**LING 572 – HW2**

**Q1**

Commands:

*info2vectors - -input train.vectors.txt - -output train.vectors*

*info2vectors - -input test.vectors.txt - -output test.vectors - -use-pipe-from train.vectors*

*vectors2classify - -training-file train.vectors - -testing-file test.vectors - -trainer DecisionTree > q1.stdout 2>q1.stderr*

Training accuracy = 0.638

Test accuracy = 0.523

**Q2**

The Mallet DT learner treats the features in the input file as binary value. The *compare.sh* shell script was created to compare the classification results before and after binarization. The *compare.sh* script would call a Perl script (*binarize.pl*) that was written to binarize the features, then followed by the Mallet info2vectors and vectors2classify commands.

The results before and after binarization will be output to screen and both show the same training and test accuracies.

At the command prompt type:

*compare.sh file1 file2*

file1 is the input training vector in text format

file2 is the input test vectors in text format

Commands use in *compare.sh* script:

*#!/bin/sh*

*rm train\_bin\_vectors 2>stderr*

*rm test\_bin\_vectors 2>stderr*

*perl binarize.pl $1 $2*

*info2vectors --input $1 --output train.vectors*

*info2vectors --input $2 --output test.vectors --use-pipe-from train.vectors*

*info2vectors --input train\_bin\_vectors --output train\_bin.vectors*

*info2vectors --input test\_bin\_vectors --output test\_bin.vectors --use-pipe-from train\_bin.vectors*

*vectors2classify --training-file train.vectors --testing-file test.vectors --trainer DecisionTree > q1.stdout 2>q1.stderr*

*vectors2classify --training-file train\_bin.vectors --testing-file test\_bin.vectors --trainer DecisionTree > q1\_bin.stdout 2>q1\_bin.stderr*

*clear*

*echo*

*echo Results Before Binarization:*

*grep -i Summary\* q1.stdout*

*echo*

*echo Results After Binarization:*

*grep -i Summary\* q1\_bin.stdout*

*echo*

**Q3**

Table 1: Decision Tree with different depths

|  |  |  |
| --- | --- | --- |
| Depth | Training accuracy | Test accuracy |
| 1 | 0.453 | 0.417 |
| 2 | 0.521 | 0.527 |
| 4 | 0.638 | 0.523 |
| 10 | 0.751 | 0.6 |
| 20 | 0.856 | 0.683 |
| 50 | 0.968 | 0.7 |
| 100 | 0.969 | 0.7 |
| 1000 | 0.969 | 0.7 |

From table 1, the training and test accuracies initially increased with more depth. However the accuracies would stop improving once we have crossed beyond the optimal depth level. In this experiment, the optimal level is achieved around depth 50. Thereafter, the training and test accuracies remain unchanged even if the depth level was increased up till 1000.

It was also noted that increasing levels come at the expense of consuming more computing resources. For example, although the training and test accuracies between depth level 50 and 100 were very close, it would take a longer time coupled with more memory to compute the results at depth level 1000.

**Q5**

Table 2: Classification results for Q2(c)-(e)

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

*End of HW2 – Joint submission by*

*Scott Mantei, Student ID:*

*Wee Teck Tan, Student ID: 0937003*

*Course Name: LING 572*