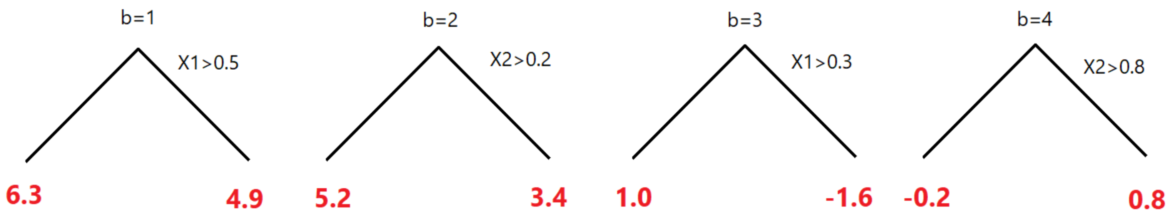
STA 325: Homework 4

DUE: 11:59pm, November 11 (on Sakai)
COVERAGE: ISL Chapter 8

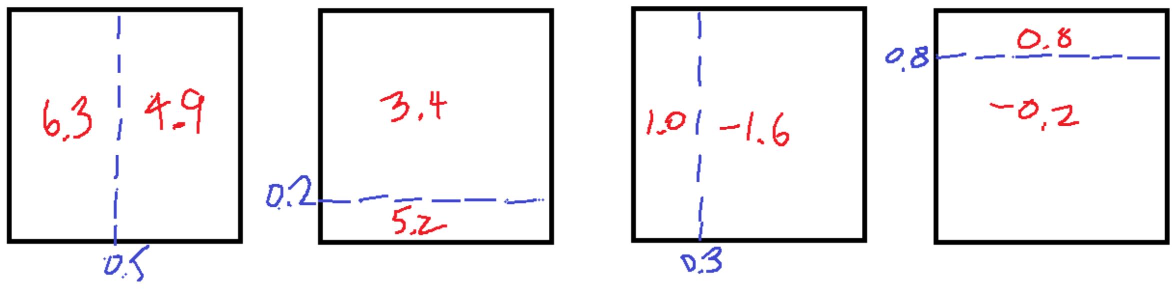
1. [**40 points**] Let's investigate some interesting properties of the boosting predictor.

   (a) [**4 points**] Suppose you run boosting on a dataset with two predictors, using an interaction depth of $d = 1$, i.e., only trees of one split (stump trees) are allowed. With a learning rate of $\lambda = 0.1$, the boosting procedure returns the following $B = 4$ decision trees $\{\hat{f}^b\}_{b=1}^4$:



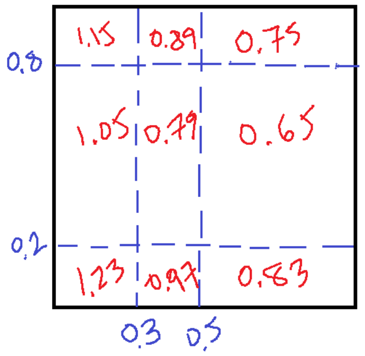
   Draw out each tree as a partition of the prediction space $\mathcal{X} = \{(x_1, x_2) : x_1 \in [0,1], x_2 \in [0,1]\}$. Label clearly each split and the predicted value within each partition.

   Answer:



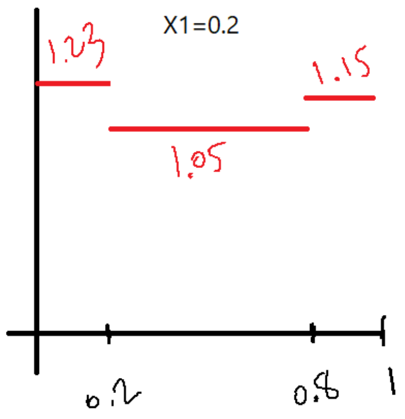
   (b) [**5 points**] Using part (a), draw out the boosted predictor $\hat{f}$ as a partition of the prediction space $\mathcal{X}$. Label clearly each split and the predicted value within each partition.

