STA 4241 Lecture, Week 12

Overview of what we will cover

- Regression trees
  - What is a tree based model?
  - How to estimate trees
  - Pros/cons of trees
- Improving prediction accuracy
  - Averaging to improve predictions
  - Bagging

Trees

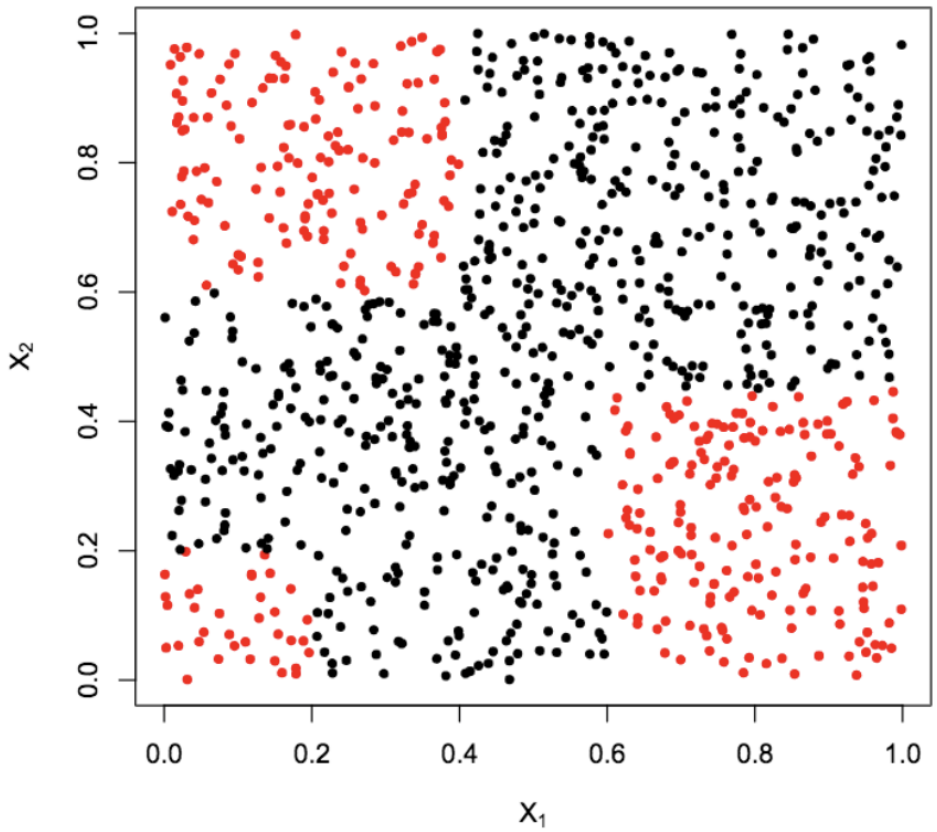
- A tree-based method splits the covariate space into segments or regions
- Prediction within a region is given by the mean or mode of the data points that fall in that region
- The splitting rules that define the regions can be summarized in a tree
  - Decision tree

Trees

- Decision trees are highly interpretable
  - Simple and easy to explain
- Nonparametric approach that does not rely on correct specification of the functional form of relationships between variables
- Unfortunately, the increased interpretation comes at a cost of predictive accuracy
  - Extensions to trees can improve their predictive performance substantially

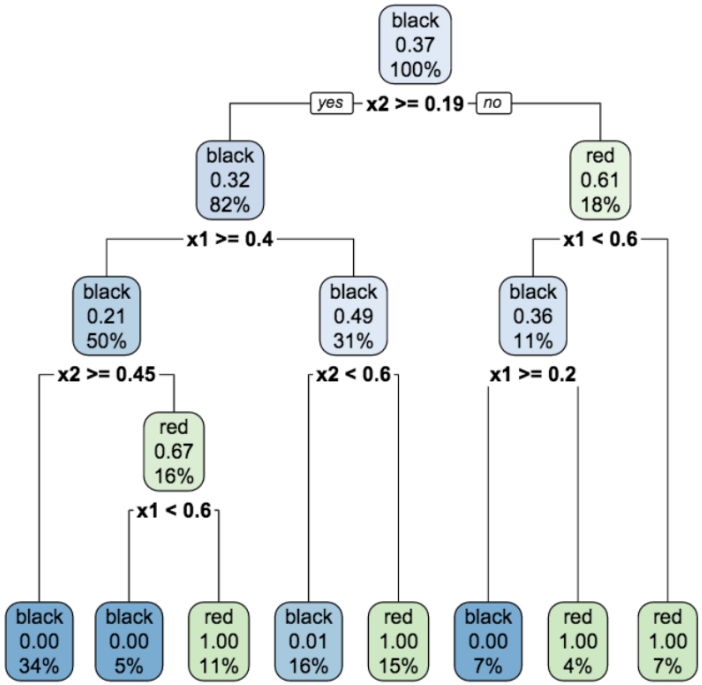
Trees

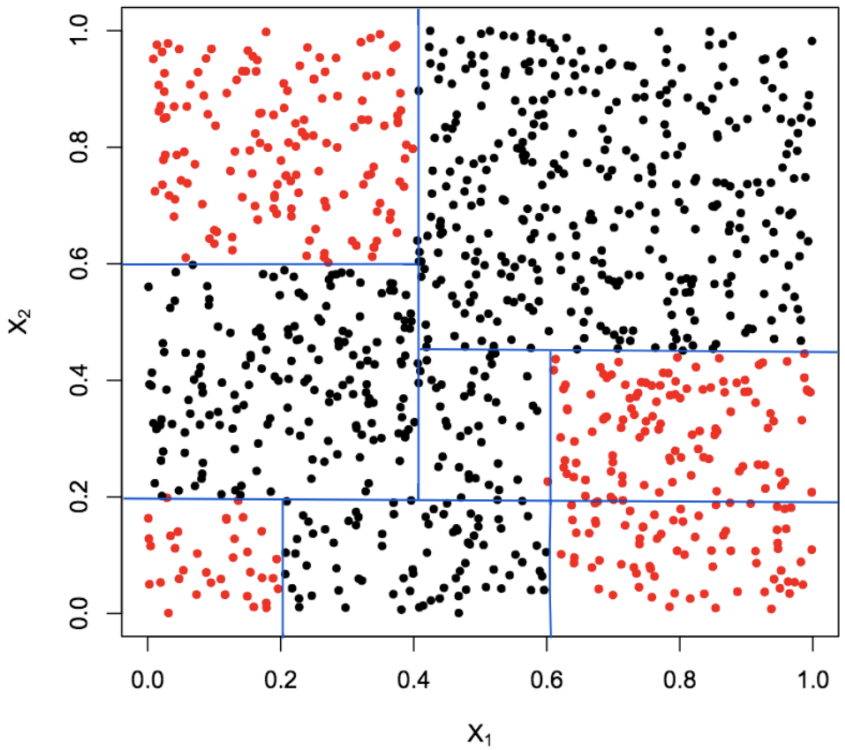
- Suppose we have two covariates and a binary outcome that is either red or black
- We can accurately classify between red and black if we split the space appropriately



