Data-X Fall 2018: Homework 8

Webscraping

Authors: Alexander Fred-Ojala

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

STUDENT NAME : Jeff Peng

SID : 3034340183

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 [4]:
```
from IPython.core.display import display, HTML
display(HTML("<style>.container {width: 90% !important; }</style>"))
from __future__ import division, print_function
```

In [5]:
```
import requests
import bs4 as bs
source=requests.get("http://www.debates.org/index.php?page=debate-transcripts")
```

In [6]:
```
print(source)
```

```
<Response [200]>
```

In [7]:
```
soup=bs.BeautifulSoup(source.content, features='html.parser')
```

In [8]:
```
years=[2012, 2008, 2004, 2000, 1996, 1988, 1984, 1976, 1960]
```

In [9]:
```
urls=[]
for a in soup.find_all('a'):
    if a.text.find('First')!=-1:
        for y in years:
            if a.text.find(str(y))!=-1:
                urls.append(a.get('href'))
print(urls)
```

```
['http://www.debates.org/index.php?page=october-3-2012-debate-transcript', 'http://www.deba
tes.org/index.php?page=2008-debate-transcript', 'http://www.debates.org/index.php?page=sept
ember-30-2004-debate-transcript', 'http://www.debates.org/index.php?page=october-3-2000-tra
nscript', 'http://www.debates.org/index.php?page=october-6-1996-debate-transcript', 'htt
p://www.debates.org/index.php?page=september-25-1988-debate-transcript', 'http://www.debate
s.org/index.php?page=october-7-1984-debate-transcript', 'http://www.debates.org/index.php?p
age=september-23-1976-debate-transcript', 'http://www.debates.org/index.php?page=september-
26-1960-debate-transcript']
```

In [10]:
```
import pandas as pd
column_titles=[]
for u in urls:
    source=requests.get(u)
    soup=bs.BeautifulSoup(source.content, features='html.parser')
    column_titles.append(soup.find('title').text)
df=pd.DataFrame(columns=column_titles, index=['debate_char_length',
                                              'war_count', 'most_common_w',
                                              'most_common_w_count'])
```

In [11]:
```
from collections import Counter
import re
```

In [12]:
```
for i,u in enumerate (urls):
    source=requests.get(u)
    soup=bs.BeautifulSoup(source.content, features='html.parser')
    words=soup.find('div',{'id':'content-sm'})

    debate_char_length=words.text.replace('\n',' ')
    df.iloc[0,i]=len(debate_char_length)

    war_count=re.findall(r"war[s!,.]*\s+", words.text.lower())
    df.iloc[1,i]=len(war_count)

    count=Counter(re.findall(r"[\w']+", words.text.lower()))
    df.iloc[2,i]=count.most_common(1)[0][0]
    df.iloc[3,i]=count.most_common(1)[0][1]
df
```

Out[12]:

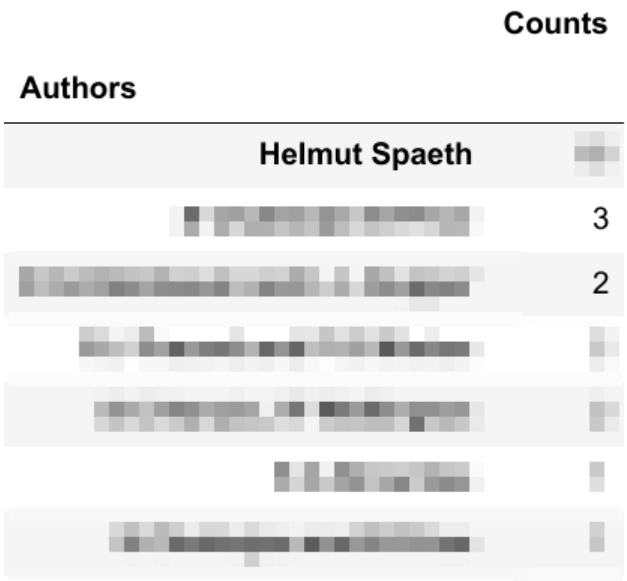
	CPD: October 3, 2012 Debate Transcript	CPD: September 26, 2008 Debate Transcript	CPD: September 30. 2004 Debate Transcript	CPD: October 3, 2000 Transcript	CPD: October 6, 1996 Debate Transcript	CPD: September 25, 1988 Debate Transcript	CPD: October 7, 1984 Debate Transcript	CPD: September 23, 1976 Debate Transcript	CPD: September 26, 1960 Debate Transcript
debate_char_length	95108	182428	82726	91071	93095	87736	87000	80837	61013
war_count	5	44	60	11	15	11	3	7	3
most_common_w	the	the	the	the	the	the	the	the	the
t_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:



In [13]:
```
source=requests.get('http://people.sc.fsu.edu/~jburkardt/datasets/regression/')
soup=bs.BeautifulSoup(source.content, features='html.parser')
urls_2=[]
for a in soup.find_all('a'):
    if a.text.find('x')!=-1:
        urls_2.append(a.get('href'))
urls_2=['http://people.sc.fsu.edu/~jburkardt/datasets/regression/'+ u for u in urls_2[:27]]
urls_2
```

Out[13]:
```
['http://people.sc.fsu.edu/~jburkardt/datasets/regression/x01.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x02.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x03.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x04.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x05.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x06.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x07.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x08.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x09.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x10.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x11.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x12.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x13.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x14.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x15.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x16.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x17.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x18.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x19.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x20.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x21.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x22.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x23.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x24.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x25.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x26.txt',
 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/x27.txt']
```

In [14]:
```
authors=[]
for u in urls_2:
    source=requests.get(u).content
    soup=bs.BeautifulSoup(source,"html.parser")
    author=re.findall(r"^.*\w+\s\w+.*,$",soup.text,flags=re.MULTILINE)
    authors.append(author[0])
authors
```

Out[14]:
```
['#    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 [15]:
```
for i,a in enumerate(authors):
    authors[i]=authors[i].replace('#','')
    authors[i]=authors[i].strip()
    authors[i]=authors[i][:-1]
authors
```

Out[15]:
```
['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 [16]:
```
authors_new=[]
for i,a in enumerate(authors):
    if(a.find('and')!=-1):
        authors_new=authors_new+(a.split(' and '))
    elif(a.find(',')!=-1):
        authors_new=authors_new+(a.split(', '))
    else:
        authors_new.append(a)
authors_new
```

Out[16]:
```
['Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'R J Freund',
 'P D Minton',
 'D G Kleinbaum',
 'L L Kupper',
 'Helmut Spaeth',
 'D G Kleinbaum',
 'L L Kupper',
 'K A Brownlee',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'S Chatterjee',
 'B Price',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'Helmut Spaeth',
 'R J Freund',
 '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 [17]:
```
import numpy as np
df = pd.DataFrame(columns=['Authors','Counts'])
df['Authors']=authors_new
df['Counts']=np.ones(len(authors_new),np.int8)
df
```

Out[17]:

	Authors	Counts
0	Helmut Spaeth	1
1	Helmut Spaeth	1
2	Helmut Spaeth	1
3	Helmut Spaeth	1
4	Helmut Spaeth	1
5	R J Freund	1
6	P D Minton	1
7	D G Kleinbaum	1
8	L L Kupper	1
9	Helmut Spaeth	1
10	D G Kleinbaum	1
11	L L Kupper	1
12	K A Brownlee	1
13	Helmut Spaeth	1
14	Helmut Spaeth	1
15	S Chatterjee	1
16	B Price	1
17	Helmut Spaeth	1
18	Helmut Spaeth	1
19	Helmut Spaeth	1
20	Helmut Spaeth	1
21	Helmut Spaeth	1
22	R J Freund	1
23	P D Minton	1
24	Helmut Spaeth	1
25	Helmut Spaeth	1
26	Helmut Spaeth	1
27	S Chatterjee	1
28	B Price	1
29	S Chatterjee	1
30	B Price	1
31	S Chatterjee	1
32	B Price	1
33	S C Narula	1
34	J F Wellington	1
35	S C Narula	1
36	J F Wellington	1

In [18]:
```
df=df.groupby(['Authors']).sum().sort_values('Counts',ascending=False)
df
```

Out[18]:

Authors	Counts
Helmut Spaeth	16
B Price	4
S Chatterjee	4
D G Kleinbaum	2
J F Wellington	2
L L Kupper	2
P D Minton	2
R J Freund	2
S C Narula	2
K A Brownlee	1

In [ ]:

In [ ]: