Using suboptimal base classifiers with AdaBoost

Chakshu Sardana
New York University
cs4511@nyu.edu

Vighnesh Birodkar
New York University
vighneshbirodkar@nyu.edu

1 Introduction

Adaboost Freund and Schapire [1999] is a popular machine learning algorithm which uses an ensemble of weak learners for prediction. If the weak learners are guaranteed to give better than random predictions for all distributions, then the training error of Adaboost decreases exponentially in terms of number of base classifiers used Schapire [2003]. Traditionally at each time step adaboost requests a function giving the least possible error from the weak learner. In this study we explore the strategy of not using the optimal base classifier with boosting at each time step. In particular, we study the effects of using a linear classifier and boosting stumps at each time step, both of which are suboptimal for the distribution provided by adaboost.

2 Adaboost

2.1 Weak learners

As defined in Mohri et al. [2012] concept class C is said to be weakly PAC-learnable if there exists an algorithm A, $\gamma > 0$, and a polynomial function $poly(\cdot, \cdot, \cdot, \cdot)$ such that for any $\epsilon > 0$ and $\delta > 0$, for all distributions D on X and for any target concept $c \in C$, the following holds for any sample size $m \geq poly(1/\epsilon, 1/\delta, n, size(c))$.

$$\Pr_{S \sim D^m}\left[R(h_s) \leq \frac{1}{2} - \gamma\right] \geq 1 - \delta \tag{1}$$

where $R(h_s)$ denotes the generalization error of h_s. The hypothesis returned by the weak learner is referred to as the base classifier.

2.2 Adaboost Algorithm

The Adaboost algorithm as given by Freund and Schapire [1999] and adapted from Mohri et al. [2012] is given in Algorithm 1

Let g be the function computed by adaboost in step 10 of Algorithm 1. Let H be the family of base classifiers and $\hat{R}_\rho(\cdot)$ be the empirical margin loss with margin ρ for a function. Then Mohri et al. [2012] show that the following holds

$$R(g) \leq \hat{R}_\rho(g/\|\alpha\|_1) + \frac{2}{\rho}\mathfrak{R}_m(H) + \sqrt{\frac{\log \frac{1}{\delta}}{2m}} \tag{2}$$

with probability at least $1 - \delta$, where $\mathfrak{R}_m$ is the expectation of empirical Radamacher complexity over all samples of size m. Mohri et al. [2012] also show that

$$\hat{R}_\rho(g/\|\alpha\|_1) \leq \left[(1+2\gamma)^{1+\rho}(1-2\gamma)^{1-\rho}\right]^{T/2} = f(\gamma)^{T/2} \tag{3}$$

Input: $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$ where $x_i \in \mathbb{R}$ and $y_i \in \{-1, +1\}$.

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

for $t = 1$ *to* T **do**

  Train base classifier with small error $\epsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$

  $\alpha_t \leftarrow \frac{1}{2} \log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$

  $Z_t \leftarrow 2[\epsilon_t(1-\epsilon_t)]^{1/2}$

  for $i = 1$ *to* m **do**

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

  end

end

$g = \sum_{t=1}^{T} \alpha_t h_t$

return $h = \text{sgn}(g)$

Algorithm 1: The Adaboost algorithm

2.3 Benefit of sub-optimal base classifiers

Taking the derivative with respect to γ of f we get

$$\begin{aligned}
\frac{df}{d\gamma} &= (1+2\gamma)^{\rho}(1+\rho)(2)(1-2\gamma)^{1-\rho} + (1-2\gamma)^{-\rho}(1-\rho)(-2)(1+2\gamma)^{1+\rho} \\
&= \Big[(1+2\gamma)^{\rho}(1-2\gamma)^{-\rho}\Big]\Big[2(1+\rho)(1-2\gamma) + (-2)(1-\rho)(1+2\gamma)\Big] \\
&= \Big[(1+2\gamma)^{\rho}(1-2\gamma)^{-\rho}\Big]2\Big[1-2\gamma+\rho-2\rho\gamma-1-2\gamma+\rho+2\rho\gamma\Big] \\
&= 4\Big[(1+2\gamma)^{\rho}(1-2\gamma)^{-\rho}\Big](\rho-2\gamma) \qquad (4)
\end{aligned}$$

