Machine Learning Benchmarks and Random Forest Regression

Mark R. Segal (`mark@biostat.ucsf.edu`)

Division of Biostatistics, University of California, San Francisco, CA 94143-0560

April 14, 2003

Abstract.

Breiman (2001a,b) has recently developed an ensemble classification and regression approach that displayed outstanding performance with regard prediction error on a suite of benchmark datasets. As the base constituents of the ensemble are tree-structured predictors, and since each of these is constructed using an injection of randomness, the method is called 'random forests'. That the exceptional performance is attained with seemingly only a single tuning parameter, to which sensitivity is minimal, makes the methodology all the more remarkable. The individual trees comprising the forest are all grown to maximal depth. While this helps with regard bias, there is the familiar tradeoff with variance. However, these variability concerns were potentially obscured because of an interesting feature of those benchmarking datasets extracted from the UCI machine learning repository for testing: all these datasets are hard to overfit using tree-structured methods. This raises issues about the scope of the repository.

With this as motivation, and coupled with experience from boosting methods, we revisit the formulation of random forests and investigate prediction performance on real-world and simulated datasets for which maximally sized trees do overfit. These explorations reveal that gains can be realized by additional tuning to regulate tree size via limiting the number of splits and/or the size of nodes for which splitting is allowed. Nonetheless, even in these settings, good performance for random forests can be attained by using larger (than default) primary tuning parameter values.

Keywords: Prediction error, Regression, UCI Repository

1. Introduction

The purpose of Figure 1 is to display two very distinct prediction error (*PE*) profiles. Identifying consequences of modeling procedures, in particular random forests, as deriving from this distinction is the objective of this paper. Both profiles are obtained from fitting regression trees, with prediction error being estimated via cross-validation and standardized by outcome variance. In the left panel, minimum prediction error is achieved at the first split, after which there is an appreciable rise in error. This increase is such that the prediction error at the maximal number of splits is “significantly” greater than the minimum prediction error; the vertical segments represent $\pm$ 1 standard error. Such profiles where, as a function of increasing model size/complexity (here number of splits), prediction error initially decreases, plateaus, and then increases are common. Indeed, prototypic depictions have this form; see Breiman et al., (1984, p. 87) and Hastie et al., (2001, p. 38). The presence of noise covariates is one factor that can contribute to such profiles.

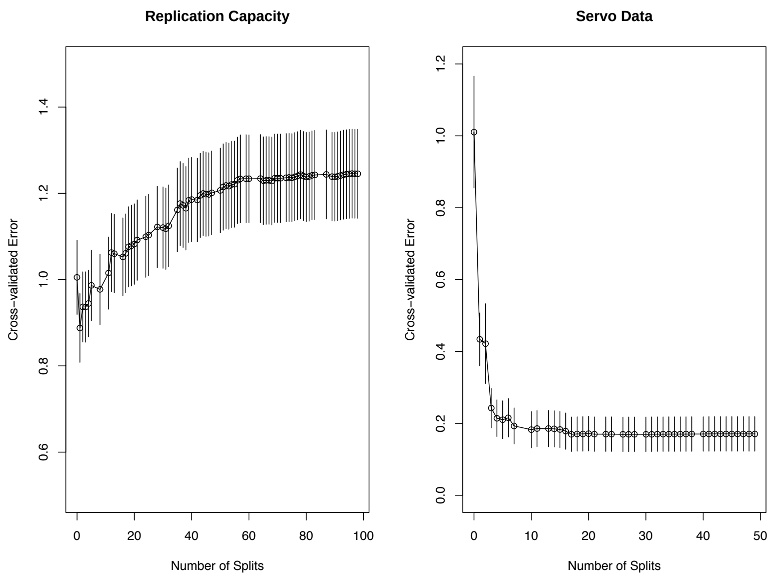

Figure 1. Contrasting prediction error profiles.

The right panel differs in that the prediction error at the maximal number of splits is (essentially) minimal. That is to say, no matter how large a tree-structured predictor we fit, we don’t overfit the data. This behavior is arguably unusual. The (servo) data, whose *PE* is profiled in the right panel, were obtained from the UCI Repository of Machine Learning Databases; see `http://www.ics.uci.edu/ mlearn/MLRepository.html`,
as extracted and converted to R (Ihaka and Gentleman, 1996), and available from `mlbench`, the Machine Learning Benchmark Problems package; see `http://cran.r-project.org/src/contrib/PACKAGES.html#mlbench`.
What is remarkable, and seemingly not appreciated, is that almost every dataset in the `mlbench` package exhibits this same behavior. Thus, benchmarking using this package, and indeed the UCI repository, may not be as generalizable as would be desirable.

