
Boosting on a Feature Budget

Lev Reyzin

LREYZIN@YAHOO-INC.COM

Yahoo! Research, 111 40th St. 17th Floor, New York, NY 10018, USA

Abstract

In this paper we explore the problem of predicting using a limited number of features per training example. We suggest that a modification of AdaBoost (or other boosting algorithms), where hypotheses are sampled from the final weighted predictor, is well-suited for this task. We derive bounds for the number of samples needed and also verify the effectiveness of this method in experiments.

1. Introduction

In this paper, we address the following problem in supervised learning. Oftentimes, while a learning algorithm has ample time to build a good predictor in the training phase, accessing the features of a new example at test time can be costly. This is the setting in medical diagnosis, where we can think of looking at a new feature as performing new test on the patient. Each test is often expensive, takes valuable time, and is often inconvenient or dangerous to the patient. Hence, the goal, loosely stated, is to predict as well as possible without performing unnecessary tests.

More precisely, we consider the setting where a learning algorithm is given a limit the number of features it may observe on a given example in the test phase, and subject to this constraint, must try to predict as accurately as possible. We call this problem **feature-efficient prediction**.

Herein, we propose using a simple modification of the boosting paradigm to obtain such an algorithm. Boosting algorithms classify new examples by taking a weighted vote of many simpler **base learners**. The idea behind our modification is simple: to train an entire boosting classifier and then, for each new example, to predict by sampling the weak learners' votes using a distribution induced by their weights. In this paper,

we show that using base learners that use only a few features and sampling a limited number of hypotheses, can still provide the strong accuracy of boosting.

2. Background

2.1. Past Work

Work on various forms of feature efficiency has taken many forms – we give a brief overview of them.

In the area of sequential analysis, Wald (1947) began a line of research considering the problem of running a clinical trial sequentially, only testing future patients if the validity of the hypothesis in question was still uncertain. A thorough treatment of this subject is given by Chernoff (1972).

In learning theory, a variant of our problem was studied by Ben-David and Dichterman (1993), who examined the theory behind learning using random partial information from examples and discussed conditions for learning in their model.

Greiner et al. (2002) considered the problem of feature-efficient prediction, where a classifier must choose which features to examine before predicting. They showed that a variant of PAC-learnability is still possible even without access to the full feature set.

The problem of predicting quickly and efficiently has also received interest due to its relevance in internet-scale applications. In that vein, Globerson and Roweis (2006) looked at building robust predictors that are resilient to corrupted or missing features.

Very recently (concurrently with this work), Cesa-Bianchi et al. (2010) studied how to efficiently learn a linear predictor, in the setting where the learner can access only a few features per example. Their algorithm also tackles a problem similar to our own, but is restricted to learning linear predictors.

2.2. Boosting

Our idea is to exploit the power of boosting algorithms – here we give a brief overview of boosting and the

Appearing in the 26th International Conference on Machine Learning, Workshop on Budgeted Learning, Haifa, Israel, 2010. Copyright 2010 by the author(s)/owner(s).

related work we rely on.

Boosting algorithms combine moderately inaccurate prediction rules and make a weighted majority vote to form a single classifier. On each round, a boosting algorithm generates a new prediction rule to use and then places more weight on the examples classified incorrectly. Hence, boosting constantly focuses on correctly classifying the hardest examples. A nice overview of boosting appears in Schapire (2003)

In this paper, we will make use of **AdaBoost** (Freund & Schapire, 1997), the most ubiquitous of the boosting algorithms. It is presented in Algorithm 1.

Algorithm 1 AdaBoost

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

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

for $t = 1, \dots, T$ **do**

 Train base learner using distribution D_t .

 Get base classifier $h_t : X \rightarrow \{-1, +1\}$.

 Let $\gamma_t = \sum_i D_t(i) y_i h_t(x_i)$

 Choose $\alpha_t = \frac{1}{2} \ln \frac{1+\gamma_t}{1-\gamma_t}$

 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)

end for

Output the final classifier:

$$H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(x) \right)$$

Soon after its appearance, it was observed that AdaBoost tends not to **overfit** training data with more rounds of boosting, despite that its complexity increases with every round. Work on this area is very rich, and we direct the reader to (Grove & Schuurmans, 1998; Schapire et al., 1998; Mason et al., 1998; Reyzin & Schapire, 2006) for more detail.

An important part of the history of trying to account for AdaBoost’s performance is Schapire et al.’s (1998) explanation of AdaBoost’s tendency not to overfit in terms of the margins of the training examples, where the **margin** is a quantity that can be interpreted as measuring the confidence in the prediction of the combined classifier. As we shall see in Section 3.2 their work drives our analysis of Algorithm 2.

3. Algorithm and Analysis

In this section we present our algorithm for feature-efficient prediction and an analysis of it.

