CMSC 422 – Assignment 3: Learning Decision Trees – Spring 2019

Induction of decision trees has been a long-standing topic of interest in machine learning. The purpose of this assignment is to give you experience with basic tree induction methods.

1. Basic Decision Tree Induction

A decision tree is desired that will classify an observed sea animal as a dolphin or not, based on the animal's length, whether it has gills, whether it has a beak, and how many teeth it has. Suppose that a number of sea animals are seen, resulting in the following complete set of training data where all possible values of each attribute are shown:

LENGTH	GILLS	BEAK	TEETH	DOLPHIN
three	no	yes	many	yes
five	yes	yes	many	no
four	no	yes	many	yes
four	yes	yes	many	no
three	no	yes	few	yes
five	yes	no	many	no
five	no	yes	many	yes
four	yes	no	many	no
five	no	yes	few	yes
four	no	yes	few	no

a. Calculate the entropy of the complete set of training data above.

b. Considered as a possible root node of a decision tree being built by ID3, what is the average entropy $\overline{H}$ associated with each of the input attributes?

c. Considering your answers in (b), what is the information gain associated with each of these four input attributes?

d. Which of the four possible input attributes would ID3 select as the root node for the decision tree it was building?

e. Show the complete decision tree that ID3 would construct for this data. You do not need to show further detailed numerical calculations, but you must make it clear and explicit why you select each attribute that appears below the root node.

To simplify calculations, use the following approximate log values in your calculations. Some needed log values are not in this table. Use a calculator to compute those.

x	0	1/10	1/6	¼	1/3	2/5	½	3/5	2/3	¾	5/6	9/10	1
$\log_2(x)$	-∞	-3.3	-2.5	-2	-1.6	-1.3	-1	-0.7	-0.6	-0.4	-0.25	-0.1	0

2. Tree Induction Using an ID3-Like System

Marsland's ML textbook web site (http://stephenmonika.net) provides Python code (specifically *dtree.py*) that can be used to induce decision trees. You will be given a modified version of Marsland's *dtree.py* in the file *DT.py* that contains a number of changes (discussed in class) to use with this problem. Modify the code (specifically function *printTree()*) so that it saves its output directly in a file named *ResultsID3.txt* that is to be turned in, and so that it not only outputs the induced trees (indented format) as it does now, but also the total number of nodes in the tree (including leaves) and the number of leaves in the tree. Your results file should also contain the number of training examples used and the number of these training examples that were subsequently classified correctly by the induced tree. To keep life simple, i.e., to minimize software that needs to be written, *in this problem only*, the complete set of provided data is to be used for both training and testing (no need to divide it into training, validation and testing portions). Obviously, this is not acceptable in general.

You will be given a data set *party.txt* where each of the 435 examples specifies how a congressional member voted on the key bills before congress in 1984. Each example specifies how that representative voted on 16 bills: n = no, y = yes, and a = abstain, and the political party of that representative (democrat versus republican), the latter being the output class. The names of these 16 input features (handicapped-infants,water-project-cost-sharing,adoption-of-the-budget, …) are listed at the start of *party.txt*, as is the class (party). Write a script file *party.py* that uses this data set to generate a decision tree that predicts a representative's political party, democrat versus republican, based on their voting record on these bills. This tree and the other information described in the paragraph above should be saved in the *ResultsID3.txt* file.

a. According to the results here, which of the 16 input features, in isolation, provides the most information in terms of determining how a congressional representative will vote? How many nodes total are in the induced tree, and how many of these are leaf nodes?

b. How would this induced decision tree classify a representative with the following voting record?

n	handicapped-infants
y	water-project-cost-sharing
n	adoption-of-the-budget-resolution
n	physician-fee-freeze
y	el-salvador-aid
y	religious-groups-in-schools
y	anti-satellite-test-ban
n	aid-to-nicaraguan-contras
y	mx-missile
n	immigration
y	synfuels-corporation-cutback
n	education-spending
a	superfund-right-to-sue
y	crime
n	duty-free-exports
a	export-administration-act-south-africa

c. Decision trees like that induced here by ID3 from the congressional data often do not provide a conceptually simple and readily understandable representation of the data on which it they are trained. What post-processing step is often used to generate a more understandable version of decision trees like this (and that also typically generalizes better to new, previously unseen data)?

3. Tree Induction Using a More Contemporary Approach

Weka (http://www.cs.waikato.ac.nz/ml/weka/), which we encountered in the last homework assignment, includes a module for inducing decision trees using more contemporary methods than ID3. This module, which is modeled after C4.5, is called J48.

Use Weka's J48 module to induce a decision tree for the data in *party.arff* that you are given. This data set is identical to the data that was used in Problem 2 except that the file suffix (".arff") has been changed to conform to Weka's expectations, and required header information has been added to the beginning of the file to define the name of the data set and the input features/attributes and their values. Use the existing default settings and parameter values of J48 (e.g., 10-fold cross validation). Copy and paste the contents of the output window in the results file *ResultsJ48.txt*, thereby saving not only the induced tree but also the performance information (error rate, confusion matrix, etc.).

a. Make a picture of the congressional party decision tree that was output by Weka for this data. Either a hand-drawn sketch or computer-generated drawing of the labeled nodes and branches is fine (i.e., this must not be the indented-text format that *printTree()* uses). What is the total number of nodes in this tree, and how many of them are leaf nodes?

b. How does the decision tree for the congressional party data produced by Weka's J48 differ from that produced in Problem 2 above by ID3? Explain why these differences occur.

c. How does Weka's decision tree classify the example representative given in Problem 2b?

d. Looking at the results given in Weka's output, what is the reported error rate for the generated decision tree, and what is the most common specific error made by the decision tree on the data?

e. Learning decision trees can be viewed as one approach to learning a set of rules or productions. Write the complete set of conjunctive if-then rules that can correctly be derived from the decision tree that Weka's J48 generated (i.e., that is equivalent to that tree assuming no rule-set simplification is done following generation of the rules).

f. Explain how the original C4.5 (not J48) software pruned the rules that it generated. No need to do any rule pruning here, just explain how C4.5 did it.

g. If additional congressional representatives existed that were not included in the training data used here, which of the two trees generated, the one using ID3 in Problem 2 or the one using J48 in this problem, would be most likely to generalize best to this new data? Why?

What should I turn in?

Be sure to retain a copy of your submitted homework if you wish to refer to it in studying for the midterm exam. It will not be possible to return the graded assignments until after the exam.

The hardcopy and electronic submissions are due at different times. The hard copy portion is due at the start of class Tues., March 12, the electronic submission is due earlier at 11:30 pm Mon. March 11.

Hardcopy: Your answers to the questions in problems 1a-e, 2a-c, and 3a-g.

Electronic submission: For problems 2 and 3, turn in a single zip file that includes the result files *ResultsID3.txt* and *ResultsJ48.txt*, your modified version of *dtree.py*, and the script file *party.py* that you wrote to do Problem 2. Be sure that you include any files you wrote that are needed to make your code run. As was done with the previous assignments, use the Computer Science Department project submission server that is located at https://submit.cs.umd.edu to submit this zip file.