Interactions and Trees

POL 212: Quantitative Analysis I

Winter 2024

Interactions

Conditional relationships

- We include an interaction term any time we are interested in a **conditional relationship**, or one in which we think the effect of one variable depends on the value of another (moderator) variable.
- For example, suppose that income has a negative effect on voters' evaluation of the Democratic Party.
- But, this effect may be stronger for one group of voters than another (perhaps voters who pay more attention to the news evaluate the parties more strongly on the basis of their economic policies).

Interaction terms

- We could test this hypothesis. Let Y_i represent evaluations of the Democratic party, X_i income in thousands of dollars, and D_i be a dummy for whether the person consumes news on a daily basis: $Y_i = \alpha + \beta_1 X_i + \beta_2 D_i + \beta_3 X_i D_i + u_i$.
- The slope and intercept of the regression line are different based on whether the person watches news or not:
  - Model for news watcher ($D_i = 1$):
    $Y_i = \alpha + \beta_1 X_i + \beta_2(1) + \beta_3 X_i(1) + u_i$
  - Model for non-news watcher ($D_i = 0$):
    $Y_i = \alpha + \beta_1 X_i + \beta_2(0) + \beta_3 X_i(0) + u_i$
- Is there a significant difference in the effect of income by news consumption? Depends on inference for β_3.

General tips on interaction terms

1. **Always** include the main effects for any variable included in an interaction.
2. To know the effect of one variable given the value of another, substitute the value of the moderator variable into your sample regression function.
3. Plots, plots, plots!

General tips on interaction terms

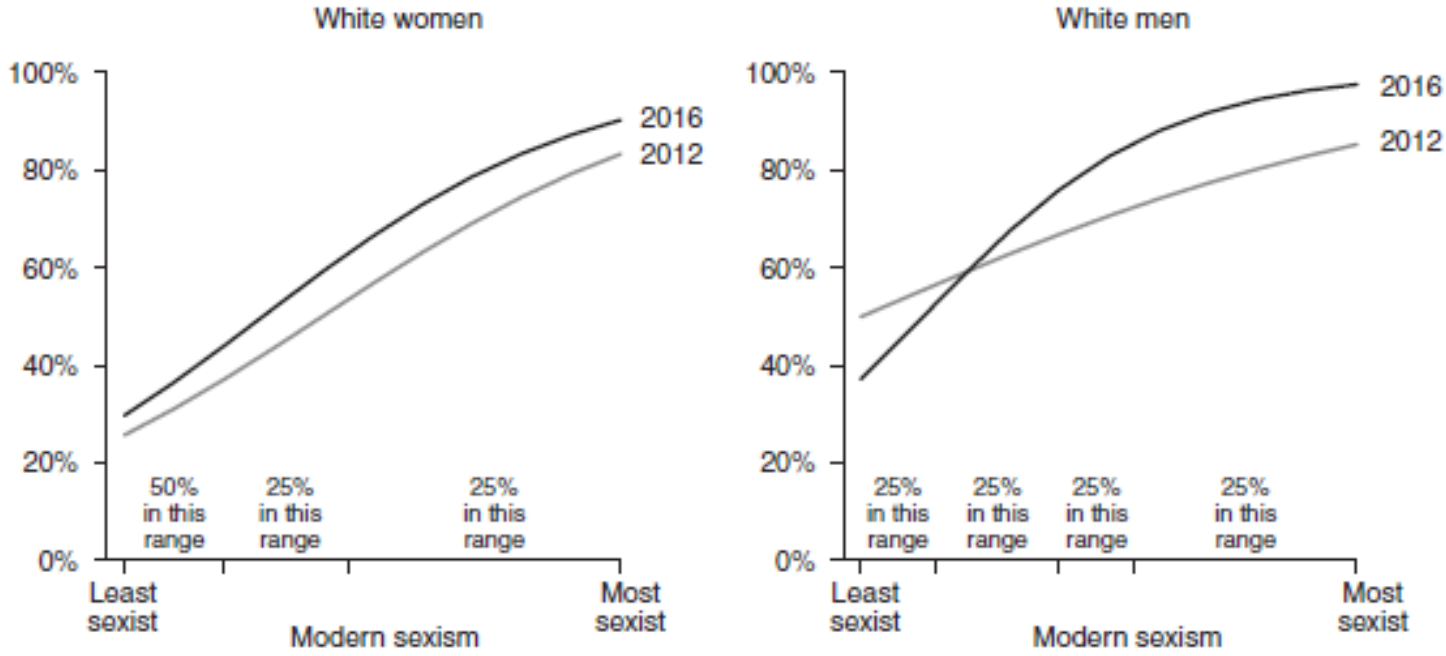
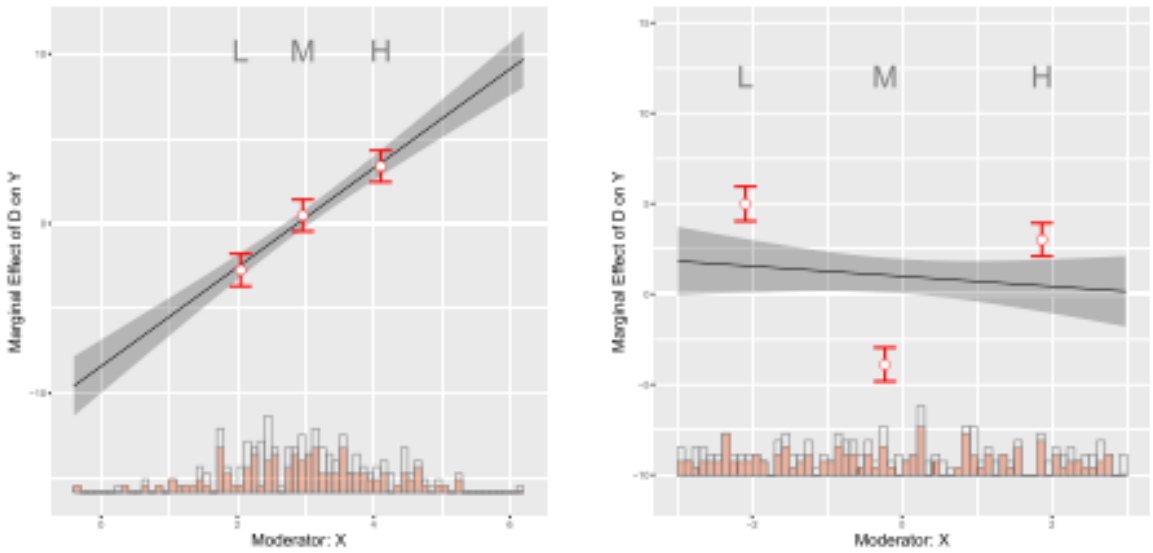


