MART

When forward stagewise additive modeling (FSAM) is done using regression trees for the components of the additive model, the approach is referred to as *multiple additive regression trees*, or MART. This is what *TreeNet* does when creating a regression model, with a shrinkage factor incorporated to encourage *slow learning*. (The term to be added is multiplied by a constant between 0 and 1, resulting in only a partial adustment. Of course another term can be added to try to make a correction (but it too will be dampened). The slow progress leads to a lot of terms, but tends to improve performance.)

The predictive rule for a regression tree can be expressed as

$$T(\vec{x}) = \sum_{j=1}^{J} \gamma_j \, I(\vec{x} \in R_j),$$

where the R_j are the disjoint regions of the partition of the space (corresponding to the terminal nodes of the tree), and the γ_j are the predicted values for the various regions. If trees are the components used in FSAM, which were previously represented generically by

$$\beta \, b(\vec{x}; \gamma),$$

wlog we can take β to be 1 and can let $b(\vec{x}; \gamma)$ be $T(\vec{x})$, with the parameter γ including all of the γ_j and all of the information needed to specify the R_i (the splitting variables and values, and the information needed to specify the overall structure of the tree). The value of J is considered to be fixed in advance.

For the optimization of the parameters at each step in the FSAM procedure, with the squared-error and absolute error loss functions, given the R_j, the determination of γ_j is easy. (P. 319 of HTF seems to suggest that the absolute error loss fn leads to complications, but I really don't think that it does.) The hard part is finding the R_j which minimize the average loss for the training sample. In practice, this minimization isn't done, and an approximate solution which results from a greedy, top-down recursive partitioning algorithm is used instead.

The situation becomes more complicated if the Huber loss fn is used to provide a decent degree of robustness. Although a one-step approximation can be used to assign a predicted value to each region in the partition, performing the greedy recursive partitioning to identify a good set of regions takes longer. (Note: Sec. 10.10 of HTF describes how *gradient boosting* can be used to perform approx. numerical optimization with a general loss fn. Things are not necessarily so messy if some specific loss functions are considered.)

Typically, small trees are used at each step. If stumps are used, the resulting model is an additive "main effects" model, incorporating no interaction effects. Even though each tree component contributes a relatively simple step f'n response surface, when a large number of such surfaces are superimposed, the resulting step f'n response surface can well approximate a continuous response surface.

TreeNet's implementation of MART uses a random subsample of the training data at each step — this makes it run quicker and guards against overfitting (as does using the estimated generalization errors to select the right number of terms).

If you want to save a *grove file* so that you can score other data, you must specify this during the model setup. You cannot create and save a grove file after the model has been built.