**Programming assignment 1: Decision Tree**

Part 1:

The decision tree we built is univariate and binary and is split by the idea of Information gain (Entropy).

At first, we tried to build a decision tree that was n-ary based on how many unique values the attributes have. For example, since attribute “location” has five values, the node split by “location” would have five children nodes. However, it is complex to build a tree of nodes with many branches, leading to difficulties in getting the results after inputting test data. Besides, it also against the rule of making simple decisions, therefore we decided to construct a binary decision tree.

The data structure we use is tree which containing node value, pointer to ‘true’ child and pointer to ‘false’ child. Node value is a list including attribute and value used to filter. For instance, “['Occupied', 'Low']” means question “Is attribute ‘Occupied’ low?”. If yes, then ‘true’ child is returned, otherwise ‘false’ child is returned.

The tree is split based on the node attribute. At each node, we choose the attributes that can result in minimum entropy (maximum information gain) by looping through all the attributes. Since it is the binary tree, further looping through the unique values of each attribute(column) is needed. To make code more organized, function “column\_least\_entropy” was wrote to calculate the lowest entropy of each column. Afterwards, the train set is split into two subsets and the above process is repeated on each of them until the subset has the same value or no further split can be made.

Some optimizations were made to make to eliminate unnecessary calculations/loops. For example, once we get the zero entropy based on a certain split, then it should be returned as the best split of that node. Additionally, to increase the readability of the program, it was reorganized and some of variables/functions were renamed.

The biggest challenge was due to neglect the stop point of the loop. At the beginning, the program worked well, but then the error occurred at the specific stage. After careful inspection, we finally found that it was caused by the different results but with the same attributes set (index 17,20). To deal with this issue, we decided to make the predictions based on the majority of label if such case happens. Nonetheless, if each label has equal counts, then a random choice of ‘Yes’ and ‘No’ will be made when predicting.

Finally, the decision tree is printed as following:



The prediction for test data (occupied = Moderate; price = Cheap; music = Loud; location = City-Center; VIP = No; favorite beer = No) is “Yes” based on the above tree.

<https://www.geeksforgeeks.org/decision-tree/>

