

An Empirical Study of Training and Testing Error in Boosting

David D. Latham

June 1, 2001

Senior Honors Thesis  
Advised By Javed A. Aslam

Abstract

Bounds have been proven for both training and testing error for the boosting algorithm AdaBoost, but in practice neither seem to produce a particularly tight bound. In this paper we share some observations of these bounds from empirical results, and then explore some properties of the algorithm with an eye towards finding an improved bound for the performance of AdaBoost. Based on our empirical evidence, the error of a hypothesis which labels examples probabilistically based upon the confidence of the vote of the weak hypotheses forms a tighter bound for the training error.

I – Introduction

Boosting is a method of using a weak learning algorithm to generate a set of hypotheses and combining them to form a stronger hypothesis. In the PAC learning model, a weak learning algorithm is one which will produce a hypothesis which correctly classifies instances at least a little bit better than half the time. In essence, a weak learner is something which produces a result which has learned something about a target concept. A strong learner can produce a hypothesis which will probably classify an instance correctly with an arbitrary degree of accuracy. Somewhat surprisingly, it has been shown

that given a weak learning algorithm for a target concept, one can produce a strong learning algorithm by boosting. [1]

In the learning model, a learning algorithm has access to an *example oracle* which provides examples drawn independently at random according to some fixed distribution. Examples are a pairing of an instance with a label for that instance. In practice, we have sets of labeled examples as data, some of which are given to the learning algorithm for I to learn. These we call training examples. The remaining examples we use to test the result of the learning algorithm. We call these testing examples. By calculating the error rate of a hypothesis with respect to the training and testing examples, we determine a measure of how well the algorithm has learned the training examples, and how well that hypothesis generalizes to data it has not seen before. These measures we call training error and testing error respectively. Sometimes testing error is referred to as generalization error.

It is simple to produce a hypothesis with low training error. Simply memorize the training examples, and randomly assign a label to any other instance encountered. However, this clearly will have a testing error no better than random guessing. The goal of learning is to produce a hypothesis which generalizes well and hence will have a low testing error.

Some analysis has been done with respect to bounds for the training and testing error for boosting algorithms, particularly for AdaBoost, the algorithm which we study. [2] In this paper, we examine the results of an empirical analysis of some of these bounds and explore some other ideas for possible bounds or related figures from the AdaBoost algorithm.

## II – Background

Probably Approximately Correct (or PAC) learning is a model for learning a concept. [3] A concept is simply some mapping from an instance space to a label space. For example, a concept of recognizing written characters is a mapping between a visual pattern of some possible scribble to a letter of the alphabet. Similarly, mapping a given document as a pattern of words to the label of whether it is or is not relevant to a certain topic is a concept. In the PAC learning model, given some instance space  $X$ , and some label space  $Y$ , an example is some  $(x, y)$ , where  $x \in X$  and  $y \in Y$  according to some concept. A learning algorithm is given access to an example oracle which will supply it with examples by drawing them independently at random according to some unknown fixed distribution  $D$ . A strong learning algorithm can produce a hypothesis  $h : X \rightarrow Y$  which will, with probability at least  $1 - \delta$  have error at most  $\epsilon$ . A weak learner can produce a hypothesis which, instead of satisfying  $\epsilon$  must have error a little bit better than one half.<sup>1</sup>

AdaBoost, a boosting algorithm by Freund and Schapire, [4] works by iteratively training a series of weak learners, altering the distribution each round to more heavily weight the examples which the previous learner classifies incorrectly. Here is a version of AdaBoost designed to learn a binary concept (i.e. label space of  $\{-1, +1\}$ ):

---

<sup>1</sup> For a weak learning algorithm the error of the hypothesis must in fact be

$$\frac{1}{2} - \frac{1}{p}$$

where  $p$  is some polynomial of the length of the instances and the size of the target concept.

AdaBoost :

Given:  $(x_1, y_1), \dots, (x_m, y_m)$   $x_i \in X, y_i \in \{-1, +1\}$

Initialize  $D_1(i) = 1/m$  for all  $i$ .

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

- Train weak learner using distribution  $D_t$ .
- Get weak hypothesis  $h_t : X \rightarrow \{-1, +1\}$ .
- Choose  $\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)$ ,

where  $\epsilon_t$  is the error of  $h_t$ .

- Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where  $Z_t$  is a normalization factor (chosen so that  $D_{t+1}$  will be a distribution summing to 1).

Output the final hypothesis:

$$H(x) = \text{sign} (f(x)), \text{ where } f(x) = \sum_{t=1}^T \alpha_t h_t(x).$$

Using this notation, we can express the training error as the fraction of examples such that  $H(x_i) \neq y_i$ , or equivalently the fraction of examples such that  $y_i f(x_i) < 0$ .

Schapire and Singer [5] proved the following upper bound for the training error based upon the normalization factor  $Z_t$  :

$$\frac{1}{m} |\{i : H(x_i) \neq y_i\}| \leq \prod_{t=1}^T Z_t$$

Interestingly, the training error is bounded above by the product of the normalization factors used in each round.

One concept that is useful for discussing the testing error is the margin. The margin of a given example  $(x, y)$  is defined as  $yf(x)$ , where  $f(x)$  is simply  $f$  normalized

by the sum of the  $\alpha_t$  values so that it is between  $-1$  and  $+1$  (i.e.  $f'(x) = f(x) / \sum_{t=1}^T \alpha_t$ ).

Since,  $y \in \{-1, +1\}$  and  $-1 \leq f'(x) \leq +1$ , the margin must be in the range  $[-1, +1]$ . If the margin is positive, then the sign of  $f'(x)$  must be the same as that of  $y$  and hence the hypothesis is correct on that example, otherwise it is not. The magnitude of the margin comes from the weighted vote of weak hypotheses on the given example, so it can then be understood to be a measure of confidence in the prediction. Using margins, and understanding testing error as the probability that the margin of a new instance  $x$  is at most zero, Schapire et al. [6] give us the following bound on testing error, which holds for any  $\theta > 0$ :

$$P_D[f'(x) \leq 0] \leq P_S[f'(x) \leq \theta] + \sqrt{\frac{2 \ln m \ln |H|}{m \theta^2}} + o\left(\sqrt{\frac{\ln m}{m}}\right)$$

where  $|H|$  is the size of the hypothesis space.

In other words, the testing error is essentially bounded by the sum of two things (ignoring the last term). The first is the proportion of examples whose margin is less than the given cutoff,  $\theta$ . The second is a term which is dominated by being inversely proportional to  $\theta$  and inversely proportional to the square root of the number of examples. Note that this holds for any choice of  $\theta > 0$ . This bound is interesting, because as the margins of the training examples increase the bound drops, regardless of how many rounds of boosting have occurred. This is consistent with a tendency for the performance of AdaBoost not to worsen even after very many rounds of boosting. [7]

### III – Study and Results

We ran a series of tests using data from the machine learning repository at the University of California at Irvine and from the TREC library. [8]

We began by examining the elements of the bounds for training and testing error on AdaBoost, the product of the normalization factors and margins respectively. We expected to see the product of the normalization factors provide an upper bound in roughly the same shape as the training error. This bore out on nearly all of our data, however the actual training error was consistently well below the bound given by the product of the normalization factors, as can be seen in Figure 1 and Appendix A.

Given the proven bound of the generalization error based on margins and the observation that the margins continue to increase even after the training error drops to zero, [9] we expected the margins to increase. However, the data showed that the average margin actually decreased as the rounds continued. To get a better understanding of the margins of the examples during boosting, we created a series of scatter plots of the

