

ASSIGNMENT 3

CS 432 Web Science

Mackenzie Kerchner

Contents

Problem 1	2
Problem 2	4
Problem 3	7
Problem 5	8
Problem 6	9

Problem 1























1. Download the 1000 URIs from assignment #2. "curl", "wget", or "lynx" are all good candidate programs to use. We want just the raw HTML, not the images, stylesheets, etc. Now use a tool to remove (most) of the HTML markup for all 1000 HTML documents. "python-boilerpipe" will do a fair job. Keep both files for each URI (i.e., raw HTML and processed). Upload both sets of files to your GitHub account.

Solution

```
1 import urllib2
2 import justext
3
4 import sys # weird unicode stuff
5 reload(sys)
6 sys.setdefaultencoding('utf8')
7
8 """
9     IMPORTANT: I am using justext for this, which is not perfect, and sometimes gets html tags.
10    because of this, the hash values can vary between each running of the program.
11    each file set must be rebuilt for each use because the hash's may not match
12 """
13
14 # loads the uri from a file into a list
15 initialUrlList = []
16 infile = open("tweets.json", "r")
17 for line in infile:
18     initialUrlList.append(line.strip()) # strip prevents extra unicode stuff, ex: /n for newline on each line
19
20 for link in initialUrlList:
21     try:
22         response = urllib2.urlopen(link)
23         html = response.read()
24         #print html
25         hash_name = str(hash(html)) # hash as string to name each html download
26         name = "C:\\Users\\Mack\\PycharmProjects\\anwala.github.io\\venv\\downloads\\" + hash_name
27         print name
28         with open(name, "w") as outfile:
29             outfile.write('%s\n' % html)
30         boilerplate_full = []
31         boilerplate = justext.justext(html, justext.get_stoplist("English"))
32         name = name + "~boilerplate"
33         with open(name, 'w'):
34             print "" # do nothing, just clear any past entries into the files # replace with pass?
35         for paragraph in boilerplate:
36             if not paragraph.is_boilerplate:
37                 # check for empty files?
38                 with open(name, "a") as outfile:
39                     outfile.write('%s\n' % paragraph.text)
40     except:
41         pass
```

I didn't bother to sort this part into functions as it all fits 'neatly' into a few dozen lines. The list of URIs is grabbed at the top, then each one is read into a stream and saved into a string. This string is hashed, a file with this name is created and written with the full html text. The same string is fed into a module to get the boilerplate text, and that is written to another file with the same hash with ~boilerplate appended to the name.

I used justext as I could not get boilerpipe to work at all, even by using what Joshua posted in the slack. I also attempted to use python 3.7 and boilerpipe3 and it simply would not run.

 -2146989611~boilerplate	2/28/2019 11:14 AM	File	3 KB
 -2146989611	2/28/2019 11:14 AM	File	73 KB
 -2146643985~boilerplate	2/28/2019 11:11 AM	File	2 KB
 -2146643985	2/28/2019 11:11 AM	File	65 KB
 2141791013~boilerplate	2/28/2019 11:14 AM	File	4 KB
 2141791013	2/28/2019 11:14 AM	File	66 KB
 2137089764~boilerplate	2/28/2019 11:12 AM	File	5 KB
 2137089764	2/28/2019 11:12 AM	File	66 KB
 2134123047~boilerplate	2/28/2019 11:11 AM	File	1 KB
 2134123047	2/28/2019 11:11 AM	File	57 KB
 2133820042~boilerplate	2/28/2019 11:24 AM	File	3 KB
 2133820042	2/28/2019 11:24 AM	File	64 KB
 -2129737768~boilerplate	2/28/2019 11:10 AM	File	4 KB
 -2129737768	2/28/2019 11:10 AM	File	68 KB
 -2125527066~boilerplate	2/28/2019 11:14 AM	File	4 KB
 -2125527066	2/28/2019 11:14 AM	File	63 KB
 -2124758244~boilerplate	2/28/2019 11:09 AM	File	8 KB
 -2124758244	2/28/2019 11:09 AM	File	73 KB
 -2116106043~boilerplate	2/28/2019 11:24 AM	File	3 KB
 -2116106043	2/28/2019 11:24 AM	File	67 KB
 -2115958976~boilerplate	2/28/2019 11:24 AM	File	6 KB
 -2115958976	2/28/2019 11:24 AM	File	76 KB

Above is a portion of the resulting folder.

