# Assignment 3

CS 532: Introduction to Web Science

Spring 2016

Manoj Chandra Kompalli

Finished on February 18,2016

# 1

## Question

```
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.
from the command line:

% curl http://www.cnn.com/ > www.cnn.com

% wget -O www.cnn.com http://www.cnn.com/

% lynx -source http://www.cnn.com/ > www.cnn.com

"www.cnn.com" is just an example output file name, keep in mind
that the shell will not like some of the characters that can occur
in URIs (e.g., "?", "&").  You might want to hash the URIs, like:

% echo -n "http://www.cs.odu.edu/show_features.shtml?72" | md5
41d5f125d13b4bb554e6e31b6b591eeb

("md5sum" on some machines; note the "-n" in echo -- this removes
the trailing newline.)

Now use a tool to remove (most) of the HTML markup.  "lynx" will
do a fair job:

% lynx -dump -force_html www.cnn.com > www.cnn.com.processed
Use another (better) tool if you know of one.  Keep both files
for each URI (i.e., raw HTML and processed).
```

## Answer

I had extracted all URIs from the second assignment in the same JSON format. I have then used the command

```
curl -s -L "URI" >./rawurls/filename
```

to generate the raw html content into a directory. I had also used the command below to generate processed URIs

```
lynx -dump -force_html "URI" >./processedurls/ filename
```

I used the os library to execute shell commands from the python file. I knew that I had to unique generate all 2000 URIs combined which could have either been done by using the md5 hashing technique you have mentioned in the question . Just to improve the readability of the file names I decided to use a counter and count to 1000 and append that to a string each time for all 1000 URIs. I have used this approach for both the raw URIs and processed URIs.

## Code Listing

```
import re
import os
import json

if __name__=="__main__":
        f2name='raw.txt'
        f3name='processed.txt'
        count=0
        count1=0
        file1=open('links.json','r')# file which contains 1000
            uris

        for line in file1.readlines():
                count=count+1
                newfile=str(count)+f2name #concatenates counter
                    value to a string
                one_line = json.loads(line)
                link = one_line['link']

                cmd="curl -s -L "+ link+" >./rawurls/"+ newfile
                    # shell script to print raw html content of
                    each uri
                os.system(cmd)
        for line in file1.readlines():
                count1=count1+1
                newfile1=str(count1)+f3name #concatenates
                    counter value to a string
                one_line1 = json.loads(line)
                link1 = one_line1['link']

                cmd1="lynx -dump -force_html "+ link1+" >./
                    processedurls/"+ newfile1 # shell script to
```

```
    print processed html content of each uri
os.system(cmd1)
```

```
<!--[if IE 8]>         <html class="no-js lt-ie9"> <![endif]-->
<!--[if gt IE 8]><!--> <html class="no-js"> <!--<![endif]-->
    <head>
                <meta charset="utf-8" />
        <meta http-equiv="X-UA-Compatible" content="IE=edge" /><script type="text/javascript">
        <meta name="viewport" content="width=device-width, maximum-scale=1.0, target-densityDp

        <script src="//cdn.optimizely.com/js/76980741.js"></script>

                <script>var _sf_startpt=(new Date()).getTime()</script>
        <title>Watch Johnny Depp Star in Funny or Die's Donald Trump Biopic | Rolling Stone</t
        <link rel="stylesheet" href="/assets/css/main.css">
                <script src="/assets/lib/modernizr-2.6.2.min.js"></script>

        <script src="http://c.amazon-adsystem.com/aax2/amzn_ads.js"></script>
<script>
    try {
        amznads.getAds('3050');
    } catch(e) { /*ignore*/}
</script>

<script type="text/javascript">
    var googletag = googletag || {};
    googletag.cmd = googletag.cmd || [];
    (function() {
        var gads = document.createElement("script");
        gads.async = true;
        gads.type = "text/javascript";
        var useSSL = "https:" == document.location.protocol;
        gads.src = (useSSL ? "https:" : "http:") + "//www.googletagservices.com/tag/js/gpt.js"
        var node =document.getElementsByTagName("script")[0];
        node.parentNode.insertBefore(gads, node);
    })();
</script>

<script>
try {
    amznads.setTargetingForGPTAsync('amznslots');
} catch(e) { /*ignore*/}
</script>
                                <meta name="description" content="Funny or Die have turned
                                    <meta name="news_keywords" content="Johnny Depp, donal
```

