Data-X Fall 2018: Homework 8

Webscraping

Authors: Alexander Fred-Ojala

In this homework, you will do some exercises with web-scraping.

STUDENT NAME : Shubei Wang

SID : 3034358656

Fun with Webscraping & Text manipulation

1. Statistics in Presidential Debates

Your first task is to scrape Presidential Debates from the Commission of Presidential Debates website: http://www.debates.org/index.php?page=debate-transcripts (http://www.debates.org/index.php?page=debate-transcripts).

To do this, you are not allowed to manually look up the URLs that you need, instead you have to scrape them. The root url to be scraped is the one listed above, namely: http://www.debates.org/index.php?page=debate-transcripts (http://www.debates.org/index.php?page=debate-transcripts)

1. By using `requests` and `BeautifulSoup` find all the links / URLs on the website that links to transcriptions of **First Presidential Debates** from the years [2012, 2008, 2004, 2000, 1996, 1988, 1984, 1976, 1960]. In total you should find 9 links / URLs tat fulfill this criteria. Print the urls.
2. When you have a list of the URLs your task is to create a Data Frame with some statistics (see example of output below):
   A. Scrape the title of each link and use that as the column name in your Data Frame.
   B. Count how long the transcript of the debate is (as in the number of characters in transcription string). Feel free to include `\` characters in your count, but remove any breakline characters, i.e. `\n` . You will get credit if your count is +/- 10% from our result.
   C. Count how many times the word **war** was used in the different debates. Note that you have to convert the text in a smart way (to not count the word **warranty** for example, but counting **war.**, **war!**, **war,** or **War** etc.

D. Also scrape the most common used word in the debate, and write how many times it was used. Note that you have to use the same strategy as in 3 in order to do this.

   Print your final output result.

Tips:

In order to solve the questions above, it can be useful to work with Regular Expressions and explore methods on strings like `.strip()`, `.replace()`, `.find()`, `.count()`, `.lower()` etc. Both are very powerful tools to do string processing in Python. To count common words for example I used a `Counter` object and a Regular expression pattern for only words, see example:

```
from collections import Counter
import re

counts = Counter(re.findall(r"[\w']+", text.lower()))
```

Read more about Regular Expressions here: https://docs.python.org/3/howto/regex.html (https://docs.python.org/3/howto/regex.html)

Example output of all of the answers to Question 1.2:

	October 3, 2012: The First Obama-Romney Presidential Debate	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Debate char length	94627	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
war_count	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
most_common_w	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
most_common_w_count	757	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

.

In [1]:
```
import requests
import bs4 as bs
from collections import Counter
import re
import pandas as pd
```

In [2]:
```
source = requests.get("http://www.debates.org/index.php?page=debate-transcripts")
soup = bs.BeautifulSoup(source.content, features='html.parser')
```

In [3]:
```
links = soup.find_all('a')
first_debates = list()
urls = list()
ex = re.compile('.*First.*Presidential.*Debate.*', re.IGNORECASE)
for l in links:
    text = l.text
    if(ex.match(text) != None):
        first_debates.append(text)
        urls.append(l.get('href'))
        print("Info about {}: ".format(l.text), \
          l.get('href'))
```

```
Info about October 3, 2012: The First Obama-Romney Presidential Debate:  http://www.debates.org/index.php?page=october-3-2012-debate-transcript (http://www.debates.org/index.php?page=october-3-2012-debate-transcript)
Info about September 26, 2008: The First McCain-Obama Presidential Debate:  http://www.debates.org/index.php?page=2008-debate-transcript (http://www.debates.org/index.php?page=2008-debate-transcript)
Info about September 30, 2004: The First Bush-Kerry Presidential Debate:  http://www.debates.org/index.php?page=september-30-2004-debate-transcript (http://www.debates.org/index.php?page=september-30-2004-debate-transcript)
Info about October 3, 2000: The First Gore-Bush Presidential Debate:  http://www.debates.org/index.php?page=october-3-2000-transcript (http://www.debates.org/index.php?page=october-3-2000-transcript)
Info about October 6, 1996: The First Clinton-Dole Presidential Debate:  http://www.debates.org/index.php?page=october-6-1996-debate-transcript (http://www.debates.org/index.php?page=october-6-1996-debate-transcript)
Info about September 25, 1988: The First Bush-Dukakis Presidential Debate:  http://www.debates.org/index.php?page=september-25-1988-debate-transcript (http://www.debates.org/index.php?page=september-25-1988-debate-transcript)
Info about October 7, 1984: The First Reagan-Mondale Presidential Debate:  http://www.debates.org/index.php?page=october-7-1984-debate-transcript (http://www.debates.org/index.php?page=october-7-1984-debate-transcript)
Info about September 23, 1976: The First Carter-Ford Presidential Debate:  http://www.debates.org/index.php?page=september-23-1976-debate-transcript (http://www.debates.org/index.php?page=september-23-1976-debate-transcript)
Info about September 26, 1960: The First Kennedy-Nixon Presidential Debate:  http://www.debates.org/index.php?page=september-26-1960-debate-transcript (http://www.debates.org/index.php?page=september-26-1960-debate-transcript)
```