Problem 2

2. Choose a query term, if the term is present in more than 10 documents, choose any 10 from your list. As per the example in the week 5 slides, compute TFIDF values for the term in each of the 10 documents and create a table with the TF, IDF, and TFIDF values, as well as the corresponding URIs. The URIs will be ranked in decreasing order by TFIDF values.

Solution

```
12 # finds the idf for the query term
13 while termIn < 10:
14     termIn = 0
15     keyword = "democrat"
16     # keyword = raw_input('please enter query term: ')
17     if secondpass == 1: # prevents an infinite loop on bad hardcoded keywords
18         keyword = raw_input('please enter query term: ')
19
20
21     totalDocs = 0
22
23     for filename in glob.glob(os.path.join(path, '*~boilerplate')):
24         with open(filename, 'r') as file:
25             textbody = file.read().lower() # extra step because obviously word and Word are different words
26             wordcount = Counter(textbody.split())
27             if len(textbody.split()) > 0:
28                 totalDocs += 1
29                 if wordcount[keyword] > 0:
30                     termIn += 1
31
32     # print "total documents: ", totalDocs
33     # print "documents containing term: ", termIn
34     if float(termIn) > 0:
35         idf = round(totalDocs / float(termIn), 4)
36     else:
37         pass
38     #print "test: ", float(totalDocs)/3
39     if termIn == 0:
40         print "no documents with that term"
41     elif termIn < 10:
42         print "only", termIn, " documents with that term"
43     elif termIn > float(totalDocs) / 3: # warns you of oversaturated terms
44         temp = raw_input('A large amount of documents contain this term, would you like to query a different term? ')
45         if temp == "y" or temp == "yes":
46             termIn = 0
47         secondpass = 1
48
49 idf = round(math.log(idf, 2), 4)
```

In ranker.py, I used the code from [1] to read every boilerplate file in the downloads folder containing all the relative files. When the query is entered (or hardcoded for testing) the documents are checked simply for containing the term. If there are too few documents, a new query term is requested, and if there are a large number of documents, the option is given to enter a different term. If there are an acceptable number of documents, the IDF is calculated and the program continues.

```
50     print keyword, "appears in ", termIn, " out of ", totalDocs, " documents"
51
52     print "TFIDF    TF        IDF        NAME"
53     print "-----"
54
55     # finds the tf and tfidf for each document containing the query term
56     for filename in glob.glob(os.path.join(path, '*~boilerplate')):
57         with open(filename, 'r') as file:
58             try:
59                 shortfilename = filename[10:-12] # filename without path
60                 htmlpath = filename[:-12] # path to full html file
61                 # mainEntityOfPage is where the uri is located in the html file
62
63                 textbody = file.read().lower() # extra step because obviously word and Word are different words
64                 wordcount = Counter(textbody.split())
65                 #print filename
66                 term = round(float(wordcount[keyword]), 4)
67                 total = len(textbody.split())
68                 tf = round(term / total, 4)
69                 #tf = round(tf, 4)
70                 tfidf = round(tf * idf, 4)
71                 if tfidf > 0:
72                     # absolutely disgusting method to format columns but it works for something this simple
73                     if tfidf % .001 == 0:
74                         tfidf = str(tfidf)+"0"
75                     if tf % .001 == 0:
76                         tf = str(tf)+"0"
77
78                     print tfidf, " ", tf, " ", idf, " ", shortfilename
79                     tfidf = float(tfidf)
80                     tf = float(tf)
81                     """
82                     # neater method but not working yet
83                     print ("{:f}".format(tfidf),
84                           tf,
85                           idf,
86                           shortfilename)
87                     """
88             except:
89                 pass # weird errors
90
```

The top of the table is printed, and the boilerplate text is read again to calculate each document's TF. The TFIDF and TF of each document (rounded to 4 decimal places) is printed out into rows.

