CS 451/686-02 Data Mining
Trees — Ensemble Methods

Fall 2016

Maria Daltayanni

part of the slides is credited to the ISL authors

Tree-based Methods

- For regression and classification
- These involve ***stratifying*** or ***segmenting*** the predictor space into a number of simple regions.
- To segment the predictor space, we use a set of **splitting rules**.
- The splitting rules can be summarized in a tree.
- These types of approaches are known as ***decision-tree*** methods.

Pros and Cons

- Tree-based methods are **simple** and useful for **interpretation**.
- However they typically are **not competitive** with the best supervised learning approaches in terms of **prediction accuracy**.
- We consider growing **multiple trees** which are then combined to yield a single consensus prediction.
- These methods are: ***bagging***, ***random forests***, and ***boosting***.
- Combining a large number of trees can often result in dramatic improvements in **prediction accuracy**, at the expense of some loss of interpretation.

Advantages and Disadvantages of Trees

- ▲ Trees are very **easy to explain** to people. In fact, they are even easier to explain than linear regression!
- ▲ Some people believe that decision trees more closely **mirror human decision-making** than do the regression and classification approaches seen in previous chapters.
- ▲ Trees **can be displayed graphically**, and are **easily interpreted** even by a non-expert (especially if they are small).
- ▲ Trees can easily **handle qualitative predictors** without the need to create dummy variables.
- ▼ Unfortunately, trees generally do not have the same level of **predictive accuracy** as some of the other regression and classification approaches.

However, by aggregating many decision trees, the predictive performance of trees can be substantially improved. We introduce these concepts next.

Bagging

- ***Bootstrap aggregation***, or ***bagging***, is a general-purpose procedure for **reducing the variance** of a statistical learning method.

- We introduce it here because it is particularly useful and frequently used in the context of decision trees.

- Given a set of *n* independent observations $Z_1, \ldots, Z_n$, each with variance σ^2, for the mean of the observations, $\bar{Z}$, the variance is σ^2/n.

- In other words, ***averaging a set of observations reduces variance***.

- Of course, this is not practical because we generally do not have access to multiple training sets.

Bagging– continued

- Instead, we can bootstrap, by taking repeated samples from the (single) training data set.
- In this approach we generate B different bootstrapped training data sets.
- We then train our method on the b-th bootstrapped training set in order to get $\hat{f}^{*b}(x)$, the prediction at a point x.
- We then average all the predictions to obtain:

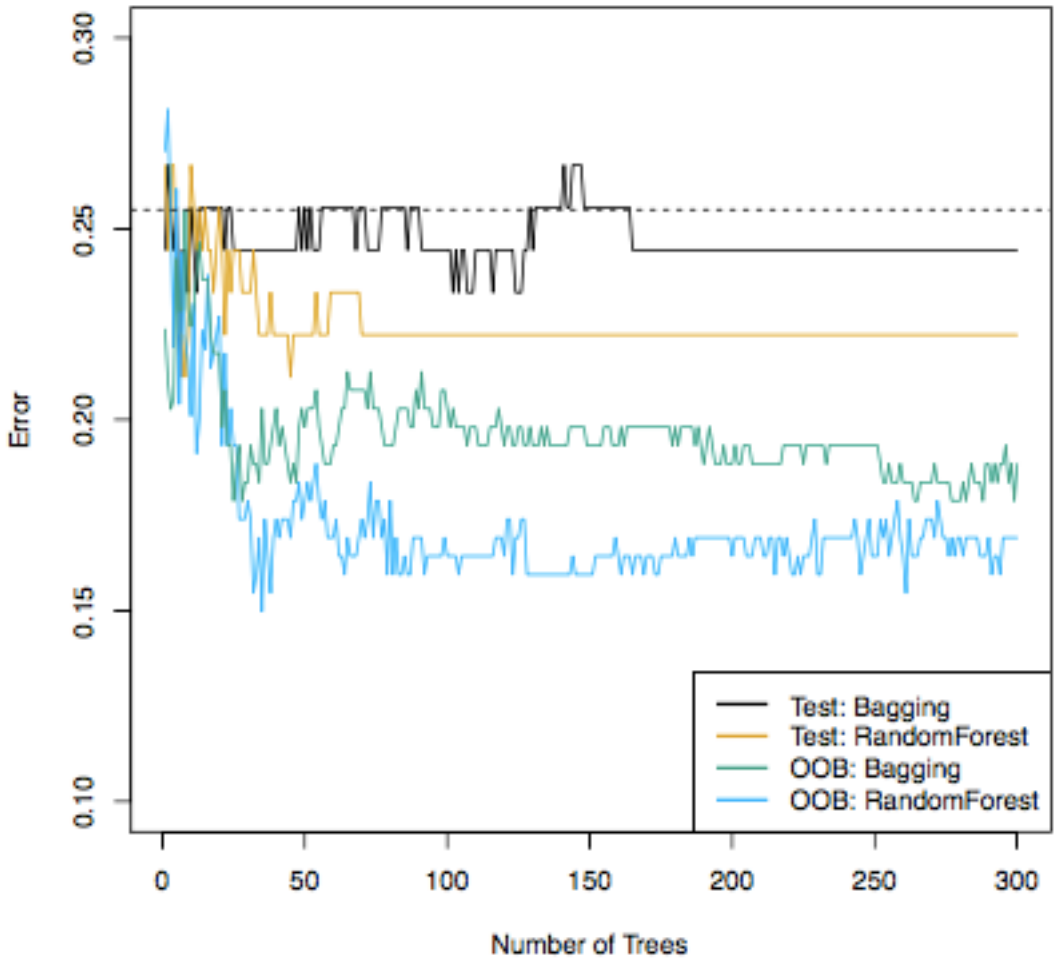
$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x).$$