Trees

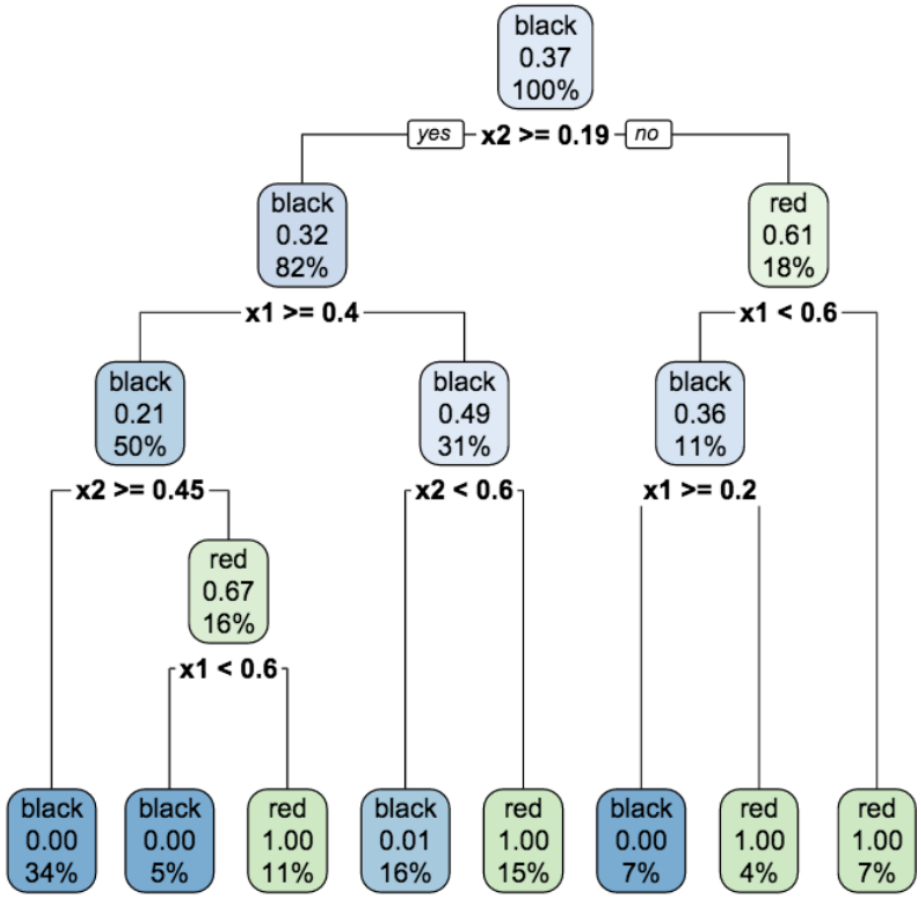
- Here are the resulting segments and corresponding tree fit to this data





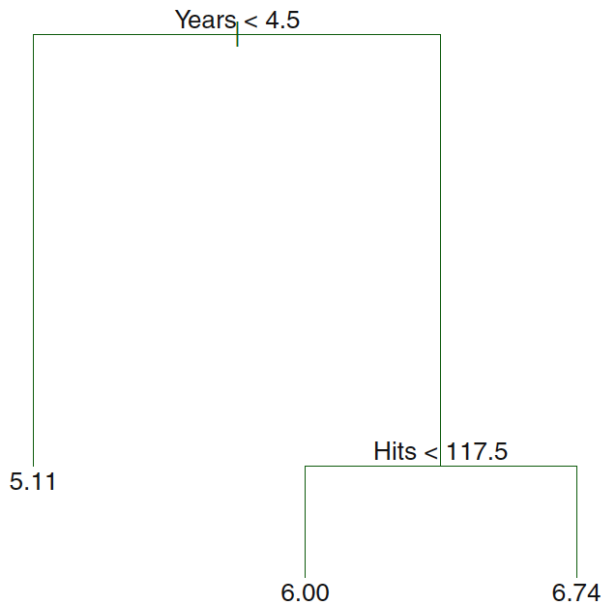
Trees

- Tree structure is very easy to explain to a non-statistician
- If you give me a point in the covariate space, say (x_1, x_2), I can move down the tree until I get to the end
  - Final boxes are known as terminal nodes or leaves
  - Classify (x_1, x_2) based on the data in the terminal mode



Trees

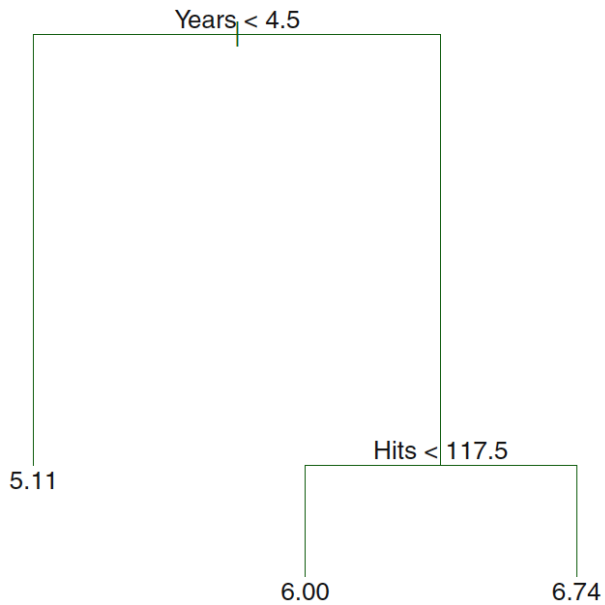
- The book provides a simple example trying to predict the salary of a baseball player
  - Covariates are years in the league and the number of hits in the previous year
  - Tree model splits on whether or not the player has been in the league for over 4.5 years and then if they have, it splits on whether they had over 117.5 hits



