Looking for info about statistics?
We wrote the book on it.
And you can read it for free!

Popular Decision Tree: Classification and Regression Trees (C&RT)

Introductory Overview - Basic Ideas

OVERVIEW

C&RT, a recursive partitioning method, builds classification and regression trees for predicting continuous dependent variables (regression) and categorical predictor variables (classification). The classic *C&RT* algorithm was popularized by Breiman et al. (Breiman, Friedman, Olshen, & Stone, 1984; see also Ripley, 1996). A general introduction to tree-classifiers, specifically to the QUEST (Quick, Unbiased, Efficient Statistical Trees) algorithm, is also presented in the context of the *Classification Trees Analysis* facilities, and much of the following discussion presents the same information, in only a slightly different context. Another, similar type of tree building algorithm is CHAID (Chi-square Automatic Interaction Detector; see Kass, 1980).

CLASSIFICATION AND REGRESSION PROBLEMS

There are numerous algorithms for predicting continuous variables or categorical variables from a set of continuous predictors and/or categorical factor effects. For example, in *GLM (General Linear Models)* and *GRM (General Regression Models)*, we can specify a linear combination (design) of continuous predictors and categorical factor effects (e.g., with two-way and three-way interaction effects) to predict a continuous dependent variable. In *GDA (General Discriminant Function Analysis)*, we can specify such designs for predicting categorical variables, i.e., to solve classification problems.

Regression-type problems. Regression-type problems are generally those where we attempt to predict the values of a continuous variable from one or more continuous and/or categorical predictor variables. For example, we may want to predict the selling prices of single family homes (a continuous dependent variable) from various other continuous predictors (e.g., square footage) as well as categorical predictors (e.g., style of home, such as ranch, two-story, etc.; zip code or telephone area code where the property is located, etc.; note that this latter variable would be categorical in nature, even though it would contain numeric values or codes). If we used simple multiple regression, or some general linear model (*GLM*) to predict the selling prices of single family homes, we would determine a linear equation for these variables that can be used to compute predicted selling prices. There are many different analytic procedures for fitting linear models (*GLM*, *GRM*, *Regression*), various types of nonlinear models (e.g., *Generalized Linear/Nonlinear Models (GLZ)*, *Generalized Additive Models (GAM)*, etc.), or completely custom-defined nonlinear models (see *Nonlinear Estimation*), where we can type in an arbitrary equation containing parameters to be estimated. CHAID also analyzes regression-type problems, and produces results that are similar (in nature) to those computed by *C&RT*. Note that various neural network architectures are also applicable to solve regression-type problems.

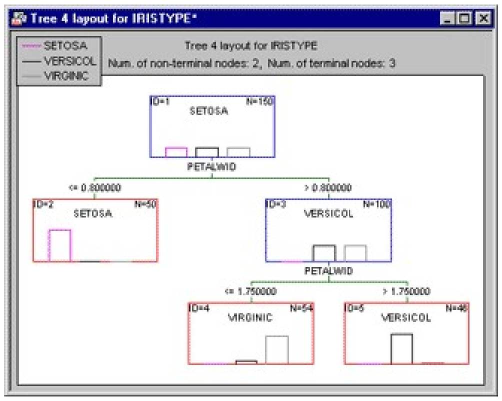
Classification-type problems. Classification-type problems are generally those where we attempt to predict values of a categorical dependent variable (class, group membership, etc.) from one or more continuous and/or categorical predictor variables. For example, we may be interested in predicting who will or will not graduate from college, or who will or will not renew a subscription. These would be examples of simple binary classification problems, where the categorical dependent variable can only assume two distinct and mutually exclusive values. In other cases, we might be interested in predicting which one of multiple different alternative consumer products (e.g., makes of cars) a person decides to purchase, or which type of failure occurs with different types of engines. In those cases there are multiple categories or classes for the categorical dependent variable. There are a number of methods for analyzing classification-type problems and to compute predicted classifications, either from simple continuous predictors (e.g., binomial or multinomial logit regression in *GLZ*), from categorical predictors (e.g., *Log-Linear analysis* of multi-way frequency tables), or both (e.g., via ANCOVA-like designs in *GLZ* or *GDA*). The *CHAID* also analyzes classification-type problems, and produces results that are similar (in nature) to those computed by *C&RT*. Note that various neural network architectures are also applicable to solve classification-type problems.