- This is called ***bagging***.

Bagging classification trees

- The above prescription is applied to regression trees.
- For classification trees:
  - for each test observation,
  - we record the class predicted by each of the B trees,
  - and take a ***majority vote***:
    - the overall prediction is the most commonly occurring class among the B predictions.

Bagging the heart data



Details of previous figure

Bagging and random forest results for the Heart data.

- The test error (black and orange) is shown as a function of B, the number of bootstrapped training sets used.
- Random forests were applied with $m = \sqrt{p}$.
- The dashed line indicates the test error resulting from a single classification tree.
- The green and blue traces show the OOB error, which in this case is considerably lower

Out-of-Bag Error Estimation

- A very straightforward way to estimate the test error of a bagged model:

- The key to bagging is that trees are repeatedly fit to bootstrapped subsets of the observations.
- On average, each bagged tree makes use of around 2/3 of the observations.

- The remaining 1/3 of the observations not used to fit a given bagged tree are referred to as the ***out-of-bag*** (OOB) observations.
- We can predict the response for the i-th observation using each of the trees in which that observation was OOB.

- This will yield around $B/3$ predictions for the i-th observation.
- **We average the predictions.**
- It is essentially the LOO cross-validation error for bagging, if B is large.

Random Forests

- ***Random forests*** provide an improvement over bagged trees by way of a small tweak that ***decorrelates*** the trees.
- This reduces the variance when we average the trees.