In a series of recent papers, Breiman has demonstrated that consequential gains in classification or prediction accuracy can be achieved by using ensembles of trees, where each tree in the ensemble is grown in accordance with the realization of a random vector. Final predictions are obtained by aggregating (voting) over the ensemble, typically using equal weights. Bagging (Breiman, 1996) represents an early example whereby each tree is constructed from a bootstrap (Efron and Tibshirani, 1993) sample drawn with replacement from the training data. The simple mechanism whereby bagging reduces prediction error for unstable predictors, such as trees, is well understood in terms of variance reduction resulting from averaging (Breiman, 1998; Hastie et al., 2001). Such variance gains can be enhanced by reducing the correlation between the quantities being averaged. It is this principle that motivates random forests.

Random forests seek to effect such correlation reduction by a further injection of randomness. Instead of determining the optimal split of a given node of a (constituent) tree by evaluating all allowable splits on all covariates, as is done with single tree methods or bagging, a subset of the covariates drawn at random, is employed. Breiman (2001a,b) argues that random forests (a) enjoy exceptional prediction accuracy, and (b) that this accuracy is attained for a wide range of settings of the single tuning parameter employed. In the next section we further detail the formulation of random forests, and reveal a potential role for a second tuning parameter. In section 3 we overview the contents of the `mlbench` package in light of Figure 1 and the above claims. Section 4 contains some additional analyses of prediction error for some expanded versions of UCI repository and other datasets. Section 5 provides some concluding discussion.

Throughout our focus is on regression, as opposed to classification, problems. The reasons for this are (i) Random forest classification has been shown to perform poorly in instances of class imbalance (Dudoit and Fridlyand, 2003). This has led to the introduction of additional class weighting parameters (Version 4) that, while overcoming deficits, complicate evaluation; and (ii) Overfitting in classification problems is profoundly influenced by the loss function employed. Indeed, it is difficult to overfit with ”0-1” loss (Friedman et al., 2000). However, it is important to note that the behavior exhibited in the right panel of Figure 1 extends to all classification datasets in `mlbench` and all additional datasets examined from the UCI repository when binomial log-likelihood loss is employed. This is despite the possibility for substantial overfitting with this loss criterion (Friedman et al., 2000).

2. Random Forests

A random forest is a collection of tree predictors $h(\mathbf{x};\theta_k),\ k = 1,\dots,K$ where $\mathbf{x}$ represents the observed input (covariate) vector of length p with associated random vector $\mathbf{X}$ and the θ_k are independent and identically distributed (*iid*) random vectors. As mentioned, we focus on the regression setting for which we have a numerical outcome, Y, but make some points of contact with classification (categorical outcome) problems. The observed (training) data is assumed to be independently drawn from the joint distribution of $(\mathbf{X},Y)$ and comprises n $(p+1)$-tuples $(\mathbf{x}_1,y_1),\dots,(\mathbf{x}_n,y_n)$.

For regression, the random forest prediction is the unweighted average over the collection: $\bar{h}(\mathbf{x}) = (1/K)\sum_{k=1}^{K} h(\mathbf{x};\theta_k)$.

As $k \to \infty$ the Law of Large Numbers ensures

$$E_{\mathbf{X},Y}(Y - \bar{h}(\mathbf{X}))^2 \to E_{\mathbf{X},Y}(Y - E_{\theta}h(\mathbf{X};\theta))^2. \tag{1}$$

The quantity on the right is the prediction (or generalization) error for the random forest, designated PE_f^*. The convergence in (1) implies that random forests do not overfit.

Now define the average prediction error for an individual tree $h(\mathbf{X};\theta)$ as

$$PE_t^* = E_{\theta}E_{\mathbf{X},Y}(Y - h(\mathbf{X};\theta))^2. \tag{2}$$

Assume that for all θ the tree is unbiased, i.e., $EY = E_{\mathbf{X}}h(\mathbf{X};\theta)$. Then

$$PE_f^* \leq \bar{\rho}PE_t^* \tag{3}$$

where $\bar{\rho}$ is the weighted correlation between residuals $Y - h(\mathbf{X};\theta)$ and $Y - h(\mathbf{X};\theta^{'})$ for independent $\theta, \theta^{'}$.

