Artificial Intelligence: Assignment 3 - Decision Trees

Submit by: 23/04/2019

April 6th, 2019

Decision Trees

Decision Trees are a very useful way of representing knowledge in a compact way. It is also useful, because the representation can be understood by non-specialists. Decision trees can be built manually, but they can also be "learned" given a set of observations. Decision tree models fit into the category of **supervised machine learning** algorithms, where one of the variables is well known and classifies all observations. This variable is the one that we use to "learn" the model.

Using as a reference the material available in Sections 18.1 to 18.3 of our textbook, Artificial Intelligence: a Modern Approach, by Peter Norvig and Stuart Russell (3ed), implement an algorithm for induction of decision trees (similar to the ID3, shown in the book, Figure 18.5). Use as node selection function the **entropy** defined in page 704 (Section 18.3.4 explains how to choose "good" attributes for each level of the decision tree). It is **highly recommended** to read chapter 18 in order to understand what is a decision tree and its objectives.

The input to your program will be a set of examples in CSV (Comma Separated Value) format. This set will have several attributes (columns of your CSV table), being the last one the variable of interest for classification (Sections 18.1 to 18.3 of the textbook show an example of such examples in a table format). The output of your program must be in the format below, for a dataset that has 5 attributes (columns attribute1, attribute2, attribute3, attribute4, and the class variable), whose class variable has 4 different values (class1, class2, class3 and class4).

```
<attribute1>
    value1:
        <attribute2>
            value1: class1 (counter1)
            value2: class2 (counter2)
    value2: class3 (counter3)
    value3:
        <attribute3>
            value1:
               <attribute4>
                   value1: class4 (counter4)
                   value2: class2 (counter5)
            value2: class3 (counter6)
```

In general, `attribute` is the root of each subtree, `value#` is one of the values of the attribute (one of the branches of your tree), `class#` is the class value assigned to that branch in the tree (and corresponds to a leaf) and `counter#` is a counter of the number of examples corresponding to that tree branch. For this particular example, there are 4 attributes, where attribute1 has 3 distinct values, attribute2 has 2 distinct values, attribute3 has 2 distinct values and attribute4

has 2 distinct values. This dataset has 4 values for the class variable (class1, class2, class3 and class4). Counter# corresponds to the frequency of values of an attribute according to the class variable.

Examples of datasets for testing can be found in the following files:

- `http://www.dcc.fc.up.pt/~ines/aulas/1819/IA/t3/restaurant.csv`
- `http://www.dcc.fc.up.pt/~ines/aulas/1819/IA/t3/weather.csv`
- `http://www.dcc.fc.up.pt/~ines/aulas/1819/IA/t3/iris.csv`

Below is a description of each one of these files:

- `restaurant`: (example from textbook, page 700) contains information about customers and restaurants (type of food, waiting time, price etc), and the class attribute (last column) says if the customer will wait or not to eat in that restaurant. The task is to generate a decision tree (as explained in theoretical class and following the ID3 algorithm available in the textbook). This decision tree must be used later to classify (answer if the customer will wait or not) new cases.
- `weather`: contains information about climate conditions to play tennis. The task is to learn a decision tree that can decide what are the best conditions to play tennis.
- `iris`: contains **numerical** information about plants of three classes: iris setosa, iris virginica and iris versicolor. The attributes are petal length and width and sepal length and width. The task is to learn a decision tree that can tell to which class a plant belongs to, given its sepal and petal lengths and widths.

Your program needs to be able to read any dataset and learn the appropriate decision tree. You should **not** write one program for each one of the datasets. This means that you will need to read the CSV table from a file and store it in memory for **any** input table.

In addition to learn a decision tree, your program must also be prepared to accept as input a file with test examples, i.e., after generating your tree, you must be able to apply your tree to new examples and be able to classify them appropriately. For example, suppose you generated a tree to the restaurant problem. Now, you can enter new examples (without any class/label) and be able to give them a proper class.

A challenge dataset can be found in `http://archive.ics.uci.edu/ml/datasets/connect-4`, where each line corresponds to a board configuration of the connect four game. Induce a decision tree for this dataset and replace the utility function used in your ASSIGNMENT 2 with the prediction of this tree to decide where to play next. (a copy of the file can be found here: `http://www.dcc.fc.up.pt/~ines/aulas/1819/IA/t3/connect4.csv`. You need to add a header to this file and probably an identifier to each instance, depending on your implementation). What is the impact of using the decision tree to choose the next move instead of using the utility function?

Important note: The dataset iris contains numerical values. You need to implement a way of discretizing these values in order to minimize the size of your decision tree.

What to deliver?

1. Written report

   Organization:

   Introduction
   - What is a decision tree?
   - What is it for?

   Algorithms for decision tree induction
   - search books and the internet for the most popular algorithms
   - explain differences among them
   - discuss about the different metrics used to select attributes to place in the tree during its construction
   - Explain how the ID3 algorithm works and discuss about its limitations

   Implementation
   - language
   - data structures used and justification for using them
   - organization of your code

   Results
   - report the generated trees for each dataset
   - show counters in each branch of the tree

   Final Comments and Conclusions

2. source code and how to compile, run etc, as well as the runtime environment used to run and test your program. The program must run from the command line.

   As usual, submission will be done through Moodle UP.