Grabbed 10 rows from console output, and put them into a table and sorted by TFDIF

TFIDF	TF	IDF	URI
0.0397	0.0119	3.3321	https://www.foxnews.com/politics/beto-orourke-fandom-escalates-in-texas-ahead-of-senate-race
0.0253	0.0076	3.3321	https://www.foxnews.com/politics/state-of-the-midterms-beto-orourke-seeks-to-capitalize-on-endorsements-as-governors-races-shift-right
0.0193	0.0058	3.3321	https://www.foxnews.com/politics/arizona-senate-race-mcsally-supporters-try-capitalizing-off-sinemas-crazy-remark
0.0183	0.0055	3.3321	https://www.foxnews.com/politics/obama-trump-make-final-midterm-push-in-florida
0.0157	0.0047	3.3321	https://www.foxnews.com/politics/mcsally-sinema-target-arizonas-undecided-voters-as-senate-race-enters-final-days
0.0133	0.0040	3.3321	https://www.foxnews.com/politics/texas-senate-race-now-costliest-in-us-history-as-beto-rakes-in-despite-trailing-cruz
0.0117	0.0035	3.3321	https://www.foxnews.com/politics/bloomberg-spent-9-5m-on-ads-against-two-california-republicans-data-show
0.0060	0.0018	3.3321	https://www.foxnews.com/politics/michigan-journalist-accidentally-says-f-ing-republican-john-james-and-victory-over-democrat-would-s
0.0047	0.0014	3.3321	https://www.foxnews.com/opinion/doug-schoen-in-a-divided-america-we-all-lose
0.0040	0.0012	3.3321	https://www.foxnews.com/opinion/dont-blame-trump-hypocritical-democrats-started-dragging-us-into-the-political-gutter-long-ago

Problem 3

3. Now rank the same 10 URIs from question #2, but this time by their PageRank.
Create a table similar to Table 1.
Briefly compare and contrast the rankings produced in questions 2 and 3.

Solution

Solution goes here

Page rank	URI
7/10	https://www.foxnews.com/politics/beto-orourke-fandom-escalates-in-texas-ahead-of-senate-race
7/10	https://www.foxnews.com/politics/state-of-the-midterms-beto-orourke-seeks-to-capitalize-on-endorsements-as-governors-races-shift-right
7/10	https://www.foxnews.com/politics/arizona-senate-race-mcsally-supporters-try-capitalizing-off-sinemas-crazy-remark
7/10	https://www.foxnews.com/politics/obama-trump-make-final-midterm-push-in-florida
7/10	https://www.foxnews.com/politics/mcsally-sinema-target-arizonas-undecided-voters-as-senate-race-enters-final-days
7/10	https://www.foxnews.com/politics/texas-senate-race-now-costliest-in-us-history-as-beto-rakes-in-despite-trailing-cruz
7/10	https://www.foxnews.com/politics/bloomberg-spent-9-5m-on-ads-against-two-california-republicans-data-show
7/10	https://www.foxnews.com/politics/michigan-journalist-accidentally-says-f-ing-republican-john-james-and-victory-over-democrat-would-s
No rank	https://www.foxnews.com/opinion/doug-schoen-in-a-divided-america-we-all-lose
No rank	https://www.foxnews.com/opinion/dont-blame-trump-hypocritical-democrats-started-dragging-us-into-the-political-gutter-long-ago

Because I parsed tweets by user rather than most recent overall, an overwhelming majority of my links lead to the same domain. Foxnews/politics has a PageRank of 7/10, while foxnews/opinion has no page rank.

Problem 5

Compute a ranking for the 10 URIs from Q2 using Alexa information (see week 4 slides). Compute the correlation (as per Q4) for all pairs of combinations for TFIDF, PR, and Alexa.

Solution

All the links have the same domain and the same Alexa rank, 237, as of the writing of this report.

Problem 6

Give an in-depth analysis, complete with examples, graphs, and all other pertinent argumentation for Kristen Stewart's (of "Twilight" fame) Erdos-Bacon number.

Solution

Kristen Stewart has an Erdos-Bacon number of 7. Her Erdos number is 5, owing to co-authoring a paper on AI, and she has a Bacon number of 2 due to sharing the Twilight spotlight with Michael sheen, who co-starred with Kevin Bacon in Frost/Nixon [2].

