3

Classification and Regression Trees (CART)

3.1 Introduction

Suppose one had a single quantitative response variable and several predictors. There is interest in $\bar{y}|x$. The task is to find the single best predictor. To do this, two kinds of searches are undertaken. First, for each predictor, all possible splits of the predictor values are considered. For example, if the predictor is age, and there are age-values of 21 through 24, all possible splits maintaining order would be 21 versus 22-24, 21-22 versus 23-24, and 21-23 versus 24. If the predictor is marital status with categories never married, married, and divorced, all possible splits would be never married versus married and divorced, married versus never married and divorced, and divorced versus never married and married. For categorical variables, there is no order to maintain.

For each predictor, the best split is selected. The baseline is the sum of squares of the response variable. For each split of a given predictor, a sum of squares is computed within each of the two splits and added. Their sum will be equal to or less than the original sum of squares for the response variable. The "best" split for each predictor is defined as the split that reduces the sum of squares the most.

Second, with the best split of each predictor determined, the best split overall is determined. The same sum of squares criterion is used along with the results from the previous step. By selecting the best split overall, the best predictor by this sum of squares criterion is implicitly chosen.

With the two-step search completed, the winning split is used to subset the data. In other words, the best split for the best predictor defines two subsets. For example, if the best split were to be 21-22 versus 23-24 years of age, all individuals 21-22 would form one subset and all individuals 23-24 would form the other subset.

There are now two partitions of the original data, defined by best split within and between the predictors. Next, the same two-step procedure is applied to each partition separately; the best split within and between predictors

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_3,

for each subset is found. This leads to four partitions of the data, and once again, the two-step search procedure is undertaken separately for each. The process can continue until there is no meaningful reduction in the error sum of squares.

As we show shortly, the result is a recursive partitioning of the data that can be represented within a basis function framework. The basis functions are indicator variables defined by the best splits. With these determined, a regression of the response on the basis functions yields regression coefficients and fit statistics as usual. In practice, there is no need to translate the partitioning into a regression model; the partitioning results stand on their own as a regression analysis. But if one wishes, the recursive partitioning can be seen as a special form of stepwise regression.

The two-step search procedure is easily generalized so that the response variable can be categorical, and in probably its most visible implementation, the recursive partitioning is called Classification and Regression Trees (CART). CART has been in use for about 20 years (Breiman et al., 1984) and remains a popular data analysis tool. In this chapter, CART is examined in considerable depth, not just because it can be of practical value, but because it raises a number of important, broader issues. It also can be a foundation for statistical learning discussed in three subsequent chapters.

The focus is on CART as it has traditionally been implemented. Although there are some recent refinements of CART (Chipman et al., 1998; Loh, 2002; Su et al., 2004), they are peripheral to the aims of this chapter. There are also CART-like procedures such as C5.0 (Quinlan, 1993) with roots in computer science. A discussion of these procedures would take us some distance from the statistical traditions emphasized here, although we later consider a paper by Hothorn and his colleagues (2006) that is somewhat more than a refinement of CART.

Chapter 2 was devoted almost entirely to quantitative response variables. Equal time and more is now given to categorical, and especially binary, response variables. As noted earlier, procedures that assign observations to classes are sometimes called "classifiers." When CART is used with categorical response variables, it is an example of a classifier.

Categorical response variables introduce a number of significant complications that either do not apply to quantitative response variables, or apply only at a much higher level of abstraction. We now need to get this material on the table, in part because it is important for classifiers in addition to CART. We also emphasize the differences among description, estimation, and forecasting. In CART, these are not just differences in how the tools are used, but go to the nuts and bolts of how the procedure performs.

This is a long and somewhat tedious chapter. An effort has been made to include only the material that is really needed. But that's a lot, and it is probably necessary to slog through it all.

3.2 An Overview of Recursive Partitioning with CART

As already noted, classification and regression trees works by a recursive partitioning of the data. Recursive partitioning is a stagewise process that sequentially breaks the data up into smaller and smaller pieces. It is "stagewise," not "stepwise," because earlier stages are not revisited after the results from later stages are known.

Recursive partitioning can be formulated within the basis function framework discussed in Chapter 2. Recall that

$$f(X) = \sum_{j=1}^{p} \sum_{m=1}^{M_j} \beta_{jm} h_{jm}(X). \tag{3.1}$$

Each of the p predictors has its own set of transformations, and all of the transformations for all predictors, each with its own weight β_{jm}, are combined in a linear fashion. Recall also that indicator variables were included as possible transformations. This is a key feature of CART.

To see the relevance of Equation 3.1 for CART, it is necessary to appreciate how CART implements recursive partitioning. The goal of CART's recursive partitioning is to exploit information contained within a set of predictors to create subsets of the data. Each subset is constructed so that the values of the response variable in each are as similar as possible. The process proceeds one partition at a time so that once a partition is constructed, it is not reconsidered when later partitions are defined.

Figure 3.1 is a three-dimensional scatterplot. There is a binary outcome G coded "A" or "B," and predictors x and z. Figure 3.1 is meant to illustrate a simple classification problem as it might be attacked by CART.

The single vertical line at, say, $z = 3$ produces the first partition. The double horizontal line at $x = 6$ produces the second partition. The triple horizontal line at $x = -4$ produces the third partition. CART constructs partitions with a series of straight-line boundaries perpendicular to the axis of the predictor being used.

In this simple illustration, the upper-left partition and the lower-right partition are fully homogeneous. This is good. There remains considerable heterogeneity in the other two partitions and in principle, their partitioning could continue. Figure 3.1 reveals that cases with $z \leq 3$ and $x > 6$ are always "A." Cases with $z > 3$ and $x \leq -4$ are always "B." Thus, we are on our way to describing distribution of the As and Bs conditional upon x and z. The regression framework still applies.

3.2.1 Tree Diagrams

CART output is often shown as an inverted tree. Figure 3.2 is a simple illustration. The full dataset is contained in the root node. The data are then broken into two mutually exclusive pieces. Cases with $x > c_1$ go to the right,

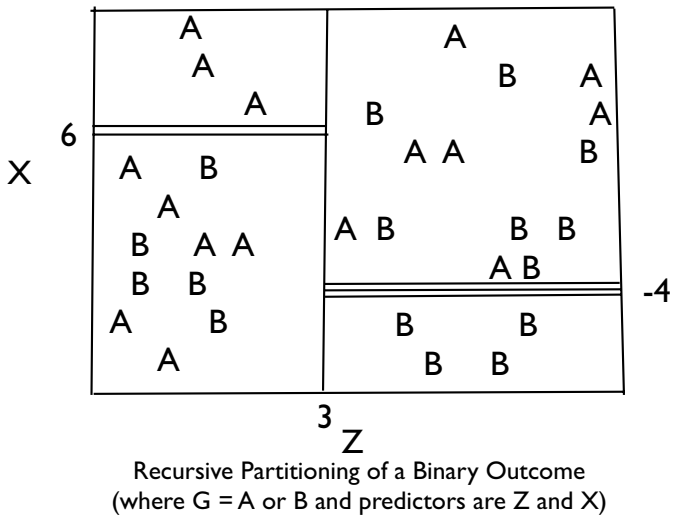

Fig. 3.1. Recursive Partitioning Logic in CART

and cases with $x \leq c_1$ go to the left. The latter are then in terminal node 1, which is not subject to any more subsetting. The former are then in an internal node that can be usefully subdivided further. The cases in the internal node are then partitioned again. Observations with $z > c_2$ go to the right and into terminal node 3. Observations with $z \leq c_2$ go to the left and into terminal node 2.

In this case, all splits beyond the initial split of the root node imply, in regression language, interaction effects. The split imposed at the internal node, for instance, only applies to observations with x-values that are greater than c_1. The impact of z depends on a value of x, which is an interaction effect.

When there is no natural order to a predictor's values, the partitioning criterion selected is usually represented by the name of the variable along with the values that go to the right (or left, depending on the software) side. For example, if ethnicity is a predictor and there are five ethnicities represented by the letters a though e, the software might represent the partitioning criterion for a given split as *ethnicity=ade*. All cases belonging to ethnic groups a,d, and e are being placed in the right-hand partition.

Splits after the initial split do not have to represent interaction effects. If an immediately subsequent partitioning of the data uses the same predictor (with a different breakpoint), the result is an additional step in the step function for that predictor. A more complicated nonlinear function results, but not an

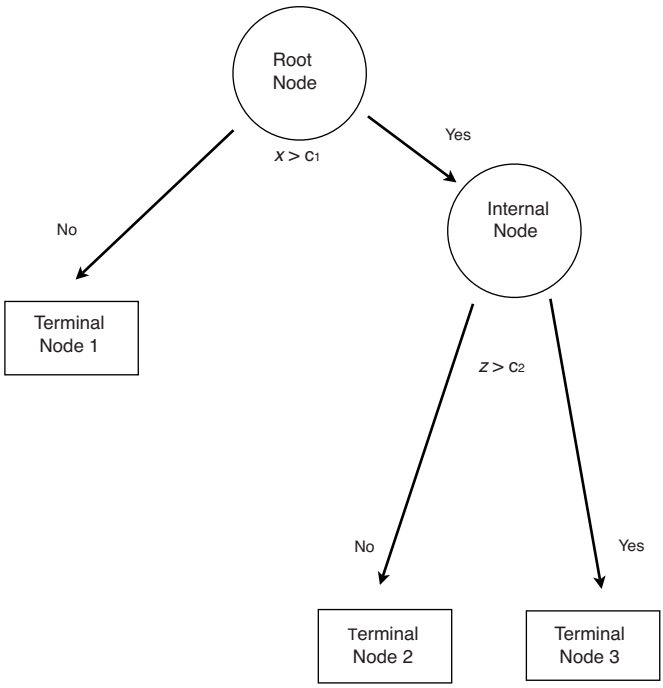

Fig. 3.2. CART tree structure.

interaction effect. In practice, however, most partitions of the data represent interaction effects.

It is easy to show that results such as those shown in Figure 3.2 can be written within the basis function framework of Equation 3.1. One just represents all of the terminal nodes with indicator variables, each of which is a function of one or more predictors (including the constant term). Thus,

$$f(X, Z) = \beta_0 + \beta_1[(I(x \leq c_1)] + \beta_2[I(x > c_1 \ \& \ z \leq c_2)] + \beta_3[I(x > c_1 \ \& \ z > c_2)]. \quad (3.2)$$

One can see again the importance of interaction effects whenever two or more predictors are needed to construct the indicator variable. Interaction effects need to be kept in mind when CART tree diagrams are interpreted.

3.2.2 Classification and Forecasting with CART

There is far more to the output from CART than a tree diagram. Within each of the terminal nodes, the proportion of "successes" and proportion of "failures" are calculated. These conditional proportions can be of significant descriptive interest. For example, if the proportion of successes in terminal node 3 is .70, one can say for cases with $x > c_1$ and $z > c_2$ that the proportion of successes is .70. Analogous statements can be made about the other terminal nodes. Ideally, these proportions will vary substantially, implying that the partitioning is making important distinctions between different kinds of cases. If you know for any given case the value of x and the value of z, it really matters for the proportion of successes.

In addition, the proportions can be used to attach labels to observations. If the majority of observations in a partition are As, all of the observations in that partition might be assigned to class A. If the majority of observations in a partition are Bs, all of the observations in that partition might be assigned to class B. These labels convey what is most typical in a partition and if the observations need to be organized into categories, provide a ready way to determine which observations belong where. When CART is used in this manner, it is being used explicitly as a classifier.

Often, the assigned classes can also be used for forecasting. Suppose one knows that observations with certain values for predictors fall in a particular partition, and that the majority of observations in that partition are, say, of category A. Then, new observations that would fall in that partition, but for which the response is unknown, might be predicted to be A as well.

3.2.3 Confusion Tables

At least as important as the tree diagram is a classification table that cross-tabulates the observed classes and the classes that CART assigns. When the observed classes and the assigned classes come from the data used to build the tree, the table can be used to understand how skillful CART has been in fitting the data. When the observed classes and the assigned classes come from data not used to build the tree, the table can be used to understand how skillful CART has been in forecasting. In either case, the cross-tabulation is often called a "confusion table." We consider confusion tables many times in the pages ahead, but a few details are important to introduce now. The confusion

	Failure Predicted	Success Predicted	Model Error
Failure	a	b	$b/(a+b)$
Success	c	d	$c/(c+d)$
Use Error	$c/(a+c)$	$b/(b+d)$	Overall Error $= \frac{(b+c)}{(a+b+c+d)}$

Table 3.1. A confusion table.

tables are structured to contain a bit more information than is customarily done.

Table 3.1 shows an idealized confusion table. There are two classes for the response variable: success and failure. The letters in the cells of the table are cell counts. For example, the letter "a" is the number of observations falling in the upper-left cell. All of the observations in that cell are characterized by an observed "failure" and a predicted "failure." If the observations are from the data used to build the tree,"predicted" means "assigned." If the observations are from data not used to build the tree, "predicted" means "forecasted." The difference between fitting and forecasting is critical in the next several chapters.

There are generally four assessments that are made from confusion tables.

1. The overall proportion of cases incorrectly classified is an initial way to assess the quality of the fit. The overall proportion of cases incorrectly forecasted is an initial way to assess forecasting skill. Both are simply the number of observations in the off-diagonal divided by the total number of observations. If all of the observations fall on the main diagonal, CART has, by this measure, performed perfectly; none of the observations are either classified or forecasted incorrectly. When no cases fall in the main diagonal, CART is a total failure. All of the observations are either classified or forecasted incorrectly.

   Clearly, a low proportion for this "overall error" is desirable, but how good is good depends on the baseline of classification or forecasting skill when no predictors are used. The real issue is how much better one does once the information in the predictors is exploited. A lot more to is said about this shortly.

2. The overall error neglects that it will often be more important to be accurate for one of the response variable classes than for another. For example, it may be more important to correctly diagnose a fatal illness than to correctly diagnose good health. This is where the row proportions shown in the far right-hand column become critical. For each actual class, the row proportion is the number of observations incorrectly classified or forecasted divided by the total of observations of that class. Each row proportion characterizes errors made by the statistical procedure or model.

When the true class is known, how common is it for the procedure to fail to identify it?

The two kinds of model failures are sometimes called "false positives" and "false negatives." Successes incorrectly called failures are false negatives. Failures incorrectly called successes are false positives. The row proportions that represent the relative frequency of model-generated false negatives and false positives should, ideally, be small. Just as for overall error, the goal is to do better using the information contained in the predictors than could be done ignoring that information. But, the exercise now is done for each row separately. It is common for the model to perform better for one outcome than the other.

3. The column proportions address a somewhat different question. They are the proportion of times when a particular class is assigned or forecasted that the assignment or forecast will be wrong. Whereas the row proportions help evaluate how well CART has performed, the column proportions help evaluate how useful the CART results are likely to be if put to work. The row proportions condition on the true class. The column proportions condition on the class assigned or forecasted. The latter, therefore, convey what would happen if a practitioner used the CART results to classify or forecast. One conditions on either predicted success or on predicted failure from which two different estimates of errors in use can be obtained. Just as for model errors, it is common for the errors in use to differ depending on the outcome. The goal is much the same as for model error: for each column, to be wrong a smaller fraction than if the predictors were ignored.

4. The lower-left cell and the upper-right cell contain, respectively, false negatives and false positives. The ratio of the number of false negatives to the number of false positives shows how the results are trading one kind of error for the other. For example, if b is 5 times larger than c, there are five false positives for every false negative. This means that CART is in this instance treating false negatives as five times more important than false positives; one false negative is "worth" five false positives. Ratios such as this play a key role in our discussion later of how to place costs on false negatives and false positives.

In summary, confusion tables are a critical diagnostic tool. We rely on them in this chapter and all subsequent ones. They also raise some important issues that are salient in the pages ahead.

3.2.4 CART as an Adaptive Nearest Neighbor Method

It can be instructive to think about CART within an adaptive nearest neighbor framework. The partitions shown in Figure 3.1 can be viewed as neighbor-

hoods defined by nearest neighbors. But unlike conventional nearest neighbor methods, CART arrives at those neighborhoods adaptively.

Consider, for example, terminal node 3 in Figure 3.2. Within that node are all observations whose values of x are greater than c_1, and whose values of z are greater than c_2. For these observations, a conditional mean or proportion can be computed. In other words, the nearest neighbors for either of these summary statistics are defined as all cases for which $x > c_1$ and $z > c_2$. Each of the observations for which this condition holds can be used to arrive at a single numerical summary for the response variable.

The neighborhood represented by the terminal nodes is adaptive in three senses. First, information from the response variable is used to determine the neighborhood. Some measure of fit is exploited. Recall that nearest neighbor methods that are not adaptive define the nearest neighbors by their similarity on predictor values alone. Second, because a large number of predictors and break points are examined, a large number of potential neighborhoods are evaluated before an actual neighborhood is defined. Third, the terminal node neighborhoods that result can be defined by different sets of predictors and different sets of cutpoints. Both are determined inductively by the CART algorithm. For example, a given predictor can help define one terminal node, but not another. Even when a given predictor is used to define more than one terminal node, it may enter at a different stage of the partitioning and use a different break point.

The terminal node neighborhoods are constructed sequentially by where in the predictor space some step function for the response is changing most rapidly. This follows from the desire to make the two resulting subsets as homogeneous as possible. Then, because for each split the single best predictor is chosen, each terminal node, and its implied neighborhood, can be defined using a subset of predictors. That is, one need not define nearest neighbors using the entire predictor space. This is in contrast to the multivariate lowess smoother discussed in the last chapter.

