0$ for any S , so this algorithm has high generalization error. However, it has error-stability $(\beta, 0)$ for any $\beta > 0$.

The point is that error stability, in the absence of bounded VC dimension, is not sufficient to prove bounds on generalization error. Since our goal is to prove bounds without regard to VC dimension, we use the notion of hypothesis stability throughout.

3 Stability and generalization error

Devroye and Wagner [4] were the first to observe a connection between the stability of an algorithm and its generalization error. Bousquet and Elisseeff [1] showed that $(\beta, 0)$ -stability implies low generalization error. In Section 3.2, we prove that (β, δ) -stability implies low generalization error. Our proof is essentially the same as theirs, but requires a stronger concentration lemma, which we prove in Section 3.1.

3.1 Independent bounded differences

The “independent bounded differences inequality” is due to Colin McDiarmid [7].

Lemma 3.1 (McDiarmid) *Let X_1, \dots, X_m be independent random variables, with X_k taking values in a set A_k for each k . Let $g: \prod A_k \rightarrow \mathbb{R}$ be a measurable function. Suppose that, for any k , if $a, a' \in \prod A_k$ differ only in the k th coordinate, that*

$$|g(a) - g(a')| \leq c_k.$$

Let $\mu = \mathbf{E}(g(X_1, \dots, X_m))$. Then, for any $\tau > 0$,

$$\Pr_{X_1, \dots, X_m} (g(X_1, \dots, X_m) - \mu) \geq \tau \leq \exp\left(\frac{-2\tau^2}{\sum c_k^2}\right)$$

The above result is sufficient for our purposes in Section 4. However, for Theorem 3.4, we will need the following, stronger statement, which allows for the possibility of a “bad” set B :

Lemma 3.2 *Let $X = (X_1, \dots, X_m)$ be independent random variables, with X_k taking values in a set A_k for each k . Let $g: \prod A_k \rightarrow \mathbb{R}$ be a measurable function. Let B be a subset of $\prod A_k$, and let $\delta = \Pr(X \in B)$.*

Suppose that, for any k , if $a, a' \in \prod A_k$ differ only in the k th coordinate, that

$$|g(a) - g(a')| \leq b_k.$$

Suppose further that, if $a \notin B$, we have

$$|g(a) - g(a')| \leq c_k.$$

Let $\mu = \mathbf{E}(g(X_1, \dots, X_m))$. Then, for any $\tau > 0$, and any $\delta_1, \dots, \delta_m > 0$,

$$\Pr_X(|g(X) - \mu| \geq \tau) \leq 2 \left(\exp\left(\frac{-2\tau^2}{\sum_k (c_k + b_k \delta_k)^2}\right) + \delta \sum_k \frac{1}{\delta_k} \right).$$

In particular, setting $\delta_k = c_k/b_k$,

$$\Pr(|g(X) - \mu| \geq \tau) \leq 2 \left(\exp\left(\frac{-\tau^2}{2 \sum_k c_k^2}\right) + \delta \sum_k \frac{b_k}{c_k} \right).$$

The set B is a “bad” set where the bound c_k does not hold; however, even on set B , the differences are bounded.

Our argument follows a proof of Lemma 3.1 by McDiarmid [8]. For any k , and any a_1, \dots, a_{k-1} with $a_i \in A_i$, let Γ be the event that $X_i = a_i$ for $1 \leq i \leq k-1$. We define $\phi: A_k \rightarrow \mathbb{R}$ by

$$\phi(x) = \mathbf{E}(g(X) \mid \Gamma, X_k = x) - \mathbf{E}(g(X) \mid \Gamma).$$

Next we define the *range* of $\phi(x)$ to be

$$\begin{aligned} \text{ran}(a_1, \dots, a_{k-1}) &= \sup_{x, y \in A_k} \{|\phi(x) - \phi(y)|\} \\ &= \sup_{x, y \in A_k} \{|\mathbf{E}(g(a_1, \dots, a_{k-1}, x, X_{k+1}, \dots, X_m)) \\ &\quad - g(a_1, \dots, a_{k-1}, y, X_{k+1}, \dots, X_m)|\}. \end{aligned}$$

For $a_1, \dots, a_m \in \prod A_k$, we define the *sum of squared ranges* by

$$R^2(a_1, \dots, a_m) = \sum_{k=1}^m (\text{ran}(a_1, \dots, a_{k-1}))^2.$$

We are now ready to state the lemma we need to prove Lemma 3.2. This lemma is also due to McDiarmid [8, Theorem 3.7].

Lemma 3.3 (McDiarmid) *Let X_1, \dots, X_m be independent random variables, with X_k taking values in a set A_k for each k . Let $g: \prod A_k \rightarrow \mathbb{R}$ be a bounded measurable function. Let \hat{r}^2 denote the maximum sum of squared ranges $\sup_a \{R^2(a)\}$. Let $\mu = \mathbf{E}(g(X))$. Then, for any $\tau \geq 0$,*

$$\Pr(g(X) - \mu \geq \tau) \leq \exp(-2\tau^2/\hat{r}^2).$$

More generally, let B be any “bad” subset of $\prod A_k$, such that $R^2(a) \leq r^2$ for each $a \notin B$. Then

$$\Pr(g(X) - \mu \geq \tau) \leq \exp(-2\tau^2/r^2) + \Pr(X \in B).$$

Proof of Lemma 3.2: Fix some k and a_1, \dots, a_{k-1} , and let Γ be the event that $X_i = a_i$ for $1 \leq i \leq k-1$. We are interested in bounding $\text{ran}(a_1, \dots, a_{k-1})$. So, choose some x, y , and some X_{k+1}, \dots, X_m ; let $a = (a_1, \dots, a_{k-1}, x, X_{k+1}, \dots, X_m)$, and let a' be the same but with y as the k th coordinate. For most choices of a , we will have $|g(a) - g(a')| \leq c_k$; for all choices, $|g(a) - g(a')| \leq b_k$. We conclude that

$$\text{ran}(a_1, \dots, a_{k-1}) \leq c_k + b_k \Pr(X \in B \mid \Gamma).$$

Now, let B_k be the subset of $\prod A_k$ defined by

$$B_k = \{(a_1, \dots, a_m) \mid \Pr(X \in B \mid \Gamma) > \delta_k\}.$$

We observe that

$$\delta = \Pr(X \in B) \geq \Pr(X \in B \mid X \in B_k) \Pr(X \in B_k) \geq \delta_k \Pr(X \in B_k),$$

and hence

$$\Pr(X \in B_k) \leq \delta/\delta_k.$$

So, define $B' = \cup_k B_k$; then

$$\Pr(X \in B') \leq \delta \sum_{k=1}^m \frac{1}{\delta_k}.$$

Now, if $a \notin B'$, we see that

$$\begin{aligned} R^2(a) &= \sum_{k=1}^m (\text{ran}(a_1, \dots, a_{k-1}))^2 \\ &\leq \sum_{k=1}^m (c_k + b_k \Pr(X \in B \mid \Gamma))^2 \\ &\leq \sum_{k=1}^m (c_k + b_k \delta_k)^2. \end{aligned}$$

The result now follows immediately from Lemma 3.3. ■

Note that, if we have $c_k = c$ and $b_k = b$, Lemma 3.2 gives us

$$\Pr(|g(X) - \mu| \geq \tau) \leq 2 \left(\exp\left(\frac{-\tau^2}{2mc^2}\right) + \frac{mb\delta}{c} \right).$$

In our applications $c \rightarrow 0$ polynomially in m , and $\delta \rightarrow 0$ exponentially in m , so this is strong enough.

