Zack Edwards

Professor Raz Saremi

EM-224A Informatics and Software Development

28 April 2020

Exercise 9

Procedure:

In order to create a word cloud of the most used words from the headlines on the front page of the NY Times first it is necessary to 'clean' the data, or transform it into usable and significant data for the computer to analyze. The first step in this process was to give our program an HTML source to pull data from, in this case, "https://www.nytimes.com/". This variable 'page', short for web page, includes all of the code that makes the website function and appears on your screen when you search for it. The next step is to create a 'soup' of words taken from the massive block of code. This is accomplished by giving the HTML parser a 'class' to look for, which will refer to a certain element on the web page. In this example, we want to examine the headlines on the NY Times website, so in order to find the class of the headlines I used the inspect element feature and found that the class for the headlines was "css-1ez5fsm esl82me1". By using this class I was able to pull 30 headlines from the NY Times home page.

The next step is to clean the data which is performed through a series of steps including removing punctuation, removing non-alphabetical characters, and lowercasing

all the words. The point of this is to remove non-significant information, and then standardize what is left. This makes it easier for the computer to read and analyze the data. For instance, the word cloud feature is based upon the percentage of instances of each word. However upper and lowercase version of words doesn't get counted as the same word and so don't reflect the actual percentage of that word in the block of text.

The next step is to create bigrams which are tuples containing each pair of words in the block. This is a tool that gives a computer slightly more context to use to analyze blocks of text. The idea is that computers find it much harder to analyze and extract meaning from a whole sentence rather than individual or pairs of words. My printed list of bigrams does not appear in tuples because in the next step we are asked to combine the list of words with the list of bigrams which will then be analyzed and turned into a word cloud, and it is much harder to analyze a list of data are some of the entries are normal and some are tuples. Specifically, tuples do not have an easy feature for lowercasing each character as lists do, which I could find.

Analysis:

Finally, to analyze the results of the word cloud. The top words from this weeks paper are all about the most pressing news which is all about coronavirus. For Tuesday The top works were 'pandemic', 'virus', 'trump', 'reopen', 'coronavirus', and 'Sweden'. All these words relate to the current pandemic we are experiencing and reflect the desire to reopen society. The discussion at the moment centres on the administration and the

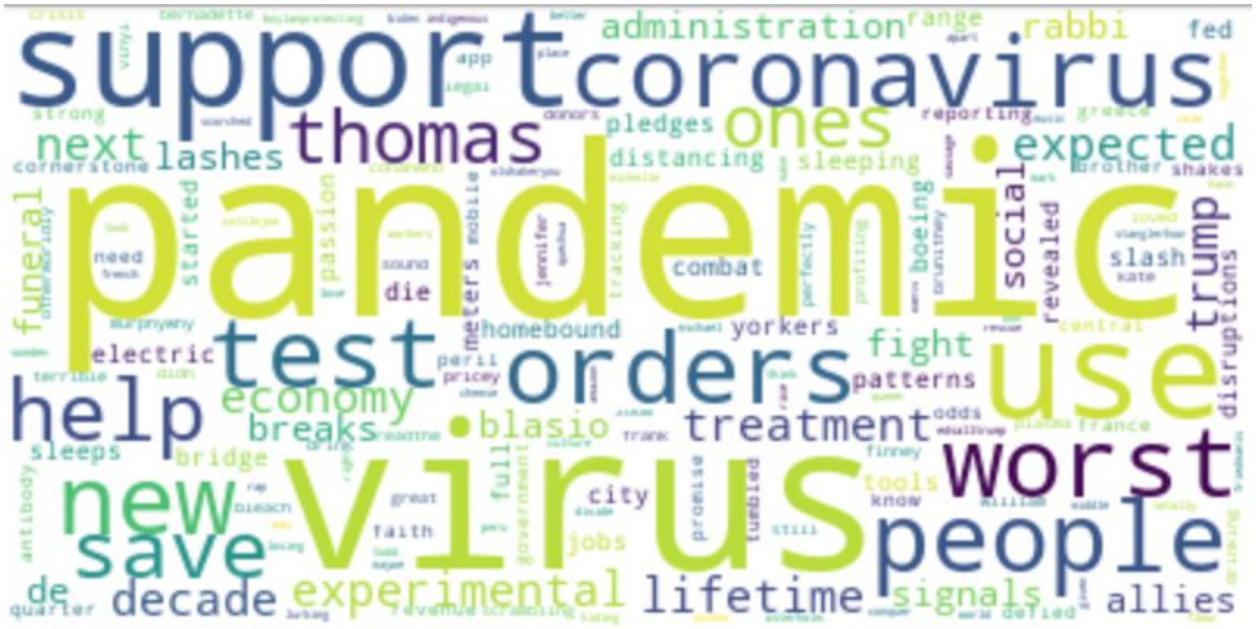
causes and effects of this lockdown. Sweden is mentioned because they have taken a different approach to the virus by not locking down, and so are an area of interest during this time. Since I completed and ran my code on Tuesday I was able to get results from Wednesday as well. This gives us the ability to compare and contrast the top words from different days of headlines just by running this program. This is already an incredibly useful and applicable tool which becomes exponentially more powerful in terms of the meaning we can extract when it is applied over a period of time. Anyway, the results from Wednesday are different from that of Tuesday, but still share some of the same top 3 words including ‘pandemic’, ‘virus’, and ‘coronavirus’. This shows how we are still discussing the ongoing lockdown. This is to be expected because this lockdown affects everyone in the united states and is internationally the most pressing news. Furthermore, it is an ongoing situation which will stay prevalent in the news long after it is resolved. The largest difference I was able to spot was the size of the word ‘trump’ which is the largest on Tuesday but one of the smallest on Wednesday. This indicates that he did something large and newsworthy to become the most used word but that it didn’t matter very much the next day or was overshadowed by newer stories.