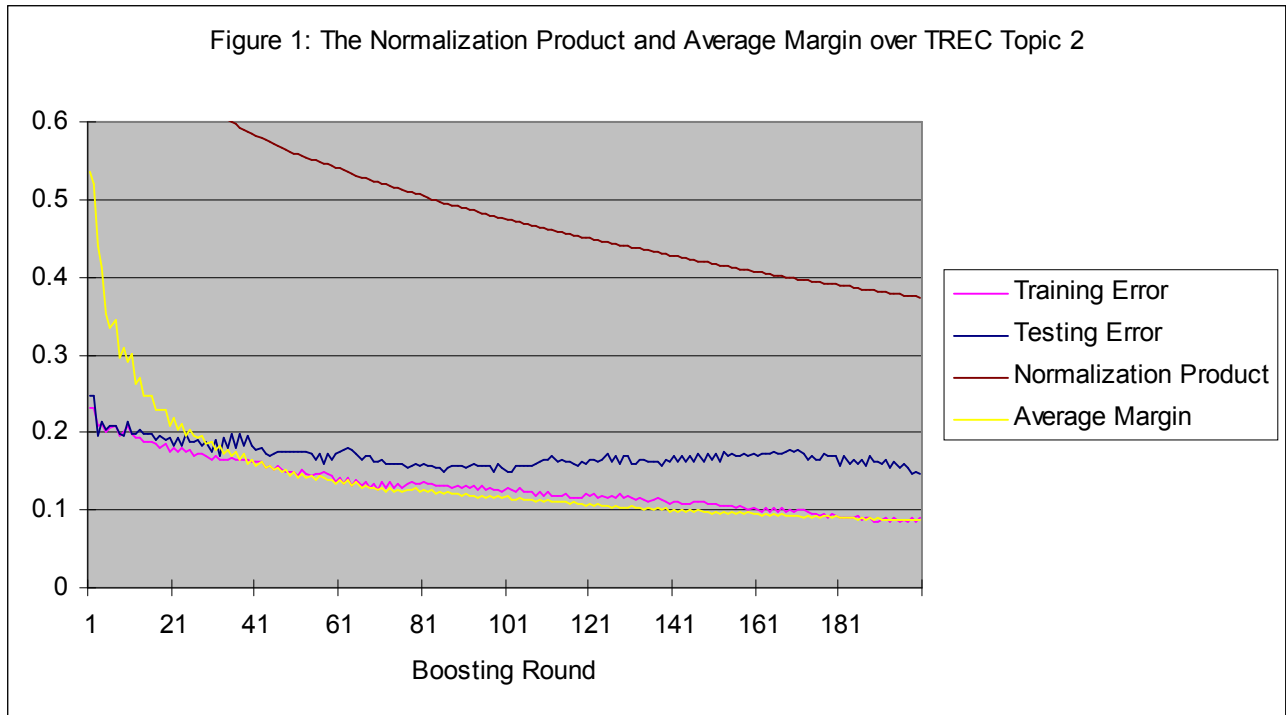
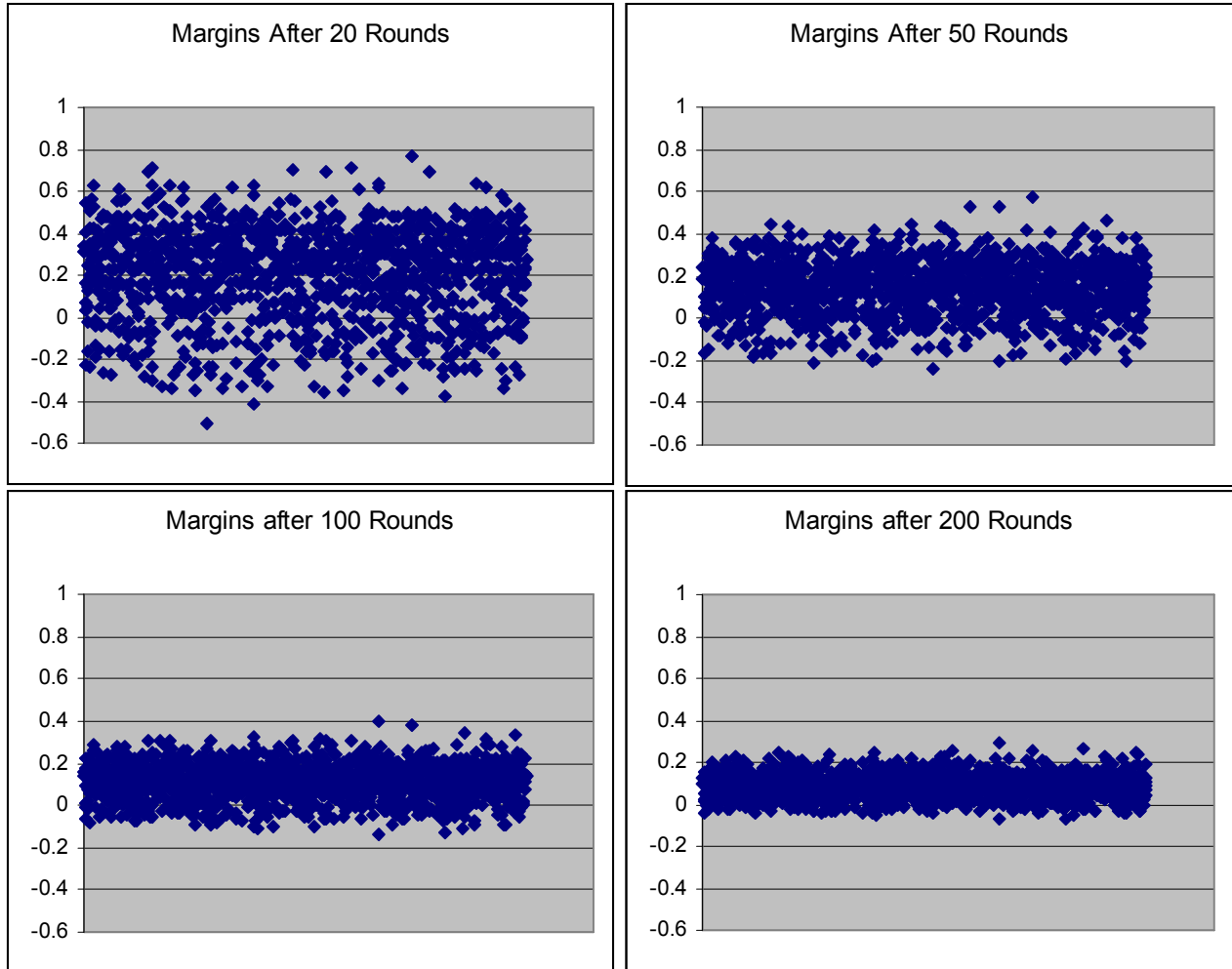


