DSCI 552 – Assignment 1: Decision Tree

Team Members	Sanjana Gopnal Swamy (gopnalsw@usc.edu)
	Amit Sankhla (asankhla@usc.edu)

Part 1: Implementation:

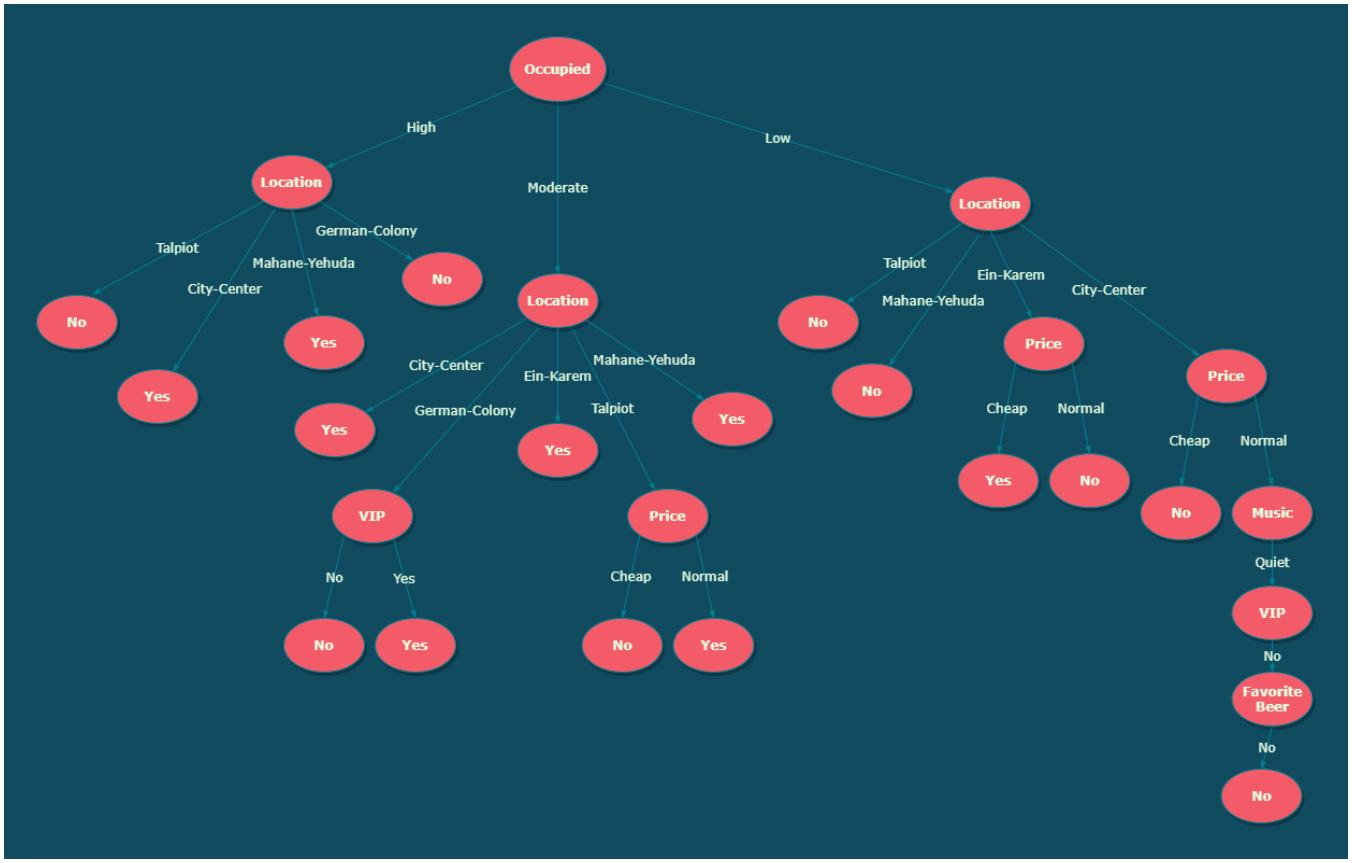
- Implementation is in python language.
- Used python list/dict to store the attributes
- The data file is stored in the same folder as the python file.
- A class DecisionTree to store features, data and root of the decision tree.
- A class TreeNode to store name of the attribute, its child nodes and its index. For the leaf nodes, the index is set to –1.

The format of the generated decision tree is as follows:

To display the decision tree, we have used DFS approach where starting from root node and having value : child_node pair at each level and making each level distinguishable by current_level*5 spaces on its left. The leaf nodes are marked either Yes or No. Below is a snapshot for it:

```
<-----------Tree----------->
Occupied
      High :  Location
            Talpiot :  No
            City-Center :  Yes
            Mahane-Yehuda :  Yes
            German-Colony :  No
      Moderate :  Location
            City-Center :  Yes
            German-Colony :  VIP
                  No :  No
                  Yes :  Yes
            Ein-Karem :  Yes
            Mahane-Yehuda :  Yes
            Talpiot :  Price
                  Cheap :  No
```

Manual visualization of the decision tree that we created:



Contributions:

1: Sanjana's contribution

- Suggested the code structure to use (like how to design a node/attribute)
- Wrote the code for data parsing and entropy calculation
- Wrote the report for the assignment
- Studied in more detail about ID3 algorithm

2: Amit's contribution

- Helped with familiarity in python language
- Wrote the recursive function for training/building decision tree
- Choose best feature and get majority label
- Implemented code for displaying the decision tree

Challenges:

Regarding deciding how to design TreeNode class, we initially thought of using name and children, use name to make sure we don’t select the same feature again. But this proved to be getting more complex and therefore later decided to add index which made tracking easier.

Prediction:

Prediction for {"Moderate", "Cheap", "Loud", "City-Center", "No", "No"}: **Enjoy = Yes**