INDR 450/550

Spring 2022

Lecture 18: Trees/forests, prescriptive analytics

April 20, 2022

Fikri Karaesmen

Announcements

- Class Exercise at the end of lecture today. If you are participating online, please upload your document under Course Contents/Class Exercises
- Lab 7 material (on lasso and ridge) and a short video are available
- Exam on May 7.
  - Review exercises are available
  - Make sure that you also review the class exercises and the homeworks

- The first seven labs were uploaded. Please follow them.

Predictive Analytics

- Remaining topics (to complete at the latest the week after the spring break)
  - Validation ✓
  - Model selection / regularization ✓
  - Non-linear regressions, ✓
  - Generalized additive models, tree based-methods ✓
  - Bagging, boosting and random forests (today)

Tree-based Methods

- Tree-based methods segment the predictor space into multiple regions in a smart way. They then make a very simple prediction for each region using the training data.
- The prediction for a given region is simply the average of the observations in the training data that fall into that region.
- Conceptually simple and easy to explain.
- But require additional work to be competitive with earlier methods (ridge, lasso, GAMs etc.)

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods

Predicting the salary of a baseball player using his experience (years) and Performance (# of hits).

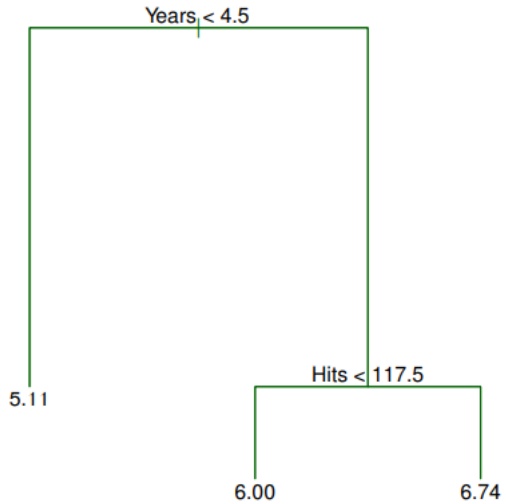

FIGURE 8.1. *For the* `Hitters` *data, a regression tree for predicting the log salary of a baseball player, based on the number of years that he has played in the major leagues and the number of hits that he made in the previous year. At a given internal node, the label (of the form* $X_j < t_k$*) indicates the left-hand branch emanating from that split, and the right-hand branch corresponds to* $X_j \geq t_k$*. For instance, the split at the top of the tree results in two large branches. The left-hand branch corresponds to* `Years<4.5`*, and the right-hand branch corresponds to* `Years>=4.5`*. The tree has two internal nodes and three terminal nodes, or leaves. The number in each leaf is the mean of the response for the observations that fall there.*

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods

- Here's a summary of the basic approach.
  1. We divide the predictor space (the space of all possible values of the predictors $(X_1, X_2, ..., X_p)$ into J distinct (non-overlapping regions) R_1, R_2,..,R_j.
  2. For any observation that falls in region R_j, we use the same prediction: the average of all observations from that region in the training set.
  3. Note that this is natural for binary variables. If it's a weekend day, we make a prediction based on past weekend days and if it's a week day, we make a prediction based on week days.
- We are making a prediction for $\mathbf{x}_p$ that falls in region R_j. The mean of all training observations in $R_j = 200$, then $\hat{y}(\mathbf{x}_p) = 500$

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: overfitting

- Typically, the tree tends to become very large (deep). This leads to decreased errors in the training data but also to overfitting.
- We use pruning to reduce the size of the tree (the number of end nodes (leaves)).
- The pruning trade-offs estimation errors against the size of the tree by using a penalty function.

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: advantages and disadvantages

- Decision trees for regression are easy to communicate and understand
- Natural for qualitative predictors
- But do not have the same level of predictive accuracy as the previous models
- There are some robustness issues. A small change in data can lead to a big change in the resulting regression tree.
- The variance of the typical tree-based regression estimator is high (much higher than a linear regression based estimator).

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Bagging

- Reducing the variance of the tree-based regression estimator is a critical issue
- Bagging (Bootstrap Aggregation) is an adaptation of the bootstrap approach.
- In bootstrapping we use multiple training sets by sampling randomly from a single training set.
- Recall that averaging over multiple estimators reduces the variance of the prediction.
- We, therefore, pick B samples from the training set to generate B training sets.

