Bias Detection in American Journalism

Authors: Aaron Carr, Azucena Faus, Dave Friesen

Company Industry: News Media

Company Size: 3 (startup)

GitHub Repository: https://github.com/fausa/Bias_Detection_in_Journalism

Abstract:

There is a general mistrust of news media where the public follows news that mimic their own political biases. But there is a growing appetite for unbiased news and growing subscriptions for centered news sources.

Problem Statement:

The question of mainstream media bias is a significant area of contention in U.S. politics. At the same time, the political 'inclination' of a news outlet often aligns with the personal perspectives of its audience. This overlap provides a unique opportunity and our project's objective: To leverage sophisticated text mining methods and supporting data science techniques to classify the political bias of leading online news sources. This classification will be based on a combination of "politically polarizing" terms as identified through an impartial (academic) source, as well as sentiment analysis context around the use of these terms (Liu et al., 2022).

Objective:

The objective of developing this classifier will be to advise our clients who are the executives at a news source that prides itself in being considered politically “centered.” We would use this classifier to analyze their content on a continuous basis and report back the overall/average political “lean” of their articles. The details regarding the metric for overall publication lean will be based on an average of left/right leaning probabilities per article. This feedback will then provide our client with actionable insights so they ensure their overall political lean metric remains centered.

Goals:

The success of this project will contribute to the achievement of the following goals:

1. Using article content from Fox News, CNN, Breitbart, and The Washington Post, along with media bias information from AllSides Media (n.d.), develop a classifier with at least a 90% F1 Score on testing data that can differentiate "left" from "right" leaning articles.
2. Train a model to predict political lean on unseen news articles that are sourced from "centered" news outlets and confirm the bias ratings from AllSides Media (n.d.).
3. Make recommendations to our client, The Hill, on whether their online journalism remains centered or has shifted in political lean, as well as provide solutions and next steps.

Ultimately, the goal of this project is to provide clients who wish to stay true to ethical, unbiased journalism with a bias rating for their news content throughout the year, giving them the opportunity to find out which articles are causing that shift and make the necessary steps to mitigate such issues.

Name of your selected dataset:

Queried news articles from CNN, Fox News, Breitbert, and The Washington Post covering most of May and part of June, 2023.

Description of your selected dataset (data source, number of variables, size of dataset, etc.):

News articles will be sourced via a REST API called NewsAPI, which logs information on “current and historic news articles published by over 80,000 worldwide sources” (NewsAPI, n.d.). Out of the many possible attributes returned for each API query, this project will use six: source name, author, title, url, publishedAt, and content. Prior to data preprocessing, an additional feature (article_text) will be used to store the scraped data from each specific URL. The size of the final dataset (*N* = 4,026) was limited by both time restrictions, as well as web scraping access to specific sites or articles.

Queries for topics of political interest are used to gather articles from explicitly chosen sources. Independent studies show the political lean for each of these sources (CNN, “left”; Fox News, “right”; The Washington Post, “left”; Breitbert News, “right”) and this will help with training and validation of our classifier (AllSides, n.d.; Ralph & Relman, 2018).

GitHub Data Locations:

Master Persisted Train/Test Dataset: data/master.csv

Associated Press Dataset: data/master_tokenized_AP.csv

The Hill Dataset: data/master_business_TheHill.csv

Methods Used:

- Exploratory data analysis (EDA)
- Text data preprocessing (e.g., normalization, tokenization)
- Term frequency-inverse document frequency (TF-IDF) vectorization
- Train/test split
- Classification
- Machine learning
- Hyperparameter tuning
- Model evaluation (e.g., confusion matrix, F1)
- Sentiment Analysis
- Topic Modeling

References

AllSides. (n.d.). *AllSides Media Bias Chart*. Retrieved June 6, 2023 from https://www.allsides.com/media-bias/media-bias-chart

Liu, R., Jia, C., Wei, J., Xu, G., & Vosoughi, S. (2022). Quantifying and alleviating political bias in language models. *Artificial Intelligence*, *304*. https://doi.org/10.1016/j.artint.2021.103654

NewsAPI. (n.d.). *Search worldwide news with code*. Retrieved June 5, 2023 from https://newsapi.org/

Ralph, P., & Relman, E. (2018, September 2). These are the most and least biased news outlets in the US, according to Americans. *Business Insider*. https://www.businessinsider.com/most-biased-news-outlets-in-america-cnn-fox-nytimes-2018-8?op=1#and-heres-how-republicans-ranked-them-from-fox-news-to-cnn-20Page

PART I:
Data Collection

ADS509_CarrA_Final_Project_01_API_v2

June 24, 2023

1 509 Final Project

This notebook connects to the NewAPI to return JSOn objects based on specific queries, loads specific elements from the object, then persists the data in a MySQL table.

1.1 Resolve dependencies

[1]:
```
! pip install newsapi-python
```

```
Requirement already satisfied: newsapi-python in
c:\users\acarr\anaconda3\envs\ds-py38-tf\lib\site-packages (0.2.7)
Requirement already satisfied: requests<3.0.0 in
c:\users\acarr\anaconda3\envs\ds-py38-tf\lib\site-packages (from newsapi-python)
(2.28.1)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\acarr\anaconda3\envs\ds-py38-tf\lib\site-packages (from
requests<3.0.0->newsapi-python) (2023.5.7)
Requirement already satisfied: idna<4,>=2.5 in c:\users\acarr\anaconda3\envs\ds-
py38-tf\lib\site-packages (from requests<3.0.0->newsapi-python) (3.4)
Requirement already satisfied: charset-normalizer<3,>=2 in
c:\users\acarr\anaconda3\envs\ds-py38-tf\lib\site-packages (from
requests<3.0.0->newsapi-python) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
c:\users\acarr\anaconda3\envs\ds-py38-tf\lib\site-packages (from
requests<3.0.0->newsapi-python) (1.26.15)
```

1.2 Globally import libraries

[2]:
```
import numpy as np
import pandas as pd
import pymysql as mysql
import matplotlib.pyplot as plt
import os
import shutil
import re
import logging
import time
import zipfile
```

```
import requests
from bs4 import BeautifulSoup
import datetime as dt
import re
import regex as rex
from collections import defaultdict, Counter
import random
#import mysql.connector

from newsapi import NewsApiClient

# Set pandas global options
pd.options.display.max_rows = 17
```

[3]:
```
today = dt.date.today()
print(today)
print(type(today))
```

```
2023-06-18
<class 'datetime.date'>
```

1.3 Connect to NewsAPI client

[4]:
```
api_key = os.environ['NewsAPIKey']

# Init
newsapi = NewsApiClient(api_key=api_key)
```

1.4 Pull article info from API

sources = newsapi.get_sources() print(sources)

[5]:
```
def news_api_urls(q=None,
                  s=None,
                  d_from='2023-05-01',
                  d_to='2023-05-31',
                  api_lst=[]):
    '''Access API and pull content from resulting JSON object'''
    all_articles = newsapi.get_everything(q=q,
                                          domains=s,
                                          from_param=d_from,
                                          to=d_to,
                                          language='en',
                                          sort_by='relevancy',
                                          page=1)

    #print(type(all_articles))
    #print(all_articles)
```

```
    #print('Article list: ', all_articles['articles'])
    #for article in all_articles['articles']:
        #print('Source ID:', article['source']['id'])
        #print('Source name:', article['source']['name'])
        #print('Author:', article['author'])
        #print('Title:', article['title'])
        #print('URL:', article['url'])
        #print('Publish date:', article['publishedAt'])
        #print('Article text:', article['content'], '\n')

    # Create a list of tuples from the dictionary data
    source_data01 = [(a['source']['name'],
                      a['author'],
                      a['title'],
                      a['url'],
                      a['publishedAt'],
                      a['content'])
                     for a in all_articles['articles']]

    api_lst.extend(source_data01)
    #print(api_lst)
    return(len(api_lst))
```

1.5 Connect to API to access URLs

1.5.1 Set API filter parameters

A request grid was established to return specific combinations of sources ('left' and 'right', as determined by referencing Allsides (2022) and Ralph et al. (2018)); dates; and keywords ("hot button" general terms, as inspired by Liu et al. (2022)). The grid was necessary to maximize the returned JSON objects, as each request/page was limited to 100 URLs. As an example, a grid of 4 sources, over 4 days, using 6 complex query terms resulted in a 96-request grid, where each request returned between 0 and 100 objects. Since there were some articles that overlapped in terms of keywords, and some sources did not include a specific keyword on any given day, the actual number of URLs was significantly less than the potential maximum.

[6]:
```
# Total API request grid: Sources x dates x keyword queries
'''Left/Right source selection critera: Allsides, 2022; Ralph et al., 2018'''
source_lst = ['slate.com', 'vox.com']
#source_lst = ['cnn', 'the-washington-post', 'fox-news', 'breitbart-news']
#source_lst = ['cnn', 'reuters', 'fox-news']
#source_lst = ['cnn', 'newsweek', 'fox-news']
#source_lst = ['newsweek']
#source_lst = ['cnn', 'fox-news']
#source_lst = ['breitbart-news']

'''Dates needed to be sliced based on NewsAPI rate limit of 100 request/day'''
#date_lst = ['2023-06-16', '2023-06-15']
```

```
date_lst = ['2023-06-14', '2023-06-13', '2023-06-12', '2023-06-11',␣
 ↪'2023-06-10', '2023-06-09']
#date_lst = ['2023-06-11', '2023-06-10', '2023-06-09', '2023-06-08']
#date_lst = ['2023-06-07', '2023-06-06', '2023-06-05', '2023-06-04']
#date_lst = ['2023-06-03', '2023-06-02', '2023-06-01', '2023-05-31',␣
 ↪'2023-05-30', '2023-05-29']
#date_lst = ['2023-06-03', '2023-06-02']
#date_lst = ['2023-06-01', '2023-05-31']
#date_lst = ['2023-05-30', '2023-05-29']
#date_lst = ['2023-05-17', '2023-05-16', '2023-05-15', '2023-05-14',␣
 ↪'2023-05-13']
#date_lst = ['2023-05-12', '2023-05-11', '2023-05-10', '2023-05-09']
#date_lst = ['2023-05-08']
#date_lst = ['2023-05-07', '2023-05-06', '2023-05-05']

'''Keyword selection critera: Liu et al., 2022'''
q_word_lst = ['gender OR male OR female OR transgender',
              'security AND (social OR national)',
              'justice OR surveillance',
              'healthcare OR "health care"',
              '''(political AND (bias OR party)) OR republican
              OR democrat OR election''',
              '''(policy AND (drug OR "affirmative action"))
              OR regulate OR regulation''']
```

1.5.2 Access API

[7]:
```
'''Run individual request for each source/data/keyword combo
to maximize data scraped'''
api_record_lst01 = []

for s in source_lst:
    #print(f'Source: {s}')
    for d in date_lst:
        #print(f'Date: {d}')
        for q in q_word_lst:
            #print(f'Query word: {q}')
            time.sleep(5 + 11 * random.random())
            lst_len = news_api_urls(q=q,
                                    s=s,
                                    d_from=d,
                                    d_to=d,
                                    api_lst=api_record_lst01)
            print(f'Source: {s}; Date: {d}; Query: {q}; Count: {lst_len}')

    # Random wait to prevent access shutdown
    time.sleep(10 + 13 * random.random())
```

```
#print(api_record_lst01)
#print(len(api_record_lst01))
```

```
Source: slate.com; Date: 2023-06-14; Query: gender OR male OR female OR
transgender; Count: 2
Source: slate.com; Date: 2023-06-14; Query: security AND (social OR national);
Count: 3
Source: slate.com; Date: 2023-06-14; Query: justice OR surveillance; Count: 8
Source: slate.com; Date: 2023-06-14; Query: healthcare OR "health care"; Count:
8
Source: slate.com; Date: 2023-06-14; Query: (political AND (bias OR party)) OR
republican
              OR democrat OR election; Count: 12
Source: slate.com; Date: 2023-06-14; Query: (policy AND (drug OR "affirmative
action"))
              OR regulate OR regulation; Count: 15
Source: slate.com; Date: 2023-06-13; Query: gender OR male OR female OR
transgender; Count: 18
Source: slate.com; Date: 2023-06-13; Query: security AND (social OR national);
Count: 22
Source: slate.com; Date: 2023-06-13; Query: justice OR surveillance; Count: 28
Source: slate.com; Date: 2023-06-13; Query: healthcare OR "health care"; Count:
29
Source: slate.com; Date: 2023-06-13; Query: (political AND (bias OR party)) OR
republican
              OR democrat OR election; Count: 34
Source: slate.com; Date: 2023-06-13; Query: (policy AND (drug OR "affirmative
action"))
              OR regulate OR regulation; Count: 34
Source: slate.com; Date: 2023-06-12; Query: gender OR male OR female OR
transgender; Count: 36
Source: slate.com; Date: 2023-06-12; Query: security AND (social OR national);
Count: 37
Source: slate.com; Date: 2023-06-12; Query: justice OR surveillance; Count: 43
Source: slate.com; Date: 2023-06-12; Query: healthcare OR "health care"; Count:
44
Source: slate.com; Date: 2023-06-12; Query: (political AND (bias OR party)) OR
republican
              OR democrat OR election; Count: 46
Source: slate.com; Date: 2023-06-12; Query: (policy AND (drug OR "affirmative
action"))
              OR regulate OR regulation; Count: 48
Source: slate.com; Date: 2023-06-11; Query: gender OR male OR female OR
transgender; Count: 51
Source: slate.com; Date: 2023-06-11; Query: security AND (social OR national);
Count: 53
Source: slate.com; Date: 2023-06-11; Query: justice OR surveillance; Count: 54
```

Source: slate.com; Date: 2023-06-11; Query: healthcare OR "health care"; Count: 56
Source: slate.com; Date: 2023-06-11; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 58
Source: slate.com; Date: 2023-06-11; Query: (policy AND (drug OR "affirmative action"))
OR regulate OR regulation; Count: 58
Source: slate.com; Date: 2023-06-10; Query: gender OR male OR female OR transgender; Count: 60
Source: slate.com; Date: 2023-06-10; Query: security AND (social OR national); Count: 61
Source: slate.com; Date: 2023-06-10; Query: justice OR surveillance; Count: 63
Source: slate.com; Date: 2023-06-10; Query: healthcare OR "health care"; Count: 63
Source: slate.com; Date: 2023-06-10; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 64
Source: slate.com; Date: 2023-06-10; Query: (policy AND (drug OR "affirmative action"))
OR regulate OR regulation; Count: 64
Source: slate.com; Date: 2023-06-09; Query: gender OR male OR female OR transgender; Count: 67
Source: slate.com; Date: 2023-06-09; Query: security AND (social OR national); Count: 72
Source: slate.com; Date: 2023-06-09; Query: justice OR surveillance; Count: 84
Source: slate.com; Date: 2023-06-09; Query: healthcare OR "health care"; Count: 85
Source: slate.com; Date: 2023-06-09; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 90
Source: slate.com; Date: 2023-06-09; Query: (policy AND (drug OR "affirmative action"))
OR regulate OR regulation; Count: 93
Source: vox.com; Date: 2023-06-14; Query: gender OR male OR female OR transgender; Count: 97
Source: vox.com; Date: 2023-06-14; Query: security AND (social OR national); Count: 99
Source: vox.com; Date: 2023-06-14; Query: justice OR surveillance; Count: 107
Source: vox.com; Date: 2023-06-14; Query: healthcare OR "health care"; Count: 108
Source: vox.com; Date: 2023-06-14; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 111
Source: vox.com; Date: 2023-06-14; Query: (policy AND (drug OR "affirmative action"))
OR regulate OR regulation; Count: 114
Source: vox.com; Date: 2023-06-13; Query: gender OR male OR female OR

transgender; Count: 116
Source: vox.com; Date: 2023-06-13; Query: security AND (social OR national); Count: 118
Source: vox.com; Date: 2023-06-13; Query: justice OR surveillance; Count: 122
Source: vox.com; Date: 2023-06-13; Query: healthcare OR "health care"; Count: 123
Source: vox.com; Date: 2023-06-13; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 126
Source: vox.com; Date: 2023-06-13; Query: (policy AND (drug OR "affirmative action"))
OR regulate OR regulation; Count: 126
Source: vox.com; Date: 2023-06-12; Query: gender OR male OR female OR transgender; Count: 128
Source: vox.com; Date: 2023-06-12; Query: security AND (social OR national); Count: 128
Source: vox.com; Date: 2023-06-12; Query: justice OR surveillance; Count: 130
Source: vox.com; Date: 2023-06-12; Query: healthcare OR "health care"; Count: 133
Source: vox.com; Date: 2023-06-12; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 138
Source: vox.com; Date: 2023-06-12; Query: (policy AND (drug OR "affirmative action"))
OR regulate OR regulation; Count: 139
Source: vox.com; Date: 2023-06-11; Query: gender OR male OR female OR transgender; Count: 140
Source: vox.com; Date: 2023-06-11; Query: security AND (social OR national); Count: 140
Source: vox.com; Date: 2023-06-11; Query: justice OR surveillance; Count: 140
Source: vox.com; Date: 2023-06-11; Query: healthcare OR "health care"; Count: 141
Source: vox.com; Date: 2023-06-11; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 143
Source: vox.com; Date: 2023-06-11; Query: (policy AND (drug OR "affirmative action"))
OR regulate OR regulation; Count: 144
Source: vox.com; Date: 2023-06-10; Query: gender OR male OR female OR transgender; Count: 144
Source: vox.com; Date: 2023-06-10; Query: security AND (social OR national); Count: 145
Source: vox.com; Date: 2023-06-10; Query: justice OR surveillance; Count: 145
Source: vox.com; Date: 2023-06-10; Query: healthcare OR "health care"; Count: 145
Source: vox.com; Date: 2023-06-10; Query: (political AND (bias OR party)) OR republican
OR democrat OR election; Count: 148

```
Source: vox.com; Date: 2023-06-10; Query: (policy AND (drug OR "affirmative
action"))
              OR regulate OR regulation; Count: 148
Source: vox.com; Date: 2023-06-09; Query: gender OR male OR female OR
transgender; Count: 148
Source: vox.com; Date: 2023-06-09; Query: security AND (social OR national);
Count: 154
Source: vox.com; Date: 2023-06-09; Query: justice OR surveillance; Count: 160
Source: vox.com; Date: 2023-06-09; Query: healthcare OR "health care"; Count:
161
Source: vox.com; Date: 2023-06-09; Query: (political AND (bias OR party)) OR
republican
              OR democrat OR election; Count: 168
Source: vox.com; Date: 2023-06-09; Query: (policy AND (drug OR "affirmative
action"))
              OR regulate OR regulation; Count: 168
```