The inequality (3) pinpoints what is required for accurate random forest regression: (i) low correlation between residuals of differing tree members of the forest, and (ii) low prediction error for the individual trees. Further, the random forest will, in expectation, decrease the individual tree error, PE_t^*, by the factor $\bar{\rho}$. Accordingly, the randomization injected strives for low correlation.

The strategy employed to achieve these ends is as follows:

1. To keep individual error low, grow trees to maximum depth.

2. To keep residual correlation low randomize via

   a) Grow each tree on a bootstrap sample from the training data.

   b) Specify $m \ll p$ (the number of covariates). At each node of every tree select m covariates and pick the best split of that node based on these covariates.

However, the central point of this paper is that the strategy in 1 controls bias but not variance: such maximal trees may be highly unstable and this instability will be reflected in inflated prediction errors. That this was not observed in empirical evaluations of random forests using the UCI repository is potentially attributable to the abovementioned properties of the repository constituents.

As operationalized by the random forest software, available from `http://www.stat.Berkeley.EDU/users/breiman/rf.html`, the size of the individual trees constituting the forest is controlled by a tuning parameter, `nthsize`. This specifies the number of cases in a node below which the tree will not split, and so determines maximal tree size. For regression forests the default value is `nthsize` = 5, and this is claimed to give generally good results. For classification forests, the default is `nthsize` = 1, asserted to always give good results. However, the user manual asserts that, in large datasets, larger values can be employed for memory and speed considerations with little loss of accuracy. We investigate impact of varying `nthsize` as well as the primary tuning parameter, *m*, in Section 4. Further, we introduce and evaluate a further tuning parameter, anticipated to be helpful in situations where deep trees overfit. This parameter, `nsplit`, governs how many splits per tree are allowable. Note that we could try to achieve such control by adaptively setting `nthsize`, however, this is clearly more awkward. Further, identically sized trees obtained from the two approaches can differ in terms of split covariates and/or cut-points.

Further motivation for constraining the number of allowable splits comes from boosting (Freund and Schapire, 1997). There are some claims that boosting represents an instance of a random forest (Breiman, 2001a) and others that the ”resemblance of boosting to such ensemble approaches is at best superficial and that boosting is fundamentally different” (Hastie et al., 2001). If we adopt the perspective of the former with some results of the latter, then there is a basis for investigating limiting the splits per tree. The results in question are given in Hastie et al., (2001, Section 10.11) and show dramatic gains from curtailing allowable tree size.

In addition to excellent prediction performance, random forests possess a number of features. These include measures of covariate importance, distinguishing forests from so-called black-box predictors (e.g., neural nets), and accurate, internal estimates of test set prediction error. The latter, so-called out-of-bag (oob) estimates, are used in our evaluations (Section 4) and are based on the fact that, for every tree, approximately $1/e \approx 1/3$ cases are not in the bootstrap sample, i.e., are oob. Test set prediction error estimates are simply obtained:

1. Run each oob case down the corresponding tree and get the associated response prediction.

2. For each case, i, average these response estimates over trees for which i was oob, giving $\hat{y}_i$.

3. For regression forests under squared error loss $(1/n)\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ is a *test set* estimate of PE_f^*. For classification problems, the same approach is used substituting an appropriate loss function.

3. UCI Repository

We have asserted that nearly all the datasets in the R package `mlbench` extracted from the UCI Repository exhibit the prediction error : model complexity behavior illustrated in the right panel of Figure 1. Indeed, this remains so for all additional datasets from the repository but not extracted that we examined. This set contains those datasets examined in Breiman (2001b). Some representative plots for regression are presented in Figure 2, while a similar series for classification are displayed in Figures 3 to 5.