But making the subsets as homogeneous as possible does not usually lead to terminal nodes that are completely homogeneous. We show later that to make the terminal nodes homogeneous, very large trees can result with very few cases in each terminal node. Such trees can be very unstable. Thus in the case of binary outcomes, for example, there will usually be a mix of 1s and 0s. One clear consequence is classification error. Whatever the label that is attached to each terminal node, it will be the incorrect label for some observations.

A second consequence is more subtle. Suppose the goal is to estimate the proportion of 1s for all observations with the same set of x-values; one is interested in $\bar{y}|x_0$, where x_0 represents the given set of x-values (e.g., Asian, female, high school students with family incomes of more than $100,000). Unless the terminal node in which all such cases land contains only those observations, there will be other observations with different sets of x-values. When a proportion of 1s is calculated, all of the observations in the node will

be used. Unless all of the proportions of 1s for each of these different sets of x-values are the same as for cases with x_0, the estimated proportion for the observations with x_0 will be biased. For example, Asian and Anglo female high school students with family incomes of more than $100,000 may be pooled in a given terminal node. If the two groups have different proportions of 1s for the response variable, a biased estimate of the proportion of 1s will follow. More is said about this later.

In summary, although smoothers, adaptive nearest neighbor methods, and CART come from very different traditions, they have important similarities. We show additional and helpful connections to other statistical learning procedures in subsequent chapters.

3.2.5 What CART Needs to Do

With the overview of CART completed, we can begin a more detailed discussion. This discussion can be put into a useful context by considering the technical problems CART must solve. There are six such problems.

1. A criterion for the subsetting is required. By what criteria will the partitions be determined?
2. A criterion for the variable selection is required. At each stage, how will the variable used to define the new partition be selected?
3. A way is needed to consider how "good" the tree is. Regressionlike fit measures can be useful, but for classification problems, there will be classification errors with which to contend. Can a tree that makes lots of mistakes in classifying or forecasting cases be "good?"
4. A way is needed to influence the size of the tree so that only useful terminal nodes are constructed. We show that this is related to the bias–variance tradeoff.
5. A way is needed to protect against overfitting. CART is another example of high-powered exploratory data analysis. How can the generalizability of the results be strengthened?
6. Ways are needed to interpret and communicate the results. Tree diagrams are a start, but by themselves neglect some important features of CART results.

The specific solutions to these problems depend in part on whether the response is categorical or quantitative: whether a classification tree or a regression tree, respectively, is desired. Here, we continue with the emphasis on classification problems for categorical response variables and address each of the six problems along the way.

Software can matter too. There are several popular implementations of CART as originally formulated by Breiman and his colleagues (1984). These differ largely in details, but sometimes more fundamental differences arise, especially as new approaches to recursive partitioning are developed. Consistent

with the use of R for all computing in this book, CART implementations in R are the focus. In addition, the focus is on the traditional CART approach because this is the structure on which more recent statistical learning procedures most commonly build.

3.3 Splitting a Node

The first problem that CART needs to solve is how to split each node using information contained in the set of predictors. For an equal interval predictor with m distinct values, there are $m - 1$ splits that maintain the existing ordering of values. So, $m - 1$ splits on that variable need to be evaluated. For example, if there are 50 distinct high school GPA scores possible, there are 49 possible splits that maintain the existing order. However, there are often algorithmic shortcuts that can capitalize, for instance, on ordering the splits by the size of the conditional mean or proportion. The same logic holds for ordinal predictors.

Order does not matter for categorical predictors. Consequently, a categorical variable with k categories has $(2^{k-1} - 1)$ possible splits. For example, if there are five ethnic categories, there are 15 possible splits. Hence, although there are sometimes shortcuts here too, the computational burdens are generally much heavier for categorical variables. There are no restrictions on how a categorical predictor is split.

Starting at the root node, CART evaluates all possible splits of all predictor variables and picks the "best" single split overall. The best split of the variable selected is better than the best split of any other predictor. The data are then partitioned according to that best split. The same process is applied to all subsequent nodes until all cases have been placed in a terminal node. Because the final partitions do not overlap, each case can only be in one terminal node. But how is "best" to be defined? It is common to focus on the "impurity" of a node. The goal is to have as little impurity overall as possible. Consequently, the "best" split is the one that reduces impurity the most. To help simplify the exposition that follows, assume a binary response variable coded 1 or 0. The term "success" for now is used to refer to outcomes coded "1" and "failure" to refer to outcomes coded "0."

Many formal expositions of CART assume the data are a random sample from a well-defined population. Then one can consider, for example, the proportion of times in a limitless number of independent samples that a success or failure would occur. Proportions computed from the sample data can be used as an estimate of these probabilities. When the data are a population, the thought experiment of a limitless number of independent samples no longer applies, and, therefore, there are no probabilities to estimate. The proportions stand on their own as descriptive statistics.

If the data are a nonprobability sample, there is the option of invoking model-based sampling, common in conventional regression (Thompson, 2002:

section 8.3). However, model-based sampling must be used very cautiously. There is no a priori model to determine how uncertainty is introduced. If there is any model at all, it is formulated inductively from the data. Consequently, all of the issues surrounding statistical inference with smoothers resurface. It can be better in practice to let the proportions computed from a nonprobability sample stand on their own as summaries of the data, but not as estimates of anything.

The exposition that follows makes an effort to consistently distinguish between proportions and probabilities. If the goal is description, a focus on proportions is sufficient. If the goal is estimation, then estimation of probabilities necessarily enters.

Suppose for now that the data are a random sample from a well-defined population, and the concept of a probability applies. Consider a given node, designated as node A. The "impurity" of node A is taken to be a nonnegative function of the probability that $y = 1$, written as $p(y = 1|A)$. If A is a terminal node, ideally it should be composed of cases that are all equal to 1 or all equal to 0. Then $p(y = 1|A)$ would be estimated as 1.0 or 0.0. Intuitively, impurity is the smallest it can be. In contrast, if half the cases are equal to 1 and half the cases are equal to 0, the estimated probability is equal to .50. A is the most impure it can be because a given case is as likely to be a 1 as it is a 0.

One can more formally build on these intuitions. Let the impurity of node A be

$$I(A) = \phi[p(y = 1|A)], \tag{3.3}$$

with $\phi \geq 0$, $\phi(p) = \phi(1-p)$, and $\phi(0) = \phi(1) < \phi(p)$. In other words, impurity is nonnegative, and symmetrical with a minimum when A contains all 0s or all 1s, and a maximum when A contains half of each. Note that the use of I in Equation 3.3 for impurity should not be confused with the use of I to represent an indicator variable. The different meanings should be clear in context.

There remains a need to define ϕ. Three definitions have been used in the past: Bayes error, the cross-entropy function, and the Gini index. In order they are:

$$\phi(p) = \min(p, 1 - p); \tag{3.4}$$

$$\phi(p) = -p\ \log(p) - (1 - p)\ \log(1 - p); \tag{3.5}$$

and

$$\phi(p) = p\ (1 - p). \tag{3.6}$$

All three functions for impurity are concave, having minimums at $p = 0$ and $p = 1$ and a maximum at $p = .5$. Entropy and the Gini index are the most commonly used, and generally give very similar results except when there are more than two response categories. Then, there is some reason to favor the Gini index (Breiman et al. 1984: 111). The Gini index is more likely to partition the data so that there is one relatively homogeneous node having

relatively few cases. The other nodes are then relatively heterogeneous and have relatively more cases. For most data analyses, this is a desirable result. Entropy tends to partition the data so that all of the nodes for a given split are about equal in size and homogeneity. This is generally less desirable. But the choice between the two impurity functions can depend on the costs associated with classification errors, which is a topics addressed shortly. indexGini index

One might legitimately wonder why CART does not directly minimize classification error. Direct minimization of overall classification error is discussed in some detail by Breiman and his colleagues (1984: Section 4.1). One serious problem is that there can be several splits for a given stage minimizing classification error. A more subtle problem is that minimizing classification error at each stage has a tendency, like entropy, to produce a tree structure that is often more difficult to interpret. For now, we focus on node impurity as just defined. However, direct minimization of classification error resurfaces as a useful goal when boosting is considered in Chapter 6.

Building on Zhang and Singer (1999; Chapters 2 and 4), a simple example may help to make the discussion of impurity more concrete. For any internal node, we focus on a potential left "daughter" node A_L, and a right "daughter" node A_R. We wish to evaluate the usefulness of a potential partitioning of the data. Table 3.2 provides the information needed. We continue to work with probabilities, although the same practical lessons follow using proportions instead. And with no important loss of generality, the illustration uses entropy as the way impurity is represented.

As before, we let $y = 1$ if there is a success and 0 otherwise. Because the data are a random sample, estimation is a legitimate enterprise; we are not limited to description alone. The estimate of $p(y = 1|A_L)$ is given by $n_{12}/n_{1.}$. Similarly, the estimate $p(y = 1|A_R)$ is given by $n_{22}/n_{2.}$.

	Failure	Success	Total
Left Node: $x \leq c$	n_{11}	n_{12}	$n_{1.}$
Right Node: $x > c$	n_{21}	n_{22}	$n_{2.}$
	$n_{.1}$	$n_{.2}$	$n_{..}$

Table 3.2. Information used to determine the usefulness of a potential split.

It follows that "entropy impurity" for the left daughter is

$$I(A_L) = -\frac{n_{11}}{n_{1.}}\log(\frac{\mathrm{n_{11}}}{\mathrm{n_{1.}}}) - \frac{\mathrm{n_{12}}}{\mathrm{n_{1.}}}\log(\frac{\mathrm{n_{12}}}{\mathrm{n_{1.}}}). \tag{3.7}$$

"Entropy impurity" for the right daughter is

$$I(A_R) = -\frac{n_{21}}{n_{2.}}\log(\frac{\mathrm{n_{21}}}{\mathrm{n_{2.}}}) - \frac{\mathrm{n_{22}}}{\mathrm{n_{2.}}}\log(\frac{\mathrm{n_{22}}}{\mathrm{n_{2.}}}). \tag{3.8}$$

Imagine that for the left daughter there are 300 observations with 100 successes and 200 failures. It follows that the impurity is $-.67(-.40) - .33(-1.11) = .27 + .37 = .64$. Imagine now that for the right daughter there are 100 observations with 45 successes and 55 failures. It follows that this impurity is $-.55(-.60) - .45(-.80) = .33 + .36 = .69$.

To put these numbers in context, it helps to consider the smallest and largest possible values for the impurity. The greatest impurity one could obtain would be for 50% successes and 50% for failures. The computed value for that level of impurity would be .693. For proportions of 1.0 or 0.0, the value of entropy impurity is necessarily 0. In short, the minimum value is 0, and the maximum is a little more than .69. The closer one gets to 50-50, where the impurity is the greatest, the closer one gets to .693. The impurity numbers computed are rather close to this upper bound and reflect, therefore, substantial heterogeneity found in both daughter nodes. It is likely that this split would not be considered to be a very good one.

Once all possible splits across all possible variables are evaluated in this manner, a decision is made about which split to use. The impact of a split is not just a function of the impurity of a node, however. The importance of each node must also be taken into account. It stands to reason that a node in which few cases are likely to fall should be less important than a node in which many cases are likely to fall. In the big picture, the former probably will not matter much, but the latter probably will.

We define the improvement resulting from a split as the impurity of the parent node minus the weighted left and right daughter impurities. If this is a large number, entropy impurity is reduced substantially.

More formally, the benefits of the split s for node A,

$$\Delta I(s, A) = I(A) - p(A_L)I(A_L) - p(A_R)I(A_R), \tag{3.9}$$

where $I(A)$ is the value of the parent impurity, $p(A_R)$ is the probability of a case falling in the right daughter node, $p(A_L)$ is the probability of a case falling in the left daughter node, and the rest is defined as before. The two probabilities can be estimated from the information such as provided in Table 3.2; they are just the marginal proportions $n_{1.}/n_{..}$ and $n_{2.}/n_{..}$.

$\Delta I(s, A)$ is essentially the reduction in the deviance and thus, there is a clear link to the generalized linear model that can prove useful when different fitting procedures are compared. CART finds the best $\Delta I(s, A)$ for each variable. The variable and split with the largest value are then chosen to define the new partition. The same approach is applied to all subsequent nodes.

It makes no difference to the CART algorithm whether the proportions computed are taken at face value as summary statistics, or as estimates of probabilities. The partitions that result are the same. What can differ is whether a given dataset or a random sample is being analyzed. This is up to the user.

The CART algorithm can keep partitioning until there is one case in each node. There is then no impurity whatsoever. Such a tree is called "saturated."

However, well before a tree is saturated, there will usually be far too many terminal nodes to interpret, and the number of cases in each will be quite small. The very small node sizes lead to very unstable results. Small changes in the data can produce trees with rather different structures and interpretations. One option is to prohibit CART from constructing any terminal nodes with sample sizes smaller than some specified value. A second option is considered shortly. And we show in later chapters that there can be ways to work usefully with saturated trees, as long as there is a very large number of them.

3.4 More on Classification

For some applications, the data analysis can stop once all of the cases are assigned to a terminal node. The partitions and the proportions of successes in each are all that matter. For example, very much within a regression framework, it may be of interest to learn which characteristics of students are associated with the estimated probability of dropping out of school. How much higher might the probability be for students whose parents did not graduate from high school themselves, compared to the probability for students whose parents did graduate (Thompson, 2002: Section 8.3)?

But often there is an important additional step. That step is classification. Using the distribution of cases in a given node, the user wants to call all cases in that node the same thing. For example, if the students in a particular terminal node have an estimated probability greater than .50 of dropping out of school, all students in that node might be labeled as high risk, and then be offered special remedial services. The data partitions constructed by CART are now fixed. Classification takes the partitions as given and applies a rule by which all the observations within a given terminal node are assigned to a single class.

Classification raises a number of new issues that revolve around the consequences of classification errors. What happens to students who are really at high risk for dropping out of school but who are not identified as such? What happens to students who are not at high risk for dropping out of school, but who are labeled as high risk? To consider such questions, we need to be a bit more clear on what fitted values are in CART.

3.4.1 Fitted Values and Related Terms

We need to broaden the discussion just a bit to consider some ways in which CART is related some other techniques and to clarify some terms that can be used in more than one way. In particular, the term "fitted values" can have several different meanings.

CART is a method to construct, using a set of predictors, a set of conditional distributions. Interest commonly centers on some measure of location for those conditional distributions. For classification problems, the conditional

proportion is usually the measure. We show later that for regression problems, the measure is usually the conditional mean. And we have already discussed how, using basis functions, explicit links to parametric regression can be made. It follows that most of the issues raised by parametric regression, and most of concepts associated with parametric regression, carry over.

But there are also some ways in which CART's links to parametric regression can be a bit confusing. To begin, one must be clear about the distinction between the value of the response variable associated with each case, and the value of the response variable that CART can assign. The former comes directly from the data themselves and is unrelated to whatever statistical procedures are applied. The latter is an output of CART. One can think of the values assigned to observations by CART as fitted values, much as in conventional regression analysis.

Quantitative Fitted Values

For classification problems, there are two kinds of fitted values. First, each terminal node can be characterized by the proportion of cases for each of the classes. The CART algorithm determines which observations go to which terminal nodes and stops. Within each node, the proportion of observations from each of the classes is then computed. Once these are computed, they can be used to characterize all the observations in a given terminal node.

For example, if the response variable is binary, the proportion of "successes" in a given terminal node can be assigned to each observation in that terminal node. If that proportion is .25, for instance, one can say that for all the cases in that node, the proportion of successes is .25. This is an illustration of description.

Sometimes it may be possible to treat the data either as a random sample from a well-defined population or as a realization of a well-defined stochastic process. Then, the computed proportions for each terminal node can be viewed as estimates of population values or as estimates of parameters associated with the stochastic process. The proportions may then be interpreted as probabilities. The proportion of successes of, say, .25 becomes an estimate of some population value or of some parameter defining the stochastic processes. When it is then assigned to each case in a given terminal node, it may be interpreted as an estimate of the probability of a success for that case.

Qualitative Fitted Values

The second kind of fitted value requires CART to take an additional step: a class must be assigned to each terminal node. As described above, one way this may be accomplished is by a majority vote within each terminal node (or plurality if there are more than two response categories). Then the class assigned to a terminal node is assigned to each observation in that terminal node. The assigned class can be represented by any set of distinct characters,

but for binary response variables, values such as "0" and "1" are common and handy. If, for instance, a given terminal node is assigned a class of "1", all observations in the node are assigned a class of "1."

Much as with the proportions assigned to observations, the classes assigned to observations may be treated as descriptions of the data on hand and if justified, as estimates as well. The class assigned is the estimated class. Even when there is no forecasting involved, the estimated class is often called the "predicted" class.

To summarize, there are in CART two kinds of fitted values used for description: the classes assigned to each terminal node and the proportions of observations that fall into each class. There are also two kinds of fitted values used for estimation: the estimated class and the estimated probability of one class versus another. These distinctions are easy enough to remember and are needed when more advanced procedures are introduced in the next chapter. There are several new ways to think about fitted values and several new kinds as well.

3.4.2 An Example

A key issue for prison administrators is understanding which inmates are likely to place themselves and others in harm's way. Use of narcotics, assaults on prison guards, and homicides are examples. Although such events are relatively rare, they carry very serious consequences. It follows that it would be very useful if such conduct could be anticipated. Then, for the high-risk inmates, preventive measures might be taken. For example, inmates from rival street gangs might be housed in different prisons. A prerequisite, however, is a way to find useful predictors of misconduct in prison.

Using data from the administrative records of a large state prison system, Figure 3.3 shows a classification tree for which inmates engage in some form of reportable misconduct while in prison. A minimum node sample size of 200 was imposed to stabilize the results and for this initial CART example, to keep the diagram very simple. The two predictor variables in Figure 3.3, selected by CART from a larger set of 12 predictors, are defined as follows.

1. term: Nominal sentence length in years. (The nominal sentence is the sentence given by the trial judge. Inmates are often released before their nominal sentence is fully served.)
2. agerec: Age at arrival at the prison reception center in years with a = 16–20, b = 21–26, c = 27–35, and d = 36 or older.

Terminal nodes are labeled "0" if the majority do not engage in misconduct and "1" if the majority do. The numbers below each terminal node show the distribution of no misconduct to misconduct. Thus, for the root node, which contains all of the data before any partitions are constructed, there are 3807 cases with no reported misconduct and 999 cases with reported misconduct.

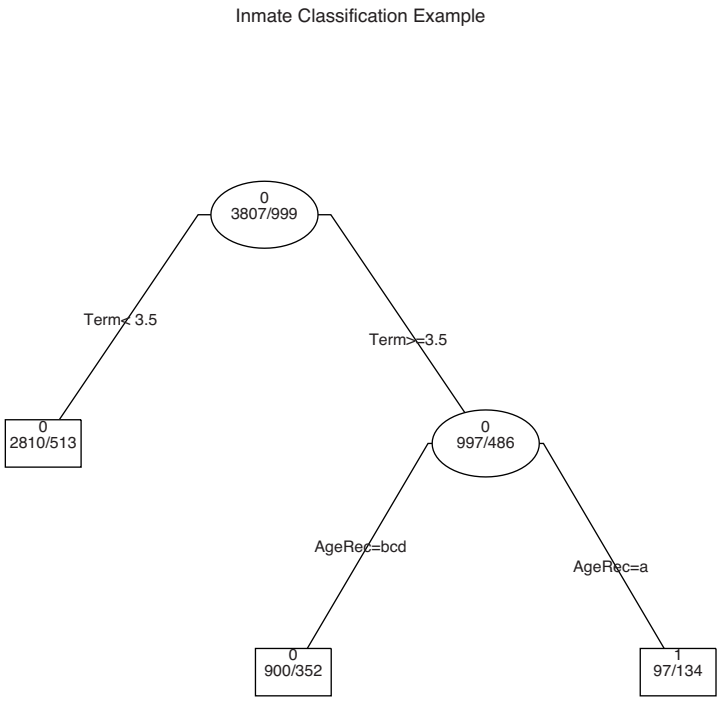

Fig. 3.3. Recursive partitioning of prison data.

Over the 18 months in which the data were collected, about 22% of the inmates had at least one reported misconduct incident.

Figure 3.3 shows a useful level of skill even using just 2 of the 12 predictors. For the far-right terminal node, there are 97 cases with no misconduct and 134 cases with misconduct. In this partition of the data, a little over 58% of the inmates have at least one reported misconduct incident. As a descriptive matter, misconduct for these inmates is about 2.5 times more common than for all inmates on the average and much higher than for any of the other terminal nodes.

All of the cases in the far-right terminal node are inmates who are serving terms of more than 3.5 years, and who arrived at the prison reception center

under the age of 21. They are very young offenders sentenced to long prison terms. It is rather difficult to receive a long prison term at so young an age. Note also that there is an interaction effect between term length and age at reception. Age at reception only has its effect for inmates who are serving terms of 3.5 years or longer.

Generally, crimes committed before the age of 18 do not count when at sentencing an offender's criminal record is considered. A long sentence usually requires several felonies or one very serious felony. So, the inmates in the right terminal node have either been very active or have engaged in very serious crimes.

That such inmates are also difficult in prison would surely be no surprise to criminologists, but Figure 3.3 contains useful information for prison administrators. Very young inmates serving long prison terms may need to be handled somewhat differently from other inmates. One might place them, for instance, in prisons where the day-to-day supervision is more intrusive.

A conventional analysis using logistic regression would likely not have performed as well by comparison. Consider what form a comparable logistic regression would have to take. The far-right terminal node would be a double interaction effect represented by a product variable constructed from two terms. The node just to its left would as well. The far left terminal node is the only main effect. It is unlikely that a researcher would have specified such a model for a logistic regression a priori. When the more likely all-main-effects model was applied to these data, the fit was dramatically worse and led to somewhat different conclusions.

But what if prison administrators want to identify a subset of inmates who are disproportionately likely to engage in misconduct? Then, the classification step is required. For the moment, we apply a simple rule: if the majority of inmates in a given node have reported incidents of prison misconduct, all inmates in that node will be classified as high risk for such behavior. This is consistent with the labels of "1" or "0" in Figure 3.3, and the resulting classifications could lead prison administrators to treat differently the inmates labeled as high risk.

Implicit in the desire to identify high-risk inmates using a set of predictors is to treat the data on hand as a random realization from whatever the stochastic process is that delivers convicted felons to prison. Then, a CART analysis of the data on hand may be used to construct estimates of the likely class for new inmates as they come in the front door and of the probability of misconduct as well. Descriptors are being treated as estimates and estimates can then be used as forecasts.

More details on how this might be done are considered later. But the basic idea is to use the tree diagram. New observations for which the response is unknown would be assigned to a terminal node based on their particular predictor values (e.g., term greater than 3.5 years and age under 21). For each observation, the class previously assigned to each terminal node would be used as the forecasted class for that observation. And for each observation,

the proportions previously used to describe each terminal node would be used as the forecasted misconduct probability for that observation.

But for such forecasts to be fully useful, a way is needed to build in the consequences of the forecasting process. And in fact, CART is making some default decisions about those consequences, which a user of the forecasts may not like. In this example, CART is treating the costs of failing to properly identify high-risk inmates the same as the costs of falsely identifying high-risk inmates. This equivalence may be undesirable, so the costs of misclassifications need further discussion. We turn to that now.

3.5 Classification Errors and Costs

Up to this point, we have proceeded with CART classifying by majority vote. For each terminal node, CART counts the number of cases in one class and the number of cases in the other class, and classifies all cases as the class with the majority of votes. When there are more than two categories, classification is by the category with the greatest number of votes, which then can be just a plurality.

Going back to Figure 3.3, and the terminal node in the lower right hand corner as an illustration, there are 97 votes (i.e., cases) for no misconduct and 134 votes (i.e., cases) for misconduct, so all cases in that node are classified as "misconduct." That implies that 97 cases are misclassified. They are classified as inmates who engaged in misconduct when they actually had not.

The other two terminal nodes in Figure 3.3 would be approached in the same way. In each of these terminal nodes, a majority vote would produce a no misconduct classification. Then, each of the cases for which there actually was misconduct would be misclassified. So, there are 352 classification errors for the terminal node in the middle and 513 classification errors for the far-left terminal node.

Whether Figure 3.3 is satisfactory from a user's point of view, however, depends on more than the number of classification errors. It depends on how the classifications will be used. Looking at the far-right terminal node again, there are 97 "false positives." If the cost of false positives is very high, the results in that node may be unsatisfactory.

Suppose the CART analysis were used for forecasting and that for new inmates classified as high risk, special housing arrangements were desirable. But such housing, which typically requires closer levels of supervision, can be very costly. Moreover, there may be a relatively small number of beds within the prison system that would be appropriate. Yet, the far terminal node in Figure 3.3 implies that for every ten inmates who really might need the special housing, there would be about seven who probably do not. Perhaps it would be better, therefore, if the threshold by which high risk inmates were classified was higher. For instance, rather than a majority vote, a two-thirds vote might be required.

Consider now either one of the other two terminal nodes. All of the new inmates falling in the middle node would not be candidates for special housing. Yet, for every ten inmates appropriately classified as low risk, there would be about four inmates who really were not. Should one of them attack a guard or another inmate, the costs could be very high. Perhaps it would be better if the threshold for high-risk inmates were lower. For instance, rather than a majority vote, a one-third vote might be required.

Thus, there would seem to be conflicting voting rules and no apparent way to reconcile them. Should the classification assigned to new inmates be determined by one-third vote, a majority vote, a two-thirds vote, or something else? And there would seem to be no solution to this problem unless costs are somehow factored in.

One could also have forecasted the probability of misconduct. But that would have implied a different kind of forecasting enterprise. At the level of the individual inmate, the truth that the world could ultimately provide is whether for that inmate there was or was not a reported incident of misconduct. So, that is what needs to be forecasted. And that is the outcome for which the costs are forecasting errors can properly be addressed. There is no observed value of the response at the level of the individual inmate that would make sense coded as a proportion. Consequently, there is no truth to which a forecasted probability can be compared.

In contrast, if the goal were to forecast the proportion of inmates in a particular group (e.g., a particular cell block) who would have a reported incident of misconduct, then the world could in principle generate a true proportion, which might be of interest to forecast. But at the level of the group, determining the conditional proportion is no longer a classification problem, but a regression problem. We consider regression trees later. Suffice it for now to say that the way costs are taken into account in regression trees is quite different.

3.5.1 Default Costs in CART

Without any apparent consideration of costs, CART can make classification decisions about the misconduct of inmates. But in fact, costs are factored in. Table 3.4 shows some results.

	Predict No Misconduct	Predict Misconduct	Model Error
No Misconduct	3710	97	.03
Misconduct	865	134	.88
Use Error	.19	.42	Overall Error = .20

Table 3.3. CART confusion table for forecasting inmate misconduct

As noted earlier, tables of the form of Table 3.4 are sometimes called confusion tables. They can summarize in a particular way, the classification skill (or as we see later, forecasting skill) of a particular classifier. Here, that classifier has been constructed by CART. The table is a cross-tabulation of the actual outcome against the outcome classified. There is a row for each of the two actual outcomes. Each column is for the outcome classified. Correct classifications are in the main diagonal. Misclassifications are in the off-diagonal elements. Thus, we learn that 962 out of 4806 cases (i.e., .20) were incorrectly classified. But, how good this is depends on the baseline.

Had no predictors been used, classification could have been done from the marginal distribution alone. On the one hand, all cases could have been classified as having no reported misconduct. Then, about .22 (999/4806) of the cases would have been incorrectly classified. All cases could have been classified as having reported misconduct. Then, about .78 (3807/4806) of the cases would have been classified incorrectly. Clearly, classifying all cases as if no misconduct had occurred leads to a far lower error proportion and using no predictors, is as good as one can do. Then, it seems that CART is not reducing misclassification all that much (.22 to .20).

However, the overall fit ignores how well CART does when the two response variable categories separated. In this case, the absence of misconduct can be classified with near perfection. In contrast, instances of misconduct are misclassified about 88% of the time. Is this a desirable balance?

The columns in Table 3.4 are also instructive. If the class of no misconduct is assigned, it is wrong for about .19 of the observations. If the class of misconduct is assigned, it is wrong for about .42 of the observations. So, mistakes are relatively more common when misconduct is assigned. Is this desirable?

For both the row and column proportions, a lot depends on the off-diagonal cells. In the process of minimizing the heterogeneity for each partition of the data, CART arrives at a result with 97 instances in which a case is classified as having reported misconduct when that is false. There are also 865 instances in which a case is classified as having no reported misconduct when that is false. Given the concern about inmate misconduct, we call the former false positives and the later false negatives.

From Table 3.4, one can see that there are 8.9 false negatives for every false positive (865/97). The default CART solution for these data trades nearly 9 false negatives for 1 false positive. So, for every inmate who might be incorrectly placed in a high-security setting, for example, there are nearly 9 inmates who might be incorrectly placed in a low-security setting. This implies that whatever the actual costs of false negatives and false positives, false positives are being treated as if they were about 9 times more costly than false negatives.

Several important lessons follow. First, CART (and every other classification procedure for that matter) must introduce costs when a classification decision is made. There is no way to circumvent this. Second, even if the data analyst never considers costs, they are built in. To not consider costs is to

leave the cost determination to the classification algorithm. Third, the way cases are classified or forecasted will vary depending on the costs introduced. As a result, the entire confusion table can change dramatically. The costs used can make a very big difference. Finally, the costs that matter are the costs of classification errors. And as the discussion of Table 3.4 illustrates, it is the ratio of those costs that is critical.

If costs are so important, there is a need to understand how they are incorporated in CART. And this will set the stage for the data analyst to introduce costs explicitly into the analysis. A key step is to appreciate the role of the "prior probabilities" associated with the categorical response variable.

3.5.2 Prior Probabilities and Costs

The marginal distribution of any categorical response variable will have a proportion of the observations in each response category. In our prison example, .22 of the inmates had a reported incident of misconduct, and .88 of the inmates did not. However, before looking at the data, one might already hold strong beliefs from past research or other information about what those marginal proportions should be. For example, the design through which the data were collected may have over sampled inmates reported for misconduct in order to have a sufficient number of them in the study. But for many uses of the results, it would make sense to weight the observations back to the actual proportion of inmates who engage in misconduct. These actual proportions can sometimes be conceptualized as the "prior probabilities" associated with the response variable. The word "prior" comes from Bayesian statistical traditions in which the "prior" refers to the beliefs of the data analyst, before the data are examined, about the probability density or distribution of some parameter.

There has been some recent work within Bayesian traditions that allows for a "pinball prior" for tree size and some features of tree shape (Wu et al., 2007). That is, key features of the tree itself are given a prior probability distribution. The ideas advanced are truly interesting, but in practice it is not clear whether there would be information available to make the pinball prior more than a convenient fiction, and there is almost no experience with this approach to date. It will be some time before we learn whether there is much payoff for real data analysis. Consequently, when the term "prior" is used from here forward, reference is being made, unless otherwise stated, to the prior distribution of the response variable only.

Before we get any farther, we need some additional notation. This notation and the surrounding discussion draws heavily on Therneau and Atkinson (1997). Suppose there are N observations, C classes for the response variable, and K terminal nodes. We define π_i for $i = 1, 2, \ldots,$ as the prior probability of being in class i. For the binary cases, i would equal 1 or 2. As just noted, these marginal probabilities are sometimes called "prior probabilities."

$L(i, j)$ is the loss matrix for incorrectly classifying a case that is really an i as a j. It is this matrix that captures the costs of classification errors. For a binary outcome, the matrix is 2 by 2, where the cost for correct classification is, with no loss of generality, taken to be zero.

We let A be some node in the tree and $\tau(x)$ be the true class for an observation x, where x represents the vector of predictor variable values for that observation. We also let $\tau(A)$ be the class assigned to node A if node A is a terminal node. Finally, N_i and N_A are the number of observations in the sample that are in class i and in node A, respectively, with N_{iA} the number of observations of class i in node A. The following relationships then hold.

1. $P(A)$ is the probability of cases appearing in node A, which is equivalent to $\sum_{i=1}^{C} \pi_i P[x \in A|\tau(x) = i]$. It can be estimated by $\sum_{i=1}^{C} \pi_i(N_{iA}/N_i)$. Note that the prior probabilities figure directly in these calculations and, as a result, can affect the tree structure.
2. Then, $p(i|A)$ is the probability of class i given that a case is in node A, or $P[\tau(x) = i|x \in A]$. It can be estimated by the number of cases of class i in node A, divided by the total number of cases in that node. It is instructive that this probability equals $\pi_i P[x \in A|\tau(x) = i]/P[x \in A]$, which can also be estimated by $\pi_i(N_{iA}/N_i)/\sum_{i=1}^{C} \pi_i(N_{iA}/N_i)$. The priors can make a real difference because the probability of a case with true class i landing in A depends in part on the probability that a case is truly of class i to begin with.
3. $R(A)$ is the "risk" associated with node A, where $\sum_{i=1}^{C} p(i|A)L(i, \tau(A))$, and where $\tau(A)$ is chosen to minimize risk. In other words, the risk associated with node A is for the binary case the probability of a case of type "1" falling in that node times costs that follow, plus the probability of a case of type "2" falling in that node times costs that follow. Risk is, therefore, a function of both the probabilities and the costs. Because the probabilities depend on the prior probabilities, the prior probabilities affect risks.
4. $R(T)$ is the risk of the entire tree T, which equals $\sum_{j=1}^{K} P(A_j)R(A_j)$, where A_j are the terminal nodes of the tree. We are now just adding the total risk associated with each node, weighting by the probability of cases falling in that node. This can also be called the "expected cost" of the entire tree.

And now the punch line. If $L(i, j) = 1$ for all $i \neq j$, and the prior probabilities π_i are taken to be the observed class proportions in the sample, then $p(i|A) = N_{iA}/N_A$, and $R(T)$ is the proportion misclassified. The same applies to the $R(A)$, the risk associated with any particular terminal node. Replacing $L(i, j) = 1$ with $L(i, j) = m$, where m is some constant, just scales up or down the risk by some arbitrary amount and makes no difference to the CART algorithm.

Therefore, if we just let the data determine everything, it is the same as (a) making the costs of all classification errors represented in the loss function

the same and (b) taking the empirical distribution as the appropriate prior distribution. The partitions that follow depend on two conditions. Classification by a majority vote of the cases in each terminal node also depends on these conditions. If either of these conditions is different, it can easily lead to different partitions and different classifications.

We are now ready to revisit the tree diagram in Figure 3.3 and the numbers in Table 3.2. These are the tree and classifications that result when the researcher does not consciously introduce the relative costs of false negatives and false positives. Figure 3.3 assumes equal costs for all classification errors in the loss function and the empirical distribution of the response variable as the prior distribution.

