Data-X Spring 2019: Homework 7

Webscraping

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

Name: Miran Tafazzul Hussain Junaidi

SID: 3034487132

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: https://www.debates.org/voter-education/debate-transcripts/ (https://www.debates.org/voter-education/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: https://www.debates.org/voter-education/debate-transcripts/ (https://www.debates.org/voter-education/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 [1988, 1984, 1976, 1960]. In total you should find 4 links / URLs that 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 C 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:

```python
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:

	September 25, 1988: The First Bush-Dukakis Presidential Debate	[redacted]	[redacted]	[redacted]
Debate char length	87488	[redacted]	[redacted]	[redacted]
war_count	[redacted]	[redacted]	[redacted]	[redacted]
most_common_w	[redacted]	[redacted]	[redacted]	[redacted]
most_common_w_count	[redacted]	[redacted]	[redacted]	[redacted]

.

In [1]:

```python
# your code here
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:90% !important; }</style>"))
```

```
...
```

In [92]:

```python
import numpy as np
import pandas as pd
import re
from collections import Counter
```

In [2]:

```python
import requests
import bs4 as bs
```

In [3]:

```python
source = requests.get("https://www.debates.org/voter-education/debate-trans
```

In [6]:

```python
content = bs.BeautifulSoup(source.content , features='html.parser')
```

In [7]:

```
print(content)
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//E
N">

<html>
<head>
<meta content="en-us" http-equiv="Content-Language"/>
<meta content="text/html;charset=utf-8" http-equiv="Content-Ty
pe"/>
<title>CPD: Debate Transcripts</title>
<link href="/wp-content/themes/debates2019/css/reset.css" rel
="stylesheet" type="text/css"/>
<link href="/wp-content/themes/debates2019/css/jc-main.css" me
dia="screen,projection" rel="stylesheet" type="text/css"/>
<link href="/wp-content/themes/debates2019/css/fonts.css" medi
a="screen,projection" rel="stylesheet" type="text/css"/>
<!--[if gte IE 5]>
        <link href="/wp-content/themes/debates2019/css/jc-iema
in.css" rel="stylesheet" type="text/css" media="screen,project
ion"  />
```

In [35]:

```
all_links = content.find_all('a')
all_first_debate = []
for i in all_links:
    bodies = i.text
    if 'First' in bodies and 'Presidential' in bodies and 'Debate' in bodie
        all_first_debate.append(i)
all_first_debate
```

Out[35]:

```
[<a href="/voter-education/debate-transcripts/september-26-201
6-debate-transcript/">September 26, 2016: The First Clinton-Tr
ump Presidential Debate</a>,
 <a href="/voter-education/debate-transcripts/october-3-2012-d
ebate-transcript/" title="October 3, 2012 Debate Transcript">O
ctober 3, 2012: The First Obama-Romney Presidential Debate</a
>,
 <a href="/voter-education/debate-transcripts/2008-debate-tran
script/" title="September 26, 2008 Debate Transcript">Septembe
r 26, 2008: The First McCain-Obama Presidential Debate</a>,
 <a href="/voter-education/debate-transcripts/september-30-200
4-debate-transcript/" title="September 30. 2004 Debate Transcr
ipt">September 30, 2004: The First Bush-Kerry Presidential Deb
ate</a>,
 <a href="/voter-education/debate-transcripts/october-3-2000-t
ranscript/" title="October 3, 2000 Transcript">October 3, 200
0: The First Gore-Bush Presidential Debate</a>,
 <a href="/voter-education/debate-transcripts/october-6-1996-d
ebate-transcript/" title="October 6, 1996 Debate Transcript">O
ctober 6, 1996: The First Clinton-Dole Presidential Debate</a
>,
 <a href="/voter-education/debate-transcripts/september-25-198
8-debate-transcript/" title="September 25, 1988 Debate Transcr
ipt">September 25, 1988: The First Bush-Dukakis Presidential D
ebate</a>,
 <a href="/voter-education/debate-transcripts/october-7-1984-d
ebate-transcript/" title="October 7, 1984 Debate Transcript">O
ctober 7, 1984: The First Reagan-Mondale Presidential Debate</
a>,
 <a href="/voter-education/debate-transcripts/september-23-197
6-debate-transcript/" title="September 23, 1976 Debate Transcr
ipt">September 23, 1976: The First Carter-Ford Presidential De
bate</a>,
 <a href="/voter-education/debate-transcripts/september-26-196
0-debate-transcript/" title="September 26, 1960 Debate Transcr
ipt">September 26, 1960: The First Kennedy-Nixon Presidential
Debate</a>]
```

In [42]:

