Discussion of Boosting Papers

Jerome Friedman Trevor Hastie Saharon Rosset
Robert Tibshirani and Ji Zhu*

July 24, 2003

1 Introduction

We congratulate the authors for their interesting papers on boosting and related topics. Jiang deals with the asymptotic consistency of Adaboost. Lugosi and Vayatis study the convex optimization of loss functions associated with boosting. Zhang studies the loss functions themselves. Their results imply that boosting-like methods can reasonably be expected to converge to Bayes classifiers under sufficient regularity conditions (such as the requirement that trees with at least $p+1$ terminal nodes are used, where p is the number of variables in the model). An interesting feature of their results is that whenever data-based optimization is performed, some form of regularization is needed in order to attain consistency. In the case of AdaBoost this is achieved by stopping the boosting procedure early, whereas in the case of convex loss optimization, it is achieved by constraining the L_1 norm of the coefficient vector. These results re-iterate, from this new perspective, the critical importance of regularization for building useful prediction models in high-dimensional space. This is also the theme of the remainder of our discussion.

Since the publication of the AdaBoost procedure by Freund and Schapire in 1996, there has been a flurry of papers seeking to answer the question: why does boosting work? Since AdaBoost has been generalized in different ways by different authors, the question might be better posed as; what are the aspects of boosting that are the key to its good performance?

2 Our view: boosting performs a high-dimensional lasso

We would like to present our current view of boosting here. In recent years, a new paradigm has emerged in flexible function fitting. There are three ingredients:

- a large dictionary $\mathcal{D}$ of basis functions for representing the function, typically as a linear expansion $f(x) = \sum_{h_\ell \in \mathcal{D}} h_\ell(x)\beta_\ell$.

*All the authors are in the Statistics Department at Stanford University

- a loss function $L(Y, f(X))$ appropriate for the problem, e.g. for regression or classification.
- a regularizer $J(\beta)$ to control the size of the coefficients in the model.

One then fits the model by minimizing the sum over the data

$$\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(\beta), \tag{1}$$

where λ is a tuning parameter that controls the trade-off between average loss and penalty. If constructed appropriately, the resulting problem is convex and hence can be solved by convex optimization methods. Support vector machines fall into this paradigm: they use an L_2 penalty, a piecewise-linear ("hinge") loss function, and a basis dictionary generated by a positive definite kernel. Although such bases can have infinite dimension, the "kernel trick" results in a finite representation and simplifies the optimization (Vapnik 1996).

Boosting methods use adaptively constructed basis functions, and a forward stagewise procedure to build the model. In Friedman, Hastie & Tibshirani (2000) we showed that AdaBoost fits an additive model in its basis functions, using a particular *exponential* loss function. This framework led to alternative and potentially better forms of boosting, by allowing the use of other loss functions and improvements in the forward stagewise procedure (Friedman et al. 2000, Friedman 1999).

In this work we noticed that slowing down the procedure through shrinkage — a kind of *slow learning* — always seemed to help. This led us to our current view of boosting. We think of the forward stagewise procedure as a numerical device for approximating a sequence of solutions to (1) when $J(\beta)$ is an L_1 penalty. The sequence is obtained by continuously relaxing the parameter λ. Chapter 10 of Hastie, Tibshirani & Friedman (2001) has a discussion of this point. More recently, Efron, Hastie, Johnstone & Tibshirani (2002) proved a result in the simplified framework of least squares regression. Given a centered outcome variable $Y = \{y_i\}_1^n$ and standardized predictors $X_j = \{x_{ij}\}_1^n,\ j = 1, 2, \ldots p$, consider the following forward-stagewise procedure for estimating the coefficients $\beta = \{\beta_j\}_1^p$.

1. Start with $\beta_j = 0$ for all j, and the residual $r = Y$.
2. Find the predictor X_j most correlated with r, and increment its coefficient β_j by some small amount ε in the direction of this correlation:

   $$\beta_j \leftarrow \beta_j + \varepsilon \cdot \text{sign}[\text{corr}(r, X_j)]$$

   Adjust r accordingly:

   $$r \leftarrow r - X_j \cdot \varepsilon \cdot \text{sign}[\text{corr}(r, X_j)]$$

3. Repeat step 2 many times

We call this "incremental forward stagewise regression". If this procedure is run for many steps, it eventually reaches the full least squares solution (modulo the granularity in ϵ). But more interestingly, we show in Efron et al. (2002) that the resulting coefficient profiles approximate the solution to an L_1 constrained regression ("lasso")

$$\beta(\lambda) = \text{argmin} \sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

That is, the profiles of the coefficients resulting from incremental forward stagewise regression looks much like the lasso solutions $\beta(\lambda)$, as λ is varied from $+\infty$ (maximum constraint) to 0 (no constraint). Figure 1 shows an example, taken from Hastie et al. (2001).

