General set up for Aim 2b:

We assume that our training set has been split into two independent subsets, $(Z_{0i}, W_{0i}), i = 1 \ldots n_0$ and a "test set" $(Z_{1i}, W_{1i}), i = 1 \ldots n_1$. We first apply an aggregate learner (e.g., $partDSA_{RF}$) to the training set $(Z_{0i}, W_{0i}), i = 1 \ldots n_0$, generating the (black box) prediction rule, $\hat{m}(w)$; then, we forecast predicted outcomes in the test set $\widehat{Z}_{1i} = \hat{m}(W_{1i}), i = 1 \ldots n$. Importantly: $\hat{m}(w)$ and $(Z_{1i}, W_{1i}), i = 1 \ldots n_1$ can be considered independent; in addition, the predictions $\widehat{Z}_{1i}, i = 1 \ldots n$ only utilize the W_{1i}s and not the Z_{1i}s. Importantly, $\widehat{Z}_{1i}, i = 1 \ldots n$ will not be used as covariates but rather in creating a target for shrinkage, thereby reducing variance.

Starting with all of the data, under L_2 loss, both *partDSA* and CART would ordinarily seek to minimize

$$\min_{c_L} \textstyle\sum_i I\{W_{1i} \in Q_L(j,s)\}(Z_{1i} - c_L)^2 + \min_{c_R} \textstyle\sum_i I\{W_{1i} \in Q_R(j,s)\}(Z_{1i} - c_R)^2 \qquad (1)$$

over all variables j and split points s, where $Q_L(j,s) = \{W | W_j \leq s\}$ and $Q_R(j,s) = \{W | W_j > s\}$ The (j,s) combination minimizing (**??**) can be determined quickly; in CART, the "best" choice is used to divide the data into two groups (nodes) and the above splitting process is then repeated within each node. *partDSA* proceeds similarly, making use of the above in the addition and substitution steps. Strategies 1 & 2 are discussed in the grant application and represent methods for modifying the loss function (**??**) used for split determination, making use of the $\widehat{Z}_{1i}$s to help guide splitting decisions. This document provides further details that underpin Strategy 2, which is described below.

Strategy 2: In addition to $\widehat{m}(\cdot)$, the ensemble building process provides (i) a measure of prediction error, $\hat{\sigma}_0^2$, derived from $(Z_{0i}, W_{0i}), i = 1 \ldots n_0$; and, (ii) B predicted outcomes for each W_{1i}, generating both a predicted mean $\widehat{Z}_{1i}$ and measure of variance, $\hat{\gamma}_i$. Strategy 2 leverages this information through penalization; the crux of this proposal involves replacing, for $v \in \{L, R\}$, the two optimization problems in (**??**) with

$$\min_{c_v} \textstyle\sum_i I\{W_{1i} \in Q_v(j,s)\} \left[(Z_{1i} - c_v)^2 + \lambda \alpha_i (c_v - \widehat{Z}_{1i})^2\right] \qquad (2)$$

for $\alpha_i > 0$, each having the weighted-average solution

$$\widehat{c}_{v,j,s}(\lambda) = r_{v,j,s}(\lambda)\bar{Z}_v(j,s) + (1 - r_{v,j,s}(\lambda))\bar{\hat{Z}}_{1v}(j,s),$$

where $r_{v,j,s}(\lambda) = 1/(1 + \lambda \bar{\alpha}_{v,j,s})$,

$$\bar{\hat{Z}}_{1v}(j,s) = \{\textstyle\sum_{W_{1i} \in Q_v(j,s)} \alpha_i \hat{Z}_{1i}\} / \{\textstyle\sum_{W_{1i} \in Q_v(j,s)} \alpha_i\}$$

and

$$\bar{\alpha}_{v,j,s} = \{\sum\nolimits_{W_{1i} \in Q_v(j,s)} \alpha_i\} / \{\sum_i I\{W_{1i} \in Q_v(j,s)\}\}.$$

The choice $\alpha_i^{-1} = \text{var}(\hat{Z}_{1i})$ is probably best from an efficiency perpsective; hence the choice of $\alpha_i = \hat{\gamma}_i^{-1}$ in the grant application text.

Note that, given $\bar{\alpha}_{j,v,s}$, more weight is placed on $\hat{\bar{Z}}_{1v}(j,s)$ when λ is larger; similarly, given λ, more weight is placed on $\hat{\bar{Z}}_{1v}(j,s)$ when $\bar{\alpha}_{j,v,s}$ is larger. Both make sense in shrinking the parameter estimate towards this weighted mean function.