3.2 (β, δ) -stability and generalization error

We are now able to prove our stronger version of the theorem of Bousquet and Elisseeff [1], using Lemma 3.2 where they use Lemma 3.1.

Theorem 3.4 *Let A be a (β, δ) -stable algorithm, such that $0 \leq c(f_S, z) \leq M$ for all $z \in \mathcal{Z}$ and all learning sets S . Then, for all $\tau > 0$, and all $m \geq 1$,*

$$\begin{aligned} \Pr_{S \sim D^m} (|\text{Err}_S(f_S) - \text{Err}_D(f_S)| > \tau + \beta + M\delta) \\ \leq 2 \exp\left(\frac{-\tau^2 m}{2(2m\beta + M)^2}\right) + \frac{4m^2 M\delta}{2m\beta + M}. \end{aligned}$$

We note, following Bousquet and Elisseeff [1], that, if $\beta = \lambda/m$ for some constant λ , and if δ decreases exponentially with m , then this gives us exponentially low generalization error.

Proof: Let $g(S) = \text{Err}_S(f_S) - \text{Err}_D(f_S)$, and let $\mu = \mathbf{E}_{S \sim D^m}(g(S))$. Following Bousquet and Elisseeff [1], we first bound μ , and then compute the constants b_k and c_k for which we can apply Lemma 3.2.

Fix some i , $1 \leq i \leq m$. Writing $S = (z_1, \dots, z_m)$, we note that

$$\mathbf{E}_{S \sim D^m}(\text{Err}_S(f_S)) = \frac{1}{m} \sum_{j=1}^m \mathbf{E}_{S \sim D^m}(c(f_S, z_j)) = \mathbf{E}_{S \sim D^m}(c(f_S, z_i))$$

because our learning algorithm is symmetric. Since $S^{i,u}$ is simply S with the i th entry replaced by u , we also have

$$\mathbf{E}_{S \sim D^m}(\text{Err}_S(f_S)) = \mathbf{E}_{S, u \sim D^{m+1}}(\text{Err}_{S^{i,u}}(f_{S^{i,u}})) = \mathbf{E}_{S, u \sim D^{m+1}}(c(f_{S^{i,u}}, u)). \quad (1)$$

By definition,

$$\mathbf{E}_{S \sim D^m}(\text{Err}_D(f_S)) = \mathbf{E}_{S, u \sim D^{m+1}}(c(f_S, u)). \quad (2)$$

So, subtracting (2) from (1), we get that, for any i ,

$$\mathbf{E}_{S \sim D^m}(g(S)) = \mathbf{E}_{S, u \sim D^{m+1}}(c(f_S, u) - c(f_{S^{i,u}}, u)).$$

Let $\psi(S, u) = c(f_S, u) - c(f_{S^{i,u}}, u)$. Our learning algorithm is (β, δ) -stable. So, if we choose S according to D^m , then, with probability $1 - \delta$, $|\psi(S, u)| \leq \beta$ for all u . For any S and u , we know $|\psi(S, u)| \leq M$. We conclude that

$$|\mathbf{E}_{S \sim D^m}(g(S))| \leq \beta + \delta M.$$

Now, for all S, i, u , we have $|g(S) - g(S^{i,u})| \leq 2M$. Assume now that A is β -stable at S , fix some u , and let S' denote $S^{i,u}$. Then

$$\begin{aligned} |\text{Err}_S(f_S) - \text{Err}_{S'}(f_{S'})| &\leq \frac{1}{m} |c(f_S, z_i) - c(f_{S'}, u)| \\ &\quad + \frac{1}{m} \sum_{j \neq i} |c(f_S, z_j) - c(f_{S'}, z_j)| \leq \frac{M}{m} + \beta. \end{aligned}$$

Also,

$$|\text{Err}_D(f_S) - \text{Err}_D(f_{S'})| \leq \mathbf{E}_{z \sim D} |c(f_S, z) - c(f_{S'}, z)| \leq \beta.$$

Hence, $|g(S) - g(S')| \leq M/m + 2\beta$ whenever A is β -stable at S .

We can now apply Lemma 3.2. We have $c_k = 2\beta + \frac{M}{m}$, and $b_k = 2M$, and hence

$$\begin{aligned} \Pr(|g(S) - \mu| \geq \tau) &\leq 2 \left(\exp\left(\frac{-\tau^2}{2m(2\beta + M/m)^2}\right) + \frac{2mM\delta}{2\beta + M/m} \right) \\ &= 2 \left(\exp\left(\frac{-\tau^2 m}{2(2m\beta + M)^2}\right) + \frac{2m^2 M \delta}{2m\beta + M} \right). \end{aligned}$$

Since $|\mu| \leq \beta + M\delta$, this concludes the proof. ■

4 Weakness

Boosting algorithms combine weak classifiers to build strong classifiers. It is thus reasonable to assume that our base learning algorithm is “weak.” In this section we define what we mean by weakness. We first discuss the weakness of spaces of classifiers in Section 4.1, and then the weakness of learning algorithms in Section 4.2. We show that, in the noisy setting, certain types of stable regularization algorithms are necessarily weak. In Section 4.3, we show that a weak learner performs poorly on almost all training sets.

4.1 Weakness of classifiers

Recall that D is a distribution on $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$; in one common case, D is induced by a distribution on \mathcal{X} and a target function $f: \mathcal{X} \rightarrow \mathcal{Y}$.

Definition 4.1 The *weakness* of a hypothesis class \mathcal{H} with respect to a distribution D on \mathcal{Z} is given by

$$\text{Weak}_D(\mathcal{H}) = \liminf_{m \rightarrow \infty} \mathbf{E}_{S \sim D^m}(\text{Err}_S(\mathcal{H})).$$

We say that \mathcal{H} is *weak* with respect to D if $\text{Weak}_D(\mathcal{H}) > 0$.

In words, we say that a hypothesis class is weak if, for sufficiently large m , the expected error rate of the best $h \in \mathcal{H}$ on m randomly chosen points in \mathcal{Z} is bounded away from zero.

It might seem simpler to reverse the role of \mathbf{E}_S and \min_h in Definition 4.1. (Recall that $\text{Err}_S(\mathcal{H}) = \min_h \{\text{Err}_S(h)\}$.) This would yield:

$$\liminf_{m \rightarrow \infty} \min_{h \in \mathcal{H}} \{\mathbf{E}_{S \sim D^m}(\text{Err}_S(h))\} = \liminf_{m \rightarrow \infty} \min_{h \in \mathcal{H}} \{\text{Err}_D(h)\} = \text{Err}_D(\mathcal{H}).$$

It is clear that

$$\text{Weak}_D(\mathcal{H}) \leq \text{Err}_D(\mathcal{H}). \tag{3}$$

In general, there is no reason for these two quantities to be the same. For example, if we consider the hypothesis class generated by the learning algorithm of Example 2.16, we have $\text{Weak}_D(\mathcal{H}) = 0$ and $\text{Err}_D(\mathcal{H}) = 1/2$. However, in the case where \mathcal{H} has bounded VC dimension, we have equality:

Proposition 4.2 *If \mathcal{H} has bounded VC dimension², then*

$$\text{Weak}_D(\mathcal{H}) = \text{Err}_D(\mathcal{H}).$$

Proof: For any $\epsilon > 0$, we know [11]:

$$\lim_{m \rightarrow \infty} \Pr_{S \sim D^m} (\sup_{h \in \mathcal{H}} \{|\text{Err}_D(h) - \text{Err}_S(h)|\} > \epsilon) = 0.$$

Let E denote $\text{Err}_D(\mathcal{H})$. For any $h \in \mathcal{H}$, $\text{Err}_D(h) \geq E$. Hence, for any $\delta, \epsilon > 0$, for sufficiently large m ,

$$\Pr_{S \sim D^m} (\min_{h \in \mathcal{H}} \{\text{Err}_S(h)\} < E - \epsilon) < \delta,$$

and therefore

$$\mathbf{E}_{S \sim D^m} (\text{Err}_S(\mathcal{H})) \geq (1 - \delta)(E - \epsilon)$$

for sufficiently large m , implying that

$$\text{Weak}_D(\mathcal{H}) \geq (1 - \delta)(E - \epsilon).$$

Since this is true for any $\delta, \epsilon > 0$, we must have $\text{Weak}_D(\mathcal{H}) \geq E$. ■

In particular, in the noisy setting, any hypothesis class of bounded VC dimension is weak; the weakness is at least the Bayes error rate.

