CS322 Fall 1997
Assignment 11

Due: 1:30pm, Friday 28 November 1997.

Question 1

In the file `~cs322/cilog/dtlearn1.pl` is an axiomatization of a decision-tree learning algorithm. This works with the data in the file `~cs322/cilog/dtlearn_t1.pl`, and produces the tree of Figure 11.3 of the textbook.

This uses a complicated procedure to determine what is the best attribute to split on. In this question, you will look at what happens when this procedure is changed.

1. Suppose you change the procedure `select_split` so that it always selects the first element of the list of attributes. What tree is found when the attributes are in the order [*author*, *thread*, *length*, *where_read*]? Does this tree represent a different function than that found with the more complicated selection mechanism? Explain.

2. What tree is found when the attributes are in the order [*where_read*, *thread*, *length*, *author*]? Does this tree represent a different function than that found with the more complicated selection mechanism or the one given for the preceding part? Explain.

3. Is there a tree that correctly classifies the training examples, but represents a different function than those found by the preceding algorithms? If so, give it. If not, explain why.

Solution to Question 1, part 1

Suppose you change the procedure `select_split` so that it always selects the first element of the list of attributes. What tree is found when the attributes are in the order [*author*, *thread*, *length*, *where_read*]? Does this tree represent a different function than that found with the more complicated selection mechanism? Explain.

This can be found by changing the definition of `select_split` to:

```
select_split(_, _, [Cond|Rem_Atts], Cond, Rem_Atts).
```

The tree can be found by asking:

```
cilog: ask dtlearn(user_action,
            [e1,e2,e3,e4,e5,e6,e7,e8,e9,e10,e11,e12,e13,e14,e15,e16,e17,e18],
            [author,thread,length,where_read], DT).
Answer: dtlearn(user_action,
```

```
            [e1, e2, e3, e4, e5, e6, e7, e8, e9, e10, e11, e12, e13, e14,
            e15, e16, e17, e18],
            [author, thread, length, where_read],
            if(author = known,
                   if(thread = new, if(length = long, skips, reads),
                                    if(length = long, skips, reads)),
                   if(thread = new, reads, skips))).
```

This tree represents a different function than that in Figure 11.3 of the book as it classifies an example with *author* = *unknown*, *thread* = *new*, *length* = *long* as *reads* whereas the tree of Figure 11.3 classifies the same example as *skips*.

Solution to Question 1, part 2

What tree is found when the attributes are in the order [*where_read*, *thread*, *length*, *author*]? Does this tree represent a different function than that found with the more complicated selection mechanism or the one given for the preceding part? Explain.

Here is the trace (reformatted for easy reading):

```
cilog: ask dtlearn(user_action,
           [e1,e2,e3,e4,e5,e6,e7,e8,e9,e10,e11,e12,e13,e14,e15,e16,e17,e18],
           [ where_read, thread, length, author],DT).
Answer: dtlearn(user_action,
               [e1, e2, e3, e4, e5, e6, e7, e8, e9, e10, e11, e12,
                    e13, e14, e15, e16, e17, e18],
               [where_read, thread, length, author],
               if(where_read = home,
                    if(thread = new, if(length = long, skips, reads),
                                     if(length = long, skips,
                                         if(author = unknown, skips, reads))),
                    if(thread = new, if(length = short, reads, skips),
                                     if(length = long, skips,
                                        if(author = unknown, skips, reads))))).
```

This represents the same function as the tree of Figure 11.3. It classifies every example in the same way.

Solution to Question 1, part 1

Is there a tree that correctly classifies the training examples, but represents a different function than those found by the preceding algorithms? If so, give it. If not, explain why.

There are 4 missing cases for which there are no examples. Thus there are $2^4 = 16$ different Boolean functions that classify the examples correctly, of which we have found two! There are 14 other such trees.

Here is one such tree:

```
if(author = known,
       if(length = long, skips, reads),
       if(where_read = work,
          if(thread = follow_up, skips, reads),
          skips))).
```

which predicts an example with $author = unknown, thread = new, length = short, where\_read = home$ will have classification *skips*, where the other trees classifies this as *reads*.

Question 2

Suppose that in the output of a neural network, we assign any value over 0.5 to be true and any less than 0.5 to be false (i.e., any positive value before the activation function is true, and a negative value is false).

Given the final parameter settings, of Figure 11.7 of the text, give a logical formula (or a decision tree or a set of rules) that represents the Boolean function that is the value for the hidden units and the output units. This formula or set of rules shouldn't refer to any real numbers.

Hint: one brute-force method is to go through the 16 combinations of values for the inputs to each hidden unit and determine the truth value of the output. A better method is to try to understand the functions themselves. Your meta-interpreter for parameterized logic programs may help here.

Solution to Question 2

- h_1 represents the Boolean function "*short* and (*known* or *new*)" (which is the same function as the decision tree of Figure 11.3).
- h_2 represents the Boolean function "not *known* and not *new* and not *short*".
- *reads* represents the Boolean function "h_1". That is, *reads* is true if h_1 is true and is false otherwise.

Thus the neural network makes the same predictions on both seen and unseen examples as the decision tree!