Machine Learning: Proceedings of the Fourteenth International Conference, 1997.

Boosting the margin: A new explanation for the effectiveness of voting methods

Robert E. Schapire **Yoav Freund**
AT&T Labs*
600 Mountain Avenue
Murray Hill, NJ 07974 USA
{schapire, yoav}@research.att.com

Peter Bartlett
Dept. of Systems Engineering
RSISE, Aust. National University
Canberra, ACT 0200 Australia
Peter.Bartlett@anu.edu.au

Wee Sun Lee
Electrical Engineering Department
University College UNSW
Australian Defence Force Academy
Canberra ACT 2600 Australia
w-lee@ee.adfa.oz.au

Abstract. One of the surprising recurring phenomena observed in experiments with boosting is that the test error of the generated hypothesis usually does not increase as its size becomes very large, and often is observed to decrease even after the training error reaches zero. In this paper, we show that this phenomenon is related to the distribution of *margins* of the training examples with respect to the generated voting classification rule, where the margin of an example is simply the difference between the number of correct votes and the maximum number of votes received by any incorrect label. We show that techniques used in the analysis of Vapnik's support vector classifiers and of neural networks with small weights can be applied to voting methods to relate the margin distribution to the test error. We also show theoretically and experimentally that boosting is especially effective at increasing the margins of the training examples. Finally, we compare our explanation to those based on the bias-variance decomposition.

1 INTRODUCTION

In recent years, there has been growing interest in learning algorithms which achieve high accuracy by voting the predictions of several classifiers. For example, several researchers have reported significant improvements in the performance of decision-tree learning algorithms such as C4.5 or CART using voting methods [4, 5, 7, 8, 10, 15].

We refer to each of the hypotheses that is combined in the vote as a *base hypothesis* and to the final voted hypothesis as the *combined hypothesis*.

As examples of the effectiveness of these methods, consider the results of the following two experiments using the "letter" dataset.[1] In the first experiment, we used Breiman's bagging method [4] on top of C4.5. That is, we reran C4.5 many times on random "bootstrap" subsamples and combined the computed trees using simple voting. On the left of Figure 1, we have shown the training and test error curves (lower and upper curves, respectively) of the combined hypothesis as a function of the number of trees combined. The test error of C4.5 on this dataset (run just once) is 13.8%. The test error of bagging 1000 trees is 6.6%, a significant improvement. (Both of these error rates are indicated in the figure as horizontal grid lines.)

In the second experiment, we used Freund and Schapire's AdaBoost algorithm [12] on top of C4.5 for the same dataset. This algorithm is similar to bagging, except that the subsamples are chosen in a manner which concentrates on the "hardest" examples. The results of this experiment are also shown in Figure 1. Note that boosting drives the test error down even further to just 3.1%.

These error curves reveal a remarkable phenomenon, first observed by Drucker and Cortes [8], and later by Quinlan [15] and Breiman [5]. Ordinarily, as hypotheses become more and more complex, we expect their generalization error eventually to degrade. Yet these curves reveal that test error does not increase for either method even after 1000 trees have been combined (by which point, the combined hypothesis involves more than two million decision-tree nodes). How can it be that such complex hypotheses have such low error rates? This seems especially surprising for boosting in which each new decision tree is trained on an ever more specialized subsample of the training set.

Another apparent paradox is revealed in the error curve for AdaBoost. After just five trees have been combined, the training error of the combined hypothesis has already dropped to zero, but the test error continues to drop[2] from 8.4% on round 5 down to 3.1% on round 1000. Surely, a combination of five trees is much simpler than a combination of 1000 trees, and both perform equally well on the training set (perfectly, in fact). So how can it be that the larger and more complex combined hypothesis performs so much better on the test set?

*AT&T Labs is planning to move from Murray Hill. The new address will be: 180 Park Avenue, Florham Park, NJ 07932-0971.

[1] All the non-synthetic datasets used in this research are part of the "StatLog" database, which can be retrieved electronically from the UCI repository http://www.ics.uci.edu/~mlearn/MLRepository.html.

[2] Even when the training error of the combined hypothesis reaches zero, AdaBoost continues to obtain new base hypotheses by training the base learning algorithm on different subsamples of the data. Thus, the combined hypothesis continues to evolve, even after its training error reaches zero. See Section 3 for more detail.

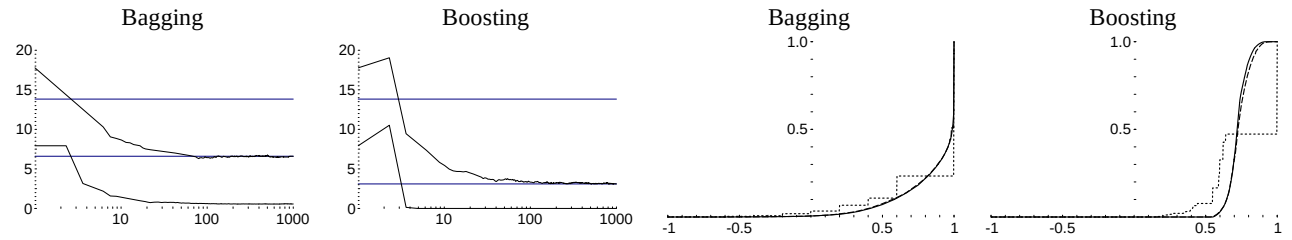

Figure 1: Error curves and margin distribution graphs for bagging and boosting C4.5 on "letter" dataset.

The results of these experiments seem to contradict Occam's razor, one of the fundamental principles in the theory of machine learning. This principle states that in order to achieve good test error, the hypothesis should be as simple as possible, and that the difference between the training error and the test error will increase when the number of parameters that are needed to describe the hypothesis increases.

Indeed, such an analysis of boosting (which could also be applied to bagging) was carried out by Freund and Schapire [12] using the methods of Baum and Haussler [2]. This analysis predicts that the test error eventually will increase as the number of base hypotheses combined increases. Such a prediction is clearly incorrect in the case of the experiments described above, as was pointed out by Quinlan [15] and Breiman [5]. The apparent contradiction is especially stark in the boosting experiment in which the test error continues to decrease even after the training error has reached zero.

