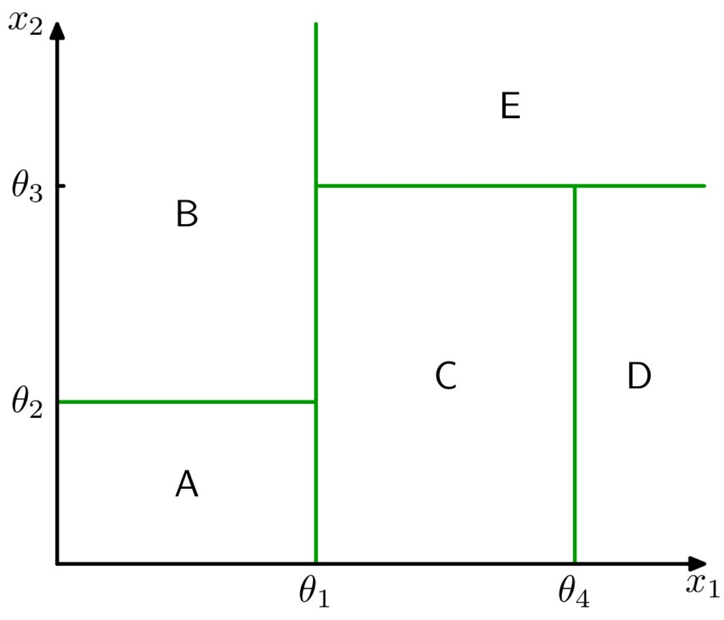
Introduction to Machine Learning

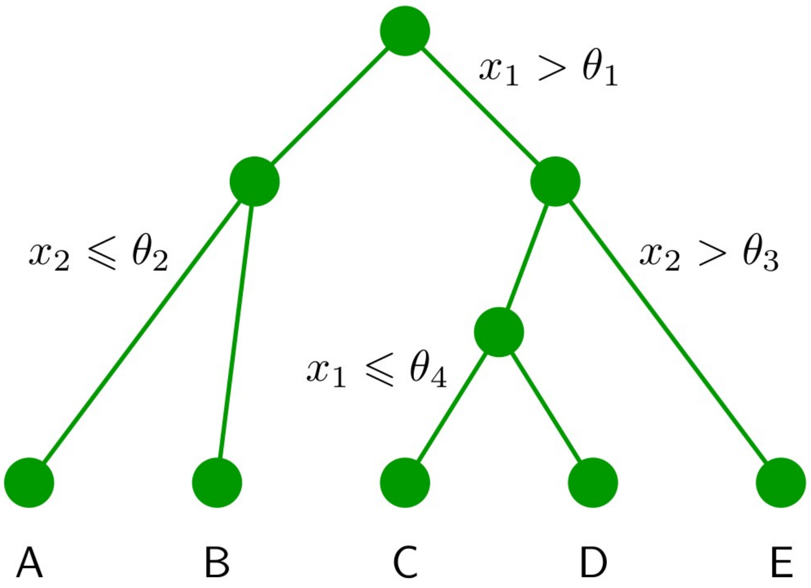
Lecture 8
Tree-based models I

Goker Erdogan
26 – 30 November 2018
Pontificia Universidad Javeriana

Decision trees

- A popular technique for supervised learning
  - Both for regression and classification
- Idea
  - Split the input space into rectangular regions
  - Predict using a constant value for each region



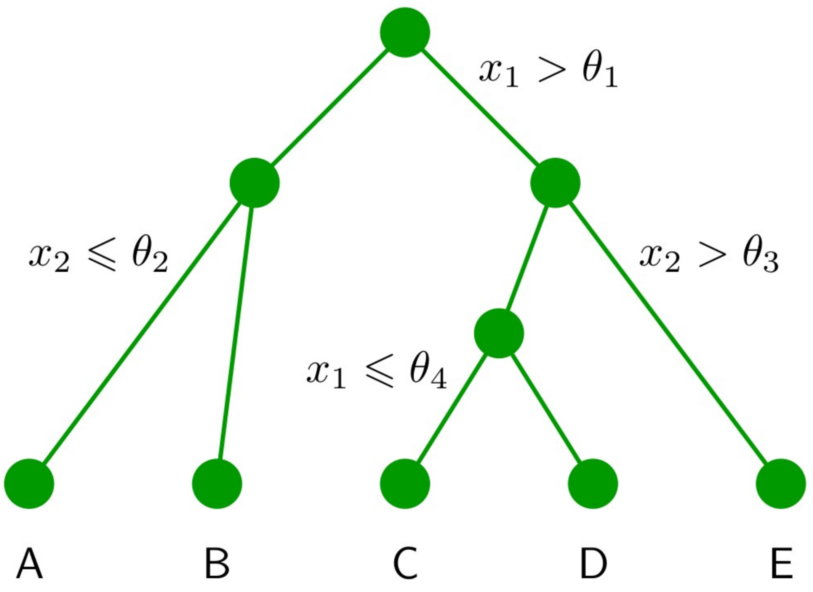


[3]

Decision trees

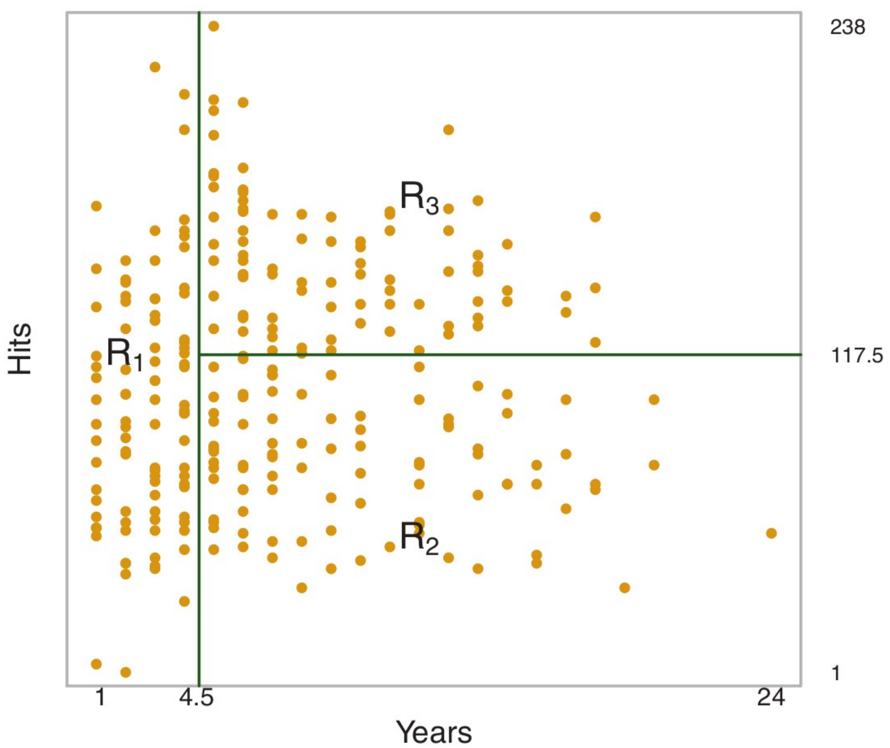
- Advantages
  - Easy to interpret
  - Easy to display graphically
  - Non-linear

- Terminology
  - Internal nodes
  - Branches
  - Leaves or terminal nodes



Regression trees

- Given N training samples of $\{x, y\}_{n=1...N}$
- Split the input space into rectangles
- For each leaf (region)
  - prediction = the mean of the target values in that region
  - Why? Because that is the prediction that minimizes error (sum-of-squares).