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

Trees

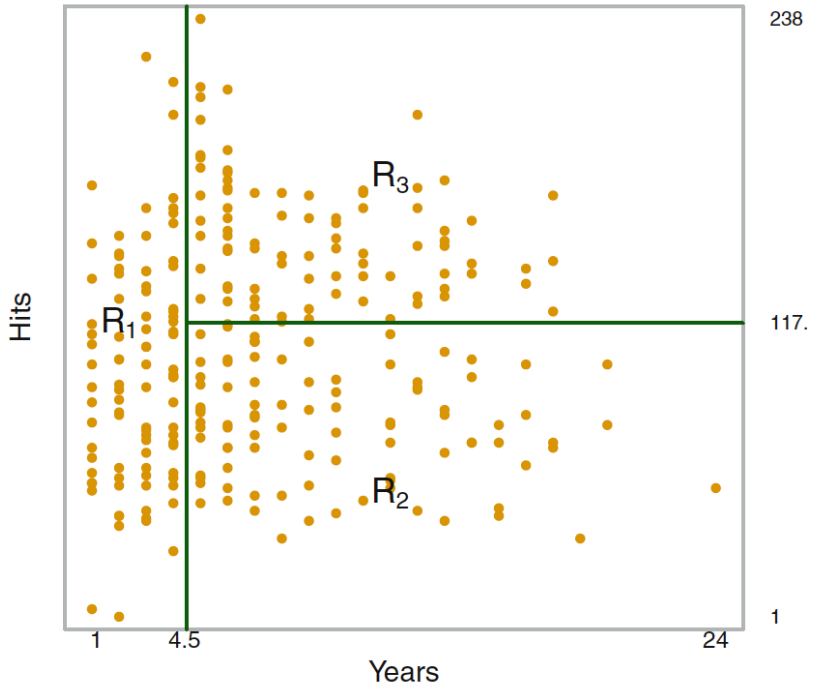
- If you've been in the league less than 4.5 years you're predicted to have a log salary of 5.11
- If you've been in the league over 4.5 years and have over 117.5 hits, then your log salary is estimated to be 6.74



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

Trees

- Here is an illustration of the segments from the previous tree
- Data points that fall within any region R_j are used to make predictions for future points in that region



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

Trees

- The previous tree has a very nice interpretation
- The most important variable affecting salary is the number of years in the league
- If a player has been in the league less than 4.5 years then the number of hits does not affect their salary
- Number of hits does impact salary for players who have been in the league longer than 4.5 years

Trees

- How do we find and estimate trees?
- Trees break the predictor space into J distinct and non-overlapping regions
  - Denote by R_j for $j = 1, \ldots, J$
- Every observation falls into one of these regions
- Predictions in these regions are taken to be the average value of the data points in the region

Trees

- Making predictions in a particular region is not difficult
- The difficult part is deciding how to break the predictor space into regions
- Very large number of ways to define regions
  - Computationally infeasible to examine all such ways
- Trees break the space into high-dimensional rectangles
  - Increases simplicity and interpretability

Trees

- We will define our prediction to be

$$\widehat{Y}_i = \widehat{f}(X_i)$$

- Trees build this function in the following manner

$$f(X) = \sum_{j=1}^{J} c_j 1(X \in R_j)$$

- If our outcome is continuous then $c_j \in \mathbb{R}$, otherwise c_j is a category label

Trees

- Now we need to find the regions defined by R_j
- As with before, we aim to reduce the RSS defined now by

$$\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2 = \sum_{j=1}^{J}\sum_{i \in R_j}(Y_i - \widehat{Y}_{R_j})^2$$

  where $\widehat{Y}_{R_j}$ is the mean of the training observations that fall in R_j
- Too computationally intensive to consider all possible partitions that define J regions

Trees

- We will use a greedy algorithm to speed up computations and find the regions R_j
  - Greedy because we optimize decisions at each step instead of looking ahead and trying to find the overall best solution
  - Saw this previously with the forward stepwise selection algorithm
- The name of the algorithm we use is recursive binary splitting
- Top-down algorithm that starts at the top of the tree and repeatedly makes splits that improve RSS

Trees

- Define $\{X|X_j < s\}$ to be the region of the predictor space in which X_j takes a smaller value than s
- Our first split is to find the covariate j and location s such that if we split the predictors into regions given by $\{X|X_j < s\}$ and $\{X|X_j \geq s\}$ the reduction in RSS is maximized
- To simplify notation we will use the following

$$R_1(j,s) = \{X|X_j < s\} \quad R_2(j,s) = \{X|X_j \geq s\}$$

- It is also important to define predictions in these regions as

$$\widehat{Y}_{R_1} = \frac{\sum_{i=1}^n 1(X_i \in R_1(j,s))Y_i}{\sum_{i=1}^n 1(X_i \in R_1(j,s))} \quad \widehat{Y}_{R_2} = \frac{\sum_{i=1}^n 1(X_i \in R_2(j,s))Y_i}{\sum_{i=1}^n 1(X_i \in R_2(j,s))}$$

Trees

- Now we find the values of j and s that minimize

$$\sum_{i:X_i \in R_1(j,s)} (Y_i - \widehat{Y}_{R_1})^2 + \sum_{i:X_i \in R_2(j,s)} (Y_i - \widehat{Y}_{R_2})^2$$

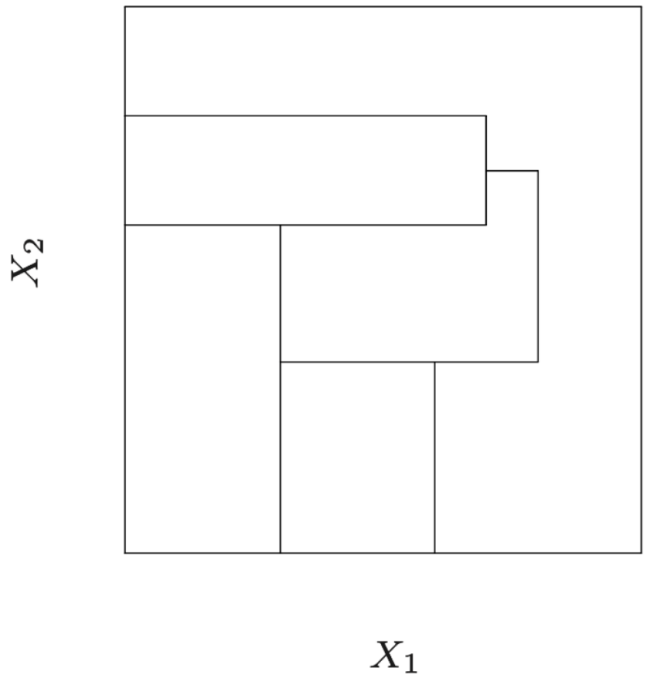
- Finding j and s that minimize this expression can be done relatively quickly
- Even though there are infinitely many possibilities for $s \in \mathbb{R}$, we only need to look at $n - 1$ distinct points
  - Points that change which subjects are in which group

Trees

- Now that we have split the predictor space into two regions, we repeat the process
- We don't split the entire predictor space
  - Only consider splitting one of the two regions defined by the previous step
- Then we will have three regions and we will consider splits within each of these three regions
- Process repeated until some stopping criteria is met
  - Each region has a small number of individuals in it

