Classification Model Selection

Given the dataset that we had was processed very similar to our Seattle dataset, a similar methodical approach to finding the best model was utilized. As before, we fit a naive bayes model to function as our baseline for comparison. We fit a Naive Bayes model with *review_scores_rating* as our response variable. A 5 fold cross validation returned a misclassification error on our validation set of 28.22%. There was definite improvement here, and with the similar idea that support vector machines fit with different kernel functions could separate the two classes effectively, we decided to fit an SVM.

Three SVM models were fit with three different kernel functions: a linear, polynomial and radial SVM. 5 fold cross validation returned misclassification errors of 25.50%, 28.22% and 22.65% respectively. The radial kernel, as expected, performed best on the data since it is the kernel with the most flexibility. Interestingly, the polynomial kernel SVM performed worse than the linear kernel SVM, indicating that a linear fit would perform better on the data. Our next course of action would be to fit the data through a gradient boosting technique and a random forest. These two techniques were chosen because they produced some of the best results in our previous analysis and logically, we assumed they would be productive on the Boston dataset as well.

The boosting method utilizing the AdaBoost learner was fit on the same formula as the other techniques and its cross-validated misclassification error was 22.98%. The Radial SVM performed better than the boosted tree technique. It's possible that the data is especially amenable to a complex decision boundary that the radial kernel provides. The ensemble learner (boosting) wasn't as productive at classification.

Finally, we fit a random forest on the training data with *review_scores_rating* as the response variable. Since the boosted method didn't do substantially better than the radial SVM, we hypothesized that random forest wouldn't do much better than the boosting technique. The cross-validated misclassification error for the random forest was 21.74%. It performed the best out of all techniques tried. We tuned the random forest for the number of variables randomly sampled as candidates for the best possible split. The tuned classifier had marginally better misclassification error of 21.23%.