[2]

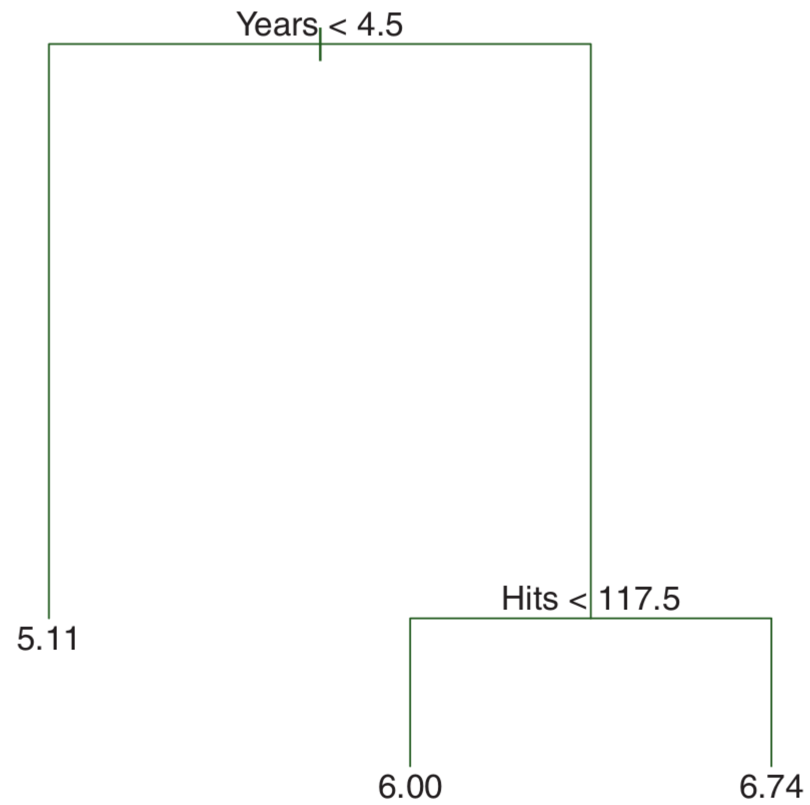


FIGURE 8.1. *For the* `Hitters` *data, a regression tree for predicting the log salary of a baseball player, based on the number of years that he has played in the major leagues and the number of hits that he made in the previous year. At a given internal node, the label (of the form $X_j < t_k$) indicates 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.*

Fitting a regression tree

- Regression tree with M leaves (regions)
  - Given x, our prediction is f(x)

$$f(x) = \sum_{m=1}^{M} \bar{y}_m I(x \in R_m)$$

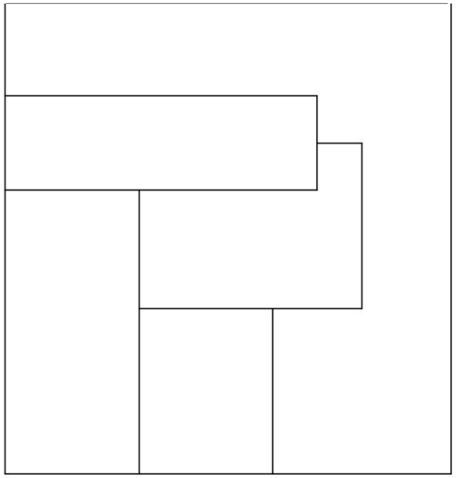
  - where $\bar{y}_m$ is the average of all y in R_m

- Given a training set $\{x, y\}_{n=1...N}$
  - Find the partitioning of space (i.e., rectangles) that minimizes error

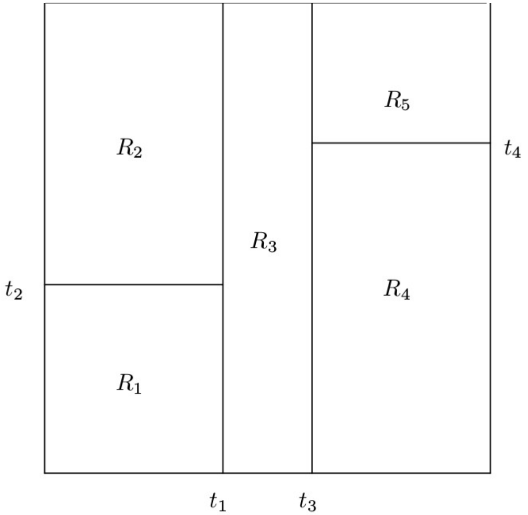
$$RSS = \sum_{m} \sum_{i \in R_m} (y_i - \bar{y}_m)^2$$

Fitting a regression tree

- Why not consider all possible partitionings?
  - The number of possible partitionings is huge!



  - We consider only (recursive) binary splits on single variables



Fitting a regression tree

- Various algorithms
  - CART
  - C4.5 (C5.0)
- These are usually
  - Top-down: Start from all data at the root
  - Greedy: At each point, pick the split that maximizes reduction in error

General outline for regression tree fitting procedure

- For each feature j
  - Find the best split position t
  - Calculate the reduction in RSS
- Pick the feature j that reduces the error the most
- Add a new node splitting feature j at value t
- Repeat until reached a stopping criteria

Fitting a regression tree

- How do we find the best split point t for a feature?

$$R_1(j,t) = \{X|X_j < t\} \text{ and } R_2(j,t) = \{X|X_j \geq t\}$$

$$\min_t \sum_{i:x_i \in R_1} (y_i - \bar{y}_{R_1})^2 + \sum_{i:x_i \in R_2} (y_i - \bar{y}_{R_2})^2$$

- No easy method
- Try every possible (or a subset of all) t
  - Look at the training data, find all unique y values and sort them
  - Try each y one by one and calculate the error
  - Pick the one that has the minimum error

When do we stop?