Trees

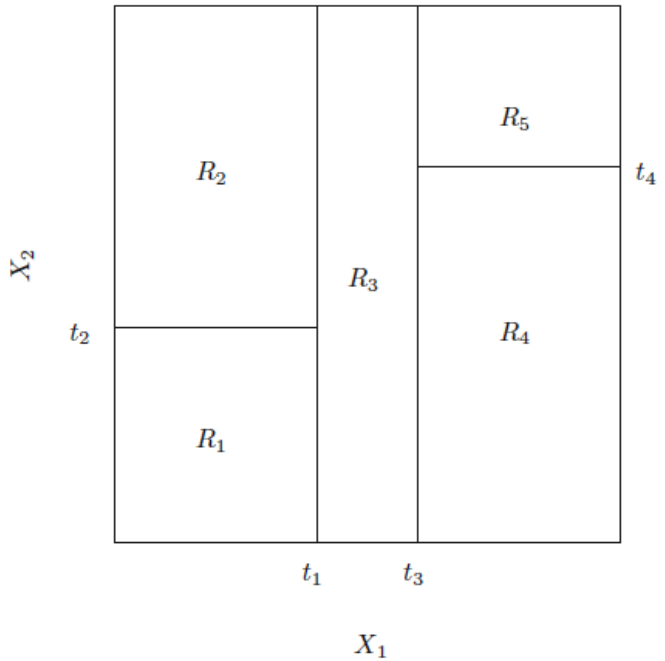
- Note that this greedy algorithm restricts the types of regions we can obtain
  - High-dimensional rectangles
- The two-dimensional partition below could not be obtained from recursive binary partitioning

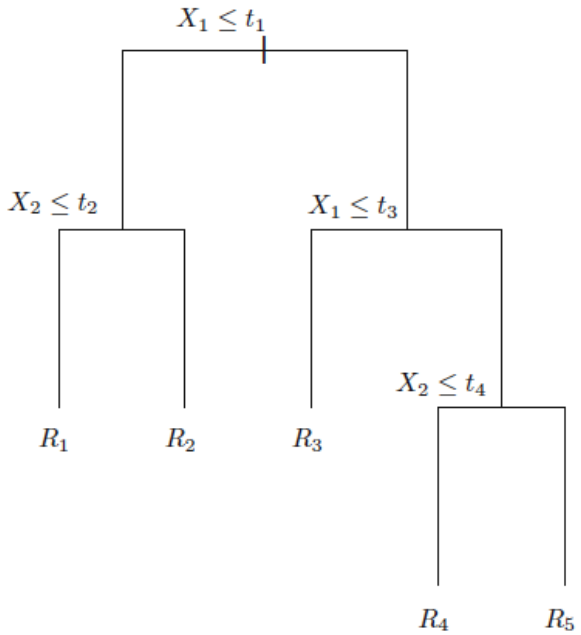


James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An introduction to statistical learning. New York: springer.

Trees

- Here is an example partition of the covariate space and corresponding tree

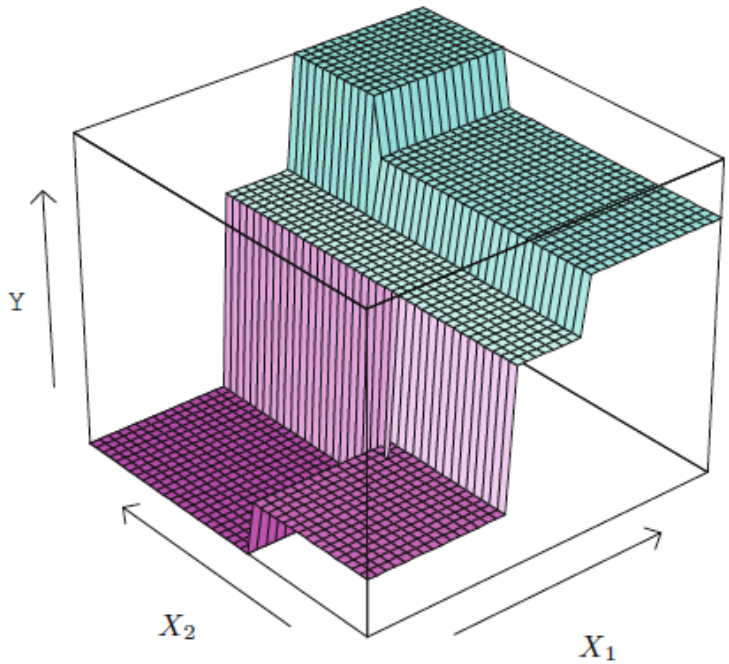




James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An introduction to statistical learning. New York: springer.

Trees

- The resulting predicted function $f(\boldsymbol{X})$ is given below
  - Flat within regions represented by R_j



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

Bias-variance trade-off

- The flexibility of the resulting surface depends on how many regions J we end up with
  - One region leads to a flat surface
    - High bias, low variance
  - n regions leads to a highly flexible surface
    - Low bias, high variance, overfitting
- The final model lies somewhere between these two extremes
  - How do we decide this?

Bias-variance trade-off

- Larger trees might lead to poor out of sample performance due to overfitting and high variability
- Smaller trees might perform better despite introducing a small amount of bias
- Pruning a tree provides a way to alleviate issues from fitting an overly large tree
  - First fit a large tree T_0
  - Then reduce the size of the large tree to obtain a subtree

Bias-variance trade-off

- It would be very computationally intensive to examine every possible subtree of the large tree
- Interestingly, for any value of α there is a subtree $T \subset T_0$ such that

$$\sum_{m=1}^{|T|} \sum_{X_i \in R_m} (Y_i - \widehat{Y}_{R_m})^2 + \alpha |T|$$

  is minimized
- Note that $|T|$ is the number of terminal nodes in a tree
- This means that we don't have to consider all possible trees T
  - Only need to consider possible α values

