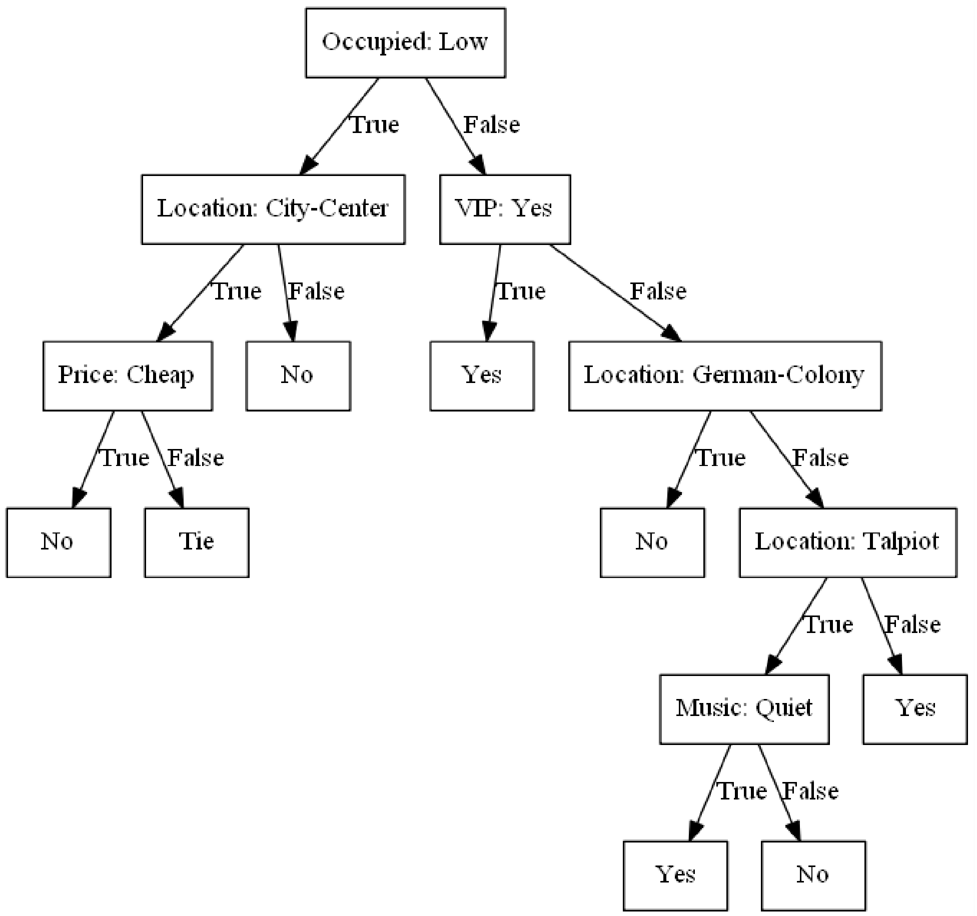
INF552 MACHINE LEANING FOR DATA INFOMATICS

ASSIGNMENT 1: DECISION TREE

PART 1: IMPLEMENTATION

In this assignment, we produced 2 different types of decision tree. Not only the programming language and output format differs, but also the input data format isn't alike. However, the **predictions** for the test data, "*occupied = Moderate; price = Cheap; music = Loud; location = City-Center; VIP = No; favorite beer = No*", are the same: **"Enjoy" = "Yes".**

TREE #1 OUTPUT



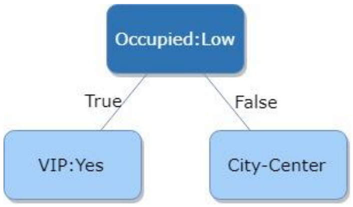
TREE #1: FURTHER DESCRIPTION

DATA STRUCTURES USED IN THE PROGRAM:

To store the tree, the most important data structure used in this program is Dictionary. It takes me a lot of time to figure out how should it looks like in order to properly design the program. In short, a Dictionary contains several Keys attached with Values. In this case, a Dictionary represents a node in the tree. The keys of Dictionary are “value”, “True” and “False”, and the corresponding values are features. For example, assume a Dictionary called Mother exists, and it looks like this:

Mother = {“value ”: “Occupied: Low”, “True”: “VIP: Yes”, “False”: “Location: City-Center”}

This Dictionary can be further transform into a Tree graph:



CODE-LEVEL OPTIMIZATIONS PERFORMED:

1. Arithmetic Mean VS Geometric Mean

Firstly, I calculate the information gain of each feature by arithmetic mean. Soon I found out that the tree tends to split data in a insufficient way.