CLASSIFICATION AND REGRESSION TREES (C&RT)

In most general terms, the purpose of the analyses via tree-building algorithms is to determine a set of *if-then* logical (split) conditions that permit accurate prediction or classification of cases.

CLASSIFICATION TREES

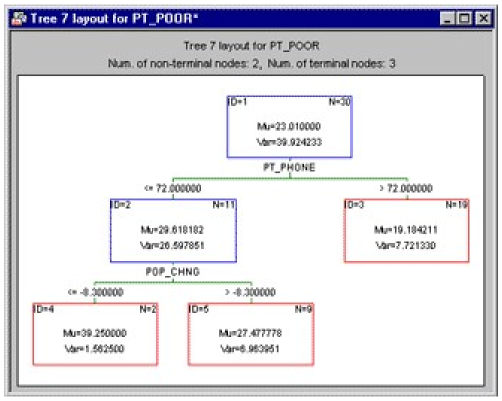
For example, consider the widely referenced Iris data classification problem introduced by Fisher [1936; see also *Discriminant Function Analysis* and *General Discriminant Analysis (GDA)*]. The data file *Irisdat* reports the lengths and widths of sepals and petals of three types of irises (Setosa, Versicol, and Virginic). The purpose of the analysis is to learn how we can discriminate between the three types of flowers, based on the four measures of width and length of petals and sepals. Discriminant function analysis will estimate several linear combinations of predictor variables for computing classification scores (or probabilities) that allow the user to determine the predicted classification for each observation. A classification tree will determine a set of logical if-then conditions (instead of linear equations) for predicting or classifying cases instead:



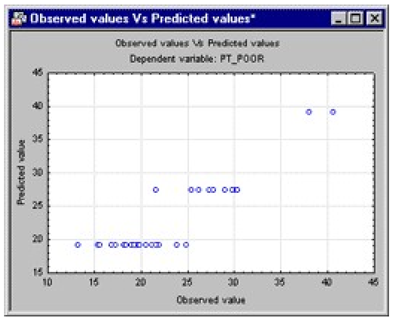
The interpretation of this tree is straightforward: If the petal width is less than or equal to 0.8, the respective flower would be classified as Setosa; if the petal width is greater than 0.8 and less than or equal to 1.75, then the respective flower would be classified as Versicol; else, it belongs to class Virginic.

REGRESSION TREES

The general approach to derive predictions from few simple if-then conditions can be applied to regression problems as well. This example is based on the data file *Poverty*, which contains 1960 and 1970 Census figures for a random selection of 30 counties. The research question (for that example) was to determine the correlates of poverty, that is, the variables that best predict the percent of families below the poverty line in a county. A reanalysis of those data, using the regression tree analysis [and v-fold cross-validation, yields the following results:



Again, the interpretation of these results is rather straightforward: Counties where the percent of households with a phone is greater than 72% have generally a lower poverty rate. The greatest poverty rate is evident in those counties that show less than (or equal to) 72% of households with a phone, and where the population change (from the 1960 census to the 170 census) is less than -8.3 (minus 8.3). These results are straightforward, easily presented, and intuitively clear as well: There are some affluent counties (where most households have a telephone), and those generally have little poverty. Then there are counties that are generally less affluent, and among those the ones that shrunk most showed the greatest poverty rate. A quick review of the scatterplot of observed vs. predicted values shows how the discrimination between the latter two groups is particularly well "explained" by the tree model.



ADVANTAGES OF CLASSIFICATION AND REGRESSION TREES (C&RT) METHODS

As mentioned earlier, there are a large number of methods that an analyst can choose from when analyzing classification or regression problems. Tree classification techniques, when they "work" and produce accurate predictions or predicted classifications based on few logical if-then conditions, have a number of advantages over many of those alternative techniques.