One issue to be decided whether the current version of strategy is the most appropriate formulation. Another key implementation issue is choosing λ. We discuss these in more detail towards the end.

1 Important background calculations

1.1 Background for estimating c_L and c_R

Let $\omega_i, i \geq 1$ be nonnegative weights, where at least one is positive. Let $\alpha_i, i \geq 1$ and $A_i, i \geq 1$ respectively be sequences of positive and real-valued constants. Let $Z_i, i \geq 1$ be a sequence of random variables. Finally let $\lambda > 0$ be given and consider the problem of minimizing

$$Q(c) = \textstyle\sum_i \omega_i \left[(Z_i - c)^2 + \lambda\alpha_i(c - A_i)^2\right]$$

in c. If we differentiate $Q(c)$ with respect to c, set $Q'(c) = 0$, and solve for c, we obtain

$$c(\lambda) = \frac{\sum_i \omega_i Z_i}{\lambda \sum_i \omega_i \alpha_i + \sum_i \omega_i} + \frac{\lambda \sum_i \omega_i \alpha_i A_i}{\lambda \sum_i \omega_i \alpha_i + \sum_i \omega_i}.$$

Doing a bit of algebra,

$$c(\lambda) = r(\lambda)\frac{\sum_i \omega_i Z_i}{\sum_i \omega_i} + (1 - r(\lambda))\frac{\sum_i \omega_i \alpha_i A_i}{\sum_i \omega_i \alpha_i}. \tag{3}$$

where

$$r(\lambda) = \frac{\sum_i \omega_i}{\lambda \sum_i \omega_i \alpha_i + \sum_i \omega_i} = \frac{1}{(1 + \lambda\bar{\alpha})}, \tag{4}$$

and

$$\bar{\alpha} = \frac{\sum_i \omega_i \alpha_i}{\sum_i \omega_i}. \tag{5}$$

Clearly, $r(\lambda) \in [0,1]$ and so this is just a shrinkage estimator that balances the observed weighted average (which we'd get if $\lambda = 0$)

$$\frac{\sum_i \omega_i Z_i}{\sum_i \omega_i}$$

with the $\alpha-$modified weighted average of the As

$$\frac{\sum_i \omega_i \alpha_i A_i}{\sum_i \omega_i \alpha_i}.$$

(which we'd get as $\lambda \to \infty$).

1.2 Derivation of Within-Node Prediction Error

Now, let $\tilde{Z}$ be independent of $H = \{Z_i, i \geq 1\}$. We can define the conditional prediction error using $c(\lambda)$ as

$$CPE(\lambda) = E\left[(\tilde{Z} - c(\lambda))^2 | H\right]$$

and the prediction error $PE(\lambda) = E_H\left[CPE(\lambda)\right]$ (here, E_H denotes the expectation wrt distribution of H). We would like to know what λ minimizes $PE(\lambda)$. Note that $c(\lambda)$ is known given H under the assumptions made at the beginning of the previous subsection.

We can write

$$(\tilde{Z} - c(\lambda))^2 = \tilde{Z}^2 - 2\tilde{Z}c(\lambda) + [c(\lambda)]^2.$$

Defining $\sigma_Z^2 = var(\tilde{Z})$ and $\mu_Z = E[\tilde{Z}]$ we have

$$CPE(\lambda) = \sigma_Z^2 + \mu_Z^2 - 2\mu_Z c(\lambda) + [c(\lambda)]^2.$$

Hence

$$PE(\lambda) = \sigma_Z^2 + \mu_Z^2 - 2\mu_Z E_H[c(\lambda)] + E_H[[c(\lambda)]^2].$$

Let $\mu_c(\lambda) = E_H[c(\lambda)]$ and $\sigma_c^2(\lambda) = var_H(c(\lambda))$; then, we can rewrite this last expression as

$$PE(\lambda) = \sigma_Z^2 + \mu_Z^2 - 2\mu_Z\mu_c(\lambda) + \sigma_c^2(\lambda) + \mu_c^2(\lambda).$$

Now, suppose $E[Z_i] = \delta$ for each i; then,

$$\mu_c(\lambda) = r(\lambda)\delta + K_1(1 - r(\lambda)).$$

for

$$K_1 = \frac{\sum_i \omega_i \alpha_i A_i}{\sum_i \omega_i \alpha_i}.$$

Similarly, if $var(Z_i) = \gamma$ for each i, then

$$var_c(\lambda) = r^2(\lambda) K_2 \gamma$$

for

$$K_2 = \frac{\sum_i \omega_i^2}{[\sum_i \omega_i]^2}$$

As result we may write

$$PE(\lambda) = \sigma_Z^2 + \mu_Z^2 - 2\mu_Z[r(\lambda)\delta + K_1(1 - r(\lambda))] + r^2(\lambda)K_2\gamma + [r(\lambda)\delta + K_1(1 - r(\lambda))]^2$$