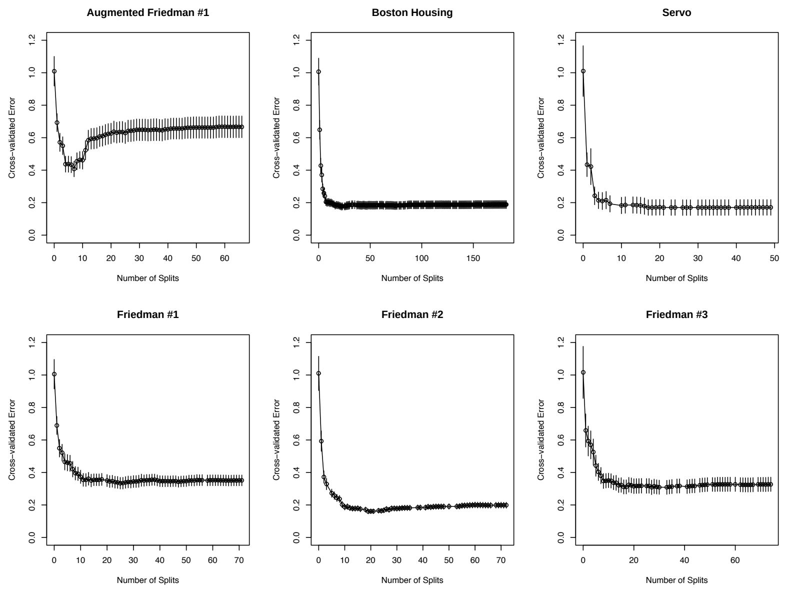

Figure 2. UCI Repository: Regression tree prediction error profiles. Note that the upper left plot corresponds to modification of a synthetic repository dataset in order to achieve a non-monotone error profile; see Section 4.

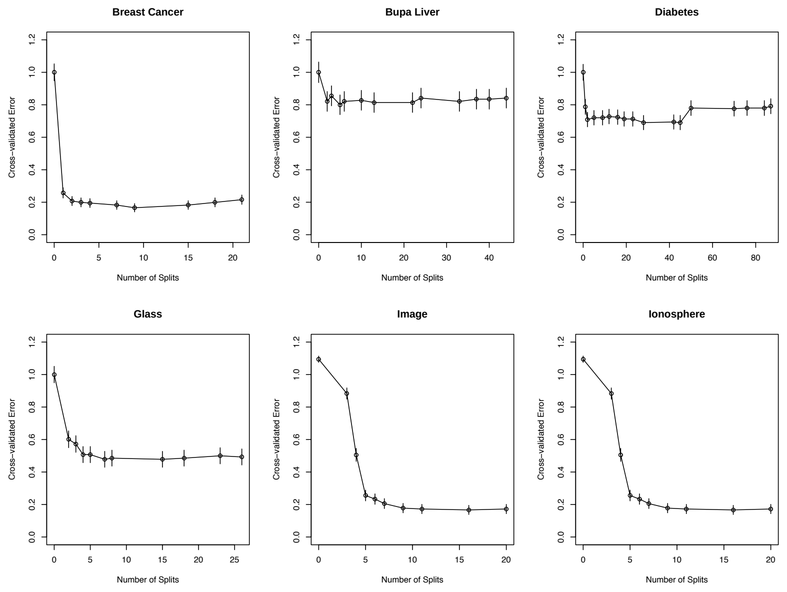

Figure 3. UCI Repository: Classification tree prediction error profiles, I.

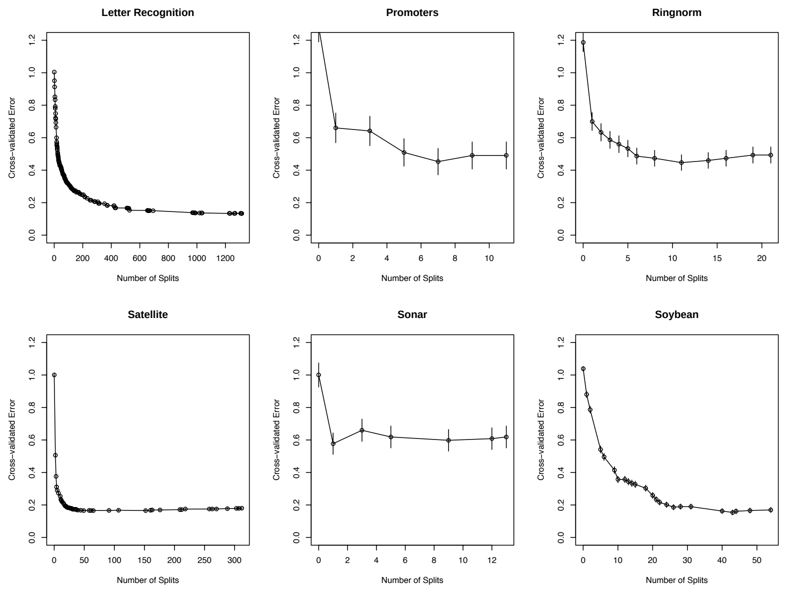

Figure 4. UCI Repository: Classification tree prediction error profiles, II.