Breiman [5] and others have argued that voting methods work primarily by reducing the "variance" of a learning algorithm. This explanation is useful for bagging in that bagging tends to be most effective when the variance is large. However, for boosting, this explanation is, at best, incomplete. As will be seen in Section 5, large variance of the base hypotheses is *not* a requirement for boosting to be effective. In some cases, boosting even increases the variance while reducing the overall generalization error.

In this paper, we present an alternative theoretical analysis of voting methods, applicable, for instance, to bagging, boosting, "arcing" [5] and ECOC [7]. Our approach is based on a similar result presented by Bartlett [1] in a different context. We prove rigorous upper bounds on the generalization error of voting methods in terms of a measure of performance of the combined hypothesis on the training set. Our bounds also depend on the number of training examples and the "complexity" of the base hypotheses, but do *not* depend explicitly on the number of base hypotheses. Besides explaining the shape of the observed learning curves, our analysis may be helpful in understanding why these algorithms fail or succeed, and may also lead to the design of even more effective voting methods.

The key idea of this analysis is the following. In order to analyze the generalization error, one should consider more than just the training error, i.e., the *number* of incorrect classifications in the training set. One should also take into account the *confidence* of the classifications. Here, we use a measure of the classification confidence for which it is possible to *prove* that an improvement in this measure of confidence *on the training set* guarantees an improvement in the upper bound on the generalization error.

Consider a combined hypothesis whose prediction is the result of a vote (or a weighted vote) over a set of base hypotheses. Suppose that the weights assigned to the different base hypotheses are normalized so that they sum to one. Fixing our attention on a particular example, we refer to the sum of the weights of the base hypotheses that predict a particular label as the *weight* of that label. We define the classification *margin* for the example as the difference between the weight assigned to the correct label and the maximal weight assigned to any single incorrect label. It is easy to see that the margin is a number in the range $[-1, 1]$ and that an example is classified correctly if and only if its margin is positive. A large positive margin can be interpreted as a "confident" correct classification.

Now consider the distribution of the margin over the whole set of training examples. To visualize this distribution, we plot the fraction of examples whose margin is at most x as a function of $x \in [-1, 1]$. We refer to these graphs as *margin distribution graphs*. On the right side of Figure 1, we show the margin distribution graphs that correspond to the experiments described above. The graphs show the margin distributions for bagging and boosting after 5, 100 and 1000 iterations, indicated by short-dashed, long-dashed (mostly hidden) and solid curves, respectively.

Our main observation is that both boosting and bagging tend to increase the margins associated with examples and converge to a margin distribution in which most examples have large margins. Boosting is especially aggressive in its effect on examples whose initial margin is small. Even though the training error remains unchanged (at zero) after round 5, the margin distribution graph changes quite significantly so that after 100 iterations all examples have a margin larger than 0.5. In comparison, on round 5, about 7.7% of the examples have margin below 0.5. As we see from this and other experiments detailed later in the paper, this type of reduction in the fraction of training examples with small margin is a good predictor of improvements in the test error.

The idea that maximizing the margin can improve the generalization error of a classifier was previously suggested and studied by Vapnik [19] and led to his work with Cortes

on support-vector classifiers [6], and with Boser and Guyon [3] on optimal margin classifiers. In Section 6, we discuss the relation between our work and Vapnik's in greater detail.

Shawe-Taylor et al. [16] gave bounds on the generalization error of these classifiers in terms of the margins, and Bartlett [1] used related techniques to give a similar bound for neural networks with small weights. Since voting classifiers are a special case of these neural networks, an immediate consequence of Bartlett's result is a bound on the generalization error of a voting classifier in terms of the fraction of training examples with small margin.

In Section 2, we use a similar but simpler approach to give a slightly better bound. Here we give the main intuition behind the proof. This idea brings us back to Occam's razor, though in a rather indirect way. Recall that an example is classified correctly if its margin is positive. If an example is classified by a large margin (either positive or negative), then small changes to the weights in the majority vote are unlikely to change the label. If most of the examples have a large margin then the classification error of the original majority vote and the perturbed majority vote will be similar. Suppose now that we had a small set of weighted majority rules that was fixed ahead of time, called the "approximating set." One way of perturbing the weights of the hypothesis majority vote is to find a nearby rule within the approximating set. As the approximating set is small, we can guarantee that the error of the approximating rule on the training set is similar to its generalization error, and as its error is similar to that of the original rule, the generalization error of the original rule should also be small. Thus, we are back to an Occam's razor argument in which instead of arguing that the classification rule itself is simple, we argue that the rule is *close* to a simple rule.

Boosting is particularly good at finding hypotheses with large margins in that it concentrates on those examples whose margins are small (or negative) and forces the base learning algorithm to generate good classifications for those examples. This process continues even after the training error has reached zero, which explains the continuing drop in test error.

In Section 3, we show that the powerful effect of boosting on the margin is not merely an empirical observation but is in fact the result of a provable property of the algorithm. Specifically, we are able to prove upper bounds on the number of training examples below a particular margin in terms of the training errors of the individual base hypotheses. Under certain reasonable conditions, these bounds imply that the number of training examples with small margin drops exponentially fast with the number of base hypotheses.

In Section 4, we give more examples of margin distribution graphs for other datasets, base learning algorithms and combination methods.

In Section 5, we discuss the relation of our work to bias-variance decompositions, and in Section 6, we compare our work to Vapnik's optimal margin classifiers.

2 GENERALIZATION ERROR AS A FUNCTION OF MARGIN DISTRIBUTIONS

In this section, we prove that achieving a large margin on the training set results in an improved bound on the generalization error. This bound does not depend on the number of hypotheses that are combined in the vote. The approach we take is similar to that of Shawe-Taylor et al. [16] and Bartlett [1], but the proof here is simpler and more direct. A slightly weaker version of Theorem 1 is a special case of Bartlett's main result.

We give a proof for the special case in which there are just two possible labels $\{-1, +1\}$. The proof can be generalized to larger finite sets of labels.

