Introduction to ML

# 1. Description

In this assignment you’ll be doing the following:

1. Read a raw text file containing movie reviews and ratings.
2. Remove the punctuations and stopwords (e.g. is, and, the, etc.) from the reviews and write the contents to a clean review file.
3. Associate a value for all words found in the clean reviews based on the average rating of the word.
4. Using the map between the words and their associated values, rate reviews from another file that doesn’t have any rating already.
5. Write the ratings of the reviews to an output file.

## 1.1. Clean the data

The raw data with the reviews and the rating will be present in a file named [rawReviewRatings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/rawReviewRatings.txt). Every review is given a rating between 0 - 4 (both inclusive). The following are interpretation of the rating values:

* 0 = Negative
* 1 = Somewhat Negative
* 2 = Neutral
* 3 = Somewhat Positive
* 4 = Positive

Each line of review follows the following format:

<rating> <review>

Example:

4 The Jungle Book is awesome!

where 4 is the rating and the text “The Jungle Book is awesome!” is the review.

Now, your first task is to read the contents of this file and **clean the data!**

The following steps needs to be performed in the same order for cleaning the data:

1. Read the raw reviews from an input file. e.g. [rawReviewRatings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/rawReviewRatings.txt)
2. Replace the hyphens in every line of text with spaces.  
   e.g. If the review contains the word “awe-inspiring”, then it should be split into 2 words namely “awe” and “inspiring”
3. Split each line of text into multiple words using the space character as the delimiter.  
   e.g. “The Jungle Book is awesome!” becomes | “The” | “Jungle” | “Book” | “is” | “awesome!” |
4. Remove the punctuation marks from the words.  
   e.g. “awesome!” becomes “awesome” and “!” becomes “”
5. If there are any trailing/leading whitespaces, then remove them. Remember that the space character (‘ ’), tab (‘\t’), newline (‘\n’), etc. are considered as whitespaces. For a complete list of whitespace characters [this](http://en.cppreference.com/w/cpp/string/byte/isspace).  
   e.g. “zootopia ” becomes “zootopia”, “hello\t” becomes “hello”, and “world\n” becomes “world”
6. Remove the empty words. i.e. words with length == 0.
7. Remove single lettered words. i.e. words with length == 1.
8. Remove stopwords. e.g. is, and, the, etc. The list of stopwords can be found in the file [stopwords.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/stopwords.txt)
9. Write the contents of the clean data to a file named [cleanReviewRatings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/cleanReviewRatings.txt).

## 1.2. Fill the dictionary

Using the [cleanReviewRatings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/cleanReviewRatings.txt) file, create a map/dictionary for every word that is found in this file. The dictionary is of type std::unordered\_map<string, std::pair<long, long>>.

We associate a pair of values for each word namely its total\_rating and its total\_count. For example, the word “fantastic” has occurred 3 times in the file [cleanReviewRatings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/cleanReviewRatings.txt) with associated ratings of 3, 4, and 3 (see the last 3 lines in the file). Therefore the word “fantastic” gets a total\_rating of 10 (i.e. 3 + 4 + 3 = 10) and a total\_count of 3 since it had occurred 3 times in the file. In a similar way, the other entries in this map are filled up as shown below. You may want to read about a [std::pair](http://en.cppreference.com/w/cpp/utility/pair) to understand more about how this map is organized.

|  |  |  |
| --- | --- | --- |
| **word** | **total\_rating** | **total\_count** |
| fantastic | 10 | 3 |
| Zootopia | 3 | 1 |
| Dory | 1 | 1 |
| Finding | 1 | 1 |
| worst | 0 | 1 |
| The | 6 | 2 |
| inspiring | 2 | 1 |
| Jungle | 8 | 2 |
| Book | 8 | 2 |
| good | 1 | 1 |
| Jack | 0 | 1 |
| awesome | 4 | 1 |
| Lion | 5 | 2 |
| King | 5 | 2 |
| awe | 2 | 1 |
| Jill | 0 | 1 |