Since $\gamma < \frac{1}{2}$ Equation 4 is positive for $\gamma < \rho/2$. This indicates that f is increasing for $\gamma < \rho/2$ and decreasing for $\gamma > \rho/2$. We know that at $f(0) = 1$ and $f(1/2) = 0$. This implies that in the interval $0 < \gamma < 1/2$, f rises upto the point $\gamma = \rho/2$ and then decreases till 0. Figure 3 illustrates this for difference choices of ρ. Therefore if the base classifier returns a hypothesis with a larger γ, it might increase the bound on the generalization error in Equation 3 if $\gamma < \rho/2$. Hence, it might be beneficial if the weak learner returns a hypothesis which does worse as compared to the optimal at each time step. On the other hand if the weak learner can return a $\gamma > \rho/2$, it should return with a γ as large as possible in order to decrease the bound as much as possible. It suffices to say that the the the best choice of γ depends on the margin that the algorithm can achieve on the given dataset.

3 Proposed Algorithm

3.1 Weak Linear classifier

The problem of finding a linear classifier with minimal classification error is NP-hard in terms of the dimensionality of the feature vector Johnson and Preparata [1978]. However, for any finite distribution of data, there exists a linear classifier with $> 1/2$ accuracy and it can be found in polynomial time as shown by Mannor and Meir [2001]. For any input distribution D on m points, the algorithm guarantees to find a and b with $\hat{y}_i = \text{sgn}(a \cdot x + b)$ such that $\sum_{i=1}^{m} D_i 1_{\hat{y}_i \neq y_i} \leq \frac{1}{2} - \frac{1}{4m}$. We use this algorithm as one of the base classifiers in our experiments.

3.2 Sub optimal boosting stumps

Single level decision trees or boosting stumps are not weak learners as is evident from the fact that they cannot do better than half on the XOR data. In spite of that boosting stumps have been shown

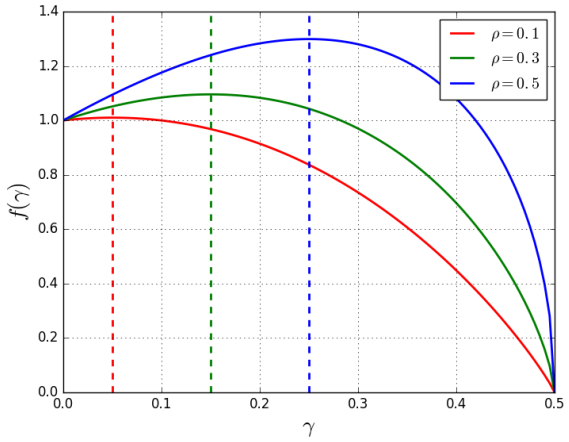


Figure 1: Plot of function f which is an upper bound on $\hat{R}(g/\|\alpha\|_1)$. The dotted line represents the maximum of each curve. To the left of the maximum, returning a sub-optimal base classifier will decrease the bound

to work well with adaboost Freund and Schapire [1996]. To study the effect of decreasing γ we consider sub-optimal boosting stumps. Instead of boosting stumps which select the best feature and best threshold to split each feature, we instead choose the 2nd, 3rd, or in general the k^{th} best feature and then perform an optimal split. This guarantees γ lesser than or equal to what would have been possible with an optimal split at each step.

4 Datasets

For the purpose of this study we used the `ocr` (with labels 1 and 7), `breastcancer`, `iris` (with labels Setosa and Versicolour) and `ionosphere` datasets from the UCI Machine Learning Repository at `https://archive.ics.uci.edu/ml/datasets.html`. The `digits`(with labels 4 and 9) dataset was obtained from the `scikit-learn` Pedregosa et al. [2011] software package. While measuring error for each time step, we divided the data into 10 sets of roughly equal size. We kept aside one set for validation and trained on the remaining 9 sets. This procedure was repeated 10 times, and we report the average error made by the classifier on each of the validation sets.

