CMSC476 HW2 - Term Weighting
Phillip Thompkins

**A Note:** Shell scripts are not attached, as I had to have a friend run the program on their computer back in the Spring, and they were unable to get the shell scripts. Luckily, I do have the runtimes for varying document corpus sizes.

**Code Improvements:**

My code for this assignment builds heavily upon the code from the first. One of the first updates I made to my code was to expand upon some of the term handlings by also searching for symbols I had neglected to include in past, like the @ symbol, and symbology from the Spanish language, like the upside-down exclamation points and question marks, as some of the documents in the corpus were in Spanish. I also added the functionality to remove words with a number in them, in case there was a word such as “5percent,” as unlikely as I thought it would be. Then I tackled mandated changes; the first that I implemented was to remove the stopwords as outlined in the stopwords.txt document. I also added preprocessing functionality to remove words that only occurred once in the corpus and words of singular length, like “a” or “I”.

Following that, I began to implement the term weighting system. The exact weighting formula I had used was (A term’s number of occurrences within the document / Total number of term occurrences in the document) * 100 to get a simple, straightforward percentage. The term weightings are stored in an array with indices that correspond to the term’s index in the token library array. Upon output, the weightings were printed in the following format: “term: weight”.

**Data:**

Here are weights post-processing for two documents:

| **256.html** | **129.html** |
|---|---|
| elementary: 0.054945 | robots: 0.100917 |
| school: 0.042125 | public: 0.045872 |
| education: 0.034799 | page: 0.036697 |
| teaching: 0.031136 | visit: 0.036697 |
| children: 0.018315 | hospitals: 0.036697 |
| examines: 0.016484 | ep: 0.036697 |
| grade: 0.014652 | robot: 0.036697 |
| reading: 0.014652 | event: 0.027523 |
| learning: 0.014652 | colleges: 0.027523 |
| students: 0.014652 | standard: 0.027523 |
| program: 0.012821 | museums: 0.027523 |
| studies: 0.012821 | programs: 0.027523 |

While examining my runtimes compared to the graph for hw1.py (I had lost the specific numbers), I noticed that the code for Homework 2 led to faster runtimes. This makes sense, as preprocessing removed one-use words, words like “a”, and anything that was listed as a stopword. I have consolidated the runtime data into a table below.

| Corpus Size | hw1.py runtime (seconds) | hw2.py runtime (seconds) |
|---|---|---|
| 100 | 1.5 | 1.4 |
| 200 | 2.0 | 1.8 |
| 300 | 4.0 | 3.5 |
| 400 | 8.5 | 7.8 |
| 503 | 13.5 | 12.8 |