ELSEVIER

Pattern Recognition Letters 22 (2001) 413–419

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

A Bayesian boosting theorem

Richard Nock [a,*], Marc Sebban [b]

[a] *Département Scientifique Interfacultaire, Université des Antilles-Guyane, Campus de Schoelcher, 97278 Schoelcher, France*
[b] *Département d'Economie et Sciences Juridiques, Université des Antilles-Guyane, Campus de Fouillole, 97159 Pointe-à-Pitre, France*

Received 3 March 2000; received in revised form 29 September 2000

Abstract

We refine the first theorem of (R.E. Schapire, Y. Singer, in: Proceedings of the 11th Annual ACM Conference on Computational Learning Theory, 1998, pp. 80–91) bounding the error of the AdaBoost boosting algorithm, to integrate Bayes risk. This suggests the significant time savings could be obtained on some domains without damaging the solution. An applicative example is given in the field of feature selection. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Boosting; Bayes risk; Risk bounds; Feature selection

1. Introduction

Boosting is related to a general methodology in constructing classifiers, that is, functions mapping observations to classes. It is concerned by the combination of moderately accurate, "weak" classifiers (or hypotheses), into a highly accurate, "strong" one. Historically, one of the very first attempts to combine weak hypotheses in that way is due to Schapire (1990). Since then, the method has received much theoretical and practical attentions, and recently (Schapire and Singer, 1998), an important breakthrough established theoretical bases of an algorithm called AdaBoost, one of the most cited in that field. AdaBoost builds weighted linear combinations of weak hypotheses, and gives a theoretically efficient solution to all stages of the process: on which criterion and data to train weak hypotheses, how to choose their weighting coefficients, even how to interpret their outputs in terms of classification and confidence. But perhaps the most interesting feature of AdaBoost is a mechanism for weighting examples, which consists in growing the current weak hypothesis on a set of examples that were hard to classify for its predecessors. Also, it shows that optimizing the accuracy of the overall hypothesis can be done efficiently by optimizing the growth of each weak hypothesis on a criterion Z being *not* the accuracy (its name, Z, comes from the fact that it is actually the normalization coefficient of a particular distribution). Interestingly, this criterion was previously used in decision tree induction to compare various "top-down" induction schemes, in which a tree is grown by repetitively replacing leaves by logical tests (Kearns and Mansour, 1996). In that

* Corresponding author. Fax: +596-72-73-62.
E-mail addresses: rnock@martinique.univ-ag.fr (R. Nock), msebban@univ-ag.fr (M. Sebban).

PII: S0167-8655(00)00137-9

work, Z was theoretically shown to be the criterion leading to the optimal maximization of the accuracy (Kearns and Mansour, 1996).

Our contribution in this paper relies on a more explicit version of the first theorem of Schapire and Singer (1998), the main theorem proving the efficiency of AdaBoost, as well as the basis for the construction of the Z criterion to optimize. Our version encompasses cases where many examples of different classes share the same description, implying that Bayes optimal classification rule does not have zero error. It also shows that a slight modification of the Z criterion, in the examples distribution, is likely to give better convergence speeds, without degradation of the final results. These cases regarding non-zero Bayes optimum are interesting in real-world classification problems, as well as in general domains such as feature selection, for which we provide some results obtained.

2. Boosting with multiply matching observations

Let $LS = \{(\boldsymbol{x}_1, y(\boldsymbol{x}_1)), (\boldsymbol{x}_2, y(\boldsymbol{x}_2)), \ldots, (\boldsymbol{x}_m, y(\boldsymbol{x}_m))\}$ be a sequence of training examples, where each observation (or description) $\boldsymbol{x}_i$ belongs to X, and each label $y(\boldsymbol{x}_i)$ belongs to a finite label space Y. We suppose that there are only two classes, in order not to harden the proofs: classes in Y are denoted "+" and "−" and called respectively the positive and the negative class. This does not restrict in any way the utility of our result for multiclass problems: we shall explain why these problems actually boil down to particular two-classes problems.