4.2 Weakness of learning algorithms

We now turn to the weakness of learning algorithms.

Note 4.3 Throughout this section, we use \mathcal{H}_m to denote

$$\mathcal{H}_m = \{f_p \mid p \text{ has support of size at most } m\}.$$

Note that, if $m < n$, then $\mathcal{H}_m \subseteq \mathcal{H}_n$. We use \mathcal{H}_∞ to denote $\cup_m \mathcal{H}_m$; this is the set of all classifiers that could be generated by learning algorithm A .

Definition 4.4 The *weakness* of a learning algorithm A , denoted $\text{Weak}_D(A)$, is given by

$$\text{Weak}_D(A) = \liminf_{m \rightarrow \infty} \mathbf{E}_{S \sim D^m} (\text{Err}_S(\mathcal{H}_m)).$$

We say that an algorithm A is *weak* with respect to D if $\text{Weak}_D(A) > 0$.

²In fact, since we do not need a fast rate of convergence, it suffices to assume that, for all $\epsilon > 0$, $H^{\mathcal{H}}(\epsilon; m)/m \rightarrow 0$ as $m \rightarrow \infty$, where $H^{\mathcal{H}}$ is the VC entropy [11] of \mathcal{H} .

It might seem simpler to use the following notion:

Definition 4.5 The *empirical weakness* of a learning algorithm A , denoted $\text{EmpWeak}_D(A)$, is given by

$$\text{EmpWeak}_D(A) = \liminf_{m \rightarrow \infty} \mathbf{E}_{S \sim D^m}(\text{Err}_S(f_S)).$$

We say that an algorithm A is *empirically weak* with respect to D if $\text{EmpWeak}_D(A) > 0$.

Since $f_S \in \mathcal{H}_m$, we have $\text{Err}_S(f_S) \geq \text{Err}_S(\mathcal{H}_m)$ for any S , and therefore

$$\text{EmpWeak}_D(A) \geq \text{Weak}_D(A). \tag{4}$$

Empirical weakness is not sufficient for our needs. When we use our weak learner with AdaBoost, we will change the weights on the training set. For example, suppose a typical training set contains 99 points labeled 1 and a single point labeled 0. Our weak learner might reasonably ignore the single outlier, in which case $\text{Err}_S(f_S) = 1/100$. However, if we give that single point half of the weight (which is exactly what AdaBoost would do in the second round), our weak learner might now get all 100 points correct, in which case $\text{Err}_S(f_p) = 0$. For our work in Section 5, we will need to use $\text{Weak}_D(A)$.

However, in some settings, the two notions are equivalent.

Proposition 4.6 *Suppose a learning algorithm A performs empirical risk minimization on \mathcal{H}_m . Then, for any D ,*

$$\text{Weak}_D(A) = \text{EmpWeak}_D(A).$$

Proof: For any S , $\text{Err}_S(f_S) = \text{Err}_S(\mathcal{H}_m)$. ■

Proposition 4.6 applies in two important cases:

1. A performs empirical risk minimization over a hypothesis class \mathcal{H}_∞ .
2. A performs regularization as follows: first choose a class \mathcal{H}_m depending on m , then do empirical risk minimization on \mathcal{H}_m . Note that this includes Vapnik's structural risk minimization [11], although we make no assumption about the VC dimension of \mathcal{H}_m .

Another natural notion of weakness is the following:

Definition 4.7 The *hypothesis weakness* of a learning algorithm A , denoted $\text{HypWeak}_D(A)$, is given by

$$\text{HypWeak}_D(A) = \text{Err}_D(\mathcal{H}_\infty),$$

the error rate with respect to D of the best possible classifier which could be generated by A .

Proposition 4.8 For any learning algorithm A ,

$$\text{Weak}_D(A) \leq \text{HypWeak}_D(A).$$

Proof: Let \hat{h} be a classifier in \mathcal{H}_∞ such that $\text{Err}_D(\hat{h}) = \text{HypWeak}_D(A)$. We note that $\hat{h} \in \mathcal{H}_N$ for some N ; hence, for $m \geq N$, we have $\hat{h} \in \mathcal{H}_m$. So, for $m \geq N$,

$$\mathbf{E}_{S \sim D^m}(\text{Err}_S(\mathcal{H}_m)) \leq \mathbf{E}_{S \sim D^m}(\text{Err}_S(\hat{h})) = \text{Err}_D(\hat{h}).$$

The result follows. ■

We also observe a connection between empirical weakness and hypothesis weakness:

Proposition 4.9 Suppose that the generalization error of A approaches 0. That is, suppose that, for any $\epsilon > 0$,

$$\lim_{m \rightarrow \infty} \Pr_{S \sim D^m} (|\text{Err}_S(f_S) - \text{Err}_D(f_S)| > \epsilon) = 0.$$

Then

$$\text{EmpWeak}_D(A) \geq \text{HypWeak}_D(A).$$

The proof is similar to that of Proposition 4.2.

Proof: Let E denote $\text{HypWeak}_D(A)$. For any S , we have $f_S \in \mathcal{H}_\infty$, so $\text{Err}_D(f_S) \geq E$. Hence, for any $\delta, \epsilon > 0$, we have, for sufficiently large m ,

$$\Pr_{S \sim D^m} (\text{Err}_S(f_S) \leq E - \epsilon) < \delta,$$

and therefore

$$\mathbf{E}_{S \sim D^m}(\text{Err}_S(f_S)) \geq (1 - \delta)(E - \epsilon)$$

for sufficiently large m , implying that

$$\text{EmpWeak}_D(A) \geq (1 - \delta)(E - \epsilon).$$

Since this is true for any $\delta, \epsilon > 0$, we must have $\text{EmpWeak}_D(A) \geq E$. ■

Note that, when \mathcal{H}_∞ has bounded VC dimension, the conclusion of Proposition 4.9 follows from Proposition 4.2, Inequality (4), and the fact that $\text{Weak}_D(A) \geq \text{Weak}_D(\mathcal{H}_\infty)$. However, Proposition 4.9 holds more generally. In particular:

Corollary 4.10 *Let A be a (β, δ) -stable algorithm, with $\beta = \lambda/m$ and $\delta \leq e^{-\Omega(m)}$. (Note that this includes $\delta = 0$.) Then*

$$\text{EmpWeak}_D(A) \geq \text{HypWeak}_D(A).$$

Proof: By Theorem 3.4, algorithm A meets the condition of Proposition 4.9. ■

Theorem 4.11 *Suppose we are in the noisy setting, with a Bayes error rate $\eta > 0$. Then any (β, δ) -stable learning algorithm A (where $\beta = \lambda/m$ and $\delta \leq e^{-\Omega(m)}$) is empirically weak.*

Moreover, if A is (β, δ) -stable as above, and A performs regularization by minimizing empirical risk over a class \mathcal{H}_m (e.g., A performs structural risk minimization), then A is weak. More precisely,

$$\text{Weak}_D(A) = \text{EmpWeak}_D(A) = \text{HypWeak}_D(A) \geq \eta.$$

This statement holds without any condition on the VC dimension of \mathcal{H}_m or \mathcal{H}_∞ .