Figure 1: Images showing the raw html content generated for a URI

```
 #[1]publisher

IFRAME: [2]//www.googletagmanager.com/ns.html?id=GTM-6PGSH

[3]Rolling Stone
  *
  * [4]Follow @RollingStone
  *
  *

  * [5]Subscribe
  * [6]Coverwall

[7]Rolling Stone
  * [8]music
      +
          o Latest Music
            Baauer
          Baauer Is Shaking Off the Blessing and Curse of Meme Stardom
            By Andy Beta
          o [9]

Eagles of Death Metal Bring Rock, Healing at Triumphant Paris Return
            13 hours ago [10]

Grammys 2016: King Kendrick Lamar Steals the Show
            1 day ago [11]

Taylor Swift vs. Kanye West: A Beef History
            1 day ago [12]More Music News »
          o Interviews
            [13]Jenny Lewis [14]

Jenny Lewis on 'Rabbit Fur Coat' at 10, How Conor Oberst Changed Her Life
            "I've been rehearsing these songs, and they...
            [15]More Interviews »
          o Reviews
            Kanye West; The Life of Pablo; Album Review

Kanye West

The Life of Pablo
            Wiz Khalifa

Wiz Khalifa
```

Figure 2: Response without html tags,stylesheets etc

# 2

## Question

```
2.  Choose a query term (e.g., "shadow") that is not a stop word
(see week 5 slides) and not HTML markup from step 1 (e.g., "http")
that matches at least 10 documents (hint: use "grep" on the processed
files).  If the term is present in more than 10 documents, choose
any 10 from your list.  (If you do not end up with a list of 10
URIs, you've done something wrong).

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.  For
example:

Table 1. 10 Hits for the term "shadow", ranked by TFIDF.

TFIDF TF IDF URI
-------------
0.150 0.014 10.680 http://foo.com/
0.044 0.008  5.510 http://bar.com/


You can use Google or Bing for the DF estimation.  To count the
number of words in the processed document (i.e., the deonminator
for TF), you can use "wc":

% wc -w www.cnn.com.processed
    2370 www.cnn.com.processed

It won't be completely accurate, but it will be probably be
consistently inaccurate across all files.  You can use more
accurate methods if you'd like.

Don't forget the log base 2 for IDF, and mind your significant
digits!
```

## Answer

I had started off by trying to use grep command from the shell. I had figured out that grep could be used to find files with a keyword. I used keyword crime on the processed files.I used the following command to get all similar words matching keyword crime.

```
grep -lr "crime" directoryname
```

I got a few matching files from where I have randomly selected ten URIs.

```
'grep -c ' +'crime ' filename.
```

I used the command wc w to get the list of all words from a URI. I had then taken a ratio for the occurred words to total words which gave me TF.

For IDF I used the search results from Bing. Bing has a corpus value of 17 billion and queried word crime has 18 million search results. The ratio of the logarithm

Log(corpus value/doc term) gives the IDF. The product of TF,IDF gives TFIDF. I had measured the values for TFIDF .My next task was to arrange the tfidf values in descending order. More TF-IDF value indicates more occurrence in that URI or less count of total words.

## Code Listing

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import os
import commands
import math
import re
import json
import string
file=open('ten_files.txt','r')
file1=open('links.json','r').readlines()
count=0
count1=0
total=17000000000#corpus value for bing approx. 17 billion
doc_term=18000000#bing results for the queried word "crime""
idf_big=total/doc_term
print 'TF        IDF         TFIDF           URIS'
for line in file.readlines():
        count=count+1
        all=string.maketrans('','')
        nodigs=all.translate(all, string.digits)
```

```
found=line.translate(all, nodigs)
foundint=int(found)


occurence='grep -c ' +'crime '+ line.strip()
words='wc -w '+line.strip()

occur_doc=commands.getoutput(occurence)
words_doc=commands.getoutput(words)
words_total= words_doc.split(' ')[0]
tf=round(float(occur_doc)/float(words_total),5)

idf =round(math.log(idf_big)/math.log(2),5)
tfidf=round(tf*idf,5)



b=line.rstrip("processed.txt")

for line1 in file1:
        try:

                count1=count1+1

                one_line1 = json.loads(line1)
                link1 = one_line1['link']

                if(count1==foundint):



