Homework Assignment 1 - BIA 6304 - Corey Austen

Due Date: 5:45pm, Wednesday, January 24 Deliverable Format: HTML, PDF, or Word document(s). Turn in output from the tasks and the answers to the questions. They can be in the same file or separate files. Uncompiled Notebooks (files with .ipynb) will NOT be accepted.

This assignment is worth 19 points. Assignments turned in after the due date and time will lose 2 points for every day late. No assignments will be accepted after January 31.

- Answer corresponding questions (denoted by Q#) by inserting markdown text or creating a separate text document. Questions are assessed as follows:

• Complete answer using appropriate terminology and not adding unnecessary or incorrect text: 5 points. • Mostly complete answer that may not use all language appropriately: 3 points • Incomplete answer or one that uses incorrect language: 1 point • Incorrect answer: 0 points

Hint: while concise answers are appreciated, please make sure your responses are in complete sentences and are at least one paragraph long. Single word responses or sentence fragments will be considered incomplete answers.

Questions

Q1. Write a short description of what the data you have captured are and how they might be used. Make sure your answer is no longer than two paragraphs, and should at minimum answer these questions: • What information can we get from the text? What question might you answer? Why is that interesting? • What information besides the text (e.g. tweet or headline) might you need to answer your question? Why? Audience: general – management or non-technical staff.

Q2. Write a short description of what the vector space represents and how you can use it. Make sure your answer is no longer than three paragraphs, and should at minimum answer these questions:

• How do the parameter settings affect the size of the feature space? • Does using weights make sense for your question? If so, why? If not, why not? • What parameter settings might be "best" for the question you have in mind?

Hint: if you want to show numbers, a table would work well here.
Audience: technical – fellow data scientists or other technical staff.

Q3. Besides emojis and encodings that we discussed in class, what other changes might you consider making to the text, if any? What effect would you expect this to have? If you wouldn't make any other changes, explain why not. Audience: general – management or non-technical staff.

Answers

A1.

We can get a sense of what is going on in the world based on the most common words in the headlines. You might be able to answer some questions concerning internet virality, like if an action taken by a company or organization appeared in the headlines, it could be used to gauge the reaction to news very quickly, etc.

The information that you might need in order to really answer this question could be web traffic data. Some websites or blogs could be posting a considerable amount of content related to a certain news story or regarding a company, but if that particular website has low web traffic, then it would not be an accurate representation of how popular a headline is. Some websites have data somewhere on the page that shows how many times a page has been viewed, shared on Facebook, tweeted about, etc. Gathering this alongside the test would be more beneficial than grabbing the text alone.

A2.

The vector space represents the useable data that was mined from the corpus. We can use the feature space to gather as a dataset for predictive models, as a way to find the most important feature in a group of documents, etc. I created 4 different vectorizers to examine how they affect the size and feature variety within the feature space. What I noticed by doing this is that by putting more restrictive parameters in the vectorizer, the feature space would shrink. Parameters like min_df and max_df, when set to a small window, will drastically reduce size of the feature space. Removing stop words will also reduce the features. A wide window on min_df and max_df, or not setting them at all, will expand the feature space. Setting the ngram_range to more then one word will drastically increase the feature space, as single words and combinations of words are then included.

Using weights in the vectorizers for this question would not make sense. This is because the headlines are very short and we would be looking for how often a company or term appears in the headlines. The weight would indicate how important a word is in each headline, but since they are so short and vary so much, the weights will be high for a lot of terms, as seen in the vectorizers 3 and 4 that I created below. If we were really trying to answer this question (with a much larger corpus), the ideal parameters for this would probably look similar to this:

optimal_vectorizer = CountVectorizer(binary=False, min_df = .05, max_df = .8, stop_words = "english", ngram_range = (1,2), lowercase = False)

The min_df and max_df range is set this way to capture words that appear often enough to not be totally unique, but not ones that appear in every headline. Stop words would not be important. Single words and two word phrases might be important here, as a positive phrase, negative phrase, or company name could be multiple words. The lowercase setting would be useful here because the company name may be a word or phrase that consists of common terms, like Bank of America.

A3.

The text was fairly easy to scrape, but one could make is to have the text say whether it was a sound bite or a video. I'm sure this could be done by creating an if statement where it looks for a particular piece of HTML and enters a value into a dictionary. I tried to do this with the subcategories, such as "The Two-Way" and "Politics," as that can be seen in the HTML. However, I was unable to figure out how to extract this text, as a python dictionary does not allow for duplicate keys. I should have been able to extract them into a list, but I was unsure how to code that.

