Irmak Akyeli

218036902 Sec2

Question3

calculateEntropy function implements the given formula in the homework using cmath library to calculate log. This class first finds the number of total sample number, then it goes over all the sets to find the probability of the classes. Then it calculates entropy and in an array of size n, it has time complexity O(n^2).

calculateInformationGain also simply implements the given formula. Firstly, it goes over the labels array to find the maximum number which is the number of classes we will use. After it computes the number of classes it goes over the whole data array to compute the probability of the classes seen in the left subtree, right subtree and the parent node. Therefore, it requires two pass over the array to get every unknown to calculate the formula given. It then goes on to call the calculateEntropy function three times. This function with an array size taken as n also has time complexity O(n^2).

The print function recursively prints the decision tree. It calls itself with the children pointers until it encounters NULL. It has time complexity O(log(n)).

The predict function traverses the tree with an external pointer which starts pointing the root and going down on the tree until it reaches a leaf node. It also has time complexity O(log(n)).

The test function calls the predict function number of sample times and therefore it has O(n) * O(log(n)) complexity which makes O(nlog(n)).

The train function firstly parse the given file. For a file with n samples and m features it takes O(nm) time do this. Later on it takes another O(n) time to initialize the data array and O(n + m) time to initialize usedSamples and usedFeatures. Later on, it enters the recursive train function and calculates every possible information gain for all the unused features in O(mn^2) time. Then it compares them to find the maximum information gain and creates the first node. It does the same thing until information gain is less than 0 which means our tree has reached to all leaf ends. This also means that the nodes are pure and have their class numbers. There are m possible non-leaf nodes and every time the function calls itself.

It will also have to perform this action recursively to construct our tree, and it would have a worst case scenario that takes log(n) times. Therefore, the overall time complexity of the train function is O(mn^2log(n)).