

CS 432

Make-Up Extra Credit

For Assignments 3 and 4

Michelle Graham
Spring 2017

Assignment 3

In order to make this information more accurate, I extended the previous URIs from Assignment 3. I also separated the raw and processed files into their own folders. The, I used “grep” to find keywords and the number of times they occur in each processed file; this is stored in a file called “hits.txt” in the NewProcessed folder. I also used “wc” to find the word count in each processed file in the list of ten.

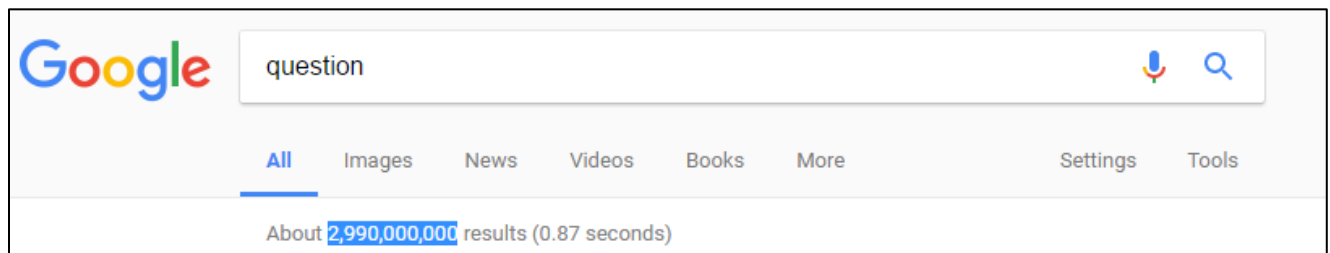
Examples:

```
mgrah@DESKTOP-30IR4AC MINGW64 /c/CS432/A3/NewProcessed
$ grep -o question processed_105.txt | wc -l
3
```

```
MINGW64:/c/CS432/A3/NewProcessed
mgrah@DESKTOP-30IR4AC MINGW64 /c/CS432/A3/NewProcessed
$ wc -w processed_105.txt
1048 processed_105.txt
```

New word search:

I have attached the Excel file where this information is stored.



The screenshot shows a Google search interface. The search bar contains the word "question". Below the search bar, there are tabs for "All", "Images", "News", "Videos", "Books", and "More". The "All" tab is selected. Below the tabs, it says "About 2,990,000,000 results (0.87 seconds)".

TFIDF	TF	IDF	URI	Document Name
0.0212	0.0029	7.4150	http://www.bbc.co.uk/news/uk-politics-39064149%0A	processed_105.txt
0.0009	0.0010	0.9393	https://www.washingtonpost.com/news/morning-mix/wp/2017/02/23/how-an-ohio-blogger-discovered-a-runner-a-harvard-grad-cheating-in-a-half-marathon-across-the-county/?tid=sm_fb&utm_term=.987daf88eb64%0A	processed_132.txt
0.0007	0.0007	0.9393	http://www.truth-out.org/news/item/39601-trump-s-anti-trans-acti	processed_135.txt
0.0016	0.0017	0.9393	http://www.independent.co.uk/news/world/americas/keith-ellison	processed_14.txt
0.0010	0.0011	0.9393	http://news.sky.com/story/stoke-central-by-election-labour-beats-u	processed_143.txt
0.0008	0.0009	0.9393	http://www.bbc.co.uk/news/business-39049889%0A	processed_145.txt
0.0013	0.0014	0.9393	http://www.dailymail.co.uk/news/article-4254246/amp/Trump-wan	processed_151.txt
0.0009	0.0009	0.9393	http://www.bbc.co.uk/news/world-asia-39073389%0A	processed_158.txt
0.0012	0.0013	0.9393	http://www.someecards.com/news/weird-news/witches-donald-tru	processed_159.txt
0.0042	0.0045	0.9393	http://www.4-traders.com/news/Yale-University-Fructose-is-genera	processed_164.txt

WC	Term Frequency	Total Documents	Doc hits
1048	3	2,990,000,000	115
	1		
1033			
1402	1		
1195	2		
941	1		
1128	1		
724	1		
1084	1		
756	1		
443	2		

Ranked from Lowest TF-IDF to Highest:

Rank in Order	URI in Order
1	http://www.truth-out.org
2	http://www.bbc.co.uk
3	http://www.bbc.co.uk
4	https://www.washingtonpost.com
5	http://news.sky.com
6	http://www.someecards.com
7	http://www.dailymail.co.uk
8	http://www.independent.co.uk
9	http://www.bbc.co.uk
10	http://www.4-traders.com

The new Page Ranks:

Page Rank	URI
0.7	http://www.bbc.co.uk
0.0	https://www.washingtonpost.com
0.7	http://www.truth-out.org
0.8	http://www.independent.co.uk
0.7	http://news.sky.com
0.9	http://www.bbc.co.uk
0.7	http://www.dailymail.co.uk
0.9	http://www.bbc.co.uk
0.6	http://www.someecards.com
0.5	http://www.4-traders.com

