Combining estimates in regression and classification

Michael LeBlanc
and
Robert Tibshirani

Department of Preventive Medicine and Biostatistics
and
Department of Statistics
University of Toronto

December 13, 1993

Abstract

We consider the problem of how to combine a collection of general regression fit vectors in order to obtain a better predictive model. The individual fits may be from subset linear regression, ridge regression, or something more complex like a neural network. We develop a general framework for this problem and examine a recent cross-validation-based proposal called "stacking" in this context. Combination methods based on the bootstrap and analytic methods are also derived and compared in a number of examples, including best subsets regression and regression trees. Finally, we apply these ideas to classification problems where the estimated combination weights can yield insight into the structure of the problem.

1 Introduction

Consider a standard regression setup: we have predictor measurements $\mathbf{x}_i = (x_{i1}, \ldots x_{ip})^T$ and a response measurement y_i on N independent training cases. Let $\mathbf{z}$ represent the entire training sample. Our goal is derive a function $c_{\mathbf{Z}}(\mathbf{x})$ that accurately predicts future y values.

Suppose we have available K different regression fitted values from these data, denoted by $c^k_{\mathbf{Z}}(\mathbf{x})$, for $k = 1, 2, \ldots K$. For example, $c^k_{\mathbf{Z}}(\mathbf{x})$ might be a least squares fit for some subset of the variables, or a ridge regression fit, or

something more complicated like the result of a projection-pursuit regression or neural network model. Or the collection of $c^k_{\mathbf{Z}}(\mathbf{x})$s might correspond to a single procedure run with K different values of an adjustable parameter. In this paper we consider the problem of how to best combine these estimates in order to obtain an estimator that is better than any of the individual fits. The class that we consider has the form of a simple linear combination:

$$\sum_{k=1}^{K} \beta_k \, c^k_{\mathbf{Z}}(\mathbf{x}). \tag{1}$$

One way to obtain estimates of $\beta_1, \ldots \beta_K$ is by least squares regression of y on $c^1_{\mathbf{Z}}(\mathbf{x}), \ldots c^K_{\mathbf{Z}}(\mathbf{x})$. But this might produce poor estimates because it doesn't take into account the relative amount of fitting that is present in each of the $c^k_{\mathbf{Z}}(\mathbf{x})$s, or the correlation of the $c^k_{\mathbf{Z}}(\mathbf{x})$s induced by the fact that all of the regression fits are estimated from the data $\mathbf{z}$. For example, if the $c^k_{\mathbf{Z}}(\mathbf{x})$s represent a nested set of linear models, $\hat{\beta}_k$ will equal one for the largest model and zero for the others, and hence will simply reproduce the largest model.

In a paper in the neural network literature, Wolpert (1992) presented an interesting idea known as "stacked generalization" for combining estimators. His proposal was translated into statistical language by Breiman (1993); he applied and studied it the regression setting, calling it "stacked regression".

Here is how stacking works. We let $c^k_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i)$ denote the leave-one out cross-validated fit for $c^k_{\mathbf{Z}}(\mathbf{x})$, evaluated at $\mathbf{x} = \mathbf{x}_i$. The stacking method minimizes

$$\sum_{i=1}^{N} \Big[y_i - \sum_{k=1}^{K} \beta_k \, c^k_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i) \Big]^2 \tag{2}$$

producing estimates $\hat{\beta}_1, \ldots \hat{\beta}_K$. The final predictor function is $v_{\mathbf{Z}}(\mathbf{x}) = \sum \hat{\beta}_k \, c^k_{\mathbf{Z}}(\mathbf{x})$.

Notice how this differs from a more standard use of cross-validation. Usually for each method k one constructs the prediction error estimate

$$\widehat{\mathrm{PE}}(k) = \frac{1}{N} \sum_{i=1}^{N} [y_i - c^k_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i)]^2. \tag{3}$$

Then we choose $c^k_{\mathbf{Z}}(\mathbf{x})$ that minimizes $\widehat{\mathrm{PE}}(k)$. In stacking we are estimating a linear combination of models rather than choosing just one.

In the particular cases that he tried, Breiman found that the linear combination $\sum_{k=1}^{K} \hat{\beta}_k c^k_{\mathbf{Z}}(\mathbf{x})$ did not exhibit good prediction performance. However when the coefficients in (2) were constrained to be non-negative, $v_{\mathbf{Z}}(\mathbf{x})$ showed better prediction error than any of the individual $c^k_{\mathbf{Z}}(\mathbf{x})$. In some cases the improvement was substantial.

This paper grew out of our attempt to understand how and why stacking works. Cross-validation is usually used to estimate the prediction error of an

estimator. At first glance, stacking seems to use cross-validation for a fundamentally different purpose, namely to construct a new estimator from the data. In this paper we derive a framework for the problem of combining estimators and as a result we cast the stacking method into more familiar terms. We also derive combination estimators based on the bootstrap, and analytic methods in some simple cases.

2 A framework for combining regression estimators

As in section 1, we have data $(\mathbf{x}_i, y_i), i = 1, 2, \ldots N$, with $\mathbf{x}_i = (x_{i1} \ldots x_{ip})$, and let $\mathbf{z}$ be the entire sample. We assume that each pair $(\mathbf{x}_i, y_i)$ is an independent realization of random variables $(\mathbf{X}, Y)$ having distribution F.

Let $C_{\mathbf{z}}(\mathbf{x}_0) = (c^1_{\mathbf{z}}(\mathbf{x}_0), \ldots c^K_{\mathbf{z}}(\mathbf{x}_0))^T$ be a K-vector of estimators evaluated at $\mathbf{X} = \mathbf{x}_0$, based on data $\mathbf{z}$.

We seek coefficients $\beta = (\beta_1, \ldots \beta_K)^T$ so that the linear combination estimator $C_{\mathbf{z}}(\mathbf{x})^T\beta = \sum \beta_k c^k_{\mathbf{z}}(\mathbf{x})$ has low prediction error. Our criterion for selecting β is

$$\tilde{\beta} = \text{argmin}_\beta \mathrm{E}_{0F}(Y_0 - C_{\mathbf{z}}(\mathbf{X}_0)^T\beta)^2. \tag{4}$$

Here $\mathbf{z}$ is fixed and E_{0F} denotes expectation under $(\mathbf{X}_0, Y_0) \sim F$.