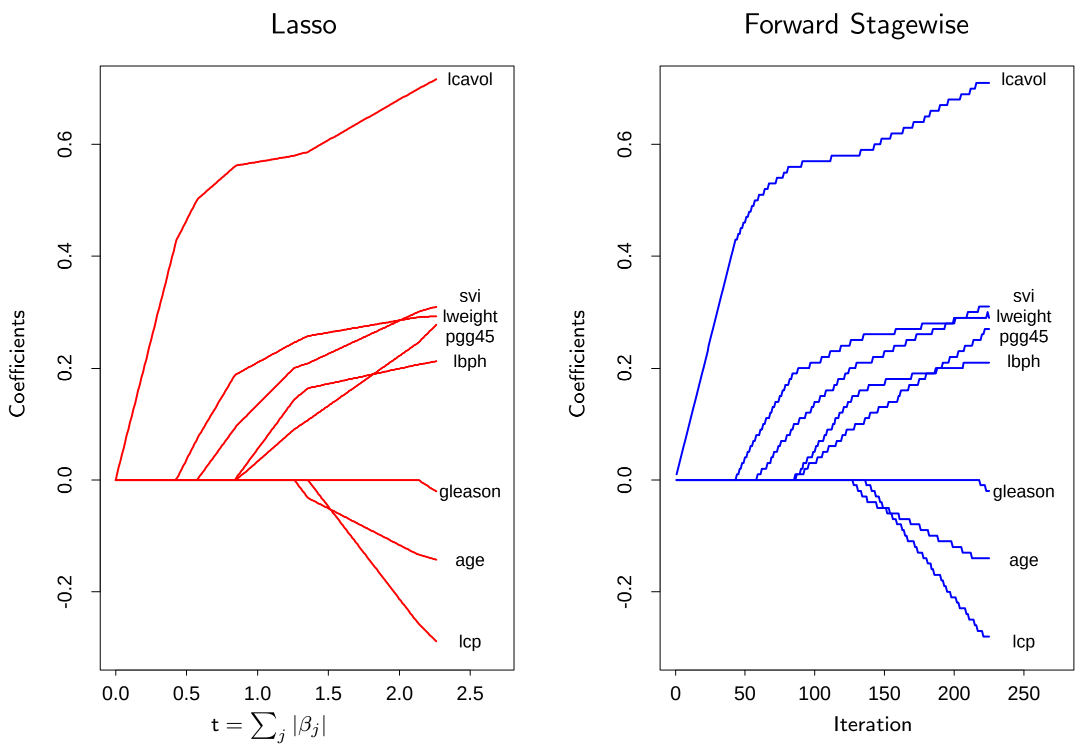

Figure 1: Profiles of estimated coefficients from linear regression, for the prostate data studied in Chapter 3 of Hastie et al. (2001). The left panel shows the results from the lasso, for different values of the bound parameter $t = \sum_j |\beta_j|$. The right panel shows the results of the incremental forward stagewise linear regression algorithm, using $M = 250$ consecutive steps of size $\varepsilon = 0.01$.

What does this have to do with boosting? Take as basis functions the set of all possible regression trees that can be grown from the given features. Suppose we want to compute the lasso path of solutions. This cannot be done directly since the number of trees is so large. Instead, take the incremental forward stagewise regression and replace the predictors X_j with basis functions that are the set of all possible regression trees that can be grown from the given features. The least squares boosting procedure of Friedman (1999) looks like:

1. Start with $F(x) = 0$, and the residual $r = Y$.

2. Fit a tree $f(x)$ to the outcome r, increment $F(x)$ with a shrunken version of $f(x)$

$$F(x) \leftarrow F(x) + \varepsilon f(x),$$

and update r:

$$r \leftarrow r - \varepsilon f(x)$$

3. Repeat step 2 many times

Now in step 2 when we fit a tree to r, we are approximately finding the tree (among all possible trees) that is most correlated with r. Hence least squares boosting can be viewed as a numerically savvy way of carrying out incremental forward stagewise regression on the space of regression trees. The latter, in turn, is an approximate way of computing the lasso path in this space.

