Michael Evans Decision Tree Documentation

User manual

This program is a guessing game similar to “20 questions.” You are asked to think of an animal and the program attempts to guess it using a binary tree built out of previous responses. It learns as it fails.

To run this program, click on the “run” button in your IDE of choice, and follow the prompts given to you. A file with a decision tree already made will be provided, but constructing one is as simple as providing a first line of “A:(<insert animal of choice here>)”. Following the program’s prompts will allow you to build up the tree and eventually make it fairly accurate.

Algorithm

Main guessing function (DecisionNode tree)

If the current node is determined to be a question

The question is asked

The user’s response is stored

If the user’s response was “yes” and the right branch of the tree exists

Recursively call the function on the right branch

If the user’s response wasn’t “yes” and the left branch of the tree doesn’t exist

Recursively call the function on the left branch

If the current node isn’t a question

The program asks the user if the answer it obtained is correct

Their response is stored

If the guess was correct

Nothing is returned

If the guess was incorrect

The program asks what the correct answer was

The response is stored

The program asks for a “yes/no” question to get better results

The question is stored

The question is made into the new root

The incorrect answer is made into the left node

The correct answer is made into the right node

The constructed tree is returned.

Reading a tree from a file(Scanner in, DecisionNode tree)

If there isn’t anything left in the file

Return null

If there is stuff left in the file

Store the next line of the file

If the stored line is a question

Set the tree to the new question node

Set the left branch to a recursive call on the new tree

Set the right branch to a recursive call on the new tree

If the stored line is an answer

Set the tree to it, because it must be a leaf

Return the constructed tree

Time Complexity

These algorithms will run in recursive time. Their time complexity cannot be easily explained.

Test Cases

Test Case 1: A normal file with a populated tree.

Test Case 2: A normal file with only one line that contains an answer.

Test Case 3: An empty file (which will create an exception that I’m not handling, because it is stated in the precondition that the file needs to have at least on line containing an answer).