This tool is very useful because it makes it easy to gauge the sentiment and topics featured in headlines the New York Times, a newspaper which often unbiasedly reflects the happenings of America as a whole.

Wordcloud:

Tuesday, April 28th 2020

Wednesday, April 29th 2020



Code input:

```
#Author Zack Edwards
#Excersize 9

import urllib.request
import pandas as pd
from bs4 import BeautifulSoup
import requests
import re
import nltk
import pandas as pd
import string
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt

page = requests.get("https://www.nytimes.com/")

soup = BeautifulSoup(page.content, 'html.parser')

word_list = []
temp_list = []
#print(soup.prettify())

#This loop reads the data from the website into a readable 'soup' by focusing on the headlines denoted
#by the code: css-1ez5fsm esl82me1
for story_heading in soup.find_all(class_="css-1ez5fsm esl82me1"):
    story_title = story_heading.text.replace("\n", " ").strip()
    new_story_title = story_title.encode('utf-8')
    word_list.append(story_title)
print("The following is the first 15 entries in a list of headlines taken from NY Times")
print(word_list[:15])

for i in word_list: #This loops gets ride of the punctuation
    l = nltk.word_tokenize(i)
    ll = [x for x in l if not re.fullmatch('[' + string.punctuation + ']+', x)]
    ll = "".join([" "+i if not i.startswith("'") and i not in string.punctuation else i for i in ll]).strip()
    temp_list.append(str(ll))
word_list = temp_list
temp_list = []

stop = open('stopwords_en.txt', 'r')
stops = [] #initializing and cleaning the list of stopwords
stop_list = stop.readlines()
for i in stop_list:
    stops.append(i[:-1])
stop_list = stops

for i in word_list: #This loops eliminates stop words from the data
    for x in i.split():
        if x not in stops:
            if "‘" in x:
                continue
```

```
                continue
            else:
                temp_list.append(x)
word_list = temp_list
temp_list = []

for i in word_list:
    if i.isdigit() == False:
        if "'" in i:
            continue
        else:
            temp_list.append(i)
word_list = temp_list

#This is our final list of words for the headlines
print('\nThe following is the first 20 cleaned words from the headlines')
print(word_list[:20])

temp_bigrm = []
bigrm = list(nltk.bigrams(word_list))
for i in bigrm:
    for x in i:
        if "'" in x:
            continue
        else:
            temp_bigrm.append(x)
bigrm = temp_bigrm
print("\nThe following is the first 20 bigrams from the headlines")
print(bigrm[:20])

#Combining the bigrams and original list of words
temp_list = word_list
for i in bigrm:
    for x in i:
        temp_list.append(x)
#print('\n', temp_list)

# Save a lower-case version of each word to a list
words_list = []
for i in word_list:
    words_list.append(i.lower())

# Eliminate non alpha elements
text_list = [word.lower() for word in words_list if word.isalpha()]

# Transforming the list into a string for displaying
text_str = ' '.join(text_list)

# Crating and updating the stopword list
stpwords = set(STOPWORDS)
stpwords.add('will')
```

```
temp_bigrm = []
bigrm = list(nltk.bigrams(word_list))
for i in bigrm:
    for x in i:
        if "‘" in x:
            continue
        else:
            temp_bigrm.append(x)
bigrm = temp_bigrm
print("\nThe following is the first 20 bigrams from the headlines")
print(bigrm[:20])

#Combining the bigrams and original list of words
temp_list = word_list
for i in bigrm:
    for x in i:
        temp_list.append(x)
#print('\n', temp_list)

# Save a lower-case version of each word to a list
words_list = []
for i in word_list:
    words_list.append(i.lower())

# Eliminate non alpha elements
text_list = [word.lower() for word in words_list if word.isalpha()]

# Transforming the list into a string for displaying
text_str = ' '.join(text_list)

# Crating and updating the stopword list
stpwords = set(STOPWORDS)
stpwords.add('will')
stpwords.add('said')

# Defining the wordcloud parameters
wc = WordCloud(background_color="white", max_words=2000,
               stopwords=stpwords)

# Generate word cloud
wc.generate(text_str)

# Store to file
wc.to_file('nytwordcloud.png')


print('\nThis is the wordcloud for the NY Times headlines')
# Show the cloud
plt.imshow(wc)
plt.axis('off')
plt.show()
```