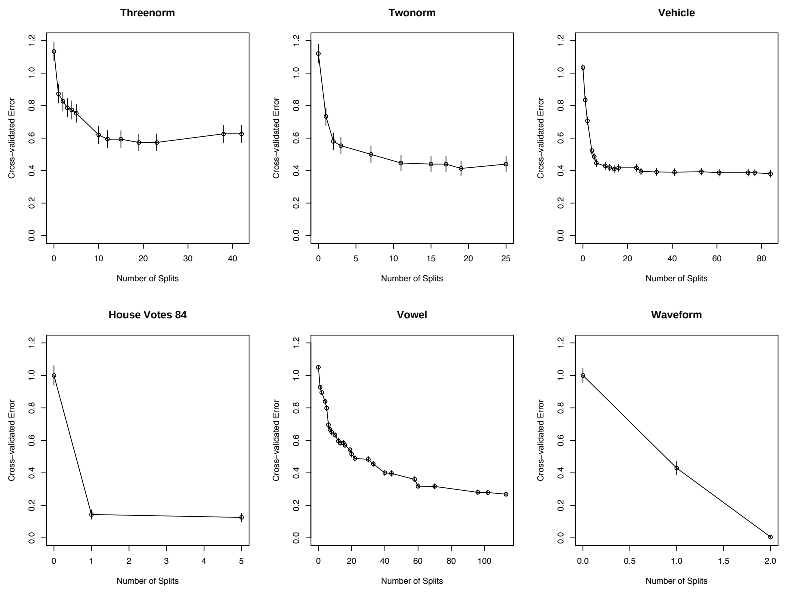

Figure 5. UCI Repository: Classification tree prediction error profiles, III.

Of course the error profiles depend on the class of model being fitted. While it is appropriate to utilize tree-structured models in dissecting the random

forest mechanism, it is also purposeful to assess whether the datasets can't be overfit under other model classes. To that end we investigate error profiles corresponding to least angle regression (lars). LARS represents a recently devised (Efron et al., 2003) technique that includes as special cases L_1 penalized regression, called the lasso, and forward stagewise regression (see Hastie et al., 2001), and further admits a highly computationally efficient implementation; see
`http://www-stat.stanford.edu/ hastie/Papers/LARS/`.

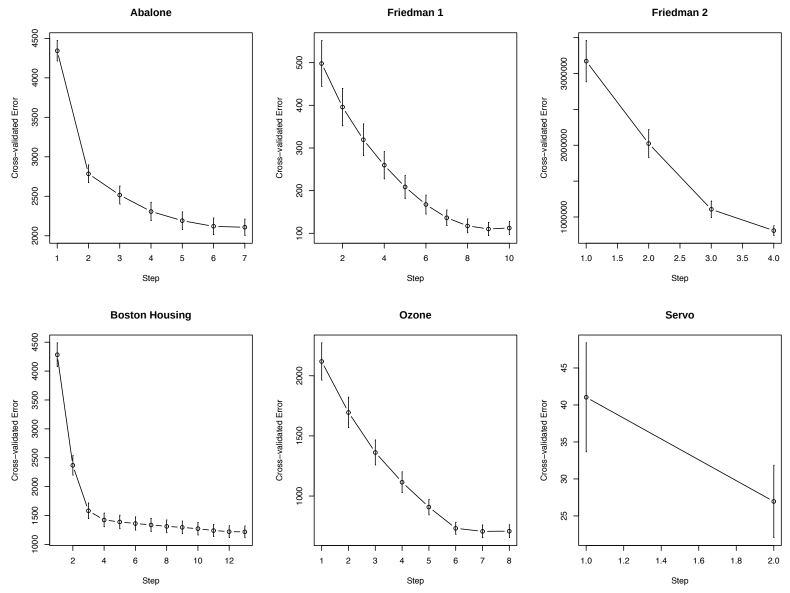

Figure 6. UCI Repository: LARS prediction error profiles.