Ranked from Lowest PR to Highest:

Rank in Order	URI in Order
1	https://www.washingtonpost.com
2	http://www.4-traders.com
3	http://www.someecards.com
4	http://www.bbc.co.uk
5	http://www.truth-out.org
6	http://news.sky.com
7	http://www.dailymail.co.uk
8	http://www.independent.co.uk
9	http://www.bbc.co.uk
10	http://www.bbc.co.uk

Question 4

To obtain the Kendall tau_b score, I attempted to utilize a library in Python called, “Scipy.” However, it was not able to be installed correctly for some unknown reason. If I had been able to install it properly, this is an example of what I would have tried to implement:

```
import scipy
import scipy.stats as stats
x1 = [12, 2, 1, 12, 2]
x2 = [1, 4, 7, 1, 0]
tau, p_value = stats.kendalltau(x1, x2)
```

Instead, I used the “Kendall” package in rStudio in order to obtain the correlation.

Data:

```
> TFIDF <-c(0.0212,0.0009,0.0007,0.0016,0.0010,0.0008,0.0013,0.0009,0.0012,0.0042)
> PR <-c(0.7,0,0.7,0.8,0.7,0.9,0.7,0.9,0.6,0.5)
```

Kendall Summary:

```
> summary(Kendall(PR,TFIDF))
Score = -7 , Var(Score) = 114.4889
denominator = 40.8901
tau = -0.171, 2-sided pvalue =0.57497
```

Question 5

Alexa provides an easy feature that generates a local (by website country origin) ranking along with a global ranking. My results are surprisingly lower than I expected them to be! On an added note, one of the cites had a drastic increase in rank in the UK as I was going through my list- [bbc.co.uk](http://www.bbc.co.uk). As you can see, it is duplicated as a base source three times! Its rank went from 79 to 5 in just a few short moments! The Alexa rank is based on the Global Rank.

Alexa Ranks:

US or UK Rank	Global Rank	URI
5	105	http://www.bbc.co.uk
42	217	https://www.washingtonpost.com
7671	27277	http://www.truth-out.org
254	478	http://www.independent.co.uk
61	1391	http://news.sky.com
5	105	http://www.bbc.co.uk
101	161	http://www.dailymail.co.uk
5	105	http://www.bbc.co.uk
2290	8495	http://www.someecards.com
25919	33620	http://www.4-traders.com

Ranked from Lowest Alexa Rank to Highest:

Rank in Order	URI in Order
1	http://www.bbc.co.uk
2	http://www.bbc.co.uk
3	http://www.bbc.co.uk
4	http://www.dailymail.co.uk
5	http://www.washingtonpost.com
6	http://www.independent.co.uk
7	http://news.sky.com
8	http://www.someecards.com
9	http://www.truth-out.org
10	http://www.4-traders.com

Question 7

Python has an outdated library called “InvertedIndex” that I wanted to try to use to make an inverted file. Unfortunately, it is not compatible with Python 2.7 and did not have enough time to reconstruct it in 2.7. This issue also occurred for another library I found called “collections.” I would like to learn more about this process in the future.

InvertedIndex Example:

```
1  import InvertedIndex
2  import InvertedIndexQuery
3
4  i = InvertedIndex.Index()
5
6  filename = 'Document1.txt'
7  file_to_index = open(filename).read()
8  document_key = filename
9
10 # index the document, using document_key as the document's
11 # id.
12 i.index(file_to_index, document_key)
13
14 filename = 'document2.txt'
15 file_to_index = open(filename).read()
16 document_key = filename
17
18 i.index(file_to_index, document_key)
19
20 search_results = InvertedIndexQuery.query('Question', i)
21 search_results.sort()
22
23 cnt = 0
24 for document in search_results:
25     cnt = cnt + 1
26     print '%d) %s' % (cnt, document[1])
```

Collections Example:

```
1  import collections
2  inverted_index = collections.defaultdict(set)
3  for line in open("/usr/share/dict/words"):
4      word = line.strip().lower() # ignore case
5      for letter in word:
6          inverted_index[letter].add(word)
```

Resources

<http://www.alexa.com/siteinfo>

<https://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.stats.kendalltau.html>

<https://cran.r-project.org/web/packages/Kendall/Kendall.pdf>

<https://www.youtube.com/watch?v=V4MgE43SrgM>

http://www2.warwick.ac.uk/fac/sci/moac/people/students/peter_cock/python/rank_correlations/

<http://legacy.python.org/workshops/1996-11/papers/InvertedIndex.html>

http://www.dalkescientific.com/writings/diary/archive/2011/12/23/inverted_index.html

Assignment 4

Question 3

I keep running into the issue of finding outdated resources for Python. I spent a lot of time trying to figure out a more recent way to handle the LinkedIn data but the way the keys are set up currently gave me some trouble. I have attached a link to the resource I was interested in using.

Resources

<https://rawgit.com/ptwobrussell/Mining-the-Social-Web-2nd-Edition/master/ipynb/html/Chapter%203%20-%20Mining%20LinkedIn.html>