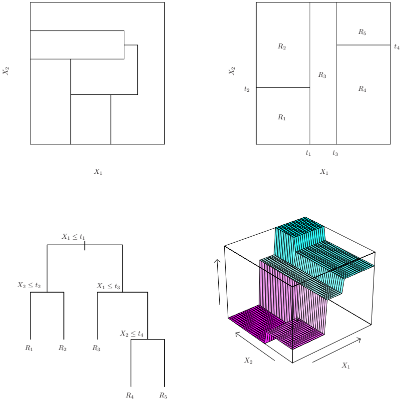
Lecture 8: Decision trees

Reading: Section 8.1

STATS 202: Data mining and analysis

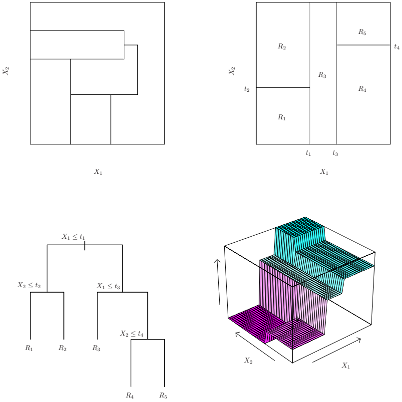
Rajan Patel

Decision trees, 10,000 foot view



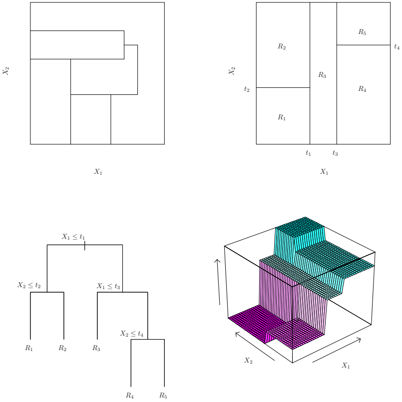
1. Find a partition of the space of predictors.
2. Predict a constant in each set of the partition.

Decision trees, 10,000 foot view



1. Find a partition of the space of predictors.
2. Predict a constant in each set of the partition.
3. The partition is defined by splitting the range of one predictor at a time.

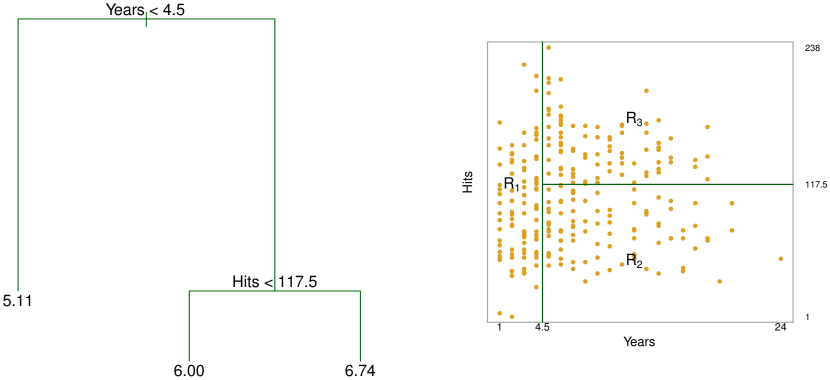
Decision trees, 10,000 foot view



1. Find a partition of the space of predictors.
2. Predict a constant in each set of the partition.
3. The partition is defined by splitting the range of one predictor at a time.

   $\rightarrow$ Not all partitions are possible.

Example: Predicting a baseball player's salary



The prediction for a point in R_i is the average of the training points in R_i.

How is a decision tree built?

- Start with a single region R_1, and iterate:

  1. Select a region R_k, a predictor X_j, and a splitting point s, such that splitting R_k with the criterion $X_j < s$ produces the largest decrease in RSS:

  $$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2$$

  2. Redefine the regions with this additional split.

How is a decision tree built?