Once again, the prediction error profiles reveal that all `mlbench` regression datasets cannot be overfit using LARS. Representative error profiles are given in Figure 6. Here "Step" (the x-axis) corresponds to stepwise addition of new covariates to the model and so model complexity increases with increasing Step.

4. Prediction Error Results

In order to evaluate whether a distorted view of the performance of random forest regression arose from the use of the `mlbench` datasets, in view of their characteristics as oultined above, we conducted a limited investigation

Table I. Prediction Errors: Friedman 1, $n = 200, p = 510$

# Splits per Tree	Minimun	# Covariates per Split (m)			
	Node Size	25	170	350	510
Unrestricted	1	20.31	13.79	11.68	10.95
	5	19.90	**11.62**	9.85	9.47
	25	19.80	11.87	10.17	9.50
10	1	21.48	12.38	10.27	9.96
	5	20.38	12.42	10.32	9.83
	25	19.77	11.87	10.17	9.50

in two settings where individual tree models do overfit. The first of these modifies one of the `mlbench` simulated datasets while the second uses the data displayed in the right panel of Figure 1. We explore the impact of tuning parameter specification for both the original primary tuning parameter (m) as well as the parameters controlling node size and number of splits.

The first synthetic dataset is designated “Friedman 1” and has been previously employed in evaluations of MARS (Friedman, 1991) and bagging (Breiman, 1996). Originally, there were 10 independent covariates, uniformly distributed on (0,1) and only five of these were related to the outcome via

$$y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \varepsilon$$

where $\varepsilon \sim N(0, \sigma^2)$. The random forest evaluations use this formulation with $n = 200$ cases and $\sigma = 1$. One modification that yields the potential for overfitting by individual tree-based predictors is to increase the number of noise variables. The effect can be appreciated by examining the first column of Figure 2: the upper panel displays cross-validated error profiles for the augmented version with 500 noise variables while the lower panel is for the original data with 5 noise variables.

Results from applying random forests with varying combinations of tuning parameters are presented in Table I. Data generation made recourse to `mlbench.friedman1` with n, σ as above. The most striking trend in *PE* is with respect to m, the number of covariates that are candidates for split variables at each node. *PE* is decreasing with increasing m, with the minimum being attained at $m = p = 510$. That is, all covariates are candidates at all nodes which coincides with bagging. The *PE* value obtained under default parameter settings is 11.62 (bold in Table I); we note that this exceeds the optimal *PE* for a single *pruned* (to 7 splits) regression tree of 10.11. There

Table II. Prediction Errors: Replication Capacity, $n = 336, p = 282$

# Splits per Tree	Minimun	# Covariates per Split (m)			
	Node Size	10	20	100	282
Unrestricted	5	589.7	590.4	**608.2**	602.9
	25	589.2	586.7	587.5	593.8
	50	594.0	583.7	582.1	584.2
5	5	602.9	592.9	575.6	578.6
	25	598.5	587.4	576.2	577.1
	50	592.4	588.4	581.2	581.6

are no clear trends with regard to total number of splits allowed or minimum node size.

The second example derives from a study of the replication capacity of HIV. Replication Capacity (RC) can loosely be considered as a measure of viral fitness. There is interest in predicting RC based on sequence level data. Here we have amino acid sequence information for positions 4 to 99 of the HIV protease gene and positions 38 to 223 of the HIV reverse transcriptase gene for a total of $p = 282$ positions (covariates). 336 viral samples were available for analysis (Segal et al., 2003). Random forest *PE* results are given in Table II.

Here trends are less apparent. For both the recommended default value of $m(\approx p/3)$ and for bagging ($m = p$), substantial improvement in *PE* can be achieved by restricting the number of splits (`nsplit = 5`). For those forests grown without such restriction, gains are realized by increasing the minimum node size (`nthsize`) for which splitting is allowed. Controlling `nthsize` proves unnecessary, or even counterproductive, for the split restricted forests. This example illustrates the previously mentioned interplay between `nsplit` and `nthsize` and arguably shows that the former provides a more expedient means for exploring a range of models.

