CS422 Project 1

Nicholas Ang

September 2022

Decision Trees

I implemented my decision trees recursively. It felt the most natural to set up the tree recursively. The list starts off empty and recursively adds to it by calling the training function on the left side of the tree and then the right side. I chose to use lists of lists to store my tree. It was easy to store the index of the path, the label values, and where the samples make up the training. However, indexing the values and extracting the information was slightly tougher since I had to make sure the index was correct when taking out samples that were already looked at. I would say I am fairly satisfied with the use of lists. Although it was a bit difficult to grasp, it worked out in the end. Possibly using linked lists and nodes would have been better as it could have been easier to store data such as the label and samples more separately.

Random Forests

The individual decision trees might have had a large variance in their accuracy because the samples they are trained on are random. For example, one tree could have been trained on 30 samples in a way that every sample tested would go to the same part of the tree and that would not result in the correct sample being labeled. Another tree could be as deep as possible and try to classify based on what it has memorized. I would say that testing depths of trees and taking the highest accuracy depth would reduce variance. Too low of a depth would cause the tree to produce no relevant information and too high of a depth would cause the tree to not be able to generalize new data. Having a depth in between reduces variance and provides a good level of accuracy.

It is beneficial to use an odd number of trees for the forest because it prevents ties from occurring. The random forest makes a prediction based on the majority prediction of the trees. If an even number of trees are used, there could be situations where an even split occurs. The random forest would not know which side to predict and would have to randomly choose one. This results in a more inaccurate result as it could predict one side when it should be the other side.

Overall Python

I felt very uncomfortable working in Python. It was strange not using semicolons and curly brackets on everything. Dictionaries were very confusing to me. 2D arrays and shrinking the array to calculate the next addition to the decision tree was tough. I constantly got errors on bad indexes and invalid values. Selecting which column to access and then splitting the column up made it very difficult to visualize what I was doing. I often had to print out the list and the whole array to see which values I was accessing. I think that I should continue practicing basic python functions and slowly work my way up to more difficult concepts. I feel that using classes and better data structures will help me in future projects.