- Start with a single region R_1, and iterate:

  1. Select a region R_k, a predictor X_j, and a splitting point s, such that splitting R_k with the criterion $X_j < s$ produces the largest decrease in RSS:

  $$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2$$

  2. Redefine the regions with this additional split.

- Terminate when there are 5 observations or fewer in each region.

How is a decision tree built?

- Start with a single region R_1, and iterate:

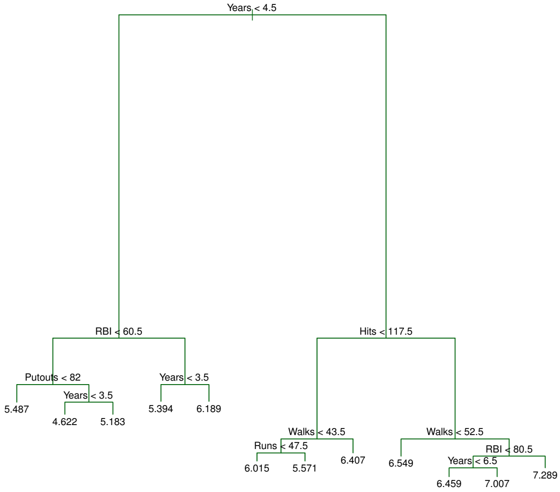
  1. Select a region R_k, a predictor X_j, and a splitting point s, such that splitting R_k with the criterion $X_j < s$ produces the largest decrease in RSS:

  $$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2$$

  2. Redefine the regions with this additional split.

- Terminate when there are 5 observations or fewer in each region.
- This grows the tree from the root towards the leaves.

How is a decision tree built?



How do we control overfitting?

- **Idea 1:** Find the optimal subtree by cross validation.

How do we control overfitting?

- **Idea 1:** Find the optimal subtree by cross validation.

  $\rightarrow$ There are too many possibilities, so we would still over fit.

How do we control overfitting?

- **Idea 1:** Find the optimal subtree by cross validation.

  $\rightarrow$ There are too many possibilities, so we would still over fit.

- **Idea 2:** Stop growing the tree when the RSS doesn't drop by more than a threshold with any new cut.

How do we control overfitting?

- **Idea 1:** Find the optimal subtree by cross validation.

  $\rightarrow$ There are too many possibilities, so we would still over fit.

- **Idea 2:** Stop growing the tree when the RSS doesn't drop by more than a threshold with any new cut.

  $\rightarrow$ In our greedy algorithm, it is possible to find good cuts after bad ones.

How do we control overfitting?

Solution: Prune a large tree from the leaves to the root.

- **Weakest link pruning:**

How do we control overfitting?

Solution: Prune a large tree from the leaves to the root.

- **Weakest link pruning:**
  - Starting with T_0, substitute a subtree with a leaf to obtain T_1, by minimizing:
$$\frac{RSS(T_1) - RSS(T_0)}{|T_0| - |T_1|}.$$

How do we control overfitting?

Solution: Prune a large tree from the leaves to the root.

- **Weakest link pruning:**
  - Starting with T_0, substitute a subtree with a leaf to obtain T_1, by minimizing:
    $$\frac{RSS(T_1) - RSS(T_0)}{|T_0| - |T_1|}.$$
  - Iterate this pruning to obtain a sequence $T_0, T_1, T_2, \ldots, T_m$ where T_m is the null tree.

How do we control overfitting?

Solution: Prune a large tree from the leaves to the root.

- **Weakest link pruning:**
  - Starting with T_0, substitute a subtree with a leaf to obtain T_1, by minimizing:
  $$\frac{RSS(T_1) - RSS(T_0)}{|T_0| - |T_1|}.$$
  - Iterate this pruning to obtain a sequence $T_0, T_1, T_2, \ldots, T_m$ where T_m is the null tree.
  - Select the optimal tree T_i by cross validation.

How do we control overfitting?

... or an equivalent procedure

- **Cost complexity pruning:**

How do we control overfitting?

... or an equivalent procedure

- **Cost complexity pruning:**
  - Solve the problem:

$$\text{minimize} \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2 + \alpha |T|.$$

How do we control overfitting?

