Data-X Spring 2019: Homework 7

Webscraping

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

Name: Jingtong Zhao

SID: 3034266733

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:

```
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:

.

In [1]:
```
# stretch Jupyter coding blocks to fit screen
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:90% !important; }</style>"))
# if 100% it would fit the screen
```

In [2]:
```
# make it run on py2 and py3
from __future__ import division, print_function

import requests # The requests library is an
# HTTP library for getting and posting content etc.

import bs4 as bs # BeautifulSoup4 is a Python library
# for pulling data out of HTML and XML code.
# We can query markup languages for specific content
from collections import Counter
import re
import pandas as pd
```

In [3]:
```
source = requests.get("https://www.debates.org/voter-educatio
n/debate-transcripts/")
# a GET request will download the HTML webpage.
print(source) # If <Response [200]> then
# the website has been downloaded succesfully
```

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

In [4]:
```
# Convert source.content to a beautifulsoup object
# beautifulsoup can parse (extract specific information) HTML
code

soup = bs.BeautifulSoup(source.content, features='html.parser
')
# we pass in the source content
# features specifies what type of code we are parsing,
# here 'html.parser' specifies that we want beautiful soup to
parse HTML code
#print(type(soup))
#print(soup) # looks a lot nicer!
```

In [5]:
```
#1.1
links = soup.find_all('a')
pre = 'https://www.debates.org'
a4 = []
for l in links:
    if l.text.count('Presidential Debate'):
        if l.text.count('First'):
            if l.text.count('1988')or l.text.count('1984') or
l.text.count('1976') or l.text.count('1960'):
                a4.append(pre+l.get('href'))
                print("Link for {}: ".format(l.text), \
                      pre+l.get('href'))
```

```
Link for September 25, 1988: The First Bush-Dukakis Presiden
tial Debate:  https://www.debates.org/voter-education/debate
-transcripts/september-25-1988-debate-transcript/
Link for October 7, 1984: The First Reagan-Mondale President
ial Debate:  https://www.debates.org/voter-education/debate-
transcripts/october-7-1984-debate-transcript/
Link for September 23, 1976: The First Carter-Ford President
ial Debate:  https://www.debates.org/voter-education/debate-
transcripts/september-23-1976-debate-transcript/
Link for September 26, 1960: The First Kennedy-Nixon Preside
ntial Debate:  https://www.debates.org/voter-education/debat
e-transcripts/september-26-1960-debate-transcript/
```

In [6]:
```
colnam = []
debate_length = []
count_war = []
most_common = []
most_common_count = []
for i in a4:
    source = requests.get(i)
    soup = bs.BeautifulSoup(source.content, features='html.parser')
#A.title
    colnam.append(soup.title.text+":"+soup.find_all('strong')[1].text)

#B.count characters
    p1 = ''
    for p in soup.find_all('p'): # print all text paragraphs on the webpage
        p1 = p1+p.text
    debate_length.append(len(p1))
#C. count war
    char = re.findall(r"[a-zA-Z\-]+", soup.text.lower())
    character =  " ".join(char)
    count = character.count('war ')+character.count('war,')+character.count('war.')+character.count('war!')+character.count('wars ')+character.count('wars,')+character.count('wars.')+character.count('wars!')
    count_war.append(count)
#D. count the most common words

    most = Counter(char).most_common(1)
    word = most[0][0]
    number = most[0][1]
    most_common.append(word)
    most_common_count.append(number)

print(colnam)
print(debate_length)
print(count_war)
print(most_common)
print(most_common_count)
```