Code Output:

```
The following is the first 15 entries in a list of headlines taken from NY Times
['Spiking Death Rates Suggest Pandemic Toll Is Undercounted', 'Trump Declares Meat Supply ‘Critical,’ Aimin
g to Reopen Plants', ''Life Has to Go On’: How Sweden Has Faced the Virus Without a Lockdown', 'Despite Tru
mp’s Nudging, Schools Are Likely to Stay Shut for Months', 'De Blasio Breaks Up Rabbi’s Funeral and Lashes
Out Over Social Distancing', ''Governors Don’t Do Global Pandemics’: Cuomo Faults Others Over Virus', 'Anal
ysis: Trump’s Response to Virus Reflects a Long Disregard for Science', 'Pence Tours Mayo Clinic and Flouts
Its Rule That All Visitors Wear a Mask', 'The Trump administration’s message on reopening continues to be c
ontradictory.', 'Investors Bet Giant Companies Will Dominate After Crisis', 'Lawmakers’ next big fight: Sho
uld businesses be liable if they reopen and workers or customers get the coronavirus?', 'Treasury Races to
Recoup Money From Big Companies That Received Loans', 'Airborne Coronavirus Detected in Wuhan Hospitals',
'Greece Has ‘Defied the Odds’ in the Pandemic. At Least, So Far.', 'Packed With Migrant Workers, Dormitorie
s Fuel Virus in Singapore']

The following is the first 20 cleaned words from the headlines
['Spiking', 'Death', 'Rates', 'Suggest', 'Pandemic', 'Toll', 'Is', 'Undercounted', 'Trump', 'Declares', 'Me
at', 'Supply', 'Critical', 'Aiming', 'Reopen', 'Plants', 'Life', 'Has', 'Go', 'On']

The following is the first 20 bigrams from the headlines
['Spiking', 'Death', 'Death', 'Rates', 'Rates', 'Suggest', 'Suggest', 'Pandemic', 'Pandemic', 'Toll', 'Tol
l', 'Is', 'Is', 'Undercounted', 'Undercounted', 'Trump', 'Trump', 'Declares', 'Declares', 'Meat']

This is the wordcloud for the NY Times headlines
```