In the special case where $\delta = \mu_Z$ and $\gamma = \sigma_Z^2$, differentiating $PE(\lambda) = 0$ with respect to λ and solving $PE'(\lambda) = 0$ gives

$$\lambda_0 = \frac{K_2\sigma_Z^2}{\bar{\alpha}(\mu_z - K_1)^2}, \tag{6}$$

where $\bar{\alpha}$ is given in (**??**).

2 First application of results to Strategy 2

.

To connect the notation of the Aim2b set-up and the notation of Section **??**, let $\omega_i = I\{W_{1i} \in Q_v(j,s)\}$, $Z_i = Z_{1i}$ and $A_i = \hat{Z}_{1i}$. Then, the formulas (**??**)-(**??**) of Section **??** give

$$\widehat{c}_{v,j,s}(\lambda) = r_{v,j,s}(\lambda)\bar{Z}_v(j,s) + (1 - r_{v,j,s}(\lambda))\hat{\bar{Z}}_{1v}(j,s),$$

where $r_{v,j,s}(\lambda) = 1/(1 + \lambda\bar{\alpha}_{v,j,s})$,

$$\hat{\bar{Z}}_{1v}(j,s) = \{\sum_i I(W_{1i} \in Q_v(j,s))\alpha_i \hat{Z}_{1i}\}/\{\sum_i I(W_{1i} \in Q_v(j,s))\alpha_i\}$$

and

$$\bar{\alpha}_{v,j,s} = n_{v,j,s}^{-1} \sum_i I(W_{1i} \in Q_v(j,s))\alpha_i$$

where $n_{v,j,s} = \sum_i I(W_{1i} \in Q_v(j,s))$.

Applying the results of Section **??** and assuming all expectation calculations are conditional on $W_{1i}, i \geq 1$ it can be shown that

$$K_2 = n_{v,j,s}^{-1}$$

and

$$K_1 = \hat{\bar{Z}}_{1v}(j,s).$$

Assuming that $E(Z_i) = \mu_Z$ and $var(Z_i) = \sigma_Z^2$ when $I(W_{1i} \in Q_v(j,s)) = 1$ (i.e., constant mean and variance within a node), the "best" within-node choice of λ via (**??**) becomes

$$\lambda_{opt} = \frac{n_{v,j,s}^{-1}\sigma_Z^2}{\bar{\alpha}_{v,j,s}(\mu_z - \hat{\bar{Z}}_{1v}(j,s))^2}. \tag{7}$$

3 Alternative view of Strategy 2

.

Let $\omega_i = I\{W_{1i} \in Q_v(j,s)\}$, $Z_i = Z_{1i}$ and $A_i = A_{v,j,s} I\{W_{1i} \in Q_v(j,s)\}$ (i.e., $A_{v,j,s}$ doesn't depend on i and thus is constant within node). Then, the formulas (**??**)-(**??**) of Section **??** give

$$\widehat{c}_{v,j,s}(\lambda) = r_{v,j,s}(\lambda)\bar{Z}_v(j,s) + (1 - r_{v,j,s}(\lambda))\hat{\bar{A}}_{1v}(j,s),$$

where $r_{v,j,s}(\lambda) = 1/(1 + \lambda\bar{\alpha}_{v,j,s})$,

$$\hat{\bar{A}}_{1v}(j,s) = \{\sum_i I(W_{1i} \in Q_v(j,s))\alpha_i A_{v,j,s}\}/\{\sum_i I(W_{1i} \in Q_v(j,s))\alpha_i\} = A_{v,j,s}$$

and

$$\bar{\alpha}_{v,j,s} = n_{v,j,s}^{-1} \sum_i I(W_{1i} \in Q_v(j,s))\alpha_i$$

where $n_{v,j,s} = \sum_i I(W_{1i} \in Q_v(j,s))$.

