Write-Up

Ibraheem Khan and Matthew Alighchi

September 26, 2019

1 DT_train_binary Implementation Details

DT_train_binary takes in our globally available X and Y along with some max-depth and returns a binary tree of said depth based on the information gain algorithm. It does so by splitting on three cases: the case where max_depth is 0, -1, and greater than 0. For the case of having max_depth being 0 it simply returns a binary tree having a single node whose label_prediction is just the average of the label set. For the case of max_depth being greater than 0, it recursively runs best_split_binary which computes the best feature to split on at the given node and then generate children nodes based on the best split. When max_depth is -1 it does something similar but instead recurses not until some specified number of times but rather recurses indefinitely until the feature list (which is the set of features for which we have not yet split on) expires.

Now, best_split_binary works by first creating various lists and lists of lists for all the data and organizes them based on some hypothetical feature split. That is, given some node, it supposes what if the node were to split its data on some feature, then places the would-be data in its left and right "children." It doesn't physically do this, but pretends to do this via the aforementioned lists and lists of lists. This way, for example, the label set is organized in a list of lists such the first list in the leftLabels is the label set of the left node should the algorithm split on the first feature in the feature list. Following this reorganization of our data, the function computes our information gain, puts them into a similarly organized list, converts that into a numpy array, and finds the argmax. This argmax tells us correspondingly what index in the feature list is the best feature to split on. Finally, this function creates the two children nodes based upon this optimal split.

The information gain function works by relying on the entropy function which is a standard implementation of the mathematical formula for entropy. The information gain implementation, too, is standard conversion from math to code, however we test edge cases for when our code may not work. For example, if our label sets are empty then our information gain must be 0.

2 DT_test_binary Implemenation Details

3 DT_train_binary_best Implementation Details

4 Testing DT_train_binary, DT_test_binary, and DT_train_binary_best

5 A Forest of Decision Trees

6 DT_train_real, DT_test_real, DT_train_real_best Implementations

7 Testing DT_train_real