3.1. Algorithm

We propose Algorithm 2 which we call AdaBoostRS (for AdaBoost, Randomly Sampled), for feature-efficient prediction. AdaBoostRS uses AdaBoost in training and then samples from AdaBoost’s hypothesis distribution. AdaBoostRS samples anew for each test example to avoid correlating their errors.

We define the function $n(h)$ to return the number of features hypothesis (of base learner) h needs to examine before deciding its prediction. In this paper, we focus our attention to decision stumps, for which, for all h_i , $n(h_i) = 1$.

Algorithm 2 AdaBoostRS

Given: $(x_1, y_1), \dots, (x_m, y_m)$

where $x_i \in X$, $y_i \in Y = \{-1, +1\}$

Generate $(\alpha_1, h_1), \dots, (\alpha_T, h_T)$ using Algorithm 1.

Given: test example x , a bound N .

Initialize: $p(i) = \frac{\alpha_i}{\sum_{i=1}^T \alpha_i}$, $k = 0$, $v = 0$, $F = \emptyset$.

while

$$k + \max_{1 \leq i \leq T} n(h_i) < N$$

do

 choose h_i according to $p(i)$

 let F_{h_i} be the set of features used by h_i

 let $l = |F_{h_i} \setminus F|$

$k = k + l$

$F = F \cup F_{h_i}$

 if $h_i(x) = 1$ then $v = v + 1$, else $v = v - 1$.

end while

Predict: $\text{sign}(v)$.

3.2. Margin Bound

Schapire et al. (1998) derived a bound on the generalization error of a voting classifier based on its training margins. The margin of example (x, y) depends on the votes $h_t(x)$ with weights α_t of all the hypotheses:

$$\text{margin}(x, y) = \frac{y \sum_t \alpha_t h_t(x)}{\sum_t \alpha_t}.$$

The magnitude of the margin represents the strength of agreement of the base classifiers, and its sign indicates whether the combined vote produces a correct prediction. Then, using the margins, one can bound the generalization error as follows.

Theorem 1 (margin bound (Schapire et al., 1998)). Let D be a distribution over $X \times \{-1, +1\}$ and let S be a sample of m examples chosen independently at random according to D . Suppose the base-classifier space H has VC-dimension d , and let $\delta \in (0, 1]$. Then with probability at least $1 - \delta$, every function f taking a weighted average over hypotheses in H satisfies the following bound for all $\theta > 0$

$$\mathbf{P}_D[yf(x) \leq 0] \leq \mathbf{P}_S[yf(x) \leq \theta] + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right).$$

We notice that this margin bound depends most heavily on the margins near the bottom of the distribution, since having generally high smallest margins allows θ to be small without $\mathbf{P}_S[yf(x) \leq \theta]$ getting too large.

Before getting to a consequence of Theorem 1, we will need the following bound.

Theorem 2 (Hoeffding’s inequality (1963)). Let X_1, \dots, X_n be independent real-valued random variables such that each $X_i \in \{a_i, b_i\}$ and let $S = \sum_{i=1}^n X_i$. Then for every $t > 0$,

$$\mathbf{P}[S - \mathbf{E}[S] < -t] \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right)$$

and also

$$\mathbf{P}[S - \mathbf{E}[S] > t] \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right).$$

Next we present a Corollary of Theorem 1, a version of which also appears within the margin bound proof in (Schapire et al., 1998).

Corollary 3. Under conditions of Theorem 1, if Algorithm 2 is run, taking N samples of hypotheses, then with probability at least $1 - \delta$,

$$\begin{aligned} \mathbf{P}_D[yf(x) \leq 0] &\leq \mathbf{P}_S[yf(x) \leq \theta] + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right) \\ &\quad + e^{-N\theta^2/2}. \end{aligned}$$

Proof. We examine the probability the sampling procedure in Algorithm 2 will yield a different result than taking the full vote from boosting and count any disagreement as an error.

We notice that because each hypothesis gives a binary vote and each weighted sampling is a random variable in $X \in \{-1, +1\}$, of mean θ . Let X_i be the i th vote, and let $U = \sum_{i=1}^N X_i$. Hence, $\mathbf{E}[U] = N\theta$.

The probability their unweighted average is ≤ 0 can be bounded by Hoeffding’s inequality.

$$\begin{aligned} \mathbf{P}[|U - \mathbf{E}[U]| \geq N\theta] &\leq e^{-\frac{2(N\theta)^2}{4N}} \\ &= e^{-N\theta^2/2}. \end{aligned}$$

This gives the additional error term and completes the proof. \square

In general, given the margins, this technique can be used to bound the number of disagreements in prediction between AdaBoost and AdaBoostRS. We see that $N = \Theta(1/\theta^2)$ is sufficient for the guarantee in Theorem 1.