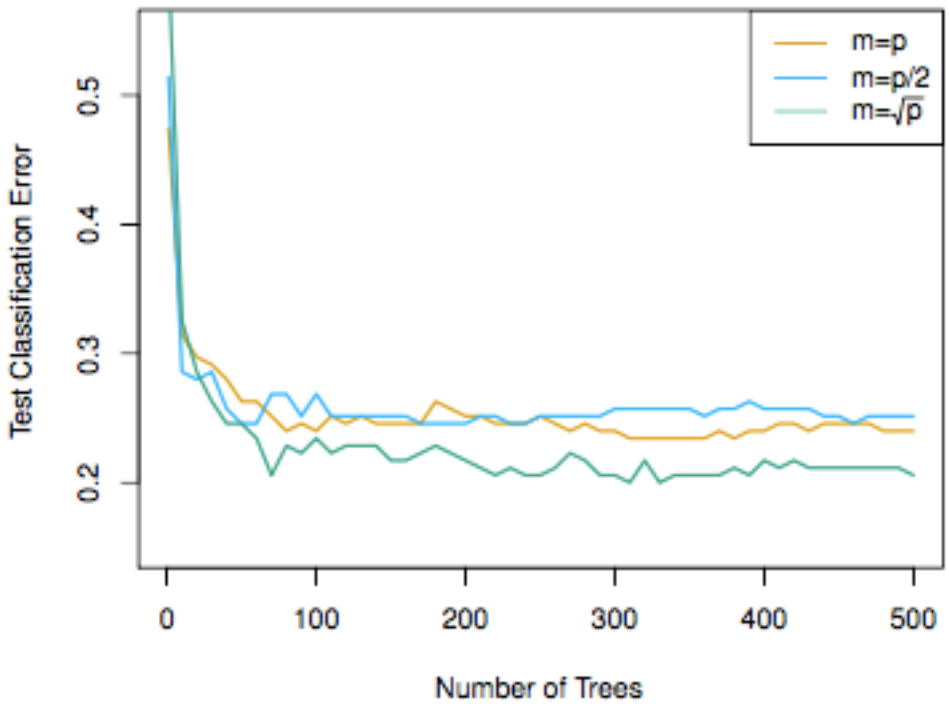
- As in bagging, we build a number of decision trees on bootstrapped training samples.
- But each time a split in a tree is considered, ***a random selection of m predictors*** is chosen as split candidates from the full set of *p* predictors.
- The split is allowed to use only one of those *m* predictors.

- A fresh selection of *m* predictors is taken at each split
- Typically we choose $m \approx \sqrt{p}$
- — that is, the number of predictors considered at each split is approximately equal to the square root of the total number of predictors (4 out of the 13 for the Heart data).

Example: gene expression data

- We applied random forests to a high-dimensional biological data set consisting of expression measurements of 4,718 genes measured on tissue samples from 349 patients.
- There are around 20,000 genes in humans, and individual genes have different levels of activity, or expression, in particular cells, tissues, and biological conditions.
- Each of the patient samples has a qualitative label with 15 different levels: either normal or one of 14 different types of cancer.
- We use random forests to predict cancer type based on the 500 genes that have the largest variance in the training set.
- We randomly divided the observations into a training and a test set, and applied random forests to the training set for three different values of the number of splitting variables m.

Results: gene expression data



Details of previous figure

- Results from random forests for the fifteen-class gene expression data set with $p = 500$ predictors.
- The test error is displayed as a function of the number of trees. Each colored line corresponds to a different value of m, the number of predictors available for splitting at each interior tree node.
- Random forests ($m < p$) lead to a slight improvement over bagging ($m = p$). A single classification tree has an error rate of 45.7%.

Boosting

- Like bagging, boosting is a general approach that can be applied to many statistical learning methods for regression or classification. We only discuss boosting for decision trees.

- Recall that bagging involves creating multiple copies of the original training data set using the bootstrap, fitting a separate decision tree to each copy, and then combining all of the trees in order to create a single predictive model.

- Notably, each tree is built on a bootstrap data set, independent of the other trees.

- Boosting works in a similar way, except that the trees are grown ***sequentially***: each tree is grown using information from previously grown trees.

Boosting algorithm for regression trees

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.
2. For $b = 1, 2, \ldots, B$, repeat:
   - 2.1 Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r).
   - 2.2 Update $\hat{f}$ by adding in a shrunken version of the new tree:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x).$$

   - 2.3 Update the residuals,

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i).$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x).$$

What is the idea behind this procedure?

- Unlike fitting a single large decision tree to the data, which amounts to ***fitting the data hard*** and potentially overfitting, the boosting approach instead ***learns slowly***.

- Given the current model, we fit a decision tree to the residuals from the model. We then add this new decision tree into the fitted function in order to update the residuals.

- Each of these trees can be rather small, with just a few terminal nodes, determined by the parameter d in the algorithm.

- By fitting small trees to the residuals, we slowly improve $\hat{f}$ in areas where it does not perform well. The shrinkage parameter λ slows the process down even further, allowing more and different shaped trees to attack the residuals.