Simplicity of results. In most cases, the interpretation of results summarized in a tree is very simple. This simplicity is useful not only for purposes of rapid classification of new observations (it is much easier to evaluate just one or two logical conditions, than to compute classification scores for each possible group, or predicted values, based on all predictors and using possibly some complex nonlinear model equations), but can also often yield a much simpler "model" for explaining why observations are classified or predicted in a particular manner (e.g., when analyzing business problems, it is much easier to present a few simple if-then statements to management, than some elaborate equations).

Tree methods are nonparametric and nonlinear. The final results of using tree methods for classification or regression can be summarized in a series of (usually few) logical if-then conditions (tree nodes). Therefore, there is no implicit assumption that the underlying relationships between the predictor variables and the dependent variable are linear, follow some specific non-linear link function [e.g., see *Generalized Linear/Nonlinear Models (GLZ)*], or that they are even monotonic in nature. For example, some continuous outcome variable of interest could be positively related to a variable Income if the income is less than some certain amount, but negatively related if it is more than that amount (i.e., the tree could reveal multiple splits based on the same variable Income, revealing such a non-monotonic relationship between the variables). Thus, tree methods are particularly well suited for data mining tasks, where there is often little *a priori* knowledge nor any coherent set of theories or predictions regarding which variables are related and how. In those types of data analyses, tree methods can often reveal simple relationships between just a few variables that could have easily gone unnoticed using other analytic techniques.

GENERAL COMPUTATION ISSUES AND UNIQUE SOLUTIONS OF C&RT

The computational details involved in determining the best split conditions to construct a simple yet useful and informative tree are quite complex. Refer to Breiman et al. (1984) for a discussion of their CART® algorithm to learn more about the general theory of and specific computational solutions for constructing classification and regression trees. An excellent general discussion of tree classification and regression methods, and comparisons with other approaches to pattern recognition and neural networks, is provided in Ripley (1996).

AVOIDING OVER-FITTING: PRUNING, CROSSVALIDATION, AND V-FOLD CROSSVALIDATION

A major issue that arises when applying regression or classification trees to "real" data with much random error noise concerns the decision when to stop splitting. For example, if we had a data set with 10 cases, and performed 9 splits (determined 9 if-then conditions), we could perfectly predict every single case. In general, if we only split a sufficient number of times, eventually we will be able to "predict" ("reproduce" would be the more appropriate term here) our original data (from which we determined the splits). Of course, it is far from clear whether such complex results (with many splits) will replicate in a sample of new observations; most likely they will not.

This general issue is also discussed in the literature on tree classification and regression methods, as well as neural networks, under the topic of "overlearning" or "overfitting." If not stopped, the tree algorithm will ultimately "extract" all information from the data, including information that is not and cannot be predicted in the population with the current set of predictors, i.e., random or noise variation. The general approach to addressing this issue is first to stop generating new split nodes when subsequent splits only result in very little overall improvement of the prediction. For example, if we can predict 90% of all cases correctly from 10 splits, and 90.1% of all cases from 11 splits, then it obviously makes little sense to add that 11th split to the tree. There are many such criteria for automatically stopping the splitting (tree-building) process.

Once the tree building algorithm has stopped, it is always useful to further evaluate the quality of the prediction of the current tree in samples of observations that did not participate in the original computations. These methods are used to "prune back" the tree, i.e., to eventually (and ideally) select a simpler tree than the one obtained when the tree building algorithm stopped, but one that is equally as accurate for predicting or classifying "new" observations.

Crossvalidation. One approach is to apply the tree computed from one set of observations (learning sample) to another completely independent set of observations (testing sample). If most or all of the splits determined by the analysis of the learning sample are essentially based on "random noise," then the prediction for the testing sample will be very poor. Hence, we can infer that the selected tree is not very good (useful), and not of the "right size."