Applying the results of Section **??** and assuming all expectation calculations are conditional on $W_{1i}, i \geq 1$ it can be shown that

$$K_2 = n_{v,j,s}^{-1}$$

and

$$K_1 = A_{v,j,s}.$$

Assuming that $E(Z_i) = \mu_Z$ and $var(Z_i) = \sigma_Z^2$ when $I(W_{1i} \in Q_v(j,s)) = 1$ (i.e., constant mean and variance within a node), the "best" within-node choice of λ via (**??**) becomes

$$\lambda_{opt} = \frac{n_{v,j,s}^{-1}\sigma_Z^2}{\bar{\alpha}_{v,j,s}(\mu_z - A_{v,j,s})^2}.$$

Notice that selecting

$$A_{v,j,s} = \hat{\bar{Z}}_{1v}(j,s) = \{\sum_i I(W_{1i} \in Q_v(j,s))\alpha_i \hat{Z}_{1i}\}/\{\sum_i I(W_{1i} \in Q_v(j,s))\alpha_i\}$$

gives the same results as in the last section. Selecting instead

$$A_{v,j,s} = \hat{\bar{Z}}_{1v}(j,s) = \{\sum_i I(W_{1i} \in Q_v(j,s))\hat{Z}_{1i}\}/\{\sum_i I(W_{1i} \in Q_v(j,s))\}$$

gives an alternative shrinkage target. There are other choices as well.

The point here is that Strategy 2 can be viewed as a procedure for shrinking the node-specific estimates towards some node-specific average predicted value.

4 Tree Building and Prediction Error

We assume that our training set has been split into two independent subsets, $(Z_{0i}, W_{0i}), i = 1 \ldots n_0$ and $(Z_{1i}, W_{1i}), i = 1 \ldots n_1$. Some aggregate learner is applied to the set $(Z_{0i}, W_{0i}), i = 1 \ldots n_0$, generating the (black box) prediction rule, $\hat{m}(w)$; then, we forecast predicted outcomes in the independent set $\widehat{Z}_{1i} = \hat{m}(W_{1i}), i = 1 \ldots n$. Observe that $\hat{m}(w)$ and $(Z_{1i}, W_{1i}), i = 1 \ldots n_1$ can be considered independent; in addition, the predictions $\widehat{Z}_{1i}, i = 1 \ldots n$ only utilize the W_{1i}s and not the Z_{1i}s.

Suppose that $\hat{\mu}(W, \lambda)$ denotes the final prediction rule obtained using the data $(Z_{1i}, W_{1i}, \hat{Z}_{1i}), i = 1 \ldots n_1$ – meaning, this is obtained from our proposed penalized loss procedure for fixed λ. Let $\mathcal{N}_1(\lambda), \ldots, \mathcal{N}_{K(\lambda)}(\lambda)$ be the partitions obtained in the final structure built with fixed λ; then, we know

$$\hat{\mu}(W, \lambda) = \sum_{j=1}^{K(\lambda)} I\{W_{1i} \in \mathcal{N}_j(\lambda)\}\hat{c}_j(\lambda)$$

(piecewise constant predictor within each partition/node). Here,

$$\hat{c}_j(\lambda) = r_j(\lambda)\bar{Z}_{1j} + (1 - r_j(\lambda))\hat{\bar{Z}}_{1j}$$

where $r_j(\lambda) = 1/(1 + \lambda\bar{\alpha}_j(\lambda))$, $\bar{Z}_{1j}$ is the node-specific mean of the Z_{1i}s,

$$\bar{\hat{Z}}_{1j} = \{\sum_i I(W_{1i} \in \mathcal{N}_j(\lambda)\alpha_i\widehat{Z}_{1i}\}/\{\sum_i I(W_{1i} \in \mathcal{N}_j(\lambda))\alpha_i\}$$

and

$$\bar{\alpha}_j(\lambda) = \{\sum_i I(W_{1i} \in \mathcal{N}_j(\lambda))\alpha_i\}/\{\sum_i I(W_{1i} \in \mathcal{N}_j(\lambda))\}.$$

This is a very complicated function of λ and the within-node procedure described in Section **??** probably cannot be directly adapted to choose a global λ.

Per Efron & Tibshirani (1993) and Efron (2004),

$$\text{err}(\lambda) := \sum_{i=1}^{n_1}(Z_{1i} - \hat{\mu}(W_{1i}, \lambda))^2$$

is a version of the "apparent" prediction error because $\hat{\mu}(W_{1i}, \lambda)$ is built using the data $(Z_{1i}, W_{1i}, \hat{Z}_{1i}), i = 1 \ldots n_1$. This is an optimistic assessment of error and we don't want to use it to choose λ. Following Efron (2004) a preferred measure of error is

$$\text{Err}(\lambda) := E_{Z_{20},W_{20}}\left[(Z_{20} - \hat{\mu}(W_{20}, \lambda))^2\right]$$