When Algorithm 2 runs with decision stumps as the base hypotheses, we can be more concrete about the analysis. A decision stump is a one-node decision tree, which is a base learner that uses only one feature to make its prediction. If an example x has n features, then there are $2n$ possible decision stumps, one for the presence and absence of each feature. There, we can replace d with $\ln(n)$ in the margin bound – and also know that the number of samples we can take bounds the number of features that can be examined.

We also note the existence of similar (and sometimes stronger) margin bounds (see Breiman(1999), Langford et al. (2001), Wang et al. (2008)). These carry similar forms and use many of the same techniques as (Schapire et al., 1998) – Corollary 3 could be modified to give other margin bounds for AdaBoostRS.

Finally, this algorithm is easy to run online by using an online boosting algorithm instead of AdaBoost.

3.3. Balls and Bins

Another advantage of this method is that often a hypothesis will be selected that can be evaluated by looking at features that have already been examined. If we take k samples uniformly at random from n features, then the probability a given feature is not sampled is

$$\left(1 - \frac{1}{n}\right)^k = \left(1 - \frac{1}{n}\right)^{nk/n} \leq \left(\frac{1}{e}\right)^{k/n}$$

Therefore the expected number of unsampled features is $n\left(\frac{1}{e}\right)^{k/n}$. Thus, even if we draw $k = n$ samples, we still are expected to have left a constant fraction of the features unsampled.

Substituting $k = \Theta(1/\theta^2)$, we get $n\left(\frac{1}{e}\right)^{1/n\theta^2}$, which means that if θ is fixed, even as we have more features, we need to sample a vanishing fraction to meet the full margin bound results.

Table 1. Dataset sizes, and numbers of features, for training and test.

	census	splice	ocr 17	ocr 49
training	1000	1000	1000	1000
test	5000	2175	5000	5000
num features	130	239	402	402

Of course, the α distribution is not uniform on the decision stumps, and therefore on the features as well. This only improves the situation, as it is not hard to see that the uniform distribution on features is the most pessimistic case for our analysis. We shall see in Section 4 that the number of features not looked at is significantly better than this bound.

3.4. Observations

We observe that Algorithm 2 has some other nice properties. For one, it does not need to know the limit on number of features in advance. In the medical setting, it gives a procedure for running tests until the patient refuses any more or until time runs out.

One advantage of AdaBoostRS over (Cesa-Bianchi et al., 2010) is that it, like AdaBoost, can be a non-linear predictor if, say, decision trees (instead of stumps) are used as the base classifier. Assuming the trees are of small depth, the algorithm should remain feature-efficient and still be able to exploit the power of non-linearity.

4. Experiments

4.1. Data

We considered the following datasets: census, splice, ocr17, and ocr49, all available from the UCI repository. The splice dataset was modified to collapse the two splice categories into one to create binary-labeled data.

Also, ocr17 and ocr49 contain randomly chosen subsets of the NIST database of handwritten digits consisting only of the digits 1 and 7, and 4 and 9 (respectively); in addition, the images have been scaled down to 14×14 pixels, each with only four intensity levels. Table 1 shows the number of training and test examples used in each.

4.2. Results

For all our experiments, we ran AdaBoost and AdaBoostRS for 500 rounds. Figures 1 and 2 (log plot) show that the error rate of AdaBoostRS falls exponentially in the number of samples it uses, which is predicted by Corollary 3. Figure 1 also makes clear

that AdaBoostRS’s error rate quickly asymptotes to the error rate of AdaBoost, and also that the number of features used does not rise as quickly as the number of samples taken, which is behavior that we would expect from our analysis in Section 3.3

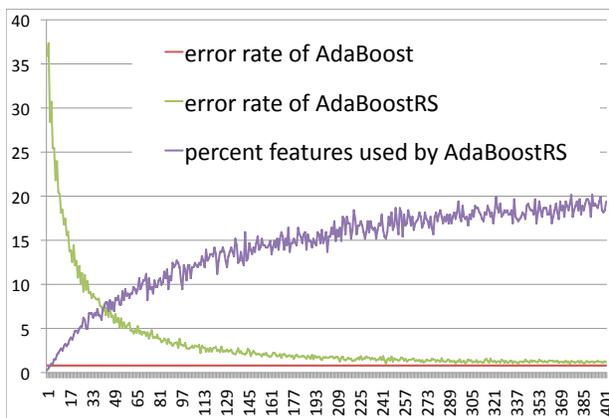


Figure 1. A graph of the error rate of AdaBoostRS on the ocr17 dataset and the percent of features it is using. The horizontal axis is the number of samples drawn by AdaBoostRS.