Figure 2: Margins on TREC Topic 2



margins after 20, 50, 100, and 200 rounds of boosting. In the beginning, the margins of the examples are more widely distributed between  $-1$  and  $+1$ , but as the rounds progress, the higher margins decrease, and the lower margins increase. It appears that, because AdaBoost weights the training examples each round to favor the incorrectly judged examples (those with negative margins), not only do the negative margins increase, but the positive ones tend to decrease because their examples have lesser weights. This ends up clustering the margins in a narrow range, and explains why the average of the margins decreases over time, especially in the beginning.

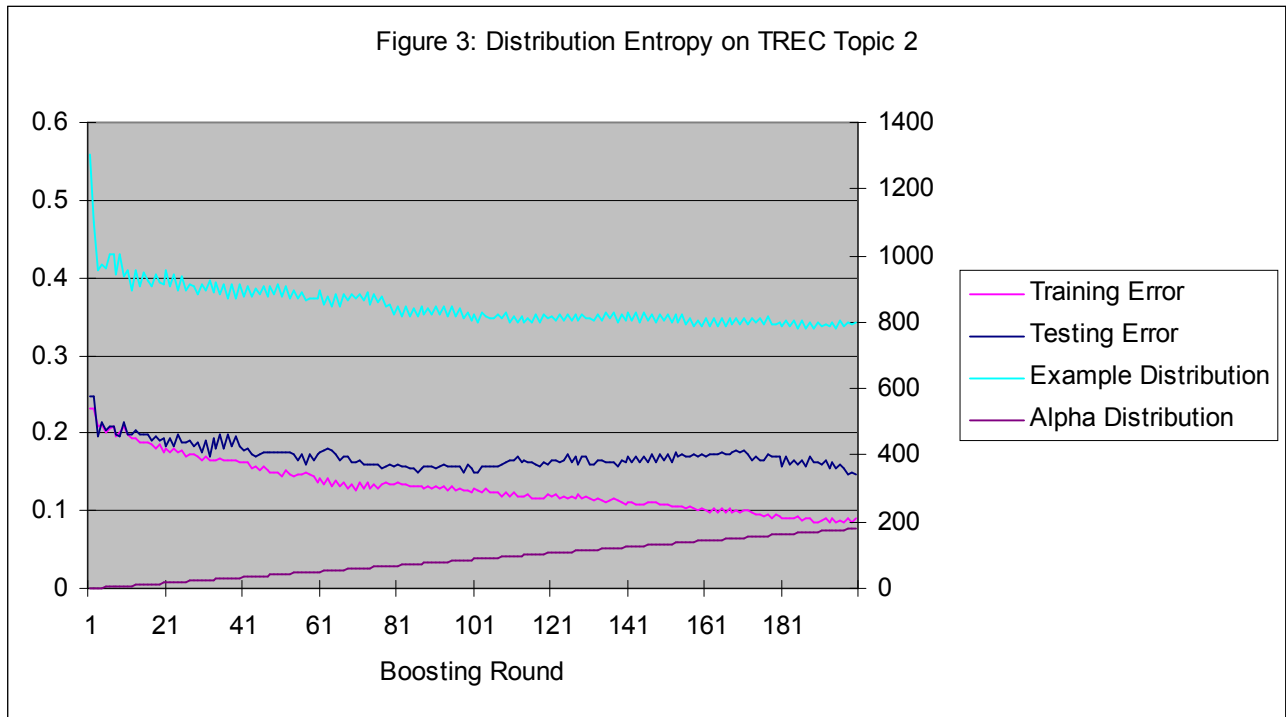
This raises the issue of the behavior of the distribution which AdaBoost is using