Proof: We combine Propositions 4.6 and 4.8 and Corollary 4.10. We also note that

$$\text{HypWeak}_D(A) = \text{Err}_D(H_\infty) \geq \eta.$$

■

4.3 Implications of weakness

It is possible for a hypothesis class to be weak, but for $\text{Err}_S(\mathcal{H})$ to be zero for a particular S . (For example, it is possible that S contains m copies of the same point.) Lemma 4.12 states that this is unlikely. Note that Lemmas 4.12 and 4.13 apply to any hypothesis class \mathcal{H} , without any assumption on VC dimension, weakness, or stability.

Lemma 4.12 Let $\mu = \mathbf{E}_{S \sim D^m}(\text{Err}_S(\mathcal{H}))$. For any $\tau > 0$,

$$\Pr_{S \sim D^m}(\mu - \text{Err}_S(\mathcal{H}) \geq \tau) \leq \exp(-2m\tau^2/M^2).$$

The proof of Lemma 4.12 uses McDiarmid's method of bounded differences; we first show, in Lemma 4.13, that the method applies.

Lemma 4.13 For any $S \in \mathcal{Z}^m$, $i \in \{1, \dots, m\}$, and $u \in \mathcal{Z}$,

$$|\text{Err}_S(\mathcal{H}) - \text{Err}_{S^{i,u}}(\mathcal{H})| \leq \frac{M}{m}.$$

Recall that M is the maximum value of $c(h, z)$ for any classifier h and any $z \in \mathcal{Z}$.

Proof: Let S' denote $S^{i,u}$. Write $S = (z_1, \dots, z_m)$ and $S' = (z'_1, \dots, z'_m)$, where $z_j = z'_j$ for all $j \neq i$. Since S and S' are simply two sets which differ in the i th coordinate, we may assume without loss of generality that $\text{Err}_S(\mathcal{H}) \leq \text{Err}_{S'}(\mathcal{H})$.

Let \hat{h} be a classifier so that $\text{Err}_S(\mathcal{H}) = \text{Err}_S(\hat{h})$. Then

$$\text{Err}_{S'}(\hat{h}) = \frac{1}{m} \sum_{j=1}^m c(\hat{h}, z'_j) = \frac{1}{m} \left(\sum_{j \neq i} c(\hat{h}, z_j) \right) + \frac{1}{m} c(\hat{h}, z'_i) \leq \text{Err}_S(\hat{h}) + \frac{M}{m}.$$

Hence,

$$\text{Err}_S(\mathcal{H}) \leq \text{Err}_{S'}(\mathcal{H}) \leq \text{Err}_{S'}(\hat{h}) \leq \text{Err}_S(\hat{h}) + \frac{M}{m} = \text{Err}_S(\mathcal{H}) + \frac{M}{m}.$$

This proves that

$$|\text{Err}_S(\mathcal{H}) - \text{Err}_{S'}(\mathcal{H})| \leq \frac{M}{m}. \quad \blacksquare$$

We can now prove Lemma 4.12.

Proof of Lemma 4.12: Define $g: \mathcal{Z}^m \rightarrow \mathbb{R}$ by $g(S) = -\text{Err}_S(\mathcal{H})$. By Lemma 4.13, $g(S)$ satisfies the conditions of Lemma 3.1 with $c_k = M/m$. We have $\sum c_k^2 = M^2/m$. Hence, for any $\tau > 0$,

$$\Pr_{S \sim D^m}(-\text{Err}_S(\mathcal{H}) + \mu \geq \tau) \leq \exp(-2m\tau^2/M^2). \quad \blacksquare$$

In particular,

Corollary 4.14 *Let A be a weak learning algorithm, and let $\epsilon_* = \text{Weak}_D(A)/2$. Then, for sufficiently large m ,*

$$\Pr_{S \sim D^m} (\text{Err}_S(\mathcal{H}_m) \leq \epsilon_* + \frac{M}{m}) \leq \exp(-m\epsilon_*^2/2M^2).$$

Proof: For notational simplicity, let E_m denote $\mathbf{E}_{S \sim D^m}(\text{Err}_S(\mathcal{H}_m))$. Let $r = 3 \text{Weak}_D(A)/4 = 3\epsilon_*/2$. Then, by Lemma 4.12,

$$\Pr_{S \sim D^m} (E_m - \text{Err}_S(\mathcal{H}_m) \geq \epsilon_*/2) \leq \exp(-2m(\epsilon_*/2)^2/M^2).$$

and therefore

$$\Pr_{S \sim D^m} (\text{Err}_S(\mathcal{H}_m) \leq E_m - \epsilon_*/2) \leq \exp(-m\epsilon_*^2/2M^2).$$

By our definition of $\text{Weak}_D(A)$, $\liminf_{m \rightarrow \infty} E_m = 2\epsilon_*$, so, for sufficiently large m , $E_m > 7\epsilon_*/4$. Also, for sufficiently large m , $M/m < \epsilon_*/4$. We conclude that, for sufficiently large m ,

$$E_m - \epsilon_*/2 > \frac{5}{4}\epsilon_* > \epsilon_* + \frac{M}{m}.$$

Hence,

$$\begin{aligned} \Pr_{S \sim D^m} (\text{Err}_S(\mathcal{H}_m) \leq \epsilon_* + \frac{M}{m}) &\leq \Pr_{S \sim D^m} (\text{Err}_S(\mathcal{H}_m) \leq E_m - \epsilon_*/2) \\ &\leq \exp(-m\epsilon_*^2/2M^2). \end{aligned}$$

■

Corollary 4.14 implies the following theorem:

Theorem 4.15 *Let A be a weak learning algorithm, and let $\epsilon_* = \text{Weak}_D(A)/2$. Let $B_m \subset \mathcal{Z}^m$ denote the set of all S such that either*

$$\text{Err}_S(\mathcal{H}_m) \leq \epsilon_*$$

or

$$\exists i, u \quad \text{Err}_{S^{i,u}}(\mathcal{H}_m) \leq \epsilon_*.$$

Then, for sufficiently large m ,

$$\Pr_{S \sim D^m} (S \in B_m) \leq \exp(-m\epsilon_*^2/2M^2)$$

Proof: Suppose that $S \in B_m$. One possibility is that $\text{Err}_S(\mathcal{H}_m) \leq \epsilon_*$. Otherwise, we must have some i, u such that $\text{Err}_{S^{i,u}}(\mathcal{H}_m) \leq \epsilon_*$. In this case, by Lemma 4.13, $\text{Err}_S(\mathcal{H}_m) \leq \epsilon_* + M/m$.

Therefore, in either case, if $S \in B_m$, $\text{Err}_S(\mathcal{H}_m) \leq \epsilon_* + M/m$. The result now follows from Corollary 4.14. \blacksquare

5 The algorithmic stability of AdaBoost

In this section, we show that AdaBoost is almost-everywhere stable. In Section 5.1, we define AdaBoost. In Section 5.2, we prove some technical lemmas. Finally, in Section 5.3, we prove our main theorem.