To be more specific, comparing 2 features X and Y. Feature X can split a list of data from [0, 0, 0, 0, 0, 1] into [0] and [0, 0, 0, 0, 1]; feature Y splits the data into [0, 0, 0] and [0, 0, 1]. Which feature is better? Feature Y seems to be more reasonable, but, in fact, if we use arithmetic mean to calculate information gain, feature X would be chosen. See the table below:

Feature	Splitted Data	Arithmetic	Geometric
X	[0], [0, 0, 0, 0, 1]	0.3609	0.6016
Y	[0, 0, 0], [0, 0, 1]	0.4591	0.4591

In the end, I changed my entropy method in my Python code.

2. Use Dictionary/ pandas.Dataframe Instead of List/ List of List to Store Data

Inspired from my previous Python experiences, using pandas.Dataframe instead of a list of list is more sufficient when processing data. I think that’s because the package has optimized their function to retrieve and update data in dataframes. Also, using a dictionary instead of list of list, isn’t only more straightforward for reading, but also save time from indexing data.

CHALLENGE:

The biggest challenge, for me, is to develope a recursive algorithm. I spent most of the time thinking about how to let my tree grow.

TREE #2 OUTPUT

```
Occupied
    High
        Location
        Talpiot
            No
        City-Center
            Yes
        German-Colony
            No
        Ein-Karem
            Tie
        Mahane-Yehuda
            Yes
    Moderate
        Location
        Talpiot
            Price
            Expensive
                Tie
            Normal
                Yes
            Cheap
                No
        City-Center
            Yes
        German-Colony
            VIP
            No
                No
            Yes
                Yes
        Ein-Karem
            Yes
        Mahane-Yehuda
            Yes
    Low
        Price
        Expensive
            No
        Normal
            Location
            Talpiot
                Tie
            City-Center
                Music
                Loud
                    Tie
                Quiet
                    VIP
                    No
                        Favorite Beer
                        No
                            Tie
                        Yes
                            Tie
                    Yes
                        Tie
            German-Colony
                Tie
            Ein-Karem
                No
            Mahane-Yehuda
                Tie
        Cheap
            No
```

TREE #2: FURTHER DESCRIPTION

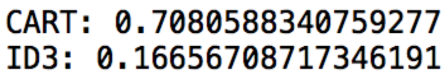
DATA OUTPUT DESCRIPTION:

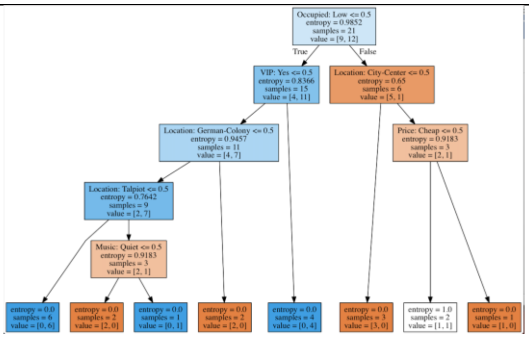
I print my Decision Tree in a horizontal way: The first word(Occupied) represents the most important attribute used for classification. Then the rows below this word represent its possible values (High, Moderate, Low). The third word(Location), which is indented, represents the secondly important attribute for the classified data, and so on. The leaf nodes are the classification results, including “Yes”, “No”, “Tie”.