It is of interest to note that the best *PE* achieved by the suite of random forests examined coincides with the *PE* attained from a single pruned tree. Such an optimally pruned tree features just a single split. In the “modified Friedman 1” example above, bagging and forests proved effective in reducing *PE* (compared to a single pruned tree) despite the large number of noise inputs. However, such gains are not evidenced for the RC - amino acid sequence data. One contributing factor relates to the nature of sequence data. For a variety of biological reasons we anticipate strong between position dependencies. Indeed, applying the likelihood ratio / permutation testing approach developed by Bickel et al., (1996) for assessing correlation when dealing with categorical (e.g. amino acid levels) covariates reveals that, for reverse transcriptase positions, approximately 40% of all possible pairwise position correlations are *simultaneously* significant ($p < 0.01$). It is this strong between position correlation that thwarts the effectiveness of the random forest variance reduction strategy: $\bar{\rho}$ in (3) will be large.

5. Discussion

This paper was motivated by the observation that all regression (and indeed classification) datasets in the `mlbench` package were not overfit by maximally grown trees. This had potential implications for random forests, since (i) evaluation of random forests made recourse to these datasets, and (ii) individual trees comprising the forest are grown to maximal depth. Thus, a more wide-ranging evaluation featuring datasets for which maximal trees did overfit, was indicated.

The results from the attendant investigation admit several interpretations. Firstly, even in such settings, random forests can achieve good prediction performance. This performance is either comparable or superior to the optimally pruned single tree and can also improve on bagging. However, not all fitted random forests attained such good performance. Further, not only was performance modest at the recommended default settings of tuning parameters, but it also seems difficult to provide precise guidelines for chosing these parameters so as to ensure good performance. One recommendation would be to take *m* large in these situations where there are many noise inputs. The introduced parameter, `nsplit`, also appears useful. Finally, if the prediction accuracy gains realized by random forests are not substantial then, from both a computational and interpretational standpoint, the additional work does not seem warranted.

Of course, these suggestions are based on a limited study of a specialized situation wherein there are relatively very few important inputs. Breiman (2001b) demonstrated the effectiveness of random forests for settings with large numbers of weak inputs. But, such a framework is distinct from settings with large numbers of noise inputs. That such latter datasets arise is exemplified by the replication capacity – amino acid sequence study. Many microarray datasets will also be of this flavor.

Returning to the `mlbench` package and the UCI repository it is important to note that the experience with maximal tree and LARS models not overfitting will not necessarily generalize to other models. As LARS fits (L_1) penalized regression models there are built in safeguards against overfitting. Nonetheless, an expansion of the package/repository to include datasets that broaden the range of signal to noise inputs seems desirable.

References

Bickel, P. J., Cosman, P. C., Olshen, R. A., Spector, P. C., Rodrigo, A. G., & Mullins, J. I. (1996). Covariability of V3 loop amino acids. *AIDS Research and Human Retroviruses*, 12, 1401-11.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees*. Belmont, CA: Wadsworth.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24, 123-140.

Breiman, L. (2001a). Statistical modeling: the two cultures. *Statistical Science*, 16, 199-215.

Breiman, L. (2001b). Random forests. *Machine Learning*, 45, 5-32.

Dudoit, S., & Fridlyand, J. (2003). Classification in microarray experiments. In T. P. Speed (Ed.), *Statistical Analysis of Gene Expression Microarray Data*. Boca Raton, FL: Chapman & Hall/CRC Press.

Efron, B., & Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. New York, NY: Chapman & Hall.

Efron, B., Hastie, T. J., Johnstone, I., & Tibshirani, R. J. (2003). Least angle regression. *Annals of Statistics*, in press.

Freund, Y., & Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55: 119-139.

Friedman, J. H. (1991). Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, 19, 1-67.

Friedman, J. H., Hastie, T. J., & Tibshirani, R. J. (2000). Additive logistic regression: A statistical view of boosting (with discussion). *Annals of Statistics*, 28, 337-407.

Hastie, T. J., Tibshirani, R. J., & Friedman, J. H. (2001). *The Elements of Statistical Learning*. New York, NY: Springer.

Ihaka, R., & Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5, 299-314.

Segal, M. R., Barbour, J. D., Wrin, T., Deeks, S. G., Hecht, F. M, Hellman, N. S., Petropoulos, C. J., & Grant, R. M. (2003). Wide variation in replication capacity of wild-type HIV-1. Technical Report, Center for Bioinformatics and Molecular Biostatistics, University of California, San Francisco.