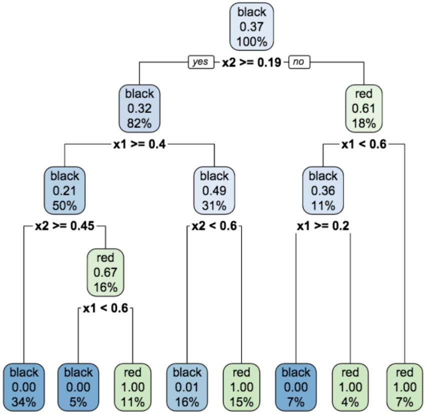
STA 4241 Lecture, Week 13

Overview of what we will cover

- Improving prediction for tree-based models
    - Random forests
    - Boosting
- Variable importance measures
- Examples

Tree review

- Last week we learned about tree based models for predicting Y given X
  - Split predictor space into non-overlapping regions and create predictions in each region

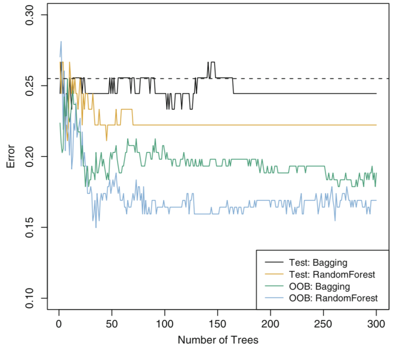


Tree review

- While these models are easy to interpret and relatively flexible, they have some drawbacks
  - Don't estimate simple functions well
  - Don't have good out of sample prediction performance
- We saw that bagging was a method to improve predictions of tree-based models
  - Create new data sets by bootstrapping
  - Fit a regression tree to each bootstrap sample
  - Take average of these trees as your final estimate

Tree review

- We concluded last time with an example that utilized bagging
    - An approach called random forests was doing better than bagging
    - Today we'll see why that is



James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An introduction to statistical learning. New York: springer.

Random forests

- Random forests and bagging are extremely similar approaches
  - Bagging is a special case of random forests!
- Random forests also involves creating B regression trees, one for each bootstrapped data set
- Final estimate is given by

$$\widehat{f}_{rf}(X) = \frac{1}{B} \sum_{b=1}^{B} \widehat{f}^{*b}(X)$$

- Appears identical to bagging

Random forests

- The key difference is how the individual trees are constructed
- At every split of a tree, random forests only consider $m \leq p$ of the predictors to choose from
- Instead of finding the split that reduces RSS (or the Gini index for classification) the most over all possibilities, we find the split using one of the m randomly chosen covariates that reduces RSS the most

Random forests

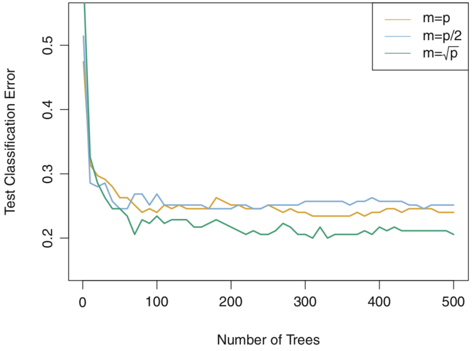
- This seems at first like it shouldn't work well
  - Limiting the available covariates to split on could hurt prediction performance
- It turns out that this de-correlates the individual trees that we are averaging over
  - Remember that averaging works best with lower correlations
  - Smaller variability in the overall predictions
- This can make each individual tree a worse predictor, but improves the average of the B trees

Random forests

- Imagine a setting with one covariate that is very strongly associated with the outcome
    - Other predictors have small to moderate associations
- In bagging, nearly all of your trees will split on this one, strong covariate first
    - All the trees are similar
    - Higher correlation
- Random forests allows some trees to split on other variables first, leading to much different trees

Random forests

- Performance of random forests with different *m* values on an example with 500 covariates
    - A single tree had an error of 45%



James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An introduction to statistical learning. New York: springer.