In [4]:
```
df = pd.DataFrame(index = ['Debate char length', 'war_count', 'most_common_w', \
                           'most_common_w_count'], columns = first_debates)
for i in range(9):
        url = urls[i]
        srs = requests.get(url)
        sp = bs.BeautifulSoup(srs.content, features = 'html.parser')
        text = sp.find('p').text
        char_len = len(text.replace("\n",""))
        war = re.compile('[^a-zA-Z]war[^a-zA-Z]',re.IGNORECASE)
        count_war = len(war.findall(text))
        c = Counter(re.findall(r"[\w']+", text.lower()))
        most_common_w = c.most_common(1)[0][0]
        most_common_w_count = c.most_common(1)[0][1]
        df.iloc[:,i] = [char_len, count_war, most_common_w, most_common_w_count]
        c.clear()
df
```

Out[4]:

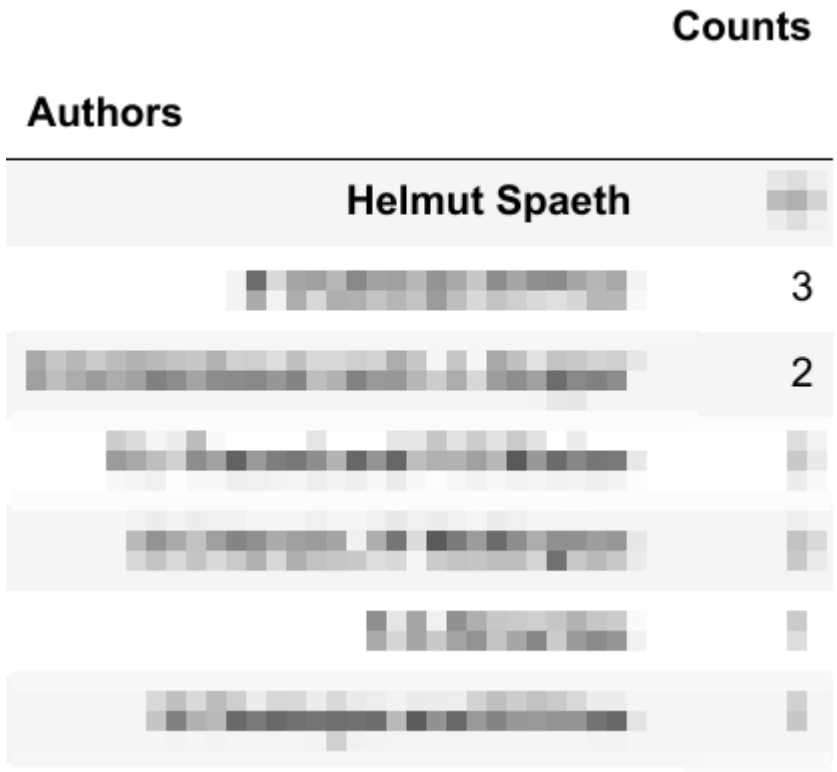
	October 3, 2012: The First Obama-Romney Presidential Debate	September 26, 2008: The First McCain-Obama Presidential Debate	September 30, 2004: The First Bush-Kerry Presidential Debate	October 3, 2000: The First Gore-Bush Presidential Debate	October 6, 1996: The First Clinton-Dole Presidential Debate	September 25, 1988: The First Bush-Dukakis Presidential Debate	October 7, 1984: The First Reagan-Mondale Presidential Debate	September 23, 1976: The First Carter-Ford Presidential Debate	September 26, 1960: The First Kennedy-Nixon Presidential Debate
Debate char length	94594	182386	82685	91040	93057	87458	86654	80701	60901
war_count	3	44	64	11	14	8	2	7	3
most_common_w	the	the	the	the	the	the	the	the	the
most_common_w_count	757	1470	857	919	876	804	867	857	779

