Lab 2

Reza Shisheie

2708062

Problem Statement:

The goal of this project is mining text and get cosine similarity of different html pages.
The following step are taken to get the final results:

1. Assumptions
2. Data needs to be extracted, cleaned, and saved in a txt file.
3. Each file will be opened and the number of occurrence of each word list is counted and saved in an array
4. The previously generated array has to be cleaned by removing the occurrence of data which is a subdomain of another word
5. Using the cleaned array cosine similarity is calculated.

Problem Description:

There are a few

1. ***Assumptions:***
   To calculate the correct occurrence of each word, first all other variants of that word has to be counted. For instance all variants of engineering are engineer, engineers, and engineering. These are saved into a word list for each word and they are counted in each text. Here is a list of all of them:

   ```
   word = [['engineering', 'engineer', 'engineers'],
           ['professor', 'professors', 'prof'],
           ['research', 'researcher', 'researchers', 'researching'],
           ['data'],
           ['mining'],
           ['data mining']]
   ```

   This should be noted that those exact words ONLY are counted and for instance the word “searched” is not counted under the category “research”

2. ***def fileSave(url),*** **Data extraction, cleaning, and saving:**
   This whole process is done in this function. Data is extracted first. Then html data is converted to text using “BeautifulSoup” library. In some saved data Java code was found, thus they are removed by “script.extract()”. Then all data is saved into “Doc.txt” files.
   One important note about this section is that the saved text has special characters such as “?” and “!”, and etc. The proper way to do this stage is properly clean all these characters by removing all of them and all text has to be sorted and saved into a tokenized list and one by one the occurrence is calculated. To find bigrams, however, one needs to look for the first word in the list. Once found the second word has to be compared with the second word in bigram if they are the same, it has to be counted for the bigram occurrence. This stage, however, is very time consuming both on implementation and running. To provide a simpler solution I am using a predefined library which is very accurate but might have some small error. This helps me to

skip the complexity of data cleaning and get to analysis, while making sure that I understand the proper way to do this stage.

3. ***def count_words(array, url, the_word, jj),* Counting the occurrence of each word list:**
   In this process, occurrence of each word list is calculated. Using “re” library and “re.findall” command I can find the occurrence of each exact term and data is saved into an array.

4. **def cleanDup(array, col, des), Cleaning up counts:**
   In stage 3 all occurrence of each word list is calculated and saved into array. However, all “data” and “mining” words are also counted under “data mining” too. This can cause error as a page which is just about “data” in general yields a very high similarity to a page which is just about “data mining”. In order to avoid this problem all occurrence of “data mining” is removed from the list of “data”. The easiest way is subtracting the number of occurrence of “data mining” from “data” and “mining”

Results:

here is a screen shot of results

```
arsh@arsh-Precision-5520:~/Dropbox/Academia/CIS660/Lab 2$ python lab2_1.0.py
started...

files saved...

+------+-------------+-----------+----------+-------+--------+-------------+
|      | engineering | professor | research |  data | mining | data mining |
+------+-------------+-----------+----------+-------+--------+-------------+
| Doc1 |     3.0     |    0.0    |   4.0    |  0.0  |  0.0   |     0.0     |
| Doc2 |     5.0     |    2.0    |   3.0    |  1.0  |  0.0   |     0.0     |
| Doc3 |     0.0     |    0.0    |   11.0   |  0.0  |  0.0   |     0.0     |
| Doc4 |     6.0     |    0.0    |   23.0   | 273.0 | 178.0  |    148.0    |
| Doc5 |     6.0     |    0.0    |   23.0   | 273.0 | 178.0  |    148.0    |
+------+-------------+-----------+----------+-------+--------+-------------+
array saved...

+------+-------------+-----------+----------+-------+--------+-------------+
|      | engineering | professor | research |  data | mining | data mining |
+------+-------------+-----------+----------+-------+--------+-------------+
| Doc1 |     3.0     |    0.0    |   4.0    |  0.0  |  0.0   |     0.0     |
| Doc2 |     5.0     |    2.0    |   3.0    |  1.0  |  0.0   |     0.0     |
| Doc3 |     0.0     |    0.0    |   11.0   |  0.0  |  0.0   |     0.0     |
| Doc4 |     6.0     |    0.0    |   23.0   | 125.0 |  30.0  |    148.0    |
| Doc5 |     6.0     |    0.0    |   23.0   | 125.0 |  30.0  |    148.0    |
+------+-------------+-----------+----------+-------+--------+-------------+
array cleaned...

+------+----------------+----------------+----------------+----------------+----------------+
|      |      Doc1      |      Doc2      |      Doc3      |      Doc4      |      Doc5      |
+------+----------------+----------------+----------------+----------------+----------------+
| Doc1 |      1.0       | 0.864692030547 |      0.8       | 0.111409900701 | 0.111409900701 |
| Doc2 | 0.864692030547 |      1.0       | 0.480384461415 | 0.18164222837  | 0.18164222837  |
| Doc3 |      0.8       | 0.480384461415 |      1.0       | 0.116473987097 | 0.116473987097 |
| Doc4 | 0.111409900701 | 0.18164222837  | 0.116473987097 |      1.0       |      1.0       |
| Doc5 | 0.111409900701 | 0.18164222837  | 0.116473987097 |      1.0       |      1.0       |
+------+----------------+----------------+----------------+----------------+----------------+
cosine smilarity done
```

the first table is occurrence of each word list without removing the occurrence of subdomain words. The second table is the occurrence of each word list after removing the subdomain words, and the last one is the cosine similarity table.

As you see the cosine similarity of Doc4 and Doc5 are the same as the html reader read the whole Wikipedia page on “data mining”.

“Doc1” and “Doc2” has a 0.86 percent similarity as both pages are from school and school-related words are counted on them.

“Doc3” has better similarity with “Doc1” and “Doc2” than “Doc4” and “Doc5”. It is the Cleveland Clinic page and it is much closer to school pages than Wikipedia page.

“Doc1” and “Doc2” have the least similarity with “Doc4” and “Doc5” as one is a Wikipedia page and the other one is a school page.

The same inductions can be made from the number of occurrence of each word too. The occurrence of “engineering”, “professor” and “research” in “Doc1” and “Doc2” is much more than “data”, “mining”, and “data mining”. While it is opposite in “Doc4” and “Doc5”. “Doc3”, however, is more tilted toward “Doc1” and “Doc2” as it only has occurrence of the word “research” which is dominant in “Doc1” and “Doc2”. This induction is aligned with the same induction made in the previous section.