For simplicity, our discussion has focussed on least-squares boosting. It also applies to other forms of boosting that use different loss functions(Friedman 2001), for example AdaBoost, which is based on exponential loss (Rosset, Zhu & Hastie 2002).

3 The "bet on sparsity" principle

Now for any of this to be of practical importance, there must be an inherent reason (other than the ease of implementation) to prefer an L_1 penalty to say an L_2 penalty for these kinds of problems. Suppose we have 10K data points and our model is a linear combination of a million trees. Suppose also that the true population coefficients of these trees arose from a Gaussian distribution. Then we know that in a Bayesian sense the best predictor would be a ridge regression; that is, we should use an L_2 rather than an L_1 penalty when fitting the coefficients. On the other hand, if there are only a small number (e.g. 1000) of non-zero true coefficients, the lasso (L_1 penalty) will work better. We think of this as a *sparse* scenario, while the first case (Gaussian coefficients) as *dense*. Note however that in the dense scenario, although the L_2 penalty is best, neither method does very well since there is too little data from which to estimate such a large number of non-zero coefficients. This is the *curse of dimensionality* taking its toll. In a sparse setting, we can potentially do well with the L_1 penalty, since the number of non-zero coefficients is small. The L_2 penalty fails again.

In other words, use of the L_1 penalty follows what we call the **bet on sparsity** principle for high-dimensional problems:

Use a procedure that does well in sparse problems, since no procedure does well in dense problems.

These comments need some moderation:

- For any given application, the degree of sparseness/denseness depends on the unknown true target function, and the chosen dictionary $\mathcal{D}$.

- The notion of sparse vs dense is relative to the size of the training data set and/or the signal-to-noise ratio (SNR). Larger training sets allow us to estimate coefficients