where (Z_{20}, W_{20}) is independent of $(Z_{1i}, W_{1i}), i = 1 \ldots n_1$ and $\hat{\mu}(w, \lambda)$ is held fixed in the expectation calculation. Calculations in Efron (2004, Eqn. 2.8) show

$$E[\text{Err}(\lambda)] = E[\text{err}(\lambda) + 2cov(Z_{20}, \hat{\mu}(W_{20}, \lambda))];$$

this implies $\text{err}(\lambda) + 2cov(Z_{20}, \hat{\mu}(W_{20}, \lambda))$ is an unbiased estimator of $E[\text{Err}(\lambda)]$ (which is just the expected prediction error); here, the covariance term acts as a bias correction. However, except in simple linear smoothing problems, $cov(Z_{20}, \hat{\mu}(W_{20}, \lambda))$ is not easy to calculate or otherwise estimate analytically.

Efron (2004) proposes to use a parametric bootstrap procedure to deal with this problem. Again, consider a fixed λ. Following Efron (2004), suppose we generate the b^{th} bootstrap sample $Z^*_{1i}(b) \sim N(\hat{\mu}(W_{1i}, \lambda), \hat{\sigma}^2), i = 1, \ldots, n_1$ where

$$\hat{\sigma}^2 = n_1^{-1}\sum_{i=1}^{n_1}(Z_{1i} - \hat{\mu}(W_{1i}, \lambda))^2.$$

Although I'm not 100% certain, it seems to me that one could instead use $\hat{m}(\cdot)$ in place of $\hat{\mu}(W_{1i}, \lambda)$ to generate bootstrap samples as described above; this might be preferred since bootstrapping doesn't depend on λ and we only need to do this once.

For each $b = 1, \ldots, B$ we run our code on $\{(Z^*_{1i}(b), W_{1i}, \hat{Z}_{1i}), i = 1, \ldots, n_1\}$ to obtain a new $\hat{\mu}^*(w, \lambda)$. We can compute for each $i = 1, \ldots, n_1$

$$C^*_i(\lambda) = \frac{1}{B-1}\sum_{b=1}^{B}\hat{\mu}^*(W_{1i}, \lambda)(Z^*_{1i}(b) - \bar{Z}^*_{1i}), \qquad \bar{Z}^*_{1i} = \frac{1}{B}\sum_{b=1}^{B}Z^*_{1i}(b).$$

and then define the boostrap corrected error as

$$\text{err}_{cor}(\lambda) = \text{err}(\lambda) + 2\sum_{i=1}^{n_1} C_i^*(\lambda)$$

If run for λ on a grid then it should be possible to choose the λ that minimizes this quantity (or a smoothed version of it).

5 Code

Code has been written that implements Strategy 2. The key step not fully described earlier is estimating λ. We have started with the λ estimate in (7) at the root node. We multiply that estimate by a constant c and do a grid search on [0,cλ]. The final $\hat{\lambda}$ is the one gives the best prediction error in three-fold cross-validation. In this cross-validation one third of the data is used to fit a random forest (training set), a second third is used to build a tree that optimizes (2) (test set), and the last third is used to estimate prediction error (validation set). This process is repeated so that each third is used only once as the training, test, or validation set. (There are actually six combinations that could be used, but right now we are using only three of them.)

Once we have $\hat{\lambda}$ we build a tree based on (2). Half the data is used for the training set and half the data is used for the test set. The function to do the work is called aim2, and it can be found in the file code.R. To build an rpart tree by hand, a list of functions needs to be fed to the rpart call. Here we call that list aim2.list, and use the argument method=aim2.list. Important functions are an initialization function (aim2.init), an evaluation function (aim2.eval), and a splitting function (aim2.split). Note that in aim2.init my y variable contains three columns: the outcome variables Z_{1i}, the αs, and the predicted values $\widehat{Z_{1i}}$. In aim2.eval the value of (2) is computed for the chosen split. In aim2.split the optimal split is found. This is done currently by looping through every value of every variable. Future effort will be undertaken to see if the loop can be removed.

The input to aim2 is

- dat is data frame to which model is fit
- nreps is not used right now
- ngrid is the number of lambdas in the grid search
- mult is the number multiplied times the intial lambda the is the maximum lambda in the grid search
- seed fixes the random number generator for reproducibility
- outvar is the name of the outcome variable in the fitting

The output form aim2 is

- final.fit is rpart tree chosen by optimal lambda

- lambdas are the values of lambda from grid search
- final.lambda is the optimal lambda chosen by 3-fold cross-validation

The last three lines of code.R shows how to run aim2 on the Boston housing data.