Of course in real problems we don't know E_{0F} and so we have to derive sample-based estimates. The obvious estimate of $g(\mathbf{z}, F, \beta) = \mathrm{E}_{0F}(Y_0 - C_{\mathbf{z}}(\mathbf{X}_0)^T\beta)^2$ is the plug-in or resubstitution estimate

$$g(\mathbf{z}, \hat{F}, \beta) = \mathrm{E}_{0\hat{F}}(Y_0 - C_{\mathbf{z}}(X_0)^T\beta)^2 = \frac{1}{N}\sum_1^N (y_i - C_{\mathbf{z}}(\mathbf{x}_i)^T\beta)^2, \tag{5}$$

whose minimizer is the least squares estimator

$$\hat{\beta}_{\mathrm{LS}} = [\sum_i (C_{\mathbf{z}}(\mathbf{x}_i)C_{\mathbf{z}}(\mathbf{x}_i)^T]^{-1}[\sum_i C_{\mathbf{z}}(\mathbf{x}_i)y_i]. \tag{6}$$

What's wrong with $\hat{\beta}_{\mathrm{LS}}$? The problem is that the data $\mathbf{z} = (\mathbf{x}_1, y_1,) \ldots (\mathbf{x}_n, y_n)$ is used in two places: in constructing the $C_{\mathbf{z}}$ and in evaluating the error between y_i and $C_{\mathbf{z}}(x_i)$. As a result:

1. $\sum_i C_{\mathbf{z}}(\mathbf{x}_i)C_{\mathbf{z}}(\mathbf{x}_i)^T$ and $\sum_i C_{\mathbf{z}}(\mathbf{x}_i)y_i$ are biased estimates of their population analogues

2. $g(\mathbf{z}, \hat{F}, \beta)$ is a biased estimator of $g(\mathbf{z}, F, \beta)$.

Although description (1) above may seem to be more direct, in section 3 we will find that description (2) is more useful.

The stacking method estimates the prediction error $g(\mathbf{z}, F, \beta)$ by

$$\frac{1}{N}\sum_{1}^{N}(y_i - C_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i)^T\beta)^2. \tag{7}$$

The corresponding minimizer gives the stacking estimator of $\tilde{\beta}$:

$$\hat{\beta}_{\mathrm{St}} = [\sum_i (C_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i)C_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i)^T]^{-1}\sum_i C_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i)y_i. \tag{8}$$

Stacking uses different data to construct C and to evaluate the error between y and C, and hence should produce a less biased estimator of $g(\mathbf{z}, F, \beta)$.

Remark A. Note that there is no intercept in the linear combination in (4). This makes sense when each estimator $c^k_{\mathbf{Z}}(\mathbf{x})$ is an estimator of Y, so that $E[c^k_{\mathbf{Z}}(\mathbf{x})] \approx EY$, and hence there are values of β giving $E[\sum \beta_k c^k_{\mathbf{Z}}(\mathbf{x})] \approx EY$ (e.g. take $\sum \beta_k = 1$). In the more general case, each $c^k_{\mathbf{Z}}(\mathbf{x})$ does not necessarily estimate Y: for example each $c^k_{\mathbf{Z}}(\mathbf{x})$ might be an adaptively chosen basis function in a nonlinear regression model. Then we would want to include an intercept in the linear combination: this causes no difficulty in the above framework, since we can just set $c^1_{\mathbf{Z}}(\mathbf{x}) \equiv 1$.

3 Bootstrap estimates

One can apply the bootstrap to this problem by bias-correcting the quantities $\sum_i C_{\mathbf{Z}}(\mathbf{x}_i)C_{\mathbf{Z}}(\mathbf{x}_i)^T$ and $\sum_i C_{\mathbf{Z}}(\mathbf{x}_i)y_i$ appearing in the least squares estimator (6). However, it is more useful to approach the problem in terms of bias correction of the prediction error function $g(\mathbf{z}, \hat{F}, \beta)$. We then minimize the bias-corrected function $\tilde{g}(\mathbf{z}, \hat{F}, \beta)$, to obtain an improved estimator $\tilde{\beta}$. The estimator that we obtain from this procedure is in fact the least squares estimator that uses bias corrected versions of $\sum_i C_{\mathbf{Z}}(\mathbf{x}_i)C_{\mathbf{Z}}(\mathbf{x}_i)^T$ and $\sum_i C_{\mathbf{Z}}(\mathbf{x}_i)y_i$. The advantage of approaching the problem through the prediction error function is that regularization of the estimator can be incorporated in a straightforward manner (section 5).

The method presented here is somewhat novel in that we bias-correct a *function* of β, rather than a single estimator, as is usually the case. A similar proposal in a different setting can be found in McCullagh and Tibshirani (1988).

The bias of $g(\mathbf{z}, \hat{F}, \beta)$ is

$$\Delta(F, \beta) = \mathrm{E}_F[g(\mathbf{z}, F, \beta) - g(\mathbf{z}, \hat{F}, \beta)]. \tag{9}$$

Given an estimate $\hat{\Delta}(F, \beta)$, our improved estimate is

$$\hat{g}(\mathbf{z}, F, \beta) = g(\mathbf{z}, \hat{F}, \beta) + \hat{\Delta}(F, \beta). \tag{10}$$

Finally, an estimate of β is obtained by minimization of expression (10).

How can we estimate $\Delta(F, \beta)$? Note that the stacking method implicitly uses the quantity

$$\frac{1}{N}\sum_{1}^{n}(y_i - C_{\mathbf{z}^{(-i)}}(\mathbf{x}_i)^T\beta)^2 - \frac{1}{N}\sum_{1}^{n}(y_i - C_{\mathbf{z}}(\mathbf{x}_i)^T\beta)^2$$

to estimate $\Delta(F, \beta)$.

Here we outline a bootstrap method that is similar to the estimation of the optimism of an error rate (Efron, 1982; Efron and Tibshirani, 1994 chapter 17). The bootstrap estimates $\Delta(F, \beta)$ by

$$\Delta(\hat{F}, \beta) = \mathrm{E}_{\hat{F}}[g(\mathbf{z}^*, \hat{F}, \beta) - g(\mathbf{z}^*, \hat{F}^*, \beta)] \tag{11}$$

where $\hat{F}^*$ is the empirical distribution of a sample drawn with replacement from $\mathbf{z}$.

The advantage of the simple linear combination estimator is that the the resulting minimizer of (10) can be written down explicitly (c.f section 8). Let

$$\begin{aligned} g_1(\mathbf{z}, \hat{F}) &= \frac{1}{N}\sum_{1}^{N} C_{\mathbf{z}}(\mathbf{x}_i)C_{\mathbf{z}}(\mathbf{x}_i)^T \\ g_2(\mathbf{z}, \hat{F}) &= \frac{1}{N}\sum_{1}^{N} C_{\mathbf{z}}(\mathbf{x}_i)y_i \\ \Delta_1(\hat{F}) &= \mathrm{E}_{\hat{F}}[g_1(\mathbf{z}^*, \hat{F}) - g_1(\mathbf{z}^*, \hat{F}^*)] \\ \Delta_2(\hat{F}) &= \mathrm{E}_{\hat{F}}[g_2(\mathbf{z}^*, \hat{F}) - g_2(\mathbf{z}^*, \hat{F}^*)] \end{aligned} \tag{12}$$

Then the minimizer of (10) is

$$\hat{\beta}_{\mathrm{Bo}} = [g_1(\mathbf{z}, \hat{F}) + \Delta_1(\hat{F})]^{-1}[g_2(\mathbf{z}, \hat{F}) + \Delta_2(\hat{F})]. \tag{13}$$

In simple terms, we bias-correct both $\sum_1^N C_{\mathbf{z}}(\mathbf{x}_i)C_{\mathbf{z}}(\mathbf{x}_i)^T/N$ and $\sum_1^N C_{\mathbf{z}}(\mathbf{x}_i)y_i/N$, and the resulting estimator is just the least squares estimator that uses the bias-corrected versions.