Figure 8.11.
Whites' sexism and likelihood of voting for the Republican presidential candidate. Findings based on statistical models that also account for party identification, self-reported ideology, and attitudes toward African Americans.
Source: 2012–16 VOTER Survey.

General tips on interaction terms

4. Be **extremely** cautious when interacting continuous variables (see Hainmueller et al. 2019 article).



(a) Linear Marginal Effect (b) Nonlinear Marginal Effect

Figure 2. Conditional marginal effects from binning estimator: simulated samples. *Note:* The above plots show the estimated marginal effects using both the conventional linear interaction model and the binning estimator: (a) when the true marginal effect is linear; (b) when the true marginal effect is nonlinear (quadratic). In both cases, the treatment variable *D* is dichotomous.

Interaction terms in R

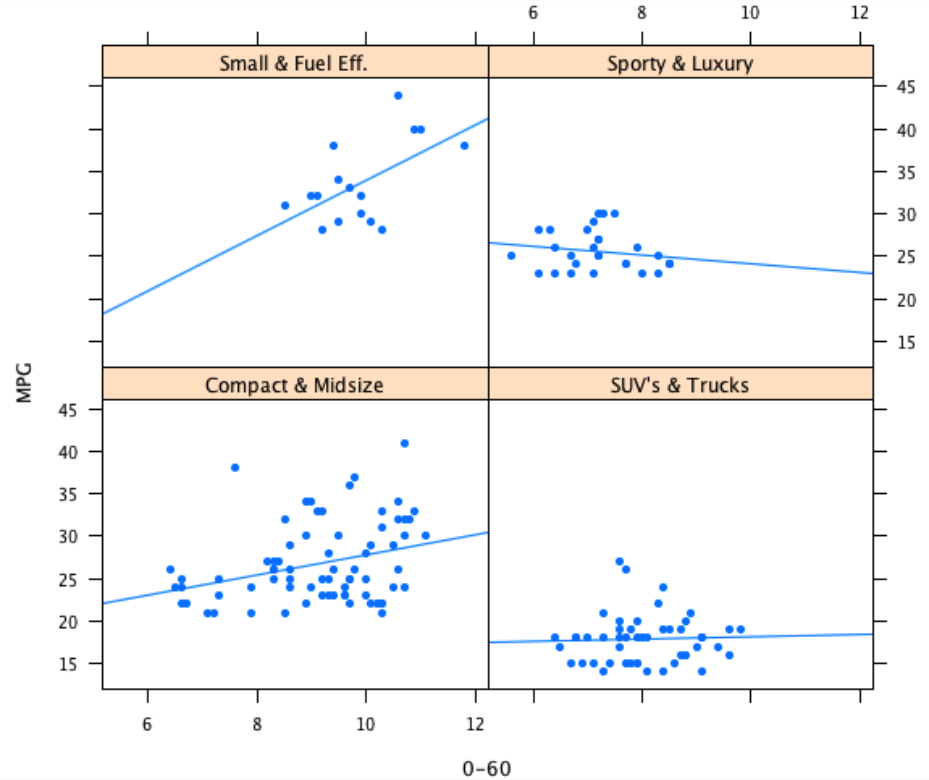
OLS in R

```
library(effects)
ols <- lm(HS.Grad ~ Murder*Frost.abovemean, data=states)
summary(ols)
plot(effect("Murder*Frost.abovemean", ols,
xlevels=list(Frost.abovemean=c(0,1))))
```

Interpretation?