... or an equivalent procedure

- **Cost complexity pruning:**
  - Solve the problem:

$$\text{minimize} \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2 + \alpha |T|.$$

How do we control overfitting?

... or an equivalent procedure

- **Cost complexity pruning:**
  - Solve the problem:

$$\text{minimize} \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2 + \alpha |T|.$$

  - When $\alpha = \infty$, we select the null tree.

How do we control overfitting?

... or an equivalent procedure

- **Cost complexity pruning:**
  - Solve the problem:

$$\text{minimize} \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2 + \alpha |T|.$$

  - When $\alpha = \infty$, we select the null tree.
  - When $\alpha = 0$, we select the full tree.

How do we control overfitting?

… or an equivalent procedure

- **Cost complexity pruning:**
  - Solve the problem:

$$\text{minimize} \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2 + \alpha |T|.$$

  - When $\alpha = \infty$, we select the null tree.
  - When $\alpha = 0$, we select the full tree.
  - The solution for each α is among $T_1, T_2, \dots, T_m$ from weakest link pruning.

How do we control overfitting?

... or an equivalent procedure

- **Cost complexity pruning:**
  - Solve the problem:

$$\text{minimize} \sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2 + \alpha |T|.$$

  - When $\alpha = \infty$, we select the null tree.
  - When $\alpha = 0$, we select the full tree.
  - The solution for each α is among $T_1, T_2, \ldots, T_m$ from weakest link pruning.
  - Choose the optimal α (the optimal T_i) by cross validation.

Cross validation

1. Construct a sequence of trees $T_0, \ldots, T_m$ for a range of values of α.

Cross validation

1. Construct a sequence of trees $T_0, \ldots, T_m$ for a range of values of α.
2. Split the training points into 10 folds.

Cross validation

1. Construct a sequence of trees $T_0, \ldots, T_m$ for a range of values of α.
2. Split the training points into 10 folds.
3. For $k = 1, \ldots, 10$,
   - For each tree T_i, use every fold except the kth to estimate the averages in each region.
   - For each tree T_i, calculate the RSS in the test fold.

Cross validation

1. Construct a sequence of trees $T_0, \ldots, T_m$ for a range of values of α.
2. Split the training points into 10 folds.
3. For $k = 1, \ldots, 10$,
   - For each tree T_i, use every fold except the kth to estimate the averages in each region.
   - For each tree T_i, calculate the RSS in the test fold.
4. For each tree T_i, average the 10 test errors, and select the value of α that minimizes the error.

Cross validation

1. Construct a sequence of trees $T_0, \ldots, T_m$ for a range of values of α.
2. Split the training points into 10 folds.
3. For $k = 1, \ldots, 10$,
   - For each tree T_i, use every fold except the kth to estimate the averages in each region.
   - For each tree T_i, calculate the RSS in the test fold.
4. For each tree T_i, average the 10 test errors, and select the value of α that minimizes the error.

WRONG WAY TO DO CROSS VALIDATION!

Cross validation, the right way

1. Split the training points into 10 folds.

Cross validation, the right way

1. Split the training points into 10 folds.
2. For $k = 1, \ldots, 10$, using every fold except the kth:
   - Construct a sequence of trees $T_1, \ldots, T_m$ for a range of values of α, and find the prediction for each region in each one.
   - For each tree T_i, calculate the RSS on the test set.

Cross validation, the right way

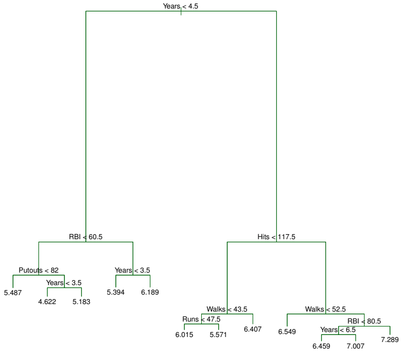
1. Split the training points into 10 folds.
2. For $k = 1, \ldots, 10$, using every fold except the kth:
   - Construct a sequence of trees $T_1, \ldots, T_m$ for a range of values of α, and find the prediction for each region in each one.
   - For each tree T_i, calculate the RSS on the test set.
3. Select the parameter α that minimizes the average test error.

