CSE 223 Programming Assignment 3: Decision Trees!

May 8, 2020

(UPDATED 8 May 2020 as described in class on 7 May 2020: you can assume the database file is perfect, i.e. not corrupt. When opening the database file, if it does not exist, just report an error (your own message - don't throw an exception) and exit.

1 Introduction

For this assignment, you'll be writing code to implement, use and extend a decision tree. This is similar in spirit to the "20 Questions" game (though we're not restricting this to 20 questions).

This assignment will be discussed in detail in class, but here are some additional details:

• you'll need to implement a binary tree, where each node contains 2 pieces of data: a string of text, and a flag indicating whether this is a question or an answer;

• To play the game, begin at the root (it will be a question), and ask the question;

• based on the user's response, travel to the left (YES) or right (NO) sub-tree, and repeat, until you reach an answer;

• if the answer is correct, no action is necessary;

• if the answer is wrong, ask for the answer and a new question to discriminate the correct answer from the incorrect one you guessed; then make a new node from that questions and those two answers, and insert it into the tree in place of the incorrect answer.

Additionally, the initial decision tree should be read from a file (20Q.txt by default), and, after being modified, re-written to the same file. The name of this database should be modifiable by specifying a single command-line argument. Remember to traverse in PRE-ORDER (NLR) to simplify the reading/writing of this file.

Here's a sample initial database file:

```
Q:
Is it an animal
A:
dog
A:
rock
```

(this file should have 6 lines total, with no empty lines).

As always, I strongly recommend approaching this in pieces: make a binary tree class, then extend it to support this decision tree; then hard-wire some nodes and get it to traverse/play the game. Work on the File I/O separately; work on extending the tree separately; and so on.

You may use built-in primitives from Java such as Scanner, PrintWriter, etc. but create your own tree class, your own file parser, your own traversal code, and so on.

Do not use Eclipse, IntelliJ, or any other IDEs: work from the command line on the Linux server (or your own Linux environment). This will ensure you know the syntax for defining classes, writing a main method, and so on.

2 Implementation

Make a main method in a class named PA3 (inside a file named PA3.java). This should be executable by saying

```
java PA3
```

which will read from a database named "20Q.txt" (report an error and exit gracefully if it does not exist). Additionally, you can work with a given database by saying

```
java PA3 filename
```

which will read from the database named filename (and update it as needed) (and again, if the file does not exist, report an error and exit gracefully).

This should run your code to play 20 questions, asking the user to think of an item, and then asking questions to guess what the item is. Again, **details will be given in class**. This is only a high-level overview of the assignment.

If the database file exists, you may assume it is a perfect file, with the expected structure etc. You are not responsible for handling corrupt files.

3 Testing

I have put a pair of databases on the linux server:

```
/tmp/20Q.txt
```

and

```
/tmp/20QBig.txt
```

Feel free to copy these into your own directory and work with them there (the copies in /tmp are read-only). 20Q.txt is the sample initial database described above. 20QBig.txt is a larger database holding around 20,000 items (originally given to me by a UW student).

4 Grading

Your assignment must be downloadable from GITLab by me in order to be graded. I will download using the command:

```
git clone git@gitlab.com:yourGITid/CSE223PA3.git
```

You can mimic this yourself to confirm that your repository is located and named correctly. There's no sure way to check that you have added me as a reporter: just make sure you do!

Points are awarded as follows:

- Project can be downloaded and compiled: 20 points
- Program asks questions and makes a guess: 10 points
- Program acts according to the database file (i.e. plays the game as it's supposed to): 30 points
- User can specify a different database file: 10 points
- Program updates database file if it guesses wrong: 15 points
- style: 15 points if you have **all of the following**:
  - consistent indenting, placement of braces, etc.
  - all code commented (**every line** unless it's extremely obvious why that line is there)
  - each method described by an introductory block of comments.
  - entire class described by a block of comments in the beginning (in your own words).

5 Submission

Create a repository named CSE223PA3 on GITLab. The repository should contain all of your .java files **in the top level of the repository**. You can include any other files, subdirectories, etc. as you like (I will ignore them). Add me (nickmacias) as a reporter on your project.