Let $\mathcal{H}$ denote the space from which the base hypotheses are chosen, for example, for C4.5 or CART, it is the space of decision trees. A base hypothesis $h \in \mathcal{H}$ is a mapping from an instance space X to $\{-1, +1\}$. We assume that examples are generated independently at random according to some fixed but unknown distribution $\mathcal{D}$ over $X \times \{-1, +1\}$. The training set is a list of m pairs $S = \langle (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \rangle$ chosen according to $\mathcal{D}$. We use $\mathbf{P}_{(x,y)\sim\mathcal{D}}[A]$ to denote the probability of the event A when the example (x, y) is chosen according to $\mathcal{D}$, and $\mathbf{P}_{(x,y)\sim S}[A]$ to denote probability with respect to choosing an example uniformly at random from the training set. When clear from context, we abbreviate these by $\mathbf{P}_{\mathcal{D}}[A]$ and $\mathbf{P}_{S}[A]$. We use $\mathbf{E}_{\mathcal{D}}[A]$ and $\mathbf{E}_{S}[A]$ to denote expected value in a similar manner.

We define the *convex hull* $\mathcal{C}$ of $\mathcal{H}$ as the set of mappings that can be generated by taking a weighted average of hypotheses from $\mathcal{H}$:

$$\mathcal{C} \doteq \left\{ f : x \mapsto \sum_{h \in \mathcal{H}} a_h h(x) \;\middle|\; a_h \geq 0;\ \sum_h a_h = 1 \right\}$$

where it is understood that only finitely many a_h's may be nonzero.[3] The majority vote rule that is associated with f gives the wrong prediction on the example (x, y) only if $yf(x) \leq 0$. Also, the margin of an example (x, y) in this case is simply $yf(x)$.

The following theorem, the main result of this section, states that with high probability, the generalization error of any majority vote hypothesis can be bounded in terms of the number of training examples with margin below a threshold θ, plus an additional term which depends on the number of training examples, some "complexity" measure of $\mathcal{H}$, and the threshold θ (preventing us from choosing θ too close to zero).

Because of space constraints, we prove the theorem only in the case that the base hypothesis space $\mathcal{H}$ is finite, such as the set of all decision trees of a given size over a set of discrete-valued features. In this case, our bound depends

[3] A finite support is not a requirement for our proof but is sufficient for the application here which is to majority votes over a finite number of base hypotheses.

only on $\log|\mathcal{H}|$, which is roughly the description length of a hypothesis in $\mathcal{H}$. This means that we can tolerate very large hypothesis classes.

If $\mathcal{H}$ is infinite—such as the class of decision trees over *continuous* features—we state our bound, without proof, in terms of the VC-dimension of $\mathcal{H}$.

Note that the theorem applies to *every* majority vote hypothesis, regardless of how it is computed. Thus, the theorem applies to any voting method, including boosting, bagging, etc.

Theorem 1 *Let S be a sample of m examples chosen independently at random according to $\mathcal{D}$. Assume that the base hypothesis space $\mathcal{H}$ is finite, and let $\delta > 0$. Then with probability at least $1-\delta$ over the random choice of the training set S, every weighted average function $f \in \mathcal{C}$ satisfies the following bound for all $\theta > 0$:*

$$\mathbf{P}_{\mathcal{D}}\left[yf(x) \leq 0\right] \leq \mathbf{P}_{S}\left[yf(x) \leq \theta\right] + O\left(\frac{1}{\sqrt{m}}\left(\frac{\log m \log|\mathcal{H}|}{\theta^2} + \log(1/\delta)\right)^{1/2}\right).$$

More generally, for finite or infinite $\mathcal{H}$ with VC-dimension d, the following bound holds as well:

$$\mathbf{P}_{\mathcal{D}}\left[yf(x) \leq 0\right] \leq \mathbf{P}_{S}\left[yf(x) \leq \theta\right] + O\left(\frac{1}{\sqrt{m}}\left(\frac{d\log^2(m/d)}{\theta^2} + \log(1/\delta)\right)^{1/2}\right).$$

Proof: We only prove the first case in which $\mathcal{H}$ is finite.

For the sake of the proof we define $\mathcal{C}_N$ to be the set of *unweighted* averages over N elements from $\mathcal{H}$:

$$\mathcal{C}_N \doteq \left\{ f : x \mapsto \frac{1}{N}\sum_{i=1}^{N} h_i(x) \,|\, h_i \in \mathcal{H} \right\}.$$

We allow the same $h \in \mathcal{H}$ to appear multiple times in the sum. This set will play the role of the *approximating set* in the proof.

Any majority vote hypothesis $f \in \mathcal{C}$ can be associated with a distribution over $\mathcal{H}$ as defined by the coefficients a_h. By choosing N elements of $\mathcal{H}$ independently at random according to this distribution we can generate an element of $\mathcal{C}_N$. Using such a construction we map each $f \in \mathcal{C}$ to a distribution $\mathcal{Q}$ over $\mathcal{C}_N$.

Our goal is to upper bound the generalization error of $f \in \mathcal{C}$. For any $g \in \mathcal{C}_N$ and $\theta > 0$ we can separate this probability into two terms:

$$\begin{aligned}\mathbf{P}_{\mathcal{D}}\left[yf(x) \leq 0\right] &\leq \mathbf{P}_{\mathcal{D}}\left[yg(x) \leq \theta/2\right] \\ &\quad + \mathbf{P}_{\mathcal{D}}\left[yg(x) > \theta/2 \mid yf(x) \leq 0\right].\end{aligned}$$

As this inequality holds for any $g \in \mathcal{C}_N$, we can take the expected value of the right hand side with respect to the distribution $\mathcal{Q}$ and get:

$$\begin{aligned}\mathbf{P}_{\mathcal{D}}\left[yf(x) \leq 0\right] &\leq \mathbf{P}_{\mathcal{D},g\sim\mathcal{Q}}\left[yg(x) \leq \theta/2\right] \\ &\quad + \mathbf{P}_{\mathcal{D},g\sim\mathcal{Q}}\left[yg(x) > \theta/2 \mid yf(x) \leq 0\right] \\ &= \mathbf{E}_{g\sim\mathcal{Q}}\left[\mathbf{P}_{\mathcal{D}}\left[yg(x) \leq \theta/2\right]\right] \\ &\quad + \mathbf{E}_{\mathcal{D}}\left[\mathbf{P}_{g\sim\mathcal{Q}}\left[yg(x) > \theta/2 \mid yf(x) \leq 0\right]\right]. \quad (1)\end{aligned}$$