[8]:
```python
#print(api_record_lst01)
print(len(api_record_lst01))
```

```
168
```

[9]:
```python
# Convert result list to set to eliminate duplicates
api_record_set01 = set(api_record_lst01)
#print(api_record_set01)
api_record_lst02 = list(api_record_set01)
#print(api_record_lst02)
print(len(api_record_lst02))
```

```
91
```

1.6 Initiate MySQL connection

[10]:
```python
'''Set local environment variables to hide user name & password citation:
https://www.geeksforgeeks.org/how-to-hide-sensitive-credentials-using-python/
'''
user_name = os.environ['MySQLUSRAC']
user_pass = os.environ['MySQLPWDAC']

# Instantiate connection
db_conn = mysql.connect(host='localhost',
                        port=int(3306),
                        user=user_name,
                        passwd=user_pass,
                        db='509_final_proj')

# Create a cursor object
cursor = db_conn.cursor()
```

[11]:
```
tbl_names = pd.read_sql('SHOW TABLES', db_conn)

display(tbl_names)
print(type(tbl_names))
```

```
C:\Users\acarr\AppData\Local\Temp\ipykernel_17008\4193860975.py:1: UserWarning:
pandas only supports SQLAlchemy connectable (engine/connection) or database
string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested.
Please consider using SQLAlchemy.
  tbl_names = pd.read_sql('SHOW TABLES', db_conn)
```

```
  Tables_in_509_final_proj
0                 nar_temp
1            news_articles
```

```
<class 'pandas.core.frame.DataFrame'>
```

1.6.1 Establish logging policy

[12]:
```
'''Logging citations (see additional code in following code blocks:
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/;
https://docs.python.org/3/howto/logging.html#logging-basic-example;
https://docs.python.org/3/howto/logging.html#logging-to-a-file;
https://docs.python.org/3/howto
/logging-cookbook.html#using-a-rotating-log-file-handler;
https://docs.python.org/3/howto
/logging-cookbook.html#using-a-timed-rotating-file-handler
'''

# Set up logging
logging.basicConfig(level=logging.INFO,
                    filename='pymysql.log',
                    filemode='a',
                    format='''>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>\n%(asctime)s -
                    %(levelname)s - %(message)s''')
```

1.6.2 Update individual tables

Update news_articles table from API

[13]:
```
nat_tbl_name = 'nar_temp'
nwa_tbl_name = 'news_articles'
```

[14]:
```
'''Using cursor and loading into temp file:
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/;
https://pynative.com/python-mysql-insert-data-into-database-table/
'''

# Execute query and measure execution time
start_time = time.time()
```

```
# Wipe temp table
try:
    nat_dlt_tble_stmnt = f"""DELETE FROM {nat_tbl_name}"""
    cursor.execute(nat_dlt_tble_stmnt)
    logging.info(f'''Successfully executed query:\n{nat_dlt_tble_stmnt}\n\n
    Records scanned: {cursor.rowcount}''')
except mysql.Error as e:
    logging.error(f'Error executing query:\n{nat_dlt_tble_stmnt}\n\n{e}')
finally:
    end_time = time.time()
    logging.info(f'''Time taken: {end_time - start_time:.3f} seconds\n
    >>>>>>>>>>>>>>>>>>>>>>>>>>>>\n\n''')

# Execute query and measure execution time
start_time = time.time()

# Load data from CSV file into a temporary table
try:
    nat_csv_load_stmnt = f"""
    INSERT INTO {nat_tbl_name}
    (
    source_name,
    author,
    title,
    url,
    publish_date,
    content
    )
    VALUES (%s, %s, %s, %s, %s, %s)
    """

    # Execute the query with multiple values
    cursor.executemany(nat_csv_load_stmnt, api_record_lst02)
    #cursor.execute(nat_csv_load_stmnt)
    logging.info(f'''Successfully executed query:\n{nat_csv_load_stmnt}\n\n
    Records scanned: {cursor.rowcount}''')
except mysql.Error as e:
    logging.error(f'Error executing query:\n{nat_csv_load_stmnt}\n\n{e}')
finally:
    end_time = time.time()
    logging.info(f'''Time taken: {end_time - start_time:.3f} seconds\n
    >>>>>>>>>>>>>>>>>>>>>>>>>>>>\n\n''')

# Execute query and measure execution time
start_time = time.time()
```

```
# Insert new records into main table
try:
    nwa_load_stmnt = f"""
    INSERT INTO {nwa_tbl_name}
    (
    source_name,
    author,
    title,
    url,
    publish_date,
    content
    )
    SELECT
        tp.source_name,
        tp.author,
        tp.title,
        tp.url,
        tp.publish_date,
        tp.content
    FROM {nat_tbl_name} AS tp
    LEFT JOIN {nwa_tbl_name} AS mn
    ON tp.title = mn.title
    AND CAST(LEFT(tp.publish_date,10) AS DATE)=CAST(LEFT(mn.publish_date,10) AS␣
 ↪DATE)
    AND tp.author = mn.author
    """
    cursor.execute(nwa_load_stmnt)
    logging.info(f'''Successfully executed query:\n{nwa_load_stmnt}\n\n
    Records scanned: {cursor.rowcount}''')
except mysql.Error as e:
    logging.error(f'Error executing query:\n{nwa_load_stmnt}\n\n{e}')
finally:
    end_time = time.time()
    logging.info(f'''Time taken: {end_time - start_time:.3f} seconds\n
    >>>>>>>>>>>>>>>>>>>>>>>>>>>>>\n\n''')

# Execute query and measure execution time
start_time = time.time()

# Wipe temp table
try:
    cursor.execute(nat_dlt_tble_stmnt)
    logging.info(f'''Successfully executed query:\n{nat_dlt_tble_stmnt}\n\n
    Records scanned: {cursor.rowcount}''')
except mysql.Error as e:
    logging.error(f'Error executing query:\n{nat_dlt_tble_stmnt}\n\n{e}')
finally:
```

```
    end_time = time.time()
    logging.info(f'''Time taken: {end_time - start_time:.3f} seconds\n
    >>>>>>>>>>>>>>>>>>>>>>>>>>>>>\n\n''')
```

1.6.3 Commit changes and close cursor and connection instances

[15]:
```
# Commit the changes to the database
db_conn.commit()

# Close the cursor and database connection
cursor.close()
db_conn.close()
```

1.7 References

- AllSides. (2022, March 15). *AllSides Media Bias Chart version 6: Updated ratings for NPR, Newsmax, and more*. https://www.allsides.com/blog/new-allsides-media-bias-chart-version-6-updated-ratings-nprnewsmax-and-more

- Liu, R., Jia, C., Wei, J., Xu, G., & Vosoughi, S. (2022). Quantifying and alleviating political bias in language models. *Artificial Intelligence*, 304. https://doi.org/10.1016/j.artint.2021.103654

- Ralph, P., & Relman, E. (2018, September 2). These are the most and least biased news outlets in the US, according to Americans. *Business Insider*. https://www.businessinsider.com/most-biased-news-outlets-in-america-cnn-fox-nytimes-2018-8?op=1#and-heres-how-republicans-ranked-them-from-fox-news-to-cnn-20

ADS509_CarrA_Final_Project_02_MySQL_v1

June 24, 2023

1 509 Final Project

This notebook queries the URLs from the MySQL table, scrapes the entire URL HTML contents, then inserts it into a pandas dataframe. The scraped data is then persisted in two ways: 1) The HTML content is written it back to another column in the MySQL table; 2) A copy of the full DF is written to a CSV file for further processing (i.e., processing).

1.1 Globally import libraries

[1]:
```
import numpy as np
import pandas as pd
import pymysql as mysql
import matplotlib.pyplot as plt
import os
import shutil
import re
import logging
import time
import zipfile
import requests
from bs4 import BeautifulSoup
import datetime as dt
import re
import regex as rex
from collections import defaultdict, Counter
import random
#import mysql.connector

# Set pandas global options
pd.options.display.max_rows = 17
```

1.2 Initiate MySQL connection

[2]:
```
'''Set local environment variables to hide user name & password citation:
https://www.geeksforgeeks.org/how-to-hide-sensitive-credentials-using-python/
'''
user_name = os.environ['MySQLUSRAC']
user_pass = os.environ['MySQLPWDAC']
```

```
# Instantiate connection
db_conn = mysql.connect(host='localhost',
                        port=int(3306),
                        user=user_name,
                        passwd=user_pass,
                        db='509_final_proj')

# Create a cursor object
cursor = db_conn.cursor()
```

[3]:
```
tbl_names = pd.read_sql('SHOW TABLES', db_conn)

display(tbl_names)
print(type(tbl_names))
```

```
C:\Users\acarr\AppData\Local\Temp\ipykernel_26388\4193860975.py:1: UserWarning:
pandas only supports SQLAlchemy connectable (engine/connection) or database
string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested.
Please consider using SQLAlchemy.
  tbl_names = pd.read_sql('SHOW TABLES', db_conn)
```

	Tables_in_509_final_proj
0	nar_temp
1	news_articles

```
<class 'pandas.core.frame.DataFrame'>
```

1.2.1 Establish logging policy

[4]:
```
'''Logging citations (see additional code in following code blocks:
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/;
https://docs.python.org/3/howto/logging.html#logging-basic-example;
https://docs.python.org/3/howto/logging.html#logging-to-a-file;
https://docs.python.org/3/howto
/logging-cookbook.html#using-a-rotating-log-file-handler;
https://docs.python.org/3/howto
/logging-cookbook.html#using-a-timed-rotating-file-handler
'''
# Set up logging
logging.basicConfig(level=logging.INFO,
                    filename='pymysql.log',
                    filemode='a',
                    format='''>>>>>>>>>>>>>>>><<<<<<<<<<<<<<<<\n%(asctime)s -
                    %(levelname)s - %(message)s''')
```

1.2.2 Read URLs from MySQL table to perform web scraping

[5]:
```
nat_tbl_name = 'nar_temp'
nwa_tbl_name = 'news_articles'
```

[6]:
```
'''Connect to MqSQL table in batches citation:
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/
'''

# Batch size (number of URLs to process at a time)
batch_size = 10000

# Get the total number of URLs in the table
count_query = f"SELECT COUNT(*) FROM {nwa_tbl_name}"
cursor.execute(count_query)
total_urls = cursor.fetchone()[0]
print(f'URL Count: {total_urls}')

# Start timer
start_time = dt.datetime.today()

# Calculate the number of batches required
num_batches = (total_urls // batch_size) + 1

# Process URLs in batches
for batch in range(num_batches):
    offset = batch * batch_size

    # Retrieve URLs from the MySQL table in the current batch
    query = f'''
    SELECT url FROM {nwa_tbl_name}
    WHERE article_text IS NULL
    AND (source_name="CNN"
        OR source_name="The Washington Post"
        OR source_name="Fox News"
        OR source_name="Slate Magazine"
        OR source_name="Vox"
        OR source_name="Breitbart News")
    LIMIT {batch_size}
    OFFSET {offset}
    '''

    print(query)
    cursor.execute(query)
    urls = cursor.fetchall()
    print(f'URL batch size: {len(urls)}')
```

```
    # Iterate over the URLs and scrape their contents
    for url in urls:
        url = url[0]  # Extract the URL from the tuple

        # Make an HTTP request to the URL
        response = requests.get(url)
        time.sleep(5 + 11 * random.random())

        # Check if the request was successful
        if response.status_code == 200:
            # Parse the HTML content using BeautifulSoup
            soup = BeautifulSoup(response.content, 'html.parser')

            # Extract the raw text from the HTML
            #print(soup.prettify())
            #raw_text = soup.get_text()
            raw_text = soup.prettify()


            # Update the MySQL table with the scraped text
            update_query = '''
            UPDATE news_articles SET article_text = %s
            WHERE url = %s
            '''

            print('.', end='')
            #print(update_query)
            cursor.execute(update_query, (raw_text, url))
            db_conn.commit()
        else:
            print(f'Response: {response.status_code}')

# End timer script
end_time = dt.datetime.today()
time_elapse = end_time - start_time
print(f'Start Time = {start_time}')
print(f'End Time = {end_time}')
print(f'Elapsed Time = {time_elapse}')
```

```
URL Count: 6283
```

```
    SELECT url FROM news_articles
    WHERE article_text IS NULL
    AND (source_name="CNN"
        OR source_name="The Washington Post"
        OR source_name="Fox News"
        OR source_name="Slate Magazine"
        OR source_name="Vox"
```

```
        OR source_name="Breitbart News")
    LIMIT 10000
    OFFSET 0

URL batch size: 41
Response: 404
Response: 404
…Start Time = 2023-06-18 00:56:01.517790
End Time = 2023-06-18 01:15:26.593576
Elapsed Time = 0:19:25.075786
```

1.2.3 Send MySQL records to CSV

[7]:
```
slct_tbl_full_df01 = pd.read_sql(
                                 '''
                                 SELECT * FROM news_articles
                                 WHERE article_text IS NOT NULL
                                     AND (source_name="CNN"
                                     OR source_name="The Washington Post"
                                     OR source_name="Fox News"
                                     OR source_name="Slate Magazine"
                                     OR source_name="Vox"
                                     OR source_name="Breitbart News")
                                 ''',
                                 db_conn)
```

```
C:\Users\acarr\AppData\Local\Temp\ipykernel_26388\1187134288.py:1: UserWarning:
pandas only supports SQLAlchemy connectable (engine/connection) or database
string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested.
Please consider using SQLAlchemy.
  slct_tbl_full_df01 = pd.read_sql(
```

[8]:
```
'''Dir nav citation:
https://softhints.com/python-change-directory-parent/
'''
curr_dir = os.path.abspath(os.curdir)
print(curr_dir)
os.chdir("..")
up1_dir = os.path.abspath(os.curdir)
print(up1_dir)
```

```
C:\Users\acarr\Documents\GitHub\ADS509_Final_project\deliverables
C:\Users\acarr\Documents\GitHub\ADS509_Final_project
```

[9]:
```
# change `data_location` to the location of the folder on your machine.
data_location = 'data_restricted'

file_name = 'data_raw_amc.csv'
```

```
file_path = os.path.join(up1_dir, data_location, file_name)

print(f'CSV file path: {file_path}')
```

```
CSV file path: C:\Users\acarr\Documents\GitHub\ADS509_Final_project\data_restric
ted\data_raw_amc.csv
```

