10-605 Homework #2 Report

Chen Sun
chens1

Did you receive any help whatsoever from anyone in solving this assignment? No.

Did you give any help whatsoever to anyone in solving this assignment? No.

1. Compare and discuss the performance for the full links vs full abstract data.
The training time and testing time of links is a little longer than that of the abstract data. Looking into the data, this is because most of the features in the links data are connected by "_", thus a single feature string is much longer than a word token. If stored directly in hashtable, the string may take up too much memory, thus it may print to the standard output more frequently than that of the abstract data. Even it was stored as the hashcode, the buffer to hold the string was still large. Thus, the training time is longer. With more frequent output, the sort may take more time and merge has more lines to proceed.
The precision rate has significant difference: the abstract data is 71.58% in full and the link is 62.95%. This is related to the numbers of features each training /test instances may have in the data. For each instance, the links data have fewer features than abstract data, even though they are longer in each instance.
In total, the vocabulary size for links is 994,469 and the size for abstract data is 2,109,714. This could show that the data in abstract data have a larger vocabulary, which may help to improve the precision in prediction.

2. Using a local copy of the RCV1.small train.txt le from Assignment 1, compare the performance of creating your Naive Bayes feature dictionary for last assignment and this assignment. Time all parts of the dictionary creation (including, for example, sorting and combining counts for your Assignment 2 solution)
In Assignment 1, the command is:

```
time cat RCV1.small_train.txt  | java NBTrain
```

In this Assignment 2, the command is:

```
time cat RCV1.small_train.txt  | java -Xmx128m NBTrain|sort -k1,1|java -Xmx128m NBTest
```

Trial	Assignment 1 Time(user)(s)	Assignment 2 Time(user)(s)
1	3.267	8.183
2	3.230	8.135
3	3.185	8.117

4	3.233	8.130
5	3.583	8.123
6	3.274	8.904
7	3.281	8.078
8	3.177	8.278
9	3.251	8.073
10	3.240	8.102
Avg	3.2721	8.2123

The time difference clearly shows that the running time of last assignment is quite smaller than that of this assignment. There might be several reasons: First of all, the first creation do not have a memory limitation. Program can use as much as memory as they can and keep the hashtable all in the memory. Secondly, the assignment 2 involves sorting. Since it is mergesort on locally, it taks O(NlogN) time to do sorting, which is also a time overhead on this.

3. How can we get the most informative features of a specific class given the trained naive Bayes model? In other words, if we are interested in figuring out the most predictive features for a class of wikipedia articles (e.g. German), what can we do?
Given the class the wikipedia articles, we can use something similar to cross validation to train the model and find the most common features among all the models. We can use k - 1 articles to train the model, and the left one article to do validation. After these, we can have k models, and find the common features among the models which have relatively high appearances in the data. That would be most predictive features in the article.