We bound both terms in (1) separately, starting with the second term. Consider a fixed example (x, y) and take the probability inside the expectation with respect to the random choice of g. It is clear that $f(x) = \mathbf{E}_{g\sim\mathcal{Q}}\left[g(x)\right]$ so the probability inside the expectation is equal to the probability that the average over N random draws from a distribution over $\{-1, +1\}$ is different by $\theta/2$ from its expected value. The Chernoff bound yields

$$\mathbf{P}_{g\sim\mathcal{Q}}\left[yg(x) > \theta/2 \mid yf(x) \leq 0\right] \leq e^{-N\theta^2/8}. \quad (2)$$

To upper bound the first term in (1) we use the union bound. That is, the probability over the choice of S that there exists *any* $g \in \mathcal{C}_N$ and $\theta > 0$ for which

$$\mathbf{P}_{\mathcal{D}}\left[yg(x) \leq \theta/2\right] > \mathbf{P}_{S}\left[yg(x) \leq \theta/2\right] + \epsilon_N$$

is at most $(N+1)|\mathcal{C}_N|e^{-2m\epsilon_N^2}$. The exponential term comes from the Chernoff bound which holds for any single choice of g and θ. The term $(N+1)|\mathcal{C}_N|$ is an upper bound on the number of such choices where we have used the fact that, because of the form of functions in $\mathcal{C}_N$, we need only consider values of θ of the form $2i/N$ for $i = 0, \ldots, N$. Note that $|\mathcal{C}_N| \leq |\mathcal{H}|^N$.

Thus, if we set $\epsilon_N = \sqrt{(1/2m)\ln((N+1)|\mathcal{H}|^N/\delta_N)}$, and take expectation with respect to $\mathcal{Q}$, we get that, with probability at least $1-\delta_N$

$$\mathbf{P}_{\mathcal{D},g\sim\mathcal{Q}}\left[yg(x) \leq \theta/2\right] \leq \mathbf{P}_{S,g\sim\mathcal{Q}}\left[yg(x) \leq \theta/2\right] + \epsilon_N \quad (3)$$

for every choice of θ, and every distribution $\mathcal{Q}$.

To finish the argument we relate the fraction of the training set on which $yg(x) \leq \theta/2$ to the fraction on which $yf(x) \leq \theta$, which is the quantity that we measure. Notice that

$$\begin{aligned}\mathbf{P}_{S,g\sim\mathcal{Q}}\left[yg(x) \leq \theta/2\right] &\leq \mathbf{P}_{S}\left[yf(x) \leq \theta\right] \\ &\quad + \mathbf{E}_{S}\left[\mathbf{P}_{g\sim\mathcal{Q}}\left[yg(x) \leq \theta/2 \mid yf(x) > \theta\right]\right]. \quad (4)\end{aligned}$$

To bound the expression inside the expectation we use the Chernoff bound as we did for Equation (2) and get

$$\mathbf{P}_{g\sim\mathcal{Q}}\left[yg(x) \leq \theta/2 \mid yf(x) > \theta\right] \leq e^{-N\theta^2/8}. \quad (5)$$

Let $\delta_N = \delta/(N(N+1))$ so that the probability of failure for any N will be at most $\sum_{N\geq 1}\delta_N = \delta$. Then combining Equations (1), (2), (3), (4) and (5), we get that, with probability at least $1-\delta$, for every $\theta > 0$ and every $N \geq 1$:

$$\begin{aligned}\mathbf{P}_{\mathcal{D}}\left[yf(x) \leq 0\right] &\leq \mathbf{P}_{S}\left[yf(x) \leq \theta\right] \\ &\quad + 2e^{-N\theta^2/8} + \sqrt{\frac{1}{2m}\ln\frac{N(N+1)^2|\mathcal{H}|^N}{\delta}}. \quad (6)\end{aligned}$$

Finally, the statement of the first part of the theorem follows by setting $N = \lceil (4/\theta^2) \ln(m/\ln|\mathcal{H}|) \rceil$. ∎

We can extend the proof given above to the more general case in which $\mathcal{H}$ may be infinite using VC-dimension rather than $\log|\mathcal{H}|$. Here, we sketch an alternative, more general approach which can be applied to any class of real-valued functions. The use of an approximating class, such as $\mathcal{C}_N$ in the proof of Theorem 1, is central to our approach. We refer to such an approximating class as a sloppy cover. More formally, for a class $\mathcal{F}$ of real-valued functions, a training set S of length m, and positive real numbers θ and ϵ, we say that a function class $\hat{\mathcal{F}}$ is an *ϵ-sloppy θ-cover of $\mathcal{F}$ with respect to* S if, for all f in $\mathcal{F}$, there exists $\hat{f}$ in $\hat{\mathcal{F}}$ with $\mathbf{P}_{x\sim S}\left[|\hat{f}(x) - f(x)| > \theta\right] < \epsilon$. Let $\mathcal{N}(\mathcal{F}, \theta, \epsilon, m)$ denote the maximum, over all training sets S of size m, of the size of the smallest ϵ-sloppy θ-cover of $\mathcal{F}$ with respect to S. Standard techniques allow us to prove that the probability (over the random choice of S) that there exists an f in $\mathcal{F}$ for which

$$\mathbf{P}_{\mathcal{D}}\left[yf(x) \le 0\right] > \mathbf{P}_{S}\left[yf(x) \le \theta\right] + \epsilon$$

is no more than $2\mathcal{N}(\mathcal{F}, \theta/2, \epsilon/8, 2m)\exp(-\epsilon^2 m/32)$. (The proof is essentially identical to that of Theorem 1 in Bartlett [1].)

The second part of Theorem 1 can now be proved by constructing a sloppy cover using the same probabilistic argument in the proof above, i.e., by choosing an element of $\mathcal{C}_N$ randomly by sampling functions from $\mathcal{H}$. In addition, this result leads to a slight improvement (by log factors) of the main result of Bartlett [1], which gives bounds on generalization error for neural networks with real outputs in terms of the size of the network weights and the margin distribution.

3 THE EFFECT OF BOOSTING ON MARGIN DISTRIBUTIONS

We now give theoretical evidence that Freund and Schapire's [12] AdaBoost algorithm is especially suited to the task of maximizing the number of training examples with large margin.

