CMPT459: Data Mining, Fall 2024

Instructor: Martin Ester
TA: Arash Khoeini
Assignment 1

In this assignment, you will implement a decision tree classifier using the provided files, without relying on any GenAI tools or third-party libraries, except for NumPy and pandas. You will then apply your classifier to categorize individuals as earning either less than 50K or more than 50K based on their features. The provided dataset includes 14 features: 6 continuous and 8 categorical. You can find more information about this dataset here.

You are provided with three files:

- **main.py:** This is the main file used to run your decision tree implementation. It takes and parses the necessary arguments from the user, reads the input data, and then trains and evaluates a decision tree. The only logic you need to implement here is handling missing values in both the training and testing data.
- **node.py:** This file contains the *Node* class, which serves as the data structure for building the decision tree. By default, each *Node* object is a leaf node (node.is_leaf = True) unless you set its children, at which point it becomes a decision node. Using the node.set_children method will automatically set node.is_leaf = False. A decision node is defined as a node object that: (1) has a name corresponding to the feature used for splitting, (2) has the following properties set: node.children, node.is_numerical, and node.threshold (for numerical features only), (3) has node.is_leaf = False. The *Node* class is fully implemented, so no modifications are needed. However, I highly recommend carefully reviewing its implementation to understand its behavior.
- **decision_tree:** This file contains the *DecisionTree* class, which you need to complete in order to finish this assignment. Please read the implementation documentation carefully before writing any code. The main methods you need to implement are fit, predict, split_node, gini, and entropy. You are also encouraged to add new helper methods if needed.

You are provided with the file main.py which you will use to run your implementation. It accepts the following arguments:

- --train-data: Train data path
- --test-data: Test data path
- --criterion: The function to measure the quality of a split. Supported criteria are “gini” for the Gini index and “entropy” for the information gain.
- --maxdepth: The maximum depth of the tree.

- --min-sample-split: The minimum number of samples required to split an internal node.

[IMPORTANT] You should submit a *[student-id].zip* file. The zip file should include the following files:

- *report.pdf*
- *decision_tree.py*
- *main.py*
- *node.py*

Running *python3.10* command on *main.py* file with the mentioned arguments must print out your train and the test accuracy of task #1.

Deadline: make your submission no later than **23:59 pm on October 4th**. You will lose %10 of the marks for submissions after this deadline, as long as it's not more than 24 hours late. You will lose all the marks for submissions after that time.

Libraries: No third-party library is allowed except NumPy and pandas.
You MUST provide YOUR OWN implementation. It MUST be implemented from scratch i.e. not using scikit-learn libraries. You will be marked on the correctness of your implementation. You are also encouraged to do any data preprocessing that you think might help the classification accuracy.

Please refrain from using any GenAI tools while completing this assignment. While it might be tempting, doing so would completely defeat the purpose of the task. The goal is for you to gain hands-on experience implementing one of the most widely used data mining methods. It's a rewarding process, and once you've built an algorithm yourself, it's something you'll never forget!

a) [10 marks] Present a pseudo-code for a simple Decision Tree:
   i) The Gini index is used as a split criterion. [2 marks]
   ii) The trees are grown deep, i.e. they are grown until all training examples corresponding to a leaf node belong to the same class. [3 marks]
   iii) Works both on categorical and numeric data. [5 marks]

b) [90 marks] Implement the decision tree algorithm using Python programming language. Your implementation must include the following functions (method signatures might be different based on the specific needs of your implementation):

   - *fit* [5 marks]
   - *split_node* [5 marks]

- *predict* [5 marks]
- *gini* [5 marks]
- *entropy* [5 marks]

Use your implementation to complete the following tasks:

1. Train a decision tree model with maxdepth=10, min_sample_split=20. Use the Gini index as the criterion. Report the accuracy of the training and testing data. [10 marks]
2. Do the same experiment as 1, but this time with Information Gain as the criterion. How does this change the accuracy of training and test datasets? Discuss it in your report. [10 marks]
3. Repeat Task 2 using at least 10 different values for the *min_sample_split* parameter. For each value, plot the accuracy of the trained decision tree. Include a discussion of the plot in your report. [10 marks]
4. In your report explain why the accuracy of the training data is not equal to 100%. [5 marks]
    a. Is there any way to train a tree that yields 100% accuracy on the training dataset? What parameters should you change to get a tree with perfect training accuracy? How would that affect the accuracy of the test data? Explain if such a classification model has a high variance or bias. [10 marks]
5. Train 16 decision trees with min_sample_split=10 and maxdepth=[1,2,3,...,16]. Use the Gini index as the criterion. Plot the accuracy of the 16 models on the test dataset in your report. Discuss your results. [20 marks]