Random forests

- As with bagging, we don't need to worry about overfitting with large B
  - Larger B is always good
- Generally people choose $m = \sqrt{p}$ or $m = p/3$
  - Small values of m can be useful with many correlated predictors
  - Can use cross-validation to choose this parameter
- Bagging is the special case where $m = p$

Random forests example

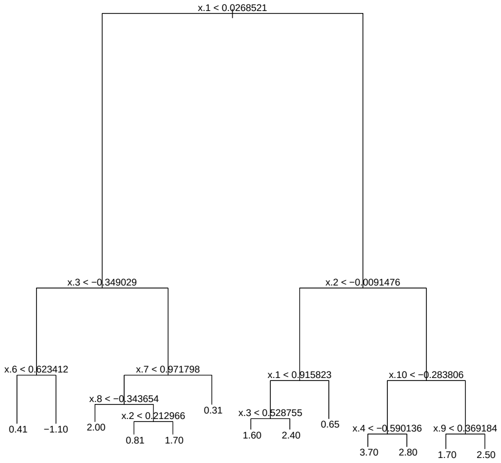
- Let's evaluate performance on a simulated example with 10 covariates
- Suppose our true model is

$$E(Y|\boldsymbol{X}) = 1(X_1 > 0) + 1(X_2 > 0) + 1(X_3 > 0)$$

- We will compare three approaches
  - Single tree that is pruned using cross-validation
  - Bagging
  - Random forests with varying m

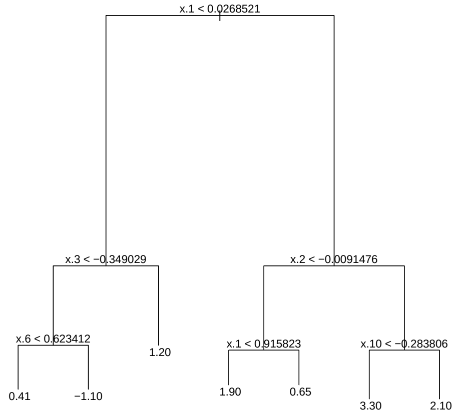
Random forests example

- Below is the full tree before pruning
  - Clearly some unnecessary nodes/splits



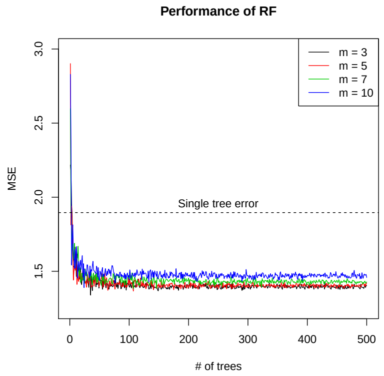
Random forests example

- Here is the tree after pruning
  - Better than before, though still a couple of splits on variables that don't have an association with the outcome



Random forests example

- Here is the performance of random forests as we increase the number of trees and vary m
  - $m = 10$ corresponds to bagging
  - Random forests do slightly better than bagging here



Variable importance measures

- As we discussed, the main drawback of random forests is the lack of interpretability of the model
- Single trees allow us to see which covariates are most important
- For random forests (and bagging) we can construct variable importance measures
  - Positive values indicating how important each covariate is
  - Provides some interpretation of your model fit

Variable importance measures

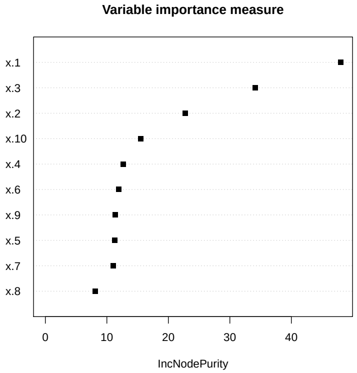
- One way to construct a variable importance measure is by measuring the total amount that the MSE is reduced by splitting on each covariate
- Average the reduction in MSE obtained by splitting on each covariate across all B trees
  - Larger reductions imply more important covariates
- For classification we can use reductions in the Gini index instead of MSE