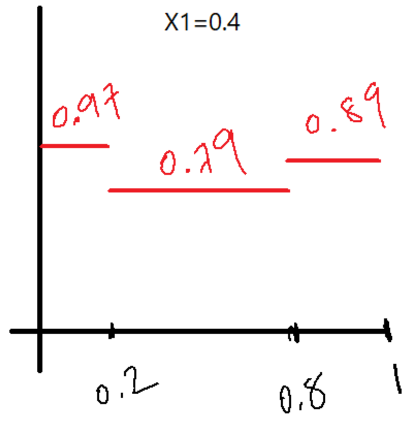
   Answer:



(c) [**8 points**] With fixed $x_1 = 0.2$, draw out the boosted predictor function $\hat{f}$ as a function of only x_2. Do the same thing with fixed $x_1 = 0.4$. How are these two functions different? Similarly, draw the boosted predictor function $\hat{f}$ as a function of only x_1, with x_2 fixed at 0.3 and at 0.9. How are these two functions different?

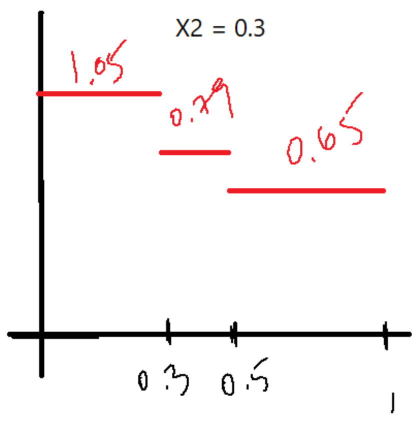
Answer: The bagged predictors with x_1 fixed at 0.2 and 0.4:

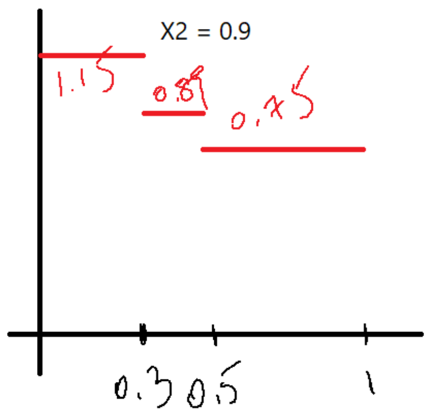




Both predictors are the same modulo a shift of 0.26. In other words, both predictors exhibit exactly the same nonlinear behavior, with a different intercept term.

The bagged predictors with x_2 fixed at 0.3 and 0.9:





Both predictors are the same modulo a shift of 0.10. In other words, both predictors exhibit exactly the same nonlinear behavior, with a different intercept term.

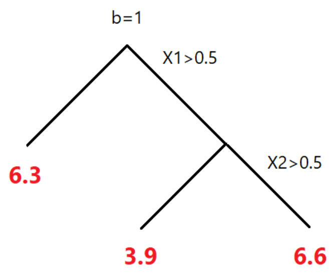
(d) [**5 points**] From your findings in part (c), does the boosted predictor $\hat{f}$ have any interaction effects? What is one property of $\hat{f}$ which allows for interpretability? Explain.

Answer: With an interaction depth of $d = 1$, the boosted predictor $\hat{f}$ does not have any interaction effects, since the effect of a predictor is the same regardless of what the other predictors are. In other words, $\hat{f}$ is *additive*. An additive model provides better interpretability (remember GAMs!), since it allows us to investigate and interpret nonlinear effects for each predictor, independent of other predictors.

(e) [**5 points**] Should more trees be added to the boosted predictor (i.e., should B be increased)? Why? Use parts (a) and (b) to support your answer.

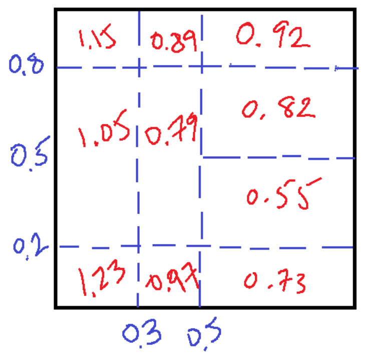
Answer: Yes, B should be increased much more. From part (a), the first tree (which minimizes RSS with $d = 1$ split) has terminal node predictions of 6.3 and 4.9, which suggests that the response variable is on average higher than 4.9. The boosted predictor $\hat{f}$ in part (b), on the other hand, yields predictions which are much lower than 4.9, which results in high test errors. One reason for this is that boosting learns *slowly*; the learning rate of $\lambda = 0.1$ shrinks the effect of each fitted tree, which prevents $\hat{f}$ from overfitting to a single tree and allows for different-shaped trees to attack the residuals. Because of this slow learning, boosting must performed with a large number of trees B to ensure low test errors.