to weigh the examples. Does the distribution continue to shift towards weighting more heavily a certain small subset of examples which are commonly classified incorrectly? A related issue is the development of the set of weights  $\alpha_t$  assigned to the weak hypotheses generated by the weak learner. If the distribution over the examples begins to skew towards certain examples, does AdaBoost reach a point when new hypotheses are weighted less?

To examine these questions, we look to a measure of the entropy involved. Entropy is an information theoretic concept which measures the amount of uncertainty in a random variable. Given a random variable  $x$  over a probability function  $p$ , the entropy is given by  $-\sum_x p(x) \log p(x)$ . The base of the logarithm only alters the result by a constant factor, so for our purposes we will use base two, which gives an entropy measured in “bits.” The toss of a fair coin then has 1 bit of entropy, and the roll of an 8-sided die has 3 bits of entropy. Raising 2 to the power of the entropy of a distribution gives in a sense the number of possibilities of an equal distribution with that entropy.

We measured the entropy of the distribution of weights over the examples, and measured the entropy of the weights of the weak hypothesis after first normalizing them so that they sum to one. Then, raising 2 to the power of the entropy intuitively gives a measure of how many examples are being effectively trained from, and also how many weak hypotheses are effectively voting on the label of an example. We hoped to observe some relationship between these values and either the training or the testing error.

As can be seen in Figure 3, the effective number of training examples drops sharply in the beginning, but very slowly thereafter. At times, the shape of the curve is similar to that of the testing error, but not consistently.



The effective number of voters given by the weak hypotheses usually grows almost linearly, and at a slope of nearly 1, suggesting that the distribution of weights assigned to the hypotheses is fairly even as the boosting rounds progress. Hence the weights assigned to new hypotheses are nearly equal to those of previous hypotheses. Though again no relationship is apparent to either the training or testing error, it is interesting that AdaBoost continues to weight new hypotheses nearly the same as the early ones which were generated over more even distributions of examples.