Conditioning plots (or coplots)

- Simple idea: separate scatterplots between *X* and *Y* for each level of a conditioning variable(s).



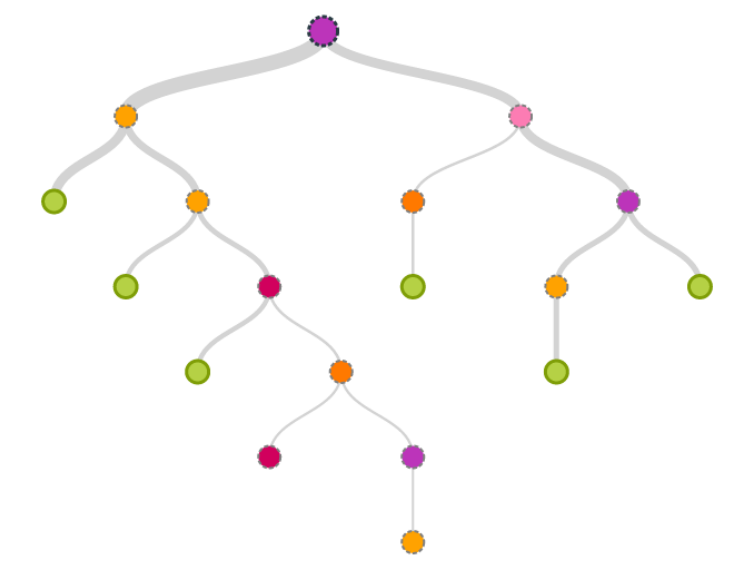
Conditioning plots (or coplots)

Code

```
library(car)
library(dplyr)
states <- as.data.frame(state.x77)
names(states) <- make.names(names(states), unique = TRUE)
states$Frost.quartile <- ntile(states$Frost, 4)
library(lattice)
xyplot(Life.Exp ~ Income | factor(Frost.quartile), data=states,
type=c("p","smooth"))
```