Space limitations permit only a terse review of their algorithm. We adopt the notation used in the previous section, and restrict our attention to the binary case.

Boosting works by sequentially rerunning a base learning algorithm, each time using a different distribution over training examples. That is, on each round $t = 1, \ldots, T$, a distribution D_t is computed over the training examples, or, formally, over the set of indices $\{1, \ldots, m\}$. The goal of the base learning algorithm then is to find a hypothesis h_t with small error $\epsilon_t = \mathbf{P}_{i\sim D_t}\left[y_i \neq h_t(x_i)\right]$. The distribution used by AdaBoost is initially uniform ($D_1(i) = 1/m$), and then is updated multiplicatively on each round: $D_{t+1}(i) = D_t(i)\exp(-y_i\alpha_t h_t(x_i))/Z_t$. Here, $\alpha_t = \frac{1}{2}\ln((1-\epsilon_t)/\epsilon_t)$ and Z_t is an overall normalization constant chosen so that D_{t+1} sums to one. It can be verified that, in our case, $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$. The final combined hypothesis is a weighted majority vote of the base hypotheses, namely, $\mathrm{sign}(f)$ where $f(x) = \left(\sum_{t=1}^{T}\alpha_t h_t(x)\right) / \left(\sum_{t=1}^{T}\alpha_t\right)$. Note that, on round t, AdaBoost places the most weight on examples (x, y) for which $y\sum_{t'=1}^{t-1}\alpha_{t'}h_{t'}(x)$ is smallest. This quantity is exactly the margin of the combined hypothesis computed up to this point.

Freund and Schapire [12] prove that if the training error rates of all the base hypotheses are bounded below $1/2$ so that $\epsilon_t \le 1/2 - \gamma$ for some $\gamma > 0$, then the training error of the combined hypothesis decreases exponentially fast with the number of base hypotheses that are combined. The training error is equal to the fraction of examples for which $yf(x) \le 0$. It is a simple matter to extend their proof to show that if θ is not too large, then the fraction of training examples for which $yf(x) \le \theta$ also decreases to zero exponentially fast with the number of base hypotheses (or boosting iterations).

Theorem 2 *Suppose the base learning algorithm, when called by AdaBoost, generates hypotheses with weighted training errors $\epsilon_1, \ldots, \epsilon_T$. Then for any θ, we have that*

$$\mathbf{P}_{(x,y)\sim S}\left[yf(x) \le \theta\right] \le 2^T \prod_{t=1}^{T}\sqrt{\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}}. \quad (7)$$

Proof sketch: Note that if $yf(x) \le \theta$ then

$$y\sum_{t=1}^{T}\alpha_t h_t(x) \le \theta\sum_{t=1}^{T}\alpha_t$$

and so

$$\exp\left(-y\sum_{t=1}^{T}\alpha_t h_t(x) + \theta\sum_{t=1}^{T}\alpha_t\right) \ge 1.$$

Therefore,

$$\begin{aligned}
&\mathbf{P}_{(x,y)\sim S}\left[yf(x) \le \theta\right] \\
&\quad\le \mathbf{E}_{(x,y)\sim S}\left[\exp\left(-y\sum_{t=1}^{T}\alpha_t h_t(x) + \theta\sum_{t=1}^{T}\alpha_t\right)\right] \\
&\quad= \frac{\exp\left(\theta\sum_{t=1}^{T}\alpha_t\right)}{m}\sum_{i=1}^{m}\exp\left(-y_i\sum_{t=1}^{T}\alpha_t h_t(x_i)\right) \\
&\quad= \exp\left(\theta\sum_{t=1}^{T}\alpha_t\right)\left(\prod_{t=1}^{T}Z_t\right)\sum_{i=1}^{m}D_{T+1}(i)
\end{aligned}$$

where the last equality follows from the definition of D_{T+1}. Plugging in the values of α_t and Z_t gives the theorem. ∎

To understand the significance of the result, assume that, for all t, $\epsilon_t \le 1/2 - \gamma$ for some $\gamma > 0$. Then we can simplify the upper bound in Equation (7) to:

