1. Our Name and UNI

Andrew Shu [ans2120@columbia.edu](mailto:ans2120@columbia.edu)

Ran Bi [rb2651@columbia.edu](mailto:rb2651@columbia.edu)

1. Files that we submit
2. Jave code files
3. Makefile
4. Readme
5. How to run our program
   1. Unpack the .tar.gz archive.
   2. Build the program using ‘make’
   3. Run the program using ‘make exec’
6. Internal design of our project
   1. First, we read the documents in the dataset and get every word occurring in these files. We use TreeMap to store the words and corresponding occurrences. Then generate COMMON&WORDS files according to their occurrences. Meantime, we build five tables using HashMap: 1. docIds: store the name of every document and its unique ID; 2. wordIds: store every word and its unique ID; 3. idWords: the reverse table of wordIds; 4. wordDocs: store every word ID and the IDs of documents which contain this word; 5. docWords: the reverse table of wordDocs.
   2. Do Apriori algorithm to find LargeItemSets. First, we get the values of minsup and minconf from the user. Generate the one item largeItemset from the file of sortedwords, which is also the WORDS file. Then, we use the function of Apriorigen to generate the candidate itemsets and prune some sets according to minsup.
   3. Generate association rules according the LargeItemSets. Due to we use the class of Itemset to store every largeItemset, we get the support value for every Itemset. Then for every Itemset, we compute the support value for every word in this Itemset. And we can get the confidence for every association rule by divide the support of Itemset by the support of each word. After that, prune some association rules by minconf.