Cem Ergin

COMP 421 HW#5 Report:

Before implementing I carefully read chapter 9 from our book and saw that Classification Trees and Regression Trees are very much alike in principle. The suggested that algorithm explained in its univariate Classification Tree section can be modified to create a Regression Tree quite easily. I followed this suggestion and constructed my algorithm similar to the one explained there.

I constructed 2 classes for this implementation. The first and more important one is called TreeNode and it takes advantage of binary tree structure to split data points recursively. The second class DecisionTree works as a wrapper class around TreeNode for ease of use.

TreeNode is initialized with a pruning value which it uses throughout its implementation. It has a fit(X) function that does most of the work by examining the input data. It first checks is the length of data is smaller than the pre-pruning value declared upon initialization. If the length is smaller it creates a leaf node and sets its prediction value as the mean of Y values the data. Otherwise, it uses the x values in the data to find suitable locations to split and creates its left and right nodes recursively until leaf nodes are reached.

To predict, TreeNode starts from Root Node and checks if split value of nodes exists. If it exists, predict\_one function applies the split rule and continues on checking the appropriate child nodes until a leaf node is reached where a prediction value is stored.

Like the previous homework, the RMSE values of the sample runs is affected drastically by the data points used to training and testing the algorithm. In runs where the training data resembles testing data more, the resulting RMSE was lower. I got a wide range of RMSE values over different sample runs but overall, I observed that RMSE tends to get lower as the pruning constant increases. The pruning values in the range of 1 to 6 seemed to give the highest RMSE values which suggests that the overfitting occurs.