1 Tree Based Models

The idea behind this family of methods is to segment the region into subspaces and the prediction for any subspace is made by taking the *mean* or *mode* of the response in that region.
A simple decision tree may not give performance at par with linear regression or generalized additive models, but when combined with techniques like bagging and boosting, multiple trees can yield significant reduction in bias.
A simple decision tree finds its utility in being relatively simpler in interpretation than it's derived non-linear family.

1.1 Regression Trees

The algorithm behind building trees consists of two basic steps

1. Divide the predictors $X_1, X_2, \ldots, X_n$ into J **distinct** and **non-overlapping** regions $R_1, R_2, \ldots, R_J$.
2. For making a prediction in the region R_j, simply take the **response mean** of all the data points following in that region.

With the above definition, the loss becomes

$$RSS = \sum_{j=1}^{J} \sum_{i \in R_J} (y_i - \bar{y}_{R_J})^2$$

But minimizing this loss is infeasible in practice as the total number of possible partitions are too large !

To overcome this, we build the trees in a greedy top down approach called **recursive binary splitting**. This method aims to find the best possible split at each level in a greedy manner.

1.1.1 The Algorithm

To start off, the alogithm will try to split the data into two regions $\{X|X_j < s\}$ and $\{X|X_j \geq s\}$ using the values of predictor X_j and the search is performed across all the predictors to choose the one which minimizes the RSS. Mathematically

$$R_1(j,s) = \{X|X_j < s\} \text{ and } R_2(j,s) = \{X|X_j \geq s\}$$
$$\underset{j,s}{\text{minimize}} \sum_{i:x_i \in R_1} (y_i - \bar{y}_{R_1})^2 + \sum_{i:x_i \in R_2} (y_i - \bar{y}_{R_2})^2$$

where $\bar{y}_{R_1}$ and $\bar{y}_{R_2}$ are the response mean in the region R_1 and R_2 respectively.

The same algorithm is run recursively. The only change that is made that after the first split, we now wolr independently on two separate regions. For each region, we will not be able to consider all the possible values of s since the region has been restricted. Otherwise, the RSS term will remain the same.

The process is stopped when a pre-determined stopping criteria is reached. For making a prediction, we will first determine the region in which the test observation falls and then make the prediction based on the mean of the training samples in this region.

1.1.2 Tree Pruning

The tree build using above strategy can have good results on the training set, but will yield poor performance on the test set due to the high complexity. A better strategy is to build **short trees which are less complex and produce lower variance at cost of little bias** and are more interpretable.

A simple strategy can be to build only when there is a decrease in RSS above a threshold. This approach can be short-sighted as some future good branches can be missed.

Tree pruning is to **build a large tree and then shrink it back to obtain a subtree**. The RSS of a subtree can be found via cross validation. This approach is expensive for all possible subtrees.

Cost complexity Pruning or **weakest link pruning** is an approach to overcome this problem. For a non-negative parameter α, there exists a subtree T of the large tree T_0 such that the following is minimized

$$RSS = \sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \bar{y}_{R_m})^2 + \alpha |T|$$

where $|T|$ is the number of terminal nodes in the tree. The formulation is similar to lasso and here we are controlling the complexity of the tree via a parameter α. Larger the α, the less number of terminal nodes there would be in the tree.

The algorithm can then be summarized as follows

1. Use *binary recursive splitting* to obtain a large tree on the data set. Stop only when at a terminal node, the number of observations is less than a minimum.
2. Apply *cost complexity pruning* in order to obtain a small tree T as a function of the cost parameter α (fix a set of α and get a set of subtrees).
3. Use K-fold cross validation to choose α. For each of the folds $k = 1, 2, \ldots, K$
   (a) Repeat steps 1 and 2 excluding the data from the kth fold to obtain a family of subtrees as a function of α.
   (b) Evaluate the mean predicted validation error as a function of α.
   Choose the value of α that minimizes the average error across all the folds.
4. Select the substree from step 2 that corresponds to the chosen value of α.