5.1 Definition of AdaBoost

We now give a definition of AdaBoost. We follow the description of Freund and Schapire [5], who introduced the algorithm.

The input is a set of m examples $z_i = (x_i, y_i)$. We are given an initial distribution p^1 on the input points; let $w^1 = p^1$, and let $Z_1 = 1$. Let T denote the number of rounds we wish to perform.

For each $t = 1, \dots, T$:

1. Call the weak learner with distribution p^t . The weak learner returns a hypothesis $h_t : \mathcal{X} \rightarrow [0, 1]$.
2. For each i , let $a_i^t = |h_t(x_i) - y_i|$, the error of h_t on instance i .
3. Let $\epsilon_t = \sum_{i=1}^m p_i^t a_i^t$, the error rate of h_t with respect to p^t (i.e., $\text{Err}_{p^t}(h_t)$).
4. Let $\beta_t = \epsilon_t / (1 - \epsilon_t)$, and let $\alpha_t = \ln(1/\beta_t)$.
5. Reweight the data: for all i , let $w_i^{t+1} = w_i^t \beta_t^{1-a_i^t}$.
6. Normalize the distribution: let $Z_{t+1} = \sum_{i=1}^m w_i^t$, and $p_i^{t+1} = w_i^{t+1} / Z_{t+1}$.
7. Let $G_t(x) = \sum_{s=1}^t \alpha_s h_s(x)$.

The output of the algorithm is a classifier H_T constructed from G_T (see Section 5.1.1).

Note that we allow our hypotheses h_i to take values in $[0, 1]$. For such continuous hypotheses, the choice of β_t and α_t above may not be optimal; see

Schapire and Singer [10] for a discussion of this question. However, since we need a concrete definition for our analysis, we stick with the original choices of β_t and α_t described above.

5.1.1 Constructing the final hypothesis

The remaining question is how to construct H_T from G_T . Our goal is that H_T be a function from $\mathcal{X} \rightarrow [0, 1]$. We discuss three alternatives.

The simplest approach is to threshold G_T at $\frac{1}{2} \sum_{t=1}^T \alpha_t$:

$$H_T^{\text{sign}}(x) = \text{sign} \left(G_T(x) - \frac{1}{2} \sum_{t=1}^T \alpha_t \right)$$

This is correct if we want our output to be a $\{0, 1\}$ -classifier. However, such a discrete-valued function can never be stable unless it is constant, so we cannot use this definition.

Another approach is to normalize G_T :

$$H_T^{\text{norm}}(x) = \frac{G_T(x)}{\sum_{t=1}^T \alpha_t}.$$

This definition is used, for example, in the margin analysis of Schapire et al. [9]. However, this normalization would lead to certain technical difficulties for us when $\sum \alpha_t$ is very small. We return to this issue in Note 5.7.

The third approach, and the one we will take, is to first construct a continuous function $\sigma: \mathbb{R} \rightarrow [0, 1]$, which forces the output into the desired range. We then define

$$H_T(x) = \sigma \left(G_T(x) - \frac{1}{2} \sum_{t=1}^T \alpha_t \right).$$

For simplicity's sake, we use the function:

$$\sigma(v) = \begin{cases} 0 & \text{if } v < -1/2 \\ v + 1/2 & \text{if } -1/2 < v < 1/2 \\ 1 & \text{if } v > 1/2 \end{cases}$$

This choice of σ has the advantage that, for all $v, w \in \mathbb{R}$, $|\sigma(v) - \sigma(w)| \leq |v - w|$.

We could replace σ with any function satisfying a Lipschitz condition; this would change the bound in Theorem 5.8 by a constant factor. See Note 5.6 for details.

5.2 Bounding the behavior of AdaBoost

In this section, we prove some lemmas bounding the various quantities used by AdaBoost.

Lemma 5.1 *For any $\eta > 1$, and any $t \in [0, 1]$,*

$$\eta^t - 1 - t(\eta - 1) \leq 0.$$

Proof: Let $\psi(t) = \eta^t - 1 - t(\eta - 1)$. Clearly $\psi(0) = \psi(1) = 0$.

Since $\psi'(t) = \eta^t \ln \eta - (\eta - 1)$, the equation $\psi'(t) = 0$ has only one solution. This solution must thus lie between 0 and 1, and we must have either $\psi(t) < 0$ for all $t \in (0, 1)$ or $\psi(t) > 0$ for all $t \in (0, 1)$.

Finally, we observe that

$$\psi\left(\frac{1}{2}\right) = \sqrt{\eta} - \frac{1}{2} - \frac{1}{2}\eta = -\frac{1}{2}(\sqrt{\eta} - 1)^2 < 0.$$

■

Lemma 5.2 *Let m be a positive integer, and $\epsilon \in (0, \frac{1}{2}]$. Let $\eta = (1 - \epsilon)/\epsilon$. Suppose that we are given p_1, \dots, p_m where $p_i > 0$ and $\sum_i p_i = 1$. Then, for any a_1, \dots, a_m with $a_i \in [0, 1]$ and $\sum_i p_i a_i = \epsilon$, let*

$$\phi(a_1, \dots, a_m) = \sum_{i=1}^m p_i \eta^{a_i}.$$

Then $1 \leq \phi(a_1, \dots, a_m) \leq 2(1 - \epsilon)$.

Proof: The lower bound on ϕ is immediate from $\eta > 1$ and $\sum_i p_i = 1$. We now prove the upper bound.

Since the space of possible $\{a_i\}$ is compact, the function $\phi(a_1, \dots, a_m)$ attains some maximum value. We first show that, at the maximum, all a_i (except possibly one) are equal to 0 or 1.

Suppose for a contradiction that, at the point this maximum is attained, we have a_i and a_j both in $(0, 1)$ for some i, j . Let $\kappa = p_i a_i + p_j a_j$, and consider the following function of x :

$$g(x) = p_i \eta^x + p_j \eta^{\frac{(\kappa - p_i x)}{p_j}}.$$

The domain of g is all x such that $x \in [0, 1]$ and $\frac{(\kappa - p_i x)}{p_j} \in [0, 1]$, or, equivalently, $x \geq \max\{0, (\kappa - p_j)/p_i\}$ and $x \leq \min\{1, \kappa/p_i\}$.

Since a_1, \dots, a_m is a maximum of ϕ , the function $g(x)$ must attain its maximum at a_i ; since both a_i and a_j are in $(0, 1)$, this means that $g(x)$ attains its maximum on the interior of its domain.

However, we note that

$$\begin{aligned} \frac{d}{dx}g(x) &= p_i \eta^x \ln \eta + p_j \eta^{\frac{(\kappa - p_i x)}{p_j}} \ln \eta \left(\frac{-p_i}{p_j} \right) \\ &= p_i \ln \eta \left(\eta^x - \eta^{\frac{(\kappa - p_i x)}{p_j}} \right). \end{aligned}$$

So $g'(x)$ is zero only at $x_0 = \kappa/(p_i + p_j)$. Since $\eta > 1$, we know that $g'(x)$ is negative to the left of x_0 and positive to the right, so $g(x)$ attains a minimum at x_0 . Hence $g(x)$ must attain its maximum on an endpoint of its domain. We have reached a contradiction, so we conclude that, at the maximum value of ϕ , all a_i (except possibly one) are equal to 0 or 1.