Bias-variance trade-off

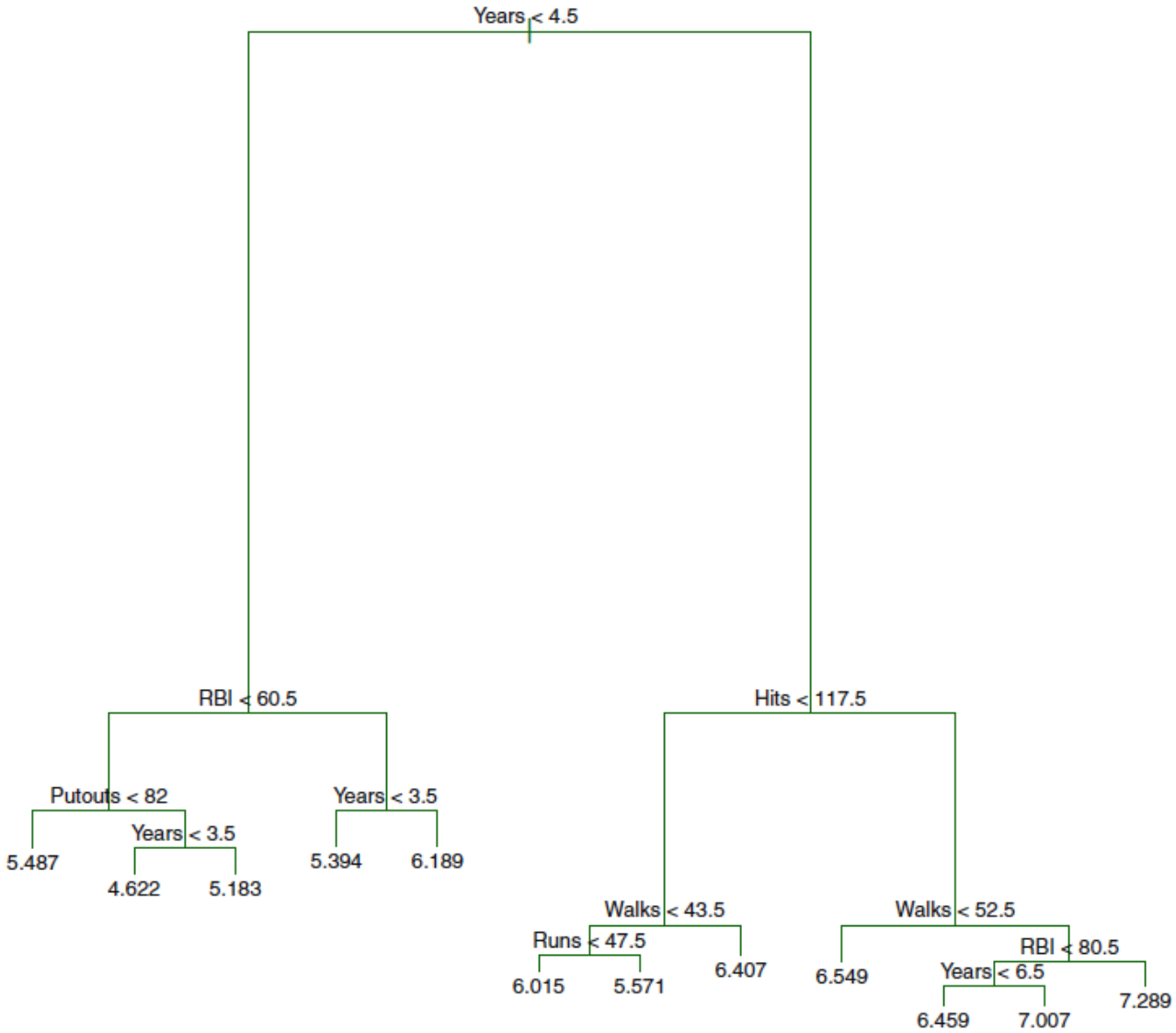
- This looks a lot like our penalized regression approaches that we have seen previously
- The first term measures the quality of the model fit on the training data
- The second term penalizes overly complex trees that could overfit to the training data
- α controls the bias-variance trade-off
  - $\alpha = 0$ implies no penalty and $T_0 = T$
  - Larger α values favor smaller trees

Bias-variance trade-off

- It turns out that it is computationally straightforward to evaluate the subtree that corresponds to each α value
- Therefore we can perform k-fold cross-validation on α to find the value that optimizes the bias-variance trade-off
- We then proceed with the tree that corresponds to the chosen α value

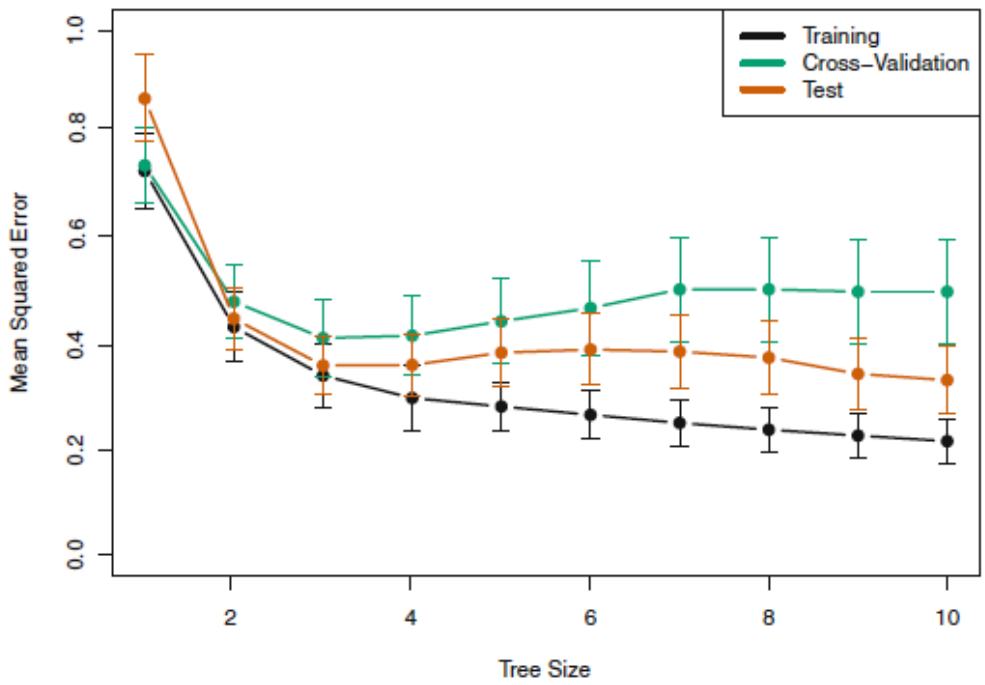
Baseball example revisited

- Below is the full tree before pruning the hitter data



Baseball example revisited

- We need to determine how much to prune the tree
- We see that the CV error is minimized at a tree size of 3
  - Interestingly, the test set error rate is low at a size of 3, but actually minimized at 10 nodes



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

Classification trees

- All of the previous discussion was restricted to continuous outcomes
- How do we translate these ideas to binary or categorical responses?
- The first issue is how to construct $\widehat{Y}_i$ once we have R_j
- $\widehat{Y}_i$ is simply the class label that is most prevalent in R_j
  - If we have 20 training observations in R_j with 12 of them being class A and 8 of them being class B, then we predict new observations in R_j to be class A

Classification trees

- We use the same recursive partitioning algorithm to find the splits
- RSS is no longer a good measure to determine the quality of the resulting tree
  - Need an analogue for categorical data
- Can use one of three quantities
  - Classification accuracy
  - The gini index
  - Entropy

Classification trees

- Suppose that there are K classes that the outcome can be
- Define $\widehat{p}_{mk}$ to be the proportion of training observations falling in R_m that are class k
- The classification error rate is simply the proportion of subjects in R_j that don't belong to the class with the highest $\widehat{p}_{mk}$ value

$$\text{Error} = 1 - \max_k(\widehat{p}_{mk})$$

- If 70% of the observations are in one class, then the error rate is 30%

Classification trees

- It has been shown that the classification error rate is not the best metric when determining how to grow trees
- The Gini index is one of the preferred approaches

$$G = \sum_{k=1}^{K} \widehat{p}_{mk}(1 - \widehat{p}_{mk})$$

- Remember that the bernoulli variance is $p(1-p)$ where p is the proportion of successes
- This measure gives a sense of the total variability across the K classes