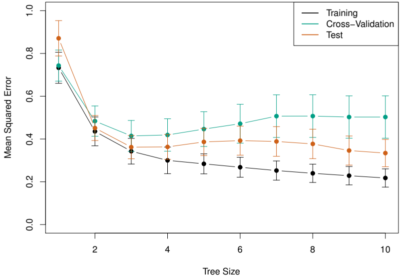
Cross validation, the right way

1. Split the training points into 10 folds.
2. For $k = 1, \ldots, 10$, using every fold except the kth:
   - Construct a sequence of trees $T_1, \ldots, T_m$ for a range of values of α, and find the prediction for each region in each one.
   - For each tree T_i, calculate the RSS on the test set.
3. Select the parameter α that minimizes the average test error.

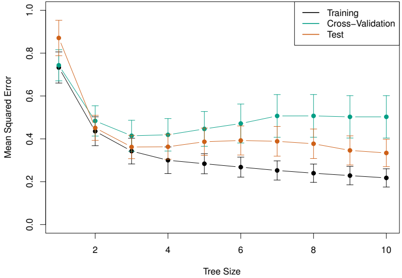
Note: We are doing all fitting, **including the construction of the trees**, using only the training data.

Example. Predicting baseball salaries





Example. Predicting baseball salaries



Classification trees

- They work much like regression trees.

Classification trees

- They work much like regression trees.
- We predict the response by **majority vote**, i.e. pick the most common class in every region.

Classification trees

- They work much like regression trees.
- We predict the response by **majority vote**, i.e. pick the most common class in every region.
- Instead of trying to minimize the RSS:

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \bar{y}_{R_m})^2$$

we minimize a classification loss function.

Classification losses

- The 0-1 loss or misclassification rate:

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} \mathbf{1}(y_i \neq \hat{y}_{R_m})$$

- The Gini index:

$$\sum_{m=1}^{|T|} q_m \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

  where $\hat{p}_{m,k}$ is the proportion of class k within R_m, and q_m is the proportion of samples in R_m.

- The cross-entropy:

$$-\sum_{m=1}^{|T|} q_m \sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk}).$$

Classification losses

- The Gini index and cross-entropy are better measures of the purity of a region, i.e. they are low when the region is mostly one category.

Classification losses

- The Gini index and cross-entropy are better measures of the purity of a region, i.e. they are low when the region is mostly one category.
- **Motivation for the Gini index:**

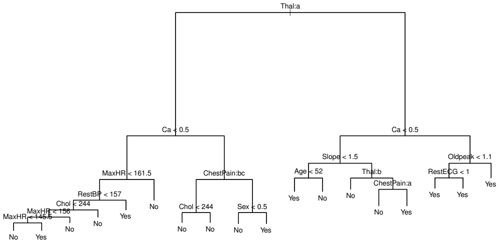
  If instead of predicting the most likely class, we predict a random sample from the distribution $(\hat{p}_{1,m}, \hat{p}_{2,m}, \ldots, \hat{p}_{K,m})$, the Gini index is the expected misclassification rate.

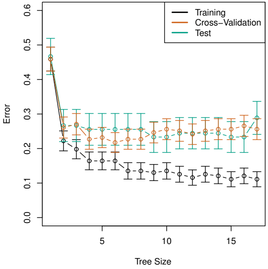
Classification losses

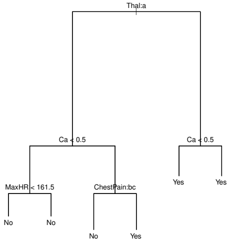
- The Gini index and cross-entropy are better measures of the purity of a region, i.e. they are low when the region is mostly one category.
- **Motivation for the Gini index:**

  If instead of predicting the most likely class, we predict a random sample from the distribution $(\hat{p}_{1,m}, \hat{p}_{2,m}, \ldots, \hat{p}_{K,m})$, the Gini index is the expected misclassification rate.
- It is typical to use the Gini index or cross-entropy for growing the tree, while using the misclassification rate when pruning the tree.