Now, let a_1, \dots, a_m be a point at which ϕ attains its maximum. Assume that, for $j \neq i$, $a_j \in \{0, 1\}$. Let U be the sum of p_j over all $j \neq i$ with $a_j = 0$, and V be the sum of p_j over all $j \neq i$ with $a_j = 1$. Then we have

$$\begin{aligned} p_i + U + V &= 1 \\ p_i a_i + V &= \epsilon \\ \phi(a_1, \dots, a_m) &= p_i \eta^{a_i} + U + V \eta \\ &= p_i \eta^{a_i} + (1 - V - p_i) + V \eta \\ &= p_i (\eta^{a_i} - 1) + 1 + (\epsilon - p_i a_i) (\eta - 1) \\ &= p_i (\eta^{a_i} - 1 - a_i (\eta - 1)) + 1 + \epsilon \eta - \epsilon \\ &\leq 2(1 - \epsilon) \end{aligned}$$

where the last inequality follows from Lemma 5.1 and from $\eta = (1 - \epsilon)/\epsilon$.

Since $\phi(a_1, \dots, a_m) \leq 2(1 - \epsilon)$ at the maximum value of ϕ , we have proven the desired result. ■

Lemma 5.3 *Suppose we run AdaBoost for one round starting from two distributions p, p' . For the run starting from $p^1 = p$, we define $h, a_i, \epsilon, \beta, \alpha$ as in Section 5.1, all for $t = 1$. We write w_i for w_i^2 , Z for Z_2 , and q for p^2 . We define $h', a'_i, \epsilon', \beta', \alpha', w', Z', q'$ similarly starting from p' .*

Suppose the weak learner has L_1 -stability λ . Further suppose that

$$\text{Err}_p(\mathcal{H}_m), \text{Err}_{p'}(\mathcal{H}_m) \geq \epsilon_*.$$

Then, for all $x \in \mathcal{X}$, the following inequalities hold:

$$|h(x) - h'(x)| \leq \lambda \|p - p'\| \quad (5)$$

$$|\epsilon - \epsilon'| \leq (\lambda + 1) \|p - p'\| \quad (6)$$

$$|\alpha - \alpha'| \leq 2 \frac{\lambda + 1}{\epsilon_*} \|p - p'\| \quad (7)$$

$$|\alpha h(x) - \alpha' h'(x)| \leq 4 \frac{\lambda + 1}{\epsilon_*} \|p - p'\| \quad (8)$$

$$\sum_{i=1}^m |w_i - w'_i| \leq 2 \frac{\lambda + 1}{\epsilon_*} \|p - p'\| \quad (9)$$

$$|Z - Z'| \leq 2 \frac{\lambda + 1}{\epsilon_*} \|p - p'\| \quad (10)$$

$$\epsilon \leq Z \leq 2\epsilon \quad (11)$$

$$\|q - q'\| \leq 4 \frac{\lambda + 1}{\epsilon_*^2} \|p - p'\| \quad (12)$$

$$\text{Err}_q(\mathcal{H}_m), \text{Err}_{q'}(\mathcal{H}_m) \geq \epsilon_*/2. \quad (13)$$

Proof: We let I denote the interval $[\epsilon_*, 1/2]$; thus $\epsilon, \epsilon' \in I$.

We now prove each of the claims in the lemma.

Proof of (5): From the stability of the weak learner, we have, for any $x \in \mathcal{X}$,

$$|c(h, (x, 0)) - c(h', (x, 0))| \leq \lambda \|p - p'\|.$$

We note that $c(h, (x, 0)) = |h(x) - 0| = h(x)$, and similarly $c(h', (x, 0)) = h'(x)$. So,

$$|h(x) - h'(x)| \leq \lambda \|p - p'\|.$$

□

Proof of (6): Recall that $a_i = |h(x_i) - y_i|$, where $y_i \in \{0, 1\}$ and $h(x_i) \in [0, 1]$. So

$$|a_i - a'_i| = ||h(x_i) - y_i| - |h'(x_i) - y_i|| = |h(x_i) - h'(x_i)|. \quad (14)$$

Therefore,

$$\begin{aligned}
|\epsilon - \epsilon'| &= \left| \sum_i p_i a_i - \sum_i p'_i a'_i \right| \\
&\leq \left| \sum_i p_i (a_i - a'_i) \right| + \left| \sum_i a'_i (p_i - p'_i) \right| \\
&\leq \sum_i p_i |h(x_i) - h'(x_i)| + \sum_i |p_i - p'_i| \\
&\leq \lambda \|p - p'\| + \|p - p'\| = (\lambda + 1) \|p - p'\|.
\end{aligned}$$

□

Proof of (7): Recall that $\alpha = \ln \frac{1-\epsilon}{\epsilon}$.

$$\begin{aligned}
|\alpha - \alpha'| &= \left| \ln \frac{1-\epsilon}{\epsilon} - \ln \frac{1-\epsilon'}{\epsilon'} \right| \\
&\leq |\epsilon - \epsilon'| \max_{x \in I} \left| \frac{d}{dx} \ln \frac{1-x}{x} \right| \\
&= |\epsilon - \epsilon'| \max_{x \in I} \left| -\frac{1}{x(1-x)} \right| \\
&< |\epsilon - \epsilon'| \frac{2}{\epsilon_*}.
\end{aligned}$$

Using (6), we get

$$|\alpha - \alpha'| \leq 2 \frac{\lambda + 1}{\epsilon_*} \|p - p'\|.$$

□

Proof of (8): Combining (7) and (5), we have, for any $x \in \mathcal{X}$,

$$\begin{aligned}
|\alpha h(x) - \alpha' h'(x)| &\leq |\alpha h(x) - \alpha h'(x)| + |\alpha h'(x) - \alpha' h'(x)| \\
&= \alpha |h(x) - h'(x)| + h'(x) |\alpha - \alpha'| \\
&< \ln \frac{2}{\epsilon_*} \lambda \|p - p'\| + 2 \frac{\lambda + 1}{\epsilon_*} \|p - p'\| \\
&< 4 \frac{\lambda + 1}{\epsilon_*} \|p - p'\|.
\end{aligned}$$

□

Proof of (9): Recall that, for any i , we have

$$w_i = D(x_i)\beta^{1-a_i}.$$

Let b_i denote $1 - a_i$, and b'_i denote $1 - a'_i$. By (14),

$$|b_i - b'_i| = |a_i - a'_i| = |h(x_i) - h'(x_i)|. \quad (15)$$

For any i ,

$$\begin{aligned} |w_i - w'_i| &= |p_i\beta^{b_i} - p'_i\beta^{b'_i}| \\ &\leq |p_i - p'_i|\beta^{b_i} + p'_i|\beta^{b_i} - \beta^{b'_i}| + p'_i|\beta^{b_i} - \beta^{b'_i}|. \end{aligned} \quad (16)$$

We now bound each of the three terms in (16).