HOW TO IMPROVE MY CODE:

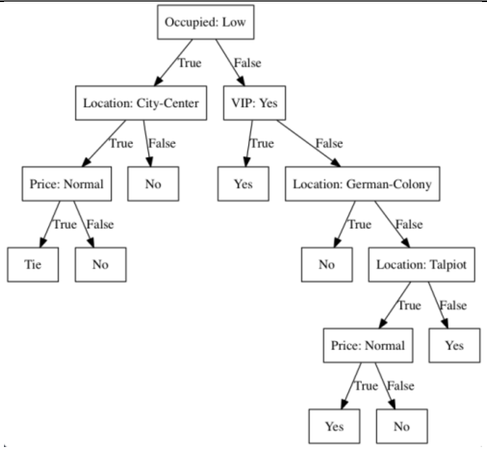
My code is based on the idea of ID3. It can only process distributed value instead of continued value. Therefore, I think I can manage the continued data to make it distributed, and my code cannot process missing data. For the further implementation, I could set default value for the missing data based on the major value in this attribute. Moreover, if there is any conflicting data such as the input data No.18 and No.21, my process just classified all the attributes to the end. It’s meaningless and will result in over fitting. As a consequence, taking advantage of pruning would be a better way to find the suitable condition.

PART 2: COMPARISON

<table>
<tr><td></td><td>CART</td><td>ID3</td></tr>
<tr><td>Attribute type</td><td>Handles both categorical and numerical value</td><td>Handles only categorical value</td></tr>
<tr><td>Operation Time</td><td colspan="2">


CART: 0.7080588340759277
ID3: 0.16656708717346191
</td></tr>
<tr><td>Decision Tree output</td><td>


Occupied: Low <= 0.5 entropy = 0.9852 samples = 21 value = [9, 12]
True
False
VIP: Yes <= 0.5 entropy = 0.8366 samples = 15 value = [4, 11]
Location: City-Center <= 0.5 entropy = 0.65 samples = 6 value = [5, 1]
Location: German-Colony <= 0.5 entropy = 0.9457 samples = 11 value = [4, 7]
Price: Cheap <= 0.5 entropy = 0.9183 samples = 3 value = [2, 1]
Location: Talpiot <= 0.5 entropy = 0.7642 samples = 9 value = [2, 7]
Music: Quiet <= 0.5 entropy = 0.9183 samples = 3 value = [2, 1]
entropy = 0.0 samples = 6 value = [0, 6]
entropy = 0.0 samples = 2 value = [2, 0]
entropy = 0.0 samples = 1 value = [0, 1]
entropy = 0.0 samples = 2 value = [2, 0]
entropy = 0.0 samples = 4 value = [0, 4]
entropy = 0.0 samples = 3 value = [3, 0]
entropy = 1.0 samples = 2 value = [1, 1]
entropy = 0.0 samples = 1 value = [1, 0]

</td><td>


</td></tr>
<tr><td>Interpretation</td><td>Take first attribute(Occupied) as an example, if Occupied: Low <= 0.5 is true, then the result would be VIP: Yes <= 0.5, which means that Occupied is either Moderate or High and would get the result VIP: Yes</td><td>Take the same example as CART, if Occupied: Low is False, then the result would be VIP: Yes.</td></tr>
<tr><td>Conclusion</td><td colspan="2">
1. CART is much faster than ID3

2. The results of ID3 is much easier to interpret and understand
</td></tr>
</table>

PART 3: APPLICATION

In the retail industry, enterprises could use information technology to explore the data from a large number of transaction records to find out the customers' consumption characteristics, needs and other useful information. Besides, they could meet their customer needs, improve customer loyalty and further increase the profit through marketing strategies. Enterprises would achieve their goal by embracing the knowledge of Decision Tree. Based on consumer purchasing behavior, decision tree could divide each costumer into different types of group representing various purchasing pattern and then establish a customer market segment and identify the target customer base as a business support marketing strategy.

GROUP MEMBER

Jung-Kang, Su	2389753352	Research and edit
Yuhsi Chou	6048573191	Create Tree1
Zhang Mujie	2621330761	Create Tree2

REFERENCE

1 https://www.quora.com/What-are-the-differences-between-ID3-C4-5-and-CART