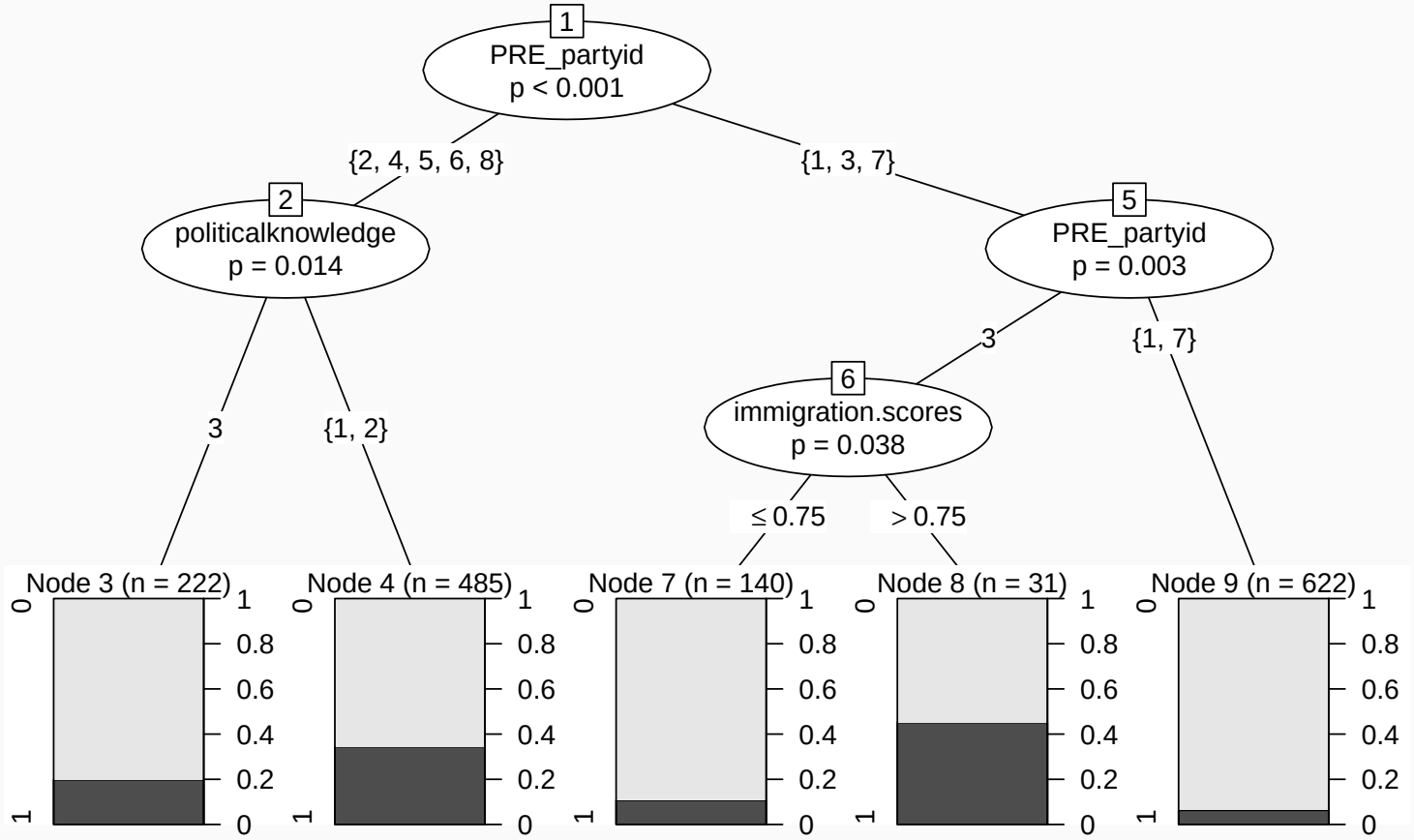
Tree-Based Methods

Decision Trees



- Decision trees use splitting rules to stratify or segment the predictor space into a number of simple regions, either for regression or classification problems.
- We can present these rules in tree form (indeed, doing so helps facilitate interpretation).

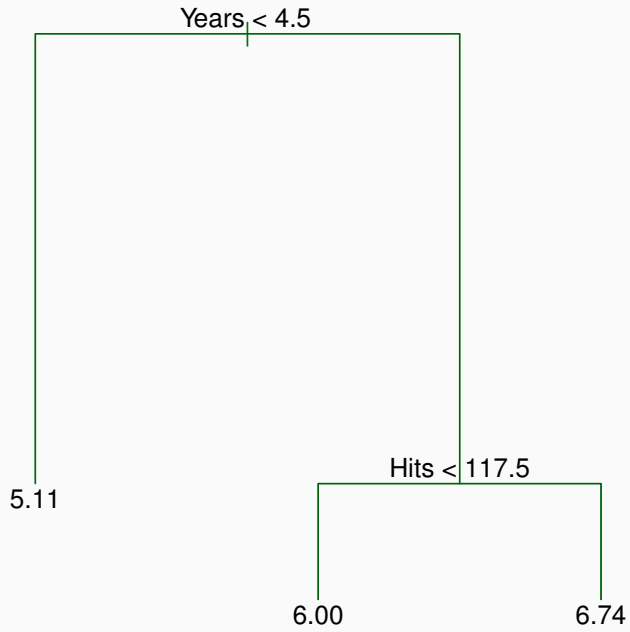
Decision Tree Example



Single vs. Multiple Trees

- 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.
- Hence we also discuss aggregation methods such as random forests and boosting. These methods grow multiple trees which are then combined to yield a single consensus prediction.
- Combining a large number of trees often reuslts in **dramatic improvements** in prediction accuracy, at the expense of some loss interpretation.
- We'll focus on *single* regression and classification trees for the moment.

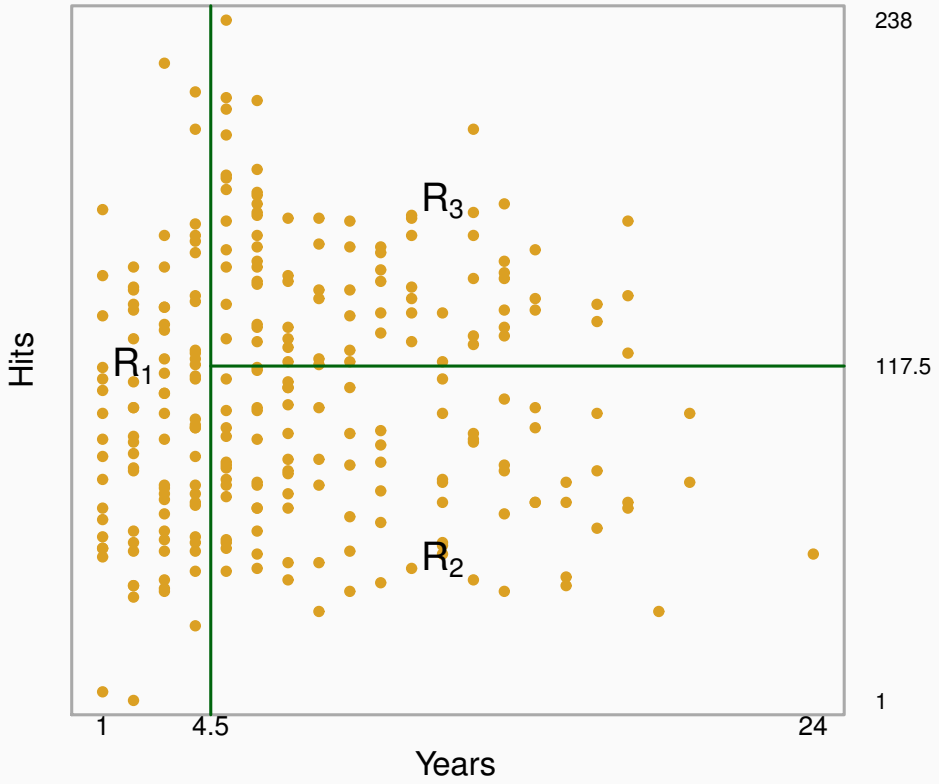
Regression Trees



- This is an example of a simple regression tree for the `Hitters` data. It predicts the (logged) salary of a baseball player using the number of years that he has played in the major leagues and the number of hits that he made in the previous year.
- This tree has two **internal nodes** (the rules) and three **terminal nodes** or **leaves** (the mean response for observations in the node).

