

Boosting on a Budget: Sampling for Feature-Efficient Prediction

Lev Reyzin

Georgia Institute of Technology

Motivation and Setting

Looking at features is often costly.
tests in medicine (can be dangerous)
examining document features (time consuming)

Features have different associated costs.
some tests are more expensive than others

Our goal is **feature-efficient prediction**:
predicting without the total cost of features
examined exceeding a budget B .

Our Idea

Our idea: train a full ensemble (like AdaBoost),
then to predict, sample from the weak learner
distribution in test time.

Inspired by the margin bound of Schapire et al.
(’98); its proof uses a sampling procedure.

Cesa-Bianchi et al. (’10) studied feature-
efficient learning of linear predictors. Our
method, however, allows for non-linearity
because the weak learners can be arbitrary.

Our work builds on the margins theory and
answers the natural question of how effective a
sampling procedure can be.

Related to sequential analysis (Wald ’47), etc.

AdaBoostRS_{AC}

Algorithm 3 AdaBoostRS_{AC} (Arbitrary Cost)

Given: $(x_1, y_1), \dots, (x_m, y_m)$
where $x_i \in X, y_i \in Y = \{-1, +1\}$.
Given feature costs: $c_1, \dots, c_n \geq 1$.
Generate $(\alpha_1, h_1), \dots, (\alpha_T, h_T)$ using AdaBoost

Given: test example x , a bound B .

Initialize:

$$p(i) = \frac{\alpha_i}{c(h_i) \sum_{i=1}^T (\alpha_t / c(h_t))}$$

and $k = 0, F = \emptyset, \tau = 0$ and $v_0(x) = 0$.

while

$$k + \max_{1 \leq i \leq T} c(h_i) < B$$

do

choose h_i according to $p(i)$

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

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

$k = k + \sum_{x^j \in l} c_j$

$F = F \cup F_{h_i}$

if $h_i(x) = 1$ **then**

$v_{\tau+1}(x) = v_{\tau}(x) + c(h_i)$

else

$v_{\tau+1}(x) = v_{\tau}(x) - c(h_i)$.

end if

increment τ by 1

end while

Predict: $\text{sign}(v_{\tau})$.

this provably works (see paper)
convergence rate a function of **margins**

Experimental Results

1) exponential convergence: as theory predicts

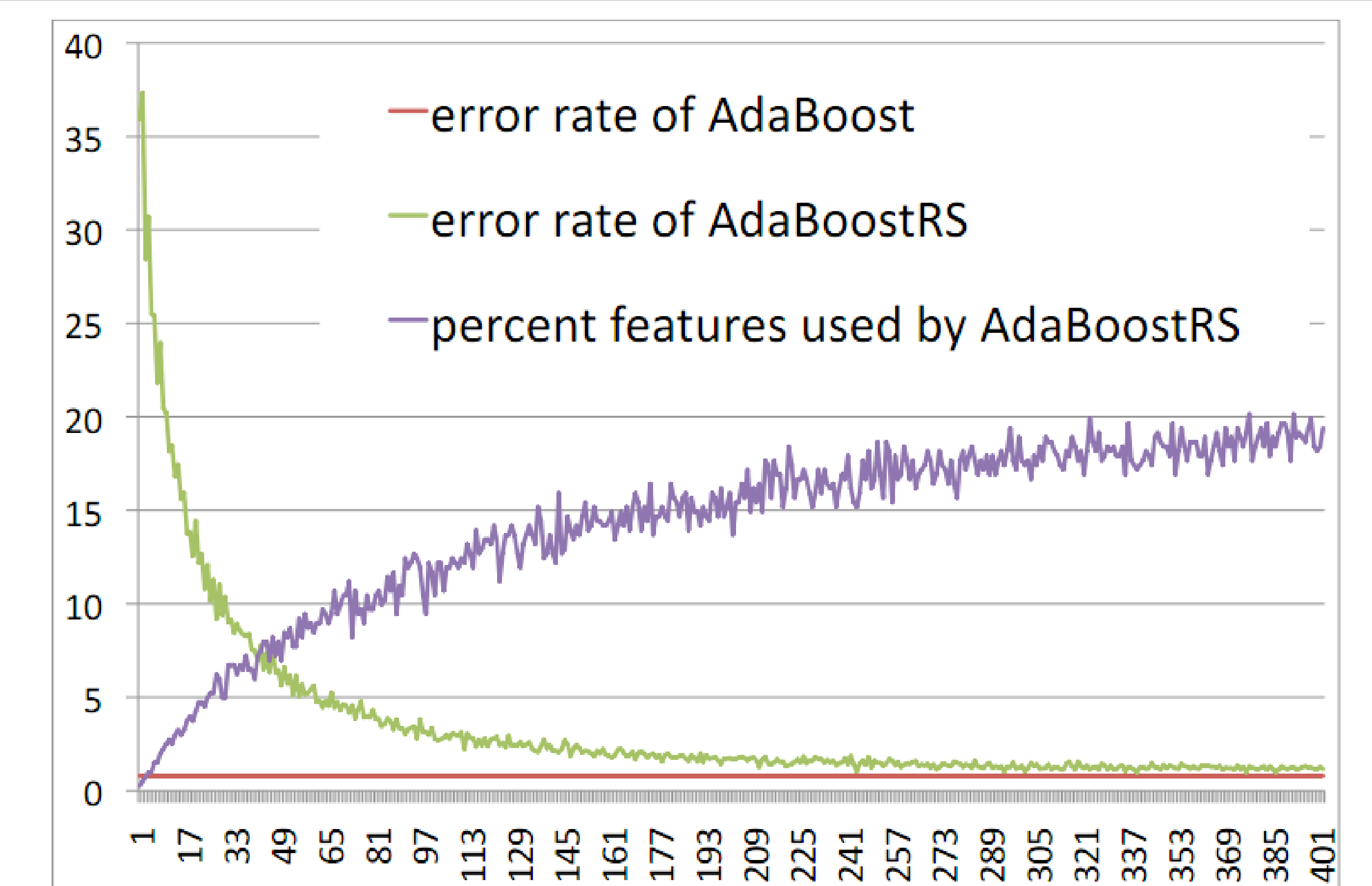


Figure 1. A graph of the error rate of AdaBoostRS on the ocr17 dataset and the percent of features it is using. The horizontal axis is the number of samples drawn by AdaBoostRS.

2) biasing / importance sampling (by cost) helps

Table 5. Error rates (in percent), averaged over 50 trials, of AdaBoostRS_{AC} and AdaBoostRS using budgets of 10 and 20 when features have random costs drawn i.i.d. from $[0, 1]$. The underlying algorithm, AdaBoost, is run for 500 rounds.

	AdaBoostRS _{AC}	AdaBoostRS
census ($B = 11$)	32.2	32.8
census ($B = 21$)	25.5	26.4
splice ($B = 11$)	25.7	27.0
splice ($B = 21$)	19.2	20.4
ocr17 ($B = 11$)	9.2	10.5
ocr17 ($B = 21$)	3.5	4.3
ocr49 ($B = 11$)	27.4	28.3
ocr49 ($B = 21$)	20.2	21.4