```
['CPD: September 25, 1988 Debate Transcript:The First Bush-D
ukakis Presidential Debate', 'CPD: October 7, 1984 Debate Tr
anscript:The First Reagan-Mondale Presidential Debate', 'CPD
: September 23, 1976 Debate Transcript:The First Carter-Ford
Presidential Debate', 'CPD: September 26, 1960 Debate Transc
ript:The First Kennedy-Nixon Presidential Debate']
[87531, 86551, 80778, 60980]
[11, 3, 7, 3]
['the', 'the', 'the', 'the']
[802, 868, 857, 780]
```

In [7]:
```
df = pd.DataFrame([debate_length,count_war,most_common,most_c
ommon_count],columns=colnam,index=['Debate Char Lenght','war_
count','most_common_w','most_common_w_count'])
display(df)
```

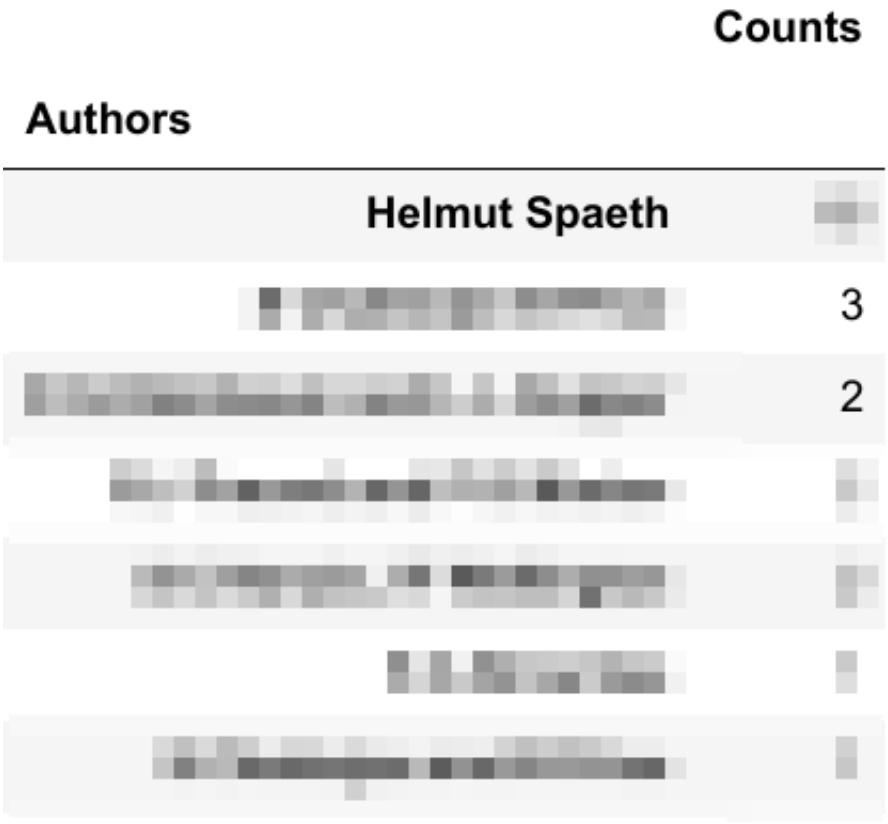
	CPD: September 25, 1988 Debate Transcript:The First Bush-Dukakis Presidential Debate	CPD: October 7, 1984 Debate Transcript:The First Reagan-Mondale Presidential Debate	CPD: September 23, 1976 Debate Transcript:The First Carter-Ford Presidential Debate	CPD Septembe 26, 196 Debat Transcript:Th Firs Kennedy Nixo Presidenti Debat
Debate Char Lenght	87531	86551	80778	6098
war_count	11	3	7	
most_common_w	the	the	the	th
most_common_w_count	802	868	857	78

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 [8]:
```
source = requests.get("http://people.sc.fsu.edu/~jburkardt/datasets/regression/")
soup = bs.BeautifulSoup(source.content, features='html.parser')

links = soup.find_all('a')
url = []
add = 'http://people.sc.fsu.edu/~jburkardt/datasets/regression/'
for l in links:
    if l.text.count('x'):
        url.append(add+l.get('href'))
url = url[0:27]
```

In [9]:
```
fifthline = [];
for i in url:
    source = requests.get(i)
    soup = bs.BeautifulSoup(source.content, features='html.parser')
    for line in soup:
        body = line.split('\n')
        fifthline.append((body[4]).strip("").strip(',').strip('#'))
author = Counter(fifthline).items()
print(author)

author_name = []
author_time = []
for i in author:
    author_name.append([i][0][0])
    author_time.append([i][0][1])
```

```
dict_items([('    Helmut Spaeth', 16), ('    R J Freund and P D Minton', 2), ('    D G Kleinbaum and L L Kupper', 2), ('K A Brownlee', 1), ('    S Chatterjee and B Price', 1), ('S Chatterjee, B Price', 3), ('    S C Narula, J F Wellington', 2)])
```

In [10]:
```
df = pd.DataFrame([author_name,author_time],index=['Authors',
'Counts']).transpose()
df = df.sort_values(by ='Counts',ascending = False)
df = df.set_index('Authors')
display(df)
```

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