Regression Trees

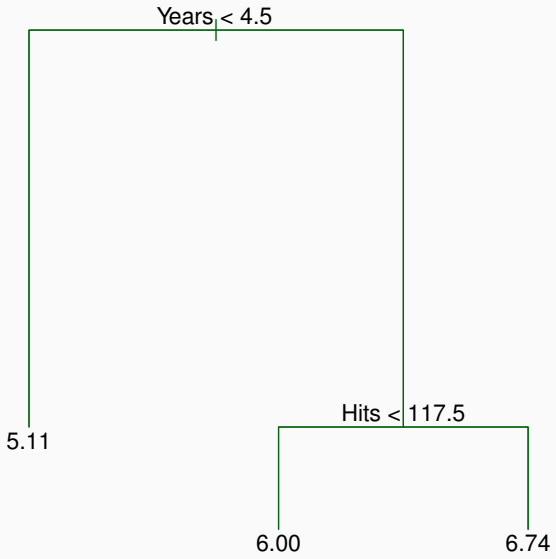
We can present the same model in terms of how it partitions the predictor space into regions R:



Some Quick Notes

- Note that the regions (R_1, R_2, R_3) are identical to the terminal nodes/leaves: they both define rules for outputting predictions.
- Also note that trees are drawn upside down (leaves on the bottom, not the top).

Interpretation



- Years is the most important factor in determining Salary, and players with less experience earn lower salaries than more experienced players.
- Given that a player is less experienced, the number of Hits that he made in the previous year seems to play little role in his Salary.
- But among players who have been in the major leagues for five or more years, the number of Hits made in the previous year does affect Salary, and players who made more Hits last year tend to have higher salaries.

The Tree-Building Process

How did we get here (that is, decide on which variables to split, and how?) Let's detail the entire process:

- We divide the predictor space–that is, the set of possible values for $X_1, X_2, \ldots, X_p$–into J distinct and non-overlapping regions, $R_1, R_2, \ldots, R_J$.
- For every observation that falls into the region R_j, we make the same prediction, which is simply the mean of the response values for the training observations in R_j.

The Tree-Building Process

- In theory, the regions could have any shape. However, we choose to divide the predictor space into P-dimensional rectangles for simplicity and for ease of interpretation of the resulting predictive model.
- The goal is to find regions $R_1, \ldots, R_J$ that minimize the RSS, given by:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2 \tag{1}$$

where $\hat{y}_{R_j}$ is the mean response for the training observations within the jth region.

The Tree-Building Process

- Unfortunately, it is computationally infeasible to consider every possible partition of the feature space into J boxes.
- For this reason, we take a top-down, greedy approach that is known as recursive binary splitting.
- The approach is **top-down** because it begins at the top of the tree and then successively splits the predictor space; each split is indicated via two new branches further down on the tree.
- It is **greedy** because at each step of the tree-building process, the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.

Steps of the Tree-Building Process

1. We first select the predictor X_j and the cutpoint s such that splitting the predictor space into the regions $\{X|X_j < s\}$ and $\{X|Xj \geq s\}$ leads to the greatest possible reduction in the loss function (e.g., RSS).
2. We then repeat the process, looking for the best predictor and best cutpoint in order to split the data further so as to minimize the RSS within each of the resulting regions.
3. However, this time, instead of splitting the entire predictor space, we split one of the two previously identified regions. We now have three regions.
4. Again, we look to split one of these three regions further, so as to minimize the RSS. The process continues until a stopping criterion is reached (e.g., no region contains more than five observations, or we've reached maximum **depth**).

Predictions from the Regression Tree

Predictions from single regression trees are simple: we simply pass testing observations through the tree and use the mean of the training observations in the terminal node as the prediction.

Pruning to Prevent Overfitting

- We again confront the bias-variance tradeoff.
- A smaller tree with fewer splits (that is, fewer regions) might lead to lower variance and better interpretation at the cost of a little bias.
- One possible alternative is to grow the tree only so long as the decrease in the RSS due to each split exceeds some (high) threshold.
- This strategy will result in smaller trees, but is itself problematic: a seemingly useless split early on in the tree might be followed by a very good split (large reduction in RSS) later on.