[10]:
```
slct_tbl_full_df01.to_csv(file_path, index=False)
```

[11]:
```
print(type(slct_tbl_full_df01))
display(slct_tbl_full_df01.head(11))
#display(slct_tbl_full_df01['article_text'][0])
```

```
<class 'pandas.core.frame.DataFrame'>
```

	text_id	source_name	author	title	url	publish_date
0	1	CNN	Clare Foran,Nicky Robertson	Senate races to avert default but vote timing …	https://www.cnn.com/2023/06/01/politics/senate…	2023-06-01T09:00:40Z
1	2	Fox News	Paul Steinhauser	First on Fox: Pro-Tim Scott super PAC launches…	https://www.foxnews.com/politics/pro-scott-sup…	2023-06-01T16:12:56Z
2	3	Fox News	Greg Wehner	Pennsylvania bus driver allegedly used duct ta…	https://www.foxnews.com/us/pennsylvania-bus-dr…	2023-05-31T00:21:20Z
3	4	Fox News	Michael Ruiz	Bob Lee murder: Cash App founder seen with sus…	https://www.foxnews.com/us/bob-lee-murder-cash…	2023-05-31T20:30:12Z
4	5	Fox News	Brooke Singman	Senate GOP demands answers on security clearan…	https://www.foxnews.com/politics/senate-gop-de…	2023-05-31T22:11:38Z
5	6	Fox News	Peter Kasperowicz	Will AI ever be smart enough to decipher feder…	https://www.foxnews.com/politics/ai-smart-enou…	2023-06-01T06:00:30Z
6	7	Fox News	Fox News Staff	SEAN HANNITY: Here's what you need to know abo…		
7	8	Fox News	Melissa Rudy	Ozempic, Wegovy and pregnancy risk: What you n…		
8	9	Fox News	Associated Press	NATO ramps up pressure on Turkey to drop objec…		
9	10	Fox News	Fox News Staff	Mike Lee goes off on Biden-McCarthy debt ceili…		
10	11	Fox News	Associated Press	Indiana police officer, suspect hospitalized f…		

```
6   https://www.foxnews.com/media/sean-hannity-her…  2023-05-31T02:53:55Z
7   https://www.foxnews.com/health/ozempic-wegovy-…  2023-06-01T15:58:10Z
8   https://www.foxnews.com/world/nato-ramps-press…  2023-06-01T16:54:03Z
9   https://www.foxnews.com/media/mike-lee-goes-bi…  2023-06-01T17:00:26Z
10  https://www.foxnews.com/us/southern-indiana-po…  2023-05-31T12:39:31Z

                                          article_text content
0   <!DOCTYPE html>\n<html data-layout-uri="cms.cn…    None
1   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
2   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
3   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
4   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
5   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
6   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
7   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
8   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
9   <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
10  <!DOCTYPE html>\n<html data-n-head="%7B%22lang…    None
```

1.2.4 Commit changes and close cursor and connection instances

[12]:
```
# Commit the changes to the database
db_conn.commit()

# Close the cursor and database connection
cursor.close()
db_conn.close()
```

ADS509_CarrA_Final_Project_03_Query_v1

June 24, 2023

1 509 Final Project

This notebook reads in a CSV file that contain an attribute with raw HTML, then parses the article text using Beautiful Soup. A copy of the full DF is written to a CSV file for sharing with all colloborators in GitHub.

1.1 Globally import libraries

[1]:
```
import numpy as np
import pandas as pd
import pymysql as mysql
import matplotlib.pyplot as plt
import os
import shutil
import re
import logging
import time
import datetime as dt
import zipfile
import requests
from bs4 import BeautifulSoup
import datetime
import re
import regex as rex
from collections import defaultdict, Counter
import random
import json
#import mysql.connector

# Set pandas global options
pd.options.display.max_rows = 17
```

1.2 Upload data from CSV

[2]:
```
'''Dir nav citation:
https://softhints.com/python-change-directory-parent/'''
curr_dir = os.path.abspath(os.curdir)
print(curr_dir)
```

```
os.chdir("..")
up1_dir = os.path.abspath(os.curdir)
print(up1_dir)
```

```
C:\Users\acarr\Documents\GitHub\ADS509_Final_project\deliverables
C:\Users\acarr\Documents\GitHub\ADS509_Final_project
```

[3]:
```
# change `data_location` to the location of the folder on your machine.
data_r_location = 'data_restricted'
data_location = 'data'

file_in_name = 'data_raw_amc.csv'
file_out_name = 'data_parsed_amc.csv'

file_in_path01 = os.path.join(up1_dir, data_r_location, file_in_name)
file_out_path = os.path.join(up1_dir, data_location, file_out_name)

print(f'CSV file in path: {file_in_path01}')
print(f'CSV file out path: {file_out_path}')
```

```
CSV file in path: C:\Users\acarr\Documents\GitHub\ADS509_Final_project\data_rest
ricted\data_raw_amc.csv
CSV file out path:
C:\Users\acarr\Documents\GitHub\ADS509_Final_project\data\data_parsed_amc.csv
```

[4]:
```
slct_tbl_full_df01 = pd.read_csv(file_in_path01)
display(slct_tbl_full_df01.head())
```

```
   text_id source_name                      author  \
0        1         CNN  Clare Foran,Nicky Robertson
1        2    Fox News             Paul Steinhauser
2        3    Fox News                  Greg Wehner
3        4    Fox News                Michael Ruiz
4        5    Fox News               Brooke Singman

                                              title  \
0  Senate races to avert default but vote timing …
1  First on Fox: Pro-Tim Scott super PAC launches…
2  Pennsylvania bus driver allegedly used duct ta…
3  Bob Lee murder: Cash App founder seen with sus…
4  Senate GOP demands answers on security clearan…

                                               url          publish_date  \
0  https://www.cnn.com/2023/06/01/politics/senate…  2023-06-01T09:00:40Z
1  https://www.foxnews.com/politics/pro-scott-sup…  2023-06-01T16:12:56Z
2  https://www.foxnews.com/us/pennsylvania-bus-dr…  2023-05-31T00:21:20Z
3  https://www.foxnews.com/us/bob-lee-murder-cash…  2023-05-31T20:30:12Z
4  https://www.foxnews.com/politics/senate-gop-de…  2023-05-31T22:11:38Z
```

```
                                              article_text content
0  <!DOCTYPE html>\n<html data-layout-uri="cms.cn…     NaN
1  <!DOCTYPE html>\n<html data-n-head="%7B%22lang…     NaN
2  <!DOCTYPE html>\n<html data-n-head="%7B%22lang…     NaN
3  <!DOCTYPE html>\n<html data-n-head="%7B%22lang…     NaN
4  <!DOCTYPE html>\n<html data-n-head="%7B%22lang…     NaN
```

1.3 Extract article data

1.3.1 Check for missing articles

[5]:
```
count_nan = slct_tbl_full_df01['article_text'].isnull().sum()

# printing the number of values present
# in the column
print('Number of NaN values present: ' + str(count_nan))
```

```
Number of NaN values present: 0
```

1.3.2 Parse the article text from the column with raw HTML

[6]:
```
slct_tbl_full_df02 = slct_tbl_full_df01.copy()
#slct_tbl_full_df02 = slct_tbl_full_df01.sort_values(by=['source_name'])
#slct_tbl_full_df02 = slct_tbl_full_df02.reset_index()

print(f'DF instances: {len(slct_tbl_full_df02)}')

slct_tbl_full_df02['article_parsed'] = ''

total_urls = len(slct_tbl_full_df02)

# Start timer
start_time = dt.datetime.today()

# Use multiple parsing methods to extract the article from raw HTML
for i, row in enumerate(slct_tbl_full_df02.itertuples(), 1):
    #print(f'Enumeration #: {i}')
    #print(row[7])
    soup = BeautifulSoup(row[7], 'html.parser')

    # Check for available JSON object
    try:
        script_tag = soup.find('script', {'type': 'application/ld+json'})
        if script_tag == None:
            json_err01 = f'''JSON Object = None: Index: {i-1}; source: {row[2]};
            URL: {row[5]}'''
        else:
            article_json = json.loads(script_tag.string)
```

```
            article_content = article_json['articleBody']
            slct_tbl_full_df02.at[row.Index, 'article_parsed'] = article_content

    # If no JSON object available, use Beautiful Soup to look for available
    # HTML tags
    except TypeError:
        print(f'Type Error')

    except KeyError:
        json_err02 = f'''Missing JSON key: Index: {i-1}; source: {row[2]};
        URL: {row[5]}'''
        article_body = soup.find('div', class_='article__content-container')

        if article_body is None: #forfoxandbreitbert(sometimes)
            #print('Class != article__content-container')
            article_body = soup.find('p', class_="speakable")
            if article_body is None: #breitbert(most)
                #print('Class != speakable')
                article_body = soup.find('div', class_='entry-content')
                if article_body is None: #WashPost
                    #print('Class != entry-content')
                    article_body = soup.find('div',class_='article-body')

        if article_body is not None:
            article_text = article_body.get_text()
            slct_tbl_full_df02.at[row.Index, 'article_parsed'] = article_text
            #print('Rejoice, parse was successful!')
        else:
            print('\nParse not successful...')
            try:
                print(json_err02)
            except:
                pass

    print('.', end='')

# End timer script
end_time = dt.datetime.today()
time_elapse = end_time - start_time
print(f'Start Time = {start_time}')
print(f'End Time = {end_time}')
print(f'Elapsed Time = {time_elapse}')
```

```
DF instances: 6153
…
Parse not successful…
Missing JSON key: Index: 64; source: CNN;
        URL: https://www.cnn.com/politics/live-news/us-debt-ceiling-deadline-
```

```
talks-05-31-23/index.html
…
Parse not successful…
Missing JSON key: Index: 87; source: CNN;
        URL: https://www.cnn.com/politics/live-news/us-debt-ceiling-senate-
vote-06-01-23/index.html
…
…
…
…
Parse not successful…
Missing JSON key: Index: 407; source: CNN;
        URL: https://www.cnn.com/politics/live-news/us-debt-ceiling-deadline-
talks-05-30-23/index.html
…
…
…
…
Parse not successful…
Missing JSON key: Index: 665; source: CNN;
        URL: https://www.cnn.com/europe/live-news/russia-ukraine-war-
news-05-06-23/index.html
…
…
…
…
Parse not successful…
Missing JSON key: Index: 948; source: The Washington Post;
        URL: https://www.washingtonpost.com/dc-md-va/interactive/2023/proud-
boys-trial-timeline-jan6-videos-chats/
…
…
…
Parse not successful…
Missing JSON key: Index: 1122; source: CNN;
        URL: https://www.cnn.com/us/live-news/brownsville-texas-car-
crash-05-08-23/index.html
…
…
…
Parse not successful…
Missing JSON key: Index: 1341; source: CNN;
        URL: https://www.cnn.com/politics/live-news/george-santos-federal-
charges-05-10-23/index.html
…
…
…
…
```

…
…
…
…
…
…
Parse not successful…
Missing JSON key: Index: 2065; source: The Washington Post;
        URL: https://www.washingtonpost.com/nation/interactive/2023/texas-
title-42-end/
…
Parse not successful…
Missing JSON key: Index: 2101; source: CNN;
        URL: https://www.cnn.com/politics/live-news/trump-cnn-town-
hall/index.html
…
…
…
Parse not successful…
Missing JSON key: Index: 2290; source: CNN;
        URL: https://www.cnn.com/politics/live-news/trump-russia-probe-durham-
report/index.html
…
…
…
Parse not successful…
Missing JSON key: Index: 2513; source: CNN;
        URL: https://www.cnn.com/europe/live-news/russia-ukraine-war-
news-05-16-23/index.html
…
…
…
…
…
…
…
Parse not successful…
Missing JSON key: Index: 3040; source: CNN;
        URL: https://www.cnn.com/politics/live-news/biden-mccarthy-meeting-debt-
ceiling-05-16-23/index.html
…
…
…
Parse not successful…
Missing JSON key: Index: 3225; source: CNN;
        URL: https://www.cnn.com/europe/live-news/russia-ukraine-war-
news-06-03-23/index.html
…

…
…
…
…
Parse not successful…
Missing JSON key: Index: 3566; source: The Washington Post;
        URL: https://www.washingtonpost.com/elections/candidates/chris-
christie-2024/
…
Parse not successful…
Missing JSON key: Index: 3611; source: The Washington Post;
        URL: https://www.washingtonpost.com/elections/candidates/mike-
pence-2024/
…
Parse not successful…
Missing JSON key: Index: 3650; source: CNN;
        URL: https://www.cnn.com/politics/live-news/nikki-haley-cnn-town-
hall/index.html
…
…
…
Parse not successful…
Missing JSON key: Index: 3839; source: The Washington Post;
        URL: https://www.washingtonpost.com/elections/candidates/doug-
burgum-2024/
…
…
…
Parse not successful…
Missing JSON key: Index: 4037; source: CNN;
        URL: https://www.cnn.com/politics/live-news/mike-pence-cnn-town-
hall/index.html
…
…
…
…
Parse not successful…
Missing JSON key: Index: 4337; source: CNN;
        URL: https://www.cnn.com/politics/live-news/mar-a-lago-documents-probe-
latest/index.html
…
Parse not successful…
Missing JSON key: Index: 4414; source: CNN;
        URL: https://www.cnn.com/politics/live-news/trump-indictment-
documents-06-11-23/index.html
…
…
…

```
Parse not successful…
Missing JSON key: Index: 4622; source: CNN;
        URL: https://www.cnn.com/audio/podcasts/one-
thing/episodes/3c097086-458f-47fb-baa7-b01d00348cce
…
…
Parse not successful…
Missing JSON key: Index: 4753; source: The Washington Post;
        URL: https://www.washingtonpost.com/politics/interactive/2023/door-
county-bellwether-politics/
…
Parse not successful…
Missing JSON key: Index: 4812; source: CNN;
        URL: https://www.cnn.com/politics/live-news/trump-indictment-classified-
documents-06-09-23/index.html
…
…
…
…
Parse not successful…
Missing JSON key: Index: 5106; source: CNN;
        URL: https://www.cnn.com/politics/live-news/donald-trump-indictment-
court-appearance-06-13-23/index.html
…
…
Parse not successful…
Missing JSON key: Index: 5212; source: CNN;
        URL: https://www.cnn.com/politics/live-news/trump-indictment-
documents-06-12-23/index.html
…
…
Parse not successful…
Missing JSON key: Index: 5371; source: CNN;
        URL: https://www.cnn.com/politics/live-news/chris-christie-town-
hall/index.html
…
…
…
…
…
…
…
…
…
…
Parse not successful…
Missing JSON key: Index: 6114; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/neil-gorsuch-so-good-
```

```
native-americans-scotus.html
.
Parse not successful…
Missing JSON key: Index: 6115; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/the-slatest-june-
sixteenth.html
.
Parse not successful…
Missing JSON key: Index: 6116; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/amy-coney-barrett-
supreme-court-native-win.html
.
Parse not successful…
Missing JSON key: Index: 6117; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/mccarthy-house-
republicans-freedom-caucus-budget-shutdown.html
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6119; source: Slate Magazine;
        URL: https://slate.com/business/2023/06/cava-stock-ipo-investors-fast-
casual-mediterranean.html
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6121; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/daniel-ellsberg-dead-
pentagon-papers-vietnam-war.html
.
Parse not successful…
Missing JSON key: Index: 6122; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/iran-us-informal-
nuclear-talks-what-it-means.html
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6124; source: Slate Magazine;
        URL: https://slate.com/business/2023/06/francis-suarez-president-
republicans-miami-crypto.html
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6126; source: Slate Magazine;
        URL: https://slate.com/human-interest/2023/06/supreme-court-affirmative-
action-decisions-race.html
.Type Error
.
Parse not successful…
```

```
Missing JSON key: Index: 6128; source: Slate Magazine;
        URL: https://slate.com/culture/2023/06/cormac-mccarthy-dead-garbage-el-
paso-texas.html
.
Parse not successful…
Missing JSON key: Index: 6129; source: Slate Magazine;
        URL: https://slate.com/culture/2023/06/black-mirror-season-6-joan-is-
awful-netflix.html
.
Parse not successful…
Missing JSON key: Index: 6130; source: Slate Magazine;
        URL: https://slate.com/human-interest/2023/06/pride-protests-
republicans-protests-muslims-michigan.html
.Type Error
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6133; source: Slate Magazine;
        URL: https://slate.com/podcasts/amicus/2023/06/justice-barretts-indian-
child-welfare-act-indigenous-rights
.Type Error
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6136; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/missouri-anti-trans-
legislation-judaism-testimony.html
.Type Error
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6139; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/the-slatest-news-and-
politics-newsletter-catch-up-on-slates-top-stories-from-wednesday-june-15.html
.Type Error
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6142; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/democrats-supreme-
court-survey-voters-mad.html
.
Parse not successful…
Missing JSON key: Index: 6143; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/supreme-court-donald-
trump-jack-smith-florida.html
.Type Error
.Type Error
```

```
.
Parse not successful…
Missing JSON key: Index: 6146; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/trump-classified-
documents-case-indictment-dangers-questions.html
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6148; source: Slate Magazine;
        URL: https://slate.com/news-and-politics/2023/06/target-starbucks-pride-
threats.html
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6150; source: Slate Magazine;
        URL: https://slate.com/podcasts/political-gabfest/2023/06/aileen-cannon-
presides-in-a-donald-trump-case-again-political-gabfest
.Type Error
.
Parse not successful…
Missing JSON key: Index: 6152; source: Slate Magazine;
        URL: https://slate.com/podcasts/the-waves/2023/06/menstruation-how-
white-supremacy-and-the-patriarchy-ruined-a-very-normal-time-of-the-month
.Start Time = 2023-06-18 14:01:20.481350
End Time = 2023-06-18 14:08:36.000522
Elapsed Time = 0:07:15.519172
```