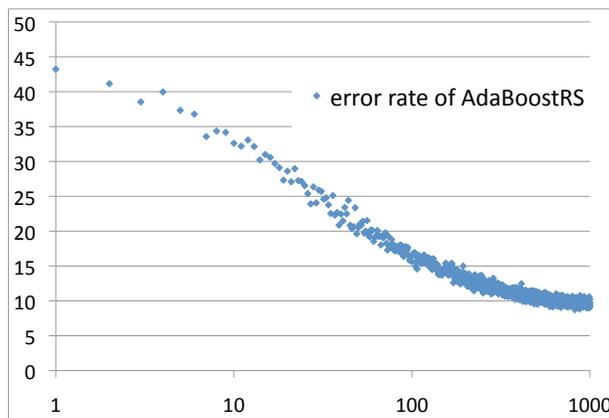


Figure 2. A graph of the error rate of AdaBoostRS on the splice dataset, as a function of the number of samples. The horizontal axis is on a log scale.

In Table 2 we examine the number of samples needed to achieve the given relative error rates compared to AdaBoost. Table 3 shows the percent of the features that were used to achieve the relative error rates of AdaBoost. It is clear that, at least for these datasets, the α distribution is sufficiently lumpy (far from uniform) as to give AdaBoostRS good relative error rates to AdaBoost without using too many features, even when taking many samples.

Table 2. Number of Samples Needed to Reach Given Relative Error Rates to AdaBoost.

	100%	50%	25%	10%
census (18.3)	15	77	211	637
splice (8.1)	97	205	421	823
ocr17 (0.8)	178	244	339	505
ocr49 (6.5)	167	296	694	1020

Table 3. Percent of Features Used to Reach Given Relative Error Rates to AdaBoost.

	100%	50%	25%	10%
census (18.3)	10.0	29.2	40.8	48.5
splice (8.1)	26.8	38.9	52.7	61.5
ocr17 (0.8)	16.4	18.4	19.9	20.6
ocr49 (6.5)	21.6	26.4	31.3	32.3

5. Discussion

In this work, we examined a boosting approach to tackling the feature-efficient learning problem. We leave open many interesting problems:

- What is the best boosting algorithm to use given this framework?
- Is this method more effective on a sparse feature set?
- How do we handle non-uniform feature costs?
- We noticed, in experiments, that deterministically using the top-weighted features performs well, but this method is not theoretically justified – can we compare the two?
- How does our work compare with the recent developments by Cesa-Bianchi et al. (2010)?

Finally, ever since the appearance of the margin bounds in Schapire et al. (1998), it has been a natural question how well a hypothesis-sampling algorithm would perform. This work begins an examination of this question.

6. Acknowledgments

I thank Rob Schapire for helpful discussions and the anonymous reviewers for useful suggestions.

References

Ben-David, Shai and Dichterman, Eli. Learning with restricted focus of attention. In *COLT*, pp. 287–296, New York, NY, USA, 1993. ACM.

Breiman, Leo. Prediction games and arcing classifiers. *Neural Computation*, 11(7):1493–1517, 1999.

Cesa-Bianchi, Nicolò, Shalev-Shwartz, Shai, and Shamir, Ohad. Efficient learning with partially observed attributes. *CoRR*, abs/1004.4421, 2010.

Chernoff, Herman. *Sequential Analysis and Optimal Design*. SIAM, 1972.

Freund, Yoav and Schapire, Robert E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

Globerson, Amir and Roweis, Sam T. Nightmare at test time: robust learning by feature deletion. In *ICML*, pp. 353–360, 2006.

Greiner, Russell, Grove, Adam J., and Roth, Dan. Learning cost-sensitive active classifiers. *Artif. Intell.*, 139(2):137–174, 2002.

Grove, Adam J. and Schuurmans, Dale. Boosting in the limit: Maximizing the margin of learned ensembles. In *AAAI/IAAI*, pp. 692–699, 1998.

Hoeffding, Wassily. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

Langford, John, Seeger, Matthias, and Megiddo, Nimrod. An improved predictive accuracy bound for averaging classifiers. In *ICML*, pp. 290–297, 2001.

Mason, Llew, Bartlett, Peter L., and Baxter, Jonathan. Direct optimization of margins improves generalization in combined classifiers. In *NIPS*, pp. 288–294, 1998.

Reyzin, Lev and Schapire, Robert E. How boosting the margin can also boost classifier complexity. In *ICML*, pp. 753–760, 2006.

Schapire, Robert E. The boosting approach to machine learning: An overview. In *Nonlinear Estimation and Classification*. Springer, 2003.

Schapire, Robert E., Freund, Yoav, Bartlett, Peter, and Lee, Wee Sun. Boosting the margin: A new explanation for the effectiveness of voting methods. *the Annals of Statistics*, 26(5):1651–1686, 1998.

Wald, Abraham. *Sequential Analysis*. Wiley, 1947.

Wang, Liwei, Sugiyama, Masashi, Yang, Cheng, Zhou, Zhi-Hua, and Feng, Jufu. On the margin explanation of boosting algorithms. In *COLT*, pp. 479–490, 2008.