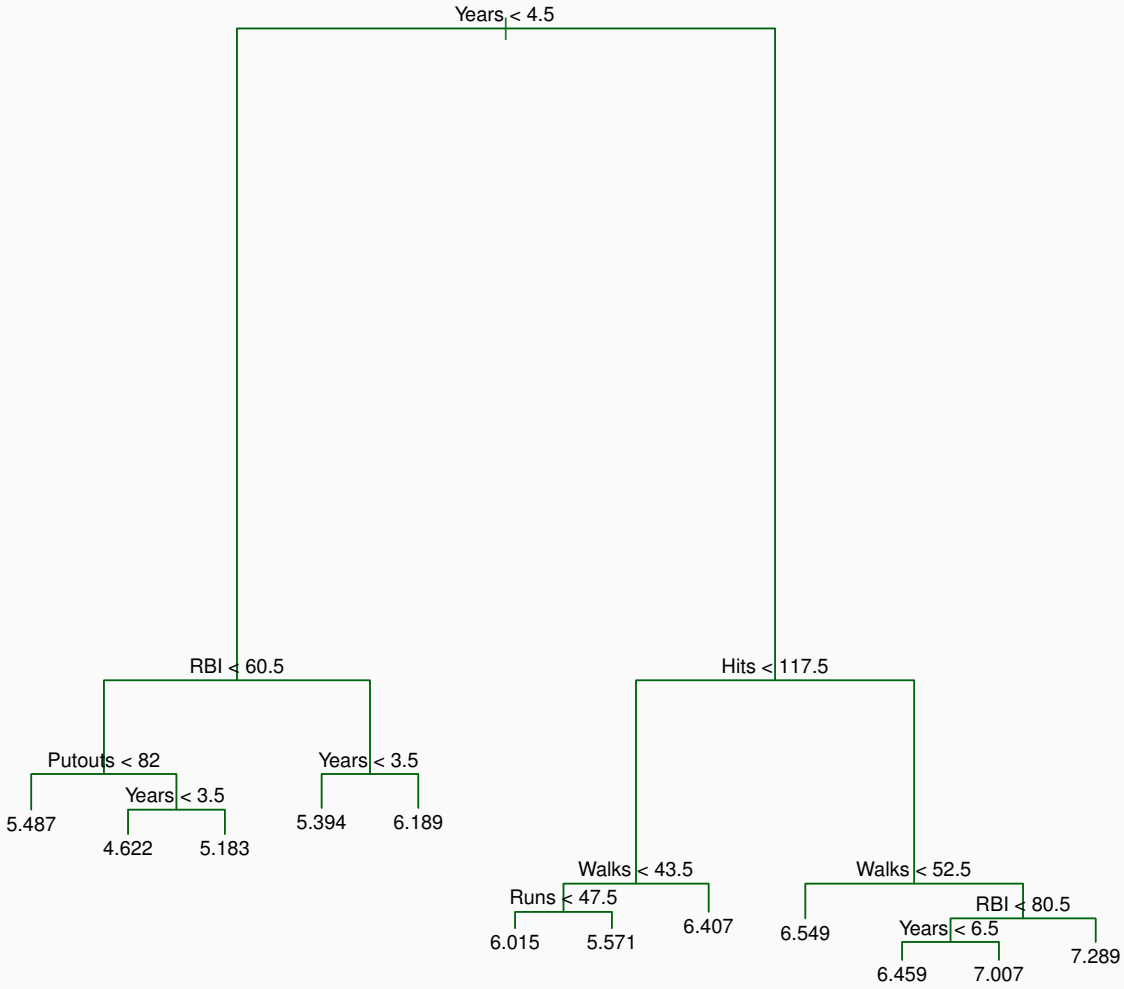
Pruning to Prevent Overfitting

- A better strategy is to grow a very large tree (call it T_0), and then prune it back in order to obtain a subtree–this is known as **cost complexity pruning**.
- We consider a sequence of trees indexed by a nonnegative tuning parameter α. For each value of α there corresponds a subtree $T \subset T_0$ that minimizes:

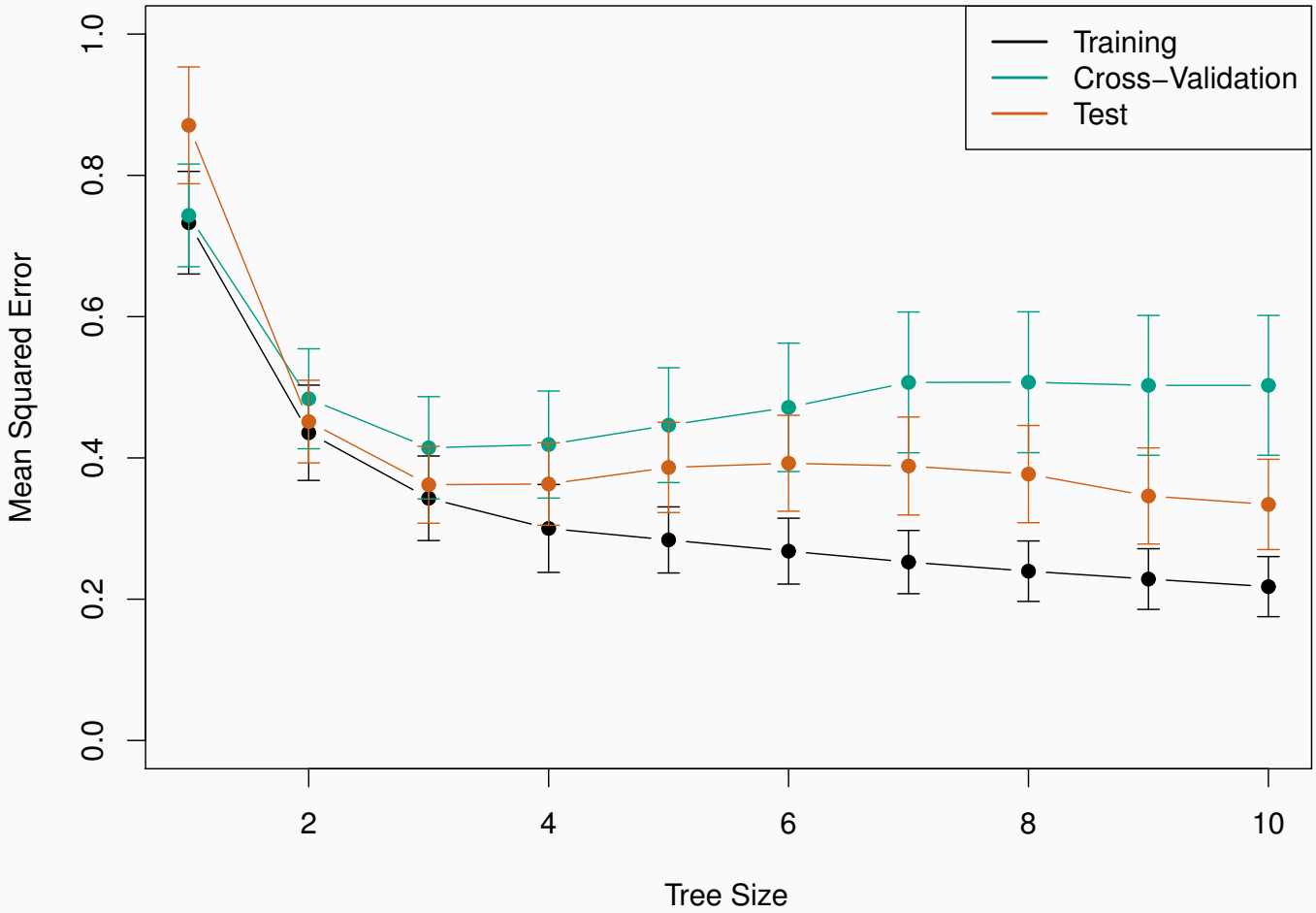
$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T| \tag{2}$$

where $|T|$ indicates the number of terminal nodes of the tree T, R_m is the region corresponding to the mth terminal node, and $\hat{y}_{R_m}$ is the mean of the training observations in R_m. The complexity parameter α is selected using cross-validation.

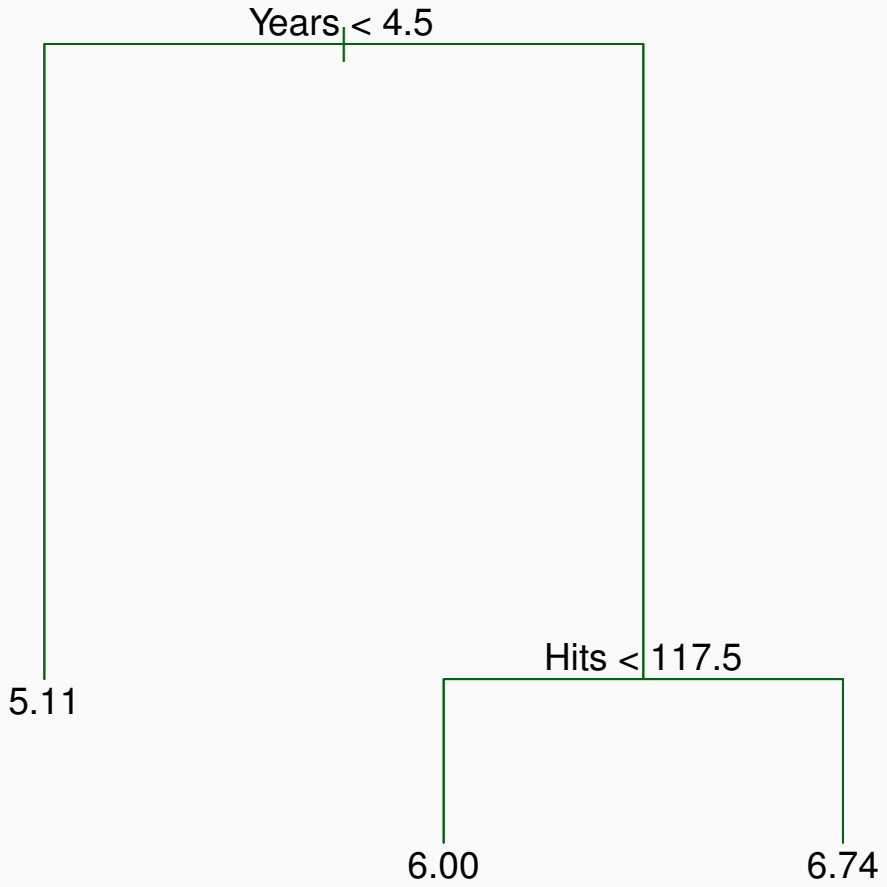
An Unpruned Tree from the `Hitters` Data



CV Error Minimized by a Tree Size of 3



Hence, we Prune our Tree Back to Size Three



Classification Trees

- Classification trees are very similar to regression trees, except that we are of course using it to predict a qualitative response rather than a quantitative one.
- For a classification tree, we predict that each observation belongs to the *most commonly occurring class* of training observations in the region to which it belongs.