- Popular stopping criteria
  - Stop when there are fewer than K samples in each region
  - Stop when tree reaches a maximum depth
- Another idea
  - Split if only reduction in RSS > R
  - Empirically not a good rule
  - Too short-sighted
- We don’t want a very large tree
  - Risk of overfitting
  - Think of tree size as a complexity parameter
- Build a large tree then prune it

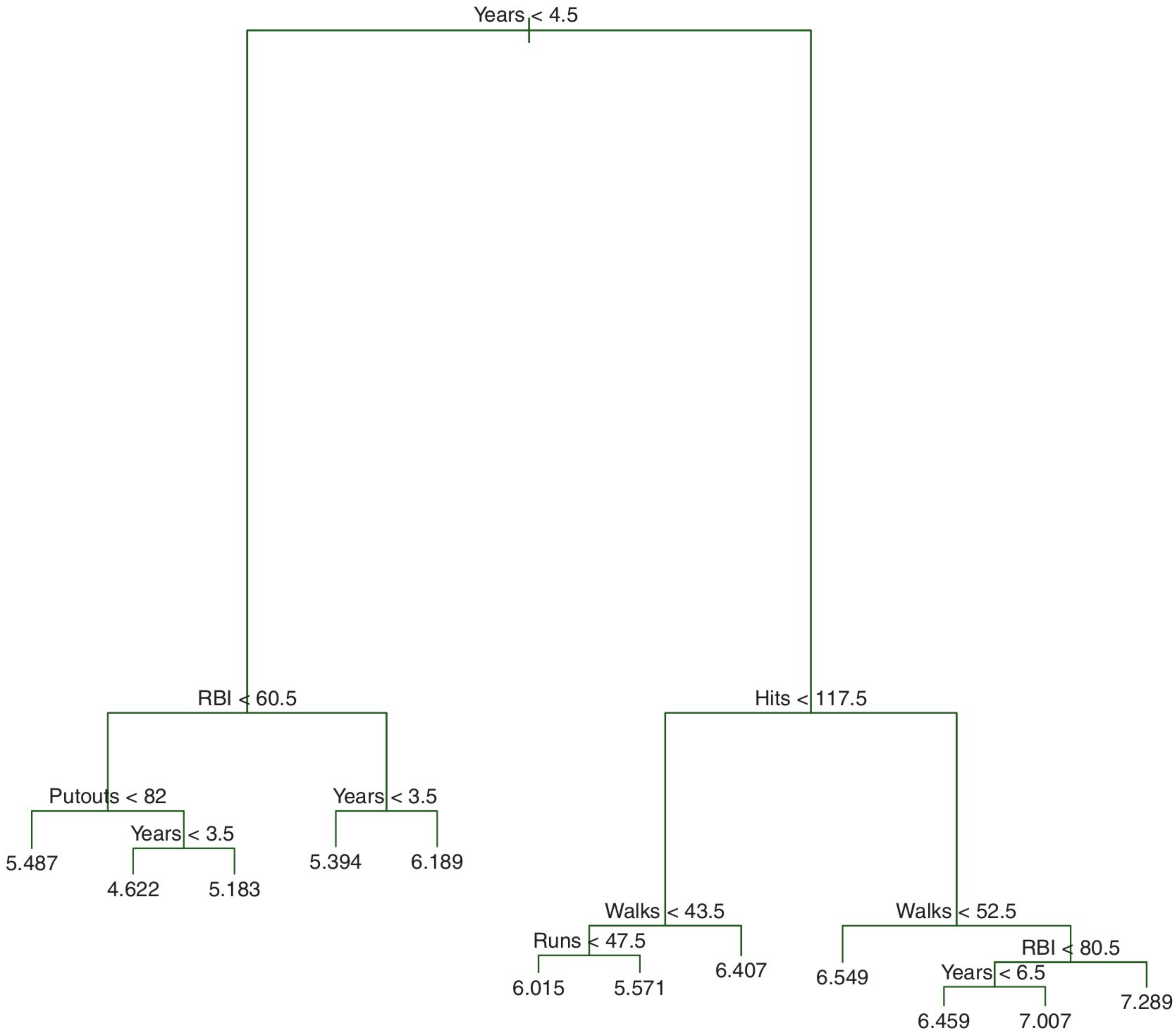


FIGURE 8.4. *Regression tree analysis for the* `Hitters` *data. The unpruned tree that results from top-down greedy splitting on the training data is shown.*

How to prune the tree

- Weakest link pruning
  - For each internal node
    - Calculate the increase in error if we removed that node
  - Remove the node with smallest increase in error
  - Evaluate the new tree on a validation set
  - Repeat