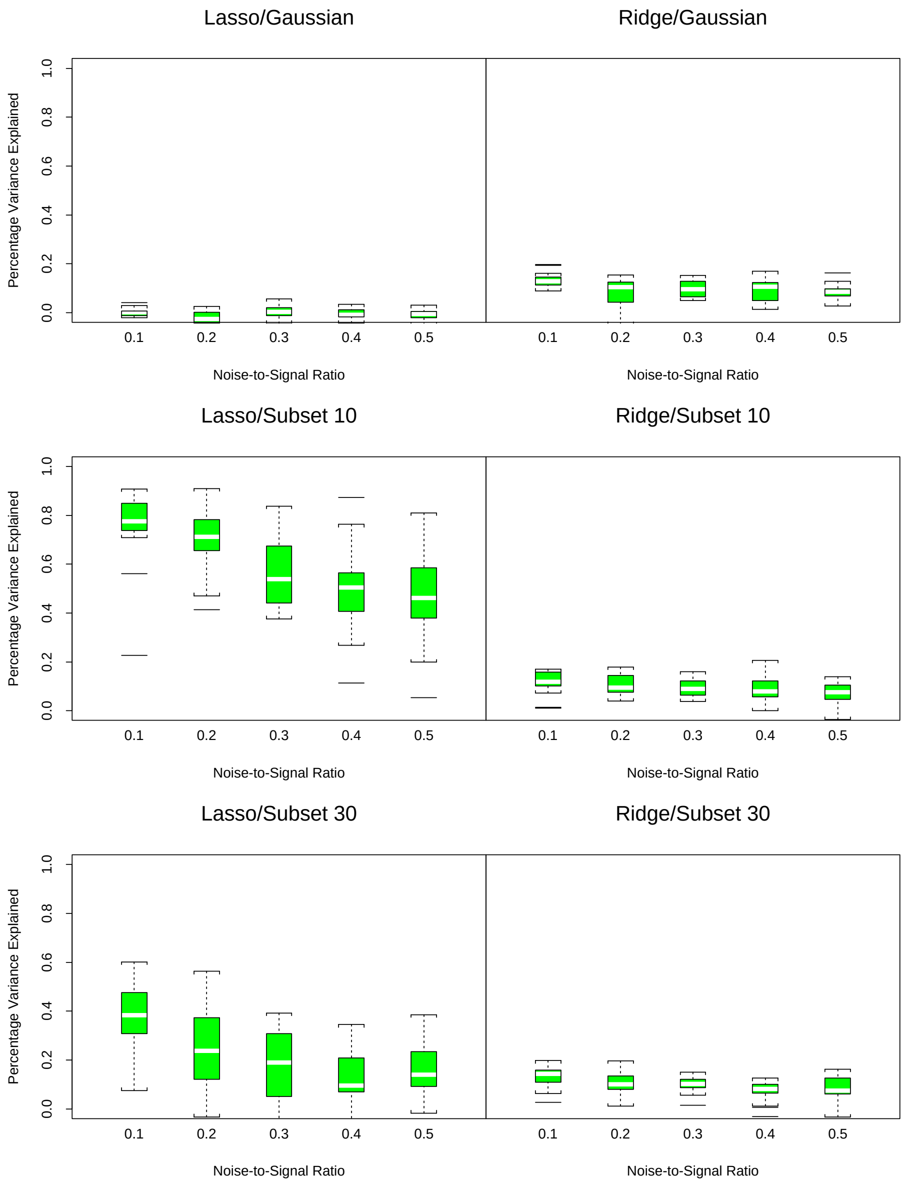

Figure 2: Simulations that show the superiority of the L_1 (lasso) penalty over L_2 (ridge) in regression. Each run has 50 observations with 300 independent Gaussian predictors. In the top row all 300 coefficients are nonzero, generated from a Gaussian distribution. In the middle row, only 10 are nonzero generated from a Gaussian, and the last row has 30 non zero. In each case the coefficients are scaled so that the signal variance $\text{var}(X^T\beta)$ is 1. The noise variance varies from 0.1 to 0.5 (noise to signal ratio). Lasso is used in the left column, ridge in the right. In both cases we used a series of 100 values of λ, and picked the value that minimized the theoretical test error. In the figures we report the percentage variance explained (in terms of mean squared error), displayed as boxplots over 20 realizations for each combination.

with smaller standard errors. Likewise in situations with large SNR, we can identify more nonzero coefficients with a given sample size than in situations where the SNR is smaller.

- The size of the dictionary plays a role as well. Increasing the size of the dictionary may lead to a sparser representation for our function, but the search problem becomes more difficult.

Figure 2 illustrates these points in the context of linear regression. The details are given in the caption. Note that we are not using the training data to select λ, but rather are reporting the best possible behavior for each method in the different scenarios. The L_2 penalty performs poorly everywhere. The Lasso performs reasonably well in the only two situations where it can (sparse coefficients). As expected the performance gets worse as the SNR decreases, and as the model becomes denser.

These empirical results are supported by a large body of theoretical results (Donoho, Johnstone, Hoch & Stern 1992, Donoho & Johnstone 1994, Donoho, Johnstone, Kerkyachairan & Picard 1995, Donoho & Elad 2002) that support the superiority of L_1 estimation in sparse settings.

References

Donoho, D. & Elad, M. (2002), Optimally sparse representation in general(non-orthogonal) dictionaries vy l_1 minimization, Technical report, Statistics Department, Stanford University.

Donoho, D. & Johnstone, I. (1994), 'Ideal spatial adaptation by wavelet shrinkage', *Biometrika* **81**, 425–55.

Donoho, D., Johnstone, I., Hoch, J. & Stern, A. (1992), 'Maximum entropy and the nearly black object', *J. Royal. Statist. Soc. B.* **54**, 41–81.

Donoho, D., Johnstone, I., Kerkyachairan, G. & Picard, D. (1995), 'Wavelet shrinkage; asymptopia? (with discussion)', *J. Royal. Statist. Soc.* **57**, 301–337.

Efron, B., Hastie, T., Johnstone, I. & Tibshirani, R. (2002), Least angle regression, Technical report, Stanford University.

Freund, Y. & Schapire, R. E. (1996), Experiments with a new boosting algorithm, *in* 'Machine Learning: Proceedings of the Thirteenth International Conference', Morgan Kauffman, San Francisco, pp. 148–156.

Friedman, J. (1999), Greedy function approximation: the gradient boosting machine, Technical report, Stanford University.

Friedman, J. (2001), 'Greedy function approximation: a gradient boosting machine', *Annals of Statistics* **29**, 1180.

Friedman, J., Hastie, T. & Tibshirani, R. (2000), 'Additive logistic regression: a statistical view of boosting (with discussion)', *Annals of Statistics* **28**, 337–307.

Hastie, T., Tibshirani, R. & Friedman, J. (2001), *The Elements of Statistical Learning; Data mining, Inference and Prediction*, Springer Verlag, New York.

Rosset, S., Zhu, J. & Hastie, T. (2002), Boosting as a regularized path to a maximum margin classifier, Technical report, Statistics Department, Stanford University.

Vapnik, V. (1996), *The Nature of Statistical Learning Theory*, Springer-Verlag, New York.