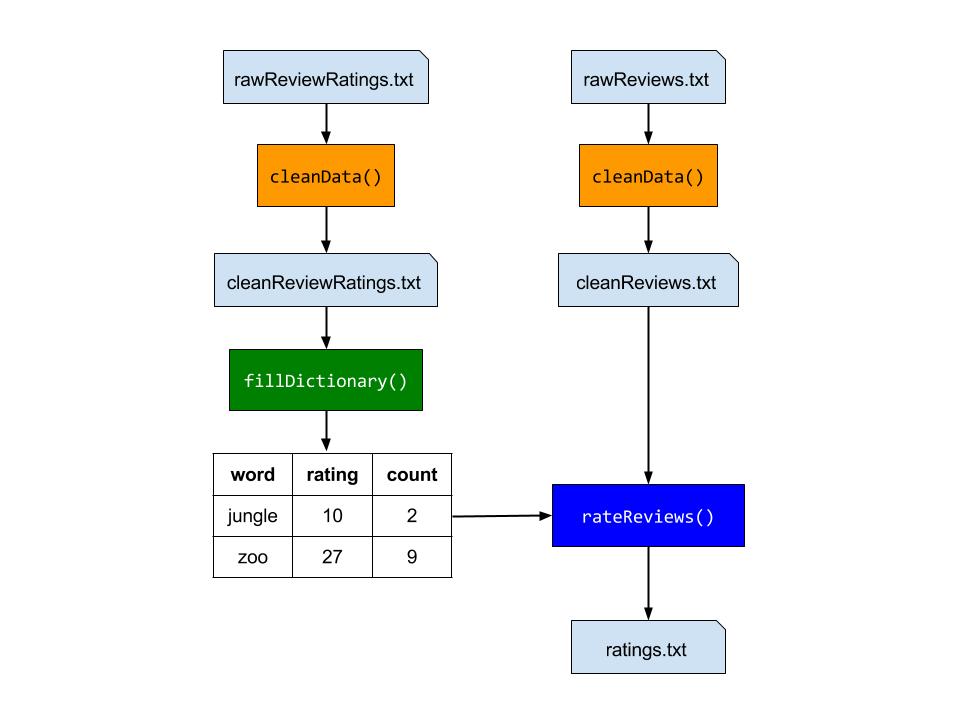
## 1.3. Rate new reviews

Using the map/dictionary that we created in the previous step, we are going to read new unrated reviews from a file (e.g. [rawReviews.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/rawReviews.txt)) and predict a rating for each review in this file. The predicted rating for each review is written to an output file named [ratings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/ratings.txt).

How do we predict the ratings for the unrated reviews?

1. Read the input file with the unrated raw reviews (e.g. [rawReviews.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/rawReviews.txt)).
2. Clean the data and produce an output file named [cleanReviews.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/cleanReviews.txt). The process for cleaning the data is exactly the same as we did in step 2.1.
3. Rate each review by finding the rating for each word from the map/dictionary that we created in step 2.2. The rating for a line of review is the average value of the rating of all the words in the review. If some word in this unrated review is not found in the map/dictionary, then that word is given a neutral rating of 2. If a review is empty (i.e. the review contains no words in it), then such a review is also given a neutral rating of 2.  
     
   e.g. Let see how we computed the rating for the 2nd review in [cleanReviews.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/cleanReviews.txt) (i.e. “The Lion King fantastic”). We lookup the map/dictionary that we created before and get the average rating for each word in this review. The average ratings of each word in this review is shown below:  
   The: 6/2 = 3  
   Lion: 5/2 = 2.5  
   King: 5/2 = 2.5  
   fantastic: 10/3 = 3.33  
     
   Based on these individual values, this line gets and average review of 2.83 as shown in the file [ratings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/ratings.txt).  
     
   Rating for this line = (3 + 2.5 + 2.5 + 3.33) / 4 = 2.83  
   We are dividing by 4 since this review contains 4 words in total.  
     
   Another example: “Finding Nemo great”  
   Finding: 1  
   Nemo: 2 (because it is NOT found in the map)  
   great: 2 (because it is also NOT found in the map)  
     
   Rating for this line = (1 + 2 + 2) / 3 = 1.67
4. Write the ratings for all the reviews to an output file named [ratings.txt](http://pages.cs.wisc.edu/~gerald/cs368/assignments/a2/ratings.txt).

A high-level process diagram of the steps explained in 2.1 - 2.3 is shown. Hope this visualization helps you to understand the idea in a better way! :)



That’s it! Now, after completing this step you may pat yourself on the back for doing an amazing job with this assignment! :) Also, if you have not taken Artificial Intelligence or Machine Learning before then you have just now completed your first exercise in Machine Learning. You just created a program that may predict the ratings of movie reviews based on some learning it did before based on some reviews that already had ratings associated with them. Well, even though our algorithm is very simple, this is the basic idea behind Machine Learning. Newcomers, welcome to the world of Artificial Intelligence! :)