For any description $\boldsymbol{x}$ over X, define $n^+(\boldsymbol{x})$ (resp. $n^-(\boldsymbol{x})$) to be the number of positive (resp. negative) examples having the description $\boldsymbol{x}$. Define $\delta(\boldsymbol{x}) = |n^+(\boldsymbol{x}) - n^-(\boldsymbol{x})|/(n^+(\boldsymbol{x}) + n^-(\boldsymbol{x}))$. The optimal class prediction for some description $\boldsymbol{x}$ in LS is the class $\arg\max_{c\in\{+,-\}} n^c(\boldsymbol{x})$, which we write $y^*(\boldsymbol{x})$ for short. Finally, for some predicate P, define as $[\![P]\!]$ to be 1 if P holds, and 0 otherwise; define as $\pi(\boldsymbol{x}, \boldsymbol{x}')$ to be the predicate "$\boldsymbol{x}'$ and $\boldsymbol{x}$ are identical descriptions", for arbitrary descriptions $\boldsymbol{x}$ and $\boldsymbol{x}'$.

Suppose that each weak hypothesis is returned by a *weak learner*, receiving as input LS and a distribution D over S. The output of the weak learner is a weak hypothesis $h_t : X \to [-1, 1]$. In the bi-class setting, the sign of the output represents the class. Fig. 1 presents the original AdaBoost algorithm, as described in (Schapire and Singer, 1998). Coefficients α_t are scaling coefficients, to map the votes of h_t to $\mathbb{R}$ itself. The absolute magnitude of a vote may be interpreted as a confidence (Schapire and Singer, 1998). The update in the distribution is important as it leads to prove that the unweighted training error is upperbounded by the product of the Z_t (Schapire and Singer, 1998, Theorem 1), which we state for completeness.

AdaBoost (**Set of examples** $\{(\mathbf{x}_i, y(\mathbf{x}_i))\}_{i=1}^m$):
Initialize:
$\forall i \in \{1, 2, ..., m\}, D_1((\mathbf{x}_i, y(\mathbf{x}_i))) = \frac{1}{m}$;
for $t = 1, 2, ..., T$
Train weak learner using D_t;
Get weak hypothesis $h_t : X \to \mathbb{R}$;
Choose α_t:
$\alpha_t = \frac{1}{2}\log\left(\frac{1+r_t}{1-r_t}\right)$
$r_t = \sum_{i=1}^m D_t((\mathbf{x}_i, y(\mathbf{x}_i)))y(\mathbf{x}_i)h_t(\mathbf{x}_i)$
Update:
$\forall i \in \{1, 2, ..., m\}, D_{t+1}((\mathbf{x}_i, y(\mathbf{x}_i))) = \frac{D_t((\mathbf{x}_i, y(\mathbf{x}_i)))e^{-\alpha_t y(\mathbf{x}_i)h_t(\mathbf{x}_i)}}{Z_t}$;
endfor
return final hypothesis: $H(\mathbf{x}) = \mathrm{sign}\left(\sum_{t=1}^T \alpha_t h_t(\mathbf{x})\right)$

Fig. 1. AdaBoost as described in (Schapire and Singer, 1998). Z_t is the normalization factor for distribution D_{t+1}.

Theorem 1 (Schapire and Singer, 1998). *The following bound holds on the training error of H*:

$$\frac{|\{\boldsymbol{x} : H(\boldsymbol{x}) \neq y(\boldsymbol{x})\}|}{m} \leqslant \prod_t Z_t.$$

Actually, as we now show, this theorem can be generalized.