- Note we evaluate on a validation set
  - e.g., use k-fold cross-validation

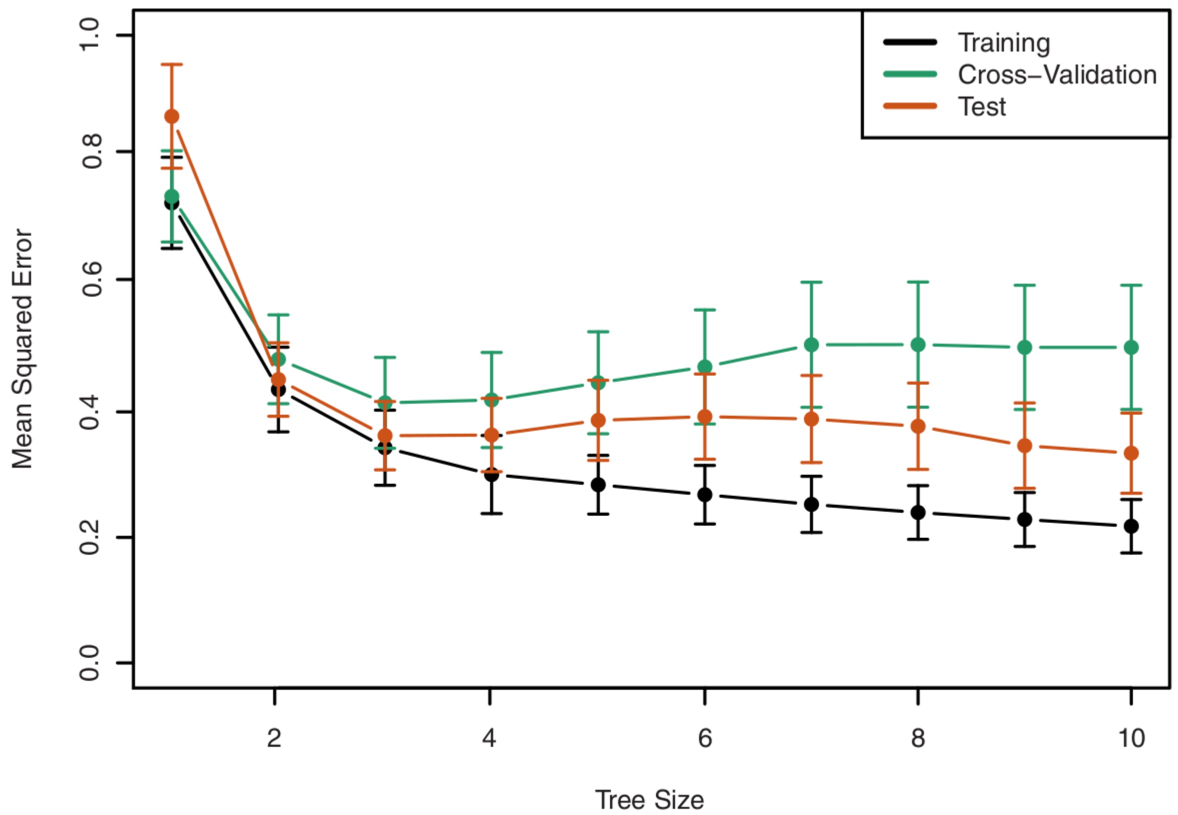


FIGURE 8.5. *Regression tree analysis for the* `Hitters` *data. The training, cross-validation, and test MSE are shown as a function of the number of terminal nodes in the pruned tree. Standard error bands are displayed. The minimum cross-validation error occurs at a tree size of three.*

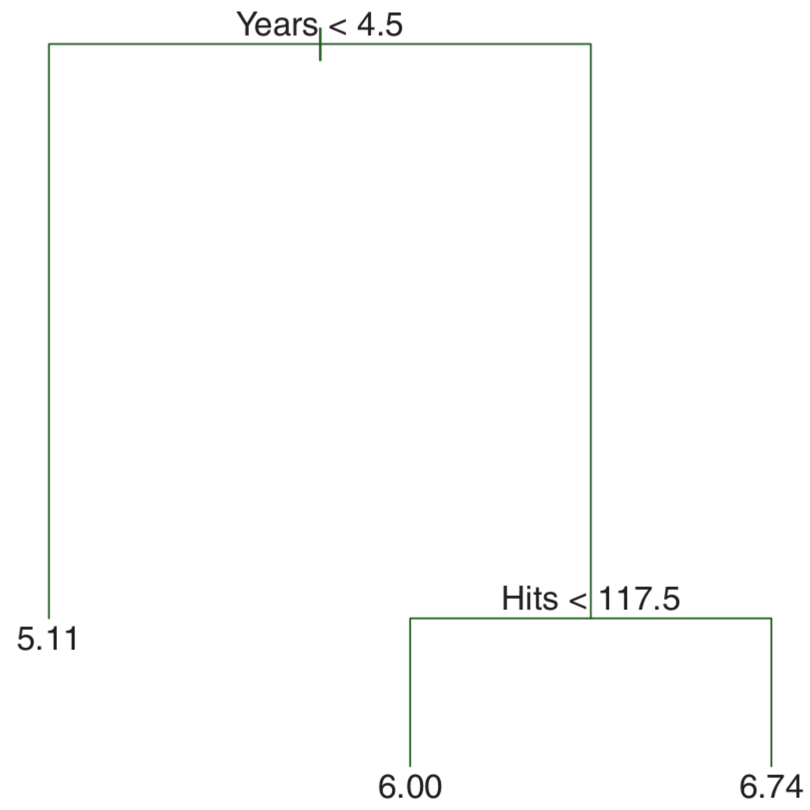


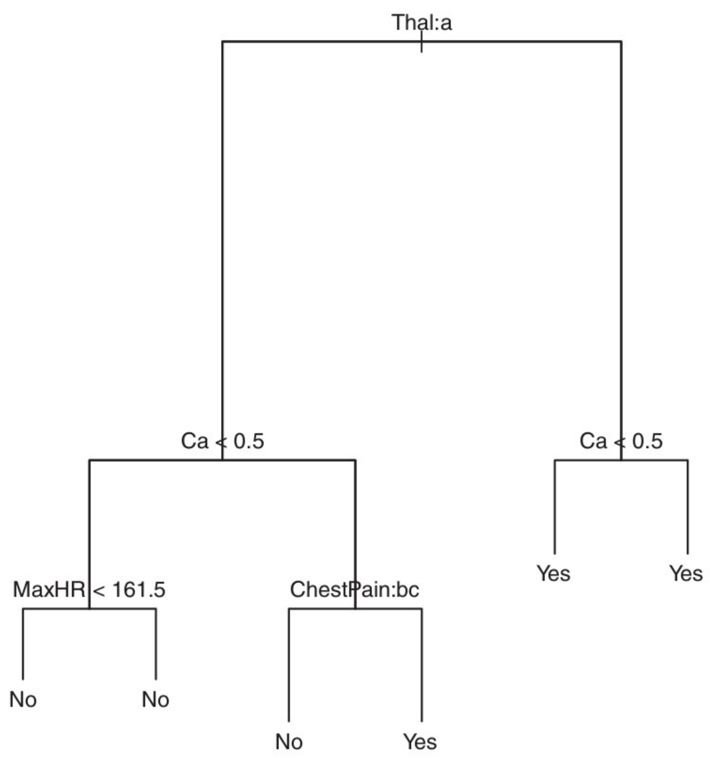
FIGURE 8.1. *For the* `Hitters` *data, a regression tree for predicting the log salary of a baseball player, based on the number of years that he has played in the major leagues and the number of hits that he made in the previous year. At a given internal node, the label (of the form $X_j < t_k$) indicates 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.*

Classification tree

- Classification tree with M leaves (regions)
  - Given x, our prediction is f(x)

$$f(x) = \sum_{m=1}^{M} k_m I(x \in R_m)$$

  - where k_m is the majority class in R_m



How to split nodes

- We want to minimize misclassification rate

$$R_1(j,t) = \{X|X_j < t\} \text{ and } R_2(j,t) = \{X|X_j \geq t\}$$

$$\min_t \; N_{R_1} E(R_1,k)) + N_{R_2} E(R_2,k))$$

- N_{R_i} is the number of samples falling in region R_i
- E(R, k) is some error function

- What can E be?
  - Misclassification rate: $1 - \max_k(\bar{p}_{R,k})$
  - Gini index: $\sum_k \bar{p}_{R,k}(1 - \bar{p}_{R,k})$
  - Entropy: $-\sum_k \bar{p}_{R,k} \log(\bar{p}_{R,k})$