$$\left(\sqrt{(1-2\gamma)^{1-\theta}(1+2\gamma)^{1+\theta}}\right)^T.$$

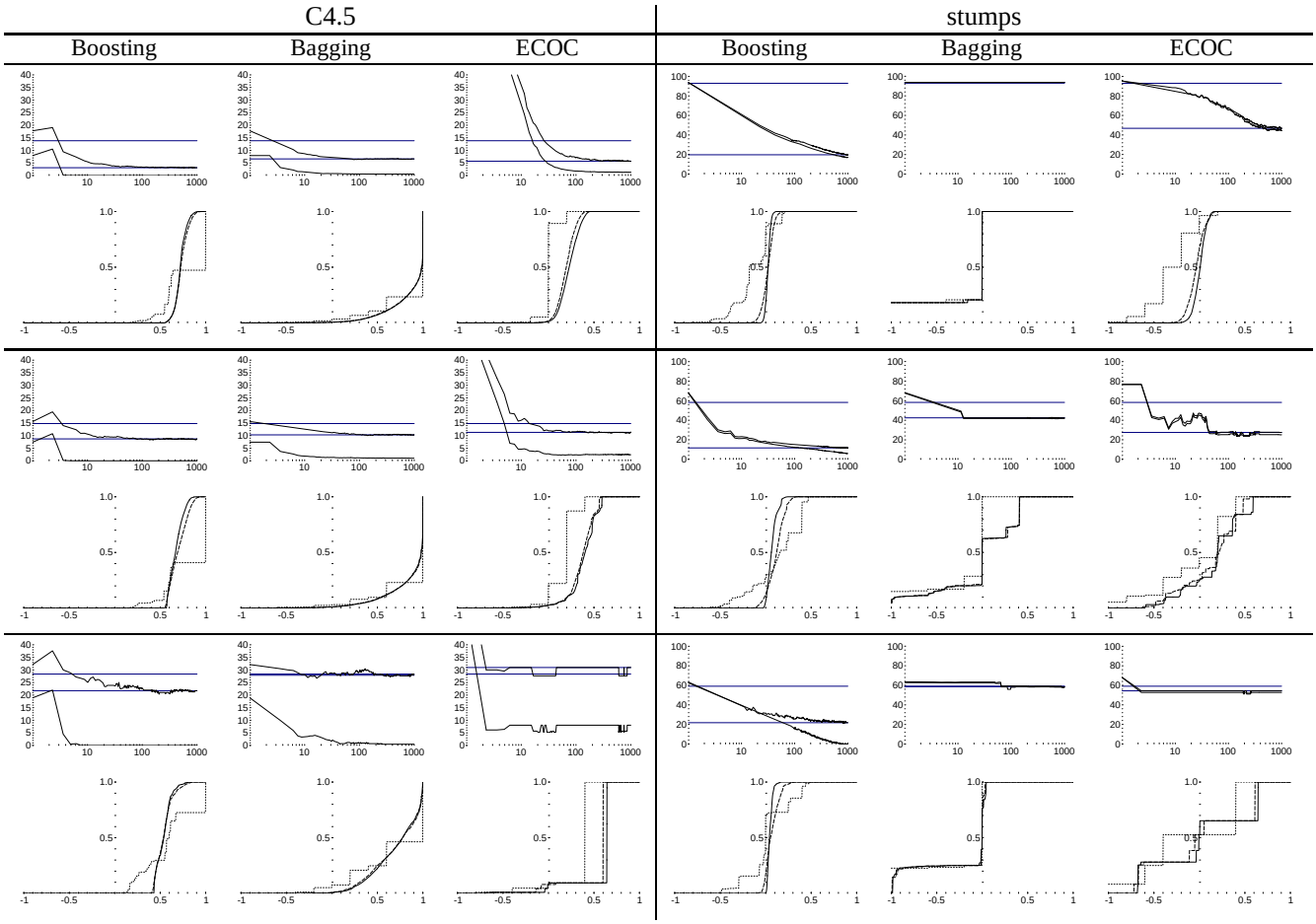

Figure 2: Error curves and margin distribution graphs for several learning methods on "letter" (top), "satimage" (middle) and "vehicle" (bottom) datasets.

If $\theta < \gamma$, it can be shown that the expression inside the parentheses is smaller than 1 so that the probability that $yf(x) \leq \theta$ decreases exponentially fast with T.[4]

Although this theorem applies only to binary classification problems, Freund and Schapire [12] give extensive treatment to the multi-class case. All of their results can be extended to prove analogous theorems about margin distributions for this more general case.

4 MORE MARGIN DISTRIBUTION GRAPHS

Figure 2 shows a series of error curves and margin distribution graphs for a variety of datasets and learning methods. Vertically, the figure is split into three sections corresponding to the three "StatLog" datasets tested — "letter," "satimage" and "vehicle." The first two came with their own test sets. For the vehicle dataset, we randomly selected half of the data to be held out as a test set. Note that these curves represent only a single run of each algorithm.

In addition to bagging and boosting, we also used a variant of Dietterich and Bakiri's method of error-correcting output codes [7], which can be viewed as a voting method. However, rather than carefully constructing error-correcting codes, we simply used random output codes which are highly likely to have similar properties.

For the base learning algorithm, we used C4.5, and we also used a simple algorithm for finding the best single-node, binary-split decision tree (a decision "stump"). Since this latter algorithm is very weak, we used the "pseudoloss" versions of boosting and bagging. (See Freund and Schapire [12, 10] for details.)

As explained in the introduction, each of the learning curve figures shows the training error (bottom) and test error (top) curves. We have also indicated as horizontal grid lines the error rate of the base hypothesis when run just once, as well as the error rate of the combined hypothesis after 1000

[4] We can show that if γ is known in advance then an exponential decrease in the probability can be achieved (by a slightly different boosting algorithm) for any $\theta < 2\gamma$. However, we don't know how to achieve this improvement when no nontrivial lower bound on $1/2 - \epsilon_t$ is known a priori.

iterations. Note the log scale used in these figures.

Margin distribution graphs are shown for 5, 100 and 1000 iterations indicated by short-dashed, long-dashed (sometimes barely visible) and solid curves, respectively.

It is interesting that, across datasets, all of the learning algorithms tend to produce margin distribution graphs of roughly the same character. As already noted, when used with C4.5, boosting is especially aggressive at increasing the margins of the examples, so much so that it is "willing" to suffer significant reductions in the margins of those examples that already have large margins. This can be seen in the figure where we observe that the maximal margin in the final hypothesis is bounded well away from 1. Contrast this with the final graph for bagging in which as many as half of the examples have a margin of 1.

The graphs for ECOC with C4.5 resemble in shape those for boosting more so than bagging, but tend to have overall lower margins.

Note that, on every dataset, both boosting and bagging eventually achieve perfect or nearly perfect accuracy on the training sets (at least 99%), but the generalization error for boosting is better. The explanation for this is evident from the margin distribution graphs where we see that, for boosting, far fewer training examples have margin close to zero.

When used with stumps, boosting is unable to achieve such large margins since, consistent with Theorem 2, the base hypotheses have much higher training errors. Presumably, such low margins do not adversely affect the generalization error because the complexity of decision stumps is so much smaller than that of full decision trees.

5 RELATION TO BIAS-VARIANCE THEORY

One of the main explanations for the improvements achieved by voting classifiers is based on separating the expected error of a classifier into a *bias* term and a *variance* term. While the details of these definitions differ from author to author [5, 13, 14, 18], they are all attempts to capture the following quantities: The bias term measures the *persistent* error of the learning algorithm, in other words, the error that would remain even if we had an infinite number of independently trained hypotheses. The variance term measures the error that is due to *fluctuations* that are a part of generating a single hypothesis. The idea is that by averaging over many hypotheses one can reduce the variance term and in that way reduce the expected error.

In this section, we discuss a few of the strengths and weakness of bias-variance theory as an explanation for the performance of voting methods, especially boosting.

The bias-variance decomposition for classification. The origins of bias-variance analysis are in quadratic regression. Averaging several independently trained regression functions will never increase the expected error. This encouraging fact is nicely reflected in the bias-variance separation of the expected quadratic error. Both bias and variance are always nonnegative and averaging decreases the variance term without changing the bias term.