As with most applications of the bootstrap, we must use bootstrap sampling (Monte Carlo simulation) to approximate these quantities. Full details are given in the algorithm below.

Summary of bootstrap algorithm for combining estimators

1. Draw B bootstrap samples $\mathbf{z}^{*b} = (\mathbf{x}^{*b}, y^{*b})$, $b = 1, 2, \ldots B$, and from each sample derive the estimators $C_{\mathbf{Z}^{*b}}$.

2. Evaluate $C_{\mathbf{Z}^{*b}}$ on both the original sample and on the bootstrap sample, and compute

$$\begin{aligned}\hat{\Delta}_1 &= \frac{1}{B}\Big[\frac{1}{N}\sum_{i=1}^{N} C_{\mathbf{Z}^{*b}}(x_i)C_{\mathbf{Z}^{*b}}(x_i)^T - \frac{1}{N}\sum_{i=1}^{N} C_{\mathbf{Z}^{*b}}(x_i^{*b})C_{\mathbf{Z}^{*b}}(x_i^{*b})^T\Big] \\ \hat{\Delta}_2 &= \frac{1}{B}\Big[\frac{1}{N}\sum_{i=1}^{N} C_{\mathbf{Z}^{*b}}(x_i)y_i - \frac{1}{N}\sum_{i=1}^{N} C_{\mathbf{Z}^{*b}}(x_i^{*b})y_i^{*b}\Big].\end{aligned}$$

   This gives corrected variance and covariances

$$\begin{aligned}M_{CC} &= \frac{1}{N}\sum_{i=1}^{N} C_{\mathbf{Z}}(x_i)C_{\mathbf{Z}}(x_i)^T + \hat{\Delta}_1 \\ M_{Cy} &= \frac{1}{N}\sum_{i=1}^{N} C_{\mathbf{Z}}(x_i)y_i + \hat{\Delta}_2.\end{aligned}$$

3. Use M_{CC} and M_{Cy} to produce a (possibly regularized) regression of y on C. The regression coefficients $\hat{\beta}$ are the (estimated) optimal combination weights.

The regularized regression mentioned in step 3 is discussed is section 5.

Remark B. A key advantage of the simple linear combination estimator is that the minimizers of the bias-corrected prediction error (10) can be written down explicitly. In section 8) we discuss more general tuning parameter selection problems where this will typically not be possible, so that the proposed procedure may not be computationally feasible.

Remark C. Suppose that each $c_{\mathbf{Z}}^k(\mathbf{x}_i)$ is a linear least squares fit for a fixed subset of p_k variables. Then it is easy to show that the kth element of the bootstrap correction $\Delta_2(\hat{F})$ is

$$-p_k\hat{\sigma}^2 \tag{14}$$

where $\hat{\sigma}^2$ is the estimated variance of y_i. Thus the bootstrap correction $\Delta_2(\hat{F})$ adjusts $\sum C_{\mathbf{Z}}(\mathbf{x}_i)y_i/N$ downward to account for the number of regressors used in each $c_{\mathbf{Z}}^k$. The elements of $\Delta_1(\hat{F})$ are more complicated. Let the design matrix for $c_{\mathbf{Z}}^k$ be Z,with a corresponding bootstrap value of Z_k^*. Then the kkth element of $\Delta_1(\hat{F})$ is

$$\hat{\sigma}^2\{E_{\hat{F}}[(Z_k^{*T}Z_k^*)^{-1}]Z_k^TZ_k - p\} \geq 0, \tag{15}$$

the inequality following from Jensen's inequality. Thus $\Delta_1(\hat{F})$ will tend to inflate the diagonal of $\sum C_{\mathbf{Z}}(\mathbf{x}_i)C_{\mathbf{Z}}(\mathbf{x}_i)^T/N$ and hence shrink the least squares estimator $\hat{\beta}_{LS}$. The off-diagonal elements of $\Delta_1(\hat{F})$ are more difficult to analyze: empirically, they seem to be negative when the $c^k_{\mathbf{Z}}$s are positively correlated.

4 Linear estimators and generalized cross-validation

Suppose each of the estimators $c^k_{\mathbf{Z}}(\mathbf{x}_i)$, $i = 1, 2, \ldots N$, can be written as $H_k\mathbf{y}$ for some fixed matrix H_k. For example $H_k\mathbf{y}$ might be the least squares fit for a fixed subset of the variables $X_1, X_2 \ldots X_p$, or it might be a cubic smoothing spline fit.

We can obtain an analytic estimate of the combination weights, by approximating the cross-validation estimate. Let h^k_{ii} be the iith element of H_k. A standard approximation used in generalized cross-validation gives

$$\begin{aligned} c^k_{\mathbf{Z}^{(-i)}}(\mathbf{x}_i) &= \frac{c^k_{\mathbf{Z}}(\mathbf{x}_i) - y_i \cdot h^k_{ii}}{1 - h^k_{ii}} \\ &\approx \frac{c^k_{\mathbf{Z}}(\mathbf{x}_i) - y_i \cdot \mathrm{tr}(H_k)/N}{1 - \mathrm{tr}(H_k)/N} \equiv \tilde{c}^k_{\mathbf{Z}}(\mathbf{x}_i). \end{aligned} \tag{16}$$

Therefore a simple estimate of the combination weights can be obtained by least squares regression of y_i on $\tilde{c}^k_{\mathbf{Z}}(\mathbf{x}_i)$. Denote the resulting estimate by $\hat{\beta}_{\mathrm{GCV}}$.

When $c^k_{\mathbf{Z}}(\mathbf{x}_i)$ is an adaptively chosen linear fit, for example a best subset regression, the above derivation does not apply. However, one might try ignoring the adaptivity in the estimators and use above analytic correction anyway. We explore this idea in example 1 of section 6.

5 Regularization

From the previous discussion we have four different estimators of the combination weights β:

1. $\hat{\beta}_{\mathrm{LS}}$: the least-squares estimator defined in (6).
2. $\hat{\beta}_{\mathrm{St}}$: the stacked regression estimator defined in (8).
3. $\hat{\beta}_{\mathrm{Bo}}$: the bootstrap estimator defined in (13)
4. $\hat{\beta}_{\mathrm{GCV}}$: the generalized cross-validation estimator defined below equation (16), available for linear estimators, $c^k_{\mathbf{Z}}(\mathbf{x}) = H_k\mathbf{y}$.