Variable importance measures

- There are many ways to construct variable importance measures
- One nice approach is to construct a score analogous to out-of-bag error that compares the prediction error on trees with and without a particular covariate
- All variable importance measures have major drawbacks
  - No cutoff that determines a significant predictor
  - Do not control type-I error or false discovery rates
  - Arbitrary scale that is hard to interpret
  - Only provides a measure of relative importance of each predictor

Variable importance measures

- Here is one such variable importance metric applied to the simulated example
  - Correctly identifies that the first 3 covariates are most important



Boosting

- Boosting is another approach to improving prediction performance for regression trees
- Similar to bagging, it is not unique to tree-based models and can be used generally
  - Trees are simply one approach that can be greatly improved with boosting
- We will focus on continuous outcomes for now
  - There is an extension to categorical outcomes, though it is slightly more complicated

Boosting

- Boosting also builds many distinct regression trees, $\widehat{f}^b(X)$
- Bagging and random forests build trees independently of each other
  - On independent bootstrap draws of the data
- Boosting is a sequential algorithm, which builds trees that help capture any remaining signal that hasn't been captured by earlier trees
  - Fit tree to the residuals from the model
  - Repeat a large number of times

Boosting algorithm

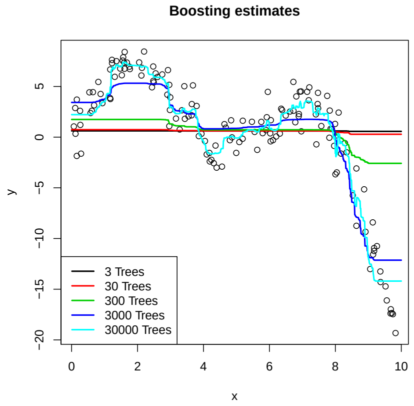
- The algorithm for boosting is described below
  1. Set $\widehat{f}(X) = 0$ and $r_i = Y_i$ for $i = 1, \dots, n$
  2. For $b = 1, \dots, B$, repeat:
     - (a) Fit a tree with d splits to the training data. Here $\boldsymbol{X}$ are your covariates and $\boldsymbol{r}$ is your outcome
     - (b) Update $\widehat{f}$ by adding a shrunken version of this tree
       $$\widehat{f}(X) \leftarrow \widehat{f}(X) + \lambda \widehat{f}^b(X)$$
     - (c) Update the residuals
       $$r_i \leftarrow r_i - \lambda \widehat{f}^b(X)$$
  3. Output the final, boosted model as
     $$\widehat{f}(X) = \sum_{b=1}^{B} \lambda \widehat{f}^b(X)$$

Boosting

- The main idea is to slowly improve $\widehat{f}$ in areas where it does not fit the data well
- Each new tree tries to capture signal that isn't already captured with previous trees
  - Because it looks at the residuals
- Boosting learns slowly and requires a large number of trees, B, to find a well-fitting function
  - Speed of learning depends on λ where $0 < \lambda < 1$
  - Magnitude of λ influences how many trees we need

Boosting

- We can see that boosting slowly adapts to the true function as B grows



Boosting

- Unlike random forests, boosting does not require deep trees to succeed
    - Simple trees work very well for boosting
- Unlike random forests, you can overfit by setting B too large
- Multiple tuning parameters for boosting
    - λ, the speed of learning
    - B the number of trees
    - d, the number of splits in each tree

Boosting example

- Let's try boosting on the additive model from earlier

$$E(Y|\boldsymbol{X}) = 1(X_1 > 0) + 1(X_2 > 0) + 1(X_3 > 0)$$

- Boosting will tend to approximate simple, additive functions such as this one better than random forests
- This is because we can specify d to be small and we get a sum of simple trees

Boosting example

- Here are the testing MSE values for each approach, under a couple of different tuning parameters
- In practice, we would want to run cross-validation to choose these tuning parameters optimally

Approach	Testing MSE
Single Tree	1.90
RF, $m = 3$	1.40
RF, $m = 10$	1.47
Boosting, $\lambda = 0.001$, $B = 5000$, $d = 1$	1.27
Boosting, $\lambda = 0.005$, $B = 2000$, $d = 2$	1.42