Chapter 8: Tree-Based Methods

Tree-based methods for regression and classification involve the stratification of the predictor space into a number of simple regions. As a result, to make a prediction for a given observation, we typically use the mean or the mode of the training observations to which it belongs.

The set of splitting rules used to segment the predictor space can be summarized in a tree. These types of approaches are known as decision tree methods. We will review both regression and classification trees.

Regression Trees

There two steps to the process of building a regression tree:

Step 1: Divide the predictor space into distinct and non-overlapping regions.

Step2: For every observation that falls into a specific region, we make the same prediction, which is the mean of the response values for the training observations in this region.

The regions could have any shape. The algorithm usually chooses to divide the predictor space into high-dimensional rectangles or boxes for simplicity and interpretability of the resulting predictive model.

We use the top-down, greedy computational approach known as recursive binary splitting for efficiency. The approach is top-down because it begins at the top of the tree (all observations belong to a single region as a starting point) and then successively splits the predictor space.

Each split is defined with two new branches further down on the tree. It is greedy because at each step of the tree-building process, the best split is made at that particular split, rather than looking ahead and picking a split that will lead to a better tree in some future step.

The process described above may produce good predictions on the training data, but it will likely overfit the data, leading to poor test performance.

We need to define a smaller tree with fewer regions. This will lead to lower variance and better interpretation at the cost of a little bias. We can introduce a tuning parameter for building a regression by constructing the following algorithm:

- Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

- Apply cost complexity pruning to the large tree and obtain a sequence of best subtrees, as a function of the tuning parameter.

- Use k-fold cross-validation to choose the value of the tuning parameter.

- Return the subtree that corresponds to the chosen value of the tuning parameter.

Classification Trees

A classification tree is very similar to a regression tree, except that it is used to predict a qualitative response rather than a numerical one.

For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs.

We use recursive splitting to grow a classification tree by observing the classification error rate, which is simply the fraction of the training observations in that region that do not belong to the most common class.

There are also two other measures to evaluate a classification tree: the Gini index (equation 8.6), which is a measure of total variance across all classes and cross-entropy (equation 8.7).

Advantages and Disadvantages of Trees

Here are some of the advantages in using tree based methods:

- They are easy to explain.
- Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches.
- Trees can be displayed graphically.
- Trees can easily handle qualitative predictors without the need to create dummy variables.

There are also two major disadvantages in using tree based methods:

- Trees generally don't have the same level of predictive accuracy as some of the other regression and classification methods.
- Trees can be very non-robust. A small change in the dataset can cause a large change in the final estimated tree.

Bagging, Random Forests, Boosting

We can use bootstrap to improve decision trees high variance component. **Bagging** is a bootstrap procedure of reducing the variance.

To apply bagging to regression trees, we simply construct B regression trees using B bootstrapped training sets and average the resulting predictions.

These trees are grown deep, and are not pruned. Hence each individual tree has high variance but averaging these B trees reduces the variance.

We can also apply bagging to predict a quantitative outcome Y. For a given test observation, we can record the class predicted by each of the B trees, and then take a majority vote: the overall prediction is the most commonly occurring class among the B predictions.

We can estimate the test error by utilizing the fact that each bagged tree makes use of around two-thirds of the observations. The remaining one-third is not used to fit a bagged tree and we refer to it as the out-of-bag observations (OOB).

If an observation belongs to OOB, we can predict the response for this observation using all of the constructed trees. We can then average these predictions to obtain a single prediction for a specific observation.

The resulting OOB error is a valid estimate of the test error for the bagged model.

Random forests provide an improvement over bagged trees by way of a small tweak to disassociate the trees. As in bagging, we build a number of decision trees on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of predictors is chosen as split candidates from the full set of predictors.

Finally, we can also apply **boosting** to improve bagging by sequentially creating trees. Each tree is grown by using information from previously grown trees. Boosting doesn't require bootstrap sampling, but each tree is fit on a modified version of the original data set.

Boosting has three tuning parameters: the number of trees, the shrinkage parameter and the number of splits in each tree.

Assignments for class discussion

Lab: Decision Trees