In our discussion we have derived each of these estimates as minimizers of some roughly unbiased estimate of prediction error $\hat{g}(\mathbf{z}, \hat{F}, \beta)$. However in our simulation study of the next section (and also in Breiman's 1993 study),

we find that none of these estimators work well. All of these are unrestricted least squares estimators, and it turns out that some sort of regularization may needed to improve their performance. This is not surprising, since the same phenomenon occurs in multiple linear regression. In that setting, the average residual sum of squares is unbiased for the true prediction error, but its minimizer— the least squares estimator— does not necessarily possess the minimum prediction error. Often a regularized estimate, for example a ridge estimator or a subset regression, has lower prediction error than the least squares estimator.

In terms of our development of section 2, the prediction error estimate $\hat{g}(\mathbf{z}, F, \beta)$ is approximately unbiassed for $g(\mathbf{z}, F, \beta)$ for *each fixed value* β, but it is no longer unbiassed when a estimator $\hat{\beta}$ is substituted for β. Roughly unbiassed estimators for $g(\mathbf{z}, F, \hat{\beta})$ can be constructed by adding a regularization term to $\hat{g}(\mathbf{z}, F, \beta)$ and this leads to regularized versions of the four estimators listed above.

We investigate a number of forms of shrinkage in this paper. Most regularizations can be defined by the addition of a penalty function $J(\beta)$ to the corrected prediction error function $g(\mathbf{z}, F, \beta)$:

$$\tilde{\beta} = \text{argmin}_{\beta}[\hat{g}(\mathbf{z}, F, \beta) + \lambda \cdot J(\beta)] \tag{17}$$

where $\lambda \geq 0$ is a regularization parameter. Let M_{CC} and $M_{C\mathbf{y}}$ be the bias corrected versions of $\sum_i C_{\mathbf{z}}(\mathbf{x}_i) C_{\mathbf{z}}(\mathbf{x}_i)^T$ and $\sum_i C_{\mathbf{z}}(\mathbf{x}_i) y_i$ respectively, obtained by either the bootstrap, cross-validation or generalized cross-validation as described earlier. We consider two choices for $J(\beta)$. Rather than shrink towards zero (as in ridge regression) it seems somewhat more natural to shrink $\hat{\beta}$ towards $(1/K, 1/K, \ldots 1/K)^T$, since we might put prior weight $1/K$ on each model fit $c^k_{\mathbf{z}}$. To regularize in this way, we choose $J(\beta) = ||\beta - (1/K)\mathbf{1}||^2$, leading to the estimate

$$\tilde{\beta} = (M_{CC} + \lambda \cdot I)^{-1}[M_{C\mathbf{y}} + (\lambda/K)\mathbf{1}]. \tag{18}$$

We could choose λ by another layer of cross-validation or bootstrapping, but a fixed choice is more attractive computationally. After some experimentation we found that a good choice the value of λ such that the Euclidean distance between $\hat{\beta}$ and $(1/K, \ldots 1/K)^T$ is reduced by a fixed factor (75%)

Another regularization forces $\sum \hat{\beta}_i = 1$; this can be achieved by choosing $J(\beta) = \beta^T \mathbf{1} - 1$ leading to

$$\tilde{\beta} = M_{CC}^{-1}[M_{C\mathbf{y}} - \lambda \mathbf{1}] \tag{19}$$

where λ is chosen so that $\tilde{\beta}^T \mathbf{1} = 1$.

To generalize both of these, one might consider estimators of the form $(M_{CC} + \lambda_1 I)^{-1}[M_{C\mathbf{y}} + \lambda_2 \mathbf{1}]$ and use a layer of cross-validation or bootstrapping to estimate the best values of λ_1 and λ_2. This is very computationally intensive, and we did not try it in this study.

Still another form of regularization is a non-negativity constraint, suggested by Breiman (1993). An algorithm for least squares regression under the constraint $\beta_k \geq 0, k = 1, 2 \ldots K$ is given in Lawson and Hanson (1974), and this can be used to constrain the weights in the stacking and GCV procedures. To apply this procedure in general, we minimize $\hat{g}(\mathbf{z}, F, \beta)$ (equation 10) under the constraint $\beta_k \geq 0, k = 1, 2 \ldots K$ leading to

$$\tilde{\beta} = \text{argmin}_{\beta}(\beta^T M_{CC} \beta - 2 M_{Cy} \beta) \tag{20}$$

This problem can be solved by a simple modification of the algorithm given in Lawson and Hansen (1974).

Finally, we consider a simple combination method that is somewhat different in spirit than other proposed methods, but which constrains $\tilde{\beta}_k \geq 0$ and $\tilde{\beta}^T \mathbf{1} = 1$. We let $\tilde{\beta}_k$ be the relative performance of the kth model. For instance, one might use

$$\tilde{\beta}_k = \frac{L(\mathbf{y}, \hat{\theta}_k, c_{\mathbf{Z}}^k)}{\sum_j L(\mathbf{y}, \hat{\theta}_j, c_{\mathbf{Z}}^k)}$$

where $L(\mathbf{y}, \hat{\theta}_k, c_{\mathbf{Z}}^k)$ is the maximized likelihood for model k. For a normal model

$$\tilde{\beta}_k = \frac{\hat{\sigma}_k^{2-n/2}}{\sum_j \hat{\sigma}_j^{2-n/2}},$$

where $\hat{\sigma}_k^2$ is the mean residual error. The estimator can also be motivated as an "estimated posterior mean" where one assumes a uniform prior assigning mass $1/K$ to each model as we remark below. We replace $\hat{\sigma}_k^2$, the resubstitution estimate of prediction error for model k, with a K-fold cross-validation estimate of prediction error.

Remark D. The relative fit estimator and the other combination estimators of the linear form

$$\sum_{k=1}^{K} \beta_k c_{\mathbf{Z}}^k(\mathbf{x}) \tag{21}$$

can be related to the Bayesian formulation for the prediction problem. For instance, the predictive mean of an a new observation Y_0 with a predictor vector X_0 can be expressed as

$$E(Y_0|\mathbf{z}, X_0) \quad = \quad \int E(Y_0|X_0, \theta, M_k, \beta) p(\theta, \beta, M_k|\mathbf{z}) d\theta d\beta dM_k$$

where M_k represents the kth component model and $\theta = (\theta_1, \cdots \theta_k)$ represents the parameters corresponding to all component models. The predictive mean can be re-expressed as

$$
\begin{aligned}
E(Y_0|\mathbf{z}, X_0) &= \int\int \left[\sum_k E(Y_0|X_0, \theta, M_k, \beta) p(M_k|\theta, \beta, \mathbf{z})\right] p(\theta, \beta|\mathbf{z}) d\theta d\beta \\
&= \int\int \left[\sum_k C(X_0; \theta, M_k) p(M_k|\theta, \beta, \mathbf{z})\right] p(\theta, \beta|\mathbf{z}) d\theta d\beta
\end{aligned}
$$

where $C(X_0; \theta, M_k)$ represents the expected output for model M_k given θ (and β) at X_0. This formulation also motivates the non-negative weights (and weights that sum to one) for the component model outputs. Note, the choice of independent normal priors for β_i corresponds to the ridge type shrinkage and a prior consisting of point masses on the coordinate unit vectors corresponds to the relative fit weights.