Classification Trees: Error Rates

We also need to switch from RSS to an appropriate loss function. We usually use an information-based statistic, commonly one of these two options:

1. **Gini impurity**: A measure of node *heterogeneity*, with small values indicating that a node contains mostly observation from a single class. Let $\hat{p}_{mk}$ represent the proportion of training observations in the mth node that are from the kth class:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{3}$$

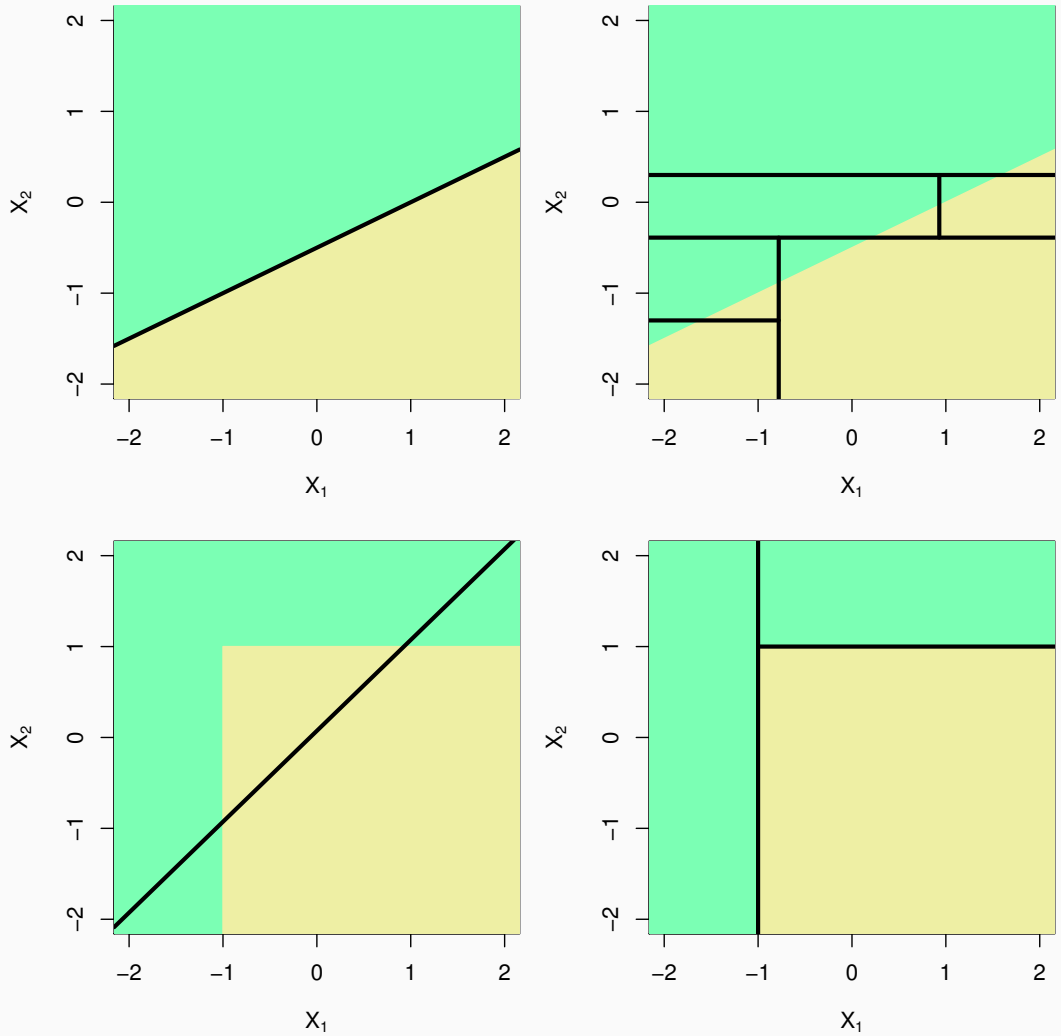
and so G will be small if all of the $\hat{p}_{mk}$ values are close to 0 or 1.

Classification Trees: Error Rates

2. **Deviance**: A measure of cross-entropy. Like the Gini index, it takes on smaller values as the nodes become more “pure” (dominanted by observations of a single class):

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \tag{4}$$

Linear Models (on left) vs. Trees (on right)



Summary of Single Decision Trees

- On the plus side, single decision trees are easy to interpret and explain, are a useful model of choice behavior, and can easily handle qualitative predictors and response variables.
- But, generally have lower levels of predictive accuracy than other models–including standard regression models.
- The good news is that **aggregated** or **ensemble** decision trees have *very high* levels of predictive accuracy–indeed, they are regularly among the best performing predictive methods.