[7]:
```
# Review sample
display(slct_tbl_full_df02.iloc[6119])
#display(slct_tbl_full_df02.iloc[6119]['article_text'])
```

```
text_id                                                      8413
source_name                                        Slate Magazine
author                                               Nitish Pahwa
title           Wall Street’s Newest Darling Is an Unprofitabl…
url             https://slate.com/business/2023/06/cava-stock-…
publish_date                                 2023-06-16T22:22:57Z
article_text    <!DOCTYPE html>\n<html data-env="prod" data-la…
content         Stock markets aint what they used to be. Since…
article_parsed
Name: 6119, dtype: object
```

[8]:
```
display(slct_tbl_full_df02[slct_tbl_full_df02['article_parsed']==''].head(11))
print(len(slct_tbl_full_df02[slct_tbl_full_df02['article_parsed']==''])
```

```
     text_id          source_name  \
64         65                  CNN
74         75             Fox News
87         88                  CNN
```

```
255       257                  CNN
407       566                  CNN
541       760            Fox News
665      1173                  CNN
948      1456  The Washington Post
1122     1737                  CNN
1341     2025                  CNN
2065     2749  The Washington Post

                                                  author  \
64    By <a href="/profiles/adrienne-vogt">Adrienne …
74                                          Sofie Watson
87    By <a href="/profiles/maureen-chowdhury">Maure…
255                                                  NaN
407   By Mike Hayes and <a href="/profiles/maureen-c…
541                                 Danielle Tuntigian
665                                    By Sophie Tanno
948                                      Adriana Usero
1122  By <a href="/profiles/aditi-sandal">Aditi Sang…
1341  By <a href="/profiles/adrienne-vogt">Adrienne …
2065   Arelis Hernández, Whitney Shefte, Jabin Botsford

                                                   title  \
64               The latest on the US debt ceiling deal
74    FOX NEWS CHANNEL TO DEBUT NEW WEEKEND PRIMETIM…
87               The latest on the US debt ceiling deal
255                         UPDATE: WAR ON GAY PRIDE…
407              The latest on the US debt ceiling deal
541   SEN. JONI ERNST AND SEN. JEANNE SHAHEEN TO PAR…
665                             Russia's war in Ukraine
948   Proud Boys revealed: Videos, secret chats show…
1122  8 killed when driver plows into crowd outside …
1341             George Santos charged in federal probe
2065         Texas uses aggressive tactics to arrest…

                                                     url          publish_date  \
64    https://www.cnn.com/politics/live-news/us-debt…  2023-05-31T12:35:50Z
74    https://press.foxnews.com/2023/06/fox-news-cha…  2023-06-01T18:33:43Z
87    https://www.cnn.com/politics/live-news/us-debt…  2023-06-01T13:46:13Z
255   https://view.newsletters.cnn.com/messages/1685…  2023-06-01T21:55:37Z
407   https://www.cnn.com/politics/live-news/us-debt…  2023-05-30T14:12:09Z
541   https://press.foxnews.com/2023/05/sen-joni-ern…  2023-05-30T17:07:24Z
665   https://www.cnn.com/europe/live-news/russia-uk…  2023-05-06T09:46:06Z
948   https://www.washingtonpost.com/dc-md-va/intera…  2023-05-05T16:00:23Z
1122  https://www.cnn.com/us/live-news/brownsville-t…  2023-05-08T14:28:31Z
1341  https://www.cnn.com/politics/live-news/george-…  2023-05-10T12:52:24Z
2065  https://www.washingtonpost.com/nation/interact…  2023-05-10T23:25:36Z
```

```
                                                article_text  \
64    <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
74    <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
87    <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
255   <!DOCTYPE html>\n<html lang="en" xmlns:o="urn:…
407   <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
541   <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
665   <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
948   <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
1122  <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
1341  <!DOCTYPE html>\n<html lang="en">\n <head>\n  …
2065  <!DOCTYPE html>\n<html lang="en">\n <head>\n  …

                                                content article_parsed
64                                                  NaN
74                                                  NaN
87                                                  NaN
255                                                 NaN
407                                                 NaN
541                                                 NaN
665   Russia has been thwarting US-made mobile rocke…
948   The man on the left in the video below alleged…
1122  Editors Note: This story contains graphic desc…
1341  Rep. George Santos, the freshman congressman w…
2065  BRACKETTVILLE, Tex. \r\nThere was something ab…

75
```

[9]:
```
count_p_nan = slct_tbl_full_df02['article_parsed'].isnull().sum()

# printing the number of values present
# in the column
print('Number of NaN values present: ' + str(count_p_nan))
```

```
Number of NaN values present: 0
```

1.3.3 Write datafrme to CSV

[10]:
```
slct_tbl_full_df03 = slct_tbl_full_df02[['source_name',
                                         'author',
                                         'title',
                                         'url',
                                         'publish_date',
                                         'article_parsed']]
```

slct_tbl_full_df03.to_csv(file_out_path, index=False)

Data Collection

API and WEBSCRAPE articles

These articles will be used to:

- TRAIN and TEST MODEL
  - from CNN, FOX, BREITBERT, and WASHINGTON POST
- CLASSIFY Stakeholder's Data using trained MODEL
  - from THE HILL

In [ ]:

```
#! pip install pyLDavis

import pyLDAvis

pyLDAvis.enable_notebook()
from tqdm.auto import tqdm

import pyLDAvis.lda_model
import pyLDAvis.gensim_models

import numpy as np
import pandas as pd
import pymysql as mysql
import matplotlib.pyplot as plt
import os
import shutil
import re
import logging
import time
import zipfile
import requests
from bs4 import BeautifulSoup
import datetime
import re
import regex as rex
from collections import defaultdict, Counter
import random
import requests
from bs4 import BeautifulSoup
import datetime
import json
from wordcloud import WordCloud
from tabulate import tabulate
from sklearn.svm import SVC


import sqlite3
import nltk
from string import punctuation
from nltk.corpus import stopwords
import re
import emoji
from nltk.metrics import ConfusionMatrix
import itertools
import collections

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.decomposition import NMF
```

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, f1_score, recall_score, precision_recall_curve
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
## Depracated:
# from sklearn.metrics import plot_confusion_matrix
## New version:
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.svm import LinearSVC
from sklearn.decomposition import LatentDirichletAllocation

#import mysql.connector

# Set pandas global options
pd.options.display.max_rows = 17

import warnings
warnings.filterwarnings('ignore')
warnings.simplefilter('ignore')
```

Connect to NewsAPI client

In [ ]:

```
from newsapi import NewsApiClient

api_key = os.environ['NewsAPIKey']

# Init
newsapi = NewsApiClient(api_key=api_key)
```

Pull article URLs and info from API

/v2/top-headlines/sources

sources = newsapi.get_sources() print(sources)

In [ ]:

```
def news_api_urls(q=None,
                  s=None,
                  d_from='2023-05-01',
                  d_to='2023-05-31',
                  api_lst=[]):
    all_articles = newsapi.get_everything(q=q,
                                          sources=s,
                                          from_param=d_from,
                                          to=d_to,
                                          language='en',
                                          sort_by='relevancy',
                                          page=1)

    print(type(all_articles))
    print(all_articles)
    #print('Article list: ', all_articles['articles'])
    for article in all_articles['articles']:
        print('Source ID:', article['source']['id'])
        print('Source name:', article['source']['name'])
        print('Author:', article['author'])
        print('Title:', article['title'])
        print('URL:', article['url'])
```

```
        print('Publish date:', article['publishedAt'])
        print('Article text:', article['content'], '\n')

    # Create a list of tuples from the dictionary data
    source_data01 = [(a['source']['name'],
                      a['author'],
                      a['title'],
                      a['url'],
                      a['publishedAt'],
                      a['content'])
                     for a in all_articles['articles']]

    api_lst.extend(source_data01)
    #print(api_lst)
    print(len(api_lst))
```

Data Collection:

Connect to API to access URLs

Set API filter parameters

Extract Training Data for Classifier

Two Left leaning and Two Right leaning news sources are queried, collected, cleaned, and prepared for model training and test

In [ ]:
```
# Parameters to extract API News data from 4 sources:

source_lst = ['the-washington-post','cnn','fox-news','breitbart-news']


date_lst = ['2023-05-18', '2023-05-19', '2023-05-21', '2023-05-22', '2023-05-23']


q_word_lst = ['justice OR surveillance', 'healthcare OR "health care"',
              '(political AND (bias OR party)) OR republican OR democrat OR election',
              'security AND (social OR national)', '(policy AND (drug OR "affirmative acti
```

Extract Client Data for Business Application of our Classifer

Data from what is considered a Centered news source is extracted, preprocessed, and prepared to run through our trained Classifier. We report back the data gathered, the overall lean found in that data, and give details on the lean, if a political lean is found in the data.

In [ ]:
```
# Parameters to extract API News data from 1 centered source:

source_lst = ['associated-press']


date_lst = ['2023-05-18', '2023-05-19', '2023-05-20', '2023-05-21', '2023-05-22', '2023-05
            '2023-05-24', '2023-05-25', '2023-05-17', '2023-05-16', '2023-05-15', '2023-05
            '2023-05-26', '2023-05-27', '2023-05-28', '2023-05-29', '2023-05-30', '2023-05
            '2023-05-13', '2023-05-12']

#, '2023-05-11', '2023-05-10', '2023-05-09', '2023-05-08']

q_word_lst = ['justice OR surveillance', 'healthcare OR "health care"',
              '(political AND (bias OR party)) OR republican OR democrat OR election',
              'security AND (social OR national)', '(policy AND (drug OR "affirmative acti
```

In [ ]:
```
# Parameters to extract API News data from 1 centered source:

source_lst = ['the-hill']


date_lst = ['2023-05-18', '2023-05-19', '2023-05-20', '2023-05-21', '2023-05-22', '2023-05
            '2023-05-24', '2023-05-25', '2023-05-17', '2023-05-16', '2023-05-15', '2023-05
            '2023-05-26', '2023-05-27', '2023-05-28', '2023-05-29', '2023-05-30', '2023-05
#date_lst = ['2023-06-01','2023-06-02','2023-06-03','2023-06-04','2023-06-05','2023-06-06
#            '2023-06-08','2023-06-09','2023-06-10','2023-06-11','2023-06-12','2023-06-13
#            '2023-06-15']

q_word_lst = ['justice OR surveillance', 'healthcare OR "health care"',
              '(political AND (bias OR party)) OR republican OR democrat OR election',
              'security AND (social OR national)', '(policy AND (drug OR "affirmative acti
```

Access API (collect URLs)

Using the defined parameters above, extract URL and other data information from the desired news sources.

In [1]:
```
api_record_lst01 = []
for s in source_lst:
    print(f'Source: {s}')
    for d in date_lst:
        print(f'Date: {d}')
        for q in q_word_lst:
            print(f'Query word: {q}')
            time.sleep(5 + 11 * random.random())
            news_api_urls(q=q,
                          s=s,
                          d_from=d,
                          d_to=d,
                          api_lst=api_record_lst01)
            print(s, d, q)
    time.sleep(10 + 13 * random.random())

#print(api_record_lst01)
#print(len(api_record_lst01))
```

Save into DataFrames:

Four sources data (used to TRAIN and TEST the MODEL):

In [ ]:
```
api_record_df2 = pd.DataFrame (list (api_record_lst01),
                               columns = ['Source','Author','Title','URL','date','cont
```

Centered Client data (stakeholder's data that will be CLASSIFIED using above mentioned MODEL):

In [ ]:
```
api_record_df = pd.DataFrame (list (api_record_lst01),
                              columns = ['Source','Author','Title','URL','date','cont
#api_record_df.to_csv("News_API_Reuters2.csv", sep=',')
api_record_df.to_csv("News_API_TheHill3.csv", sep=',')
```

In [ ]:
```
api_record_df.shape
```

```
(69, 6)
```

Code to combine different dataframes for training (my portion): (no need to run again)

In [ ]:
```
api_record_df_final = pd.concat([api_record_df2, api_record_df])
```

In [ ]:
```
api_record_df_final.to_csv("News_API_FOX_CNN_Breitbert_May18_23_May31.csv", sep=',')
api_record_df.to_csv("News_API_FOX_CNN_Breitbert_May20_May31.csv", sep=',')
api_record_df2.to_csv("News_API_FOX_CNN_Breitbert_May1819_May2123.csv", sep=',')
```

In [ ]:
```
api_record_df_filtered = api_record_df[api_record_df.Source != 'The Washington Post']
```

In [ ]:
```
api_data_df3=pd.read_csv('News_API_FOX_CNN_Breitbert_May18_23_May31_2.csv')
```

In [ ]:
```
api_record_df_final3 = api_data_df3[api_data_df3.Source != 'The Washington Post']
```

In [ ]:
```
api_record_df_final3.to_csv("News_API_FOX_CNN_Breitbert_May18_23_May31_3.csv", sep=',')
api_record_df_final3.shape
api_record_df_filtered
api_record_df2.shape
```

In [ ]:
```
api_record_df_final2 = pd.concat([api_record_df2, api_record_df_filtered])
api_record_df_final2.shape
```

In [ ]:
```
api_record_df_final2.to_csv("News_API_FOX_CNN_Breitbert_May18_23_May31_2.csv", sep=',')
api_data_df_final3=pd.read_csv('News_API_FOX_CNN_Breitbert_May18_23_May31_2.csv')
```

HTML scrape of URLs (to collect Article content)

FOX, CNN, Breitbert:

In [ ]:
```
slct_tbl_full_df02 = api_record_df_final3.copy()



slct_tbl_full_df02['article_parsed'] = ''

total_urls = len(slct_tbl_full_df02)

# Start timer
start_time = datetime.date.today()


for i, row in enumerate(slct_tbl_full_df02.itertuples(), 1):
    print(row[5])
    article=[]
    #url = slct_tbl_full_df02.at[i,'URL']
    url=row[5]
    response = requests.get(url)
    time.sleep(5 + 10*random.random())

    soup = BeautifulSoup(response.text, 'html.parser')

    try:
        script_tag = soup.find('script', {'type': 'application/ld+json'})
        if script_tag == None:
            print('\nJSON Object == None: Index:', i-1, '; source:',
```

```
                    row[2],
                    '; URL:', row[5])
        else:
            article_json = json.loads(script_tag.string)
            article_content = article_json['articleBody']
            slct_tbl_full_df02.at[row.Index, 'article_parsed'] = article_content

    except KeyError:
        print('\nMissing JSON key: Index:', i-1, '; source:',
              row[2],
              '; URL:', row[5])
        article_body = soup.find('div', class_='article__content-container')

        if article_body is None: #forfoxandbreitbert(sometimes)
            print('Class != article__content-container')
            article_body = soup.find('p', class_="speakable")
            if article_body is None: #breitbert(most)
                print('Class != speakable')
                article_body = soup.find('div', class_='entry-content')
                if article_body is None: #WashPost
                    print('Class != entry-content')
                    article_body = soup.find('div',class_='article-body')

        if article_body is not None:
            article_text = article_body.get_text()
            slct_tbl_full_df02.at[row.Index, 'article_parsed'] = article_text
        else:
            print('Could not parse')

    print('.', end='')

# End timer script
end_time = datetime.date.today()
time_elapse = end_time - start_time
print(f'Start Time = {start_time}')
print(f'End Time = {end_time}')
print(f'Elapsed Time = {time_elapse}')
```

In [ ]:

```
slct_tbl_full_df02.to_csv("News_API_FOX_CNN_Breitbert_May18_23_May31_3.csv", sep=',')
```

The Hill:

In [ ]:

```
slct_tbl_full_df02 = api_record_df.copy()


slct_tbl_full_df02['article_parsed'] = ''

total_urls = len(slct_tbl_full_df02)