Classification trees

- The Gini index is closer to zero when many of the $\widehat{p}_{mk}$ are close to zero or 1
  - These are situations when most observations are correctly classified
  - Gini index measures node purity
    - Most observations are from the same class

- Entropy is another measure that can be used

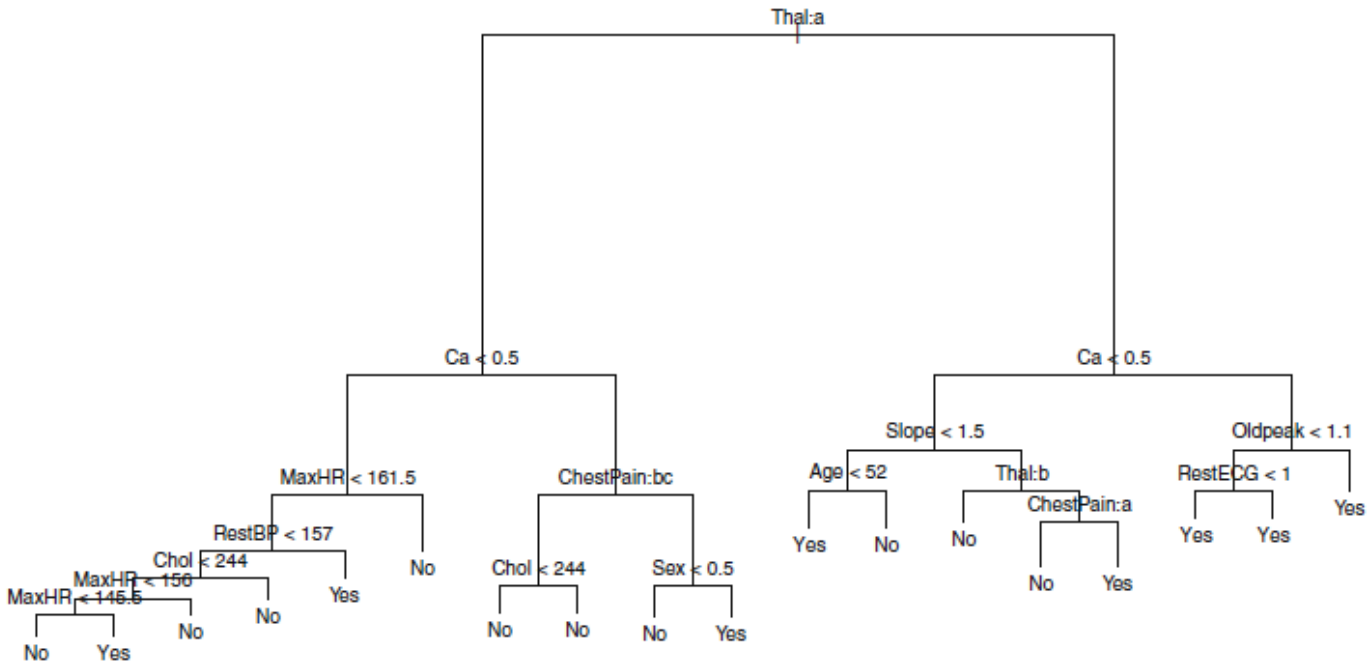
$$D = -\sum_{k=1}^{K} \widehat{p}_{mk} \log \widehat{p}_{mk}$$

Classification trees

- It is a bit less obvious, but entropy is also positive and is closer to zero when $\widehat{p}_{mk}$ is closer to 0 or 1
- The Gini index and entropy function behave similarly and tend to give very similar values
- Either of these approaches can be used to decide how to split and build regression trees
- Any of the three can be used as a metric to help prune trees
  - Classification accuracy favored if you want good out of sample prediction

Classification trees

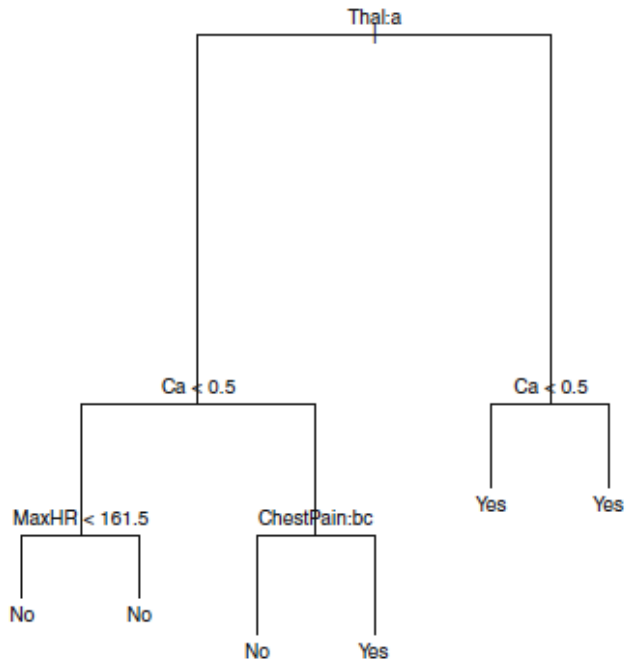
- Here is a classification tree for a binary outcome before pruning



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

Classification trees

- And here is the tree after pruning and cross-validation is applied to get a smaller tree with better predictive accuracy



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

Advantages and disadvantages of trees

- There are two major advantages of trees
  - Simple and interpretable
  - Easily account for a lack of additivity, i.e. interactions
- Interpretability can be very useful when explaining the model to a non-statistician
  - Decision trees replicate decision processes used by humans
- The tree structure allows for the effects of each covariate to depend on the values of others
  - Deeper trees lead to higher-order interactions

Advantages and disadvantages of trees

- There are also major disadvantages of trees
- The tree structure can be very restrictive in terms of the types of functions that are well modeled by trees
- The fact that trees can model non-additive functions is an advantage, but also a disadvantage
  - By construction, trees have to be non-additive!
- What if we want to model the following function

