Model Selection and Pruning

In general, nonparametric models are adaptive and can exhibit a high degree of flexibility that may ultimately result in overfitting if no measures are taken to counteract it. Although overfit models can achieve zero error on training data (provided they have a sufficiently large number of parameters), they will almost certainly perform poorly when presented with new observations or instances (i.e., they do not generalize well to the prediction of “new” cases). MARSplines tends to overfit the data as well. To combat this problem, it uses a pruning technique (similar to that in classification trees) to limit the complexity of the model by reducing the number of its basis functions.

MARSplines as a Predictor (Feature) Selection Method

The selection of and pruning of basis functions in MARSplines makes this method a very powerful tool for predictor selection. The MARSplines algorithm will pick up only those basis functions (and those predictor variables) that make a “sizeable” contribution to the prediction. The Results dialog of the Multivariate Adaptive Regression Splines (MARSplines) module will clearly identify (highlight) only those variables associated with basis functions that were retained for the final solution (model).

Applications

MARSplines has become very popular recently for finding predictive models for “difficult” data mining problems, i.e., when the predictor variables do not exhibit simple and/or monotone relationships to the dependent variable of interest. Because of the specific manner in which MARSplines selects predictors (basis functions) for the model, it generally does well in situations in which regression-tree models are also appropriate, i.e., where hierarchically organized successive splits on the predictor variables yield accurate predictions. In fact, this technique is as much a generalization of regression trees as it is of multiple regression. The “hard” binary splits are replaced by “smooth” basis functions.

A large number of graphs can be computed to evaluate the quality of the fit and to aid with the interpretation of results. Various code generator options are available for saving estimated (fully parameterized) models for deployment in C/C++/C#, Visual Basic, or PMML.

The MARSplines Algorithm

Implementing MARSplines involves a two-step procedure that is applied successively until a desired model is found. In the first step, we build the model (increase its complexity) by repeatedly adding basis functions until a user-defined maximum level of complexity is reached. (We start with the simplest—the constant; then we iteratively add the next term, of all possible, that most reduces training error.) Once we have built a very complex model, we begin a backward procedure to iteratively remove the least significant basis functions from the model, i.e., those whose removal leads to the least reduction in the (least-squares) goodness of fit.