# Start timer
start_time = datetime.date.today()


for i, row in enumerate(slct_tbl_full_df02.itertuples(), 1):
    print(row[4])
    article=[]
    #url = slct_tbl_full_df02.at[i,'URL']
    url=row[4]
    response = requests.get(url)
    time.sleep(5 + 10*random.random())

    soup = BeautifulSoup(response.text, 'html.parser')
```

```
    try:
        script_tag = soup.find('script', {'type': 'application/ld+json'})
        if script_tag == None:
            print('\nJSON Object == None: Index:', i-1, '; source:',
                  row[1],
                  '; URL:', row[4])
        else:
            article_json = json.loads(script_tag.string)
            article_content = article_json['articleBody']
            slct_tbl_full_df02.at[row.Index, 'article_parsed'] = article_content

    except KeyError:
        print('\nMissing JSON key: Index:', i-1, '; source:',
              row[1],
              '; URL:', row[4])
        #article_body = soup.find('div', class_='article__content__6hMn9') #Reuters scrape
        #article_body = soup.find('div', class_='Article') #AP scrape attempt
        article_body = soup.find('div', class_='article__text | body-copy | flow') #theHi

        if article_body is None: #forfoxandbreitbert(sometimes)
            print('Class != article__content-container')
            article_body = soup.find('p', class_="speakable")
            if article_body is None: #breitbert(most)
                print('Class != speakable')
                article_body = soup.find('div', class_='entry-content')
                if article_body is None: #WashPost
                    print('Class != entry-content')
                    article_body = soup.find('div',class_='article-body')

        if article_body is not None:
            article_text = article_body.get_text()
            slct_tbl_full_df02.at[row.Index, 'article_parsed'] = article_text
        else:
            print('Could not parse')

    print('.', end='')

# End timer script
end_time = datetime.date.today()
time_elapse = end_time - start_time
print(f'Start Time = {start_time}')
print(f'End Time = {end_time}')
print(f'Elapsed Time = {time_elapse}')
```

```
https://thehill.com/homenews/house/4010248-greene-plans-file-articles-impeachment-biden/

Missing JSON key: Index: 0 ; source: The Hill ; URL: https://thehill.com/homenews/house/40
10248-greene-plans-file-articles-impeachment-biden/
.https://thehill.com/policy/energy-environment/4011061-markey-bill-would-restore-ban-on-u-
s-fossil-fuel-exports/

Missing JSON key: Index: 1 ; source: The Hill ; URL: https://thehill.com/policy/energy-env
ironment/4011061-markey-bill-would-restore-ban-on-u-s-fossil-fuel-exports/
.https://thehill.com/policy/healthcare/4009038-what-a-default-could-mean-for-medicaid-and-
other-federal-benefits/

Missing JSON key: Index: 2 ; source: The Hill ; URL: https://thehill.com/policy/healthcar
e/4009038-what-a-default-could-mean-for-medicaid-and-other-federal-benefits/
.https://thehill.com/business/4010770-with-biden-in-japan-harris-pushes-white-house-messag
e-on-debt-ceiling/

Missing JSON key: Index: 3 ; source: The Hill ; URL: https://thehill.com/business/4010770-
with-biden-in-japan-harris-pushes-white-house-message-on-debt-ceiling/
.https://thehill.com/homenews/house/4010248-greene-plans-file-articles-impeachment-biden/
```

Missing JSON key: Index: 4 ; source: The Hill ; URL: https://thehill.com/homenews/house/4010248-greene-plans-file-articles-impeachment-biden/
.https://thehill.com/homenews/house/4011190-conservatives-make-last-minute-push-for-border-security-in-debt-ceiling-package/

Missing JSON key: Index: 5 ; source: The Hill ; URL: https://thehill.com/homenews/house/4011190-conservatives-make-last-minute-push-for-border-security-in-debt-ceiling-package/
.https://thehill.com/policy/healthcare/4009038-what-a-default-could-mean-for-medicaid-and-other-federal-benefits/

Missing JSON key: Index: 6 ; source: The Hill ; URL: https://thehill.com/policy/healthcare/4009038-what-a-default-could-mean-for-medicaid-and-other-federal-benefits/
.https://thehill.com/business/4010770-with-biden-in-japan-harris-pushes-white-house-message-on-debt-ceiling/

Missing JSON key: Index: 7 ; source: The Hill ; URL: https://thehill.com/business/4010770-with-biden-in-japan-harris-pushes-white-house-message-on-debt-ceiling/
.https://thehill.com/homenews/house/4010248-greene-plans-file-articles-impeachment-biden/

Missing JSON key: Index: 8 ; source: The Hill ; URL: https://thehill.com/homenews/house/4010248-greene-plans-file-articles-impeachment-biden/
.https://thehill.com/blogs/blog-briefing-room/4011488-hyundai-kia-agree-to-pay-200-million-to-settle-lawsuit-after-rise-in-thefts/

Missing JSON key: Index: 9 ; source: The Hill ; URL: https://thehill.com/blogs/blog-briefing-room/4011488-hyundai-kia-agree-to-pay-200-million-to-settle-lawsuit-after-rise-in-thefts/
.https://thehill.com/policy/healthcare/4009038-what-a-default-could-mean-for-medicaid-and-other-federal-benefits/

Missing JSON key: Index: 10 ; source: The Hill ; URL: https://thehill.com/policy/healthcare/4009038-what-a-default-could-mean-for-medicaid-and-other-federal-benefits/
.https://thehill.com/homenews/senate/4011114-cruz-opens-a-probe-into-anheuser-busch-over-dylan-mulvaney-partnership/

Missing JSON key: Index: 11 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4011114-cruz-opens-a-probe-into-anheuser-busch-over-dylan-mulvaney-partnership/
.https://thehill.com/homenews/state-watch/4012940-florida-health-care-providers-are-pausing-gender-affirming-care-for-adults/

Missing JSON key: Index: 12 ; source: The Hill ; URL: https://thehill.com/homenews/state-watch/4012940-florida-health-care-providers-are-pausing-gender-affirming-care-for-adults/
.https://thehill.com/business/4012762-66-progressive-lawmakers-urge-biden-to-use-14th-amendment-in-debt-ceiling-fight/

Missing JSON key: Index: 13 ; source: The Hill ; URL: https://thehill.com/business/4012762-66-progressive-lawmakers-urge-biden-to-use-14th-amendment-in-debt-ceiling-fight/
.https://thehill.com/homenews/campaign/4011971-youngkin-fuels-2024-speculation-with-campaign-style-video/

Missing JSON key: Index: 14 ; source: The Hill ; URL: https://thehill.com/homenews/campaign/4011971-youngkin-fuels-2024-speculation-with-campaign-style-video/
.https://thehill.com/business/ap-american-airlines-and-jetblue-must-abandon-their-partnership-in-the-northeast-federal-judge-rules/

Missing JSON key: Index: 15 ; source: The Hill ; URL: https://thehill.com/business/ap-american-airlines-and-jetblue-must-abandon-their-partnership-in-the-northeast-federal-judge-rules/
.https://thehill.com/homenews/administration/4013134-white-house-optimistic-about-path-forward-despite-serious-differences-on-budget-talks/

Missing JSON key: Index: 16 ; source: The Hill ; URL: https://thehill.com/homenews/administration/4013134-white-house-optimistic-about-path-forward-despite-serious-differences-on-budget-talks/
.https://thehill.com/homenews/senate/4014417-senate-democrat-calls-potential-for-debt-default-a-manufactured-crisis/

Missing JSON key: Index: 17 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4014417-senate-democrat-calls-potential-for-debt-default-a-manufactured-crisis/
.https://thehill.com/homenews/senate/4014204-van-hollen-lawmakers-should-move-to-plan-b-as-debt-ceiling-talks-stall/

Missing JSON key: Index: 18 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4014204-van-hollen-lawmakers-should-move-to-plan-b-as-debt-ceiling-talks-stall/
.https://thehill.com/homenews/administration/4014239-yellen-sidesteps-question-on-which-bills-would-go-unpaid-if-us-defaults-on-debt/

Missing JSON key: Index: 19 ; source: The Hill ; URL: https://thehill.com/homenews/administration/4014239-yellen-sidesteps-question-on-which-bills-would-go-unpaid-if-us-defaults-on-debt/
.https://thehill.com/homenews/senate/4014417-senate-democrat-calls-potential-for-debt-default-a-manufactured-crisis/

Missing JSON key: Index: 20 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4014417-senate-democrat-calls-potential-for-debt-default-a-manufactured-crisis/
.https://thehill.com/homenews/administration/4014239-yellen-sidesteps-question-on-which-bills-would-go-unpaid-if-us-defaults-on-debt/

Missing JSON key: Index: 21 ; source: The Hill ; URL: https://thehill.com/homenews/administration/4014239-yellen-sidesteps-question-on-which-bills-would-go-unpaid-if-us-defaults-on-debt/
.https://thehill.com/homenews/senate/4014204-van-hollen-lawmakers-should-move-to-plan-b-as-debt-ceiling-talks-stall/

Missing JSON key: Index: 22 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4014204-van-hollen-lawmakers-should-move-to-plan-b-as-debt-ceiling-talks-stall/
.https://thehill.com/homenews/administration/4014242-yellen-calls-debt-ceiling-constraint-on-nations-ability-to-pay-its-bills/

Missing JSON key: Index: 23 ; source: The Hill ; URL: https://thehill.com/homenews/administration/4014242-yellen-calls-debt-ceiling-constraint-on-nations-ability-to-pay-its-bills/
.https://thehill.com/homenews/4011270-this-brewery-made-tipping-optional-heres-why/

Missing JSON key: Index: 24 ; source: The Hill ; URL: https://thehill.com/homenews/4011270-this-brewery-made-tipping-optional-heres-why/
.https://thehill.com/homenews/house/4015577-these-are-the-biggest-sticking-points-in-the-debt-ceiling-fight/

Missing JSON key: Index: 25 ; source: The Hill ; URL: https://thehill.com/homenews/house/4015577-these-are-the-biggest-sticking-points-in-the-debt-ceiling-fight/
.https://thehill.com/homenews/senate/4016038-mccarthy-has-little-room-to-maneuver-in-debt-ceiling-talks/

Missing JSON key: Index: 26 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4016038-mccarthy-has-little-room-to-maneuver-in-debt-ceiling-talks/
.https://thehill.com/blogs/blog-briefing-room/4012827-does-god-exist-only-half-of-americans-say-a-definite-yes/

Missing JSON key: Index: 27 ; source: The Hill ; URL: https://thehill.com/blogs/blog-briefing-room/4012827-does-god-exist-only-half-of-americans-say-a-definite-yes/
.https://thehill.com/homenews/administration/4015863-debt-ceiling-yellen-says-treasury-to-run-out-of-funds-by-early-june-and-possibly-june-1/

Missing JSON key: Index: 28 ; source: The Hill ; URL: https://thehill.com/homenews/administration/4015863-debt-ceiling-yellen-says-treasury-to-run-out-of-funds-by-early-june-and-possibly-june-1/
.https://thehill.com/policy/defense/4015404-anti-putin-group-claims-it-has-liberated-town-inside-russias-belgorod-region/

Missing JSON key: Index: 29 ; source: The Hill ; URL: https://thehill.com/policy/defense/4015404-anti-putin-group-claims-it-has-liberated-town-inside-russias-belgorod-region/
.https://thehill.com/business/4015711-health-insurance-for-600000-americans-at-stake-in-de

bt-ceiling-debate/

Missing JSON key: Index: 30 ; source: The Hill ; URL: https://thehill.com/business/4015711-health-insurance-for-600000-americans-at-stake-in-debt-ceiling-debate/
.https://thehill.com/business/budget/4016872-house-republicans-cancel-markups-for-spending-bills-amid-debt-ceiling-talks/

Missing JSON key: Index: 31 ; source: The Hill ; URL: https://thehill.com/business/budget/4016872-house-republicans-cancel-markups-for-spending-bills-amid-debt-ceiling-talks/
.https://thehill.com/homenews/senate/4017344-rick-scott-issues-travel-advisory-for-socialists-warning-florida-is-openly-hostile-to-them/

Missing JSON key: Index: 32 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4017344-rick-scott-issues-travel-advisory-for-socialists-warning-florida-is-openly-hostile-to-them/
.https://thehill.com/homenews/media/4016941-trump-attacks-foxs-laura-ingraham-over-hit-piece-on-his-poll-numbers/

Missing JSON key: Index: 33 ; source: The Hill ; URL: https://thehill.com/homenews/media/4016941-trump-attacks-foxs-laura-ingraham-over-hit-piece-on-his-poll-numbers/
.https://thehill.com/homenews/house/4016326-jeffries-suggests-hed-support-spending-freeze/

Missing JSON key: Index: 34 ; source: The Hill ; URL: https://thehill.com/homenews/house/4016326-jeffries-suggests-hed-support-spending-freeze/
.https://thehill.com/homenews/house/4016098-mccarthy-end-game-on-debt-ceiling-begins-to-come-into-focus/

Missing JSON key: Index: 35 ; source: The Hill ; URL: https://thehill.com/homenews/house/4016098-mccarthy-end-game-on-debt-ceiling-begins-to-come-into-focus/
.https://thehill.com/business/4017263-house-gop-presses-fhfa-chief-on-new-mortgage-fees/

Missing JSON key: Index: 36 ; source: The Hill ; URL: https://thehill.com/business/4017263-house-gop-presses-fhfa-chief-on-new-mortgage-fees/
.https://thehill.com/business/4015711-health-insurance-for-600000-americans-at-stake-in-debt-ceiling-debate/

Missing JSON key: Index: 37 ; source: The Hill ; URL: https://thehill.com/business/4015711-health-insurance-for-600000-americans-at-stake-in-debt-ceiling-debate/
.https://thehill.com/business/4016240-mulvaney-14th-amendment-doesnt-really-solve-the-problem/

Missing JSON key: Index: 38 ; source: The Hill ; URL: https://thehill.com/business/4016240-mulvaney-14th-amendment-doesnt-really-solve-the-problem/
.https://thehill.com/policy/national-security/4017839-trump-faces-intensifying-legal-problems/

Missing JSON key: Index: 39 ; source: The Hill ; URL: https://thehill.com/policy/national-security/4017839-trump-faces-intensifying-legal-problems/
.https://thehill.com/policy/defense/4018797-who-are-the-anti-putin-groups-that-marched-into-russian-territory/

Missing JSON key: Index: 40 ; source: The Hill ; URL: https://thehill.com/policy/defense/4018797-who-are-the-anti-putin-groups-that-marched-into-russian-territory/
.https://thehill.com/homenews/4018653-newsom-knocks-target-ceo-for-pulling-lgbtq-merchandise-from-stores/

Missing JSON key: Index: 41 ; source: The Hill ; URL: https://thehill.com/homenews/4018653-newsom-knocks-target-ceo-for-pulling-lgbtq-merchandise-from-stores/
.https://thehill.com/policy/healthcare/4017827-chicago-mpox-outbreak-raises-alarm-over-summer-spread/

Missing JSON key: Index: 42 ; source: The Hill ; URL: https://thehill.com/policy/healthcare/4017827-chicago-mpox-outbreak-raises-alarm-over-summer-spread/
.https://thehill.com/homenews/house/4019483-democrats-unanimously-back-debt-ceiling-discharge-petition/

Missing JSON key: Index: 43 ; source: The Hill ; URL: https://thehill.com/homenews/house/4019483-democrats-unanimously-back-debt-ceiling-discharge-petition/
.https://thehill.com/blogs/blog-briefing-room/4018452-6-in-10-say-spending-cuts-should-accompany-increase-in-debt-ceiling-poll/

Missing JSON key: Index: 44 ; source: The Hill ; URL: https://thehill.com/blogs/blog-briefing-room/4018452-6-in-10-say-spending-cuts-should-accompany-increase-in-debt-ceiling-poll/
.https://thehill.com/business/4016740-how-us-farmland-became-a-battleground-in-the-fight-against-china/

Missing JSON key: Index: 45 ; source: The Hill ; URL: https://thehill.com/business/4016740-how-us-farmland-became-a-battleground-in-the-fight-against-china/
.https://thehill.com/homenews/4019168-greene-says-no-one-is-concerned-about-debt-default-in-republican-conference/

Missing JSON key: Index: 46 ; source: The Hill ; URL: https://thehill.com/homenews/4019168-greene-says-no-one-is-concerned-about-debt-default-in-republican-conference/
.https://thehill.com/homenews/administration/4019080-yellen-says-its-almost-certain-debt-ceiling-deadline-would-be-in-early-june/

Missing JSON key: Index: 47 ; source: The Hill ; URL: https://thehill.com/homenews/administration/4019080-yellen-says-its-almost-certain-debt-ceiling-deadline-would-be-in-early-june/
.https://thehill.com/homenews/house/4019823-democrats-erupt-in-laughter-after-greene-calls-for-decorum-in-house/

Missing JSON key: Index: 48 ; source: The Hill ; URL: https://thehill.com/homenews/house/4019823-democrats-erupt-in-laughter-after-greene-calls-for-decorum-in-house/
.https://thehill.com/homenews/4019017-haley-says-shed-sign-abortion-ban-but-more-republicans-are-needed-in-congress/

Missing JSON key: Index: 49 ; source: The Hill ; URL: https://thehill.com/homenews/4019017-haley-says-shed-sign-abortion-ban-but-more-republicans-are-needed-in-congress/
.https://thehill.com/policy/national-security/4017839-trump-faces-intensifying-legal-problems/

Missing JSON key: Index: 50 ; source: The Hill ; URL: https://thehill.com/policy/national-security/4017839-trump-faces-intensifying-legal-problems/
.https://thehill.com/homenews/campaign/4018939-7-in-10-independents-concerned-about-bidens-mental-fitness-survey/

Missing JSON key: Index: 51 ; source: The Hill ; URL: https://thehill.com/homenews/campaign/4018939-7-in-10-independents-concerned-about-bidens-mental-fitness-survey/
.https://thehill.com/homenews/house/4019823-democrats-erupt-in-laughter-after-greene-calls-for-decorum-in-house/

Missing JSON key: Index: 52 ; source: The Hill ; URL: https://thehill.com/homenews/house/4019823-democrats-erupt-in-laughter-after-greene-calls-for-decorum-in-house/
.https://thehill.com/business/4018837-moodys-chief-economist-everyone-is-going-to-get-hurt-if-us-defaults-on-debt/

Missing JSON key: Index: 53 ; source: The Hill ; URL: https://thehill.com/business/4018837-moodys-chief-economist-everyone-is-going-to-get-hurt-if-us-defaults-on-debt/
.https://thehill.com/business/4016740-how-us-farmland-became-a-battleground-in-the-fight-against-china/

Missing JSON key: Index: 54 ; source: The Hill ; URL: https://thehill.com/business/4016740-how-us-farmland-became-a-battleground-in-the-fight-against-china/
.https://thehill.com/policy/national-security/4017839-trump-faces-intensifying-legal-problems/

Missing JSON key: Index: 55 ; source: The Hill ; URL: https://thehill.com/policy/national-security/4017839-trump-faces-intensifying-legal-problems/
.https://thehill.com/homenews/campaign/4020983-desantis-says-hell-consider-pardoning-jan-6-defendants-including-trump/

Missing JSON key: Index: 56 ; source: The Hill ; URL: https://thehill.com/homenews/campaign/4020983-desantis-says-hell-consider-pardoning-jan-6-defendants-including-trump/
.https://thehill.com/regulation/court-battles/4020888-kavanaugh-joins-supreme-court-liberals-in-disagreeing-with-new-wetlands-test/

Missing JSON key: Index: 57 ; source: The Hill ; URL: https://thehill.com/regulation/court-battles/4020888-kavanaugh-joins-supreme-court-liberals-in-disagreeing-with-new-wetlands-test/
.https://thehill.com/homenews/senate/4019680-democrats-seek-unlikely-debt-ceiling-savior-mitch-mcconnell/

Missing JSON key: Index: 58 ; source: The Hill ; URL: https://thehill.com/homenews/senate/4019680-democrats-seek-unlikely-debt-ceiling-savior-mitch-mcconnell/
.https://thehill.com/latino/4020967-mexican-president-tells-florida-hispanics-dont-give-one-single-vote-to-desantis/

Missing JSON key: Index: 59 ; source: The Hill ; URL: https://thehill.com/latino/4020967-mexican-president-tells-florida-hispanics-dont-give-one-single-vote-to-desantis/
.https://thehill.com/homenews/house/4021122-house-leaves-town-with-no-debt-limit-deal/

Missing JSON key: Index: 60 ; source: The Hill ; URL: https://thehill.com/homenews/house/4021122-house-leaves-town-with-no-debt-limit-deal/
.https://thehill.com/homenews/house/4021057-35-house-conservatives-pitch-irs-and-covid-19-funding-repeal-to-push-back-debt-x-date/

Missing JSON key: Index: 61 ; source: The Hill ; URL: https://thehill.com/homenews/house/4021057-35-house-conservatives-pitch-irs-and-covid-19-funding-repeal-to-push-back-debt-x-date/
.https://thehill.com/business/4019217-gop-attack-on-irs-funding-runs-counter-to-deficit-reduction-effort/

Missing JSON key: Index: 62 ; source: The Hill ; URL: https://thehill.com/business/4019217-gop-attack-on-irs-funding-runs-counter-to-deficit-reduction-effort/
.https://thehill.com/opinion/national-security/4018271-is-iran-unlocking-the-gates-to-armageddon/

Missing JSON key: Index: 63 ; source: The Hill ; URL: https://thehill.com/opinion/national-security/4018271-is-iran-unlocking-the-gates-to-armageddon/
.https://thehill.com/business/4009135-senators-fight-over-source-of-us-deficit-as-default-looms/

Missing JSON key: Index: 64 ; source: The Hill ; URL: https://thehill.com/business/4009135-senators-fight-over-source-of-us-deficit-as-default-looms/
.https://thehill.com/homenews/campaign/4008113-desantis-sees-both-endorsed-candidates-lose/

Missing JSON key: Index: 65 ; source: The Hill ; URL: https://thehill.com/homenews/campaign/4008113-desantis-sees-both-endorsed-candidates-lose/
.https://thehill.com/homenews/campaign/4008323-trump-taunts-desantis-over-crafts-loss-in-kentucky-gop-race-rons-magic-is-gone/

Missing JSON key: Index: 66 ; source: The Hill ; URL: https://thehill.com/homenews/campaign/4008323-trump-taunts-desantis-over-crafts-loss-in-kentucky-gop-race-rons-magic-is-gone/
.https://thehill.com/business/4009135-senators-fight-over-source-of-us-deficit-as-default-looms/

Missing JSON key: Index: 67 ; source: The Hill ; URL: https://thehill.com/business/4009135-senators-fight-over-source-of-us-deficit-as-default-looms/
.https://thehill.com/blogs/blog-briefing-room/4008216-un-sees-two-in-three-chance-world-will-temporarily-reach-temperature-threshold-soon/

Missing JSON key: Index: 68 ; source: The Hill ; URL: https://thehill.com/blogs/blog-briefing-room/4008216-un-sees-two-in-three-chance-world-will-temporarily-reach-temperature-threshold-soon/
.Start Time = 2023-06-17

