Random Forests and Boosted Trees

The random forests algorithm was proposed formally by Brieman (2001). It combines the concept of random subspaces and bagging (discussed further in Chapter 13). The random forests algorithm trains a number of trees on slightly different subsets of data (bootstrap sample), in which a case is added to each subset containing random selections from the range of each variable. This group of trees are similar to an ensemble (also discussed further in Chapter 13). Each decision tree in the ensemble votes for the classification of each input case.

Following are the steps in tree growth in random forests:

1. A random sample of the number of cases is taken. Subsequent samples for other trees are done with replacement (no case is left out, even the ones included in building the previous tree).
2. A subset of variables (the number of which is represented by the term $m$) is chosen, being much less than the number of variables, and the best split (based on the Gini score) is determined on this subset of variables. Increasing the value of $m$ increases the correlation between trees (bad) and increases the predictive power of the tree (lessens the error rate), while decreasing the value of $m$ does the opposite for both. There is a fairly wide “happy medium” in between values of $m$ that are too low and too high. The $m$-number is the only setting in the algorithm to which the model is sensitive.
3. About one-third of the cases are used to create the out-of-sample testing data set. This testing data set is used to compute the prediction error rate. The average error rate is calculated from all trees built.
4. Variable importance is calculated by running the out-of-sample data set down the tree, and then the number of votes for the predicted class are counted. Then the values in each variable are randomly changed and run down the tree independently. The number of votes for the tree with the changed variable is subtracted from the number of votes for the unchanged tree to yield a measure of effect. Finally, the average effect is calculated across all trees to yield the variable importance value.

Advantages of Random Forests

• The random forests algorithm has relatively high accuracy among algorithms for classification.
• It can handle very large data sets with hundreds (even thousands) of variables.
• It provides an estimate of variables’ importance, like neural nets.
• It has a robust method for handling missing data. The most frequent value for the variable among all cases in the node is substituted for the missing value.
• It has a built-in method for balancing unbalanced data sets (one class much rarer than other classes). We will revisit this concept in Chapter 16.
• It runs fast! Hundreds of trees with many thousands of cases with hundreds of variables can be built in a few minutes on a personal computer.