Task 1

Create a data frame of news headlines similar to the example done in class. The source can be any “reputable” news website. Include the headline, the date, a category or classification, and one other descriptive term about the headline (e.g. short/long, opinion, includes video?, etc.) and show only the head and the tail of the data frame.

```
In [41]: # import module(s) into namespace
         import pandas as pd
         import numpy as np
         from bs4 import BeautifulSoup
         import requests
         from sklearn.feature_extraction.text import CountVectorizer
         from sklearn.feature_extraction.text import TfidfVectorizer
         import math
         pd.set_option('display.max_colwidth', 150)
```

I decided to use the "News" section on NPR.com

http://www.npr.com/sections/news/ (http://www.npr.com/sections/news/)

```
In [42]: # Let's see what is going on on NPR.com
         page = requests.get('https://www.npr.org/sections/news/')

         soup = BeautifulSoup(page.text, "html5lib")
```

```
In [ ]: #I'll avoid printing out the HTML.
        #print(type(soup))
```

```
In [43]: NPRobj = soup.find_all("h2",{"class":"title"})
```

Here is where I am adding in the subtopics as seen on the website. I tried to pull the data directly from the website, but I was getting caught up on the code.

In [44]:
```
slug = ["National Security", "The Two-Way", "The Two-Way", "Shots - Health New
s", "The Two-Way", "The Rise of the Contract Workers", "Monkey See", "The Two-
Way", "The Two-Way", "Shots - Health News", "The Two-Way", "The Salt", "The Tw
o-Way", "Politics", "The Two-Way", "Shots - Health News", "Shots - Health New
s", "The Two-Way", "The Two-Way", "The Two-Way", "The Two-Way", "The Two-Way",
 "The Rise of the Contract Workers", "The Torch", "Politics"]
nprdf2 = pd.DataFrame({'Sub-Category':slug}) #Creating a manual list of the ad
ded topics
nprdf2.columns = ['Sub-Category']
nprdf2.head()
```

Out[44]:

	Sub-Category
0	National Security
1	The Two-Way
2	The Two-Way
3	Shots - Health News
4	The Two-Way

In [45]:
```
import time
print(len(NPRobj))
headlines = {}

for name in NPRobj:
    headlines.setdefault(name.get_text(strip=True), [])
    headlines[name.get_text(strip=True)].append(time.strftime("%m/%d/%Y"))
    headlines[name.get_text(strip=True)].append("Front Page News")
```

```
25
```

In [46]:
```
# store the text in a useful format

nprdf1 = pd.DataFrame.from_dict(headlines,orient="index") #Creating a datafram
e
nprdf1.reset_index(level=[0], inplace=True) #Setting the index
print(nprdf1.shape)

nprdf1.columns = ['Headline', 'Date', 'Category'] #Setting Column Headers
nprdf1 = nprdf1.replace('\n','', regex=True) #Removing Spaces
```

```
(25, 3)
```

In [47]:
```
nprdf = pd.concat([nprdf1, nprdf2],axis=1) #merging the two dataframes togethe
r
nprdf.head()
```

Out[47]:

	Headline	Date	Category	Sub-Category
0	Sessions Interviewed By Special Counsel Robert Mueller As Part Of Russia Inquiry	01/24/2018	Front Page News	National Security
1	Shooting At Kentucky High School Leaves 2 Dead, At Least 17 Injured	01/24/2018	Front Page News	The Two-Way
2	Motel 6 Sued For Identifying Latino Guests For Immigration Agents	01/24/2018	Front Page News	The Two-Way
3	E-Cigarettes Likely Encourage Kids To Try Tobacco But May Help Adults Quit	01/24/2018	Front Page News	Shots - Health News
4	Ursula K. Le Guin, Whose Novels Plucked Truth From High Fantasy, Dies At 88	01/24/2018	Front Page News	The Two-Way

In [48]:
```
nprdf.tail()
```

Out[48]:

	Headline	Date	Category	Sub-Category
20	Puerto Rico's Governor Announces Plan To Privatize Island's Troubled Electric Utility	01/24/2018	Front Page News	The Two-Way
21	Trump Slaps Tariffs On Imported Solar Panels And Washing Machines	01/24/2018	Front Page News	The Two-Way
22	Will Work For No Benefits: The Challenges Of Being In The New Contract Workforce	01/24/2018	Front Page News	The Rise of the Contract Workers
23	Gus Kenworthy Will Be The Second Openly Gay Man To Compete For U.S. In Winter Games	01/24/2018	Front Page News	The Torch
24	Trump Signs Funding Bill, Bringing Shutdown To An End	01/24/2018	Front Page News	Politics

Task 2

Use the default to convert to lower case. Generate a vector space model. Try various combinations of vectorizer settings like we did in class: changing case, eliminating stop words, using min and max document frequency settings, choosing n-grams, applying Tfidf-weights, etc. Try at least 4 different versions.