```
End Time = 2023-06-17
Elapsed Time = 0:00:00
```

Check Data:

In [ ]:
```
slct_tbl_full_df02.isnull().sum()
```

```
Source            0
Author            0
Title             0
URL               0
date              0
content           0
article_parsed    0
dtype: int64
```

In [ ]:
```
api_record_df_latest=slct_tbl_full_df02.copy()
```

Make sure Word count for parsed article is correct:

In [ ]:
```
api_record_df_latest['word_count'] = api_record_df_latest['article_parsed'].apply(lambda x
```

Only include rows with a word_count greater than 1:

In [ ]:
```
api_record_df_latest = api_record_df_latest[api_record_df_latest['word_count']>1]
```

Save Data to CSV:

In [ ]:
```
api_record_df_latest.to_csv("News_API_TheHill_wordcount3.csv", sep=',')
```

Setup

In [1]:
```
from bs4 import BeautifulSoup
import json
import logging
from newsapi import NewsApiClient
import numpy as np
import os
import pandas as pd
import random
import re
import requests
import string
import time
```

Source URLs

In [ ]:
```
api_key = os.environ['NewsAPIKey']
newsapi = NewsApiClient(api_key=api_key)

sources = newsapi.get_sources(language='en', country='us')
print(' - '.join([source['id'] for source in sources['sources']]))
```

In [ ]:
```
def news_api_urls(q=None,
                  s=None,
                  d_from='2023-05-01',
                  d_to='2023-05-31',
                  api_lst=[]):
    all_articles = newsapi.get_everything(q=q,
                                          sources=s,
                                          from_param=d_from,
                                          to=d_to,
                                          language='en',
                                          sort_by='relevancy',
                                          page=1)

    for article in all_articles['articles']:
        print('Title:', article['title'])

    source_data01 = [(a['source']['name'],
                      a['author'],
                      a['title'],
                      a['url'],
                      a['publishedAt'],
                      a['content'])
                     for a in all_articles['articles']]

    api_lst.extend(source_data01)
    print(len(api_lst))
```

In [ ]:
```
# Already executed:
# run 'A': 'the-washington-post', '2023-05-30'
# run 'B': 'the-washington-post', '2023-05-20/29'
# run 'C': 'the-washington-post', '2023-05-21/22/23/24/25/26/27/28'
# run 'D': 'the-washington-post', '2023-05-10/11/12/13/14/15/16/17/18/19'
# run 'E': 'the-washington-post', '2023-05-05/06/07/08/09/31', '2023-06-01/02/03'
# run 'F': 'fox-news', '2023-05-24/25/26/27/28'
# run 'G': 'breitbart-news', '2023-05-24/25/26/27/28'
```

```
# run 'H': 'cnn', '2023-05-24/25/26/27/28'

# Last to execute:
#run = 'H'
#source_lst = ['cnn']
#date_lst = ['2023-05-24', '2023-05-25', '2023-05-26', '2023-05-27', '2023-05-28']

q_word_lst = ['justice OR surveillance', 'healthcare OR "health care"',
              '(political AND (bias OR party)) OR republican OR democrat OR election',
              'security AND (social OR national)']
```

In [ ]:
```
api_record_lst01 = []
for s in source_lst:
    print(f'Source: {s}')
    for d in date_lst:
        print(f'Date: {d}')
        for q in q_word_lst:
            print(f'Query word: {q}')
            time.sleep(5 * random.random())
            news_api_urls(q=q,
                          s=s,
                          d_from=d,
                          d_to=d,
                          api_lst=api_record_lst01)
```

Scrape articles

In [ ]:
```
api_df = pd.DataFrame(api_record_lst01, columns=['source_name', 'author', 'title', 'url',
api_df['article_text'] = ''

total_urls = len(api_df)
for i, row in enumerate(api_df.itertuples(), 1):
    print(f'Retrieving url {i} of {total_urls}...', end='')
    response = requests.get(row.url)

    if response.status_code == 200:
        print('; now Scraping...', end='')

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

        # The Washington Post
#         try:
#             script_tag = soup.find('script', {'type': 'application/ld+json'})
#             article_json = json.loads(script_tag.string)
#             article_content = article_json['hasPart']['value']
#             api_df.at[row.Index, 'article_text'] = article_content

        # Fox News
#         try:
#             script_tag = soup.find('script', {'type': 'application/ld+json'})
#             article_json = json.loads(script_tag.string)
#             article_content = article_json['articleBody']
#             api_df.at[row.Index, 'article_text'] = article_content

        # Breitbart News
#         try:
#             title_tag = soup.find('h1')
#             if title_tag is not None:
#                 title = title_tag.text
#             content_tags = soup.find_all(['p', 'blockquote'])
#             content = " ".join([tag.text for tag in content_tags if tag.text.strip() !=
#             api_df.at[row.Index, 'article_text'] = content
```

```
        # CNN News
        try:
            script_tag = soup.find('script', {'type': 'application/ld+json'})
            article_json = json.loads(script_tag.string)
            article_content = article_json['articleBody']
            api_df.at[row.Index, 'article_text'] = article_content

        except KeyError:
            print('; missing key in article JSON!', end='')

        time.sleep(5 * random.random())
        print('; done!')
    else:
        print(f'  response is {response.status_code}!')

api_df.to_csv(f'509_final_proj-{run}.csv', index=False)
```

Combine output files (*only at end of iterative process*)

In [ ]:

```
# Combine all the output files
master_df = pd.DataFrame()
for letter in string.ascii_lowercase:
    file_name = f'509_final_proj-{letter.upper()}.csv'

    if os.path.isfile(file_name):
        df = pd.read_csv(file_name)

        master_df = pd.concat([master_df, df], ignore_index=True)
    else:
        break
print(master_df.info())

# Get rid of what appear to be very cluttered (misread) article text rows
pattern = re.compile('\n{3,}')
rows_list = []
for index, row in master_df.iterrows():
    try:
        processed_row = row
        if not pattern.search(str(processed_row[6])):
            rows_list.append(processed_row)
    except Exception as e:
        print(f'Error processing row {index}: {e}')
print(f'Rows processed = {len(rows_list)}')
new_df = pd.concat(rows_list, axis=1).transpose()
new_df.columns = master_df.columns

# Save the new file
new_df.to_csv('509_final_proj.csv', index=False)
```

Combine all final files

In [2]:

```
df0 = pd.read_csv('../data/509_final_proj.csv')
df1 = pd.read_csv('../data/data_parsed_amc.csv')
df2 = pd.read_csv('../data/News_API_FOX_CNN_Breitbert_May18_23_May31_3.csv')
```

In [3]:

```
print(df0.info())
print(df1.info())
print(df2.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1594 entries, 0 to 1593
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   source_name   1594 non-null   object
 1   author        1564 non-null   object
 2   title         1593 non-null   object
 3   url           1593 non-null   object
 4   publishedAt   1593 non-null   object
 5   content       1593 non-null   object
 6   article_text  1561 non-null   object
dtypes: object(7)
memory usage: 87.3+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3390 entries, 0 to 3389
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   source_name     3390 non-null   object
 1   author          3375 non-null   object
 2   title           3390 non-null   object
 3   url             3390 non-null   object
 4   publish_date    3390 non-null   object
 5   article_parsed  3375 non-null   object
dtypes: object(6)
memory usage: 159.0+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1302 entries, 0 to 1301
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Unnamed: 0.1    1302 non-null   int64 
 1   Unnamed: 0      1302 non-null   int64 
 2   Source          1302 non-null   object
 3   Author          1302 non-null   object
 4   Title           1302 non-null   object
 5   URL             1302 non-null   object
 6   date            1302 non-null   object
 7   content         1302 non-null   object
 8   article_parsed  1295 non-null   object
dtypes: int64(2), object(7)
memory usage: 91.7+ KB
None
```

In [4]:

```
df0 = df0.rename(columns={'publishedAt': 'publish_date'})

df1 = df1.rename(columns={'article_parsed': 'article_text'})
df1['content'] = np.nan

df2 = df2.drop(columns=['Unnamed: 0.1', 'Unnamed: 0'])
df2 = df1.rename(columns={'Source': 'source_name',
                          'Author': 'author',
                          'Title': 'title',
                          'URL': 'url',
                          'date': 'publish_date',
                          'article_parsed': 'article_text'})

df = pd.concat([df0, df1, df2], ignore_index=True)

df = df.drop_duplicates(subset='article_text')
df.info()
```

```
df.to_csv('../data/master.csv', index=False)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4509 entries, 0 to 4983
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   source_name   4509 non-null   object
 1   author        4472 non-null   object
 2   title         4509 non-null   object
 3   url           4509 non-null   object
 4   publish_date  4509 non-null   object
 5   content       1158 non-null   object
 6   article_text  4508 non-null   object
dtypes: object(7)
memory usage: 281.8+ KB
```

In [ ]:

PART II:
EDA, Text Preprocessing, & Modeling

Illuminating Mainstream Media Political Bias through Text Mining

Aaron Carr, Azucena Faus, and Dave Friesen - ADS-509-01-SU23

In [1]:
```
__author__ = 'Aaron Carr, Azucena Faus, Dave Friesen'
__email__ = 'acarr@sandiego.edu, afaus@sandiego.edu, dfriesen@sandiego.edu'
__version__ = '1.0'
__date__ = 'June 2023'
```

Setup

In [2]:
```
# Import basic and data access libraries
import numpy as np
import pandas as pd
from profiler import profile, profile_cat

# Import pre-processing, model and performance evaluation libraries
from sklearn.model_selection import train_test_split

from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from nltk.tokenize import sent_tokenize, word_tokenize
from sklearn.decomposition import LatentDirichletAllocation

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import NearestCentroid
from model_process import ModelProcess

from transformers import pipeline as tpipeline

# Import lexicons
#import nltk
#nltk.download('opinion_lexicon')
from nltk.corpus import opinion_lexicon

# Import visualization libraries
from matplotlib import pyplot as plt
%matplotlib inline
import seaborn as sns
from wordcloud import WordCloud