But what does one do if as in Table 3.2 the balance of false negatives to false positives is unsatisfactory? It would seem that the easiest thing to do would be to alter the costs in the loss matrix. That way, one might be able to produce a more acceptable ratio of false negatives to false positives.

However, recall that the risk associated with a node is scaled by the product of the prior probabilities and the entries in the loss matrix. To see the consequences of this, suppose there exist a $\tilde{\pi}$ and an $\tilde{L}$ so that

$$\tilde{\pi}_i \tilde{L}(i,j) = \pi_i L(i,j). \tag{3.10}$$

Then the risk associated with that node are unchanged, and it does not matter what the particular values of $\tilde{\pi}$ and $\tilde{L}$ happen to be as long as the equality holds. This opens the door for lots of possibilities. If one just thinks of the right-hand side as the weight given to the classification errors for class i in a given node, and if more or less weight is desired, one can alter either the priors or the costs or both. In practice, it is less work to alter one of them, and the choice can depend on how the software is written. In the binary response case, if one wanted to alter the weights by altering just the prior distribution to π_i^*, one would use

$$\tilde{\pi}_i^* = \frac{\pi_i L_i^*}{\pi_i L_i^* + \pi_j L_j^*}. \tag{3.11}$$

The index i would take on one value for the no misconduct class (e.g., 1) and another value for the misconduct class (e.g., 2). The values of π_i would be the probabilities associated with the empirical prior distribution. The values of L_i^* would be the new costs. Note that because of the normalization, all that matters in the loss matrix is relative costs. Thus, one just has to know, for example, that the cost of one kind of classification error is three times the cost of another kind of classification error, not their actual costs.

Let's try an example so that the reasoning is clear. Suppose for the prison data one were to let the data determine everything. Then, the empirical prior distribution is about .8 for no misconduct and about .2 for misconduct. The cost of a false negative or a false positive is 1.0.

Suppose we now wanted the cost of a false negative to be twice the cost of a false positive: the 1 to 1 ratio would be 1 to 2. For no misconduct, we let $\pi_1 \times 1.0 = .80 \times 1.0 = .80$. For misconduct, we let $\pi_2 \times 1.0 = .20 \times 2.0 = .40$.

But then we need to normalize these values so that as probabilities they sum to 1.0. Normalizing π_1^*, we compute $(4/5)/(4/5+2/5) = .67$. Normalizing π_2^*, we compute $(2/5)/(4/5 + 2/5) = .33$. So a 1 to 2 cost ratio for a false positive to a false negative can be obtained using for the prior distribution .67 and .33. There is no need to change the values in $L(i, j)$, which in effect, still have diagonal cost elements $L(i \neq j) = 1$. Finally, you can get the same results by using Equation 3.11 with $\pi_1 L_1^* = .80 \times 1.0$ and $\pi_2 L_2^* = .20 \times 2.0$.

It would also be handy if analogous procedures were available for categorical response variables with more than two response categories. However, with more than two response categories, there is likely to be more information in the loss matrix than can be properly captured by a prior distribution. More specifically, for any given observation, the cost of all misclassifications must be the same for there to be a prior distribution that can properly represent the costs of classification errors.

For example, suppose there were three inmate misconduct categories: no misconduct, rule violations, and activities that would be crimes if committed outside of prison. Then, if an inmate actually had no incidents of reported misconduct, the costs of incorrectly placing him or her in either the rule violation category or the crime category would have to be the same. In practice, it is rare that such constraints on the loss matrix would be appropriate. As a result, relative costs would have to be introduced directly using the loss matrix. This presents no problems when the software permits such input. But it is common for there to be no allowance for a loss matrix, especially for the more sophisticated forms of statistical learning to be considered in the next two chapters. Fortunately, there are then other options, many of which are rather clever. We consider these later.

To summarize, when the CART solution is determined solely by the data, the prior distribution is empirically determined, and the costs in the loss matrix of all classification errors are the same. Costs are being assigned even if the data analyst makes no conscious decision about them. Should the balance of false negatives to false positives that results be unsatisfactory, that balance can be changed. Either the costs in the loss matrix can be directly altered, leaving the prior distribution to be empirically determined, or the prior distribution can be altered leaving the default costs untouched. Much of the software currently available makes it easier to change the prior in the binary response case. When there are more than two response categories, it will usually be easier in practice to change the costs in the loss matrix directly.

3.6 Pruning

With the discussion of costs behind us, we can now return to the problem of overly complex trees and what can be done. Recall that setting a minimum

sample size for each terminal node is one strategy. A second strategy to constrain the size of the tree is called "pruning." The pruning process removes undesirable branches by combining nodes that do not reduce heterogeneity sufficiently for the extra complexity added. The process starts at the terminal nodes and works back up the tree until all of the remaining nodes are satisfactory.

Of late, pruning has not gotten a lot of attention. The problem that pruning addresses is very real. But, as CART has become superceded, pruning has become less salient. Consequently, the discussion of pruning is relatively short. The main objective is to highlight some important issues raised in the previous chapter that figure significantly in the pages ahead.

For a tree T, recall that the overall risk is

$$R(T) = \sum_{j=1}^{K} P(A_j)R(A_j). \tag{3.12}$$

This is the sum over all terminal nodes of the risk associated with each node times the probability of cases falling in that node. It might seem that a reasonable pruning strategy would be to simply minimize Equation 3.12. What could be better than that? Unfortunately, that would leave a saturated tree untouched. CART would construct enough terminal nodes so that all were homogeneous, even if that meant one node for each observation. With all terminal nodes homogeneous, the risk associated with each would be zero. The result would be unstable nodes, serious overfitting of the data, and far too much detail to usefully interpret.

The solution is much like what was seen in the previous chapter. A penalty is introduced for complexity, and we are back into the bias–variance tradeoff. With larger trees, there will be fewer classification errors, implying less bias. But larger trees will have terminal nodes with fewer cases in each, which implies greater instability and hence, greater variance. The trick is to find a sensible balance.

To take complexity into account in CART, a popular solution is to define an objective function, called "cost complexity," for pruning that includes an explicit penalty for complexity. The penalty is not based on the number of parameters, as in conventional regression, or on the effective degrees of freedom used, as in smoothing. For CART, the penalty is a function of the number of terminal nodes. More precisely, we try to minimize

$$R_\alpha(T) = R(T) + \alpha|\tilde{T}|. \tag{3.13}$$

R_α has two parts, the total costs of the classification errors for the tree as a whole, and a penalty for complexity. For the latter, $\alpha \geq 0$ is the complexity parameter playing much the same role as λ in regression smoothers. In place of the effective degrees of freedom, $|\tilde{T}|$ is the number of terminal nodes in tree T.

The value of α quantifies the penalty for each additional terminal node. The larger the value of α, the heavier is the penalty for complexity. When $\alpha = 0$, there is no penalty and a saturated tree results. So, α is the means by which the size of the tree can be determined.

Breiman et al. (1984: Section 3.3) prove that for any value of the complexity parameter α, there is a unique smallest subtree of T that minimizes cost complexity. Thus, there cannot be two subtrees of the same size with the same cost complexity. Given α, there is a unique solution.

In many CART implementations, there are ways the software can select a reasonable value for α or for parameters that play the same role (Zhang and Singer, 1999: Section 4.2.3). These defaults are often a good place to start, but will commonly lead to results that are unsatisfactory. The tree selected may make a tradeoff between the variance and the bias that is undesirable for the particular analysis being undertaken. For example, there may be far too much detail to be usefully interpreted. Moreover, overfitting or measurement error can produce trees that make very little subject matter sense.

Alternatively, one can specify by trial and error a value of α that leads to terminal nodes, each with a sufficient number of cases and that can be sensibly interpreted. Interpretation will depend on both the number of terminal nodes and the kinds of cases that fall in each, so a substantial number of different tree models may need to be examined.

In practice, whether one determines tree complexity by using α (or some other complexity parameter), or an explicit argument to the CART procedure determining the minimum terminal node sample size, seems to make little difference. The goal is to construct a useful classification tree. How exactly that is accomplished is less important as long as the steps undertaken and the various results evaluated are recorded so that the work can be replicated.

The major risk from examining a larger number of tree models leads to overfitting. Overfitting is already a potential problem in CART and drawing on information from many trees can only make matters worse. We turn to overfitting and useful responses to it in the next chapter. Suffice it to say, it is always good to have a training dataset and a test dataset.

3.6.1 Impurity Versus $R_\alpha(T)$

At this point, one might wonder why CART does not use Equation 3.13 from the start when a tree is built, instead of some measure of node impurity. $R_\alpha(T)$ would seem to have built in all of the end-user needs very directly.

The rationale for not using a function of classification errors as a fitting criterion is discussed in Breiman et al. (1984: Section 4.1). As a technical matter, there can be at any given node, no single best split. But perhaps a more important reason is that less satisfactory trees can result. Consider two splits. For the first, there are two nodes that are about equally heterogeneous. For the second, one node is far more heterogeneous than the other. Suppose the two splits reduce impurity about the same. Yet, minimizing some function

of classification errors could lead to the first split being chosen even though the second split was preferable. For the second split, the less heterogeneous node might serve as a terminal node, or might readily lead to one. The more heterogeneous node would be more subject to further partitioning. For the first split, both nodes would likely be partitioned substantially further. In general, therefore, more complicated tree structures will follow.

There can be good subject matter reasons as well. Thinking back to the prison example, finding a single node that was filled almost completely with misconduct cases would be a very useful result, even if the other terminal nodes were quite heterogeneous. In contrast, having all of the terminal nodes with roughly the same proportions of misconduct and no misconduct cases, would not be very useful. Using node impurity as a splitting criterion will largely prevent this kind of problem.

3.7 Missing Data

Missing data are a problem for all statistical analyses, and CART is no exception. Unfortunately, missing data are all too common and they create, broadly stated, the same kind of difficulties they create for conventional linear regression. There is the loss of statistical power with the reduction in sample size and real likelihood of bias insofar as the observations lost are not effectively a random sample of the total.

There is one and only one ironclad solution to missing data regardless of the form of data analysis: don't have any. The message is that it pays to invest heavily in the data collection so that missing data do not materialize or are very rare. There are alternatives to be sure, but all are risky.

A general discussion of missing data is beyond the scope of this book, and excellent treatments are easily found (Little and Rubin, 2002). But it is important to consider how missing data can affect CART and what some of the more common responses are.

If the data are really missing "completely at random," the only loss is statistical power. By "missing completely at random" one means that the mechanism by which the data are lost is equivalent to simple random sampling. And if the number of cases lost is not large, the reduction in power is not likely to matter much. It cannot be overemphasized, however, that the burden is on the researcher to make a convincing argument that the data are missing completely at random. Proceeding simply "as if" this were true is a ruse. The results are then conditional upon the missing completely at random assumption, and may be of little interest unless the credibility of that assumption can be determined.

A fallback position is that the data are missing "conditionally at random." One can subset the data based on the values of observable variables so that for each such subset, the data are missing completely at random. By "missing conditionally at random," one means that the mechanism by which the data

are lost is equivalent to stratified random sampling. If this assumption is correct, at least to a reasonable approximation, the analysis can be conducted separately for each of the subsets and the results pooled. But again, assumptions made about the manner in which the data are missing must be argued convincingly.

If either of these assumptions can be justified, it will sometimes useful to impute the values of the missing data. It is rarely sensible to impute missing values for the response variable. One would usually exploit information in the predictors and in so doing, the relationship between the response and the predictors can be fundamentally altered; one builds in a new relationship between Y and X. But sometimes it can be helpful to impute missing data for predictors.

For example, suppose age is a predictor. Then, one might compute the mean value of age over all of the data available and use that value of age when age is missing. If age is related to other predictors, one can use conditional means for the missing data (i.e. conditioning on the values of those predictors) as the imputed values for the missing data. For example, if age is related to education, one can impute the value of age based on whether a person has a college degree. There would be one imputed value for age for those with no college degree and another imputed value for age for those with a college degree.

The key problem with such imputation procedures is when the data are ultimately analyzed, using the real data and the imputed data, the statistical procedures cannot tell which is which and necessarily treat all of the observations alike. At the very least, therefore, it is likely that estimates of the uncertainty in the results will be wrong.

In particular, the imputed values come with sampling error, and this source of uncertainty will be overlooked. Another difficulty is that the imputed values, which are just fitted values, will have less variability than the original variable itself. Consequently, the data analyzed will be too homogeneous. Together, these two features of imputed values can seriously undermine statistical inference.

There are a number of very interesting procedures that attempt to get the statistical inference right when some of the data are imputed. They need not trouble us here. We will focus on how missing data can be addressed within CART.

3.7.1 Missing Data with CART

If data are missing for the response variable, the only viable strategy is "listwise deletion." Observations with missing data on the response variable are dropped totally from the analysis. If the data are missing completely at random, the main loss is statistical power. If not, bias of unknown size and direction can be introduced.

When the data are missing for one or more predictors, there are more options. Listwise deletion remains a possible choice, especially if there are not a lot of missing data (e.g., less than 5% of the total number of observations) Listwise deletion is not fancy, but it is also easy to implement and understand.

A second option is to impute the data outside of CART itself. To take a simple illustration, one might employ conventional regression in which for the complete data a predictor with the missing data is regressed on other predictors with which it is likely to be related. The resulting regression equation can then be used to impute what the missing values might be.

For example, suppose that for employed individuals there are some missing data for income. But income is strongly related to education, age, and occupation. For the observations with no missing data, income is regressed on education, age, and occupation. Then, for the observations that have missing income data, the values for the three predictors are inserted into the estimated regression equation. Predicted values are computed, which are used to fill in the holes in the income data.

However, even if reasonably unbiased estimates can be constructed, this strategy ignores the reduced variability of the predicted values and treats the imputed values as fixed. One response is to impute several values for each observation drawing at random, in effect, from the conditional distributions implied by the regression equation. It is then possible to get a better handle on the uncertainty associated with the imputed values. Little and Rubin (2002) is the canonical reference. An application to CART can be found in a PhD dissertation by He (2006).

A third option is to address the missing data problems for predictors within CART itself. There are a number of ways this might be done. We consider here one of the better approaches, and the one available with rpart() in R.

The first place where missing data will matter is when a split is chosen. Recall that

$$\Delta I(s, A) = I(A) - p(A_L)I(A_L) - p(A_R)I(A_R), \tag{3.14}$$

where $I(A)$ is the value of the parent impurity, $p(A_R)$ is the probability of a case falling in the right daughter node, $p(A_L)$ is the probability of a case falling in the left daughter node, $I(A_R)$ is the impurity of the right daughter node, and $I(A_L)$ is the impurity of the left daughter node. CART tries to find the predictor and the split for which $\Delta I(s, A)$ is as large as possible.

Consider the leading term on the right-hand side. One can calculate its value without any predictors and so, there are no missing values to worry about. However, to construct the two daughter nodes, predictors are required. Each predictor is evaluated as usual, but using only the predictor values that are not missing. That is, $I(A_R)$ and $I(A_L)$ are computed for each optimal split for each predictor using only the data available. The associated probabilities $p(A_R)$ and $p(A_L)$ are re-estimated for each predictor based on the data actually present. This approach is undertaken with the equivalent of pairwise

deletion; calculations are undertaken with the complete data available for that operation alone.

But determining the split is only half the story. Now, observations have to be assigned to one of the two daughter nodes. How can this be done if the predictor values needed are missing? CART employs a sort of "CART-lite" to impute those missing values by exploiting "surrogate variables."

Suppose there are ten other predictors $x_1 - x_{10}$ that are to be included in the CART analysis, and suppose there are missing observations for x_1 only, which happens to be the predictor chosen to define the split. The split necessarily defines two categories for x_1.

The predictor x_1 now becomes a binary response variable with the two classes determined by the split. CART is applied with x_1 as the response and $x_2 - x_{10}$ as potential splitting variables. Only one partitioning is allowed; a full tree is not constructed. The nine predictors are then ranked by the proportion of cases in x_1 that are misclassified. Predictors that do not do substantially better than the marginal distribution of x_1 are dropped from further consideration.

The variable with the lowest classification error for x_1 is used in place of x_1 to assign cases to one of the two daughter nodes when the observations on x_1 are missing. That is, the predicted classes for x_1 are used when the actual classes for x_1 are missing. If there are missing data for the highest ranked predictor of x_1, the second highest predictor is used instead. If there are missing data for the second highest ranked predictor of x_1, the third highest ranked predictor is used instead, and so on. If each of the variables $x_2 - x_{10}$ have missing data, the marginal distribution of the x_1 split is used. For example, if the split is defined so that $x_1 < c$ sends observations to the left and $x_1 \geq c$ sends cases to the right, cases with data missing on x_1, which have no surrogate to use instead, are placed along with the majority of cases.

This is a reasonable, but ad hoc, response to missing data. One can think of alternatives that might perform better. But the greatest risk is that if there are lots of missing data and the surrogate variables are used, the correspondence between the results and the data, had they been complete, can become very tenuous. In practice, the data will rarely be missing "completely at random" or even "conditionally at random." Then, if too many observations are manufactured, rather than collected, a new kind of generalization error will be introduced. The irony is that imputation can fail just when it is needed the most.

Perhaps the best advice is to avoid the use of surrogate variables. The temptations for misuse are great, and there is no clear missing data threshold beyond which imputation is likely to produce misleading results. Imputation of the missing values for the predictors will usually be a software option, not a requirement. (But check what the default is.)

Alternatively, one should at least look carefully at the results with and without using surrogates. Results that are substantially different need to be reported to whomever is going to use the findings. There may then be a way to

choose on subject matter grounds which results are more sensible. Sometimes neither set will be sensible, which takes us back to where we began. Great efforts should be made to avoid missing data.

There is one situation, however, in which using surrogate variables is probably necessary. As becomes more clear in the pages ahead, a number of statistical difficulties can follow when the response variable is highly skewed. The danger with missing data is that the skewing can be made worse. One may then have little choice but to impute the missing data.

3.8 Statistical Inference with CART

An initial question for statistical inference is what features of a CART model might be of interest. To date, attention has centered on an overall assessment of the CART model, and by implication, some measure of fit quality. The enterprise is model selection.

But just as with smoothers, a key issue that must be addressed before statistical inference with CART is considered is whether estimation is a reasonable activity to begin with. As before, there are three scenarios.

1. There is assumed to be a $f(X)$, and the data are a random sample from a well-defined population or a random realization from a well-defined stochastic process. Estimation is worth a good hard look and so are ways to represent uncertainty.
2. There is no $f(X)$ assumed, but the intent is to construct a best guess of the values of a set of conditional proportions in a population or as features of a stochastic process. Estimation is again in play, at least in principle. Ways to represent uncertainty are as well.
3. The sole goal is description of the data on hand. Estimation is not relevant even in principle.

We begin with the first case: there is a $f(X)$ and the data were generated in a manner required for statistical inference. For this first case, statistical inference can be problematic in CART. Perhaps the most obvious reason is the need to assume negligible bias in the results. Mentioned earlier in this chapter was the likely bias in the estimated proportions about which more is said later. More important in practice is the absence of some key predictors and/or substantial measurement error in those that are available.

There are more subtle difficulties as well. For binary response variables, it might seem natural to use the deviance (or a good approximation) as a measure of fit quality and then compare two CART models using a likelihood ratio test. Recall, two models are required, the smaller one nested within the larger one. The smaller model is the model under the null hypothesis. For both, the deviance is computed. Then, the difference in the two deviances has a χ^2 distribution with degrees of freedom equal to the degrees of freedom for the smaller model subtracted from the degrees of freedom for the larger

model. When quantitative response variables for CART are discussed below, an F-test is the natural procedure. Although one can compute the deviance for both models, it would be rare to find one CART model nested within another. Therefore the very logic of the comparison is undermined, and it would also be possible to have two models with different deviances that used the same degrees of freedom.

But what are the degrees of freedom for a CART model? For a parametric regression with p regression coefficients, $p + 1$ is the number of degrees of freedom used up. The degrees of freedom remaining is $N - (p+1)$. For CART, sometimes the number of terminal nodes plus one is assumed to play the same role as $p + 1$ for a parametric regression. This is because the CART results can be re-expressed as a regression model with an indicator variable for each terminal node. However, using the number of terminal nodes to arrive at the degrees of freedom lost fails to take into account all of the searching done as the tree is grown. Many more degrees of freedom are actually used up. Moreover, it is not clear how to make appropriate adjustments, although some simulation results suggest the true degrees of freedom used up is between 5 and 10 degrees of freedom per split of the data (Hastie et al., 2001: 297). An additional complication is that often many trees are examined as the output is "tuned" to the particular needs of the analyst. Whatever the number of degrees of freedom lost when the tree is grown, they will not include the degrees of freedom lost growing prior trees.

Confidence intervals suffer from similar difficulties. The negligible-bias assumption remains an important hurdle, and one needs a value for the degrees of freedom in order to estimate of any standard errors. Thus, even obtaining an appropriate estimate of the point-by-point standard error of the fitted values is tricky.

The degrees of freedom problems spill over into the fit statistics that can be used instead of tests for model evaluation and selection. Among the most common goodness-of-fit measures used with CART are the AIC and the BIC. For a binary response variable,

$$\text{AIC} = D + 2p; \tag{3.15}$$

and

$$\text{BIC} = D + \log(n)\, p, \tag{3.16}$$

where D is the deviance, n is the sample size, and p is the degrees of freedom used up in the calculations. Unfortunately, we are once again stuck with the problem of finding a credible value for p. The prospects are really no better under the second scenario when no $f(X)$ is assumed but there is interest in one or more several conditional proportions of corresponding terminal nodes. The nodes are just subsets of the data defined by the fixed values of predictors. But, there remains the problem of how one defines the degrees of freedom and the likely bias in the estimated proportion noted earlier.

Recently, Hothorn and his colleagues (2006) have suggested another approach to tree construction that builds on and then provides some hypothesis tests. A number of statisticians have observed that other things being equal, predictors with more possible breaks have a greater chance of being selected as a partitioning variable. As a result, there is bias built into the tree structure and implicitly, the results summarized in the terminal nodes. Hothorn and his colleagues suggest a tree-building procedure that has much the same look and feel of conventional stepwise regression.

1. With a null hypothesis that each predictor is unrelated to the response, conduct a global hypothesis test.
2. If the test is not rejected, stop.
3. If the test is rejected, select as the splitting variable the predictor having the strongest relationship with the response.
4. Choose the best split using the selected predictor.
5. Repeat Steps 1–4 until no further splits are indicated.

All of the tests are based on permutation distributions in which the response is shuffled. Each predictor is subjected to a permutation test under the null hypothesis of no association. An overall p-value is also computed adjusting for multiple tests (e.g., a Bonferroni correction). If the global null hypothesis is rejected, the predictor with the smallest p-value is chosen. Then, the split can be determined as usual. The same process is applied for each subsequent partitioning of the data.

There is excellent software in R implementing these procedures. Because recursive partitioning can be an intermediate step in other statistical learning procedures, there are also interesting extensions built into the software. The procedure can be found in the library *party*.

Hothorn and his colleagues argue that selecting predictors in this fashion leads to an unbiased recursive partitioning of the data. However, some caution is warranted. First, this approach assumes, as before, that there is a true $f(X)$ one is trying to estimate with $\hat{f}(X)$, that all of the predictors in X are in the dataset, and that all are well measured.

Second, the tests also take the predictors as fixed. Sampling variation comes only from the response variable. The motivating thought experiment envisions all possible assignments of the response variable values to existing cases within a given marginal distribution of the response variable. There is, therefore, an issue of how well the permutation thought experiment corresponds to the manner in which the data were actually generated and whether one's inferences are really to be limited only to the x-values that were realized.

Third, the results depend on the sample size; other things being equal, larger samples will lead to larger trees. Larger trees will usually perform quite differently from smaller trees, as we have seen. And one does not normally seek to determine the correct model conditional on the sample size. That is, it is at least unconventional to proceed as if there were many correct models, one for each sample size.

Fourth, there will still be bias in the estimates coming from terminal nodes insofar as there is remaining heterogeneity in predictor values associated with the response variable. This too was addressed earlier.

But more generally, for a very large number of applications, these sorts of concerns are moot. The third motivating scenario is likely to be the operational one in practice. There is no $f(X)$, or no population or stochastic process, or no appropriate data-generation mechanism, or the data may be of insufficient quality (e.g., key predictors are missing). Then, the goal is far more likely to be description than estimation. There are only, then, descriptions that are more or less useful. Furthermore, there is no requirement that a single description be chosen. Different descriptions may highlight different, but instructive, features of the data. Then, it can be appropriate to report more than one set of results.

3.9 Classification Versus Forecasting

In most of the discussion of CART so far, and all of the examples, the emphasis has been on fitting the data on hand. With categorical response variables, this has been a classification exercise. A key objective of the analysis is to minimize some aggregate measure of fit that depends substantially on classification errors and their costs. References to forecasting have typically been indirect and/or brief.

As a technical matter, however, the step to true forecasting is relatively easy. One applies the results from the data analyzed to new data not used in the fitting process. A key difference is that for the data used to build the tree, both the predictor values and response values are known. For the data to be used in the forecasting exercise, only the predictor values are known. The key assumption is that the relationships between the predictors and the response for the data analyzed would be the same for the new data, within chance error, were the values of the response variable known.

Sometimes forecasts into the future are desired. Sometimes forecasts into the past as desired (often called "backcasting"). And for some forecasts, time plays no role. A CART analysis undertaken on inmates in one prison, for example, may be used to forecast the misconduct of inmates in another prison. And to confuse things a bit more, we saw in the last section that when missing data are imputed, the enterprise looks like forecasting. In each of these applications, the key idea is that some or all of the values of a variable are unknown, and there is a need for some "best-guess" values. However, although there are some important parallels with forecasting, imputation does not involve a second dataset.

In CART, the classes assigned to terminal nodes are used as the best-guess values. In CART-speak, one "drops" new cases "down" the classification tree. Each case will "land" in one (and only one) terminal node. The earlier classification of each terminal node determines the prediction for all of the

new cases when they arrive. Thus, if a given terminal node is classified as class "1," all of the observations that come to rest in that terminal node are classified as a "1," and the class represented by that "1" is the forecasted class (e.g., misconduct).

But how is the quality of those forecasts determined if the response is not yet known? There are four steps.

1. Build a classification tree as usual using a training dataset taking the costs of classification errors into account.
2. Forecast using a test dataset in which the outcomes are known.
3. Determine forecasting skill from an analysis of the forecasting errors, perhaps using the percentage of cases incorrectly forecast conditioning on the truth. Usually, this is presented in the form of a confusion table.
4. Assume that the forecasting skill demonstrated with the test data applies to new data for which forecasts are desired. This assumption can be supported if the new data are a random sample from the same population as the training and test data.

Forecasting is an important application for CART and all of the procedures discussed in the pages ahead. Moreover, the difference between classification and forecasting, often confused when the term "prediction" is used for both, figures significantly in the next chapter. We show that forecasting errors, rather than classification errors, are better tools for refining algorithmic models.

3.10 Varying the Prior, Costs, and the Complexity Penalty

Figure 3.4 shows again the tree diagram for the CART analysis of inmate misconduct. The tree diagram has been simplified a bit anticipating the need to show more complicated structures shortly. Recall that the empirical distribution of the response variable was used as the prior distribution, and the costs were assumed to be the same for false negatives and false positives. For the earlier figure, the number of terminal nodes was constrained by explicitly setting the minimum terminal node sample size. Now, for reasons that are apparent shortly, we get to the very same place by setting the value of the penalty for complexity instead.

Recall also that a confusion table was presented as part of the earlier analysis. It is now reproduced as Table 3.4. One questionable feature of the table was that there were about eight false negatives for each false positive, implying that false positives were far more costly than false negatives. At that time, there was no justification given for these or any other ratio of relative costs.

Conversations with prison officials indicated that from their point of view, false negatives were much worse than false positives. Failing to anticipate

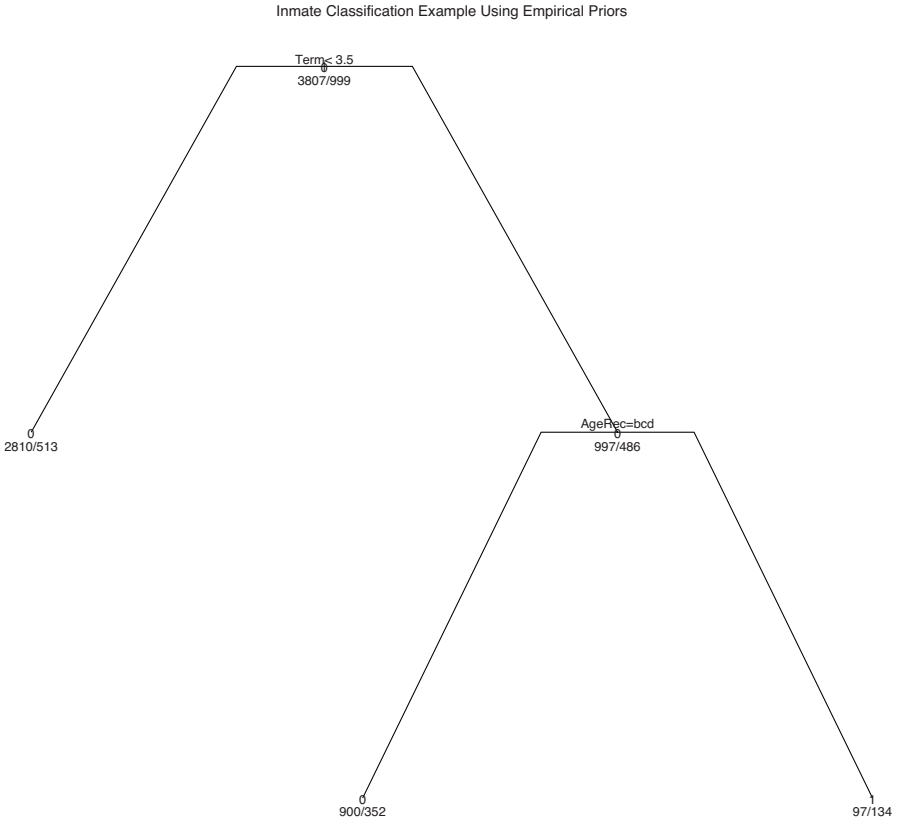

Fig. 3.4. Inmate misconduct example with empirical priors.

	Predict No Misconduct	Predict Misconduct	Model Error
No Misconduct	3710	97	.03
Misconduct	865	134	.88
Use Error	.19	.42	Overall Error = .20

Table 3.4. CART confusion table for forecasting inmate misconduct.

inmate misconduct, which could involve fatal violence, was of far greater concern than incorrectly labeling an inmate as high risk. When pushed, a prison official said that the cost of a false negative was at least five times greater than the cost of a false positive. Hence, the earlier analysis got things upside down.

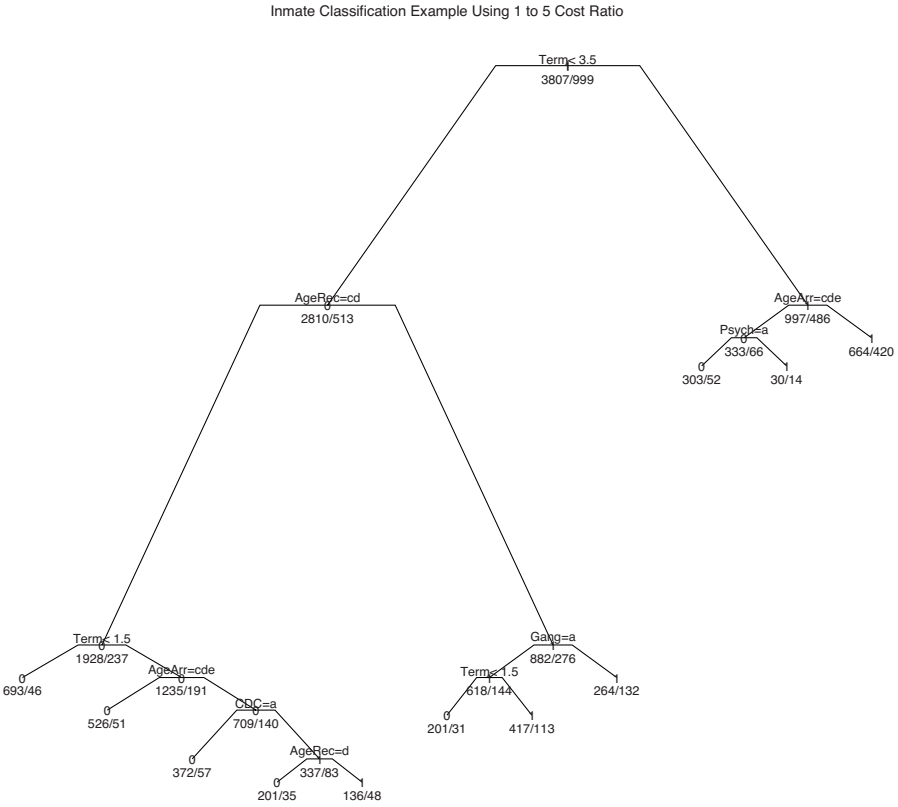

Fig. 3.5. Inmate misconduct example with one to five cost ratio.

Figure 3.5 shows the tree diagram that results when the same value for the complexity penalty is used, but the empirical prior is replaced by a new prior representing the one to five cost ratio for false positives to false neg-

atives elicited from prison officials. The new prior was calculated using the procedures described earlier.

Clearly, the results have changed substantially. Increasing dramatically the costs of false negatives relative to false positives (and it is only the relative costs that matter) leads to a very different result. There are many more terminal nodes reflecting the impact of several more variables. These now include:

1. Gang activity (Gang) with a = gang activity and b = no gang activity.
2. Age at first arrest (AgeArr) with a = 0–17, b = 18–21, c = 22–29, d = 30–35, and e = older than 35.
3. Age at arrival at the reception center (AgeRec) with a = 16–20, b = 21–26, c = 27–35, and d = 36 older than 35.
4. Mental illness (Psych) with a = ill and b = not ill.
5. Previously served time under the state's Department of Corrections (CDC) with a = served time and b = did not serve time.
6. Sentence length (Term) in years.

A larger number of predictors were included in the analysis. Figure 3.5 shows the predictors that CART selected.

We do not interpret Figure 3.5. It is quite complicated and would take us far afield. We consider a more simple CART analysis shortly. For now, the main point is that from Figure 3.5 one can see that the labeling of a terminal node with a "1" or a "0" is not the result of a majority vote of the cases in each node. The vote now takes costs into account captured by the altered prior distribution of the response. This leads directly to Table 3.5.