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Bagging (Bootstrapping)

In this example, we have a few data points. We use bootstrapping to generate multiple training sets by randomly sampling from the initial set.

This leads to estimators that have a lower variance.

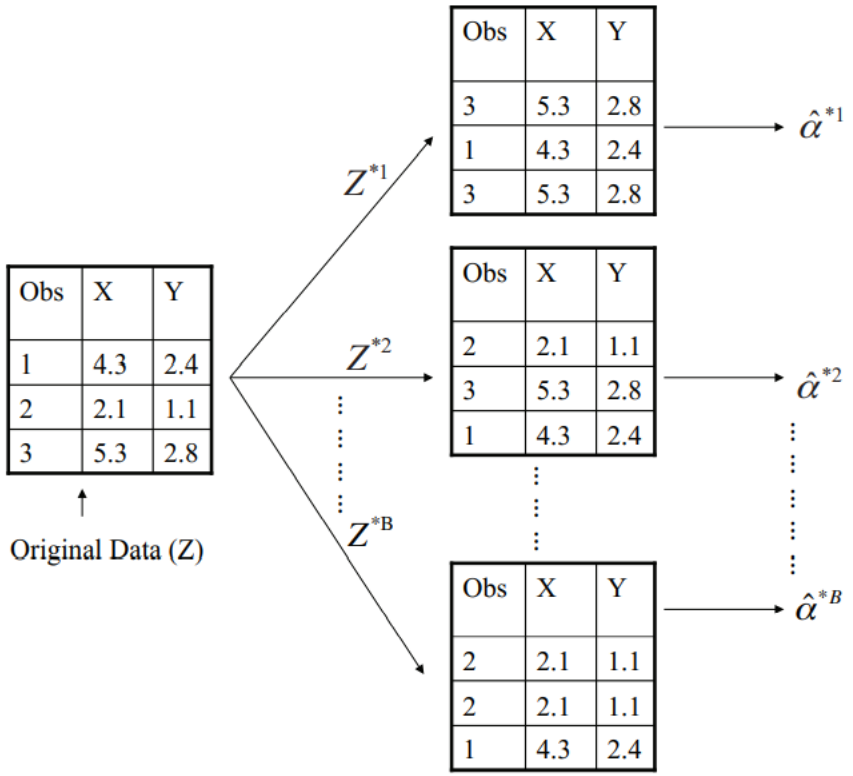

FIGURE 5.11. *A graphical illustration of the bootstrap approach on a small sample containing $n = 3$ observations. Each bootstrap data set contains n observations, sampled with replacement from the original data set. Each bootstrap data set is used to obtain an estimate of α.*

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Bagging

- We fit regression trees to each one of the B different training sets and obtain B different estimators $\hat{f}^b(x)$.
- Our final estimator is a an average:

$$f_{bag}(x) = \frac{\sum_{b=1}^{B} \hat{f}^b(x)}{B}$$

- Averaging reduces the variance of the estimator. This is always useful but especially so for tree-based approaches.
- While bagging is a smart idea, most of the B different trees that are constructed are similar. This causes the individual estimators to be correlated.
- We would get a better variance reduction if we could use estimators that were less correlated.

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Random Forests

- Random forests build on the bagging idea but implement it with averaging over estimators that are weakly correlated.
- As in bagging, we build multiple trees but for each tree, we impose the additional constraint that only m out of the p predictors are used for each tree (in each split). These m predictors are selected randomly.
- It is suggested that $m \approx \sqrt{p}$.
- In each split, we freshly select a new sample of m predictors.
- Note that when $m = p$, we are back to standard bagging.

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Random Forests

- It sounds like a bad idea to be limiting the number of predictors chosen at each split but it helps to obtain non-similar trees and leads to lower correlation of estimators between the trees than standard bagging.
- Because the trees are expected to be not very similar to each other. We have a random forest :-).
- Impressive results are obtained through random forest implementations of decision trees.