5 Experiments

5.1 Weak linear classifier

The convergence of adaboost using the weak linear classifier algorithm is extremely slow. Figure 2 shows the training error using the weak linear classifier as a base classifier. While boosting stumps provide an exponential drop as expected, using weak linear classifiers as base classifiers only produces a linear drop in training error. The poor convergence of the method can be explained by the bound on the training error of adaboost as shown by Mohri et al. [2012]

$$\hat{R}(h) \leq \exp(-2\gamma^2 T) \tag{5}$$

Since for the weak linear classifier the minimum value γ is $1/4m$, for large m, $\gamma \to 0$ and $\gamma^2 \to 0$ even faster. Since $\lim_{\gamma \to 0} \left(\exp(-2\gamma^2 T)\right) \approx 1 - 2\gamma^2 T$ the upper bound on the convergence is approximately linear. This is also seen in the experiment. It thus renders this approach impractical for real world data.

5.2 Sub optimal boosting stumps

We experimented with sub optimal boosting stumps with $k = 1, 2, 3$, i.e; choosing the best, second best and third best feature to classify using a boosting stump. The γ at each time step is plotted

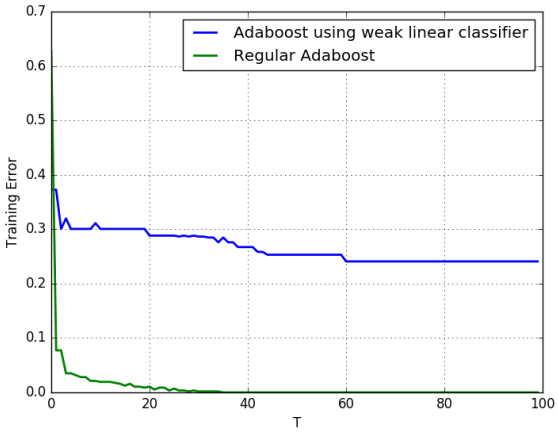


Figure 2: Convergence of Adaboost using weak linear classifier as compared to using boosting stumps on the `iris` dataset.

Dataset	**Min. Error**		
	k=1	**k=2**	**k=3**
`breastcancer`	0.021084(T=162)	**0.017575(T=65)**	0.038628(T=520)
`ionosphere`	0.062540(T=516)	0.065317(T=300)	0.076825(T=75)
`digits`	0.002703(T=13)	0.005556(T=101)	0.005556(T=110)
`ocr`	0.003863(T=33)	0.003879(T=106)	**0.002581(T=255)**

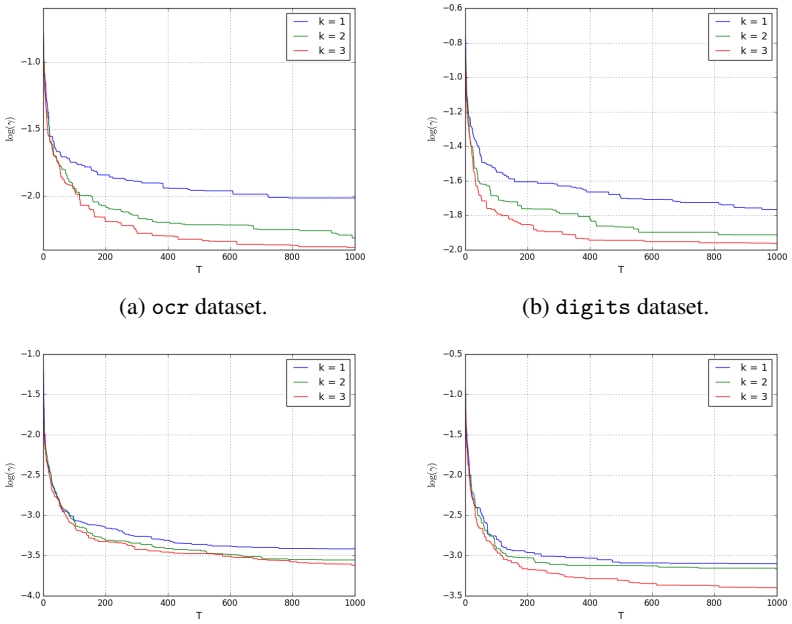
Table 1: Minimum cross validation error for each suboptimal stump along with the time step T which gave the minimum error.

in Figure 3. It can be clearly seen that γ decreases with the increase in k for all time steps. The variation of cross validation error with each time step in shown in Figure 4. In addition, Table 1 lists the least cross validation error for each dataset along with the time step T which achieved it. Suboptimal stumps with $k = 2$, and $k = 3$ achieve lesser error than regular Adaboost for the datasets `breastcancer` and `ocr` respectively. For `ionosphere` and `digits`, regular Adaboost achieves lower error. This is conducive to our argument that the best γ is data dependent.