6 Examples

6.1 Introduction

In the following examples we compare the combination methods described earlier. Throughout we use 10-fold cross-validation, and $B = 10$ bootstrap samples. Ten-fold cross-validation was found to be superior to leave-one out cross-validation by Breiman (1993), and it is computationally much less demanding. Estimated model error

$$\sum_1^N (f(x_i) - \hat{f}(x_i))^2$$

is reported; $\hat{f}(\mathbf{x}_i)$ is the estimated regression function and $f(\mathbf{x}_i)$ is the true regression function evaluated the observed predictor values $\mathbf{x}_i$. Twenty-five Monte Carlo simulations are used in each example, and the Monte-Carlo means and standard deviations of the mean are reported.

6.2 Combining linear regression models

In this example, which is modified from Breiman (1993), we investigate combinations of best subset regression models.

The predictor variables $X_1, \cdots, X_{30}$ were independent standard normal random variables and the response was generated from the linear model

$$Y = \beta_1 X_1 + \cdots + \beta_m X_{30} + \epsilon$$

where ϵ is a standard normal random variable. The β_m were calculated by $\beta_m = \gamma \alpha_m$, where α_m are defined in clusters:

$$\alpha_m = (h - |m - 7|)^2 I\{|m - 7| < h\} +$$

$$(h - |m - 14|)^2 I\{|m - 14| < h\} +$$
$$(h - |m - 21|)^2 I\{|m - 21| < h\}.$$

The constant γ was determined so that the signal to noise ratio was approximately equal to one. Each simulated data set consisted of 60 observations.

Two values $h = 1$ and $h = 4$ were considered; the case $h = 1$ corresponds to a model with three large non-zero coefficients and $h = 4$ corresponds to a model with many small coefficients. We report the average model errors in Tables 1 and 2.

As expected, for the uncombined methods, ridge regression performs well when there are many small coefficients and best subset regression is the clear choice when there were a few large coefficients.

Most of the combination methods without regularization do not yield smaller model errors than standard methods; the cross-validation combination method and generalized cross-validation method give substantially larger model errors than ordinary least squares, ridge regression or best subsets. Only, the bootstrap method for the three large coefficient model yields smaller model errors than ordinary least squares and ridge regression.

However, regularization substantially reduces the model error of the combination methods. The non-negative estimators seem to perform as well as, and sometimes far better than, the shrinkage estimators and the estimators constrained to sum to one. The bootstrap and the cross validation combination methods with non-negativity constraints yield smaller average model errors than ridge regression and ordinary least squares for both the many small coefficient model and the three large coefficient model and yield results very close to best subsets for the three coefficient model.

The relative fit weights seem to perform well relative to the other combination methods for the model with three large coefficients, but yield somewhat larger errors than some of the combination methods with regularization for the model with many small coefficients.

6.3 Combining tree-based regression models

Tree-based regression procedures typically grow a large model that overfits the data and then prune the tree and select the model, among a nested sequence of models or sub-trees, that minimizes an estimate of prediction error.

We consider the sequence of models derived by the cost-complexity pruning algorithm of the Classification and Regression Tree (CART) algorithm (Breiman, Friedman, Olshen and Stone, 1984). The cost-complexity measure of tree performance is

$$R_\alpha(T) = \sum_{h \in \tilde{T}} R(h) + \alpha \cdot [\ \#\ \text{of terminal nodes in } T\ ],$$

Table 1: Average model errors (and standard errors) for Example 1. Many weak coefficients

Method	Regularization			
	None	Non-negativity	Shrinkage	Sum to one
Least squares	29.3 (1.2)	-	-	-
Ridge (by CV)	25.2 (1.5)	-	-	-
Best Subset (by CV)	33.3 (2.2)	-	-	-
Relative Fit Weights (by CV)	27.6 (1.5)	-	-	-
Combination by least squares	29.3 (1.2)	29.3 (1.2)	29.0 (1.2)	29.3 (1.2)
Combination by CV	69.8 (5.9)	24.0 (1.6)	49.4 (4.2)	66.7 (5.7)
Combination by GCV	60.0 (12.6)	22.4 (1.0)	29.4 (1.4)	30.9 (1.5)
Combination by bootstrap	32.7 (5.6)	21.0 (3.0)	22.2 (3.1)	23.7 (2.9)

Table 2: Average model errors (and standard errors) for Example 1. Three large coefficients

Method	Regularization			
	None	Non-negativity	Shrinkage	Sum to one
Least squares	29.3 (1.2)	-	-	-
Ridge (by CV)	24.1 (0.9)	-	-	-
Best Subset (by CV)	8.5 (1.5)	-	-	-
Relative Fit Weights (by CV)	8.7 (1.5)	-	-	-
Combination by least squares	29.3 (1.2)	29.3 (1.2)	28.9 (1.2)	29.3 (1.2)
Combination by CV	45.3 (3.6)	9.5 (1.1)	28.1 (2.7)	36.7 (2.7)
Combination by GCV	52.5 (18.2)	14.0 (1.0)	27.0 (1.1)	29.5 (1.4)
Combination by bootstrap	16.5 (2.19)	10.1 (1.0)	11.3 (1.1)	12.6 (1.2)

Table 3: Average model error (and standard errors) for regression tree combinations

Method	Regularization		
	None	Non-negativity	shrinkage
Unpruned tree	92.9 (2.5)	-	-
Best tree by CV	68.4 (2.6)	-	-
Combination by Relative Fit	66.2 (2.6)	-	-
Recursive Shrinkage	57.8 (1.5)	-	-
Combination by bootstrap	88.5 (5.0)	60.1 (1.8)	68.8 (2.5)

where α is a penalty per terminal node, $R(h)$ is the resubstitution estimate of prediction error for node h and $\tilde{T}$ are the terminal nodes in the tree. For any value of α the cost complexity pruning algorithm can efficiently obtain the sub-tree of T that minimizes $R_\alpha(T')$ over all subtrees T' of T. We apply the combination methods to the sequence of optimally pruned sub-trees for $0 \leq \alpha < \infty$.

For this example, we assume predictors $X_1, \cdots, X_{10}$ are independent standard normal distributed,

$$f = I\{X_1 > 0\} + I\{X_2 > 0\} + I\{X_2 > 0\}\{X_1 > 0\} + X_3$$

and add noise generated from the standard normal distribution. Data sets of size 250 observations were generated.