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Random Forests

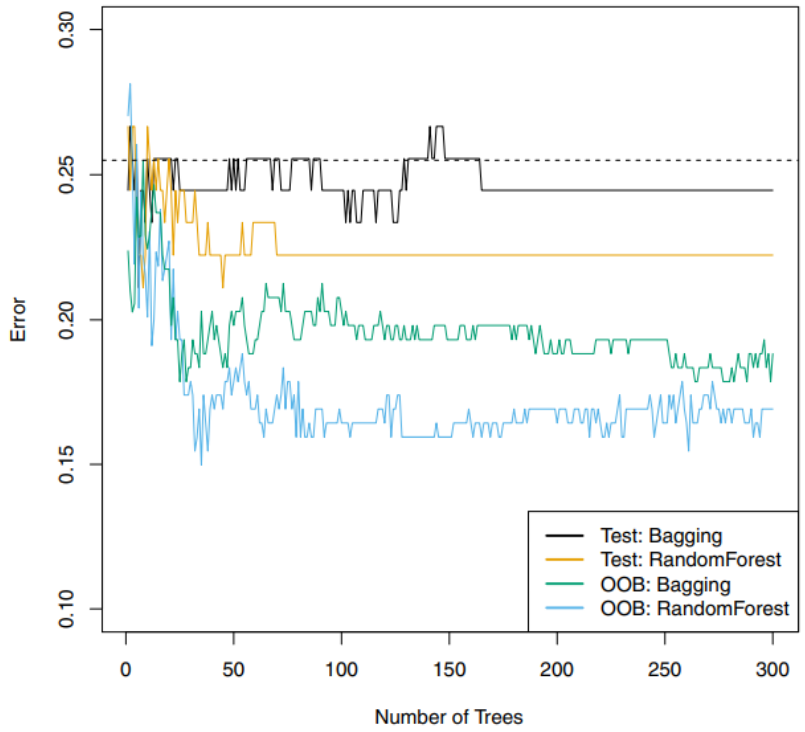

FIGURE 8.8. *Bagging and random forest results for the* `Heart` *data. The test error (black and orange) is shown as a function of* B*, the number of bootstrapped training sets used. Random forests were applied with* $m = \sqrt{p}$*. The dashed line indicates the test error resulting from a single classification tree. The green and blue traces show the OOB error, which in this case is — by chance — considerably lower.*

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Boosting

- We had mentioned boosting, a greedy approach that fits one predictor at a time and continues with greedily fitting the other predictors one at a time using the remaining residuals.
- In the context of regression trees, boosting greedily and sequentially fits one tree at a time considering the updated residuals at each stage.
- This is in contrast to bagging or random forests which fit many trees in parallel.

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Boosting

- This is conceptually very simple. But some effort is required to avoid overfitting:
  1. We have to restrict B, the number of trees to fit sequentially.
  2. We use a shrinkage parameter λ to control the rate of learning. There is a trade-off between λ and B (a small λ requires a large B).
  3. We have to control the number of splits d for each tree.

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Boosting

Algorithm 8.2 *Boosting 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:

   (a) Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r).

   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:

   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \tag{8.10}$$

   (c) Update the residuals,

   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \tag{8.11}$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x). \tag{8.12}$$

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Tree-based Methods: Boosting