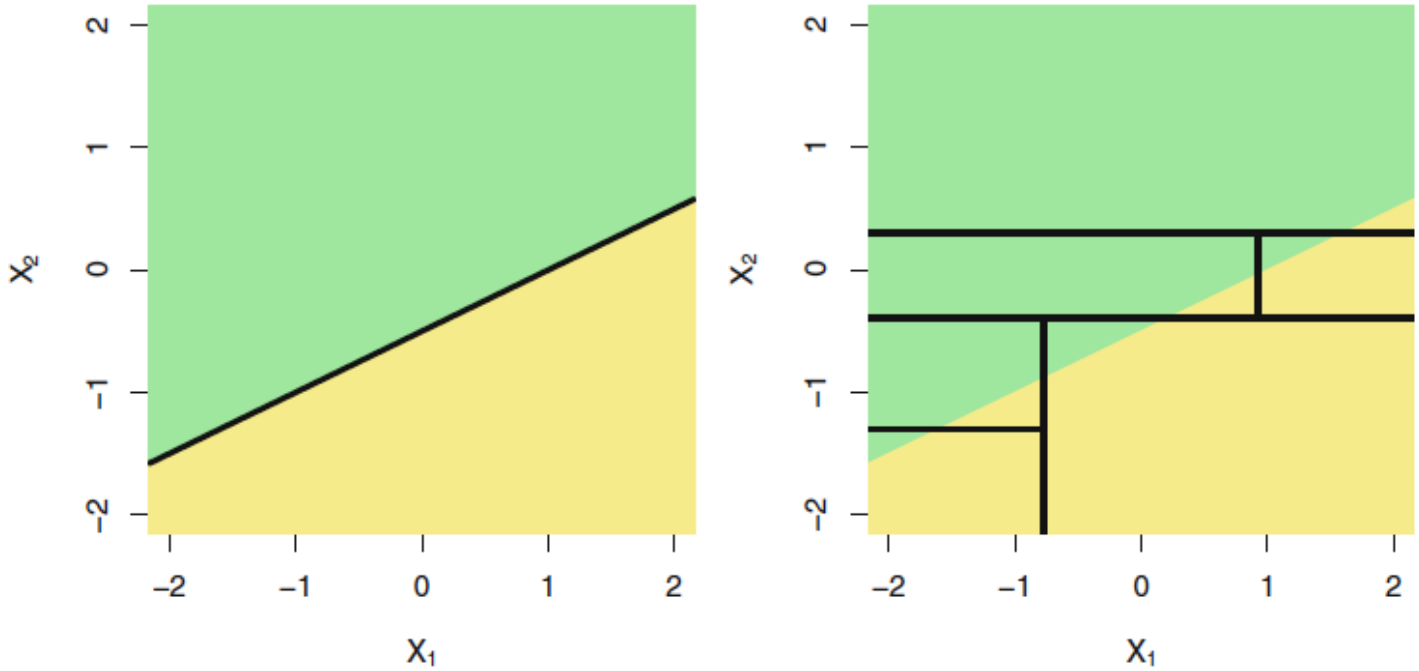
$$Y_i = c_1 1(X_1 > t_1) + c_2(X_2 < t_2) + \cdots + c_p(X_p < t_p) + \epsilon_i$$

Advantages and disadvantages of trees

- The partitioning algorithm is not equipped to capture this function even though it's relatively simple and additive
- Trees are very sensitive to small changes in the data
  - Small changes lead to very different trees
  - Imagine if a change in the data leads to a different first split
    - Entire tree is different
- Trees tend to have worse predictive accuracy than the other flexible approaches we've seen
  - Trade-off between prediction accuracy and interpretability

Advantages and disadvantages of trees

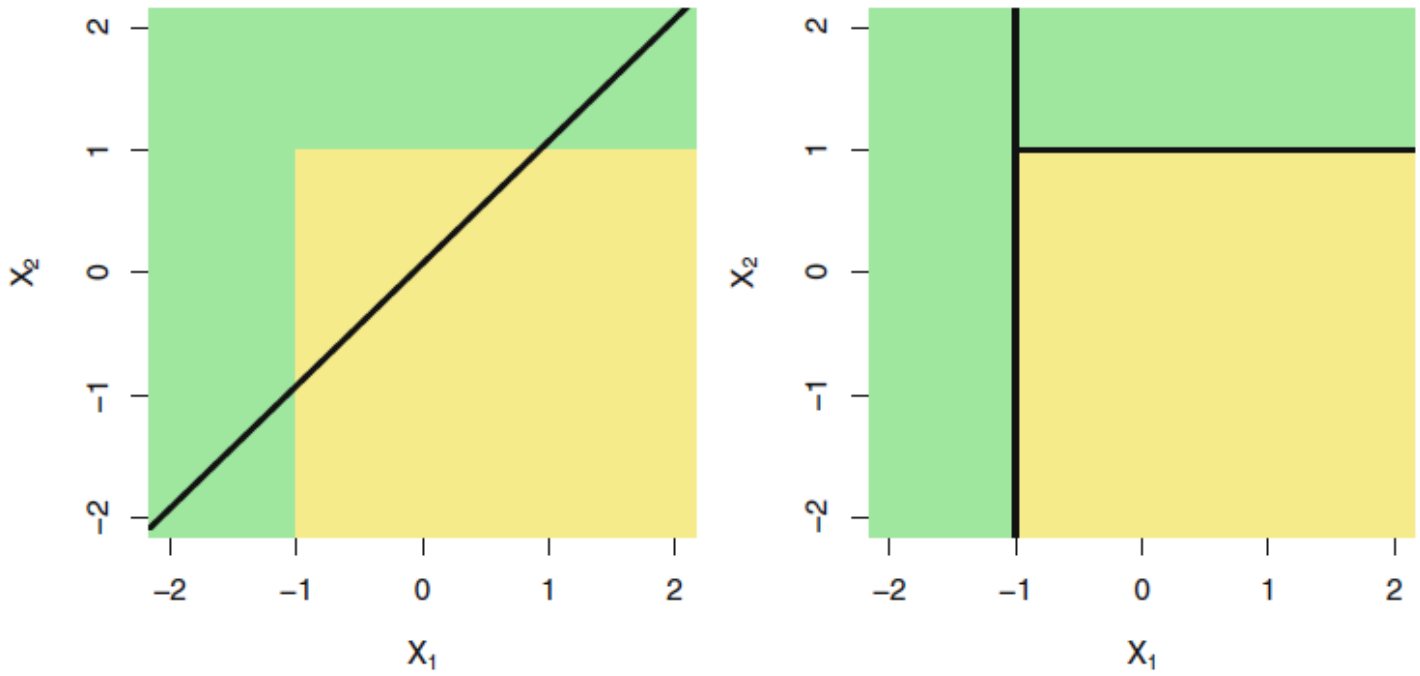
- Trees are also not well equipped to capture simple, smooth functions
- Suppose the true function is linear
  - Linear model (left panel) drastically outperforms a tree-based approach (right panel)



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

Advantages and disadvantages of trees

- Alternatively, if the true function is very nonlinear with interactions, then the tree model will outperform linear models



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

Advantages and disadvantages of trees

- Thankfully there are improvements to trees which can improve their prediction accuracy
- Main idea of each approach is to reduce the variability of the predictions at the cost of interpretability
- We will now explore three such approaches to improving prediction for trees
  - Bagging
  - Random forests
  - Boosting
- All are based in some form on making multiple trees and averaging results to reduce variability

Averaging to improve inference

- Before we discuss its application to tree models, let's briefly discuss averaging as a method to improve predictions and why averaging can help so much
- Suppose that we have M different estimators of a quantity
- Call these $\widehat{\theta}_1, \ldots, \widehat{\theta}_M$
- We could choose our favorite one, or we could simply average them
  - I will argue that averaging is better