The bootstrap combination method with ridge regression, and the tree selected by a 10-fold cross validation estimate of prediction error yielded similar model errors. The bootstrap combination method with the non-negativity constraint yielded slightly larger errors on these data sets than a recursive shrinkage method for tree models (Hastie and Pregibon, 1991, Clark and Pregibon, 1992). However, both of these methods, for this example, give substantially smaller errors than the standard method which selects the best tree by 10-fold cross validation.

We consider the results from one typical simulated data set in more detail. The unpruned tree and the best tree selected by 10-fold cross validation are given in Figures 1 and 2. The bootstrap combination method with non-negative coefficients yielded an estimate of β with three non-zero coefficients

$$\hat{\beta}^T = (0, .093, .396, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, .484, 0),$$

corresponding to pruned sub-trees of size 2, 3 and 20 terminal nodes. Note, the resulting combination model can be represented by the unpruned tree with modified estimates for the terminal nodes. The three tree models corresponding the non-negative coefficients are given in Figure 3.

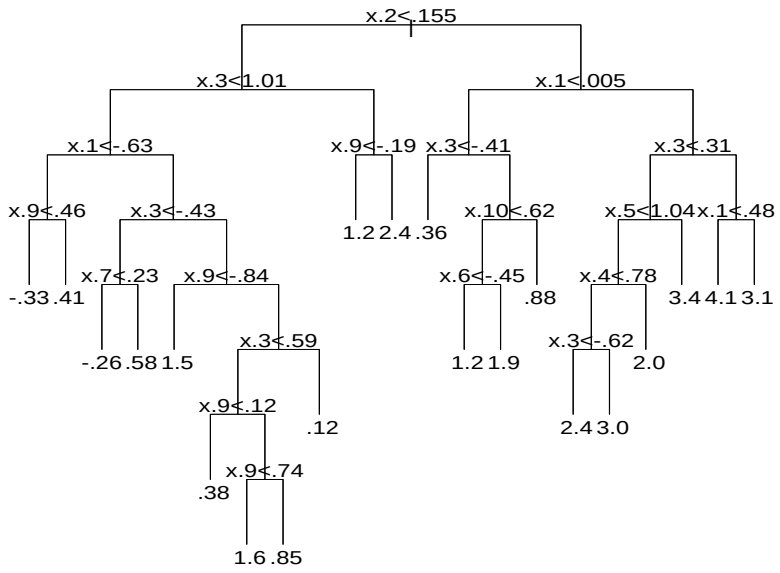

Figure 1: The unpruned tree for a typical realization

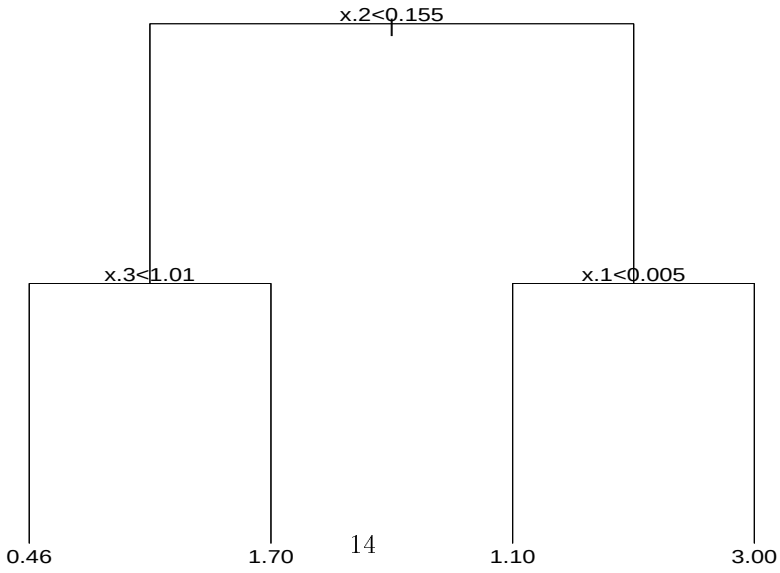

Figure 2: Pruned sub-tree selected by 10-fold cross validation for a typical realization

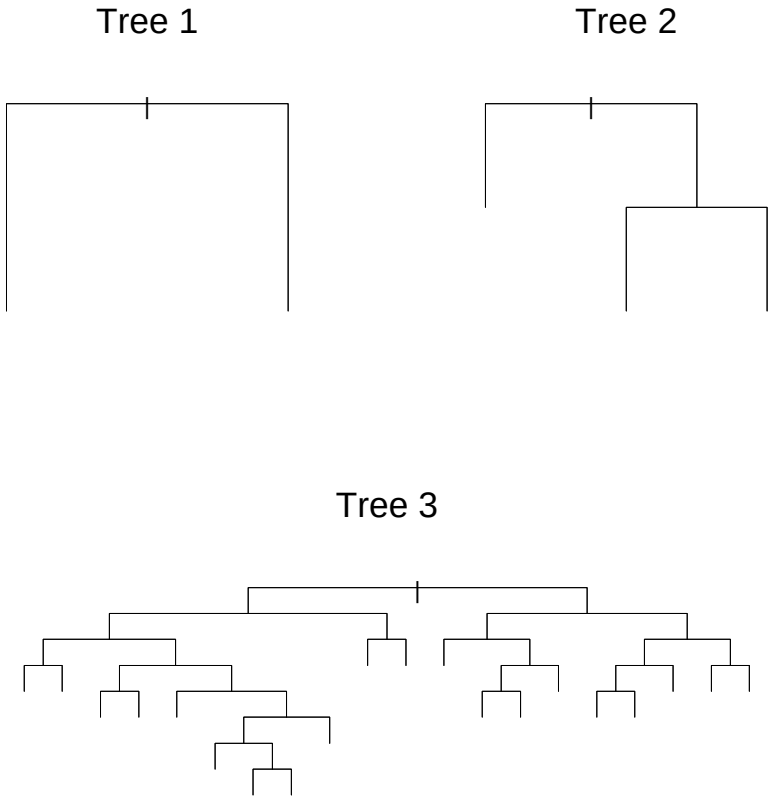

Figure 3: Sub-trees with positive weight for a typical realization

Interpretation of the non-negative coefficient model is aided by using an analysis of variance decomposition to represent the effect branches in the tree. Let $\hat{\mu}(T_j)$ be the fitted values for the tree corresponding to the jth non-zero coefficient. The fitted values of combination model can be expressed as

$$\hat{f} = .973\hat{\mu}(T_1) + .880\left[\hat{\mu}(T_2) - \hat{\mu}(T_1)\right] + .484\left[\hat{\mu}(T_3) - \hat{\mu}(T_2)\right] + 0\left[\hat{\mu}(T_{\text{full}}) - \hat{\mu}(T_3)\right] \quad (22)$$

where T_{full} is the unpruned tree. Therefore, one can interpret the combination with non-negative constraints as an increasing shrinkage of the effects in the lower branches of the tree. This explains, in part, why the performance of this combination method and the recursive shrinkage method were similar on average. Note, for data sets which yield some shrinkage factors close to 0 in a decomposition analogous to (22), one could set those factors to zero, effectively pruning the tree, to obtain a more interpretable model.