One would naturally hope that this beautiful analysis would carry over from quadratic regression to classification. Unfortunately, as has been observed before us, taking the majority vote over several classification rules can sometimes result in an *increase* in the expected classification error. This simple observation suggests that it may be inherently more difficult or even impossible to find a bias-variance decomposition for classification as natural and satisfying as in the quadratic regression case.

This difficulty is reflected in the myriad definitions that have been proposed for bias and variance [5, 13, 14, 18]. Rather than discussing each one separately, for the remainder of this section, except where noted, we follow the definitions given by Kong and Dietterich [14], and referred to as "Definition 0" by Breiman [5].

Bagging and variance reduction. The notion of variance certainly seems to be helpful in understanding bagging; empirically, bagging appears to be most effective for learning algorithms with large variance. In fact, under idealized conditions, variance is *by definition* the amount of decrease in error effected by bagging a large number of base hypotheses. This ideal situation is one in which the bootstrap samples used in bagging faithfully approximate truly independent samples. However, this assumption can fail to hold in practice, in which case, bagging may not perform as well as expected, even when variance dominates the error of the base learning algorithm.

This can happen even when the data distribution is very simple. For example, consider data generated according to the following distribution. The label $y \in \{-1, +1\}$ is chosen uniformly at random. The instance $x \in \{-1, +1\}^7$ is then chosen by picking each of the 7 bits to be equal to y with probability 0.9 and $-y$ with probability 0.1. Thus, each coordinate of x is an independent noisy version of y. For our base learner, we use a learning algorithm which generates a hypothesis that is equal to the single coordinate of x which is the best predictor of y with respect to the training set. It is clear that each coordinate of x has the same probability of being chosen as the hypothesis on a random training set, so the aggregate predictor over many independently trained samples is the unweighted majority vote over the coordinates of x, which is also the Bayes optimal predictor in this case. Thus, the bias of our learning algorithm is exactly zero. The prediction error of the majority rule is roughly 0.3%, and so a variance of about 9.7% strongly dominates the expected error rate of 10%. In such a favorable case, one would hope that bagging could get close to the error of the Bayes optimal predictor.

However, using a training set of 500 examples, the generalization error achieved by bagging is 5.6% after 200 iterations. (All results are averaged over many runs.) The reason for this poor performance is that, in each random sample, some of the coordinates of x are slightly more correlated with y and bagging tends to pick these coordinates much more often than the others. Thus, in this case,

		Kong & Dietterich [14] definitions								Breiman [5] definitions							
		stumps					C4.5			stumps					C4.5		
			error		pseudo-loss			error			error		pseudo-loss			error	
name		–	boost	bag	boost	bag	–	boost	bag	–	boost	bag	boost	bag	–	boost	bag
waveform	bias	26.0	3.8	22.8	0.8	11.9	1.5	0.5	1.4	19.2	2.6	15.7	0.5	7.9	0.9	0.3	1.4
	var	5.6	2.8	4.1	3.8	8.6	14.9	3.7	5.2	12.5	4.0	11.2	4.1	12.5	15.5	3.9	5.2
	error	44.7	19.6	39.9	17.7	33.5	29.4	17.2	19.7	44.7	19.6	39.9	17.7	33.5	29.4	17.2	19.7
twonorm	bias	2.5	0.6	2.0			0.5	0.2	0.5	1.3	0.3	1.1			0.3	0.1	0.3
	var	28.5	2.3	17.3			18.7	1.8	5.4	29.6	2.6	18.2			19.0	1.9	5.6
	error	33.3	5.3	21.7			21.6	4.4	8.3	33.3	5.3	21.7			21.6	4.4	8.3
threenorm	bias	24.5	6.3	21.6			4.7	2.9	5.0	14.2	4.1	13.8			2.6	1.9	3.1
	var	6.9	5.1	4.8			16.7	5.2	6.8	17.2	7.3	12.6			18.8	6.3	8.6
	error	41.9	22.0	36.9			31.9	18.6	22.3	41.9	22.0	36.9			31.9	18.6	22.3
ringnorm	bias	46.9	4.1	46.9			2.0	0.7	1.7	32.3	2.7	37.6			1.1	0.4	1.1
	var	−7.9	6.6	−7.1			15.5	2.3	6.3	6.7	8.0	2.2			16.4	2.6	6.9
	error	40.6	12.2	41.4			19.0	4.5	9.5	40.6	12.2	41.4			19.0	4.5	9.5
Kong &	bias	49.2	49.1	49.2	7.7	35.1	7.7	5.5	8.9	49.0	49.0	49.0	5.3	29.7	5.1	3.5	6.2
Dietterich	var	0.2	0.2	0.2	5.1	3.5	7.2	6.6	4.3	0.4	0.3	0.5	7.5	8.9	9.8	8.5	6.9
	error	49.5	49.3	49.5	12.8	38.6	14.9	12.1	13.1	49.5	49.3	49.5	12.8	38.6	14.9	12.1	13.1

Table 1: Bias-variance experiments using boosting and bagging on synthetic data. Columns labeled with a dash indicate that the base learning algorithm was run just once.

the behavior of bagging is very different from its expected behavior on truly independent training sets.

Boosting, on the same data, achieved a test error of 0.6%.

Boosting and variance reduction. Breiman [5] argued that boosting is primarily a variance-reducing procedure. Some of the evidence for this comes from the observed effectiveness of boosting when used with C4.5 or CART, algorithms known empirically to have high variance. As the error of these algorithms is mostly due to variance, it is not surprising that the reduction in the error is primarily due to a reduction in the variance. However, our experiments show that boosting can also be highly effective when used with learning algorithms whose error tends to be dominated by bias rather than variance.[5]

We ran boosting and bagging on four artificial datasets described by Breiman [5], as well as the artificial problem studied by Kong and Dietterich [14]. Following previous authors, we used training sets of size 200 for the latter problem and 300 for the others. For the base learning algorithm, we tested C4.5. We also used a very simple base learning algorithm that essentially finds the single-node binary-split decision tree (a decision "stump") that minimizes either the training error or the "pseudoloss" (see Freund and Schapire [10] for details). We then estimated bias, variance and average error of these algorithms by rerunning them many times. For these experiments, we used both the bias-variance definitions given by Kong and Dietterich [14] and those proposed more recently by Breiman [5]. For multi-class problems, following Freund and Schapire [10], we tested both error-based and pseudoloss-based versions of bagging and boosting. For two-class problems, only the error-based versions were used.