$\bar{p}_{R,k} =$ Ratio of points from class k in region R

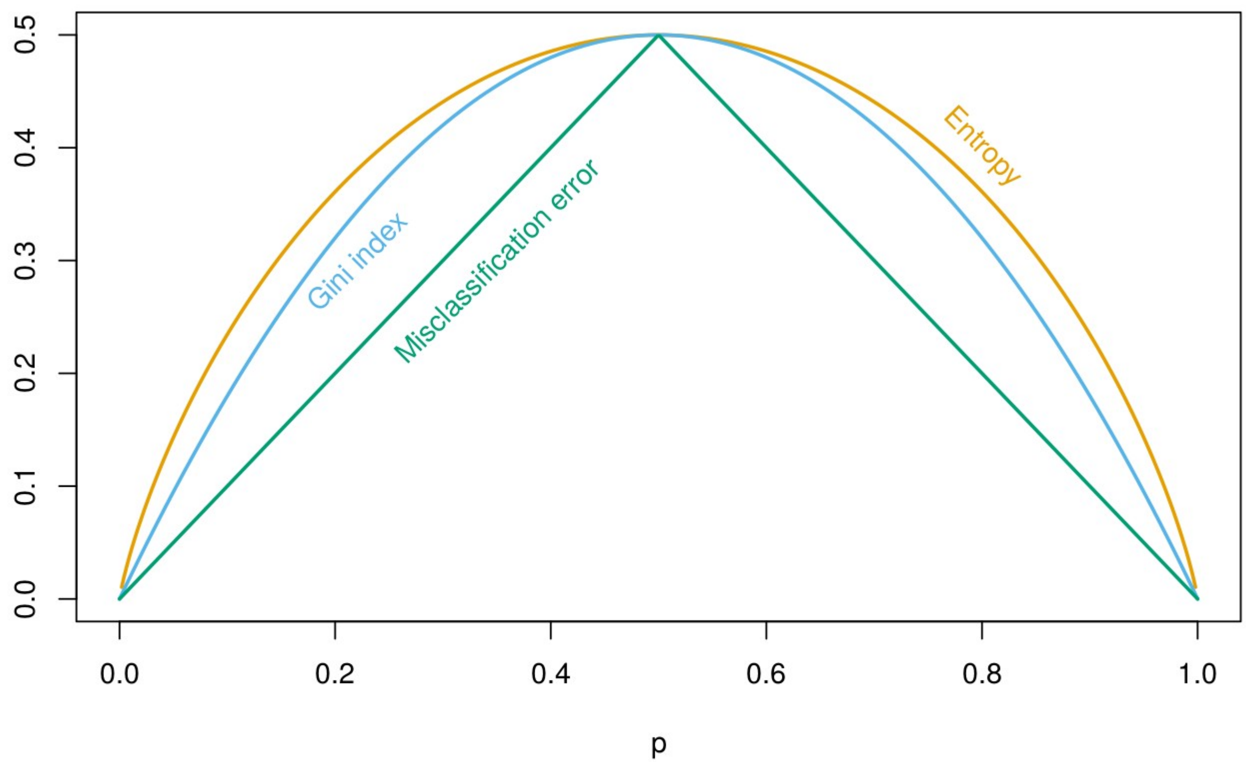
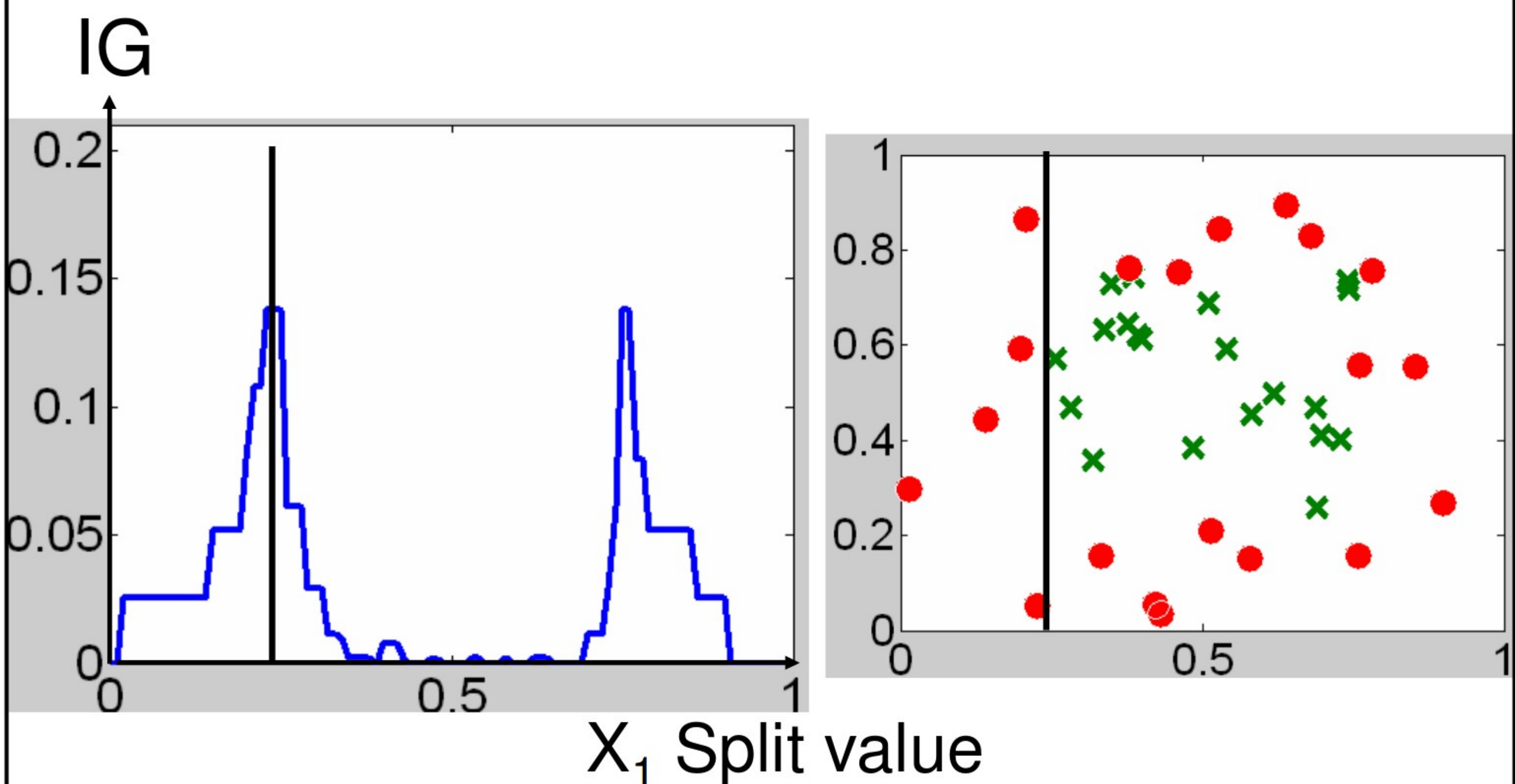


FIGURE 9.3. *Node impurity measures for two-class classification, as a function of the proportion p in class 2. Cross-entropy has been scaled to pass through* $(0.5, 0.5)$.

