Homework III
Datamining
Computer Science
Indiana Universty
Bloomington, IN

M.M. Dalkilic

March 3, 2015

Introduction

This homework is short, but I hope fun! The problems will help elucidate the material. Please remember to label the homework correctly and include your statement of ownership. There are plenty of opportunities for you to do more and, consequently, learn more.

Problems

1. Assume you have a tab delimited file `table.txt`. The first row are attribute labels, and each subsequent row are values. You will write a program that builds a classification tree $\mathcal{T}$ from `table`, then use $\mathcal{T}$ to predict classes given an input. There will be a slight twist with this homework as explained after an example.

   Assume `table` is this file (a tuple ID is shown on the left, but isn't an explicit part of the table):

	`A`	`B`	`C`	`D`
t_1	good	1	Y	snow
t_2	bad	1	Y	cloudy
t_3	good	2	N	cloudy
t_4	ugly	3	Y	sun
t_5	good	2	Y	snow
t_6	bad	1	N	rain
t_7	ugly	3	Y	sun
t_8	bad	2	N	snow
t_9	bad	1	Y	rain

You should confirm there are nine tuples and four attributes.

(a) If you pick only one attribute as a class label and at least one attribute as a feature, how many different trees can you build?

(b) How many distinct functional dependencies are possible in this relation?

(c) If you find a functional dependency (or something reasonably close), how can that impact your tree?

2. Write a function `ID3tree` that reads two files:

   - a build tree file that has on the first line the name file to use, the second line the class label, and the third line a tab delimited list of attributes to use for features;
   - an input file that has on each line inputs to be classified.

   and displays the inputs with a probability associated with each possible class label.

   For example, supposed you want to use attributes `A` and `D` as features and `C` as the label from `table.txt` and you want to classify `bad rain` and `ugly snow`. We put this tuple into `test.txt`

   Here is the build tree file `tree.txt`

```
table.txt
C
A D
```

   Here is the input file `test.txt`

```
bad rain
ugly snow
```

   Here is a run:

```
>D3tree tree.txt test.txt

tuple: bad rain
class: 1 .5, 2 .4, 3 .1

tuple: ugly snow
class: 1 .4, 2 .4, 3 .2

>
```

3. Here's where we have some fun! Assume now that you will allow partial inputs – `*` means you don't care what the input is. For example, say in the `test.txt`, you don't care for the second input what the value of `D` is. So this updated file would look like:

   Here is the input file `test2.txt`

   ```
   bad rain
   ugly *
   ```

   Your program should still display the probability of class labels:

   ```
   >D3tree tree.txt test2.txt

   tuple: bad rain
   class: 1 .5, 2 .4, 3 .1

   tuple: ugly *
   class: 1 0, 2 .4, 3 .6

   >
   ```

   (a) What will your program give if the inputs are all `*`?

   (b) How will the class probabilities change when you add `*`?

   (c) Think of the limit of having all `*` from none. For example, assume you initially are classifying on attributes $A_1, A_2, \ldots, A_n$. You place one `*` for each attributes, then you place two for any two, three for any three, *etc.*. What is the limit of the class probabilities (revisit the previous two questions now if you want to)

4. Add 3-fold cross validation to your build. Split your table into approximatley three equal sizes $\Delta_1, \Delta_2, \Delta_3$. You will train and test three times.

Build	Test
$\Delta_1 \cup \Delta_2$	Δ_3
$\Delta_2 \cup \Delta_3$	Δ_1
$\Delta_3 \cup \Delta_1$	Δ_2

   Report the class label probabilites as an average of the probabilities from the test data. You will have to think how to report this to the user, since the averages will not necessarily sum to one.