The results are summarized in Table 1. Clearly, boosting is doing more than reducing variance. For instance, on "ringnorm," boosting decreases the overall error of the stump algorithm from 40.6% to 12.2%, but actually increases the variance from −7.9% to 6.6% using Kong and Dietterich's definitions, or from 6.7% to 8.0% using Breiman's definitions.

Breiman also tested boosting with a low-variance base learning algorithm—namely, linear discriminant analysis (LDA)—and attributed the ineffectiveness of boosting in this case to the "stability" (low variance) of LDA. The experiments with the fairly stable stump algorithm suggest that stability in itself may not be sufficient to predict boosting's failure. Our theory is more specific and rigorous in pinpointing the reasons that boosting can fail. Taken together, Theorem 1 and Theorem 2 state that boosting can perform poorly only if: (1) there is insufficient training data relative to the "complexity" of the base hypotheses, or (2) the training errors of the base hypotheses (the ϵ_t's in Theorem 2) become too large too quickly.

6 RELATION TO VAPNIK'S MAXIMAL MARGIN CLASSIFIERS

The use of the margins of real-valued hypotheses to predict generalization error was previously studied by Vapnik [19] in his work with Boser and Guyon [3] and Cortes [6] on optimal margin classifiers. This method, like boosting, aims to find a linear combination of base hypotheses which maximizes margins on training data. However, the two methods differ in the assumptions used on the norms of the base hypotheses and the weights. In the optimal margin method, the sum of the squared outputs of the base hypotheses and the sum of the squared weights (that is, the squared l_2 norms) are both assumed to be bounded. In boosting, the maximum absolute value of the base hypotheses (l_∞ norm) and the sum of the absolute values of the weights (l_1 norm) are assumed to be bounded. These assumptions mean that boosting is particularly suited for the case when all the base hypotheses have a similar output range, such as $\{-1, +1\}$. On the other hand, the optimal margin method is suitable for the case where the magnitudes of the base hypotheses

[5] In fact, the original goal of boosting was to reduce the error of so-called "weak" learning algorithms which tend to have very large bias.

decay as their number increases, such as in the dual spaces used by Vapnik. Related to this, the optimal margin method uses quadratic programming for its optimization, whereas the boosting algorithm can be seen as a method for approximate linear programming [11].

Vapnik [19] showed that, with an l_2 constraint on a set of points, the set of normalized (in l_2) linear combinations of these points that have margin at least γ on all points has VC-dimension that decreases as $1/\gamma^2$. This result implies bounds on the generalization error in terms of the expected margin on test points (but typically this is not known). Shawe-Taylor et al. [16] used techniques from the theory of learning real-valued functions to give bounds on generalization error in terms of margins on the *training examples*, for linear combinations with an l_2 bound on coefficients. Shawe-Taylor et al. [17] also gave related results for arbitrary real classes, and Bartlett [1] gave bounds of this type for neural networks with bounded weights (in l_1). (Like boosting, neural network learning algorithms attempt to find functions with large margins.)

Vapnik [19] gave an alternative analysis of optimal margin classifiers, based on the number of *support vectors*, i.e., the number of examples that define the final hypothesis. This analysis is preferable to the analysis that depends on the *size* of the margin when only a few of the training examples are support vectors. Previous work [9] has suggested that boosting also can be used as a method for selecting a small number of "informative" examples from the training set. Investigating the relevance of this type of bound when applying boosting to real-world problems is an interesting open research direction.

7 OPEN PROBLEMS

The methods in this paper allow us to upper bound the generalization error of a classifier based on simple statistics which can be measured using the training data. We believe that these bounds give correct qualitative predictions regarding the behavior of the generalization error. Unfortunately, however, in their current form, our upper bounds are too pessimistic to be used as actual numerical estimates of the error, something which would be quite useful in practice. An important open problem is to derive more careful and precise bounds which can be used for this purpose. Besides paying closer attention to constant factors, such an analysis might also involve the measurement of more sophisticated statistics.

ACKNOWLEDGMENTS

Many thanks to Leo Breiman for a poignant email exchange which challenged us to think harder about these problems. Thanks also to all those who contributed to the datasets used in this paper.

REFERENCES

[1] Peter L. Bartlett. For valid generalization, the size of the weights is more important than the size of the network. In *Advances in Neural Information Processing Systems 9*, 1997.

[2] Eric B. Baum and David Haussler. What size net gives valid generalization? In *Advances in Neural Information Processing Systems I*, pages 81–90. Morgan Kaufmann, 1989.

[3] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

[4] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[5] Leo Breiman. Bias, variance, and arcing classifiers. Technical Report 460, Statistics Department, University of California at Berkeley, 1996.

[6] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.

[7] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, January 1995.

[8] Harris Drucker and Corinna Cortes. Boosting decision trees. In *Advances in Neural Information Processing Systems 8*, pages 479–485, 1996.

[9] Yoav Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[10] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 148–156, 1996.

[11] Yoav Freund and Robert E. Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pages 325–332, 1996.

[12] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, To appear. An extended abstract appeared in EuroCOLT'95.

[13] Ron Kohavi and David H. Wolpert. Bias plus variance decomposition for zero-one loss functions. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 275–283, 1996.

[14] Eun Bae Kong and Thomas G. Dietterich. Error-correcting output coding corrects bias and variance. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 313–321, 1995.

[15] J. R. Quinlan. Bagging, boosting, and C4.5. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 725–730, 1996.

[16] John Shawe-Taylor, Peter L. Bartlett, Robert C. Williamson, and Martin Anthony. A framework for structural risk minimisation. In *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pages 68–76, 1996.

[17] John Shawe-Taylor, Peter L. Bartlett, Robert C. Williamson, and Martin Anthony. Structural risk minimization over data-dependent hierarchies. Technical Report NC-TR-96-053, Neurocolt, 1996.

[18] Robert Tibshirani. Bias, variance and prediction error for classification rules. Technical report, University of Toronto, November 1996.

[19] V. N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer-Verlag, 1982.