                        print tf,'  ',idf,'  ',tfidf,'  
                            ',link1

                        count1=0
                        break
                else:
                        continue


        except:
                pass
```

## Selected random ten files matching keyword

```
413processed.txt
302processed.txt
95processed.txt
110processed.txt
135processed.txt
147processed.txt
182processed.txt
183processed.txt
208processed.txt
284processed.txt
```

## TF,IDF,TF-IDF

| TF | IDF | TFIDF | URIS |
|---|---|---|---|
| 0.00295 | 9.88264 | 0.02915 | http://www.tvguide.com/news/exclusive-criminal-minds-sneak-peek-jj-theory/?ftag=twtrsoshares |
| 0.00227 | 9.88264 | 0.02243 | http://www.mirror.co.uk/news/uk-news/julian-assanges-alleged-rape-victim-7318264#ICID=sharebar_twitter |
| 0.00213 | 9.88264 | 0.02105 | http://www.occuworld.org/news/2997720 |
| 0.00186 | 9.88264 | 0.01838 | http://news.thaipbs.**or**.th/content/250076 |
| 0.00185 | 9.88264 | 0.01828 | http://www.cp24.com/news/officer-confronts-robbery-suspects-as-they-exit-gas-station-**in**-vaughan-1.2772521 |
| 0.00164 | 9.88264 | 0.01621 | http://www.liverpoolecho.co.uk/news/liverpool-news/blind-liverpool-city-centre-busker-10872530 |
| 0.00141 | 9.88264 | 0.01393 | http://www.cbc.ca/news/canada/thunder-bay/first-nations-ptsd-thunder-bay-1.3441787?**cmp**=rss |
| 0.00134 | 9.88264 | 0.01324 | http://www.heraldsun.com.au/news/victoria/letthemstay-protesters-on-eastern-freeway/news-story/3c97d64decd630493a0707c07afc192f |
| 0.00081 | 9.88264 | 0.008 | http://www.abc.net.au/news/2016-02-11/same-sex-parents-sa-win-right-**for**-both-on-birth-certificate/7157912 |
| 0.00075 | 9.88264 | 0.00741 | http://www.fox23.com/news/acting-tulsa-sheriff-proposes-idea-to-make-money-**from**-inmate-cell-phone-calls/47960035 |

## 3

### Question

3. Now rank the same 10 URIs from question #2, but this time by their PageRank. Use any of the free PR estimaters on the web, such as:

http://www.prchecker.info/check_page_rank.php
http://www.seocentro.com/tools/search-engines/pagerank.html
http://www.checkpagerank.net/

If you use these tools, you'll have to do so by hand (they have anti-bot captchas), but there is only 10. Normalize the values they give you to be from 0 to 1.0. Use the same tool on all 10 (again, consistency is more important than accuracy).

Create a table similar to Table 1:

Table 2. 10 hits for the term "shadow", ranked by PageRank.

| PageRank | URI |
|---|---|
| 0.9 | http://bar.com/ |
| 0.5 | http://foo.com/ |

Briefly compare and contrast the rankings produced in questions 2 and 3.

### Answer

I had used one of the web service http://www.seocentro.com/tools/search-engines/pagerank.html on all the ten URIs I was using for my previous problem.

I had generated the page ranks for each URI and then sorted the URIs based on the page rank.

We can compare the page rank with the TF-IDF value even though it makes very little sense. Higher TF-IDF values had good page rank on an average. Higher page rank shows more traffic, keyword density, page authority etc.I had used the shortened URIs because the complete URIs did not have less or no page rank at all.

### Comparing Page Rank,TF-IDF of URIs

| TF–IDF | URIs | PAGE RANK |
|---|---|---|
| 0.008 | http://www.abc.net.au | 0.8 |
| 0.01393 | http://www.cbc.ca | 0.8 |
| 0.02915 | http://www.tvguide.com | 0.7 |
| 0.02243 | http://www.mirror.co.uk | 0.7 |
| 0.01828 | http://www.cp24.com | 0.7 |
| 0.01324 | http://www.heraldsun.com.au | 0.7 |
| 0.01621 | http://www.liverpoolecho.co.uk | 0.5 |
| 0.02105 | http://www.occuworld.org | 0.3 |
| 0.01838 | http://news.thaipbs.**or**.th | 0.3 |
| 0.00741 | http://www.fox23.com/news | 0.2 |

# 4

## Question

