CS 6220 Data Mining — Assignment 8

Decision Tree

This assignment will require you to implement and interpret the concepts of decision tree. Keep in mind that the main objective of this assignment is to highlight the insights that we can derive from applying these techniques—the coding aspect is secondary. Accordingly, you are welcome to consult any online documentation and/or code that has been posted to the course website, so long as all references and sources are properly cited. You are also encouraged to use code libraries, so long as you acknowledge any source code that was not written by you by mentioning the original author(s) directly in your source code (comment or header).

Objectives:

1. Implement a decision tree using scikit learn.
2. Display the final decision tree.
3. Interpret results generated by the training process.

Submission:

Submit your ipynb file on the Assignment submission portal.

Grading Criteria:

Follow the instructions in the pdf, and complete each task. You will be graded on the application of the modules' topics, the completeness of your answers to the questions in the assignment notebook, and the clarity of your writing and code.

Assignment Description

The Data

The given dataset contains information of Boston housing-prices and you are already familiar with it. The dataset has 13 numeric/categorical predictive attributes. Median Value (attribute 14) is usually the target.
The dataset can be loaded using the sklearn datasets module.

What to Do

First, load the dataset from sklearn dataset using the following code snippet. You can get the features name using *data.feature_ names* and target names using *data.target_ names*

```
from sklearn.datasets import load_boston
data = load_boston()
X = data.data
y = data.target
```

Note here that this dataset as it is, is good for regression. To convert it into a classification dataset,

- Split the range of target values into three equal parts - low, mid, and high.
- Reassign the target values into into three categorical values 0, 1, and 2, representing low, mid and high range of values, respectively.

Then,

1. Split the dataset into 70% training set and 30% test set.
2. Using scikit-learn's DecisionTreeClassifier, train a supervised learning model that can be used to generate predictions for your data. A reference to how you can do that can be found in the users manual at https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassif
3. Report the tree depth, number of leaves, feature importance, train score, and test score of the tree. Let the tree depth be Td.
4. Show the visual output of the decision tree.
5. Next, Generate (Td-1) decision trees on the same training set using fixed tree depths $\{1, 2, ...(Td - 1)\}$. The tree depth can be set using *max=d*, where *d* is the depth of the tree.
6. For each of the (Td-1) trees report, tree depth, number of leaves, feature importance, train score, and test score of the tree.

7. Show the visual output of the decision tree with highest test score from the (Td-1) trees.

   To visualize the decision tree, use Graphviz library. You can find details in this link. Show the feature names and class names in the visualization.

What to Provide

Your output should contain the following:

- Report of tree depth, number of leaves, feature importance, train score, and test score of the decision tree.
- Report of tree depth, feature importance, train score, and test score of the decision tree for each value of tree depth used in step 5.
- Visual output of the decision trees. There should be two separate decision trees as mentioned previously.