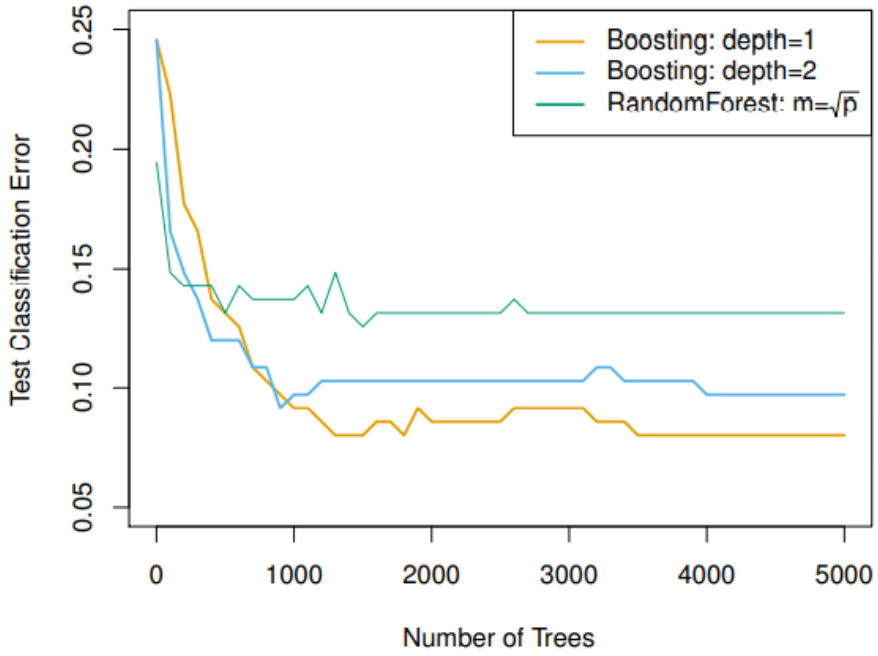

FIGURE 8.11. *Results from performing boosting and random forests on the 15-class gene expression data set in order to predict* cancer *versus* normal. *The test error is displayed as a function of the number of trees. For the two boosted models,* $\lambda = 0.01$. *Depth-1 trees slightly outperform 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 %.*

From *An Introduction to Statistical Learning*, James, Witten, Hastie, Thibshirani

Predictive Analytics: brief overview

- Simple estimators of time series:
  - moving averages, exponential smoothing,
  - estimating functional forms (trends, non-linear trends, seasonality)
  - Transformations (log, differencing)
- ARIMA models:
  - To capture the effects of auto-correlation
  - Relatively few parameters but involved approach for estimation
  - Can be combined with other methods

Predictive Analytics: brief overview

- Classification:
  - Similar in terms of predictors that can be used
  - Logistic regression
  - Focus on classification errors as criterion
  - The class assignment rule involves a trade-off
- Model reduction/shrinkage
  - Lasso and ridge regression
  - Trade-off the number of predictors against the error performance
  - Crucial with big data

Predictive Analytics: brief overview

- Time series regressions:
  - Can build regression models using only the series itself
  - Functional forms (t, t2, etc.)
  - Binary (dummy) variables to mark calendar events
- General regressions for time series:
  - Can also add other predictors in addition to calendar information
  - The approach stays the same but the number of predictors may become large
  - Overfitting is an issue

Predictive Analytics: brief overview

- Non-linear regressions, General Additive Models
  - Standard basis expansions allow for ‘automated generality’
  - Cubic splines (involving knots) are especially common
  - Allows capturing non-linearity
- Regression Trees
  - No model assumptions (linearity, non-linearity etc.)
  - Conceptually simple
  - Usually require additional strengthening for better performance (bagging, random forests, boosting)

Predictive Analytics: brief overview

- Computational power has enabled the more recent approaches
  - Large regressions
  - Ridge, lasso, PCR etc.
  - GAMs
  - Tree-based methods
  - Integer Optimization
- Overfitting is a real problem and validation should be an essential part of any project.

Prescriptive Analytics:

- Using predictions to solve optimization problems
- We will focus on operational problems
- Most problems have the following structure:

$$\min_{\mathbf{z}} E[c(\mathbf{Y}, \mathbf{z})]$$

where $\mathbf{z}$ is a decision variable and $\mathbf{Y}$ is a random variable. In addition, there could be constraints on the decision variable (i.e. $\mathbf{z} \in \mathcal{Z}$).

Prescriptive Analytics:

- Consider:

$$\min_{\mathbf{z}} E[c(\mathbf{Y}, \mathbf{z})]$$

- We focused in the first part of the course on estimating $E[\mathbf{Y}]$.
- Life would be easy if the following were true.

$$E[c(\mathbf{Y}, \mathbf{z})] = c(E[\mathbf{Y}], \mathbf{z})$$

- This would lead to a deterministic optimization problem:

$$\min_{\mathbf{z}} c(E[\mathbf{Y}], \mathbf{z})$$

Prescriptive Analytics:

- But in general:

$$E[c(\mathbf{Y}, \mathbf{z})] \neq c(E[\mathbf{Y}], \mathbf{z})$$

- And $c(\mathbf{Y}, \mathbf{z})$ is itself a random variable. This leads to more challenging and interesting optimization problems where we have to take into account the probability distribution of the random variable $\mathbf{Y}$.

Prescriptive Analytics:

- This requires obtaining more than point estimates for the predictions because we need the complete probability distribution. We have seen that sometimes the prediction errors can be easily characterized (i.e. normally distributed with an estimated standard deviation).
- For other implementations (i.e. KNN, tree-based regressions etc.), we'll try to find creative ways of obtaining estimators for the probability distribution of the prediction.