First, examine the off-diagonal cells. The balance of false negatives to false positives has been reversed. The ratio of false negatives to false positives now reflects approximately the one to five cost ratio of false negatives to false positives. The ratio is not exact because the cost ratio is but one input into the CART algorithm. CART is trying to respond to several features of the data and the analysis requested.

Second, the more desirable balance of false negatives to false positives seems to come with a price. Overall, there is an increase in the number of classification errors. The sum of the off-diagonal cells divided by the total number of cases is about .20 for Table 3.4 and about .37 for Table 3.5. The number of cases incorrectly classified has increased by 17%. So, by that yardstick we are doing substantially worse. This is a general result. Introducing any cost ratios other than one to one will increase the overall proportion of cases misclassified.

But the yardstick of correct classifications can be misleading. Once one abandons the assumption that the costs of false negatives and false positives are the same, the proportion of cases misclassified is not by itself responsive to the way the analysis has been undertaken. Classification errors are now being weighted to take differences in costs into account, but these weights are ignored when the proportion of cases misclassified is computed. All classification errors are being treated the same, even though they are not.

Third, a far more instructive way to consider how well CART has fit the data is to look at the classification errors conditional upon the actual outcome. This means focusing on the rows in Tables 3.4 and 3.5. Within a row, there is only one kind of classification error (false positives or false negatives), so the weighting problem disappears. One can then see that the proportion of misconduct cases misclassified has dropped from .88 to .27. At the same time, the proportion of no misconduct cases has increased from .03 to .40. A tradeoff between false negatives and false positives is generally to be expected and is plainly seen here.

A analogous tradeoff can be seen in the column proportions. When a case is assigned the no misconduct label, that label is now wrong for .11 of the observations. With equal costs, that proportion is .19. When a case is assigned the misconduct label, the proportion of cases labeled in error increases from .42 to .67.

To get some practical sense of what these changes mean, suppose 100 inmates are classified either as misconduct cases or no misconduct cases. For inmates given a no misconduct label, the number of inmates labeled in error drops from 19 to 11 when the equal costs assumption is replaced by the one to five costs assumption. Clearly, this would be a desirable result for prison administrators.

For inmates given a misconduct label, the number of inmates labeled in error increases from 42 to 67 when the equal costs assumption is replaced by the one to five costs assumption. For both of the assumptions about costs, far more errors are made when inmates are assigned to the misconduct class. And the one to five cost ratio makes things worse.

But, the increase in false positives is precisely the result mandated by the one to five cost ratio provided by prison administrators. The first analysis had too many false negatives relative to false positives. The second analysis was motivated by a need to correct this perceived imbalance. When the one to five cost ratio was determined, prison administrators were making a conscious decision to live with a greater number of false positives.

	Predict No Misconduct	Predict Misconduct	Model Error
No Misconduct	2296	1511	.40
Misconduct	272	727	.27
Use Error	.11	.67	Overall Error = .37

Table 3.5. CART confusion table for forecasting inmate misconduct using a one to five cost ratio.

The key point is this: although it is always desirable to have a small number of false negatives and false positives, a decision-maker may be better off with increases in either if relative costs of false negatives to false positives are more

accurately represented. More errors can actually lead to better decisions when costs are taken into account.

Figure 3.6 represents a third analysis of the same data. The one to five cost ratio is maintained, but a larger penalty is given for complexity. The intent is to simplify the tree so that only the most important and meaningful terminal nodes are included. At the same time, there is certainly nothing definitive about Figure 3.6. Another data analyst might quite properly construct a tree that was either more or less complicated.

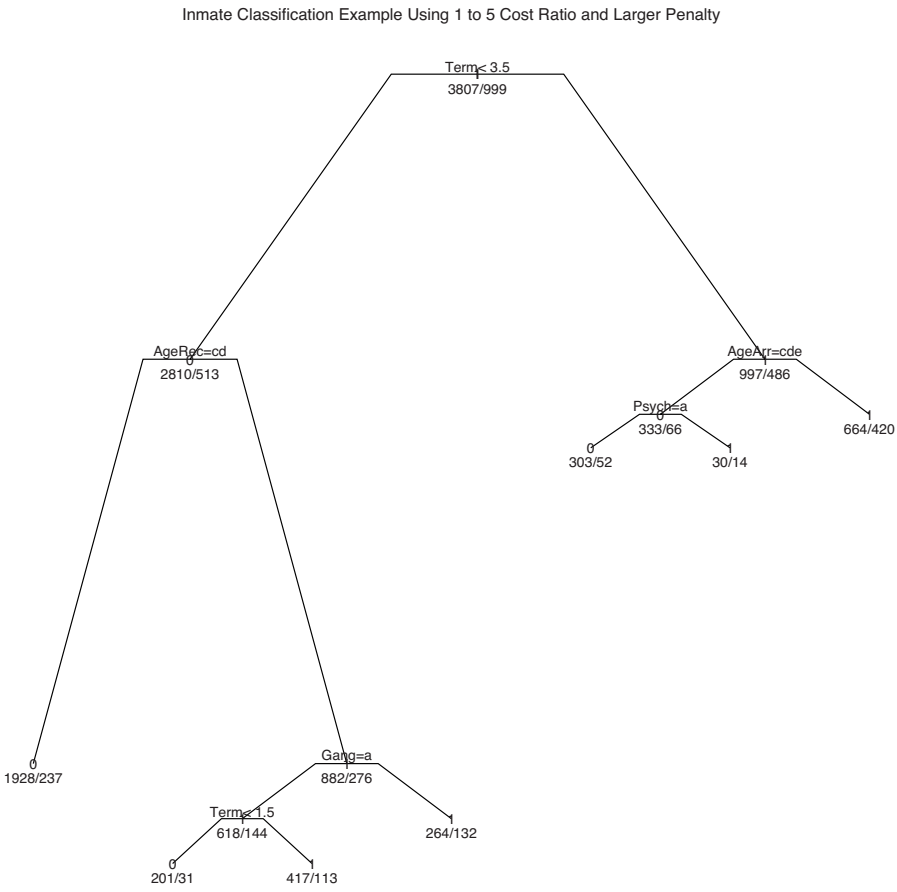

Fig. 3.6. Inmate misconduct example with one to five cost ratio and larger penalty.

Despite the use of weighted votes, the branches in the tree are interpreted exactly as before. We learn for instance, that individuals with sentences equal to 3.5 years or longer and under the age at 18 when first arrested are classified as prone to misconduct. And these inmates pose the highest risk. We also learn that individuals with sentences less than 3.5 years who are 27 years or older when they arrive at the prison reception center are not classified as prone to misconduct and pose the lowest risk. If one is interested in the proportion of inmates in each terminal node who were actually reported to have engaged in misconduct, that is available in the CART output, at least in the R implementation rpart().

The other combinations fall in between. Thus, (working down the tree) individuals who have been sentenced to less than 3.5 years, who are younger than 27 years, who have a history of gang activity, and who are actually serving very short sentences of less than 18 months, are not classified as prone to misconduct. Having a very short sentence seems to trump youth and gang activity, which are normally useful predictors of misconduct in prison. Note that if all of these factors are the same except that the sentence is between 18 months and 3.5 years, the inmate is classified as prone to misconduct.

The confusion table is not presented. Overall, there is an increase in the errors, as one would expect from a less complex classification tree. However, the smaller tree may be more stable, a topic to which we return later. Because the one to five cost ratio is maintained, the various tradeoffs associated with false negatives and false positives are essentially unchanged.

3.11 An Example with Three Response Categories

There is no formal problem in extending CART to three or more response variable categories. We return to the prison data once again and use the same predictors as before. But this time, there are three categories to the response: no misconduct, minor misconduct, and serious misconduct. These are coded as "0," "1," and "2," respectively. About 78% of the cases have no reported misconduct, about 20% have minor reported misconduct, and about 2% have serious reported misconduct. As required, these are mutually exclusive and exhaustive categories. The 2% represents very rare cases, which creates some special difficulties we address in more depth later. For now we ignore the problem.

The three response classes can be viewed as qualitatively different. In particular, most minor infractions are violations of prison rules and would, by and large, not be considered crimes if committed outside of prison. Most of the serious misconduct represents acts that would be felonies anywhere: drug trafficking, sexual assault, robbery, homicide, and the like. On the other hand, if one thinks of prison misconduct as arrayed on some scale of seriousness, the three classes are perhaps ordered. However, just as in conventional multinomial regression, CART take no account of such ordering. Whatever

information is contained in the ranks is ignored. In short, the three classes of misconduct are treated by CART as unordered categories whatever the truth may be.

The first job in the data analysis is to specify a loss matrix. Recall that when there are more than two response categories, costs cannot usually be captured in a prior distribution. So, the empirical distribution is taken as the prior distribution and costs are introduced directly with the loss matrix.

Table 3.6 shows a loss matrix containing costs roughly consistent with information provided by corrections officials. They were especially concerned about inmates who were a serious safety threat but not classified as such. In particular, if an inmate actually had an incident of reported serious misconduct (e.g., assault on a guard) and was classified as having no misconduct at all, the cost assigned is 20. If an inmate actually had an incident of reported minor misconduct (e.g., failure to report to a job assignment) and was classified as having no misconduct at all, the assigned cost is 10. The former error has twice the cost of the latter error, which in turn is substantially higher than any of the other costs. The costs of incorrectly classifying an inmate as misconduct free are relatively small.

	None Predicted	Minor Predicted	Serious Predicted
No Misconduct	0	2	5
Minor Misconduct	10	0	3
Serious Misconduct	20	10	0

Table 3.6. Costs of classification errors for three misconduct categories.

Figure 3.7 shows the resulting classification tree. It can be read just as when there were two response categories, but now the numbers associated with each node are the counts (left to right) for no misconduct, minor misconduct, and serious misconduct. When the unequal costs are used, the classification assigned to each terminal node is not by a plurality of votes. Some votes, in effect, have more weight than others.

In this instance, the very high relative costs for misclassifying serious incidents of inmate misconduct dominate the results. Of the six terminal nodes, four are classified as a "2," the coded value for serious misconduct. The two other terminal nodes are all classified as a "0," the coded value for no misconduct. And none of the terminal nodes are classified as a "1," the value for minor misconduct. The loss matrix led to only two kinds of classification: no misconduct and serious misconduct.

It is important to stress that these results are neither "right" nor "wrong." Their usefulness would depend in part on whether in retrospect, prison officials liked the values in the loss matrix that in this instance produced just two classifications: good inmates and bad inmates. Had the costs of failing

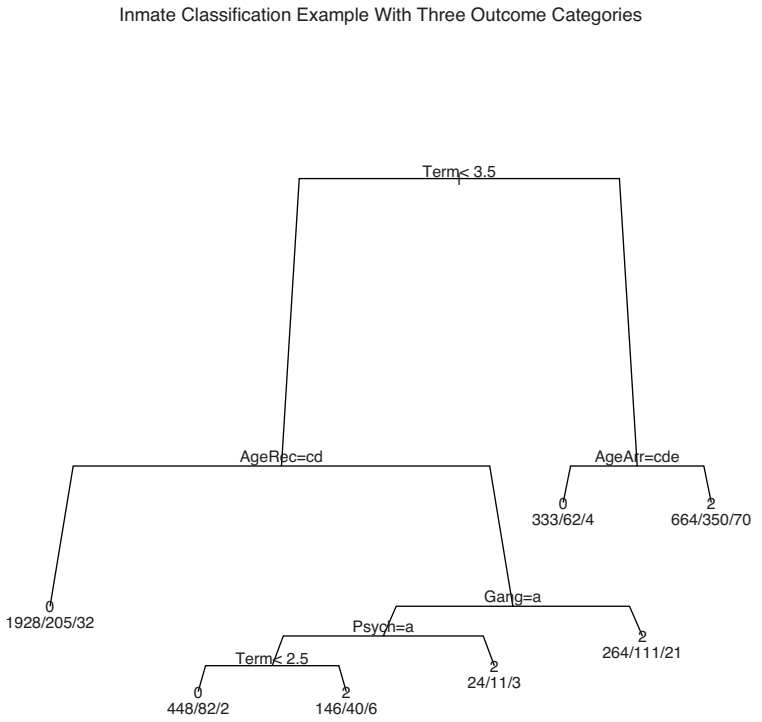

Fig. 3.7. Inmate misconduct with three response categories.

to properly classify the serious misconduct inmates been reduced relative to the other costs of misclassification, a very different set of results would have materialized, including the appearance of all three classification categories for the terminal nodes.

The roles that the predictors play are also shown, but sometimes imply complicated and even counter intuitive interpretations. One reason may be that distinctions are now being made among three kinds of inmate behavior, not two. Moreover, what some may think is an ordered set of classes is being treated as categorical only. Different patterns of association can result. Another possibility is that some of the breaks represent unstable distinctions that should not be taken seriously, a point discussed in more depth shortly.

If corrections officials were directly involved in discussions of how such results might be used, it is likely that several different patterns of costs and several different values for the complexity penalty would be applied. Trees requiring complicated and counter-intuitive interpretations might then be eliminated. For example, even if the costs in Table 3.6 were maintained, a tree pruned back to the first two breaks might well be preferable.

	Predict None	Predict Minor	Predict Serious	Model Error
No Misconduct	2709	0	1098	.29
Minor Misconduct	349	0	512	1.0
Serious Misconduct	38	0	100	.28
Use Error	.12	Undefined	.94	Overall Error = .42

Table 3.7. CART confusion table for forecasting three categories of inmate misconduct.

Another factor that would have to be taken into account is the confusion table. Table 3.7 shows the confusion table from Figure 3.7. The error percentages are computed combining the two sources of error in each row or column. Then row and column proportions are computed as before. Thus, Table 3.7 is interpreted in virtually the same manner as the binary outcome case.

Corrections officials would have to examine each row and column in Table 3.7 and decide if the numbers for the different kinds of false negatives and false positives were reasonable for their purposes. To take the most extreme example, no inmates are assigned to the minor misconduct class. It follows that all inmates who had engaged in minor misconduct are misclassified as being misconduct free or having engaged in serious misconduct. This would likely be unacceptable because in fact, minor misconduct is relatively common, can be quite disruptive, and can be a precursor to more serious incidents. In response, prison officials might favor altering the relative costs of classification errors for minor misconduct. For example, the cost of classifying cases of minor misconduct as if there were no misconduct might be increased.

There are perhaps two main messages from the analysis just summarized. First, there are in principle no logical or computational obstacles moving from two response categories to three or more response categories. But there can be problems from sparse data that will undermine a wide variety of statistical procedures, including CART. For example, some of the row proportions may be computed from very few observations. Important instabilities can result.

Second, moving from two response variable categories to three response variable categories complicates matters significantly. The CART algorithm is being implemented exactly as before. But, the loss matrix has more elements, the confusion table has more entries, and the tree diagram can be difficult to interpret. Compared to the binary case, these changes can be dramatic. For example, going from two to three response categories doubles the number of

elements in the loss matrix that need to be specified and doubles the kinds of classification error that need to be evaluated. Trial-and-error tuning that is likely to follow will usually require juggling many different parts of the output, which will be a challenge to data analysts and practitioners alike.

3.12 CART with Highly Skewed Response Distributions

Response variables that are highly skewed (often called "unbalanced") can create serious problems for any form of regression analysis. One problem can be sparse data, which can lead to unstable results or even an inability of the software to provide any results at all. Another problem is that the rare observations may have a disproportionate impact on the findings. Generalizations to the mass of the data can then be problematic. Yet another problem is that with highly skewed response variables, it can be very difficult to find predictors that are able to improve the overall fit. Thus, if the response is binary, and the marginal distribution is 95% 0s and 5% 1s, very accurate classifications can be determined from this information alone. If the 0 category is always assigned, the classifications will be correct 95% of the time without using any predictors whatsoever. It is difficult to do better than this.

There are hints in the material presented earlier of how these sorts of problems may sometimes be addressed. For example, if a prior distribution places heavy weight on the misclassification of rare cases, it is a bit like saying there are many more rare cases to be considered than the data indicate. And in fact, CART will behave as if this were so. We explore this matter in depth in the chapters to come and find that there are several other promising options. In the meantime, the message is that when the response variable is highly skewed, one must examine all CART output very carefully.

3.13 Some Cautions in Interpreting CART Results

Just as for any data analysis procedure, the output from CART always demands scrutiny before substantive conclusions are reached. There are commonly three kinds of potential problems: inappropriate response functions, unstable tree structures, and unstable classifications. All can produce results, which if taken at face value, risk serious interpretive errors.

3.13.1 Model Bias

If the goal of a CART analysis is to determine the $f(x)$, whether the function is part of a causal model or a feature of a conditional distribution, there is no guarantee that in a given sample CART will even come close. As noted more broadly in Chapter 1, there are no formal mathematical results indicating that CART will find the correct function from a given sample, even if all of

the requisite predictors are provided and even if these predictors are very well measured.

Indeed, there is good evidence, as noted earlier, that CART will tend to select predictors in a manner that introduces bias (Hothorn et al., 2006). Other things being equal, predictors with a larger number of distinct x-values are favored over predictors with a smaller number of distinct x-values when all possible splits are evaluated. In addition, the use of surrogate variables in response to missing data will further bias the selection of predictors.

Of at least equal concern is the matter of functional form. If $f(X)$ is smooth, each CART basis function, necessarily relying on indicator variables, will be incorrect. The hope is that the indicator variables will provide a useful $f(X)$ approximation. Typically, it is difficult to determine if the approximation is good, but as a formal matter, some bias is likely.