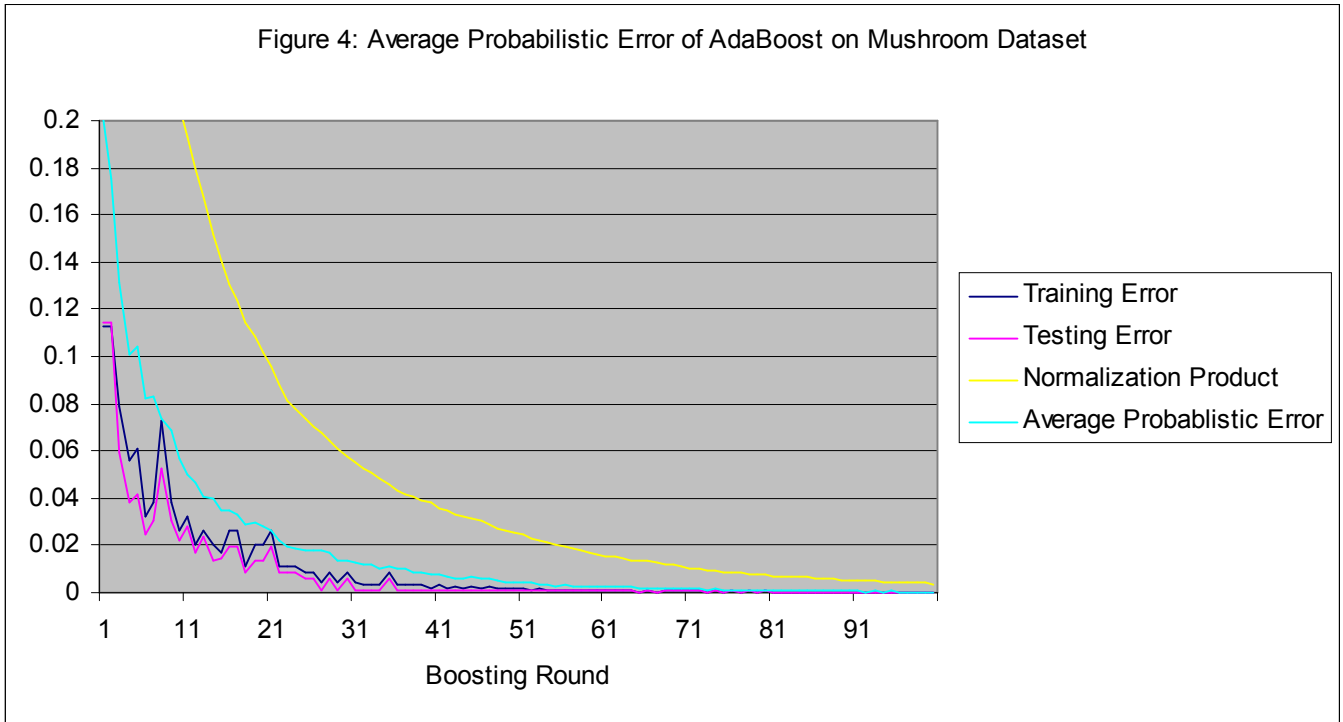
The combined hypothesis which AdaBoost produces labels an instance  $x$  based upon the sign of  $f(x)$ , if  $f$  is positive, it always assigns a label of +1, if it is negative, then it always produces a label of -1. Even if the weighted sum of all the weak hypotheses is barely above zero, the combined hypothesis labels an instance as +1. Similarly to margins, it is intuitive that instances whose sum of weights is small are more likely to be judged incorrectly. Consider a combined probabilistic hypothesis, which instead of always assigning a label based upon the sign of  $f$ , takes the magnitude of  $f$  into account as

well. Instances for which  $f(x)$  is large are labeled +1 with high probability, and instances for which  $f(x)$  is zero are labeled +1 half of the time. It seems intuitive that such a hypothesis might more closely relate training and testing error. Friedman et al. give a helpful explanation of AdaBoost in terms of odds and probabilities. [10]

To calculate the error of such a hypothesis, we need some way to transform the value of  $f(x)$  since it could potentially be any real number, negative or positive, to a probability between 0 and 1, with a value of 0 being mapped to .5 probability. There are many such functions which perform so, but one which is often used to map arbitrary real values to probability is  $\frac{e^{2f}}{1+e^{2f}}$ . There is also another intuition to use this function.

Consider  $\alpha = \frac{1}{2} \ln \left( \frac{(1-\epsilon)}{\epsilon} \right)$ , where  $\epsilon$  is the error of the weak hypothesis. The odds of an event are defined to be the ratio of the probability of its occurrence and non-occurrence, i.e.  $odds = \frac{prob\{y\}}{1-prob\{y\}}$ . The figure  $\frac{(1-\epsilon)}{\epsilon}$  can then be seen as the odds of the weak hypothesis being correct on any instance. So  $\alpha$  is half the log odds of the weak hypothesis being correct. Therefore,

$$\begin{aligned} e^{2f(x)} &= e^{2 \sum_{t=1}^T (\alpha_t h_t(x))} \\ &= e^{\sum_{t=1}^T \left( \ln \left( \frac{(1-\epsilon_t)}{\epsilon_t} \right) h_t(x) \right)} \\ &= \prod_{t=1}^T \left( e^{\ln \left( \frac{(1-\epsilon_t)}{\epsilon_t} \right) h_t(x)} \right) \\ &= \prod_{t=1}^T \left( \frac{(1-\epsilon_t)}{\epsilon_t} \right)^{h_t(x)} \end{aligned}$$