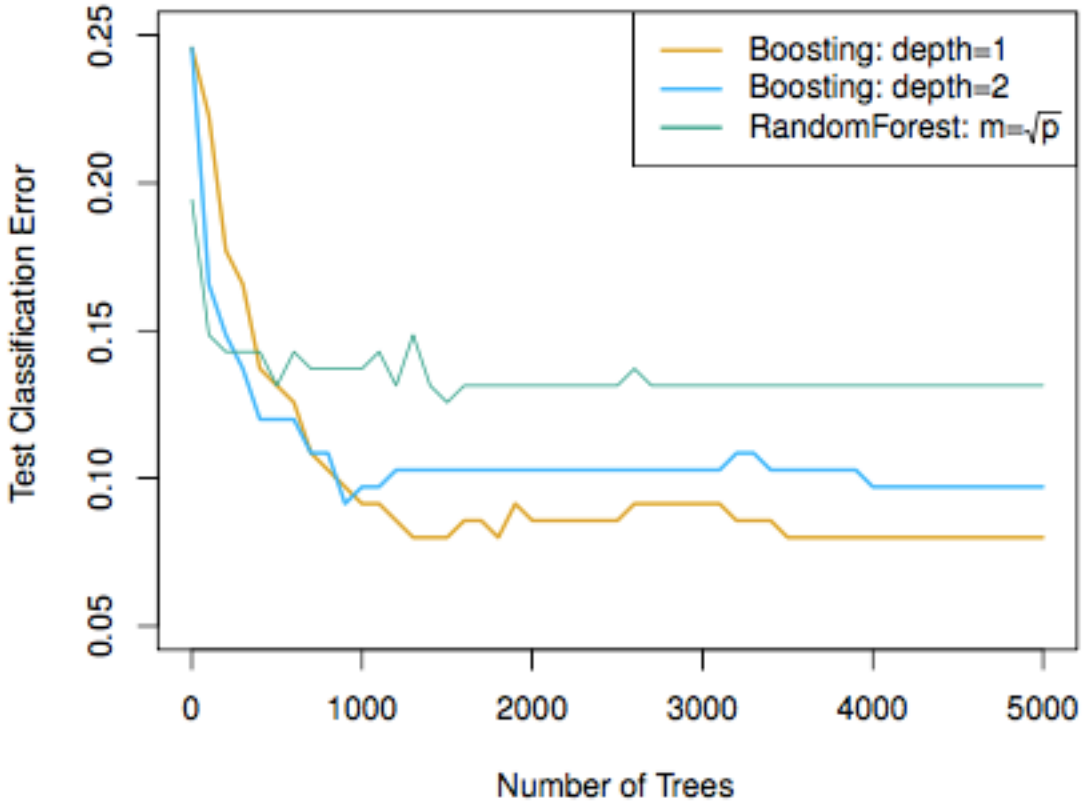
Boosting for classification

- Boosting for classification is similar in spirit to boosting for regression, but is a bit more complex. We will not go into detail here, nor do we in the text book.

- Students can learn about the details in *Elements of Statistical Learning, chapter 10.*

- The R package gbm (gradient boosted models) handles a variety of regression and classification problems.

Gene expression data continued



Details of previous figure

- Results from performing boosting and random forests on the fifteen-class gene expression data set in order to predict *cancer* versus *normal*.

- The test error is displayed as a function of the number of trees. For the two boosted models, $\lambda = 0.01$. Depth-1 trees slightly outperform depth-2 trees, and both outperform the random forest, although the standard errors are around 0.02, making none of these differences significant.

- The test error rate for a single tree is 24%.