Theorem 2 (Generalization of Schapire and Singer, 1998, Theorem 1). *Using* AdaBoost, *the following bound holds on the training error of H*:

$$\frac{|\{\boldsymbol{x} : H(\boldsymbol{x}) \neq y(\boldsymbol{x})\}|}{m} \leqslant \left(\prod_t Z_t\right) \times \boldsymbol{E}_{D_{T+1}}[\delta(\boldsymbol{x})] + \epsilon^*$$

where ϵ^* is the minimal error on S, and $\boldsymbol{E}_{D_{T+1}}[\delta(\boldsymbol{x})]$ is the expectation of $\delta(\boldsymbol{x})$ on distribution D_{T+1}.

Proof. We unravel the update rule, and get for any $\boldsymbol{x}'$

$$D_{T+1}(\boldsymbol{x}') = \frac{\mathrm{e}^{-y(\boldsymbol{x}')\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x}')}}{m\prod_t Z_t}. \tag{1}$$

Denote as $LS^* \subseteq LS$ to be the set of examples containing for any possible observation $\boldsymbol{x}$ in LS, one example $(\boldsymbol{x}, y^*(\boldsymbol{x}))$. Remark that if no two examples share the same description, then $LS^* = LS$. Otherwise, $LS^* \subset LS$. Fix some $(\boldsymbol{x}, y^*(\boldsymbol{x})) \in LS^*$. It can be seen that we have:

$$\begin{aligned}\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS} [\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!] \times [\![H(\boldsymbol{x}') \neq y(\boldsymbol{x}')]\!] &= [\![H(\boldsymbol{x}) \neq y^*(\boldsymbol{x})]\!] \times \max\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\} \\ &\quad + (1 - [\![H(\boldsymbol{x}) \neq y^*(\boldsymbol{x})]\!]) \min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\} \\ &= [\![H(\boldsymbol{x}) \neq y^*(\boldsymbol{x})]\!] \times \delta(\boldsymbol{x})(n^-(\boldsymbol{x}) + n^+(\boldsymbol{x})) + \min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}.\end{aligned}$$

Fix $\epsilon = (1/m)|\{\boldsymbol{x} : H(\boldsymbol{x}') \neq y(\boldsymbol{x}')\}|$ for short. We obtain

$$\begin{aligned}\epsilon &= \frac{1}{m} \sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS} [\![H(\boldsymbol{x}') \neq y(\boldsymbol{x}')]\!] \\ &= \frac{1}{m} \sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} \sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS} [\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!] \times [\![H(\boldsymbol{x}') \neq y(\boldsymbol{x}')]\!] \\ &= \frac{1}{m} \sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} [\![H(\boldsymbol{x}) \neq y^*(\boldsymbol{x})]\!] \times \delta(\boldsymbol{x})(n^-(\boldsymbol{x}) + n^+(\boldsymbol{x})) + \epsilon^*.\end{aligned}$$

Here, we have made use of the fact that

$$\epsilon^* = \frac{1}{m} \sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} \min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}.$$

Now, remark that $\forall(\boldsymbol{x}, y^*(\boldsymbol{x})) \in LS^*$,

$$[\![H(\boldsymbol{x}) \neq y^*(\boldsymbol{x})]\!] \leqslant \sum_{\boldsymbol{x}'\in LS} [\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!] \frac{\mathrm{e}^{-y(\boldsymbol{x}')\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x}')}}{n^-(\boldsymbol{x}) + n^+(\boldsymbol{x})}. \tag{2}$$

If the left-hand side is false, the inequality is true since the right-hand side is always positive. Otherwise, suppose that $H(\boldsymbol{x}) \neq y^*(\boldsymbol{x})$. There are $n^-(\boldsymbol{x}) + n^+(\boldsymbol{x})$ examples projecting onto $\boldsymbol{x}$, those for which $\pi(\boldsymbol{x}, \boldsymbol{x}')$ is