Example. Heart dataset.







Some advantages of decision trees

- Very easy to interpret!

Some advantages of decision trees

- Very easy to interpret!
- Closer to human decision-making.

Some advantages of decision trees

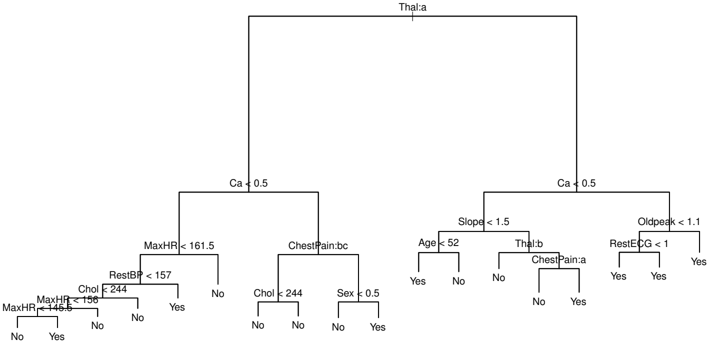
- Very easy to interpret!
- Closer to human decision-making.
- Easy to visualize graphically.

Some advantages of decision trees

- Very easy to interpret!
- Closer to human decision-making.
- Easy to visualize graphically.
- They easily handle qualitative predictors and missing data.

Example. Heart dataset.

How do we deal with categorical predictors?



Categorical predictors

- If there are only 2 categories, then the split is obvious. We don't have to choose the splitting point s, as for a numerical variable.
- If there are more than 2 categories:
  - Order the categories according to the average of the response:

    $$\texttt{ChestPain} : \texttt{a} > \texttt{ChestPain} : \texttt{c} > \texttt{ChestPain} : \texttt{b}$$

  - Treat as a numerical variable with this ordering, and choose a splitting point s.
- This is the optimal way of partitioning.

Missing data

Problem: If a sample is missing variable X_j, and a tree contains a split according to $X_j > s$, then we may not be able to assign the sample to a region.

Solution:

- When choosing a new split with variable X_j (growing the tree):
  - Only consider the samples which have the variable X_j.
  - In addition to choosing the best split, choose a second best split using a different variable, and a third best, ...
- To propagate a sample down the tree, if it is missing a variable to make a decision, try the second best decision, or the third best, ...

Bagging

- Bagging = Bootstrap Aggregating
- In the Bootstrap, we replicate our dataset by sampling with replacement:
  - Original dataset: $\mathtt{x} = \mathtt{c}(\mathtt{x1}, \mathtt{x2}, \ldots, \mathtt{x100})$
  - Bootstrap samples:
    $\mathtt{boot1} = \mathtt{sample}(\mathtt{x}, 100, \mathtt{replace} = \mathtt{True})$, ...,
    $\mathtt{bootB} = \mathtt{sample}(\mathtt{x}, 100, \mathtt{replace} = \mathtt{True})$.
- We used these samples to approximate the Standard Error of a parameter estimate:

$$\mathsf{SE}(\hat{\beta}_1) \approx \mathsf{SD}(\hat{\beta}_1^{(1)}, \ldots, \hat{\beta}_1^{(B)})$$

Bagging

- In **Bagging** we average the predictions of a model fit to many Bootstrap samples.

  Example. Bagging the Lasso

  - Let $\hat{y}^{L,b}$ be the prediction of the Lasso applied to the bth bootstrap sample.
  - Bagging prediction:

$$\hat{y}^{\text{boot}} = \frac{1}{B} \sum_{b=1}^{B} \hat{y}^{L,b}.$$

When does Bagging make sense?

When a regression method or a classifier has a tendency to overfit, Bagging reduces the variance of the prediction.