2. Download and read in specific line from many data sets

Scrape the first 27 data sets from this URL http://people.sc.fsu.edu/~jburkardt/datasets/regression/ (http://people.sc.fsu.edu/~jburkardt/datasets/regression/) (i.e. `x01.txt - x27.txt`). Then, save the 5th line in each data set, this should be the name of the data set author (get rid of the `#` symbol, the white spaces and the comma at the end).

Count how many times (with a Python function) each author is the reference for one of the 27 data sets. Showcase your results, sorted, with the most common author name first and how many times he appeared in data sets. Use a Pandas DataFrame to show your results, see example. Print your final output result.

Example output of the answer for Question 2:



Authors	Counts
Helmut Spaeth	
[illegible]	3
[illegible]	2
[illegible]	
[illegible]	
[illegible]	
[illegible]	

In [9]:
```
author = list()
base_url =  "http://people.sc.fsu.edu/~jburkardt/datasets/regression/"
source = requests.get(base_url).content
soup = bs.BeautifulSoup(source,'html.parser')
for i in soup.find_all('a'):
    file_name = i.get('href')
    if '28.txt' in file_name:
        break
    if 'txt' in file_name:
        print("saving information from "+file_name+'...')
        file = requests.get(base_url+file_name).text
        author.append(re.search(r"Reference:\n#\n# +([a-zA-Z ,]+[a-zA-Z]),\n#",str(file)).group(1))
```

```
saving information from x01.txt...
saving information from x02.txt...
saving information from x03.txt...
saving information from x04.txt...
saving information from x05.txt...
saving information from x06.txt...
saving information from x07.txt...
saving information from x08.txt...
saving information from x09.txt...
saving information from x10.txt...
saving information from x11.txt...
saving information from x12.txt...
saving information from x13.txt...
saving information from x14.txt...
saving information from x15.txt...
saving information from x16.txt...
saving information from x17.txt...
saving information from x18.txt...
saving information from x19.txt...
saving information from x20.txt...
saving information from x21.txt...
saving information from x22.txt...
saving information from x23.txt...
saving information from x24.txt...
saving information from x25.txt...
saving information from x26.txt...
saving information from x27.txt...
```

In [10]:
```
author
```

Out[10]:
```
['Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'R J Freund and P D Minton',
 'D G Kleinbaum and L L Kupper',
 'Helmut Spaeth',
 'D G Kleinbaum and L L Kupper',
 'K A Brownlee',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'S Chatterjee and B Price',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'R J Freund and P D Minton',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'S Chatterjee, B Price',
 'S Chatterjee, B Price',
 'S Chatterjee, B Price',
 'S C Narula, J F Wellington',
 'S C Narula, J F Wellington']
```

In [11]:
```
df = pd.DataFrame(columns=['counts'], index=set(author))
df.index.name = 'authors'
for name in df.index:
    df.loc[name] = author.count(name)
df.sort_values('counts',ascending=0,inplace=True)
df
```

Out[11]:

authors	counts
Helmut Spaeth	16
S Chatterjee, B Price	3
S C Narula, J F Wellington	2
R J Freund and P D Minton	2
D G Kleinbaum and L L Kupper	2
S Chatterjee and B Price	1
K A Brownlee	1