V-fold crossvalidation. Continuing further along this line of reasoning (described in the context of crossvalidation above), why not repeat the analysis many times over with different randomly drawn samples from the data, for every tree size starting at the root of the tree, and applying it to the prediction of observations from randomly selected testing samples. Then use (interpret, or accept as our final result) the tree that shows the best average accuracy for cross-validated predicted classifications or predicted values. In most cases, this tree will not be the one with the most terminal nodes, i.e., the most complex tree. This method for pruning a tree, and for selecting a smaller tree from a sequence of trees, can be very powerful, and is particularly useful for smaller data sets. It is an essential step for generating useful (for prediction) tree models, and because it can be computationally difficult to do, this method is often not found in tree classification or regression software.

REVIEWING LARGE TREES: UNIQUE ANALYSIS MANAGEMENT TOOLS

Another general issue that arises when applying tree classification or regression methods is that the final trees can become very large. In practice, when the input data are complex and, for example, contain many different categories for classification problems and many possible predictors for performing the classification, then the resulting trees can become very large. This is not so much a computational problem as it is a problem of presenting the trees in a manner that is easily accessible to the data analyst, or for presentation to the "consumers" of the research.

ANALYZING ANCOVA-LIKE DESIGNS

The classic (Breiman et. al., 1984) classification and regression trees algorithms can accommodate both continuous and categorical predictor. However, in practice, it is not uncommon to combine such variables into analysis of variance/covariance (ANCOVA) like predictor designs with main effects or interaction effects for categorical and continuous predictors. This method of analyzing coded ANCOVA-like designs is relatively new and. However, it is easy to see how the use of coded predictor designs expands these powerful classification and regression techniques to the analysis of data from experimental designs (e.g., see for example the detailed discussion of experimental design methods for quality improvement in the context of the Experimental Design module of Industrial Statistics).

Computational Details

The process of computing classification and regression trees can be characterized as involving four basic steps:

- Specifying the criteria for predictive accuracy
- Selecting splits
- Determining when to stop splitting
- Selecting the "right-sized" tree.

These steps are very similar to those discussed in the context of Classification Trees Analysis (see also Breiman et al., 1984, for more details). See also, Computational Formulas.

SPECIFYING THE CRITERIA FOR PREDICTIVE ACCURACY

The classification and regression trees (*C&RT*) algorithms are generally aimed at achieving the best possible predictive accuracy. Operationally, the most accurate prediction is defined as the prediction with the minimum costs. The notion of costs was developed as a way to generalize, to a broader range of prediction situations, the idea that the best prediction has the lowest misclassification rate. In most applications, the cost is measured in terms of proportion of misclassified cases, or variance. In this context, it follows, therefore, that a prediction would be considered best if it has the lowest misclassification rate or the smallest variance. The need for minimizing costs, rather than just the proportion of misclassified cases, arises when some predictions that fail are more catastrophic than others, or when some predictions that fail occur more frequently than others.

Priors. In the case of a categorical response (classification problem), minimizing costs amounts to minimizing the proportion of misclassified cases when priors are taken to be proportional to the class sizes and when misclassification costs are taken to be equal for every class.

The *a priori* probabilities used in minimizing costs can greatly affect the classification of cases or objects. Therefore, care has to be taken while using the priors. If differential base rates are not of interest for the study, or if we know that there are about an equal number of cases in each class, then we would use equal priors. If the differential base rates are reflected in the class sizes (as they would be, if the sample is a probability sample), then we would use priors estimated by the class proportions of the sample. Finally, if we have specific knowledge about the base rates (for example, based on previous research), then we would specify priors in accordance with that knowledge The general point is that the relative size of the priors assigned to each class can be used to "adjust" the importance of misclassifications for each class. However, no priors are required when we are building a regression tree.

Misclassification costs. Sometimes more accurate classification of the response is desired for some classes than others for reasons not related to the relative class sizes. If the criterion for predictive accuracy is Misclassification costs, then minimizing costs would amount to minimizing the proportion of misclassified cases when priors are considered proportional to the class sizes and misclassification costs are taken to be equal for every class.