true. Any example $\boldsymbol{x}'$ participating to the increase of $\max\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}$ satisfies $H(\boldsymbol{x}') \neq y(\boldsymbol{x}')$, since $y(\boldsymbol{x}') = y^*(\boldsymbol{x})$, and therefore $\mathrm{e}^{-y(\boldsymbol{x}')\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x}')} \geqslant 1$. Fix $a = -y(\boldsymbol{x})\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x}) \geqslant 0$. The other examples $\boldsymbol{x}'$, participating to the increase of $\min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}$, correspond to a value of the exponential which is $\mathrm{e}^{-a} \leqslant 1$. We obtain

$$\begin{aligned}\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS} [\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\mathrm{e}^{-y(\boldsymbol{x}')\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x}')} &= \max\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}\mathrm{e}^{a} + \min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}\mathrm{e}^{-a} \\ &= (\max\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\} - \min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\})\mathrm{e}^{a} \\ &\quad + \min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}(\mathrm{e}^{a} + \mathrm{e}^{-a}) \geqslant (\max\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\} \\ &\quad - \min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\}) + 2\min\{n^-(\boldsymbol{x}), n^+(\boldsymbol{x})\} \geqslant n^-(\boldsymbol{x}) + n^+(\boldsymbol{x})\end{aligned}$$

thus, the right-hand side of inequality (2) is $\geqslant 1$. We now finish to upperbound the error

$$\begin{aligned}\epsilon &\leqslant \frac{1}{m}\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*}\ \sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS} [\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\mathrm{e}^{-y(\boldsymbol{x}')\sum_{t=1}^{T}\alpha_t h_t(\boldsymbol{x}')} \times \delta(\boldsymbol{x}) + \epsilon^* \\ &= \sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}\left(\prod_t Z_t\right) \times D_{T+1}(\boldsymbol{x}') \times \left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} [\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right) + \epsilon^* \\ &= \left(\prod_t Z_t\right) \times \sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} D_{T+1}(\boldsymbol{x}')[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right) + \epsilon^* = \left(\prod_t Z_t\right) \times \boldsymbol{E}_{D_{T+1}}[\delta(\boldsymbol{x})] + \epsilon^*.\end{aligned}$$

This concludes the proof. □

As pointed out by Schapire and Singer (1998), Theorem 1 shows that the minimization of the error is ensured by minimizing each Z_t. Of course, their theorem is valid when considering datasets with non-zero Bayes optimum, but their formula is much less explicit in that case, particularly because it does not unveil ϵ^*, itself a necessary lowerbound for the error. Since this theorem is crucial to formulate the existence of the Z_t criterion, a more explicit formulation is likely to yield better results. Actually, in our case, a faster convergence rate can be expected by optimizing the diminution of the error at time t, ϵ_t, as follows. We optimize the ratio $(\epsilon_t - \epsilon^*)/(\epsilon_{t-1} - \epsilon^*) = Z'_t$, since ϵ^* is a lowerbound for the error on LS. We have

$$\begin{aligned}Z'_t &= Z_t\frac{\boldsymbol{E}_{D_{t+1}}[\delta(\boldsymbol{x})]}{\boldsymbol{E}_{D_t}[\delta(\boldsymbol{x})]} \\ &= Z_t\frac{\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} D_{t+1}((\boldsymbol{x}', y(\boldsymbol{x}')))[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right)}{\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} D_t((\boldsymbol{x}', y(\boldsymbol{x}')))[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right)} \\ &= \frac{\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} D_t((\boldsymbol{x}', y(\boldsymbol{x}')))\mathrm{e}^{-y(\boldsymbol{x}')(\alpha_t h_t(\boldsymbol{x}'))}[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right)}{\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} D_t((\boldsymbol{x}', y(\boldsymbol{x}')))[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right)} \\ &= \frac{\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}\mathrm{e}^{-y(\boldsymbol{x}')(\alpha_t h_t(\boldsymbol{x}'))}D_t((\boldsymbol{x}', y(\boldsymbol{x}')))\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*}[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right)}{\sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS}D_t((\boldsymbol{x}', y(\boldsymbol{x}')))\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*}[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})\right)} \\ &= E_{\boldsymbol{x}', D'_t}\left[\mathrm{e}^{-y(\boldsymbol{x}')(\alpha_t h_t(\boldsymbol{x}'))}\right]\end{aligned}$$