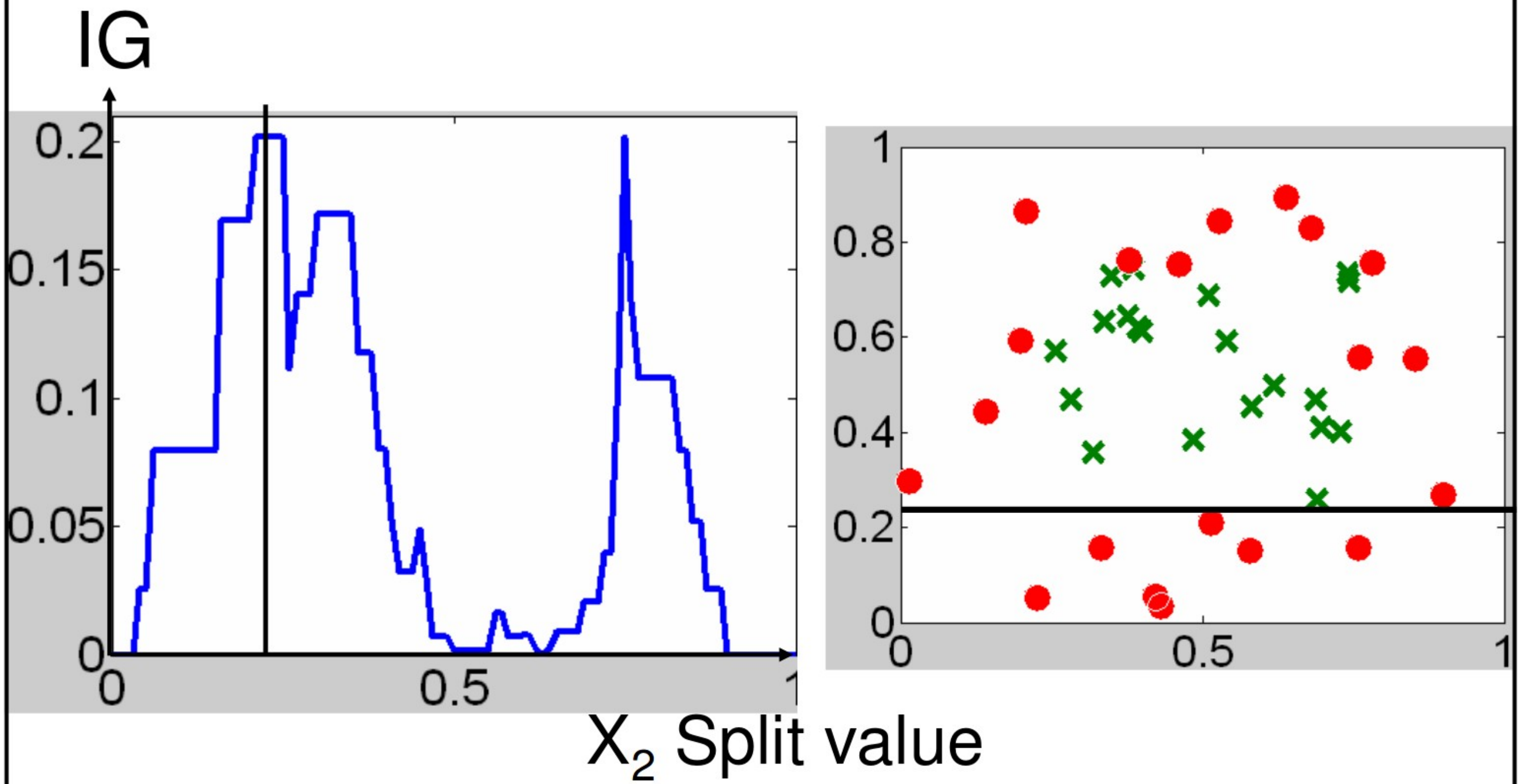
Why Gini or entropy?

- Gini and entropy works better in practice
- They care more about purity
  - Split 1: Left → (300, 100), Right → (100, 300)
  - Split 2: Left → (200, 400), Right → (200, 0)
  - Misclassification rate: 0.25 for both
  - But split 2 is purer
- Gini and entropy are differentiable

- Classification tree fitting
  - Use the same algorithm as regression trees
    - Using Gini/entropy instead of RSS to split nodes
  - Prune
    - Use misclassification rate to do the pruning



Best split value (max Information Gain) for X_1 attribute: 0.24 with IG = 0.138



Best split value (max Information Gain) for X_2 attribute: 0.234 with IG = 0.202

[5]