If the goal is to obtain estimates of the response conditional probabilities and classifications, there are related difficulties. If the functional form estimated is substantially in error, it likely the terminal node proportions and the classifications that follow will be substantially biased. And as discussed earlier, unless the terminal nodes are homogeneous, probability and classification estimates for individual cases will likely be biased.

In summary, even under the best of circumstances, unless the $f(X)$ is a step function, there will be biases in any CART estimates. The practical question is how serious the biases are likely to be. This is one important reason why CART has been largely superseded by procedures considered in later chapters.

3.13.2 Model Variance

CART partitions the data into more and more homogeneous subsets and then assigns a class label to each terminal node. The class that CART assigns to each case depends on the terminal node where that case comes to rest and the class label assigned to that node. There are several ways in which these steps can lead to unstable results.

The most obvious cause of instability is a small number of observations in any node. Then, a very few observations can send CART down one kind of branching structure rather than another. There can be a tipping effect in which a split relatively high up in the tree structure that depends on a few data points has cascading impacts on later splits. If those few observations are removed from the data, or if they are replaced with different observations, a different tree structure with different terminal nodes can follow.

Why might the new observations be different? They are likely to be different if there is a second random sample from the same population; random sampling error can make them different. They are also likely to be different if the predictor with which the split was constructed was measured with random error. This implies that a new realization of the data created by a new round of measurements can easily produce very different results. For example,

if SAT score is a predictor, the scores from the test taken in the junior year of high school will likely differ, at least a bit, from the scores taken in the senior year of high school at least because of noise alone.

Fortunately, small samples in particular nodes are easily spotted, and there are several good remedies within the usual CART software. One can, for example, increase the minimum number of observations in all nodes or increase the value of the complexity penalty. Larger node sizes will result, and the stability of the output will tend to increase.

A less apparent source of instability derives from terminal nodes that remain relatively heterogeneous. If the split in a given terminal node is near the 50–50 threshold, the movement of just a few cases across terminal nodes can change the classes assigned to each terminal node and dramatically alter the classes assigned to observations within them. Note that this is not a problem caused by a small number of observations in terminal nodes, although if there are few observations in terminal nodes, the instability caused by near-even proportions is made worse.

Instability resulting from heterogeneous terminal nodes is relatively easy to spot when the empirical distribution is used for the prior distribution and when the costs of false negatives and false positives are taken to be the same. For each terminal node, the numbers of observations in each response class can be easily inspected and compared. But when the classifications assigned to terminal nodes depend on more than the within-node counts, it is necessary to dig deeply in CART output to determine what is going on.

Even if one is able to conclude that heterogeneous terminal nodes are problematic, there is often little to be done. The cause lies in weak predictors that are unable to partition the data so that relatively homogeneous subsets result. And under such conditions, it may not be wise to take the tree structure or fitted values very seriously.

A related problem affects how a splitting variable is chosen. That variable may be the predictor from the preceding split, or another predictor. If the same variable, one has a step function representing nonlinear features of the relationship between that predictor and the response variable. If a new variable, one has an interaction effect representing a different kind of step nonlinearity. One predictor's relationship with the response variable depends on the value of the other predictor. The subject matter interpretations can also be very different. Yet, the choice between the two may be precarious, especially when predictors are highly correlated with each other.

Figure 3.8 shows possible partitions of the data when predictors x and z are highly correlated. In this illustration, CART constructs an initial partition with the solid vertical line. To the left of that line, B-values dominate. To the right of the line, A-values dominate. Both partitions are, therefore, more homogeneous than the dataset as a whole. The partitioning is a success.

But what should the next partition be? Consider a partition on the right side of Figure 3.8. There is clearly a cluster of all A-values in the upper right-

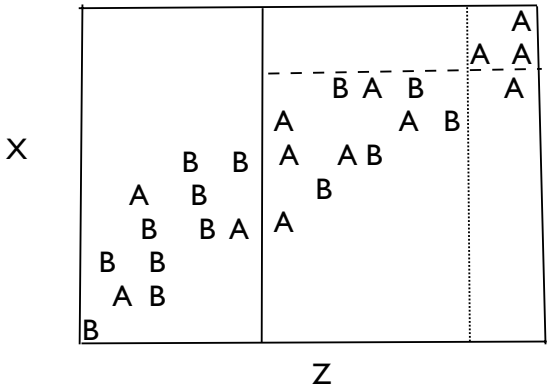

Fig. 3.8. High instability in CART.

hand corner. But because the two predictors are highly correlated, there are two ways to isolate them, which lead to very similar reductions in impurity.

The horizontal dashed line constructs a partition based on x. That partition isolates three of the four target values and implies an interaction effect. Given the previous split on z, the next split should be on x. An interaction effect results.

The other partitioning is accomplished using the vertical dotted line. This one uses z, and may be slightly preferred because now all four of the target values are isolated. As a result, a second step is introduced in the step function through which z is related to the response.

Both splits address nonlinearities in how the response is related to predictors. But one solution is a more complicated step function, and the other is an interaction effect. The choice between the two in this illustration depends on a single observation. With CART, there is always the possibility of instabilities of this sort. But just as in linear regression, potential instabilities are more likely to materialize if the predictors are highly correlated. Two or more predictors can more readily compete for the same partition of the data.

Figure 3.9 illustrates how a lower correlation between the two predictors can help. With a less clustered scatterplot, it is more difficult to find a small number of observations that both x and z can isolate. Notice that either of the competing partitions in Figure 3.8 now include many observations that fall into one partition but not the other. The two predictors are not competing

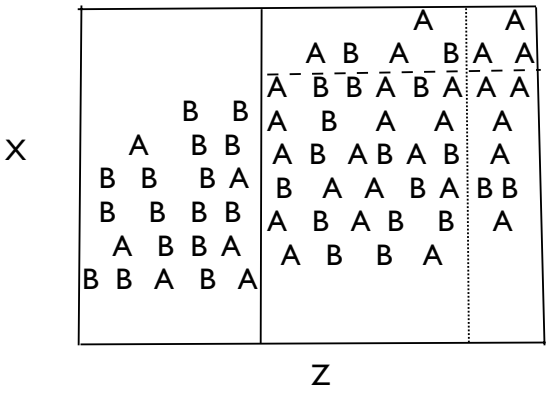

Fig. 3.9. Low instability in CART.

any longer for nearly the same set of observations. The choice between the two is now less likely to be made on a stray observation or two.

An unstable tree structure does not automatically mean that the assigned classes are unstable as well. In the case of two highly correlated predictors, for example, one can obtain rather different tree structures depending on which variable is chosen. And the different tree structure can certainly lead to different interpretations of how the response is related to the predictor. However, the classes assigned to observations, once they come to rest in a terminal node, may be much the same. A very different tree structure does not necessarily imply very different classifications. There may be two or more ways to get essentially the same classification results.

More generally, if the goal is accurate classification with little concern about describing how inputs are related to outputs, an unstable tree structure may not matter. What matters is whether under an alternative tree structure, observations tend to land in terminal nodes that classify those observations in the same way. It is not important whether there are more terminal nodes or fewer. The path taken to those nodes is not important either. And from that perspective, which highly correlated predictors are actually used to define the splits is formally irrelevant. The goal is stable classification.

In summary, just as in parametric regression, tree instability can result from small samples, weak predictors, or highly correlated predictors. Tree structure and/or the classes assigned to observations can be adversely affected. However, because of CART's stagewise structure, the particular tree that

results can be far more fragile than the output from a parametric regression. A single tweak near the top of the tree can fundamentally alter all that follows.

Two recommendations follow. First, there is (once again) no substitute for careful examination of the classification tree to make sure the tree makes sense. Splits that violate common sense, well-accepted theory, or past research may need to be discounted and may even call the entire tree into question.

Second, it can be very useful to get some empirical sense of how stable the results really are. As shown in the next chapter, a good strategy is to construct several bootstrap samples of the data and apply CART to each. If the tree structures are substantially different, interpreting the results from any one tree is risky. It is also important to compare the classifications from each of the trees. Substantial differences imply that the classifications assigned to observations cannot be relied upon.

A good place to start an examination of the classes assigned is with confusion tables. One question is whether the confusion tables for the different trees are alike. Another question is whether the same classes were assigned to the same cases over bootstrap samples; how consistently were the observations classified. One can compare the assigned classes for cases selected in two or more of the bootstrap samples. This idea is closely related to the "margin" an observation has and is discussed at some length in later chapters.

A related approach is to use the proportion of cases of a particular class in each terminal node. Sometimes these proportions can be considered estimates of the probability that an observation in a given terminal node is a member of the assigned class for that node. Then, for cases that appear in any pair of bootstrap samples, a scatterplot can be constructed using the terminal node proportions from two different trees. Substantial departures from the 45-degree line indicate meaningfully different assigned probabilities.

A more compelling approach is to determine if the different CART models classify test data in the same manner. When no test sample is available, there are creative approaches that depend on resampling, in much the same spirit as cross-validation. How such data may be obtained and exploited is a very important topic that is addressed in later chapters.

If one concludes that CART results are unstable, it can sometimes be helpful to apply CART to the original data again after setting the tuning parameters to produce more stable results. Once again, increasing minimum sample sizes for all nodes or increasing the value of the complexity penalty can be helpful. In the next chapter we consider other alternatives.

3.14 Regression Trees

The emphasis in this chapter has been on categorical response variables. Classification was the goal. The reasons were both pedagogical and practical. By concentrating on categorical response variables, the full range of fundamental issues surrounding CART are raised.

But CART is certainly not limited to categorical response variables. Quantitative response variables are also fair game. And with the discussion of categorical response variables largely behind us, a transition to quantitative response variables is relatively straightforward. It is possible to be brief.

A key difference with regression trees is the splitting criterion employed. For the conventional regression case, the impurity of a node is represented by the within-node sum of squares for the response:

$$i(\tau) = \sum (y_i - \bar{y}(\tau))^2, \tag{3.17}$$

where the summation is over all cases in node τ, and $\bar{y}(\tau)$ is the mean of those cases. Then, much as before, the split s is chosen to maximize

$$\Delta(s, \tau) = i(\tau) - i(\tau_L) - i(\tau_R). \tag{3.18}$$

No cost weights can be used because there is no reasonable way to consider false positives and false negatives without a categorical response variable. Then, to get the impurity for the entire tree, one sums over all terminal nodes to arrive at $R(T)$. Regression trees can be pruned, but usually with a penalty based on the AIC or some other statistic that takes the number of estimated parameters into account. Overall fit quality is then based on a summary statistic, such the root mean squared error or a measure that adjusts the degrees of freedom, much as in conventional parametric regression.

There is no classification as such. Each observation is placed in a terminal node and is then assigned the mean of that node. The assigned mean indicates how a case is "classified." The collection of means for all of the terminal nodes are, therefore, fitted values analogous to the fitted values from conventional parametric regression. They represent how the numerical response is related to the predictors.

Just as in parametric regression, it is fitted values that are typically used in forecasting. With these conditional means in hand, each new observation for which the outcome is unknown is placed in a terminal node, depending on its predictor values. The conditional mean of that node is generally taken as the "best guess" of what the value of the response variable should be.

All of the earlier concerns about CART still apply, save for those linked to the classes assigned. Potential bias and instability remain serious problems. And possible remedies are also effectively the same.

3.14.1 An Illustration

Key output of a regression tree is usually presented as a tree diagram, much as in the categorical response variable case. And as before, interaction effects can dominate. There are often easy extensions to the generalized linear model so that a response measured in counts, for example, can be used.

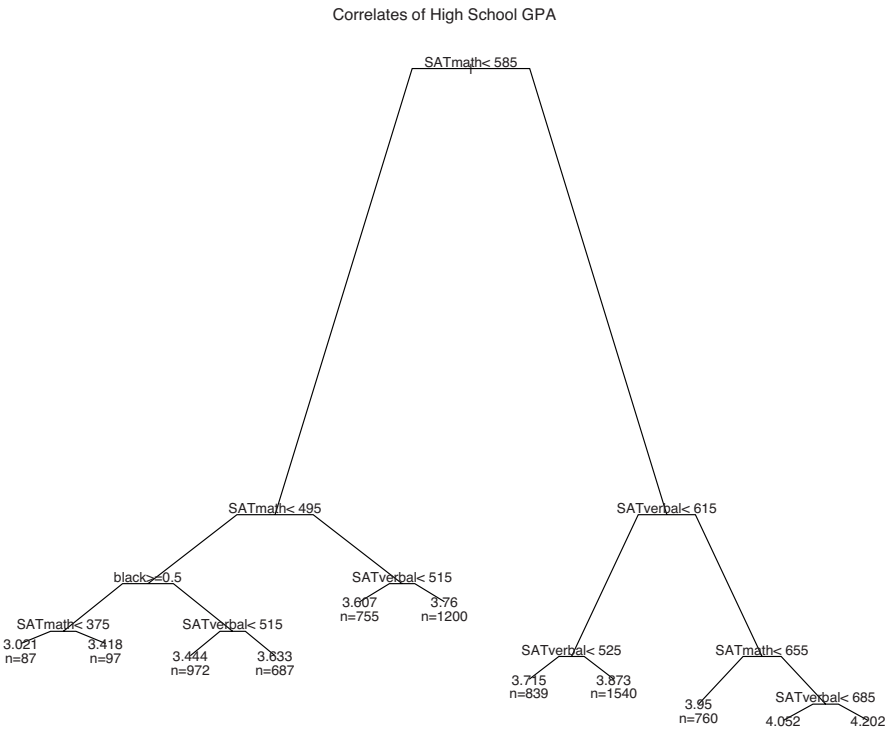

Fig. 3.10. Predictors of grade point average in high school.

Figure 3.10 shows a regression tree for correlates of high school grade point average for applicants to the large public university discussed earlier. Predictors include verbal SAT scores, mathematics SAT scores, and ethnicity. Grade point average can be as high as 5.0 because the scoring gives performance in advanced placement classes extra weight. The mean GPA in each node is shown along with the number of cases. The mean GPA for each terminal node is the fitted value assigned to all observations in that node.

The CART tree indicates that high school GPA is a nonlinear function of performance on the SATs. Ethnicity also seems to matter a bit. Only about 30% of the variance in high school GPA is accounted for with the available predictors. Clearly, GPA is measuring a lot in addition to whatever the SATs

measure, at least among applicants to this university. This is important because it implies that the high school GPA by itself contains potentially important information that cannot be obtained from the SATs.

The means in the terminal nodes range from a high of 4.202 to a low of 3.021. Interpretation follows directly from the tree. At each break point, the cases that meet the split criterion go to the left daughter node. For illustrative purposes, an indicator variable is constructed for each ethnic group. Because these are coded so that the value "1" means the presence of a given ethnicity and "0" the absence, splits are characterized by the halfway point of .5.

Just as in the classification case, many of the break points after the first can represent interaction effects For example, applicants in the node with an average GPA of 3.607 score below 515 on the verbal SAT and between 585 and 495 on the mathematics SAT. Applicants in the node with the highest average GPA score above 685 on the verbal SAT and 655 on the mathematics SAT. The race effect found for applications with middling verbal and math SATs is small and makes little intuitive sense. It may be a chance artifact.

3.14.2 Some Extensions

Save for some details, regression trees can be interpreted much as are classification trees. CART output can be represented in part by a tree diagram that shows how the predictors are related to the response. There are clear links to conventional linear regression as well, with conditional means as fitted values, variation around the fitted values as a product of within-node residuals, and familiar measures of fit.

For regression trees, costs are addressed with the fitting function itself, which in this case is the error sum of squares. Just as with conventional regression, therefore, the fitting assumes a quadratic loss function. Overestimates of the conditional mean are treated the same as underestimates of the conditional mean, and large residuals are given especially heavy weight (thanks to squaring).

We have already seen that when CART is used for classification, symmetric costs are often inappropriate. Yet they are required when the response variable is quantitative. For example, if an admissions officer is trying to forecast how well a student will do in college, a prediction that pegs the student's freshman GPA one full point too high has the same costs as a prediction that pegs the student's freshman GPA one full point too low. One consequence of the first error may be that the student actually struggles in college and then flunks out. Another consequence is that a student who might have done much better was not admitted instead. One consequence of the second error is not admitting a student who might have done very well in college. Another consequence is that a weaker student may have been admitted instead. None of these considerations can be easily introduced in the fitting process when the response variable is quantitative.

The heavy weight given to the largest residuals means that the fitting process can be greatly affected by influential observations, just as in conventional regression. A residual equal to 2.0 is only one point farther away from the conditional mean than a residual of 1.0, but the squaring weights the two observations four to one when the fitting is undertaken. One implication is that a few very atypical observations that are away from the mass of the data may dominate the results, which may then not characterize the bulk of the data appropriately. So, the results may not be an accurate description of the data on hand. A second implication is that the results can be very unstable with respect to random samples of the data. The story can change depending on whether the random sample analyzed happens to include the outliers. Generalization may be seriously compromised.

There are in principle all of the usual ways to "robustify" the CART fitting process. In particular, one can use a linear loss function instead of a quadratic, which implies fitting conditional medians rather than conditional means. Medians will not be affected by outliers and the linear loss function weights larger residuals less heavily than the quadratic loss function. An easier but less elegant fix is to fit a trimmed response so that, for instance, the largest 5% and the smallest 5% of the values for the response variable are dropped before the analysis begins. Some implementations of CART allow for these and other options. When they are available, their usefulness should be carefully assessed in the context of the data to be analyzed and the empirical questions being asked. At the moment, rpart() in R does not allow for linear loss.

Even if concerns about a few influential observations are not significant, it may be appropriate to abandon quadratic loss on other grounds. If there is interest in the conditional medians, linear loss follows naturally. If the response is a count, a Poisson formulation may be appropriate. Then the Poisson deviance is used as the splitting criterion. In R, rpart() currently allows for the Poisson. A rather different set of problems is generated because of CART's well-known proclivity to overfit and favor predictors with many possible splits. The former can lead to generalization error and the latter can lead to bias in the tree structure. As noted earlier, Hothorn et al. (2006) have proposed some novel means to address both problems. There are some important questions about how well their approach will work in practice, but the requisite software is available in R (in the library *party*) and is certainly worth trying.

3.14.3 Multivariate Adaptive Regression Splines (MARS)

Multivariate Adaptive Regression Splines (MARS) can be viewed as another kind of smoother, in the traditions of the last chapter, or as a twist on classification and regression trees. For ease of exposition, this discussion builds on CART, and it is brief. An excellent and more extensive examination of MARS can be found in Hastie et al. (2001: Section 9.4).

A key difference between CART and MARS is in the nature of the basis functions used. The MARS formulation is the broadly familiar

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X), \tag{3.19}$$

where as before, there are M weighted basis functions $h_m(X)$. Likewise, the basis functions are still determined for each split by searching over all predictors and thresholds on those predictors before the best partition is selected. But in CART, the result is a step function. In MARS, the result is a V-shaped function composed of two linear splines, with its point at the threshold value. The two spline functions are mirror images of each other; hence the V-shape. Hastie et al. (2001: 283) call the two splines a "reflected pair."

Another important difference is that nodes may be split more than once. To take a simple example, the root node is initially split in two as usual. But at a later step, the root node may be revisited and split again through a new product variable. This capacity exists for all internal nodes as well.

In short, a MARS model takes the form of Equation 3.19, where the basis functions can be reflected pairs or the product of reflected pairs. Thus, MARS can fit increasingly higher-order interaction terms, just as CART does, but these are the product of linear splines not the product of indicator variables.

Estimation is done by least squares. There are often various tuning parameters that can help determine, for instance, how complex a model is permitted. The output of MARS can include the equation actually estimated, and an ANOVA-type partitioning of the explained variance to represent predictor "importance." A lot more is said about variable "importance" in the next several chapters.

MARS can be extended to classification tasks, where logistic regression replaces linear regression as the primary engine. For both regression problems or classification problems, MARS will sometimes perform better than CART. Its main comparative advantage is the ability to better capture additive models. At the same time, MARS suffers from many of the same weaknesses as CART.

One must be very clear that when MARS is used to describe how predictors are linked to a response, MARS is an exploratory tool. There is no pretense of producing a causal model despite the fact that Equation 3.19 can have much the look and feel of a conventional regression equation. The weights represented by the β_m are regression coefficients to be sure, but they have no necessary causal interpretation. MARS is in the same tradition as all of the procedures we have been considering.

MARS has its advocates, but it does not seem to have the same popularity as CART. One reason may be that MARS is not available in the more popular, large software packages such as STATA or SPSS, nor in free computing environments such as R. A license for MARS must be obtained from Salford Systems. And like CART, MARS seems to have been superseded by newer procedures (Friedman, 1991) that are discussed in the pages ahead.

3.15 Software Issues

All of the computing done is this book was implemented in R. Within R, the best CART implementation arguably is in the procedure rpart() in the R library *rpart*. It is a very powerful and flexible implementation, especially given all of the other capabilities available in R. CART is also available in a number of conventional statistical packages (e.g., SPSS), although the algorithms employed, the options provided, and user interfaces can vary dramatically. For those readers who have not tried to use CART, it may be helpful to briefly consider what the required inputs are likely to be.

1. The response variable needs to be specified, usually with information about whether it is to be treated as quantitative or categorical. CART software can sometimes get confused if the response is binary and represented by an indicator variable or some other numeric values. The safest way to proceed in rpart(), for example, is to explicitly label binary outcomes as a categorical variable (i.e., a factor).
2. The predictor variables, which can be quantitative, categorical, or both, need to be specified. If categorical, it may be important to indicate that explicitly. Otherwise, numerical values, really meant to be just category labels, may be treated as an equal interval scale. Among the issues that usually need to be thought through is whether any product variable for interaction effects should be constructed and entered as predictors or whether it is preferable to let CART construct interaction terms as needed.
3. The method, which usually means either a regression tree or a classification tree, needs to be determined. There are sometimes other options available for count data or survival data. Although some programs can determine the proper method from the nature of the response variable specified, it is usually a good idea to specify the method explicitly in a separate argument. That way, the user knows for certain what method is being used. Moreover, sometimes a CART algorithm will make the wrong choice.
4. The fitting function to be minimized needs to be specified. Sometimes this is determined when the method of analysis is selected and sometimes not. In rpart() with a categorical response variable, one has to specify whether the Gini index or the entropy is to be used.
5. The costs of false negatives and false positives need to be determined, sometimes with an appropriate prior distribution, a loss matrix, or some other means. It is also very important to understand what the defaults on these costs are.
6. A penalty for complexity is often required. It may be the value of α or some transformation of it. In rpart(), for example, the relevant parameter is *cp*, a standardization of α. In the regression case, the value of *cp* is α divided by the error sum of squares in the root node. In the classification case, α is divided by the cost complexity $R(T_0)$ of the root node.

The standardization makes the *cp*-value a fraction and allows for better comparisons across the results from different response variables.

7. Other tuning parameters are often available, such as the maximum number of splits and the minimum sample size for terminal nodes. It is very important to learn what the default values of the tuning parameters are. Although they are usually set to reasonable values, one should not assume that they are.
8. Often one can request predicted values either for the training data or for test data, and specify what form they should take. For classification trees, one usually has a choice between the predicted class or the predicted probability of membership in a class.
9. There are usually a number of output options for different kinds of graphics and summary tables. Which mix of outputs is appropriate depends on the data analysis task. For example, a richer mix of tabular outputs will often be needed the first time a training dataset is analyzed or if there are suspicions about the quality of the data. Output issues also arise for graphical output. For example, getting tree diagrams into a form that is aesthetically pleasing and easy to read can take some doing.
10. It is also likely that any CART software will have its fair share of quirks. With rpart(), for example, priors are entered as a list of the form c(.30,.70). But which element of the list is for which category? Perhaps the arguments should be entered as c(.70,.30). In rpart(), the proper sequence is determined by numerical or alphabetical order. For example, if the response is coded "1" or "2", c(.30,.70) assumes a prior distribution in which 30% of the cases are 1s and 70% of the cases are 2s. If the response is coded "yes" or "no," c(.30,.70) assumes a prior distribution in which 30% of the cases are no and 70% of the cases are for yes.
11. Programs will differ in how well they check for errors and how clearly they communicate with the user when problems are discovered. Continuing with the example of specifying priors, rpart() checks to see if they add to 1.0 and if not, tells the user very clearly. It is far less apparent from CART rpart() output if the loss matrix is constructed incorrectly.

3.16 Summary and Conclusions

CART can sometimes be an effective statistical learning tool. It is relatively easy to use, builds directly on familiar regression procedures, does not demand great computing power, and generates output that can be presented in an accessible manner. CART also provides a useful way to introduce the costs of classification errors. However, CART also has some important limitations.

First, if there is a true $f(X)$ outside of the data on hand, there is no reason to believe that CART's $\hat{f}(X)$ will provide an unbiased estimate. Despite the flexible ways in which CART can respond to data, substantial bias is a real possibility. As noted many times already, if one is interested in the $f(X)$,

one needs X. And X must be well measured. But even if these demanding prerequisites are met, the CART algorithm introduces some important constraints. The step functions used can badly misrepresent smooth functions of X, and the conditional proportions or means in terminal nodes will usually be determined in part by nearest neighbors that do not necessarily have the same population value for the response variable. One's best hope is that the flexibility CART provides will produce a $\hat{f}(X)$ that is less biased than an alternative derived from a parametric procedure.

Second, the splitting decisions can be very unstable. A few observations can in some situations dramatically affect which variables are selected and the precise values used for partitioning the data. Then, all subsequent partitions can be affected. This instability is closely related to overfitting, which can substantially limit the generalizability of the results. The findings from the data examined may not generalize well to other random samples from the same population (let alone to data from other populations). The problem of overfitting is addressed head-on in the next chapter.

Third, even moderately elaborate tree diagrams will seriously tax substantive understanding. The problem is not just complexity. CART is trying in a single-minded manner to use associations in the data to maximize the homogeneity of its data partitions. How those associations come to be represented may have nothing remotely to do with subject matter understandings or how subject matter experts think about those associations.

For example, well-accepted theory and past empirical research may argue strongly for a relatively smooth nonlinear relationship between a predictor and a response. CART may represent the relationship as a complicated step function. Even more confusing, the nonlinearities may be picked up in interaction effects with other predictors, thanks to associations in the data among all of the predictors and between the predictors and the response variable. The splitting process imposes no subject matter constraints on how the nonlinearities are captured. The subsetting can unfold within a single predictor (i.e., a step function) or within two more more predictors (i.e., interaction effects) without regard for what can be sensibly interpreted.

More troubling, if there is no well-accepted theory or strong empirical research to guide interpretation, the data analyst can be at the mercy of the CART algorithm. Thus, interaction effects empirically revealed may force interpretations that are essentially artifacts of how the subsetting was done. In this case, researchers will not just be confused, but risk being led astray.

Fourth, the exploratory nature of CART and the substantial likelihood of bias mitigates against the sensible use of statistical inference. Thus, the role of random sampling error can be very difficult to integrate into any subject matter conclusions. Assessments of the stability of the results can be very helpful, but these are somewhat different from the familiar confidence intervals and hypothesis test p-values.

In summary, it is important to distinguish between conditional means or proportions and the tree structure. Using the tree structure to interpret how

inputs are related to outputs is often a bad idea. Or put more constructively, there needs to be strong subject matter information to protect against misleading interpretations. If interest centers on the conditional means or proportions alone, however, one can sometimes be more hopeful. Trees that differ because of instability will often produce similar sets of fitted values. Thus if CART is used solely as a classifier, the results may be helpful. But if there is a need to explain why cases are classified as they are, the instability may be debilitating.

Fortunately, there is more in the bag of statistical learning tricks than CART. As we soon show, there is really no need to risk CART's significant limitations. One can do better and in the next chapter, we begin to consider how.

Exercises

Problem Set 1

The purpose of this exercise is to provide an initial sense of how CART compares to conventional linear regression.

1. To begin, construct a regression dataset with known properties:

```
x1=rnorm(300)
x2=rnrom(300)
error=2*rnorm(300)
y1=1+(2*x1)+(3*x2)+error
```

   Apply conventional linear regression using lm(). Then apply rpart(), and print the tree using text(). Compare the regression output to the way in which the data were actually generated. Compare the tree diagram to the way in which the data were actually generated. Compare how well linear regression and CART fit the data. (This may take a little doing depending on what summary measures of fit rpart() provides. One easy option is to construct the fitted values with predict() and then regress the fitted values on the observed values to get fit measure comparable to those from the linear regression analysis.) What do you conclude about the relative merits of linear regression and CART when the $f(X)$ is actually linear and additive?

2. Now, redefine the two predictors as binary factors and reconstruct the response variable.

```
x11=(x1 > 0)
x22=(x2 > 0)
y=1+(2*x11)+(3*x22)+error
```

Proceed as before comparing linear regression to CART. How do they compare? What do you conclude about the relative merits of linear regression and CART when the $f(X)$ is actually a step function and additive?

3. Under what circumstances is CART likely to perform better than linear regression? Consider separately the matter of how well the fitted values correspond to the observed values and the interpretation of how the predictors are related to the response.

Problem Set 2

The goal of the following exercises is to give you some hands-on experience with CART in comparison to some of the procedures covered in earlier chapters. An initial hurdle is getting R to do what you want. Make generous use of help(). Also, I have provided a number of hints along the way. However, I have tried to guide you to results in the least complicated way possible and as a consequence, some of the more subtle features of CART are not explored. Feel free to play with these in addition. You can't break anything.

Load the data set called "frogs" from the DAAG library. The data are from a study of ecological factors that may affect the presence of certain frog populations. The binary response variable is pres.abs. Use the help command to learn about the data. For ease of interpretation, limit yourself to the following predictors: altitude, distance, NoOfPools, NoOfSites, avrain, meanmin and meanmax.

1. Use logistic regression from glm() to consider how the predictors are related to whether frogs are present. Which predictors seem to matter? Do their signs make sense?

2. Using the procedure stepAIC() from the MASS library with the default for stepwise direction, find the model that minimizes the AIC. Which predictors remain? Do their signs make sense?

3. Using table() or xtabs(), construct a confusion table for the model arrived at by the stepwise procedure. The observed class is pres.abs. You will need to assign class labels to cases to get the "predicted" class. The procedure glm() stores under the name "fitted.values" the estimated conditional probabilities of the presence of frogs. If the probability is greater than .5, assign a "1" to that case. If the probability is equal to or less than .5, assign a "0" to that case. Now cross-tabulate the true class by the assigned class. What fraction of the cases is classified incorrectly? Is classification more accurate for the true presence of frogs or the true absence of frogs? What is a rationale for using .5 as the threshold for class assignment?

4. Using your best model from the stepwise procedure, apply the generalized additive model. Use smoothers for all of the predictors. Look at the

numerical output and the smoothed plots. How do the results compare to those from logistic regression?

5. Construct a confusion table for the model arrived at through GAM. Once again, the observed class is pres.abs. Use the same logic as applied previously to determine the assigned class. What fraction of the cases is classified incorrectly? Is classification more accurate for the true presence of frogs or the true absence of frogs? How do these results compare to the GLM results?

6. Going back to using all of the predictors you began with, apply CART to the frog data via the procedure rpart() in the library *rpart*. For now, accept all of the default settings. But it is usually a good idea to specify the method (here, method= "class") rather than let rpart() try to figure it out from your response variable. Use the print() command to see some key numerical output. Try to figure out what each piece of information means. Use the plot() and text() commands to construct a tree diagram. What predictors does CART select as important? How do they compare with your results from GLM and GAM? How do the interpretations of the results compare?

7. Use predict() to assign class labels to cases. You will need to use the help command for predict.rpart() to figure out how to do this. Then construct a confusion table for the assigned class and the observed class. What fraction of the cases is classified incorrectly? Is classification more accurate for the true presence of frogs or the true absence of frogs? How do these results compare to the GLM and GAM results? If the three differ substantially, explain why you think this has happened. Alternatively, if the three are much the same explain why you think this has happened.

8. Run the CART analysis again with different priors. Take a close look at the information available for rpart() using the help command. For example, for a perfectly balanced prior in rpart() you would include parms=list(prior= c(.50.50)). Try a prior of .5 for presence and then a prior of .30 for presence. (For this rpart() parameter, the prior probability of 0 comes first and the prior probability of 1 comes second.) What happens to the amount of classification error overall compared to the default? What happens to the ratio of false negatives to false positives? (To understand better what is going on look again at Section 3.5.2.)

9. Using Equation 3.11 set the prior so that false negatives are ten times more costly than false positives (with pres.abs = 1 called a "positive" and pres.abs = 0 called a "negative"). Apply CART. Study the output from print(), the tree diagram using plot() and text(), and the confusion table. What has changed enough to affect your interpretations of the results? What has not changed enough to affect your interpretations of the results?

10. Construct two random samples with replacement of the same size as the dataset. Use the sample() command to select at random the rows of data you need and use those values to define a new sample with R's indexing capability, x[r,c]. For the two new samples, apply CART with the default parameters. Construct a tree diagram for each. How do the two trees compare to each other and to your earlier results with default settings? What does this tell you about how stable your CART results are and about potential problems with overfitting.

11. Repeat what you have just done, but now set the minimim terminal node size to 50. You will need the argument control=rpart.control (minbucket=50)) in your call to rpart(). How do the three trees compare now? What are the implications for overfitting in CART?

Problem Set 3

Here is another opportunity to become familiar with CART, but this time with a quantitative response variable. From the library *car*, load the data set "Freedman." The dataset contains for 100 American cities the crime rate, population size, population density, and percent nonwhite of the population. The goal is to see what is associated with the crime rate.

1. Using the generalized additive model (GAM) from the library *gam*, regress the crime rate on the smoothed values of the three predictors. Examine the numerical output and the plots. Describe how the crime rate is related to the three predictors.

2. Repeat the analysis using rpart() and the default settings. Describe how the crime rate is related to the three predictors. How do the conclusions differ from those using the generalized additive model?

3. Plot the fitted values from the GAM analysis against the fitted values from the CART analysis. The fitted values for gam() are stored automatically. You will need to construct the fitted values for CART using predict(). What would the plot look like if the two sets of fitted values corresponded perfectly? What do you see instead? What does the scatterplot tell you about how the two sets of fitted values are related?

4. Overlay on the scatterplot the least squares line for the two sets of fitted values using abline(). If that regression line had a slope of 1.0 and an intercept of 0.0, what would that indicate about the relationship between the two sets of fitted values? What does that overlaid regression line indicate about how the two sets of fitted values are related?

5. Using scatter.smooth(), apply a lowess smoother to the scatterplot of the two sets of fitted values. Try several different spans. What do you conclude about the functional form of the relationship between the two sets of fitted values?

6. For the GAM results and the CART results, use cor() to compute separately the correlations between the fitted values and the observed values for the crime rate. What procedure has fitted values that are more highly correlated with the crime rate? Can you use this to determine which modeling approach fits the data better? If yes, explain why. If no, explain why.