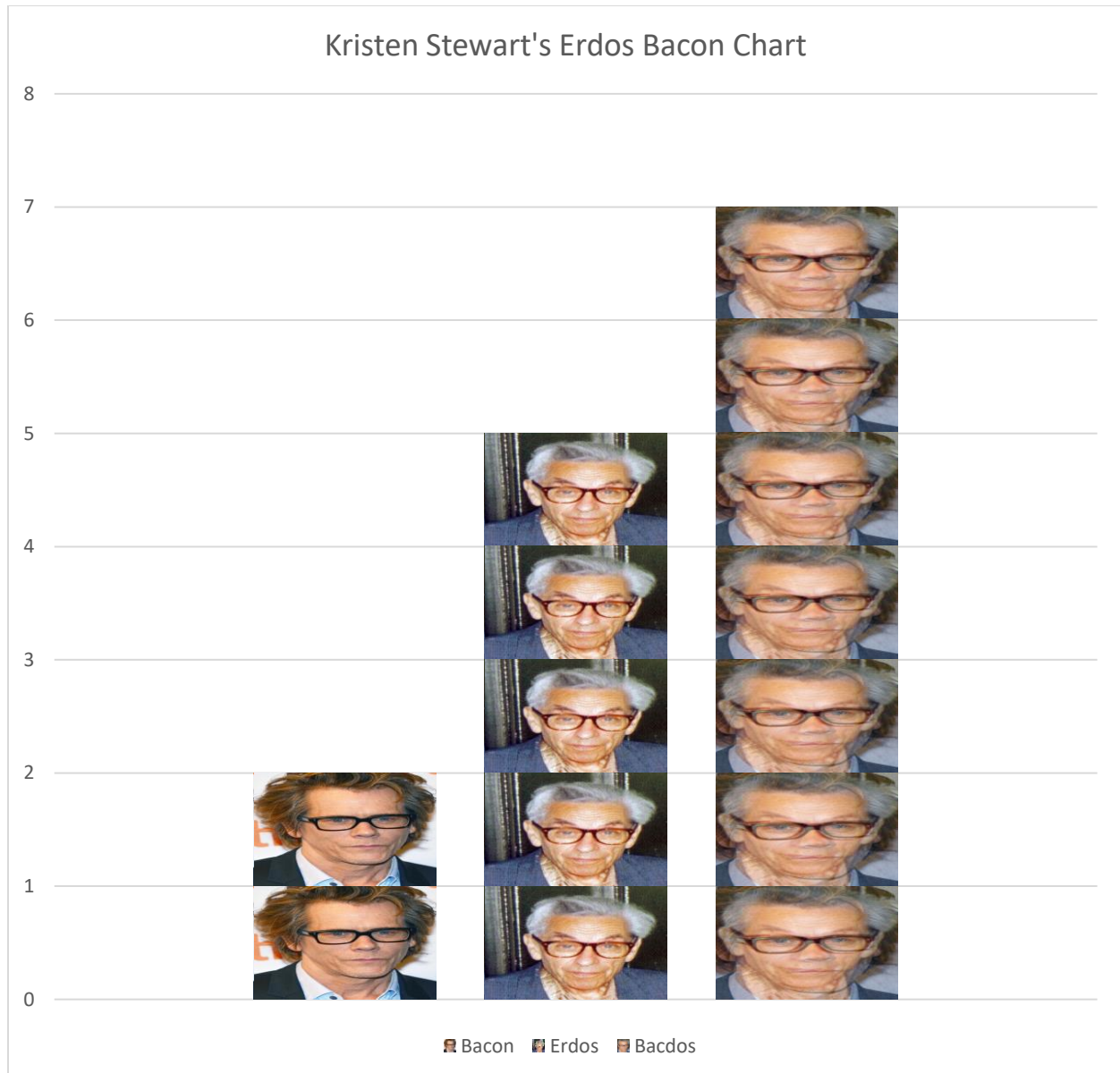


Image references:

<https://www.upi.com/Kevin-Bacon-says-Syfy-passed-on-Tremors-pilot/5361524948843/>
https://en.wikiquote.org/wiki/Paul_Erd%C5%91s

References

- [1] <https://stackoverflow.com/questions/18262293/how-to-open-every-file-in-a-folder>
- [2] https://en.wikipedia.org/wiki/Erd%C5%91s%E2%80%93Bacon_number