Case weights. Case weights are treated strictly as case multipliers. For example, the misclassification rates from an analysis of an aggregated data set using case weights will be identical to the misclassification rates from the same analysis where the cases are replicated the specified number of times in the data file.

However, note that the use of case weights for aggregated data sets in classification problems is related to the issue of minimizing costs. Interestingly, as an alternative to using case weights for aggregated data sets, we could specify appropriate priors and/or misclassification costs and produce the same results while avoiding the additional processing required to analyze multiple cases with the same values for all variables. Suppose that in an aggregated data set with two classes having an equal number of cases, there are case weights of 2 for all cases in the first class, and case weights of 3 for all cases in the second class. If we specified priors of .4 and .6, respectively, specified equal misclassification costs, and analyzed the data without case weights, we will get the same misclassification rates as we would get if we specified priors estimated by the class sizes, specified equal misclassification costs, and analyzed the aggregated data set using the case weights. We would also get the same misclassification rates if we specified priors to be equal, specified the costs of misclassifying class 1 cases as class 2 cases to be 2/3 of the costs of misclassifying class 2 cases as class 1 cases, and analyzed the data without case weights.

SELECTING SPLITS

The second basic step in classification and regression trees is to select the splits on the predictor variables that are used to predict membership in classes of the categorical dependent variables, or to predict values of the continuous dependent (response) variable. In general terms, the split at each node will be found that will generate the greatest improvement in predictive accuracy. This is usually measured with some type of node impurity measure, which provides an indication of the relative homogeneity (the inverse of impurity) of cases in the terminal nodes. If all cases in each terminal node show identical values, then node impurity is minimal, homogeneity is maximal, and prediction is perfect (at least for the cases used in the computations; predictive validity for new cases is of course a different matter...).

For classification problems, C&RT gives the user the choice of several impurity measures: The Gini index, Chi-square, or G-square. The Gini index of node impurity is the measure most commonly chosen for classification-type problems. As an impurity measure, it reaches a value of zero when only one class is present at a node. With priors estimated from class sizes and equal misclassification costs, the Gini measure is computed as the sum of products of all pairs of class proportions for classes present at the node; it reaches its maximum value when class sizes at the node are equal; the Gini index is equal to zero if all cases in a node belong to the same class. The Chi-square measure is similar to the standard Chi-square value computed for the expected and observed classifications (with priors adjusted for misclassification cost), and the G-square measure is similar to the maximum-likelihood Chi-square (as for example computed in the Log-Linear module). For regression-type problems, a least-squares deviation criterion (similar to what is computed in least squares regression) is automatically used. Computational Formulas provides further computational details.

DETERMINING WHEN TO STOP SPLITTING

As discussed in Basic Ideas, in principal, splitting could continue until all cases are perfectly classified or predicted. However, this wouldn't make much sense since we would likely end up with a tree structure that is as complex and "tedious" as the original data file (with many nodes possibly containing single observations), and that would most likely not be very useful or accurate for predicting new observations. What is required is some reasonable stopping rule. In *C&RT*, two options are available that can be used to keep a check on the splitting process; namely Minimum n and Fraction of objects.

Minimum n. One way to control splitting is to allow splitting to continue until all terminal nodes are pure or contain no more than a specified minimum number of cases or objects. In *C&RT* this is done by using the option Minimum n that allows us to specify the desired minimum number of cases as a check on the splitting process. This option can be used when *Prune on misclassification error*, *Prune on deviance*, or *Prune on variance* is active as the *Stopping rule* for the analysis.

Fraction of objects. Another way to control splitting is to allow splitting to continue until all terminal nodes are pure or contain no more cases than a specified minimum fraction of the sizes of one or more classes (in the case of classification problems, or all cases in regression problems). This option can be used when FACT-style direct stopping has been selected as the *Stopping rule* for the analysis. In *C&RT*, the desired minimum fraction can be specified as the Fraction of objects. For classification problems, if the priors used in the analysis are equal and class sizes are equal as well, then splitting will stop when all terminal nodes containing more than one class have no more cases than the specified fraction of the class sizes for one or more classes. Alternatively, if the priors used in the analysis are not equal, splitting will stop when all terminal nodes containing more than one class have no more cases than the specified fraction for one or more classes. See Loh and Vanichestakul, 1988 for details.