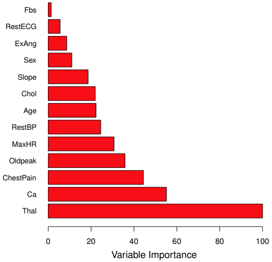
- When n is large, the empirical distribution is similar to the true distribution of the samples.
- Bootstrap samples are like independent realizations of the data.
- Bagging amounts to averaging the fits from many independent datasets, which would reduce the variance by a factor $1/B$.

Bagging decision trees

- **Disadvantage:** Every time we fit a decision tree to a Bootstrap sample, we get a different tree T^b.

  $\rightarrow$ Loss of interpretability

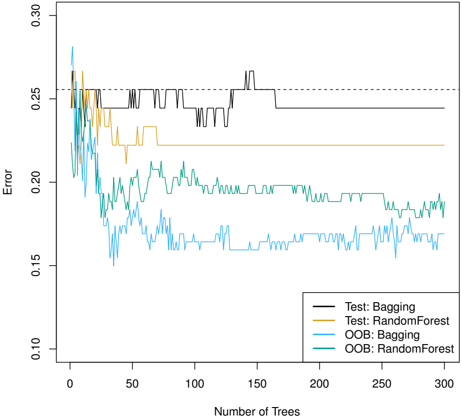
- For each predictor, add up the total amount by which the RSS (or Gini index) decreases every time we use the predictor in T^b.
- Average this total over each Boostrap estimate $T^1, \ldots, T^B$.



Out-of-bag (OOB) error

- To estimate the test error of a bagging estimate, we could use cross-validation.
- Each time we draw a Bootstrap sample, we only use 63% of the observations.
- **Idea:** use the rest of the observations as a test set.
- **OOB error:**
  - For each sample x_i, find the prediction $\hat{y}_i^b$ for all bootstrap samples b which do not contain x_i. There should be around $0.37B$ of them. Average these predictions to obtain $\hat{y}_i^{\text{oob}}$.
  - Compute the error $(y_i - \hat{y}_i^{\text{oob}})^2$.
  - Average the errors over all observations $i = 1, \ldots, n$.

Out-of-bag (OOB) error



The test error decreases as we increase B
(dashed line is the error for a plain decision tree).

Random Forests

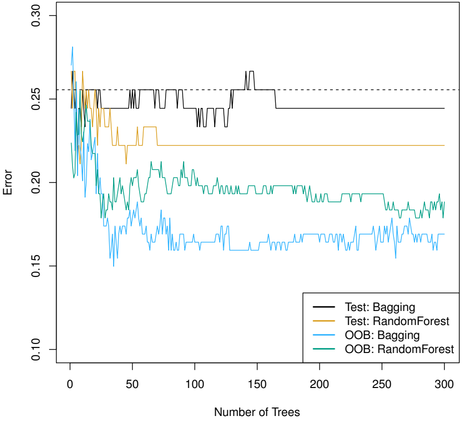
Bagging has a problem:

→ The trees produced by different Bootstrap samples can be very similar.

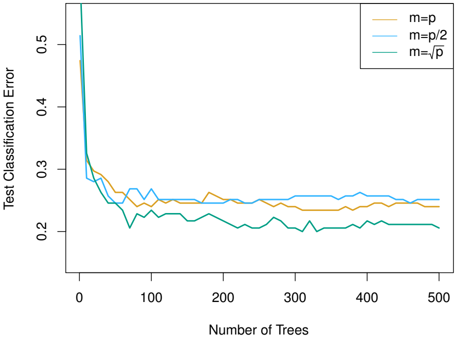
Random Forests:

- We fit a decision tree to different Bootstrap samples.
- When growing the tree, we select a random sample of $m < p$ predictors to consider in each step.
- This will lead to very different (or "uncorrelated") trees from each sample.
- Finally, average the prediction of each tree.

Random Forests vs. Bagging



Random Forests, choosing m



The optimal m is usually around $\sqrt{p}$,
but this can be used as a tuning parameter.