Supervised Learning: Decision Trees

April 12, 2021

1 Motivation

Assume we have 2 points of data for each person: Sex (M/F) and Occupation (Work/Study). What app (Game, Social Network, Messaging) should we recommend to each person? We know that we need to create some sort of flow chart (if ____ and _____ then recommend _____) that tells us how to narrow down which app to recommend, but this poses a problem. Obviously our flow chart, or decision tree, needs to have (at least, but most likely) 2 "layers" to them because we have 2 different sets of data to them. Which feature do we put near the top of our tree (thus the first thing we separate our data by)?

2 Intuition

To be the most efficient, we want to "eliminate" the most data with every decision we make. If over half of our data shows that people who Study as their occupation, results in the recommended app being a Game, then for every new data point, we can sort it immediately if the person is a Student. So, we put that at the top of the tree, and factor in sex afterwards.
We need to be able to do this mathematically. To do this, we will introduce the concept of entropy.

3 Entropy

Entropy is the measure of the number of ways we can order distinguishable items in an area. For each distinguishable item x_i,
Entropy:= $-\sum P(x_i)log(P(x_i))$, where $P(x_i)$ is the probability of drawing item x_i from the collection X.
We want each group at the end of our decision tree to have as little entropy as possible, because if entropy is zero, then all items in the collection are the same.

3.1 Using Entropy

When we split our data points by the classification we are using, we want the sum of the entropy each group in the split to be as small as possible. In other words, the entropy of the parent collection minus the sum of the entropies of each subcollection should be as large as possible (becasue the latter value is supposed to be as small as possible). In other words, we choose the feature that we want at the root node of the tree by calculating
$EntropyGain := Entropy(ParentCollection) - \sum P(Child_i)log(P(Child_i)$
for each feature, and we take the largest value and put that at the top of the tree.

3.2 Avoiding complexity

Trees can get extremely complex. They can have many leaf nodes or become too deep, etc. So we set limits on what our tree should look like, max, such that we get a simpler decision tree. So we can change the following hyperparameters to ensure our tree generalizes well:
minimum samples split: smallest number of points in a parent node to warrant splitting the group into two

or more leaf nodes.
minimum samples per leaf: smallest number of points we can allow in a resulting leaf node such that we can allow the split to happen.
maximum depth of tree: the limit on how high our tree can be. Put another way, its the most number of decreasing subsets of the starting set we can have.

The goal here is to have a tree generalize, so you won't necessarily end up with leaf nodes that all achieve an entropy of 0, because if you made that your goal, you could get 100% accuracy just by giving each point a leaf node.