(f) [**5 points**] Now suppose the first tree in part (a) is allowed to have $d = 2$ splits, yielding the following decision tree:



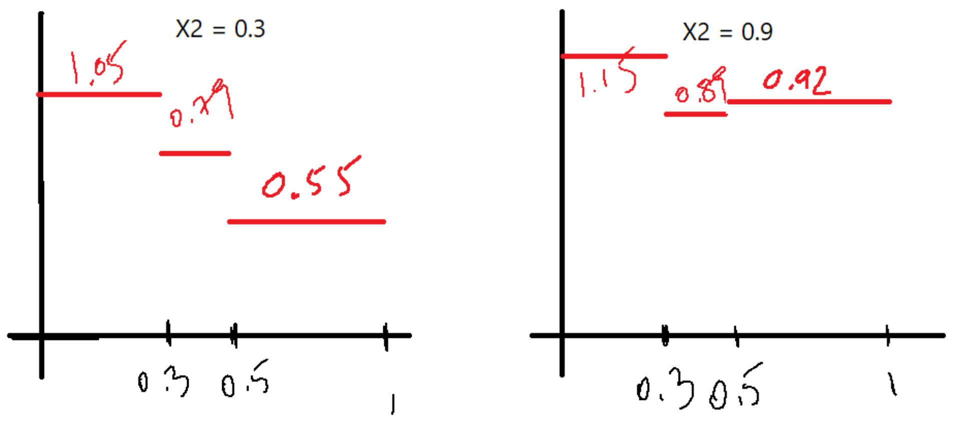
Suppose the next three trees are the same as in part (a). Draw out the boosted predictor $\hat{f}$ as a partition of the prediction space $\mathcal{X}$. Label clearly each split and the predicted value within each partition.

Answer:



(g) [**8 points**] Draw the boosted predictor function $\hat{f}$ as a function of x_1, with x_2 fixed at 0.3 and 0.9. How do these functions compare to those in part (c)? Does the new boosted predictor enjoy the property identified in part (d)? What does this mean for the interpretability of boosted trees with interaction depth $d = 2$? Explain.

Answer: The bagged predictors with x_2 fixed at 0.3 and 0.9:



Unlike the functions in part (c), which exhibit the *same* nonlinear behavior aside from an intercept shift, the two predictors here (for fixed $x_2 = 0.3$ and $x_2 = 0.9$) show *different* nonlinear behavior over x_1. The fitted predictor $\hat{f}$ is therefore not *additive*, since the effect of the first predictor x_1 changes for different values of the second predictor x_2. Because of these interaction effects, boosted trees with interaction depth $d = 2$ are less interpretable than boosted trees with $d = 1$.

2. [**18 points**] State whether each of the following statements are TRUE or FALSE. Briefly justify why in a couple of sentences.

   (a) In complexity cost pruning, the optimal tree with $\alpha = 0$ reduces to the unpruned tree from recursive binary splitting.

   TRUE. With $\alpha = 0$, complexity cost pruning simply minimizes RSS, which is achieved by the unpruned tree from recursive binary splitting.

   (b) Out-of-bag error estimation provides an efficient way to estimate test error of an unpruned decision tree.

   FALSE. OOB error estimation can be used only for bagging and random forests, where each tree is fit on a *bootstrapped* sample of the training data. Since a bootstrap sample is drawn with replacement, there will be some data points (with high probability) which are not in the sample – these points are then used for OOB error estimation. OOB estimation cannot be performed for unpruned decision trees, which are fit using the full training data.

   (c) The misclassification error rate of a classification tree always decreases in the tree-building process.