PRUNING AND SELECTING THE "RIGHT-SIZED" TREE

The size of a tree in the classification and regression trees analysis is an important issue, since an unreasonably big tree can only make the interpretation of results more difficult. Some generalizations can be offered about what constitutes the "right-sized" tree. It should be sufficiently complex to account for the known facts, but at the same time it should be as simple as possible. It should exploit information that increases predictive accuracy and ignore information that does not. It should, if possible, lead to greater understanding of the phenomena it describes. The options available in *C&RT* allow the use of either, or both, of two different strategies for selecting the "right-sized" tree from among all the possible trees. One strategy is to grow the tree to just the right size, where the right size is determined by the user, based on the knowledge from previous research, diagnostic information from previous analyses, or even intuition. The other strategy is to use a set of well-documented, structured procedures developed by Breiman et al. (1984) for selecting the "right-sized" tree. These procedures are not foolproof, as Breiman et al. (1984) readily acknowledge, but at least they take subjective judgment out of the process of selecting the "right-sized" tree.

FACT-style direct stopping. We will begin by describing the first strategy, in which the user specifies the size to grow the tree. This strategy is followed by selecting FACT-style direct stopping as the stopping rule for the analysis, and by specifying the Fraction of objects that allows the tree to grow to the desired size. *C&RT* provides several options for obtaining diagnostic information to determine the reasonableness of the choice of size for the tree. Specifically, three options are available for performing cross-validation of the selected tree; namely Test sample, V-fold, and Minimal cost-complexity.

Test sample cross-validation. The first, and most preferred type of cross-validation is the test sample cross-validation. In this type of cross-validation, the tree is computed from the learning sample, and its predictive accuracy is tested by applying it to predict the class membership in the test sample. If the costs for the test sample exceed the costs for the learning sample, then this is an indication of poor cross-validation. In that case, a different sized tree might crossvalidate better. The test and learning samples can be formed by collecting two independent data sets, or if a large learning sample is available, by reserving a randomly selected proportion of the cases, say a third or a half, for use as the test sample.

In the *C&RT* module, test sample cross-validation is performed by specifying a sample identifier variable that contains codes for identifying the sample (learning or test) to which each case or object belongs.

V-fold cross-validation. The second type of cross-validation available in *C&RT* is V-fold cross-validation. This type of cross-validation is useful when no test sample is available and the learning sample is too small to have the test sample taken from it. The user-specified 'v' value for v-fold cross-validation (its default value is 3) determines the number of random subsamples, as equal in size as possible, that are formed from the learning sample. A tree of the specified size is computed 'v' times, each time leaving out one of the subsamples from the computations, and using that subsample as a test sample for cross-validation, so that each subsample is used (v - 1) times in the learning sample and just once as the test sample. The CV costs (cross-validation cost) computed for each of the 'v' test samples are then averaged to give the v-fold estimate of the CV costs.

Minimal cost-complexity cross-validation pruning. In *C&RT*, minimal cost-complexity cross-validation pruning is performed, if *Prune on misclassification error* has been selected as the *Stopping rule*. On the other hand, if *Prune on deviance* has been selected as the *Stopping rule*, then minimal deviance-complexity cross-validation pruning is performed. The only difference in the two options is the measure of prediction error that is used. *Prune on misclassification error* uses the costs that equals the misclassification rate when priors are estimated and misclassification costs are equal, while *Prune on deviance* uses a measure, based on maximum-likelihood principles, called the deviance (see Ripley, 1996). For details about the algorithms used in *C&RT* to implement Minimal cost-complexity cross-validation pruning, see also the Introductory Overview and Computational Methods sections of Classification Trees Analysis.

