Data-X Spring 2019: Homework 7

Webscraping

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

Name: Ziyu Li

SID: 3034331915

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:

	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]:
```
# import libraries
import requests
import re
from bs4 import BeautifulSoup

# make a request
source = requests.get("https://www.debates.org/voter-education/debate-transc

# convert souce.content to a beautifulsoup object
soup = BeautifulSoup(source.content, features = 'html.parser')

# extract links/urls
links = soup.find_all('a')

First_Debate=[]
#links
for url in links:
    if 'First' in url.text and 'Presidential Debate' in url.text:
        if '1988' in url.text or '1984' in url.text or '1976' in url.text or
            First_Debate.append(url.get('href'))

for i in range(len(First_Debate)):
    First_Debate[i]='https://www.debates.org'+ First_Debate[i]
    print(First_Debate[i])
```

```
https://www.debates.org/voter-education/debate-transcripts/september-25-1
988-debate-transcript/ (https://www.debates.org/voter-education/debate-tr
anscripts/september-25-1988-debate-transcript/)
https://www.debates.org/voter-education/debate-transcripts/october-7-1984
-debate-transcript/ (https://www.debates.org/voter-education/debate-trans
cripts/october-7-1984-debate-transcript/)
https://www.debates.org/voter-education/debate-transcripts/september-23-1
976-debate-transcript/ (https://www.debates.org/voter-education/debate-tr
anscripts/september-23-1976-debate-transcript/)
https://www.debates.org/voter-education/debate-transcripts/september-26-1
960-debate-transcript/ (https://www.debates.org/voter-education/debate-tr
anscripts/september-26-1960-debate-transcript/)
```

In [2]:
```
import pandas as pd
from collections import Counter
title=[]
for url in links:
    if 'First' in url.text and 'Presidential Debate' in url.text:
        if '1988' in url.text or '1984' in url.text or '1976' in url.text or
            title.append(url.text)
# create dataframe
df = pd.DataFrame(columns=title,index=['Delete char length','war_count','mos

# b: Count how long the transcript of the debate is
d1 = []
for i in range(0,4):
    source = requests.get(First_Debate[i])
    soup = BeautifulSoup(source.content, features = 'html.parser')
    trans = soup.find('div', id='content-sm').get_text()
    trans = trans.replace('\n', '')
    d1.append(trans)
    df.iloc[0,i]=len(trans)

# c: Count how many times the word war was used in the different debates
for i in range(0,4):
    counts= Counter(re.findall(r'\bwar\b',d1[i].lower()))
    df.iloc[1,i]=counts['war']

# d: scrape the most common used word in the debates and times it used
for i in range(0,4):
    words_num=Counter(re.findall(r'\w+',d1[i].lower()))
    most_common=words_num.most_common(1)
    df.iloc[2,i]=most_common[0][0]
    df.iloc[3,i]=most_common[0][1]

df
```

Out[2]:

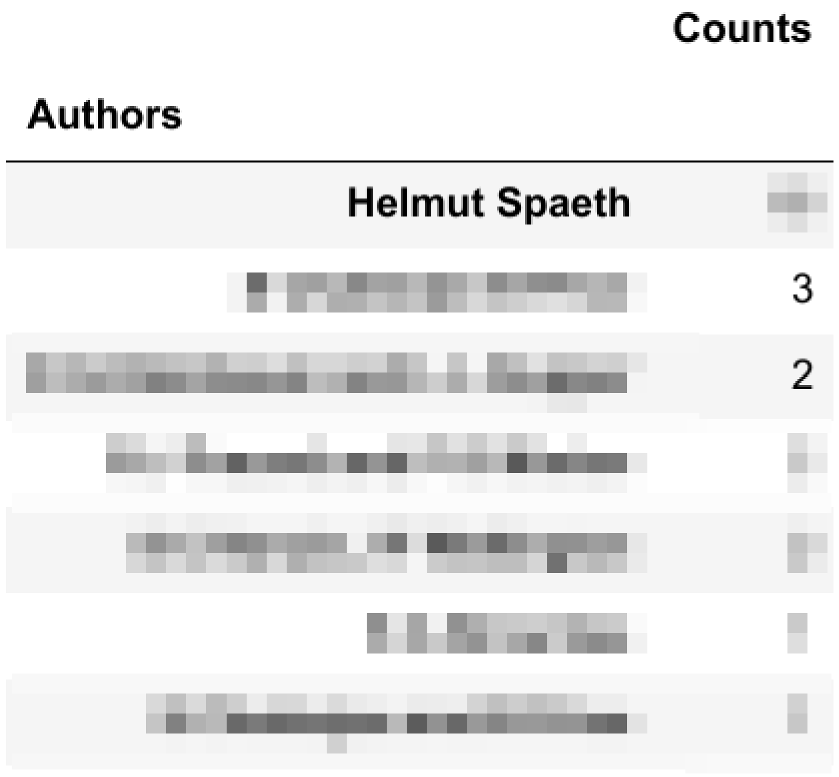
	September 25, 1988: The First Bush-Dukakis Presidential Debate	October 7, 1984: The First Reagan-Mondale Presidential Debate	September 23, 1976: The First Carter-Ford Presidential Debate	September 26, 1960: The First Kennedy-Nixon Presidential Debate
Delete char length	87488	86505	80735	60937
war_count	8	2	7	3
most_common_w	the	the	the	the
most_common_w_count	803	865	856	778

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 [6]:
```
# your code here
source = requests.get("http://people.sc.fsu.edu/~jburkardt/datasets/regressi
soup = BeautifulSoup(source.content, features = 'html.parser')
links = soup.find_all('a')

author=[]
url_0 = "http://people.sc.fsu.edu/~jburkardt/datasets/regression/"
i=0
for url in links:
    if i == 27: break
    elif '.txt' in url.text:
        link = url_0+url.get('href')
        i = i+1
        source2 = requests.get(link)
        soup2 = BeautifulSoup(source2.content)
        text = soup2.get_text()
        line_5 = text.splitlines()[4]
        line_5 = re.sub(r'[^\w\s]','',line_5).strip()
        author.append(line_5)

counts=Counter(author)
Rank=counts.most_common()
df = pd.DataFrame(Rank, columns=['Authors', 'Count'])
df
```

```
/Users/Jade/anaconda3/lib/python3.6/site-packages/bs4/__init__.py:181: Us
erWarning: No parser was explicitly specified, so I'm using the best avai
lable HTML parser for this system ("lxml"). This usually isn't a problem,
but if you run this code on another system, or in a different virtual env
ironment, it may use a different parser and behave differently.

The code that caused this warning is on line 193 of the file /Users/Jade/
anaconda3/lib/python3.6/runpy.py. To get rid of this warning, change code
that looks like this:

 BeautifulSoup(YOUR_MARKUP})

to this:

 BeautifulSoup(YOUR_MARKUP, "lxml")

  markup_type=markup_type))
```

Out[6]:

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

```
In [ ]:
```