Summary and Problem Statement

This exercise extends the work from assignment 3, using data from Boston on home values. Multiple decision tree regression models are fit to the data and the results evaluated based on cross-validation testing; the root mean-squared error value (RMSE) is used as a comparative index of fit. The end goal is to recommend a suitable model to a real estate company for their use in assessing home values.

Methodology

The data are complete. The distribution of the response variable (mv) is skewed[1]. Data were scaled in the prior exercise, they are scaled here for consistency in comparisons. A variety of models were fit; several regression and decision tree types were tried. Models are compared using the average RMSE result for a 10-fold cross-validation pass. Finally, feature importance is reported for those tree models which have that attribute. Max-features="log2" and ="auto" were tried; "log2" was selected as being less likely to overfit the models. For gradient boost regression, subsample =.5 was tried, it did not show much change over the default of 1.0.

Code Overview

We 've used this data before, so only basic exploration is done here. The code starts by initializing a list variable with the methods to be used and all required parameters for each method. Next, the data are iteratively split into training and test "folds"; a total of 10 folds were used in this exercise. The program then iterates over the list of regression methods, once for each fold. Per-fold summary data are output for each model[2] which include the R^2 value and RSME. Finally, an average of the per-fold results for each method is output[3]. Finally, the feature importance is output for 3 decision tree models which support that attribute. Code to run over unscaled data is commented out, but left in as a reference for the student.

Results and Recommendations

The gradient boosting method with a learning step of .1 had smallest RSME and the best R^2, followed by the AdaBoost and bagging ensemble. The importance of tuning parameters can be seen in the difference between a 1.0 and 0.1 learning rate for the gradient boosting method. Details of feature importance varies from model to model[4], but "lstat" and "rooms" are the top across all models. For most models, the importance number drops sharply after the top 2, however this is not the case for the gradient descent model. Its feature importance drops more gradually feature to feature. I interpret this as meaning in a gradient boost model each feature is contributing more importance than in say, a random forest model. In my models, it looks like the bottom few features could be removed from the random forest without greatly impacting the model performance,

I recommend using the gradient boost regression model, with the parameters used in the ".1" model; changing the subsample value didn't improve results, so I would use the default. This model had the smallest RSME of all the models tried, and had the added benefit of the highest R^2 value. Having the least error and accounting for the most variability in data makes this the preferred model in my opinion. From all the models, we see 'lstat' and 'rooms' are the most important explanatory variables, with the others having different importance levels based on the model. This makes sense. The 'lstat' value relates to the economic make-up of the neighborhood, and number of rooms is always key in a home price.

[1] See mv-dist.pdf

[2] See console_output.html for examples of the per-fold output

[3] See console_output.html

[4] See features.pdf for the easy to read version, or console output for the raw dump