The sequence of trees obtained by this algorithm have a number of interesting properties. They are nested, because the successively pruned trees contain all the nodes of the next smaller tree in the sequence. Initially, many nodes are often pruned going from one tree to the next smaller tree in the sequence, but fewer nodes tend to be pruned as the root node is approached. The sequence of largest trees is also optimally pruned, because for every size of tree in the sequence, there is no other tree of the same size with lower costs. Proofs and/or explanations of these properties can be found in Breiman et al. (1984).

Tree selection after pruning. The pruning, as discussed above, often results in a sequence of optimally pruned trees. So the next task is to use an appropriate criterion to select the "right-sized" tree from this set of optimal trees. A natural criterion would be the CV costs (cross-validation costs). While there is nothing wrong with choosing the tree with the minimum CV costs as the "right-sized" tree, oftentimes there will be several trees with CV costs close to the minimum. Following Breiman et al. (1984) we could use the "automatic" tree selection procedure and choose as the "right-sized" tree the smallest-sized (least complex) tree whose CV costs do not differ appreciably from the minimum CV costs. In particular, they proposed a "1 SE rule" for making this selection, i.e., choose as the "right-sized" tree the smallest-sized tree whose CV costs do not exceed the minimum CV costs plus 1 times the standard error of the CV costs for the minimum CV costs tree. In *C&RT*, a multiple other than the 1 (the default) can also be specified for the SE rule. Thus, specifying a value of 0.0 would result in the minimal CV cost tree being selected as the "right-sized" tree. Values greater than 1.0 could lead to trees much smaller than the minimal CV cost tree being selected as the "right-sized" tree. One distinct advantage of the "automatic" tree selection procedure is that it helps to avoid "over fitting" and "under fitting" of the data.

As can be been seen, minimal cost-complexity cross-validation pruning and subsequent "right-sized" tree selection is a truly "automatic" process. The algorithms make all the decisions leading to the selection of the "right-sized" tree, except for, perhaps, specification of a value for the SE rule. V-fold cross-validation allows us to evaluate how well each tree "performs" when repeatedly cross-validated in different samples randomly drawn from the data.

Computational Formulas

In Classification and Regression Trees, estimates of accuracy are computed by different formulas for categorical and continuous dependent variables (classification and regression-type problems). For classification-type problems (categorical dependent variable) accuracy is measured in terms of the true classification rate of the classifier, while in the case of regression (continuous dependent variable) accuracy is measured in terms of mean squared error of the predictor.

In addition to measuring accuracy, the following measures of node impurity are used for classification problems: The Gini measure, generalized Chi-square measure, and generalized G-square measure. The Chi-square measure is similar to the standard Chi-square value computed for the expected and observed classifications (with priors adjusted for misclassification cost), and the G-square measure is similar to the maximum-likelihood Chi-square (as for example computed in the Log-Linear module). The Gini measure is the one most often used for measuring purity in the context of classification problems, and it is described below.

For continuous dependent variables (regression-type problems), the least squared deviation (LSD) measure of impurity is automatically applied.

ESTIMATION OF ACCURACY IN CLASSIFICATION

In classification problems (categorical dependent variable), three estimates of the accuracy are used: resubstitution estimate, test sample estimate, and v-fold cross-validation. These estimates are defined here.

Resubstitution estimate. Resubstitution estimate is the proportion of cases that are misclassified by the classifier constructed from the entire sample. This estimate is computed in the following manner:

$$R(d) = \frac{1}{N}\sum_{i=1}^{N} X(d(x_n) \neq j_n)$$

where X is the indicator function;

X = 1, if the statement $X(d(x_n) \neq j_n)$ is true

X = 0, if the statement $X(d(x_n) \neq j_n)$ is false

and *d* (x) is the classifier.

The resubstitution estimate is computed using the same data as used in constructing the classifier *d* .

Test sample estimate. The total number of cases are divided into two subsamples Z_1 and Z_2. The test sample estimate is the proportion of cases in the subsample Z_2, which are misclassified by the classifier constructed from the subsample Z_1. This estimate is computed in the following way.