```
4.  Compute the Kendall Tau_b score for both lists (use "b" because
there will likely be tie values in the rankings).  Report both the
Tau value and the "p" value.

See:
http://stackoverflow.com/questions/2557863/measures-of-association-in-r-kendalls-tau-
http://en.wikipedia.org/wiki/Kendall_tau_rank_correlation_coefficient#Tau-b
http://en.wikipedia.org/wiki/Correlation_and_dependence
```

## Answer

Kendall's Tau gives the relation between TF-IDF and page rank. Tau Value closer to 1 denotes high correleation. Tau value 0 denotes no correlation .My TF-IDF and page rank vectors have given me a tau value of 0.906747. z=3.4603 ,p-value of 0.0002698.My results show that there is a lot of correlation between TF-IDF and Page Rank vectors.

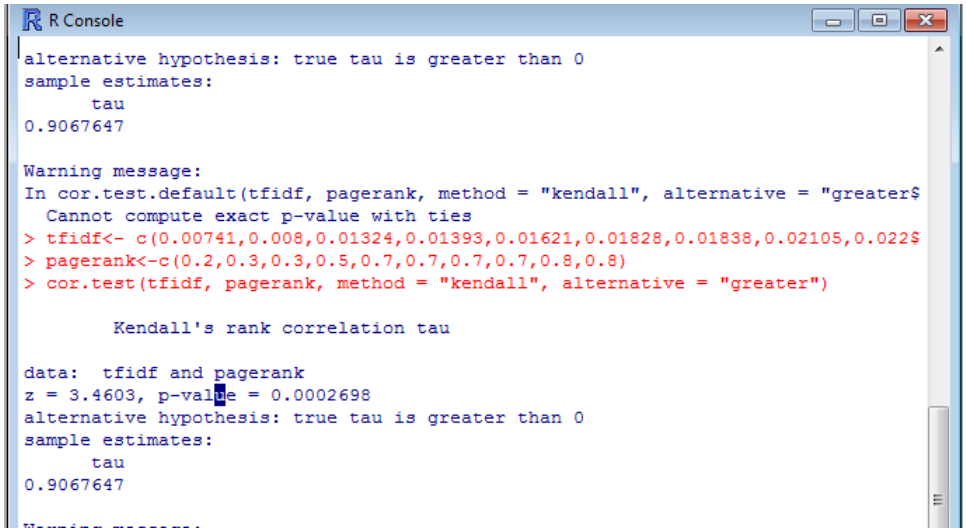


Figure 3: The output of the R console which gives Tau value

## Program to find Tau value

```
tfidf<- c
    (0.00741,0.008,0.01324,0.01393,0.01621,0.01828,0.01838,0.02105,0.02243,0.02915)

pagerank<-c(0.2,0.3,0.3,0.5,0.7,0.7,0.7,0.7,0.8,0.8)
cor.test(tfidf, pagerank, method = "kendall", alternative = "
    greater")
```

## References


[1] Tutorial to run Shell Commands in Python:. http://unix.stackexchange.com/questions/238180/execute-shell-commands-in-python/.

[2] Tutorial for Grep Command. https://en.wikipedia.org/wiki/Grepl:.

[3] Using Grep command to retrieve find files matching keyword :. http://stackoverflow.com/questions/4121803/how-can-i-use-grep-to-find-a-word-inside-a-folder.

[4] To find Corpus size of Bing:. http://www.worldwidewebsize.com/l.

[5] Web Service to find the page rank of a URI:. http://www.seocentro.com/tools/search-engines/pagerank.html.

[6] Seperating characters and digits from a string:. http://stackoverflow.com/questions/1450897/python-removing-characters-except-digits-from-stringl.