# Import utility libraries
import math
from collections import Counter, defaultdict
from tqdm import tqdm; tqdm.pandas()
```

In [3]:
```
# Set basic np, pd, and plt output defaults (keeping this code 'clean')
%run -i 'defaults.py'
```

Data Ingestion

In [4]:
```
# Instantiate and confirm master dataframe
master_df = pd.read_csv('../data/master.csv')

master_AP_df = pd.read_csv('../data/master_tokenized_AP.csv')
```

```
master_TheHill_df = pd.read_csv('../data/master_business_TheHill.csv')

print(master_df.info())

# Instantiate and confirm test dataframes
master_TheHill_df = master_TheHill_df.rename(columns={
    'Source': 'source_name',
    'Author': 'author',
    'Title': 'title',
    'URL': 'url',
    'date': 'publish_date',
})
master_TheHill_df = master_TheHill_df.drop(
    ['Unnamed: 0.1', 'Unnamed: 0', 'word_count', 'tokens', 'cleaner_text'], axis=1)

master_AP_df = master_AP_df.rename(columns={
    'Source': 'source_name',
    'Author': 'author',
    'Title': 'title',
    'URL': 'url',
    'date': 'publish_date',
    'article_parsed': 'article_text',
})
master_AP_df = master_AP_df.drop(
    ['Unnamed: 0.1', 'Unnamed: 0', 'word_count', 'tokens', 'cleaner_text', 'word_count_tok

master_ex_df = pd.concat([master_TheHill_df, master_AP_df])
print(master_ex_df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4509 entries, 0 to 4508
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   source_name   4509 non-null   object
 1   author        4472 non-null   object
 2   title         4509 non-null   object
 3   url           4509 non-null   object
 4   publish_date  4509 non-null   object
 5   content       1158 non-null   object
 6   article_text  4508 non-null   object
dtypes: object(7)
memory usage: 246.7+ KB
None
<class 'pandas.core.frame.DataFrame'>
Index: 369 entries, 0 to 187
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   source_name   369 non-null    object
 1   author        364 non-null    object
 2   title         369 non-null    object
 3   url           369 non-null    object
 4   publish_date  369 non-null    object
 5   content       369 non-null    object
 6   article_text  369 non-null    object
dtypes: object(7)
memory usage: 23.1+ KB
None
```

In [5]:

```
# "Blanket" label data based on purported source leaning
target_cls_col = 'lean'

def assign_lean(source_name):
    if source_name == 'Breitbart News' or source_name == 'Fox News':
```

```
        return 'right'
    elif source_name == 'CNN' or source_name == 'The Washington Post':
        return 'left'
    else:
        return np.nan
master_df[target_cls_col] = master_df['source_name'].apply(lambda x: assign_lean(x))
```

Tokenization and Cleaning

In [6]:

```
# Stopword removal function, with related initialization
from nltk.corpus import stopwords
sw = stopwords.words('english')
def remove_stop(tokens):
    filtered_tokens = [word for word in tokens if word not in sw]
    return(filtered_tokens)

# Token join back to string
def join_tokens(tokens):
    return ' '.join(tokens)

# Tokenizing function
def tokenize(text):
    return(text.split())  # Tokenize on white space

# Emoji-to-text conversion function
import emoji
def convert_emojis(text):
#    return emoji.demojize(text)
    return emoji.demojize(text).replace('_', ' ')

# Contains-emojis function, with related initialization
all_language_emojis = set()
for country in emoji.EMOJI_DATA :
    for em in emoji.EMOJI_DATA[country]:
        all_language_emojis.add(em)
def contains_emoji(s):
    s = str(s)
    emojis = [ch for ch in s if ch in all_language_emojis]
    return(len(emojis) > 0)

# Punctuation removal function, with related initialization
from string import punctuation
tw_punct = set(punctuation + '’') - {'#'}
def remove_punct(text, punct_set=tw_punct):
    return(''.join([ch for ch in text if ch not in punct_set]))

# Preparation (pipeline) function
def prepare(text, pipeline):
    tokens = str(text)
    for transform in pipeline:
        tokens = transform(tokens)
    return(tokens)
```

In [7]:

```
# Set pipeline
pipeline = [str.lower, remove_punct, convert_emojis, tokenize, remove_stop]

# Clean and tokenize master dataframe
master_df['article_tokens'] = master_df['article_text'].progress_apply(lambda x: prepare(x
master_df['article_text_tokenized'] = master_df['article_tokens'].progress_apply(lambda x:
print(master_df['article_tokens'])
print(master_df['article_text_tokenized'])
```

```
# Also tokenize test dataframes
master_ex_df['article_tokens'] = master_ex_df['article_text'].progress_apply(lambda x: pre
master_ex_df['article_text_tokenized'] = master_ex_df['article_tokens'].progress_apply(lar
```

```
100%|██████████████████████████████████████| 4509/4509 [00:09<00:00, 488.19it/s]
100%|██████████████████████████████████████| 4509/4509 [00:00<00:00, 79830.63it/s]
0       [travelers, alabama, driving, interstate, 65, ...
1       [federal, prosecutor, may, nearing, decision, ...
2       [federal, appeals, court, tuesday, cleared, wa...
3       [speaking, orlando, november, 2015, republican...
4                                                   [nan]
                              ...
4504    [germanys, populist, alternative, germany, afd...
4505    [president, bidens, justice, department, seemi...
4506    [incumbent, turkish, president, recep, tayyip,...
4507    [throughout, month, may, farleft, cnn, attract...
4508    [disney, known, fighting, antigrooming, legisl...
Name: article_tokens, Length: 4509, dtype: object
0       travelers alabama driving interstate 65 partie...
1       federal prosecutor may nearing decision whethe...
2       federal appeals court tuesday cleared way drug...
3       speaking orlando november 2015 republican pres...
4                                                     nan
                              ...
4504    germanys populist alternative germany afd surg...
4505    president bidens justice department seemingly ...
4506    incumbent turkish president recep tayyip erdog...
4507    throughout month may farleft cnn attracted mea...
4508    disney known fighting antigrooming legislation...
Name: article_text_tokenized, Length: 4509, dtype: object
100%|████████████████████████████████████████| 369/369 [00:00<00:00, 428.42it/s]
100%|██████████████████████████████████████| 369/369 [00:00<00:00, 64185.22it/s]
```

Descriptive Stats

In [8]:

```
# Descriptive stats function
def descriptive_stats(tokens, num_tokens=5, verbose=False):
    num_tokens = len(tokens)
    num_unique_tokens = len(set(tokens))  # set() creates unordered set of unique elements
    num_characters = sum(len(token) for token in tokens)  # Finds characters sans spaces
    lexical_diversity = num_unique_tokens / num_tokens

    if verbose:
        print(f'There are {num_tokens} tokens in the data.')
        print(f'There are {num_unique_tokens} unique tokens in the data.')
        print(f'There are {num_characters} characters in the data.')
        print(f'The lexical diversity is {lexical_diversity:.3f} in the data.')

    return([num_tokens, num_unique_tokens, lexical_diversity, num_characters])
```

In [9]:

```
# Descriptive stats across all sources
descriptive_stats([token for sublist in master_df['article_tokens'] for token in sublist])
```

Out[9]:

```
[1977106, 84569, 0.0427741355294051, 12724251]
```

In [10]:

```
# Standard dataframe profile for confirmation
profile(master_df)
```

```
100%|██████████████████████████████████████| 4509/4509 [00:00<00:00, 1287677.32it/s]
```

	Dtype	count	unique	na	na%	mean	std	min	max	skew(>=3)	<v0.01	\
source_name	object	4509.0	4.0									
author	object	4472.0	956.0	37.0	0.8							
title	object	4509.0	4509.0									
url	object	4509.0	4509.0									
publish_date	object	4509.0	4487.0									
content	object	1158.0	1158.0	3351.0	74.3							
article_text	object	4508.0	4508.0	1.0								
lean	object	4509.0	2.0									
article_tokens	object	1977106.0	84569.0									
article_text_tokenized	object	4509.0	4509.0									

In [11]:

```
# Descriptive stats aggregating function
def aggregate_and_describe(group):
    aggregate_tokens = [token for sublist in group['article_tokens'].tolist() for token in
    return descriptive_stats(aggregate_tokens)

# Aggregate descriptive stats by source; convert to dataframe; sort and output
grouped_stats = master_df.groupby('source_name').apply(aggregate_and_describe)
grouped_stats_df = pd.DataFrame(grouped_stats.tolist(), index=grouped_stats.index,
                                columns=['num_tokens', 'num_unique_tokens', 'lexical_diver
grouped_stats_df = grouped_stats_df.sort_index(ascending=False)
print(grouped_stats_df)
```

```
                    num_tokens  num_unique_tokens  lexical_diversity  num_characters
source_name
The Washington Post      366707              32341               0.09         2370171
Fox News                 828739              47097               0.06         5326935
CNN                      409422              34724               0.08         2628951
Breitbart News           372238              36815               0.10         2398194
```

Word Cloud

In [12]:

```
# Word cloud function
def wordcloud(word_freq, title=None, max_words=200, stopwords=None):
    wc = WordCloud(font_path='/Library/Fonts/Arial.ttf',
                   width=800, height=400,
                   background_color= "black", colormap="Paired",
                   max_font_size=150, max_words=max_words)

    # Convert data frame into dict
    if type(word_freq) == pd.Series:
        counter = Counter(word_freq.fillna(0).to_dict())
    else:
        counter = word_freq

    # filter stop words in frequency counter
    if stopwords is not None:
        counter = {token:freq for (token, freq) in counter.items()
                              if token not in stopwords}
    wc.generate_from_frequencies(counter)
```

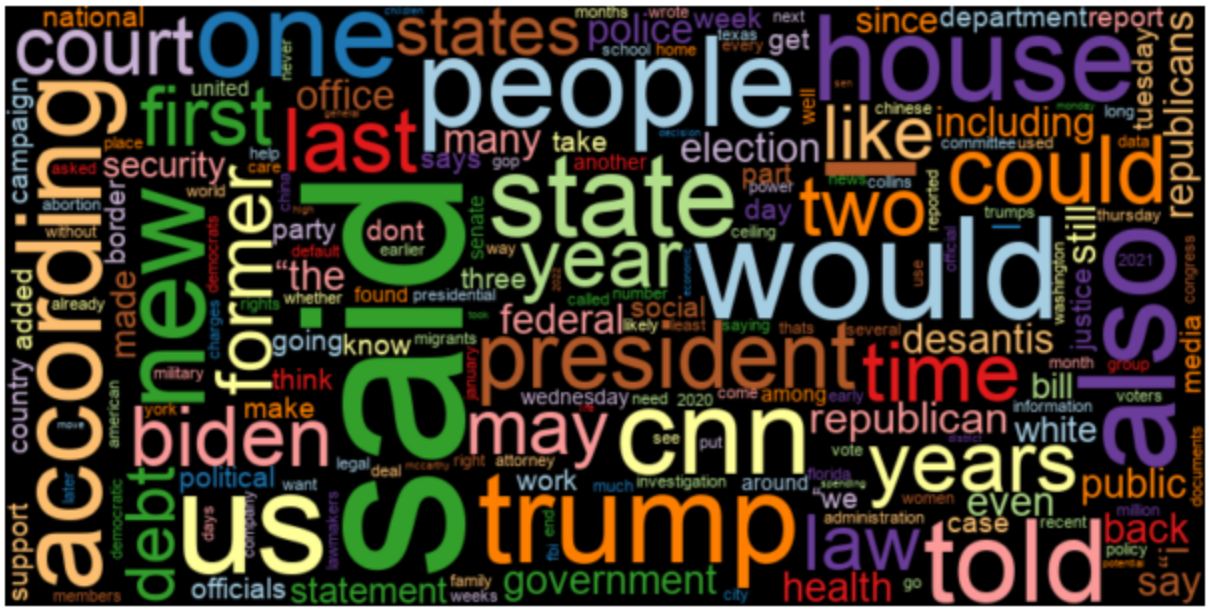
```
    plt.title(title)

    plt.imshow(wc, interpolation='bilinear')
    plt.axis("off")

    plt.show()

# Word count function counter
def count_words(df, column='article_tokens', preprocess=None, min_freq=2):
    # Process tokens and update counter
    def update(doc):
        tokens = doc if preprocess is None else preprocess(doc)
        counter.update(tokens)

    # Create counter and run through all data
    counter = Counter()
    df[column].map(update)

    # Transform counter into data frame
    freq_df = pd.DataFrame.from_dict(counter, orient='index', columns=['freq'])
    freq_df = freq_df.query('freq >= @min_freq')
    freq_df.index.name = 'token'

    return freq_df.sort_values('freq', ascending=False)
```

In [13]:

```
# Iterate and produce word cloud by source
for name, group in master_df.groupby('source_name'):
    print(f"Wordcloud for source: {name}")
    wordcloud(count_words(group)['freq'].to_dict())
```

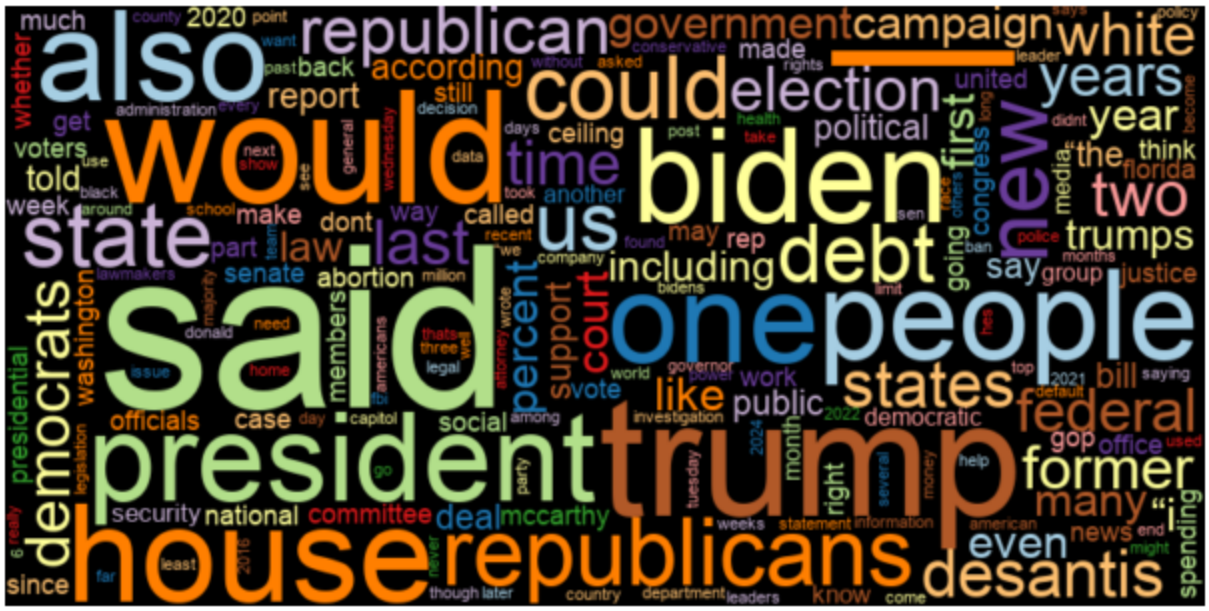
```
Wordcloud for source: Breitbart News
```


```
Wordcloud for source: CNN
```



Wordcloud for source: Fox News

Wordcloud for source: The Washington Post



Partition Data

In [14]:
```
# Set splits
train_ratio = 0.7; val_ratio = 0.20; test_ratio = 0.10

# Split and profile
train_df, test_df = train_test_split(master_df, test_size=1-train_ratio,
                                     random_state=42, stratify=master_df[target_cls_col])
val_df, test_df = train_test_split(test_df, test_size=test_ratio/(test_ratio+val_ratio),
                                   random_state=42, stratify=test_df[target_cls_col])
profile_cat(train_df, [target_cls_col])
```

```
lean -
right 71.17
 left 28.83
```

Topic Modeling

In [15]:
```
# Topic summarization function, from BTAP repo
def display_topics(model, features, no_top_words=5):
    for topic, words in enumerate(model.components_):
        total = words.sum()
        largest = words.argsort()[::-1]  # invert sort order

        print('\nTopic %02d' % topic, end=':')

        out = []
        for i in range(0, no_top_words):
            out.append(' %s (%2.2f)' % (features[largest[i]], abs(words[largest[i]]*100.0/
        print(';'.join(out), end='')
```

In [16]:
```
# Model topics by source
for source_name, group in train_df.groupby('source_name'):
    print(f'Topic modeling for source: {source_name}')

    # Transform article tokens into bag-of-words document-term sparse matrix
    count_vectorizer = CountVectorizer(min_df=0.05, max_df=0.75)
    count_vectors = count_vectorizer.fit_transform(group['article_text_tokenized'])
#     print('Vector shape:', count_vectors.shape)

    lda_model = LatentDirichletAllocation(n_components=5, random_state=42)
    W_lda_matrix = lda_model.fit_transform(count_vectors)
    H_lda_matrix = lda_model.components_

    display_topics(lda_model, count_vectorizer.get_feature_names_out())
    print('\n')