First, since $\beta < 1$ and $b_i \in [0, 1]$,

$$|p_i - p'_i|\beta^{b_i} \leq |p_i - p'_i|. \quad (17)$$

Second we note that

$$\begin{aligned} |\beta^{b_i} - \beta^{b'_i}| &= \left| \left(\frac{\epsilon}{1-\epsilon} \right)^{b_i} - \left(\frac{\epsilon'}{1-\epsilon'} \right)^{b_i} \right| \\ &\leq |\epsilon - \epsilon'| \max_{x \in I} \left| \frac{d}{dx} \left(\frac{x}{1-x} \right)^{b_i} \right| \\ &= |\epsilon - \epsilon'| \max_{x \in I} \left| \frac{b_i}{(1-x)^2} \left(\frac{x}{1-x} \right)^{b_i-1} \right| \\ &\leq |\epsilon - \epsilon'| \max_{x \in I} |b_i x^{b_i-1}| \\ &\leq |\epsilon - \epsilon'| b_i \epsilon_*^{b_i-1}. \end{aligned}$$

Since $b_i \leq 1$, $b_i \epsilon_*^{b_i-1} \leq 1/\epsilon_*$. This, combined with (6), gives

$$p'_i|\beta^{b_i} - \beta^{b'_i}| \leq p'_i \frac{\lambda + 1}{\epsilon_*} \|p - p'\|. \quad (18)$$

Third, we use (15) to see that

$$\begin{aligned} |\beta^{b_i} - \beta^{b'_i}| &\leq |b_i - b'_i| \max_{x \in [0,1]} \left| \frac{d}{dx} \beta^{tx} \right| \\ &\leq |h(x_i) - h'(x_i)| \max_{x \in [0,1]} x \beta^{tx-1} \\ &\leq |h(x_i) - h'(x_i)| \frac{1}{\beta}. \end{aligned}$$

By (5), and since $\beta' \geq \epsilon' \geq \epsilon_*$,

$$p'_i |\beta^{b_i} - \beta'^{b'_i}| \leq p'_i \frac{\lambda}{\epsilon_*} \|p - p'\|. \quad (19)$$

We now plug (17), (18), (19) into (16), yielding

$$|w_i - w'_i| \leq |p_i - p'_i| + p'_i \frac{2\lambda + 1}{\epsilon_*} \|p - p'\|.$$

Summing over all i , we get

$$\begin{aligned} \sum_{i=1}^m |w_i - w'_i| &\leq \sum_{i=1}^m |p_i - p'_i| + \sum_{i=1}^m p'_i \frac{2\lambda + 1}{\epsilon_*} \|p - p'\| \\ &= \|p - p'\| + \frac{2\lambda + 1}{\epsilon_*} \|p - p'\| < 2 \frac{\lambda + 1}{\epsilon_*} \|p - p'\|. \end{aligned}$$

□

Proof of (10): We note that

$$|Z - Z'| = \left| \sum_{i=1}^m w_i - \sum_{i=1}^m w'_i \right| \leq \sum_{i=1}^m |w_i - w'_i| < 2 \frac{\lambda + 1}{\epsilon_*} \|p - p'\|.$$

□

Proof of (11): We observe that

$$Z = \sum_{i=1}^m w_i = \sum_{i=1}^m p_i \beta^{1-a_i} = \beta \sum_{i=1}^m p_i (1/\beta)^{a_i}.$$

By Lemma 5.2,

$$1 \leq \sum_{i=1}^m p_i (1/\beta)^{a_i} \leq 2(1 - \epsilon).$$

Thus,

$$\epsilon \leq \beta \leq Z \leq 2(1 - \epsilon)\beta = 2\epsilon.$$

□

Proof of (12): Using (9), (10), and the lower bound from (11), we get

$$\begin{aligned}
\|q - q'\| &= \sum_{i=1}^m |q_i - q'_i| = \sum_{i=1}^m \left| \frac{w_i}{Z} - \frac{w'_i}{Z'} \right| \\
&\leq \sum_{i=1}^m \left| \frac{w_i - w'_i}{Z} \right| + \sum_{i=1}^m \left| \frac{w'_i}{Z} - \frac{w'_i}{Z'} \right| \\
&\leq \frac{1}{Z} \sum_{i=1}^m |w_i - w'_i| + Z' \left| \frac{1}{Z} - \frac{1}{Z'} \right| \\
&= \frac{1}{Z} \left(\sum_{i=1}^m |w_i - w'_i| \right) + |Z' - Z| \\
&< 4 \frac{\lambda + 1}{\epsilon_*^2} \|p - p'\|.
\end{aligned}$$

□

Proof of (13):

By the upper bound from (11), we have, for any i ,

$$q_i = \frac{w_i}{Z} = \frac{p_i \beta^{1-a_i}}{Z} \geq \frac{p_i \beta}{2\epsilon} = \frac{p_i}{2(1-\epsilon)} \geq \frac{p_i}{2}.$$

Hence, for any $g \in \mathcal{H}_m$,

$$\text{Err}_q(g) = \sum_{i=1}^m q_i |y_i - g(x_i)| \geq \frac{1}{2} \sum_{i=1}^m p_i |y_i - g(x_i)| = \frac{\text{Err}_p(g)}{2} \geq \frac{\epsilon_*}{2}.$$

So $\text{Err}_q(\mathcal{H}_m) \geq \epsilon_*/2$. Similarly, $\text{Err}_{q'}(\mathcal{H}_m) \geq \epsilon_*/2$. □

This concludes the lemma. ■

Lemma 5.4 *Suppose we run AdaBoost for T rounds, starting from two distributions p and p' on the training set S , yielding two sums $G_T = \sum_{t=1}^T \alpha_t h_t$ and $G'_T = \sum_{t=1}^T \alpha'_t h'_t$. Suppose the weak learner has L_1 -stability λ . Suppose further that $\text{Err}_p(\mathcal{H}_m), \text{Err}_{p'}(\mathcal{H}_m)$ are at least ϵ_* . Then, for all $x \in \mathcal{X}$,*

$$|G_T(x) - G'_T(x)| \leq c_T \|p - p'\|, \quad (20)$$

where the constant c_T is

$$c_T(\lambda, \epsilon_*) = \sum_{t=1}^T \frac{2^{t^2+1}(\lambda + 1)^t}{\epsilon_*^{2t-1}}.$$

Moreover, for all $x \in \mathcal{X}$,

$$|H_T(x) - H'_T(x)| \leq c_T \|p - p'\|. \quad (21)$$

Proof: We proceed by induction on T .

When $T = 1$, we have, by (8) from Lemma 5.3,

$$|G_1(x) - G'_1(x)| = |\alpha_1 h_1(x) - \alpha'_1 h'_1(x)| \leq 4 \frac{\lambda + 1}{\epsilon_*} \|p - p'\|,$$

which is exactly c_1 .

Choose some $T > 1$, and suppose the theorem holds for $T > 1$.

Let q, q' be the distributions at the end of the first round after starting from p, p' respectively. By (13) from Lemma 5.3,

$$\text{Err}_q(\mathcal{H}_m), \text{Err}_{q'}(\mathcal{H}_m) \geq \epsilon_*/2.$$

Let $F(x) = \sum_{t=2}^T \alpha_t(x) h_t(x)$, and $F'(x) = \sum_{t=2}^T \alpha'_t(x) h'_t(x)$. By the inductive assumption, for any x ,

$$|F(x) - F'(x)| \leq c_{T-1}(\lambda, \epsilon_*/2) \|q - q'\|,$$

so, using (8) and (12) from Lemma 5.3,

$$\begin{aligned} |G_T(x) - G'_T(x)| &= |\alpha_1 h_1(x) + F(x) - \alpha'_1 h'_1(x) - F'(x)| \\ &\leq |\alpha_1 h_1(x) - \alpha'_1 h'_1(x)| + |F(x) - F'(x)| \\ &\leq 4 \frac{\lambda + 1}{\epsilon_*} \|p - p'\| + c_{T-1}(\lambda, \epsilon_*/2) \|q - q'\| \\ &\leq \left(4 \frac{\lambda + 1}{\epsilon_*} + 4c_{T-1}(\lambda, \epsilon_*/2) \frac{\lambda + 1}{\epsilon_*^2} \right) \|p - p'\|. \end{aligned}$$

A straightforward calculation reveals that

$$4 \frac{\lambda + 1}{\epsilon_*} + 4c_{T-1}(\lambda, \epsilon_*/2) \frac{\lambda + 1}{\epsilon_*^2} = c_T(\lambda, \epsilon),$$

which concludes the proof of (20).