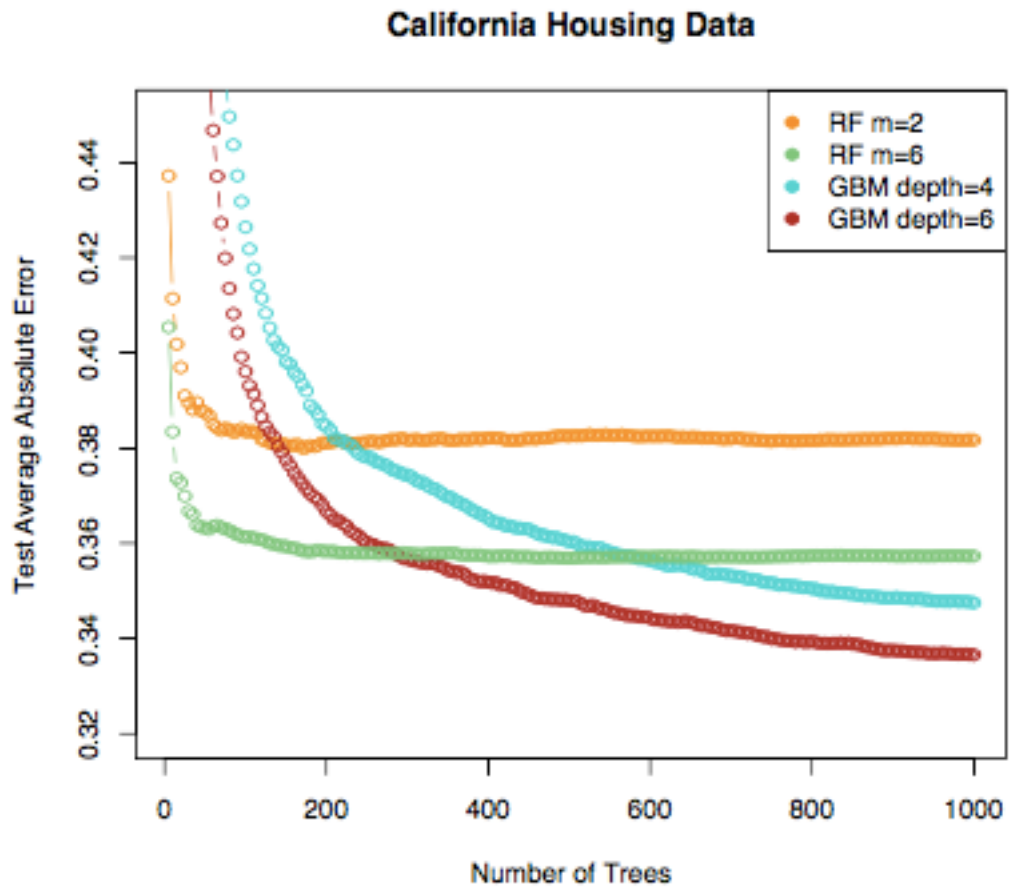
Tuning parameters for boosting

1. The ***number of trees*** B. Unlike bagging and random forests, boosting can overfit if B is too large, although this overfitting tends to occur slowly if at all. We use cross-validation to select B.

2. The ***shrinkage parameter*** λ, a small positive number. This controls the rate at which boosting learns. Typical values are 0.01 or 0.001, and the right choice can depend on the problem. Very small λ can require using a very large value of B in order to achieve good performance.