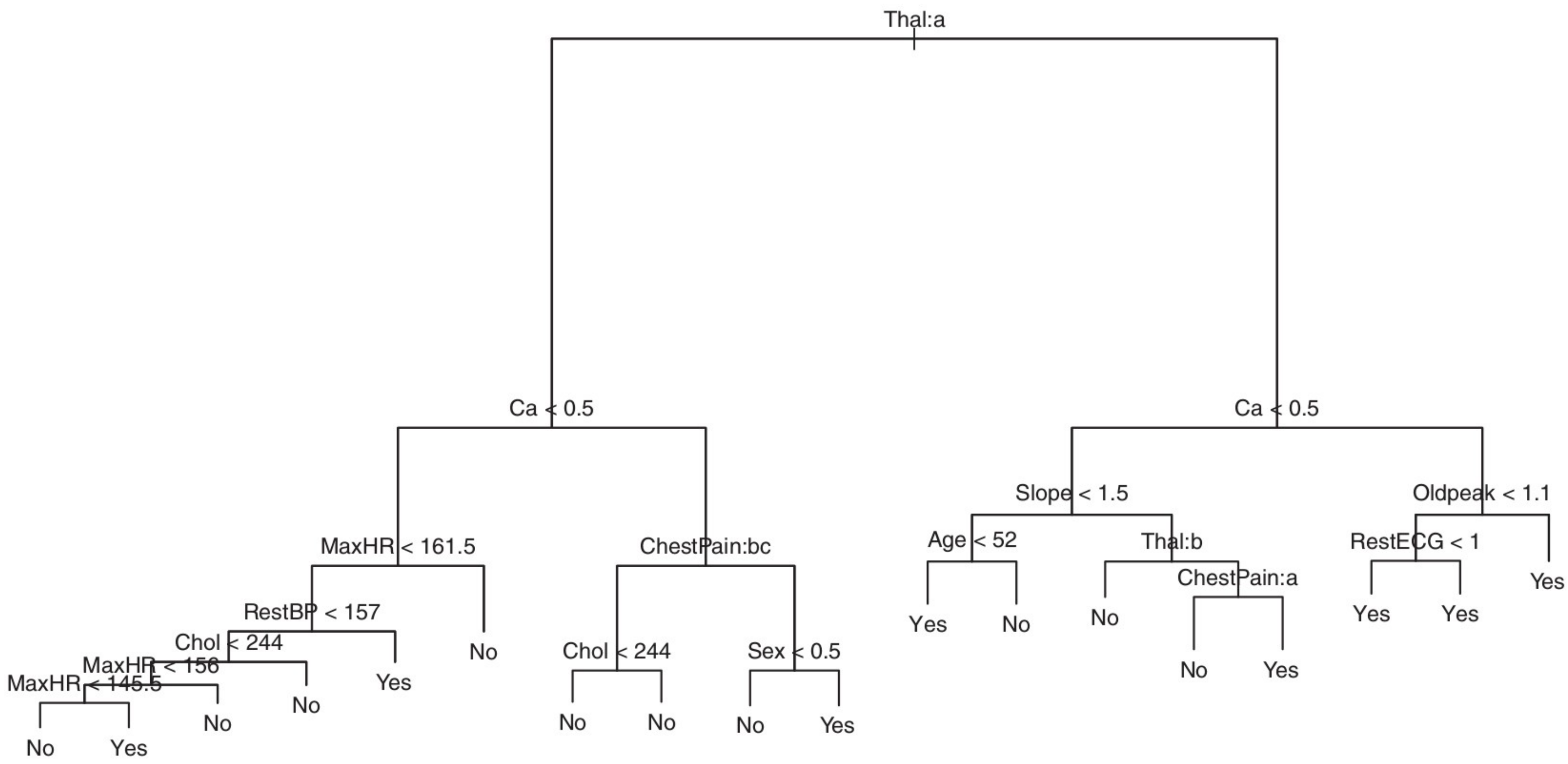


FIGURE 8.6. Heart *data.* Top: *The unpruned tree.*

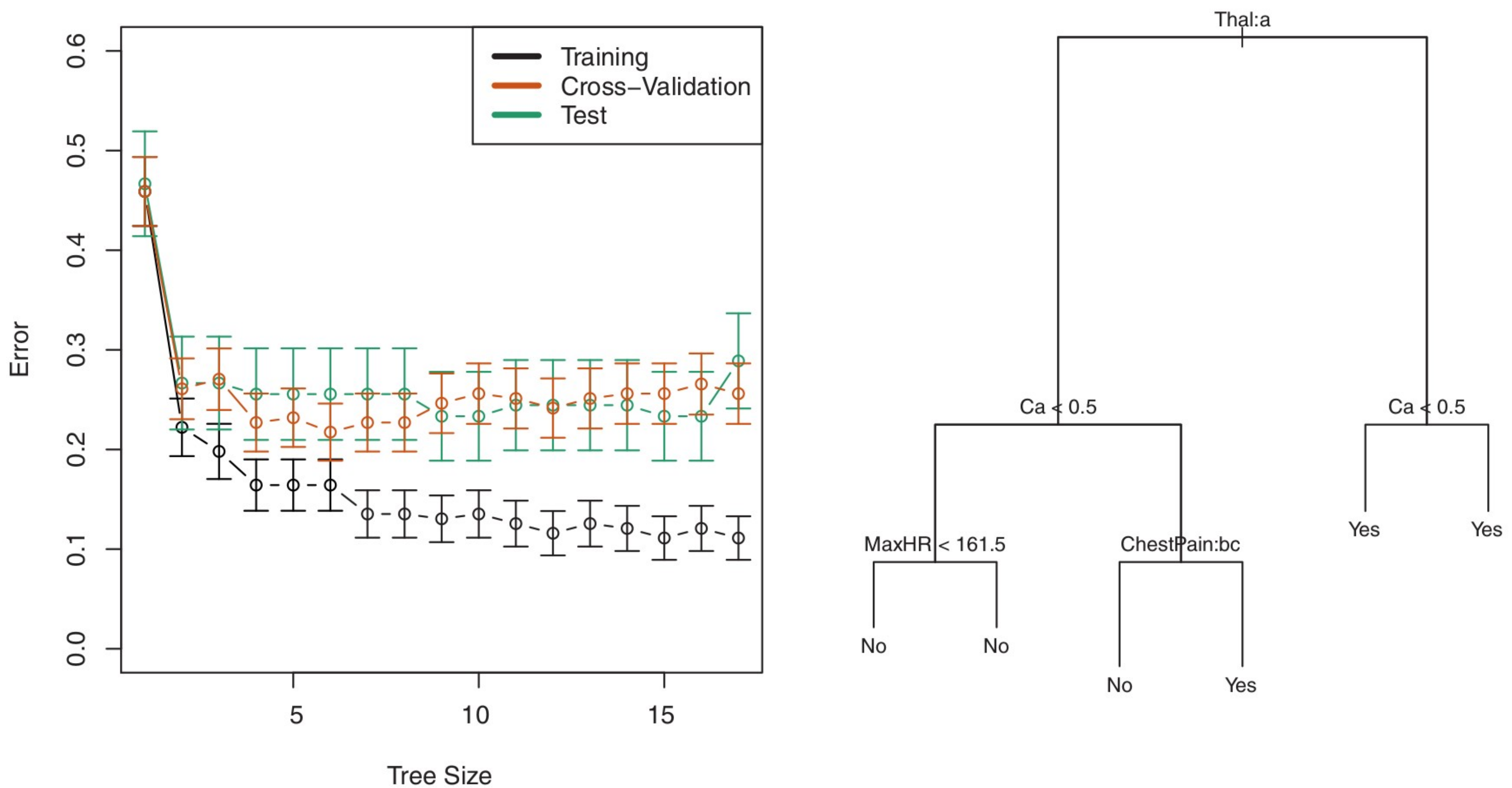


FIGURE 8.6. Heart *data.* Top: *The unpruned tree.* Bottom Left: *Cross -validation error, training, and test error, for different sizes of the pruned tree.* Bottom Right: *The pruned tree corresponding to the minimal cross-validation error.*