To prove (21), we simply observe that, by our choice of σ ,

$$|H_T(x) - H'_T(x)| = |\sigma(G_T(x)) - \sigma(G'_T(x))| \leq |G_T(x) - G'_T(x)| \leq c_T \|p - p'\|. \quad \blacksquare$$

Note 5.5 We could, if we wish, simplify the expression for $c_T(\lambda, \epsilon_*)$. The ratio between the t th and $(t + 1)$ th terms in the sum is

$$\frac{2^{2t+1}(\lambda + 1)}{\epsilon_*^2} > 2,$$

and therefore the entire sum is at most twice the largest term:

$$c_T(\lambda, \epsilon_*) \leq \frac{2^{T^2+2}(\lambda + 1)^T}{\epsilon_*^{2T-1}}.$$

Note 5.6 As we remarked in Section 5.1.1, we could replace our function σ by any function satisfying a Lipschitz condition with constant Λ . We would then conclude that

$$|H_T(x) - H'_T(x)| \leq \Lambda c_T \|p - p'\|.$$

Note 5.7 If we instead construct $H_T(x)$ by normalizing $G_T(x)$ (see the definition of H_T^{norm} in Section 5.1.1), we have more difficulty. Using the methods of this section, it is not hard to prove a bound of the form

$$|H_T^{\text{norm}}(x) - H'^{\text{norm}}_T(x)| \leq \frac{c'_T}{\sum_t \alpha_t} \|p - p'\|,$$

where $c'_T = \frac{3}{2}c_T$. So, if we assume that $\sum_t \alpha_t \geq A$ for some constant A , we get an inequality similar to (21). However, we see no reason why such an A should exist. In the case where ϵ_t is guaranteed to be at most $1/2 - \gamma$ for some $\gamma > 0$, even for one round, we can take $A = \ln \frac{1+2\gamma}{1-2\gamma}$, but we would prefer not to introduce such an assumption simply for one technical point in the proof.

5.3 Main theorem

We are now ready to prove our main theorem.

Theorem 5.8 *Suppose the weak learner A has L_1 -stability λ , and let $\epsilon_* = \text{Weak}_D(A)/2 > 0$. Then, for sufficiently large m , for all T , AdaBoost for T rounds is (β, δ) -stable, where*

$$\beta = \frac{2}{m} \sum_{t=1}^T \frac{2^{t^2+1}(\lambda + 1)^t}{\epsilon_*^{2t-1}}$$

$$\delta = \exp(-m\epsilon_*^2/2).$$

Moreover, let g_S denote the output of T rounds of AdaBoost on input S . Then, for any $\tau > 0$,

$$\Pr_{S \in \mathcal{D}^m} (|\text{Err}_S(g_S) - \text{Err}_D(g_S)| > \tau + \beta + \delta) \leq \exp(-c_1 \tau^2 m) + c_2 m^2 \exp(-m \epsilon_*^2 / 2),$$

where c_1, c_2 depend on T, λ, ϵ_* .

Proof: Let $B_m \subset \mathcal{Z}^m$ denote the set of “bad” S : more precisely, B_m is the set of all S such that either

$$\text{Err}_S(\mathcal{H}_m) \leq \epsilon_*$$

or

$$\exists i, u \quad \text{Err}_{S^{i,u}}(\mathcal{H}_m) \leq \epsilon_*.$$

By Theorem 4.15, for sufficiently large m , $\Pr(x \in B_m) \leq \delta$.

If $S \notin B_m$, then, for any i and u , $\text{Err}_S(\mathcal{H}_m)$ and $\text{Err}_{S^{i,u}}(\mathcal{H}_m)$ are both at least ϵ_* . Let H_T be the output of T rounds of AdaBoost starting from S , and let H'_T be the output starting from $S^{i,u}$. Then, by Lemma 5.4,

$$|H_T(x) - H'_T(x)| \leq c_T \frac{2}{m},$$

where c_T is the constant from Lemma 5.4. We can conclude that AdaBoost is β -stable at S whenever $S \notin B_m$. This proves the first claim.

The second claim now follows directly from Theorem 3.4. ■

Theorem 5.8 immediately implies the following:

Corollary 5.9 *Suppose that our learner is weak (in the sense of Definition 4.4) and L_1 -stable. Let g_S denote the output of T rounds of AdaBoost on input S . Then, for sufficiently small $\epsilon > 0$, for sufficiently large m ,*

$$\Pr(|\text{Err}_S(g_S) - \text{Err}_D(g_S)| > \epsilon) \leq \exp(-C \epsilon^2 m)$$

for some constant C . The constant C depends on T and on the weakness and L_1 -stability of the learner.

Proof: We use the notation of Theorem 5.8. Given $\epsilon > 0$, we choose m sufficiently large so that $\beta < \epsilon/3$. For sufficiently large m , we have $\delta < \beta$. Let $\tau = \epsilon/3$. Then, by Theorem 5.8,

$$\Pr(|\text{Err}_S(g_S) - \text{Err}_D(g_S)| > \epsilon) \leq \exp(-c_1 \epsilon^2 m / 9) + c_2 m^2 \exp(-m \epsilon_*^2 / 2).$$

As long as $c_1\epsilon^2/9 < \epsilon_*^2/2$, we have

$$c_2m^2 \exp(-m\epsilon_*^2/2) < \exp(-c_1\epsilon^2m/9)$$

for sufficiently large m . The result follows. ■

6 Conclusions

We believe that the notion of algorithmic stability is an important one, and may give us new insights into the performance of learning algorithms. An understanding of which boosting algorithms preserve stability may help us decide which boosting scheme to use with a given weak learner. We also hope that the notion of almost-everywhere stability can be used in other contexts to prove new generalization bounds.

One open question is whether the dependence on T in Theorem 5.8 can be improved. Our stability constant increases with 2^{T^2} . The VC theoretic bounds are linear in T . It seems possible that our inductive approach could be improved to yield a bound of the form $2^{O(T)}$, but a further improvement may require a significantly different technique.

References

- [1] O. Bousquet and A. Elisseeff. Algorithmic stability and generalization performance. In *Advances in Neural Information Processing Systems 13: Proc. NIPS'2000*, 2001.
- [2] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
- [3] L. Breiman. Heuristics of instability and stabilization in model selection. *Annals of Statistics*, 24(6):2350–2383, 1996.
- [4] L. P. Devroye and T. J. Wagner. Distribution-free performance bounds for potential function rules. *IEEE Trans. Inform. Theory*, 25(5):601–604, 1979.
- [5] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proc. of the Second European Conference on Computational Learning Theory*. LNCS, March 1995.

- [6] M. Kearns and D. Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural Computation*, 11:1427–1453, 1999.
- [7] C. McDiarmid. On the method of bounded differences. In *Surveys in combinatorics, 1989 (Norwich, 1989)*, pages 148–188. Cambridge Univ. Press, Cambridge, 1989.
- [8] C. McDiarmid. Concentration. In *Probabilistic methods for algorithmic discrete mathematics*, pages 195–248. Springer, Berlin, 1998.
- [9] R. 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.
- [10] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
- [11] V. N. Vapnik. *Statistical learning theory*. John Wiley & Sons Inc., New York, 1998. A Wiley-Interscience Publication.