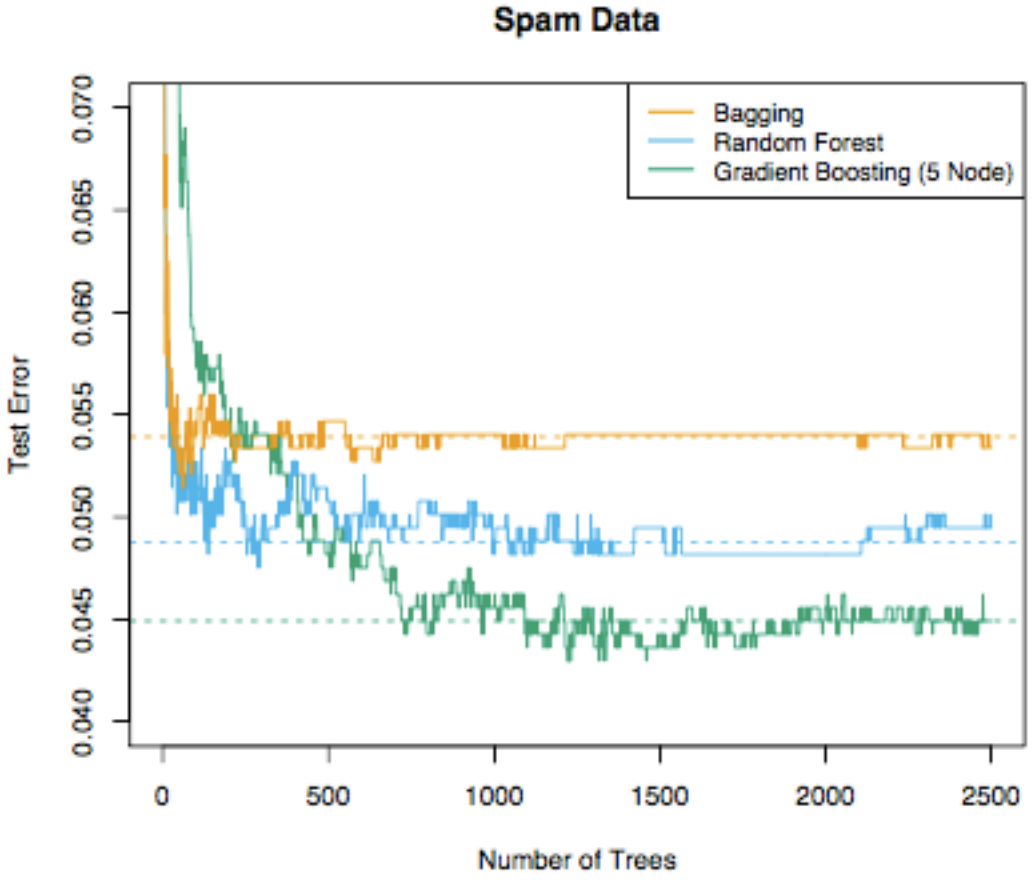
3. The ***number of splits*** d in each tree, which controls the complexity of the boosted ensemble. Often $d = 1$ works well, in which case each tree is a ***stump***, consisting of a single split and resulting in an additive model. More generally d is the ***interaction depth***, and controls the interaction order of the boosted model, since d splits can involve at most d variables.

Another regression example



from *Elements of Statistical Learning, chapter 15.*

Another classification example

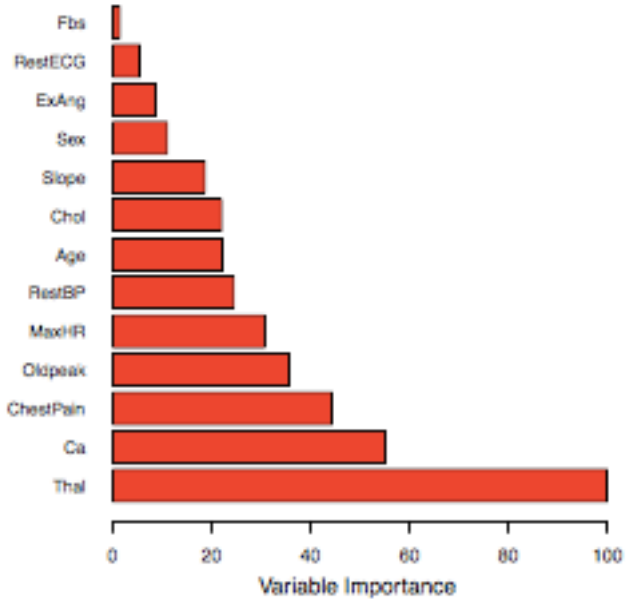


from *Elements of Statistical Learning, chapter 15.*

Variable importance measure

- For bagged/RF regression trees, we record the total amount that the RSS is decreased due to splits over a given predictor, averaged over all B trees. A large value indicates an important predictor.

- Similarly, for bagged/RF classification trees, we add up the total amount that the Gini index is decreased by splits over a given predictor, averaged over all B trees.



Variable importance plot for the Heart data

Summary

- Decision trees are simple and interpretable models for regression and classification

- However they are often not competitive with other methods in terms of prediction accuracy

- Bagging, random forests and boosting are good methods for improving the prediction accuracy of trees. They work by growing many trees on the training data and then combining the predictions of the resulting ensemble of trees.

- The latter two methods— random forests and boosting— are among the state-of-the-art methods for supervised learning. However their results can be difficult to interpret.