LING 572 Hw2
Due: 11:55pm on Jan 21, 2010

The example files are under /dropbox/09-10/572/hw2/examples/.

Q1 (5 points): Run the Mallet DT learner (i.e., the trainer's name is DecisionTree) with **train.vectors.txt** as the training data and **test.vectors.txt** as the test data. In your note file, write down

(a) the commands you use, and

(b) the training accuracy and the test accuracy.

Q2 (10 points): Does the Mallet DT learner treat the features in the input file as binary or real-value? Prove your answer by writing a tool that binarizes the features and then comparing the classification results with or without first binarizing the features. In your note file, write down the commands you use for binarization, training, testing, and comparison.

Q3 (10 points): Run the Mallet DT trainer with different depths; that is, when running vectors2classify, replace **–trainer DecisionTree** with

```
  --trainer "new DecisionTreeTrainer(nn)''
```

where nn is the depth of the decision tree.

(a) Fill out Table 1

(b) What conclusion can you draw from Table 1?

Table 1: Decision Tree with different depths

Depth	Training accuracy	Test accuracy
1		
2		
4		
10		
20		
50		
100		
1000		

Q4 (50 points): Write a script, **build_dt.sh**, that builds a DT tree from the training data, classifies the training and test data, and calculates the accuracy.

- The DT learner should treat all features as binary; that is, the feature is considered present iff its value is nonzero.
- Use information gain to select features.
- The format is: build_dt.sh training_data test_data max_depth min_gain model_file sys_output > acc_file
- training_data and test_data are the vector files in the text format (cf. **train.vectors.txt**).
- max_depth is the maximum depth of the DT, and min_gain is the minimal gain; that is, split a node x if and only if the depth of x < max_depth AND the infoGain of the split ≥ min_gain.
- model_file shows the DT tree (cf. **model4**). Each line corresponds to a leaf node in the DT and it has the format: path training_instance_num c1 p1 c2 p2 ...
  Where path is the path from the root to the leaf node, training_instance_num is the number of the training examples that "reach" the leaf node, c_i is the class label, and p_i is the probability of c_i (i.e., the percentage of the training examples at the leaf node with the label c_i).
- sys_output is the classification result on the training and test data (cf. **sys4**). Except for the comment lines that start with %, all the other lines have the following format:
  instanceName true_class_label c1 p1 c2 p2 ...
- acc_file shows the confusion matrix and the accuracy for the training and the test data (cf. **acc4**). In the confusion matrix, a[i][j] is the number of instances where the truth is class i, and the system output is class j.
- As always, model4, sys4, and acc4 are NOT gold standard. These files were created with a much smaller training dataset.

Q5 (25 points): Run build_dt.sh with **train.vectors.txt** as the training data and **test.vectors.txt** as the test data. Fill out Table 2 (where min_gain is set to 0) and Table 3 (where min_gain is set to 0.1).

Table 2: Your decision tree results when min_gain=0

Depth	Training accuracy	Test accuracy	Wall clock time (in minutes)
1			
2			
4			
10			
20			
50			

Submission: Submit a tar file via CollectIt. The tar file should include the following.

- If your team has two people, please submit only one copy. In your note file, please list the names of team members.

Table 3: Your decision tree results when min_gain=0.1

Depth	Training accuracy	Test accuracy	Wall clock time (in minutes)
1			
2			
4			
10			
20			
50			

- In your note file hw2.*, include your answers to Q1-Q3 and Q5, and any notes that you want the TA to read.
- Shell scripts for Q2 and Q4, and related source and binary code.
- The data files produced in Q5 (e.g., **acc_file.4_0** is the acc_file when the depth is 4 and min_gain is 0).