with

$$\forall(\boldsymbol{x}', y(\boldsymbol{x}')) \in LS, D'_t((\boldsymbol{x}', y(\boldsymbol{x}'))) = \frac{\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} D_t((\boldsymbol{x}', y(\boldsymbol{x}')))[\![\pi(\boldsymbol{x}, \boldsymbol{x}')]\!]\delta(\boldsymbol{x})}{\sum_{(\boldsymbol{x}'', y(\boldsymbol{x}''))\in LS}\left(\sum_{(\boldsymbol{x}, y^*(\boldsymbol{x}))\in LS^*} D_t((\boldsymbol{x}'', y(\boldsymbol{x}'')))[\![\pi(\boldsymbol{x}, \boldsymbol{x}'')]\!]\delta(\boldsymbol{x})\right)}.$$

In other words, we should strive to minimize a weighted expectation with distribution favoring the examples $(\boldsymbol{x}', y(\boldsymbol{x}'))$ for which the conditional distribution of the examples projecting onto it is greatly in favor of one class against the others. Of course, when each possible observation belongs to one class (i.e., no information is lost among the examples), the expectation is exactly Schapire–Singer's Z_t.

The determination of α'_t and r'_t to replace α_t and r_t in AdaBoost can be done by minimizing Z'_t, similarly to (Schapire and Singer, 1998). We obtain the following theorem, which gives formal expressions as well as a result on the convergence towards the minimal error on S (the proof follows exactly that of Schapire and Singer, 1998).

Theorem 3. *Assume that each h_t has range $[-1;+1]$, and that we choose $\alpha'_t = (1/2)\log((1+r'_t)/(1-r'_t))$, where $r'_t = \sum_{(\boldsymbol{x}', y(\boldsymbol{x}'))\in LS} D'_t((\boldsymbol{x}', y(\boldsymbol{x}')))y(\boldsymbol{x}')h_t(\boldsymbol{x}')$, i.e., $r'_t = \boldsymbol{E}_{D'_t}[y(\boldsymbol{x}')h_t(\boldsymbol{x}')]$. We obtain*

$$Z_t = \sqrt{1-(r'_t)^2}. \tag{3}$$

Then the training error ϵ_{T} of H after T rounds of boosting satisfies

$$\epsilon_{\mathrm{T}} \leqslant \prod_{t=1}^{T}\sqrt{1-(r'_t)^2} + \epsilon^*. \tag{4}$$

The extension to multiclass, multilabels problems follows the original ideas of Schapire and Singer (1998). The key idea is to create a set of bi-class problems, each of which predicting one class against all others, and separately solved using AdaBoost. Predicting a single class for some observation boils down to taking the class corresponding to the greatest output over all formulas.

3. An application

The first application of Theorem 2 can be obtained in the field of feature selection (Sebban and Nock, 1999). In this domain, we seek to reduce the number of description variables (or *features*) of data, to increase its interpretability, reduce its noise, and reduce the complexity of post-processing. We relate here some of the experiments of Sebban and Nock (1999) to illustrate how Theorem 2 can be used, in a particular category of feature selection methods called Wrapper models. In these models, the quality of a feature subset is evaluated by the accuracy of a particular, core algorithm. Our approach, which is a forward selection method, can be summarized as follows: at each time, we add the feature to a current feature set (initialized to $\emptyset$) which increases the most the accuracy of a formula built using the core algorithm. If no addition of a new feature increases the accuracy, then we stop and return the current feature subset.

Increasing the speed of a boosting algorithm in the case of feature selection applications is of great interest. Indeed, in that case, the core algorithm is the boosting algorithm, and even if we can avoid constructing only a little number of classifiers at its level, this might represent significant time savings at the feature selection level, which repeatedly runs the boosting algorithm. In (Sebban and Nock, 1999), two types of experiments are ran. The first ones address the validity of the boosting approach for feature

selection, the second ones evaluate the benefits which can be obtained by using our modification to Schapire–Singer's Z criterion.

