Data-X Spring 2019: Homework 7

Webscraping

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

Name: McClain Thiel

SID: 3034003600

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				
most_common_w	[redacted]	[redacted]	[redacted]	[redacted]

In [3]:

```python
# your code here
import requests
import bs4 as bs

base_url = ' https://www.debates.org/voter-education/debate-transcripts/'
source = requests.get(base_url)
soup = bs.BeautifulSoup(source.content, features='html.parser')

temp = [str(x).split('/"')[0] for x in soup.findAll('a') if ('1988' in str(x) or '1984' in str(x) or '1976' in str(x) or '1960' in str(x)) and 'The First' in str(x)]
base = 'https://www.debates.org/'
temp2 = [base + x.split('"/')[1]  for x in temp]
print('The urls for the debates specified above are: ')
for x in temp2:
    print(x, '\n')
```

```
The urls for the debates specified above are:
https://www.debates.org/voter-education/debate-transcripts/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-transcripts/september-23-1976-debate-transcript

https://www.debates.org/voter-education/debate-transcripts/september-26-1960-debate-transcript
```

In [21]:

```python
#part 2
import re
import string
from collections import Counter
import pandas as pd

columns = [bs.BeautifulSoup(requests.get(x).content, features = 'html.parser').find('title').text.replace('CPD:', '') for x in temp2]

arrs = [bs.BeautifulSoup(requests.get(x).content, features = 'html.parser') for x in temp2]


cleaned = []
char_len = []
for x in arrs:
    temp = str(x).split('</strong></p>')[1].replace('</p>\n<p>', ' ').split('</div>\n</div>')[0].lower()
    cleaned.append(temp)
    #in order from newest to oldest
    char_len.append(len(temp))

num_war = [0,0,0,0]
index  = 0
for x in cleaned:
    num = [m.start() for m in re.finditer('war', x)]

    temp = []
    for n in num:
        if x[n+3] == ' ' or x[n+3] == '.' or x[n+3] =='!' or x[n+3] == ',': # 'wars' or war- or other stuff?
            temp.append(x[n:n+5])
    num_war[index] = len(temp)
    index += 1

most_common = []
for x in cleaned:
    temp = Counter(x.split()).most_common()[0]
    most_common.append(temp)

most_common
#just gonna organize colums here honestly
c1  = [char_len[0], num_war[0], most_common[0]]
c2 = [char_len[1], num_war[1], most_common[1]]
c3 =[char_len[2], num_war[2], most_common[2]]
c4 = [char_len[3], num_war[3], most_common[3]]

temp_dict = {columns[0] : c1,
                 columns[1]: c2,
                 columns[2]: c3,
                 columns[3] : c4}

df = pd.DataFrame(data = temp_dict)
df.rename(index={0:'Number of words', 1: 'Number of times "war" is used', 2: "Most common word and number of occurances"})
```

Out[21]:

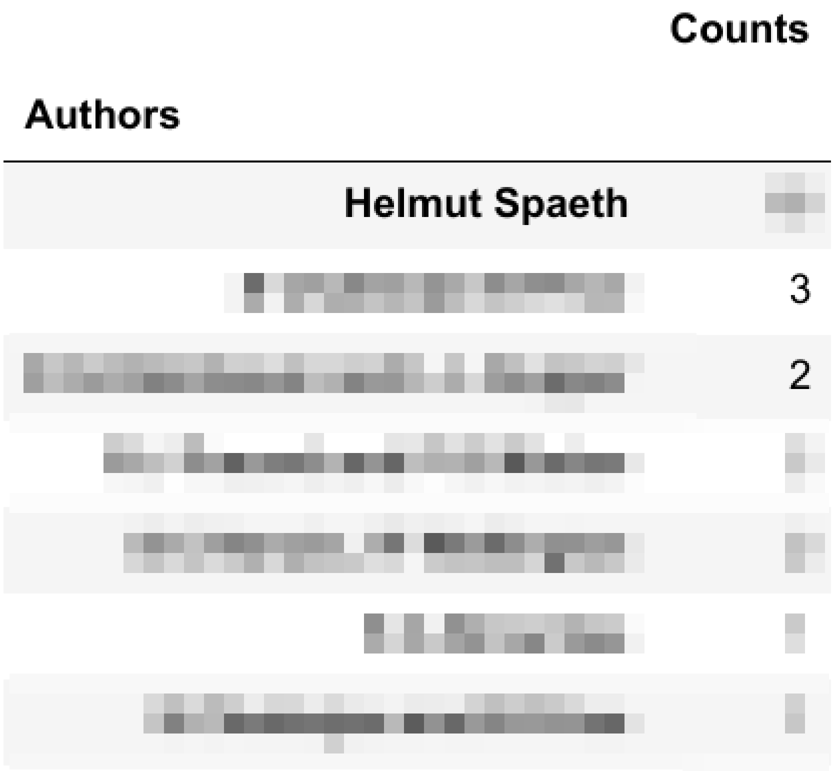
	September 25, 1988 Debate Transcript	October 7, 1984 Debate Transcript	September 23, 1976 Debate Transcript	September 26, 1960 Debate Transcript
Number of words	87641	86728	80745	60918
Number of times "war" is used	7	2	7	3
Most common word and number of occurances	(the, 798)	(the, 866)	(the, 855)	(the, 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/)](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 [29]:

```
# your code here
import operator

source = 'http://people.sc.fsu.edu/~jburkardt/datasets/regression'
gh = requests.get(source)
html = bs.BeautifulSoup(gh.content, features ='html.parser')
html.findAll('a')[6:33]

#its literally easier to just build it
urls = ['http://people.sc.fsu.edu/~jburkardt/datasets/regression/x' + st
r(x).zfill(2) + '.txt' for x in range(1,28)]

def get_fifth(url):
    """returns the damn fifth line"""
    temp = requests.get(url)
    html = bs.BeautifulSoup(temp.content, features ='html.parser')
    fifth = str(html).split('\n')[4]
    return fifth.replace('#    ', '').replace(',', ' ')

names = [get_fifth(x) for x in urls]
names

most_common = {}
for x in names:
    if x in most_common:
        most_common[x] += 1
    else:
        most_common[x] = 1

pd.DataFrame.from_dict(most_common, orient='index').rename(columns={0:'N
umber of occurences'}).sort_values(by=['Number of occurences'], ascendin
g=False)
```

Out[29]:

	Number of occurences
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