7 Classification problems

In a classification problem, the outcome is not continuous but rather falls into one of J outcome classes. We can view this as a regression problem with a multivariate J-valued response y having a one in the jth position if the observation falls in class j and zero otherwise.

Most classification procedures provide estimated class probabilities functions $\hat{\mathbf{p}}(\mathbf{x}) = (\hat{p}_1(\mathbf{x}), \ldots \hat{p}_J(\mathbf{x}))^T$. Given K such functions $\hat{\mathbf{p}}^k(\mathbf{x}) = (\hat{p}_1^k(\mathbf{x}), \ldots \hat{p}_J^k(\mathbf{x}))^T$, $k = 1, 2, \ldots K$, it seems reasonable to apply the combination strategies either to the probabilities themselves or their log ratios. That is, we take $c_{\mathbf{Z}}^{kj}(\mathbf{x})$, the jth component of $c_{\mathbf{Z}}^{k}(\mathbf{x})$, to be either

$$c_{\mathbf{Z}}^{kj}(\mathbf{x}) = \hat{p}_j^k(\mathbf{x}) \quad (23)$$

or

$$c_{\mathbf{Z}}^{kj}(\mathbf{x}) = \log \frac{\hat{p}_j^k(\mathbf{x})}{\hat{p}_J^k(\mathbf{x})} \quad (24)$$

In our experiments, we found that the use of probabilities (23) gives better classification results and is more interpretable.

Let $\hat{\mathbf{P}}(\mathbf{x}) = (\hat{p}_1^1(\mathbf{x}), \hat{p}_2^1(\mathbf{x}), \ldots \hat{p}_J^1(\mathbf{x}), \hat{p}_1^2(\mathbf{x}), \hat{p}_2^2(\mathbf{x}), \ldots)^T$, a vector of length JK. Then the combination estimator has the form

$$\theta_j(\mathbf{x}) = \beta_j^T \hat{\mathbf{P}}(\mathbf{x}); \quad j = 1, 2, \ldots J, \quad (25)$$

where β_j is a JK-vector of weights. We then apply the straightforward multivariate analogues of the procedures described earlier. The population regression problem (4) becomes a multi-response regression, producing vector-valued

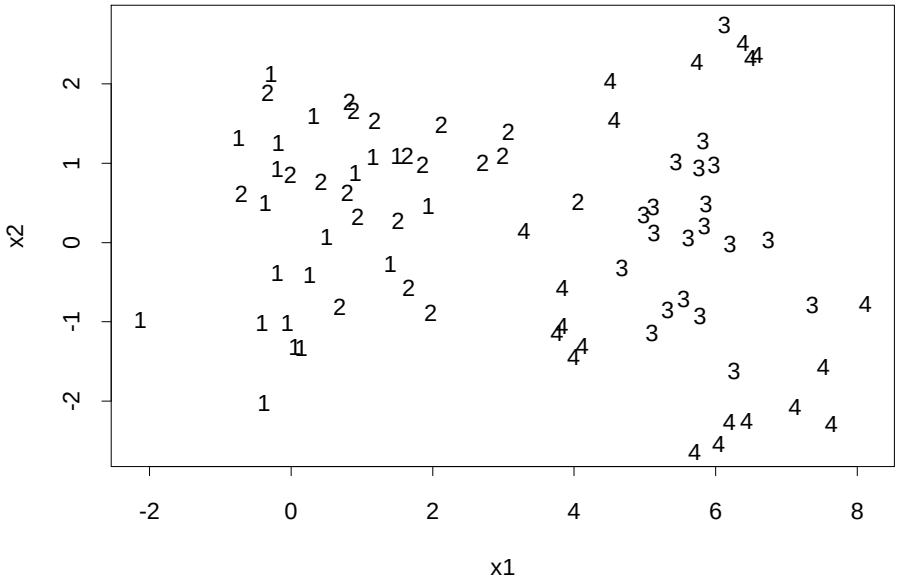

Figure 4: Typical realization for example 2.

estimates $\hat{\beta}_j^T$. The combination estimators are computed in an analogous fashion to the regression case. Finally, given the combination estimators $\hat{\theta}_j(\mathbf{x}) = \hat{\beta}_j^T \mathbf{P}(\mathbf{x})$, the estimated class probabilities are taken to be $\hat{\theta}_j(\mathbf{x}) / \sum_j \hat{\theta}_j(\mathbf{x})$. A non-negativity constraint for the weights is especially attractive here, because it ensures that these estimated probabilities lie between 0 and 1.

7.1 Example 2.: combining linear discriminant analysis and nearest neighbours

In this example the data are two dimensional, with 4 classes. A typical data realization is shown in figure 4. Classes 1, 2 and 3 are standard independent normal, centered at (0,0), (1,1) and (6,0) respectively. Class 4 was generated as standard independent normal with mean (6,0), conditional on $4.5 \leq x_1^2 + x_2^2 \leq 8$. There were 10 observations in each class.

We consider combination of linear discriminant analysis (LDA) and 1-nearest neighbour method. The idea is that LDA should work best for classes 1 and 2, but not for classes 3 and 4. Nearest neighbour should work moderately well for

Table 4: Average % test error (and standard deviations) for Example 3.

Method	Regularization	
	None	Non-negativity
LDA	36.3 (.64)	-
1-nearest neighbour	27.5 (.87)	-
Combination by least squares	26.1 (.94)	25.2 (.90)
Combination by bootstrap	27.0 (.95)	25.7 (.95)

Table 5: Average combination weights for Example 3.

	LDA				1-nearest neighbour			
Class	1	2	3	4	1	2	3	4
1	0.75	0.00	0.00	0.00	0.31	0.00	0.00	0.00
2	0.00	0.80	0.00	0.00	0.00	0.29	0.00	0.00
3	0.00	0.00	0.02	0.07	0.00	0.00	0.95	0.00
4	0.00	0.00	0.01	0.01	0.00	0.00	0.00	1.13

classes 1 and 2, and much better than LDA for classes 3 and 4. By combining the two, we might be able to improve on both of them.