To evaluate the benefits that boosting can bring to feature selection, a first experiment carried out. It is a simple comparison of the accuracy, computed by a leave-one-out cross-validation, of a nearest neighbor's rule with the whole set of features, with the one selected by boosting, and with the one selected after a simple greedy maximization of the accuracy through feature subsets. Fig. 2 presents some results obtained on 19 datasets, most of which coming from the UCI repository of machine learning database. The results are a clear advocacy for the use of boosting even in the field of feature selection. The procedure obtains for example on 13 datasets out of 19 the best results over the three methods.

A second experiment evaluates the benefits which can be obtained by using Theorem 2 instead of the original Schapire–Singer criterion, more precisely to quantify the possible increase in convergence speed. ADABOOST's main controllable criterion is the number T of base learners required. We have checked experimentally that when using boosting for feature selection, increasing parameter T modifies the final solution up to a maximal value of T, after which the final subset of features is not modified anymore. This value $T_{\max}$ was computed for both Schapire–Singer's Z criterion and ours for each of the 10 datasets. A relative gain ρ was computed to evaluate the benefit of our method, namely

$$\rho = \frac{T_{\max}(\text{Schapire–Singer}) - T_{\max}(\text{us})}{T_{\max}(\text{us})}. \tag{5}$$

Table 1 summarizes the results obtained for the datasets. Again, it is clear from the results that our modification to Schapire–Singer's Z criterion provides very good results in the field of feature selection,

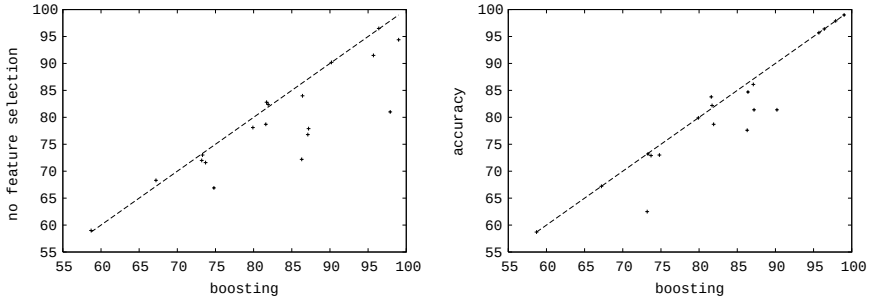

Fig. 2. Accuracy computed after boosting and without feature selection (left) and after boosting and simple accuracy optimization (right), on 19 datasets. Points below the $y = x$ line indicate better results for boosting.

Table 1
Value of relative gain ρ for 10 datasets

Dataset	ρ (%)
Bigpole	50
Echocardiogram	70
Hepatitis	25
Led	50
Heart	40
Hard1	40
Hard2	00
Vote	20
HorseColic	22
XD6	30

since gain ratios up to 70% (with 34.7% average) can be obtained, that is, the construction of up to 70% base learners can be saved. In many cases where base learners are time consuming to induce (typical examples are decision trees, lists, etc.) such a gain would be well worth the changing of Z using Theorem 2.

References

Kearns, M., Mansour, Y., 1996. On the boosting ability of top-down decision tree learning algorithms. In: Proceedings of the 28th Annual ACM Symposium on the Theory of Computing, pp. 459–468.

Schapire, R.E., 1990. The strength of weak learnability. Machine Learning.

Schapire, R.E., Singer, Y., 1998. Improved boosting algorithms using confidence-rated predictions. In: Proceedings of the 11th Annual ACM Conference on Computational Learning Theory, pp. 80–91.

Sebban, M., Nock, R., 1999. Contribution of boosting in wrapper models. In: Proceedings of the European Conference on Principles and Practice of KDD, pp. 214–222.