In other words,  $e^{2f}$  is equal to the product of the odds of each of the  $T$  weak hypotheses being correct on a given instance  $x$ . To convert odds to probability, solving the above relationship, we arrive at:

$$\text{prob} \quad y = \frac{od}{1 + od}$$

which, using  $e^{2f}$  as odds gives us the same mapping of  $\frac{e^{2f}}{1 + e^{2f}}$  to map the combined value of the hypothesis to a probability.

Using this measure of probability, we can calculate the probabilistic error of the hypothesis for a given example by determining the probability which the hypothesis will correctly classify the example. By doing this for all the training examples, we can calculate an average error of our new probabilistic combined hypothesis.

In every data set, the average probabilistic error over the training examples served as a strict upper bound for the training error, as well as being much tighter than the bound given by the product of the normalization factors. Also, like in the data in Figure 4, the

average probabilistic error forms a significantly smoother curve. This makes sense, since in the original hypothesis, instances may be labeled completely differently from round to round when the combined hypothesis is based on a value close to zero, whereas the probabilistic hypothesis more gradually shifts its labels.

#### IV – Conclusion

We examined several different values related to AdaBoost in hopes of better learning to understand the success which AdaBoost exhibits in training and generalization. As expected, the proven bound of the training error given by the product of the normalization factors was seen to be a very loose bound. Upon examining the margins of training examples as the boosting progresses, we were surprised to see that the average of the margins tended to decrease. However, the scatter plots of the margins revealed that the margins were compacting to a small positive range over time which explains the decrease in average as well as allowing for an increase in the lower end of the margins which allows for a decreasing bound in testing error.

To get a better understanding of how AdaBoost continues to alter the weight distribution over the training examples and how heavily it weights successive weak hypotheses in relation to earlier ones, we examined the entropy of these distributions (after normalizing the set of weights of the weak hypotheses, given by the  $\alpha_t$  values.) It was observed that the effective number of training examples from their distribution dropped sharply at first, but quickly leveled out to a shallow decline. It was also interesting to note that the weights assigned to weak hypotheses remained fairly consistent as boosting rounds progressed.

We had expected and hoped that the error of a probabilistic combined hypothesis, being related to the margins of examples, would have shown some correlation to the generalization error of AdaBoost, but didn't see a pattern to support it. It did provide a fairly tight upper bound to the training error, considerably tighter than that given by the product of the normalization factors. Perhaps there is some relationship still between it and the generalization error, and this is the subject of future work.

### Acknowledgements

Thanks first to Professor Aslam for his continued advice, suggestions, ideas, and support throughout the project. Thanks to Dan Scholnick '00 for the work he put in to create the platform JBoost, a Java application for running and testing boosting algorithms, which I was able to modify and use in order to perform this study. Thanks also to Mark Montague for his early encouragement and help in beginning this work.

*Soli Deo Gloria.*

### References

- [1] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. An extended abstract appeared in *Computational Learning Theory: Second European Conference, EuroCOLT '95*, pages 23-37, Springer-Verlag, 1995.
- [2] Robert E. Schapire, Yoram Singer. Improved Boosting Algorithms Using Confidence-rated Predictions. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, 1998.
- [3] Michael J. Kearns, Umesh V. Vazirani. *An Introduction to Computational Learning Theory*, MIT 1994.
- [4] Freund and Schapire. A decision-theoretic generalization... See note i.
- [5] Robert E. Schapire, Yoram Singer. Improved Boosting Algorithms... See note ii.
- [6] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, to appear. 1998.
- [7] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin... See note vi.
- [8] Text Retrieval Conference, <http://trec.nist.gov/>
- [9] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin... See note vi.
- [10] J. H. Friedman, T. Hastie, R. Tibshirani. Additive logistic regression: a statistical view of boosting. Dept. of Statistics, Stanford University Technical Report. 1998.

Appendix A