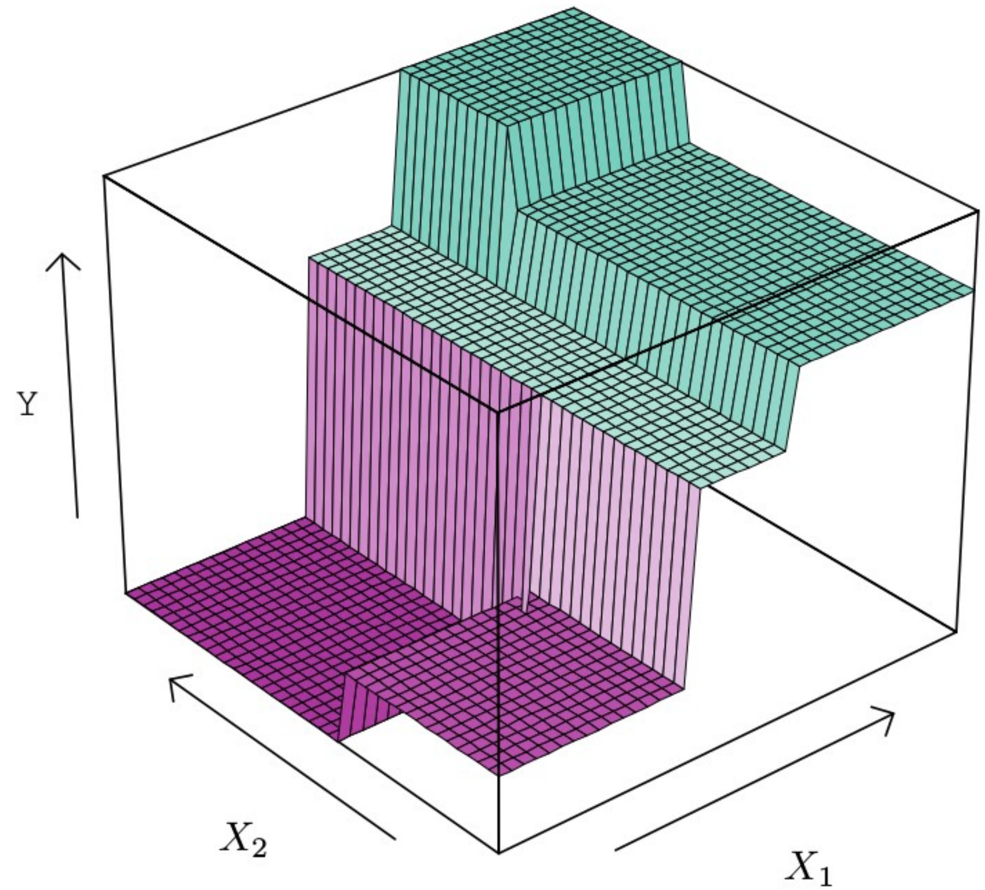
Advantages/disadvantages

- Advantages
  - Easy to interpret
  - Easy to display graphically
  - Non-linear
  - Can handle qualitative predictors
  - Can handle missing data

- Disadvantages
  - Does not perform very well
    - High variance, mostly because the tree is built sequentially
    - Various improvements: bagging/boosting/random forests

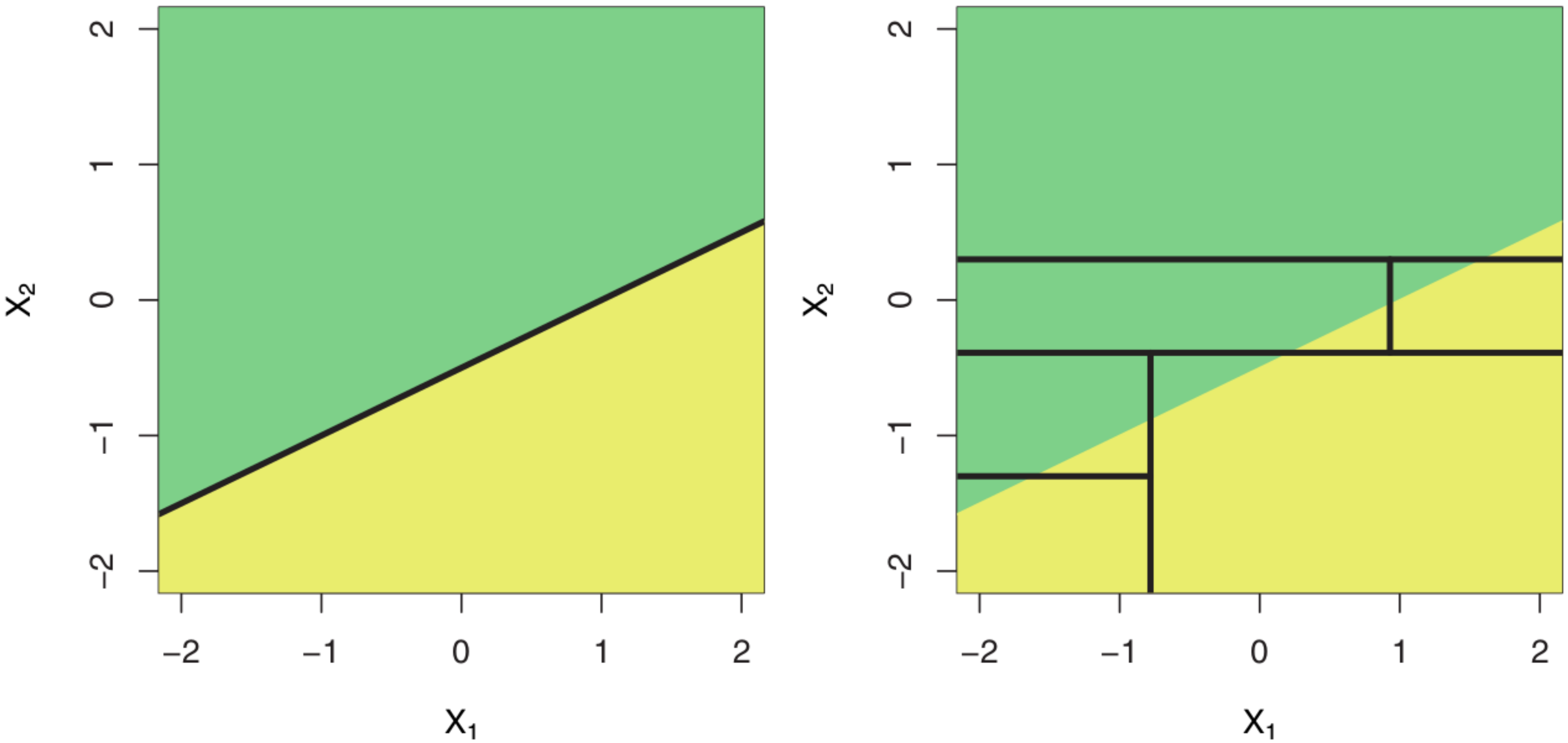
Disadvantages

- Non-smooth



Disadvantages

- Assumption of rectangular regions might be problematic



Summary

- Decision trees
- Regression trees
  - Fitting algorithm
    - Stopping criteria
  - Pruning
- Classification trees
  - Splitting criteria: Gini, entropy
- Advantages/disadvantages

- Exercises
  - Do the labs in Section 8.3.1 and 8.3.2 in ISLR

References

[1] James, Witten, Hastie, and Tibshirani. An Introduction to Statistical Learning with Applications in R. Chapter 8.

[2] Hastie, Tibshirani, and Friedman. The Elements of Statistical Learning. Chapter 9.

[3] Bishop. Pattern Recognition and Machine Learning. Chapter 14.

[4] Murphy. Machine Learning: A Probabilistic Perspective. Chapter 16.

[5] https://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15381-s06/www/DTs.pdf