Averaging to improve inference

- Define the average of these quantities as

$$\widehat{\theta}^A = \frac{1}{M}\sum_{m=1}^{M}\widehat{\theta}_m$$

- Is this a good estimator?
  - Need to look at it's statistical properties
- The bias of the averaged estimator is simply the average of the biases of individual estimators

$$E(\widehat{\theta}^A - \theta) = \frac{1}{M}\sum_{m=1}^{M}E(\widehat{\theta}_m - \theta)$$

Averaging to improve inference

- This can provide robustness against choosing one bad estimator with a large bias
- For today's discussion and for tree-based models, we will focus more on variance
- Letting Σ be the covariance matrix of $(\widehat{\theta}_1, \ldots, \widehat{\theta}_M)$ we have that

$$\text{Var}(\widehat{\theta}^A) = \frac{1}{M^2} \sum_{i=1}^{M} \sum_{j=1}^{M} \Sigma_{ij}$$

- The variance depends on the correlation between each estimator

Averaging to improve inference

- Smaller correlations lead to reductions in variance
- To see this further, let's simplify and assume that the variance of each estimator is σ^2 and the correlation between each estimator is $\rho > 0$
- In this setting, the variance of the average is given by

$$\rho\sigma^2 + (1-\rho)\frac{\sigma^2}{M}$$

- This shows the massive gains in efficiency that averaging can achieve

Averaging to improve inference

- If $\rho = 0$, the variance becomes σ^2/M
  - $1/M$ times the variance of any one estimator
- If $\rho = 1$, averaging doesn't help at all and we still obtain a variance of σ^2
  - Doesn't hurt us either!
- In reality the truth usually lies somewhere in the middle, which can lead to big decreases in the variance of our estimator

Bagging regression trees

- Bagging is an idea that builds on the concept of averaging to reduce the variability in our resulting predictions
- Bagging is a general procedure that works in broader contexts than simply regression trees
  - Very commonly used in regression trees, however
- The regression trees we have described so far can have very high variability
  - Poor out of sample predictions
  - High interpretability

Bagging regression trees

- Bagging stands for bootstrap aggregation
- The main idea of bagging is to build many trees from different data sets that are representative of the population of interest and average them
- How do we obtain multiple data sets to build trees with?
  - The bootstrap

Bagging regression trees

- Bagging is a relatively simple procedure for finding $f(X)$
  1. For $b = 1, \ldots, B$ do:
     1. Draw n samples from our data with replacement, i.e. a bootstrap sample of your data
     2. Using this bootstrap sample, estimate $\widehat{f}^{*b}(X)$, which in this case is a regression tree
  2. Define the bagging estimate as

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

Bagging regression trees

- Each of these trees is grown deep
  - Many terminal nodes
  - Very flexible estimated function
  - No pruning needs to be done
- Since these trees are grown deep, they will have low bias but very high variance
- Averaging reduces the variance to end up with an estimator with low bias and low variance

Bagging regression trees

- The choice of B is not terribly important, and should just be chosen to be very large
  - Large values of B does not lead to overfitting
  - Larger B values lead to reductions in variance, but only up to some point
- Bagging can drastically improve prediction accuracy over a single tree
- A drawback of bagging is that we have completely lost the interpretability of trees

Bagging classification trees

- We need to modify bagging slightly when doing classification
- Previously we calculated the average value via

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

- Instead we can use a majority vote approach by picking the classification that is most commonly chosen across bagged trees
- For binary classification this amounts to

$$\widehat{f}_{bag}(X) = \begin{cases} 1 & \frac{1}{B}\sum_{b=1}^{B}\widehat{f}^{*b}(X) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Bagging regression trees

- Bagging provides a natural way to estimating the testing error rate
- In any one bootstrap sample, the probability that observation i is excluded is given by

$$\left(\frac{n-1}{n}\right)^n$$

- It turns out this is very close to $1/3$ for any realistic n
- This means that roughly $B/3$ of the bagged trees did not use training point i to estimate the tree

Bagging regression trees

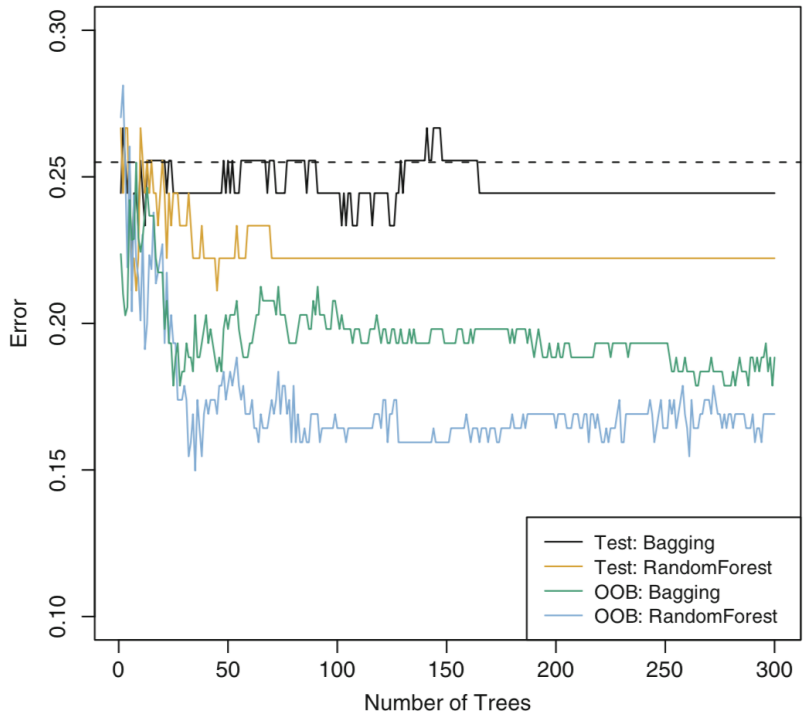
- We can predict the outcome for subject i in each of these roughly $B/3$ trees that did not include observation i
  - Samples not included in a bagged data set are called out-of-bag (OOB) observations
- Average the outcome estimate across all instances in which observation i was not included
  - Or take majority vote if performing classification
- We can then compare these estimates to the true values
  - MSE for quantitative response
  - Classification error for categorical response

Bagging regression trees

- This estimate of MSE or classification error is a valid estimate of the out of sample prediction error
- It is valid, because the observations were not used to make the predictions and therefore is not susceptible to overfitting
- Very helpful because it avoids needing to do cross-validation and can save computation time

Bagging regression trees

- Here is an example showing the testing error rates and estimates of the error rates from bagging
  - Dashed line is the error from a single tree
  - Bagging improves results, but only slightly in this case
  - Out of bag estimate of the error is somewhat optimistic here



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

Bagging regression trees

- In the previous example, an approach called random forests was also used
  - Performed better than bagging
- Random forests are another way to average over many trees
- Random forests try to improve on bagging by reducing the correlation of our estimates across trees
  - Remember that averaging works best when correlation is smaller
- We will cover random forests in the following week's lecture