Vectorizer Version 1

This Vectorizer uses word counts, a min_df of 5%, and a max_df of 80%, and does not remove stop words.

```
In [53]: #The first configuration!

vectorv1 = CountVectorizer(binary=False, min_df = .05, max_df = .8) #define th
e transformation
vectorv1_news = vectorv1.fit_transform(nprdf['Headline']) #apply the transform
ation
print(vectorv1_news.shape)

names = vectorv1.get_feature_names()   #create list of feature names
count = np.sum(vectorv1_news.toarray(), axis = 0) # add up feature counts
count2 = count.tolist()  # convert numpy array to list
count_df = pd.DataFrame(count2, index = names, columns = ['count']) # create a
 dataframe from the list
vector1df = count_df.sort_values(['count'], ascending = False)  #arrange by co
unt instead
print(vector1df.head(5)) #Show top 5 counts
```

```
(25, 27)
     count
to       8
of       7
the      6
at       5
for      5
```

Vectorizer Version 2

This Vectorizer uses word counts, a min_df of 5%, a max_df of 80%, and removes stop words.

In [54]:
```
# Configuration 2!

vectorv2 = CountVectorizer(binary=False, min_df = .1, max_df = .8, stop_words
= "english") #define the transformation
vectorv2_news = vectorv2.fit_transform(nprdf['Headline']) #apply the transform
ation
print(vectorv2_news.shape)

names = vectorv2.get_feature_names()   #create list of feature names
count = np.sum(vectorv2_news.toarray(), axis = 0) # add up feature counts
count2 = count.tolist()  # convert numpy array to list
count_df = pd.DataFrame(count2, index = names, columns = ['count']) # create a
 dataframe from the list
vector2df = count_df.sort_values(['count'], ascending = False)  #arrange by co
unt instead
print(vector2df.head(5)) #Show top 5 counts
```

```
(25, 3)
       count
says       3
trump      3
work       3
```

Vectorizer Version 3

This Vectorizer uses weights, removes stop words, a min_df of 5%, and a max_df of 70%.

In [55]:
```
#Configuration 3!

vectorv3 = TfidfVectorizer(use_idf=True, norm=None, stop_words = "english",
                       min_df = .05, max_df = .7) #define the transformation
vectorv3_dm = vectorv3.fit_transform(nprdf['Headline']) #apply the transformat
ion
print(vectorv3_dm.shape)

names = vectorv3.get_feature_names()
maxweight = np.max(vectorv3_dm.toarray(), axis = 0) # find the max weight of e
ach feature
maxweight2 = maxweight.tolist()  # convert numpy array to list

maxweight_df = pd.DataFrame(maxweight2, index = names, columns = ['count']) #
 create a dataframe from the list
maxweight_sort = maxweight_df.sort_values(['count'], ascending = False)
print(maxweight_sort.head(5)) #Show the top 5 weights
```

```
(25, 12)
             count
alert     3.159484
contract  3.159484
dies      3.159484
funding   3.159484
governor  3.159484
```

Vectorizer Version 4

This Vectorizer uses weights, removes stop words, a min_df of 5%, creates new features based on uppercase vs. lowercase words, and an ngram range of 1 to 2 words.

In [56]:
```
#Configuration 4, this time with weights!
vectorv4 = TfidfVectorizer(use_idf=True, norm=None, stop_words = "english",
                         min_df=.05, ngram_range = (1,2), lowercase=False) #de
fine the transformation
vectorv4_dm = vectorv4.fit_transform(nprdf['Headline']) #apply the transformat
ion
print(vectorv4_dm.shape)

names = vectorv4.get_feature_names()
maxweight = np.max(vectorv4_dm.toarray(), axis = 0) # find the max weight of e
ach feature
maxweight2 = maxweight.tolist()  # convert numpy array to list

maxweight_df = pd.DataFrame(maxweight2, index = names, columns = ['count']) #
 create a dataframe from the list
maxweight_sort = maxweight_df.sort_values(['count'], ascending = False)
print(maxweight_sort.head(10)) #Show the top 10 weights
```

```
(25, 29)
             count
At        5.297317
For       5.297317
The       4.932674
After     3.159484
Governor  3.159484
Why       3.159484
Plan      3.159484
Part Of   3.159484
Part      3.159484
New       3.159484
```