Classification and Regression Trees: Introduction to CART

Lawrence Hubert

July 10, 2013

The Basic Setup

An $N \times p$ matrix (of predictors), $\mathbf{X}$; N is the number of subjects; p is the number of variables.

An $N \times 1$ vector (to be predicted; the "predictee"), $\mathbf{Y}$.

If $\mathbf{Y}$ contains nominal (categorical) data (with, say, K categories), we will construct a Classification Tree;

a "classifier" uses the data in $\mathbf{X}$ to place a row in $\mathbf{X}$ into a specific category.

If **Y** contains numerical values (that are not just used for labeling), we will construct a Regression Tree.

As an example. see the Medical Admissions (binary) tree in the SYSTAT manual on CART that you have.

Finding the Binary Splits

For each numerical (predictor) variable, order its values from smallest to largest and look at the $N - 1$ possible splits between adjacent values.

For each categorical (predictor) variable (with, say, K categories), evaluate all possible $2^{K-1} - 1$ splits of the category codes into two groups.

The best split is chosen to minimize the "impurity" of the two resulting subsets that are formed.

Default Measures of Node Impurity

For numerical **Y**, use the within sum-of-squares for the two groups formed by the split;

For categorical **Y**, use the sum of the Gini diversity indices ("gdi") over the two groups:

for one group with proportions, $p_1, \ldots, p_K$, over the K groups, $\text{gdi} = 1 - \sum_{k=1}^{K} p_k^2$.

Thus, gdi $= 0$ when one proportion is 1; gdi is maximal when the proportions are all equal.

A Little History and Terminology

Morgan and Sonquist were the first (in the 1960s) to suggest this type of "recursive partitioning";

they called it AID for "Automatic Interaction Detection".

The major R routine for this is in the "rpart" set of programs.

We could say it is "stagewise" and not "stepwise" – once a split is made, we don't revisit it.

Also, the procedure is myopic, in that it only looks one step ahead;

there is no guarantee of any overall optimality for the trees constructed.

We are looking for a good classifier that "stands up" to test samples and/or cross-validation.

Remember, a "good fit" does not necessarily mean a "good model".

The major reference and reason for current popularity: *Classification and Regression Trees* (1984) (Breiman; Friedman; Olshen; Stone)

How to Use the Tree for Prediction

For a numerical **Y**, we can use the mean of the values from **Y** within the terminal subsets (nodes);

For categorical **Y**, we can use the category with the greatest proportion (the majority or plurality) of observations in the terminal subsets (nodes).

We could also impose differential costs of misclassification or different prior probabilities of group membership – this is much like what we can do in using discriminant functions.

Confusion Errors

Suppose two categories in **Y** ("success" and "failure"):

	Failure Prediction	Success Predicted
Failure	a	b
Success	c	d

overall error: $(b + c)/(a + b + c + d)$
model errors: $b/(a + b)$; $c/(c + d)$
usage errors: $c/(a + c)$; $b/(b + d)$

How to Evaluate Accuracy

a) k-fold cross-validation (the default value of k is usually 10)

the extreme of $k = N$ is the "leave-one-out" option

b) bootstrap (about 1/3 of the observations are not resampled and can be used for cross-validation)
these are called the "out-of-bag" (OOB) observations

In either case, one "drops down" the tree the unsampled cases to see how well one does.

All of this is very engineering oriented. The emphasis is mainly on whether it "works" and not on the "why" or "how".

Thus, clever mechanisms to evaluate accuracy are crucial; cross-validation is central to the CART enterprise.

For the behavioral sciences, we typically are also interested in the predictive structure of the problem, i.e., the "how" and "why".

Issues

How "deep" should the tree be grown (the "goldilocks" problem):
too deep, we "overfit" and get unstable prediction;
not deep enough, we get inaccurate prediction.

Strategies:
a) Choose minimal leaf size by a cross-validation (sample reuse) mechanism;

b) Draw a deep tree and "prune" back to get to the best cross-validation level.

A tree with one case per terminal node is called "saturated".

Older Terminology

A "training sample" (we are training the learner or classifier if we have a categorical **Y**).

There is a "test sample" of new data not used in the training that serves the purposes of cross-validation (and to assess "shrinkage").

If we "drop" the training sample down the tree, we get the "resubstitution estimate of error".

This is an overly optimistic estimate since the same data used to obtain the tree now serves as the means to evaluate it.

Going to a saturated tree gives zero resubstitution error but it is terribly overfit and unstable;

that is the reason for pruning back, or evaluating leaf size through cross-validation.

Variable Combinations

We could include a variety of linear combinations of the original variables in the predictor set.

We get closer to a linear discriminant situation that uses separating hyperplanes.

Otherwise, splits are all perpendicular to the coordinate axes.

Surrogate Splits

If we have missing data on some predictor variable for an object, and we don't know which class the object should be assigned to when that predictor is used for a particular split, we can use a similar split on another variable that is "close" – we use these (surrogate) splits to assign the object to the class.

Extensions of CART to Tree Ensembles

Boosting – this refers to a variety of methods for reweighting hard to classify objects, and redoing the training.

Bagging – this stands for bootstrap aggregation; multiple trees are produced and averages are taken over the trees for prediction purposes; out-of-bag observations are used for evaluation.

Random Forests – use random subsets of the predictors to grow the trees.

Berk's Summary Comment

Random forests and boosting are probably state-of-the-art forecasting tools.