To proceed, we require from each method, a set of estimated class probabilities for each observation. We used the estimated Gaussian probabilities for LDA; for 1-nearest neighbour we estimated the class j probability for feature vector $\mathbf{x}$ by $\exp(-d^2(\mathbf{x},j))/\sum_{j=1}^{J}\exp(-d^2(\mathbf{x},j))$, where $d^2(\mathbf{x},j)$ is the squared distance from $\mathbf{x}$ to the nearest neighbour in class j. The results of 10 simulations of the combined LDA/nearest neighbour procedure are shown in Table 4

We see that simple least squares combination, or regularized combination by bootstrap, slightly improves upon the LDA and 1-nearest neighbour rules. More interesting are the estimated combination weights produced by the non-negativity constraint. Table 5 shows the average weights from the combination by bootstrap (the combination by least squares weights are very similar). LDA gets higher weight for classes 1 and 2, while 1-nearest neighbour is used almost exclusively for classes 3 and 4, where the structure is highly nonlinear.

In general, this procedure might prove to be useful in determining which outcome classes are linearly separable, and which are not.

8 More general combination schemes

More general combination schemes would allow the β_ks to vary with $\mathbf{x}$:

$$\sum_1^K \beta_k(\mathbf{x}) c_{\mathbf{Z}}^k(\mathbf{x}). \tag{26}$$

A special case of this would allow β_k to vary only with $c_{\mathbf{Z}}^k$, that is

$$\sum_1^K \beta_k(c_{\mathbf{Z}}^k(\mathbf{x})) c_{\mathbf{Z}}^k(\mathbf{x}). \tag{27}$$

Both of these models fall into the category of the "varying coefficient" model discussed in Hastie and Tibshirani (1993). The difficulty in general is how to estimate the functions $\beta_k(\cdot)$; this might be easier in model (27) than in model (26) since the $c_{\mathbf{Z}}^k$s are all real-valued and are probably of lower dimension than $\mathbf{x}$.

One application of varying combination weights would be in scatterplot smoothing. Here each $c_{\mathbf{Z}}^k(x)$ is a scatterplot smooth with a different smoothing parameter λ_k. The simple combination $\sum \beta_k c_{\mathbf{Z}}^k(x)$ gives another family of estimators that are typically outside of the original family of estimators indexed by λ. However, it would seem more promising to allow each β_k to be a function of x and hence allow local combination of the estimates. Requiring $\beta_k(x)$ to be a smooth function of x would ensure that the combination estimator is also smooth.

Potentially one could use the ideas here for more general parameter selection problems. Suppose we have a regression estimator denoted by $\eta_{\mathbf{Z}}(\mathbf{x}, \beta)$. The estimator is computed on the data set $\mathbf{z}$, with an adjustable (tuning) parameter vector β, and gives a prediction at $\mathbf{x}$. We wish to estimate the value of β giving the smallest prediction error

$$g(\mathbf{z}, F, \beta) = \mathrm{E}_{0F}(Y_0 - \eta_{\mathbf{Z}}(\mathbf{X}_0, \beta))^2. \tag{28}$$

Then we can apply the bootstrap technique of section 3 to estimate the bias in $g(\mathbf{z}, \hat{F}, \beta) = \sum_{i=1}^N (y_i - \eta_{\mathbf{Z}}(\mathbf{x}_i, \beta)^2$. Using the notation of section 3, the biased corrected estimator has the form

$$\hat{g}(\mathbf{z}, \hat{F}, \beta) = \sum_{i=1}^N (y_i - \eta_{\mathbf{Z}}(x_i, \hat{\beta}))^2 + \hat{\Delta}(\hat{F}, \beta) \tag{29}$$

where

$$\hat{\Delta}(\hat{F}, \beta) = \quad \frac{1}{B}\left[\frac{1}{N}\sum_{i=1}^N (y_i - \eta_{\mathbf{Z}^{*b}}(\mathbf{x}_i, \beta))^2 - \frac{1}{N}\sum_{i=1}^N (y_i^{*b} - \eta_{\mathbf{Z}^{*b}}(\mathbf{x}_i^{*b}, \beta))^2\right].$$

We would then minimize $\hat{g}(\mathbf{z}, \hat{F}, \beta)$ over β. This idea may only be useful if $\eta_{\mathbf{Z}}(\mathbf{x}, \beta)$ can be written as an explicit function of β, so that $\hat{g}(\mathbf{z}, \hat{F}, \beta)$ and its

derivatives can be easily computed. In this paper we have considered the linear estimator $\eta_{\mathbf{Z}}(\mathbf{x}, \beta) = \sum \beta_k c^k_{\mathbf{Z}}(\mathbf{x})$ for which the minimizer of $\hat{g}(\mathbf{z}, \hat{F}, \beta)$ can be explicitly derived by least squares. The variable kernel estimator of Lowe (1993) is another example where this procedure could be applied: in fact, Lowe uses the cross-validation version of the above procedure to estimate the adjustable parameters β. In many adaptive procedures, e.g. tree-based regression, the estimator $\eta_{\mathbf{Z}}(\mathbf{x}, \beta)$ is a complicated function of its tuning parameters, so that minimization of $\hat{g}(\mathbf{z}, \hat{F}, \beta)$ would be quite difficult.

9 Discussion

This investigation suggests that combining of estimators, used with some regularization, can be a useful tool both for improving prediction performance and for learning about the structure of the problem. We have derived and studied a number of procedures and found that cross-validation (stacking) and the bootstrap, used with a non-negativity constraint, seemed to work best. The bootstrap estimates seem to require far less regularization; the reason for this is not clear.

An interesting question for further study is: how can we choose estimators for which combining will be effective?

ACKNOWLEDGEMENTS

We would like to thank Mike Escobar, Trevor Hastie and Geoffrey Hinton for helpful discussions and suggestions. Both authors gratefully acknowledge the support of the Natural Science and Engineering Research Council of Canada.

References

Breiman, L. (1993). Stacked regression. Technical report, Univ. of Cal, Berkeley.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Wadsworth International Group.

Clark L. and Pregibon D. (1992). *Statistical Models in S*, chapter Tree-Based models. Wadsworth International Group.

Efron, B. (1983). Estimating the error rate of a prediction rule: some improvements on cross-validation. *J. Amer. Statist. Assoc.*, 78:316–331.

Efron, B. and Tibshirani, R. (1993). *An Introduction to the Bootstrap*. Chapman and Hall, New York.

Hastie, T.J. and Tibshirani, R.J. (1993). Varying coefficient models (with discussion). *J. Royal Statist. Soc. B*, 55:757–796.

Lawson, C.L. and Hanson, R.J. (1974). *Solving Least Squares Problems.* Prentice-Hall: Englewood Cliffs, N.J.

Lowe, David G. (1993). Similarity metric learning for a variable kernel classifier. Technical report, Dept. of Comp Sci, Univ. of British Columbia.

McCullagh, P. and Tibshirani, R. (1988). A simple adjustment for profile likelihoods. *J. Royal. Statist. Soc. B.*, 52(2):325–344.

Wolpert, D. (1992). Stacked generalization. *Neural Networks*, 5:241–259.