```
final_links = []
for i in all_first_debate:
    bodies = i.text
    if '1988' in bodies or '1984' in bodies or  '1976' in bodies or  '1960'
        final_links.append(i)
final_links
```

Out[42]:

```
[<a href="/voter-education/debate-transcripts/september-25-198
8-debate-transcript/" title="September 25, 1988 Debate Transcr
ipt">September 25, 1988: The First Bush-Dukakis Presidential D
ebate</a>,
 <a href="/voter-education/debate-transcripts/october-7-1984-d
ebate-transcript/" title="October 7, 1984 Debate Transcript">O
ctober 7, 1984: The First Reagan-Mondale Presidential Debate</
a>,
 <a href="/voter-education/debate-transcripts/september-23-197
6-debate-transcript/" title="September 23, 1976 Debate Transcr
ipt">September 23, 1976: The First Carter-Ford Presidential De
bate</a>,
 <a href="/voter-education/debate-transcripts/september-26-196
0-debate-transcript/" title="September 26, 1960 Debate Transcr
ipt">September 26, 1960: The First Kennedy-Nixon Presidential
Debate</a>]
```

In [43]:

```
for i in  range(0,len(final_links)):
    final_links[i] = "https://www.debates.org" + final_links[i].get('href')
final_links
```

Out[43]:

```
['https://www.debates.org/voter-education/debate-transcripts/s
eptember-25-1988-debate-transcript/',
 'https://www.debates.org/voter-education/debate-transcripts/o
ctober-7-1984-debate-transcript/',
 'https://www.debates.org/voter-education/debate-transcripts/s
eptember-23-1976-debate-transcript/',
 'https://www.debates.org/voter-education/debate-transcripts/s
eptember-26-1960-debate-transcript/']
```

In [72]:

```
titles = []
debate_content = []
for i in final_links:
    debate_source = requests.get(i)
    debate_content.append(bs.BeautifulSoup(debate_source.content,features='
for i in debate_content:
    titles.append(i.title.text)
titles
```

Out[72]:

```
['CPD: September 25, 1988 Debate Transcript',
 'CPD: October 7, 1984 Debate Transcript',
 'CPD: September 23, 1976 Debate Transcript',
 'CPD: September 26, 1960 Debate Transcript']
```

In [151]:

```
dataframe = pd.DataFrame(columns = titles)
dataframe
```

Out[151]:

CPD: September 25, 1988 Debate Transcript	CPD: October 7, 1984 Debate Transcript	CPD: September 23, 1976 Debate Transcript	CPD: September 26, 1960 Debate Transcript

In [81]:

```
letter_count = []
index = 0
count = 0
for i in debate_content:
    for j in i.find_all('p'):
        count = count + len(j.text) -  i.text.count('/n')
    letter_count.append(count)
    count = 0
letter_count
```

Out[81]:

```
[87531, 86551, 80778, 60980]
```

In [139]:

```
words = []
war_count = []
most_occuring_word = []
count_of_most_occuring_word = []
for i in debate_content:
    word = []
    for j in i.find_all('p'):
        word = word + (re.sub('\n','',re.sub(r'[^\w\s]','',j.text.lower()))
    war_count.append(word.count('war'))
    words.append(word)

        #most_occuring_word.append([word for word, word_count in Counter(wo
```

In [140]:

```
for i in words:
    most_occuring_word.append([item for item in Counter(i).most_common(1)][
```

In [141]:

```
most_occuring_word
```

Out[141]:

```
['the', 'the', 'the', 'the']
```

In [142]:

```
for i in words:
    count_of_most_occuring_word.append([item for item in Counter(i).most_co
```

In [143]:

```
count_of_most_occuring_word
```

Out[143]:

```
[800, 868, 857, 780]
```

In [152]:

```
dataframe.loc["Number of characters"] = letter_count
dataframe.loc['Count of Word War'] = war_count
dataframe.loc["Most occuring Woard"] = most_occuring_word
dataframe.loc["Count of most occuring words"] = count_of_most_occuring_word
```

In [153]:

```
dataframe
```

Out[153]:

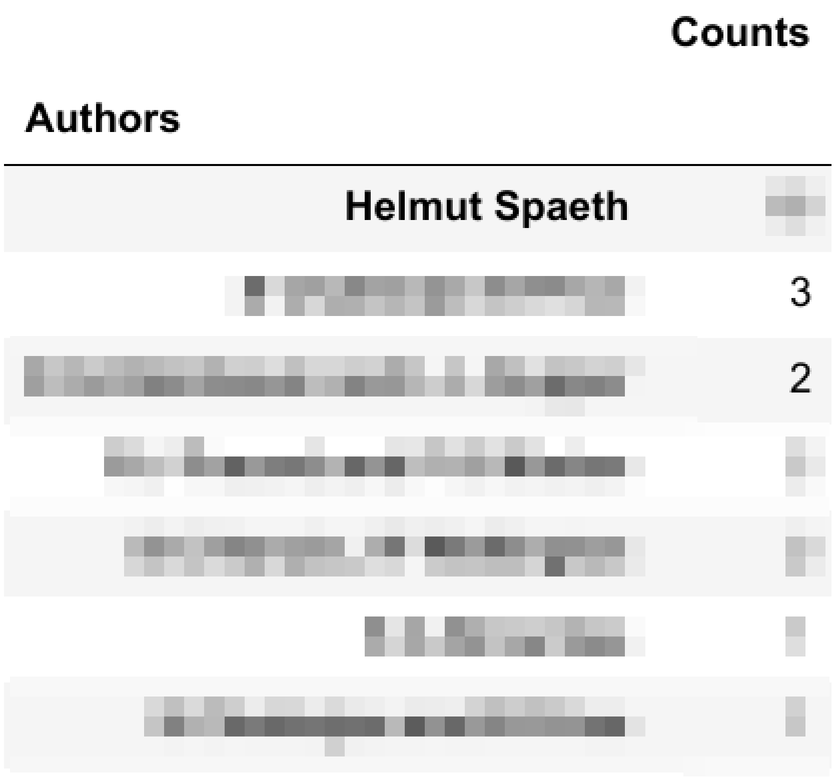
	CPD: September 25, 1988 Debate Transcript	CPD: October 7, 1984 Debate Transcript	CPD: September 23, 1976 Debate Transcript	CPD: September 26, 1960 Debate Transcript
Number of characters	87531	86551	80778	60980
Count of Word War	7	2	7	3
Most occuring Woard	the	the	the	the
Count of most occuring words	800	868	857	780

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:



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

In [166]:

```
# your code here
list_of_authors = []
for i in range(1,28):
    text_file = requests.get("http://people.sc.fsu.edu/~jburkardt/datasets/
    soup = bs.BeautifulSoup(text_file.content , features='html.parser')
    count = 0
    for word in str(soup).split():
        count+=1
        if count == 8:
            list_of_authors.append(word)
list_of_authors
```

...

In [174]:

```
dictio = {}
dictio["Helmut Spaeth"] = list_of_authors.count('Helmut')
dictio["S Chatterjee"] = list_of_authors.count('S')
dictio["R J Freund"] = list_of_authors.count('R')
dictio["D G Kleinbaum"] = list_of_authors.count('D')
dictio["K A Brownlee"] = list_of_authors.count('K')
```

In [175]:

```
dictio
```

Out[175]:

```
{'Helmut Spaeth': 16,
 'S Chatterjee': 6,
 'R J Freund': 2,
 'D G Kleinbaum': 2,
 'K A Brownlee': 1}
```

In [176]:

```
data_writers = pd.DataFrame.from_dict(dictio,orient='index',columns=['Count
```

In [177]:

```
data_writers
```

Out[177]:

	Count
Helmut Spaeth	16
S Chatterjee	6
R J Freund	2
D G Kleinbaum	2
K A Brownlee	1