decision tree methods may uncover relationships and express them in a few decision rules, which might be masked by other more computationally intensive methods.

**General Issues Related to C&RT**

*Multilevel splits.* When there is one obvious split point in the range of a variable, C&RT does well. But if there are potentially multiple split points, the binary splitting approach may oversimplify relationships between variables. See Briemann et al. (1984) for more details and challenges of determining the best binary split point. Also, an excellent discussion of both decision trees and neural nets in general is provided in Ripley (1996).

*The danger of overfitting.* Theoretically, a decision could keep on splitting until it creates terminal nodes for every case. In that case, the tree will keep splitting until not only the signal pattern is modeled, but also the noise in the data is modeled perfectly. The prediction accuracy would be perfect, but the model would probably fail miserably on other data sets (the generality is low). The challenge in building a useful tree is determining when to stop splitting, thus creating a less predictive model that is more general. This issue is an expression of the general machine learning tendency to overtrain an algorithm.

The easiest way to address this issue is to impose one or more stopping rules on the training process. Common stopping methods are

- Less than a minimum number of cases is included in the split;
- Maximum number of terminal nodes (leaves) has been reached.

After the tree building has been stopped, many algorithms begin “pruning back” the tree, by iteratively evaluating the “sensitivity” of the solution of the elimination of variables one at a time. The goal in pruning decision trees is to find the simplest model within a specified range of the highest accuracy, one that is equally as accurate (or nearly so) in predicting new cases.

*Model testing.* Many decision tree algorithms provide an option to split the input data stream into a training set and a testing set. If this option is enabled (and we strongly suggest that you do so), the algorithm iterative builds a number of candidate trees with the training set and tests each tree against the testing data set to measure the generality of the prediction. Then the algorithm can choose which tree has reasonable accuracy and good generality. If this facility is not available in your decision tree algorithm, you should split the trees outside the tool and test each candidate tree manually.

*Resampling.* If the form of testing described in the preceding paragraph appears to be a good idea, then you can understand the value in doing it many times on different random samples of the data set. The variation in the predictions among trees built on different resampled data sets is an expression of the model error, or the error that is due not to the noise in the data but rather is caused by the effect of sampling a particular set of cases. One random sample of cases may have a significantly different “view” of the response signal than another sample set. Various resampling methods will be discussed in Chapter 13.

*Large trees are problematical.* Decision trees built on complex patterns in large data sets can become quite large (unless controlled by a maximum-size stopping function). With modern computers, this size is not so much a computation problem as it is a comprehensibility problem. Complex trees are difficult to present to the “consumers” of the project results.