Let the learning sample Z of size N be partitioned into subsamples Z_1 and Z_2 of sizes N and N_2, respectively.

$$R^{ts}(d) = \frac{1}{N_2} \sum_{(x_n, j_n) \in Z_2} X(d(x_n) \neq j_n)$$

where Z_2 is the sub sample that is not used for constructing the classifier.

v-fold crossvalidation. The total number of cases are divided into v sub samples $Z_1, Z_2, \ldots, Z_v$ of almost equal sizes. v-fold cross validation estimate is the proportion of cases in the subsample Z that are misclassified by the classifier constructed from the subsample $Z - Z_v$. This estimate is computed in the following way.

Let the learning sample Z of size N be partitioned into v sub samples $Z_1, Z_2, \ldots, Z_v$ of almost sizes $N_1, N_2, \ldots, N_v$, respectively.

$$R^{ts}(d^{(v)}) = \frac{1}{N_v} \sum_{(x_n, j_n) \in Z_v} X(d^{(v)}(x_n) \neq j_n)$$

where $d^{(v)}(x)$ is computed from the sub sample $Z - Z_v$.

ESTIMATION OF ACCURACY IN REGRESSION

In the regression problem (continuous dependent variable) three estimates of the accuracy are used: resubstitution estimate, test sample estimate, and v-fold cross-validation. These estimates are defined here.

Resubstitution estimate. The resubstitution estimate is the estimate of the expected squared error using the predictor of the continuous dependent variable. This estimate is computed in the following way.

$$R(d) = \frac{1}{N} \sum_{i=1}^{N} (y_i - d(x_i))^2$$

where the learning sample Z consists of $(x_i, y_i), i = 1, 2, \ldots, N$. The resubstitution estimate is computed using the same data as used in constructing the predictor d .

Test sample estimate. The total number of cases are divided into two subsamples Z_1 and Z_2. The test sample estimate of the mean squared error is computed in the following way:

Let the learning sample Z of size N be partitioned into subsamples Z_1 and Z_2 of sizes N and N_2, respectively.

$$R^{ts}(d) = \frac{1}{N_2} \sum_{(x_i, y_i) \in Z_2}^{N} (y_i - d(x_i))^2$$

where Z_2 is the sub-sample that is not used for constructing the predictor.

v-fold cross-validation. The total number of cases are divided into v sub samples $Z_1, Z_2, \ldots, Z_v$ of almost equal sizes. The subsample $Z - Z_v$ is used to construct the predictor d. Then v-fold cross validation estimate is computed from the subsample Z_v in the following way:

Let the learning sample Z of size N be partitioned into v sub samples $Z_1, Z_2, \ldots, Z_v$ of almost sizes $N_1, N_2, \ldots, N_v$, respectively.

$$R^{CV}(d) = \frac{1}{N_v} \sum_{v} \sum_{(x_n, y_n) \in Z_v} (y_i - d^{(v)}(x_n))^2$$

where $d^{(v)}(x)$ is computed from the sub sample $Z - Z_v$.

ESTIMATION OF NODE IMPURITY: GINI MEASURE

The Gini measure is the measure of impurity of a node and is commonly used when the dependent variable is a categorical variable, defined as:

$$g(t) = \sum_{j \neq i} p(j/t)\, p(i/t)$$

if costs of misclassification are not specified,

$$= \sum_{j \neq i} C(i/j)\, p(j/t)\, p(i/t)$$

if costs of misclassification are specified,

where the sum extends over all k categories. $p(j/t)$ is the probability of category j at the node t and $C(i/j)$ is the probability of misclassifying a category j case as category i.

ESTIMATION OF NODE IMPURITY: LEAST-SQUARED DEVIATION

Least-squared deviation (LSD) is used as the measure of impurity of a node when the response variable is continuous, and is computed as:

$$R(t) = \frac{1}{N_w(t)} \sum_{i \in t} w_i f_i (y_i - \bar{y}(t))^2$$

where $N_w(t)$ is the weighted number of cases in node t, w_i is the value of the weighting variable for case i, f_i is the value of the frequency variable, y_i is the value of the response variable, and $y(t)$ is the weighted mean for node t.

To index