   FALSE. Classification trees are built using some measure of node purity (e.g., the Gini index or cross-entropy), which may pick splits which do not decrease misclassification error.

   (d) In bagging, the fitted trees are independent of each other, since the bootstrapped datasets are drawn with replacement from the training data.

   FALSE. While the bootstrapping procedure is performed independently, the presence of one (or a few) strong predictor can induce strong correlations between the bagged trees, since most trees will split first on this predictor and yield very similar fits.

   (e) In random forests, a smaller m (the number of candidate predictors for splitting) results in lower bias but higher variance for the tree ensemble.

   FALSE. Random forests with a smaller m would place greater restrictions on the tree-building process, which results in greater bias. However, a smaller m would help decorrelate the fitted trees, which results in lower variance.

   (f) In boosting, a larger shrinkage parameter λ typically requires a larger number of trees B to reduce bias in the boosted ensemble.

   FALSE. Boosting with a *smaller* shrinkage parameter λ typically requires a larger B, since the tree ensemble requires many trees to fit well when learning is slow. With a large λ, boosting with a large B can easily overfit the data (think $\lambda = 1$!).

3. **[22 points]** Let's dig deeper into the bootstrapping idea in bagging, where a "new" dataset is obtained by sampling the training data with replacement. Suppose your training data consists of the data points $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$. A *bootstrapped dataset* $\mathcal{D}^{*b} = \{(\mathbf{x}_i^*, y_i^*)\}_{i=1}^n$ consists of n data points, each independently sampled with replacement from $\mathcal{D}$.

   (a) **[2 points]** What is the probability that the first bootstrapped data point $(\mathbf{x}_1^*, y_1^*)$ equals the first data point $(\mathbf{x}_1, y_1)$?

   Answer: $1/n$.

   (b) **[2 points]** What is the probability that the second bootstrapped data point $(\mathbf{x}_2^*, y_2^*)$ does *not* equal the first data point $(\mathbf{x}_1, y_1)$?

   Answer: $1 - 1/n$.

   (c) **[4 points]** Show that the probability of the first data point $(\mathbf{x}_1, y_1)$ *not* appearing in the bootstrapped dataset $\mathcal{D}^{*b}$ is $(1 - 1/n)^n$. What is the probability of the fifth data point $(\mathbf{x}_5, y_5)$ appearing?

   Answer: Since each data point in the bootstrapped dataset is drawn independently, the event that $(\mathbf{x}_1, y_1)$ *not* appear in $\mathcal{D}^{*b}$ has probability $(1-1/n)^n$. The probability of the fifth data point $(\mathbf{x}_5, y_5)$ appearing in $\mathcal{D}^{*b}$ is $1 - (1 - 1/n)^n$.

   (d) **[5 points]** Let $p_j(n)$ be the probability of the j-th data point $(\mathbf{x}_j, y_j)$ appearing in the bootstrapped dataset $\mathcal{D}^{*b}$. What does $p_j(n)$ converge to as the number of data points grows large, i.e., as $n \to \infty$?

   Answer:

   $$\lim_{n\to\infty} p_j(n) = \lim_{n\to\infty} 1 - (1 - 1/n)^n = 1 - 1/e.$$

   (e) **[4 points]** Let's bring this back to bagging. Recall that each bagged tree is trained using a new bootstrapped dataset $\mathcal{D}^{*b}$, $b = 1, \cdots, B$. Suppose your training data is very large, i.e., $n \to \infty$. Using part (d), what % of the original data points are used (on average) to train a bagged tree? What % of the original data are *not* used (on average)?

   Answer: Around $1 - 1/e \approx 63.2\%$ of the original data points are used to train a bagged tree on average. This leaves around $1/e \approx 36.8\%$ of the original data unused on average.

   (f) **[5 points]** From part (e), it is clear that a bagged tree will not make use of all n data points on average. Are the unused points wasted? If so, explain why. If not, explain how these points may be useful.

   Answer: No, the remaining 36.8% of the data (which is unused in the bagged tree) can be used for out-of-bag error estimation. This gives an efficient way to estimate test error from the fitted tree ensemble, and eliminates the need to perform K-fold cross-validation (which is more computationally expensive, since it requires multiple model fits).