Rohan Tandon
CS 446
02/24/17

Project 2 Report

1. The code for the part A is in Tokenizer and TokenizerDriver classes. The code for part B is in the MonteCristo and MonteCristoDriver classes.

2. Implementation:
    - Tokenization: The input files were read line by line using bufferedReader. Then each line was converted to character array and then parsed individually. Each character in the line was then compared to ensure it was only of type alphanumeric. Whenever a punctuation sign was encountered, a string storing the stored in an arraylist of strings and set to null. This ensured every punctuation would tokenize the words.
    - Stopword Removal: The lemur stopword list was read and all of the words were then stored in an arraylist of strings. Then, a list of tokenized words was compared against this stopword arraylist and those words which were in both of the arraylists were removed from the tokenized arraylist.

3. Libraries used:
    - Java.util.hashmap: For storing the most common words and their counts.
    - Java.util.*: For importing data structures like arraylists.
    - Java.io.BufferedReader/Writer: For reading the input files and then writing the outputs to a new file.

4. If an apostrophe is encountered in the middle of the world then in the tokenization algorithm, this word can be ignored because in many cases removing the apostrophe will change the importance of the word. Example: Not tokenizing in the word Rosie O'Donnell will preserve the importance of the word.

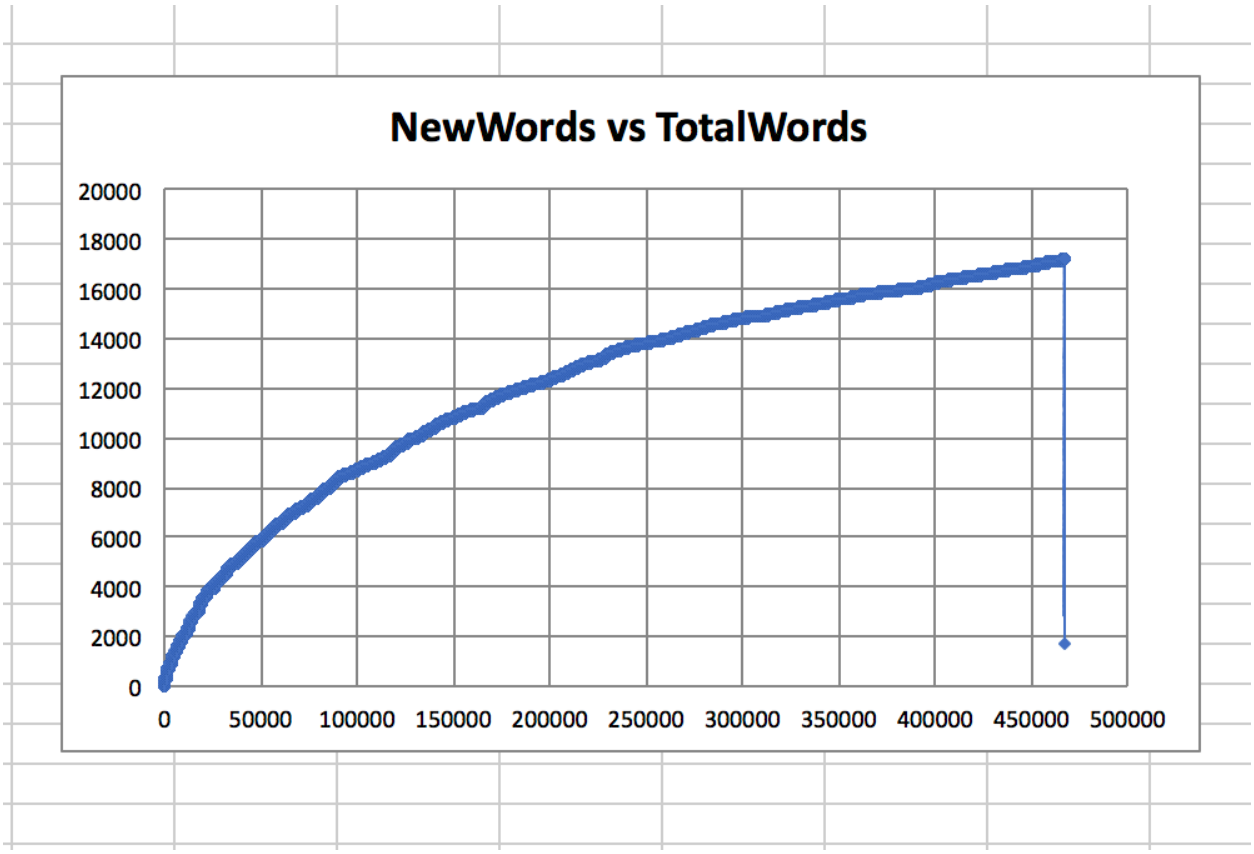
   In the stemming algorithm, the tense of the word can be identified and then all the words can be converted to the same tense. This will ensure that the word meanings are not changed completely.

5. Yes, many words that are related to the story have been mentioned the most. Example: Monte was mentioned 1136 times.

   No, there are no stopwords that are top terms but there are many irrelevant terms like "two", "time" that have been mentioned a lot.

   No, there can be no stopword list that is accurate for all documents because each document has its own set of repetitive but unimportant terms.

6.


NewWords vs TotalWords
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
0
50000
100000
150000
200000
250000
300000
350000
400000
450000
500000