6 Conclusion

For this study we tried to find out if sub-optimal base classifiers at each time step of Adaboost can provide better accuracy. We tried using a linear classifier as our base classifier but discovered that its convergence rate with Adaboost is extremely slow and unsuitable for practical use. Our second approach was to use boosting stumps which split the data with their 2nd or 3rd best feature. With the out experiments we were able to show that for some datasets, using a suboptimal boosting stump at each time step of boosting produces a lower error than what is obtained with the optimal boosting stump.
Although we see in some cases that sub optimal boosting stumps give better accuracy, we have not investigated efficient approaches to find suboptimal features and their thresholds. Another avenue for achieving better results might be to use data dependent computations to estimate the optimal γ at each time step or to estimate the slope of $f(\gamma)$ which can be used to find out whether the algorithm should spend its time finding a lesser γ.
The source code use for this study can be found at

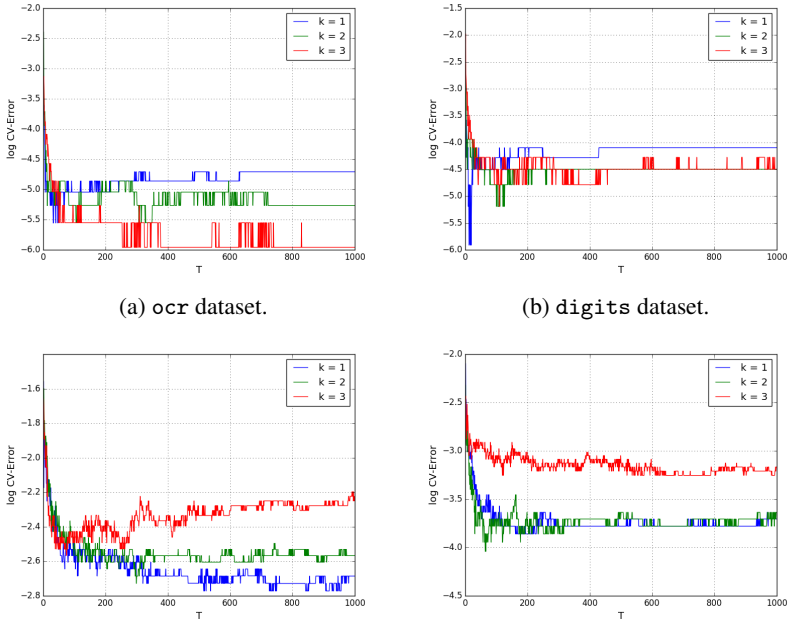
`https://github.com/vighneshbirodkar/lazyboost`



(a) `ocr` dataset. (b) `digits` dataset.

(c) `ionosphere` dataset. (d) `breastcancer` dataset.

Figure 3: Comparison of observed γ for the datasets for each k while varying time steps of Adaboost T.To visually enhance the difference, we plot $\log(\gamma)$ on the Y-axis.



(a) `ocr` dataset. (b) `digits` dataset.

(c) `ionosphere` dataset. (d) `breastcancer` dataset.

Figure 4: Variation of cross validation error with number of time steps T. We have plotted the lof of the cross validation error on the y-axis for better visualization. for each value of k.

References

Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm, 1996.

Y. Freund and R. E. Schapire. A short introduction to boosting, 1999.

D. Johnson and F. Preparata. The densest hemisphere problem. *Theoretical Computer Science*, 6(1):93 – 107, 1978. ISSN 0304-3975. doi: http://dx.doi.org/10.1016/0304-3975(78)90006-3. URL `http://www.sciencedirect.com/science/article/pii/0304397578900063`.

S. Mannor and R. Meir. Weak learners and improved rates of convergence in boosting. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 280–286. MIT Press, 2001. URL `http://papers.nips.cc/paper/1906-weak-learners-and-improved-rates-of-convergence-in-boosting.pdf`.

M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012. ISBN 026201825X, 9780262018258. Chapter 6, Boosting.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

R. E. Schapire. *The Boosting Approach to Machine Learning: An Overview*, pages 149–171. Springer New York, New York, NY, 2003. ISBN 978-0-387-21579-2. doi: 10.1007/978-0-387-21579-2_9. URL `http://dx.doi.org/10.1007/978-0-387-21579-2_9`.