    # Now plot
    num_topics = H_lda_matrix.shape[0]
    words = count_vectorizer.get_feature_names_out()

    # Determine grid size
    grid_size = math.ceil(math.sqrt(num_topics))

    fig, axs = plt.subplots(grid_size, grid_size, figsize=(5, 3))

    # Ensure 1-D array to easily index into subplot grid
    axs = axs.flatten()

    for topic_idx, topic in enumerate(H_lda_matrix):
        # Create dataframe for current topic and sort by importance
        topic_df = pd.DataFrame({'word': words, 'importance': topic})
        topic_df = topic_df.sort_values('importance', ascending=False)
```

```
        # Get top 5 words of topic and create word cloud
        top_words_dict = {row['word']: row['importance'] for idx, row in topic_df.head(5).
        wordcloud = WordCloud(width=800, height=400, background_color='white').generate_fr

        axs[topic_idx].imshow(wordcloud, interpolation='bilinear')
        axs[topic_idx].axis('off')
    #     axs[topic_idx].set_title(f"Topic #{topic_idx+1}", fontsize=20)

    # Turn off unused subplots
    for idx in range(num_topics, len(axs)):
        axs[idx].axis('off')

    plt.tight_layout()
    plt.show()
```

Topic modeling for source: Breitbart News

Topic 00: percent (5.31); desantis (2.53); trump (2.50); news (1.30); president (1.29)
Topic 01: biden (2.85); border (1.88); house (1.82); migrants (1.29); debt (1.23)
Topic 02: chinese (0.98); people (0.97); government (0.94); china (0.94); may (0.89)
Topic 03: 2023 (1.45); women (1.06); children (1.05); may (1.02); news (0.95)
Topic 04: trump (2.19); president (1.46); think (1.27); thats (1.14); people (1.03)

Topic modeling for source: CNN

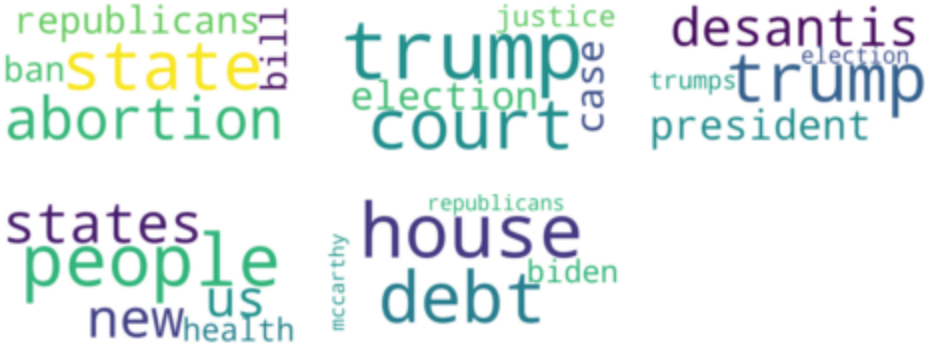
Topic 00: people (1.18); health (0.99); new (0.98); one (0.77); like (0.73)
Topic 01: us (1.97); government (0.81); china (0.71); chinese (0.65); security (0.64)
Topic 02: police (1.67); according (1.33); cnn (1.17); told (1.02); people (0.79)
Topic 03: trump (2.55); desantis (1.70); former (1.29); president (1.16); court (1.08)
Topic 04: house (2.23); debt (1.78); would (1.68); biden (1.12); bill (1.12)

Topic modeling for source: Fox News

Topic 00: ai (1.14); people (1.09); also (0.85); us (0.80); like (0.77)
Topic 01: trump (2.19); president (1.75); desantis (1.59); former (1.31); campaign (1.06)
Topic 02: biden (2.87); house (2.38); president (1.46); debt (1.14); fbi (1.12)
Topic 03: border (2.02); state (1.98); school (1.41); law (1.38); migrants (1.21)
Topic 04: police (1.77); according (0.92); told (0.90); two (0.79); one (0.78)

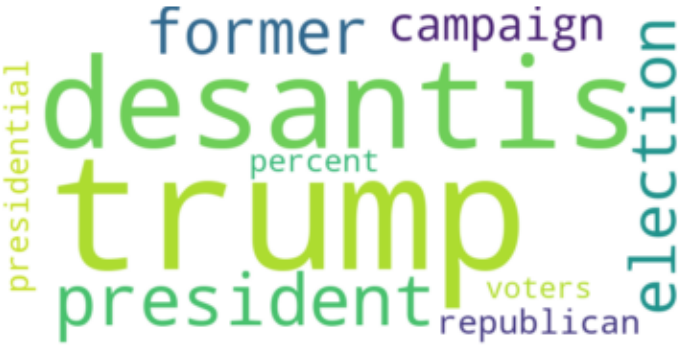
Topic modeling for source: The Washington Post

Topic 00: state (1.91); abortion (1.39); republicans (0.94); bill (0.84); ban (0.81)
Topic 01: trump (1.31); court (0.92); election (0.76); case (0.74); justice (0.72)
Topic 02: trump (3.52); desantis (1.91); president (1.06); trumps (0.85); election (0.83)
Topic 03: people (1.09); states (0.68); new (0.63); us (0.56); health (0.54)
Topic 04: house (2.33); debt (2.05); biden (1.77); republicans (1.29); mccarthy (1.19)



In [17]:

```
# Now do topic modeling on full dataset to find latent groupings
count_vectorizer = CountVectorizer(min_df=0.05, max_df=0.75)
count_vectors = count_vectorizer.fit_transform(train_df['article_text_tokenized'])

n_sources = train_df['source_name'].nunique()
lda_model = LatentDirichletAllocation(n_components=n_sources, random_state=42)
lda_output = lda_model.fit_transform(count_vectors)  # shape: (n_documents, n_topics)

num_topics = lda_model.components_.shape[0]
words = count_vectorizer.get_feature_names_out()

# Plot
grid_size = math.ceil(math.sqrt(num_topics))
fig, axs = plt.subplots(grid_size, grid_size)

# 1-D array to easily index into subplot grid
axs = axs.flatten()

for topic_idx, topic in enumerate(lda_model.components_):
    # Create dataframe for current topic and sort by importance
    topic_df = pd.DataFrame({'word': words, 'importance': topic})
    topic_df = topic_df.sort_values('importance', ascending=False)

    # Get top 10 words of the topic and create a word cloud
    top_words_dict = {row['word']: row['importance'] for idx, row in topic_df.head(10).ite
    wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_f

    axs[topic_idx].imshow(wordcloud, interpolation='bilinear')
    axs[topic_idx].axis('off')

# Turn off unused cells in plot
for idx in range(num_topics, len(axs)):
    axs[idx].axis('off')

plt.tight_layout()
plt.show()
```



Text Summarization and Sentiment Analysis

In [18]:
```
# NLTK opinion lexicon
positive_words = set(opinion_lexicon.positive())
negative_words = set(opinion_lexicon.negative())
```

In [19]:
```
# List of "assumed" political phrases
political_phrases = ['gun rights', 'voting rights', 'climate change', 'immigration reform'
                     'tax cuts', 'universal healthcare',
                     'criminal justice reform', 'income inequality']
```

In [20]:
```
# Group train_df by 'source_name' for source-level comparison
grouped_df = train_df.groupby('source_name')
#grouped_df = master_ex_df.groupby('source_name')

# Instantiate BERT pipeline
nlp = tpipeline('sentiment-analysis')

# Create dictionaries for scores
political_phrase_scores = {}
sentiment_scores = {}
sentiment_scores_T = {}

# Iterate over sources and calc TF-IDF scores vs. political phrases
for source, group in tqdm(grouped_df):
    tfidf_vectorizer = TfidfVectorizer(ngram_range=(1, 3))
    tfidf_vectors = tfidf_vectorizer.fit_transform(group['article_text_tokenized'])

    # Calc TF-IDF sum (scores) where political phrases found
    scores = {}
    sentiment = defaultdict(lambda: defaultdict(int))
    sentiment_T = defaultdict(lambda: defaultdict(int))

    # Iterate over political phrases
    for phrase in political_phrases:
        try:
            index = tfidf_vectorizer.get_feature_names_out().tolist().index(phrase)   # try
```

```
            scores[phrase] = tfidf_vectors[:, index].sum()  # and sum related score
        except ValueError:
            pass  # didn't find political phrase

        # Iterate over each article in the group to calc sentiment
        for i in group.index:
            tokenized_text = group.loc[i, 'article_text_tokenized']
            original_text = group.loc[i, 'article_text']

            # Tokenize text into sentences because we're calc'ing sentiment on phrase-rele
            sentences = sent_tokenize(tokenized_text)

            # Check each sentence if it contains the political phrase
            for sentence in sentences:
                if phrase in sentence:
                    # [Tokenize the sentence into words
                    tokens = word_tokenize(sentence)

                    # Count positive and negative words
                    for word in tokens:
                        if word in positive_words:
                            sentiment[phrase]['positive'] += 1
                        elif word in negative_words:
                            sentiment[phrase]['negative'] += 1

                    # Now use original text (not tokenized) for BERT
                    original_sentences = sent_tokenize(original_text)
                    for original_sentence in original_sentences:
                        if phrase in original_sentence:
                            # Limit sentence to first 512 tokens for BERT
                            bert_sentence = ' '.join(word_tokenize(original_sentence)[:512

                            # Get sentiment using BERT
                            result = nlp(bert_sentence)[0]
                            sentiment_T[phrase][result['label'].lower()] += 1

    # Add the scores to the dictionary
    political_phrase_scores[source] = scores
    sentiment_scores[source] = dict(sentiment)
    sentiment_scores_T[source] = dict(sentiment_T)
```

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english (http
s://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english)
100%|██████████████████████████████████████████████████| 4/4 [01:07<00:00, 16.80s/it]
```

In [21]:

```
# Calc aggregate scores against which to compare "hits" above
all_scores = np.asarray(tfidf_vectors.sum(axis=0)).flatten()
mean_score = np.mean(all_scores)
median_score = np.median(all_scores)

results_df = pd.DataFrame()

# Iterate over sources and political phrase TF-IDF scores and show results
for source_name in political_phrase_scores:
    print(f'\nScores for {source_name}:')
    phrase_scores = political_phrase_scores[source_name]
    sentiment = sentiment_scores[source_name]
    sentiment_T = sentiment_scores_T[source_name]

    # . . . by political phrase
    results = []
    for phrase in political_phrases:
        score = phrase_scores.get(phrase, 0)
        relative_to_mean = score / mean_score if mean_score != 0 else 0
        relative_to_median = score / median_score if median_score != 0 else 0
```

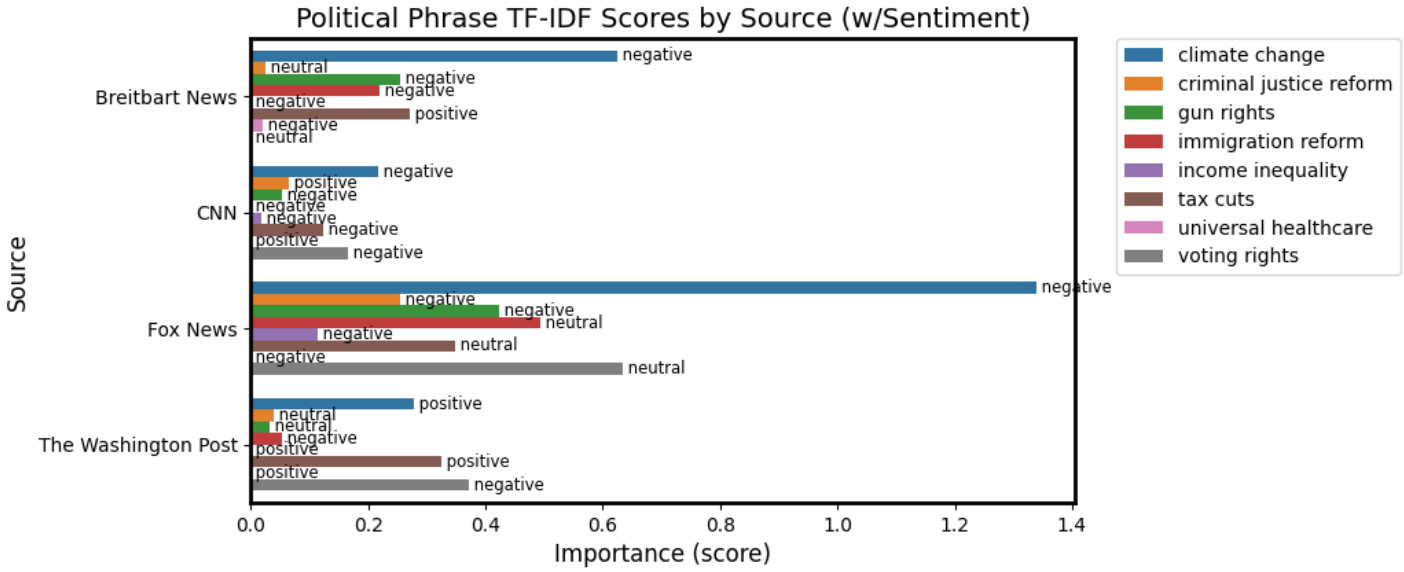
```
        # Categorize based on relative_to_median (otherwise arbitrary)
        if relative_to_median > 10:
            category = 'high'
        elif 5 < relative_to_median <= 10:
            category = 'medium'
        else:
            category = 'low'

        sentiment_phrase = sentiment.get(phrase, {'positive': 0, 'negative': 0})
        sentiment_phrase_T = sentiment_T.get(phrase, {'positive': 0, 'negative': 0})
        results.append({
            'source_name': source_name,
            'phrase': phrase,
            'score': score,
            'relative_to_mean': relative_to_mean,
            'relative_to_median': relative_to_median,
            'category': category,
            'p_sentiment': sentiment_phrase['positive'],
            'p_sentiment_T': sentiment_phrase_T['positive'],
            'n_sentiment': sentiment_phrase['negative'],
            'n_sentiment_T': sentiment_phrase_T['negative'],
            'sentiment': sentiment_phrase['positive'] + (sentiment_phrase['negative'] * -1
            'sentiment_T': sentiment_phrase_T['positive'] + (sentiment_phrase_T['negative'
        })

    for result in results:
        result['sentiment_label'] = 'positive'\
            if result['sentiment'] > 0 else 'negative' if result['sentiment'] < 0 else 'ne
        result['sentiment_label_T'] = 'positive'\
            if result['sentiment_T'] > 0 else 'negative' if result['sentiment_T'] < 0 else

    # Sort results by score
    results.sort(key=lambda x: x['score'], reverse=True)

    # Print sorted results
    for result in results:
        print(f"{result['phrase']}:  {result['category']} importance ",
              f"{result['sentiment_label']} sentiment")

    results_df = pd.concat([results_df, pd.DataFrame(results)])
```

```
Scores for Breitbart News:
climate change:  high importance  negative sentiment
tax cuts:  high importance  positive sentiment
gun rights:  high importance  negative sentiment
immigration reform:  high importance  positive sentiment
criminal justice reform:  low importance  negative sentiment
universal healthcare:  low importance  negative sentiment
voting rights:  low importance  neutral sentiment
income inequality:  low importance  neutral sentiment

Scores for CNN:
climate change:  high importance  negative sentiment
voting rights:  medium importance  negative sentiment
tax cuts:  medium importance  negative sentiment
criminal justice reform:  low importance  negative sentiment
gun rights:  low importance  negative sentiment
income inequality:  low importance  negative sentiment
immigration reform:  low importance  neutral sentiment
universal healthcare:  low importance  neutral sentiment

Scores for Fox News:
climate change:  high importance  negative sentiment
voting rights:  high importance  positive sentiment
```

```
immigration reform:  high importance  positive sentiment
gun rights:  high importance  negative sentiment
tax cuts:  high importance  negative sentiment
criminal justice reform:  high importance  negative sentiment
income inequality:  medium importance  positive sentiment
universal healthcare:  low importance  neutral sentiment

Scores for The Washington Post:
voting rights:  high importance  negative sentiment
tax cuts:  high importance  negative sentiment
climate change:  high importance  negative sentiment
immigration reform:  low importance  positive sentiment
criminal justice reform:  low importance  positive sentiment
gun rights:  low importance  negative sentiment
universal healthcare:  low importance  neutral sentiment
income inequality:  low importance  neutral sentiment
```

In [22]:

```
# Sort DataFrame by 'source' and 'phrase' to match order of bars in plot
sorted_df = results_df.sort_values(['source_name', 'phrase'])

# Create barplot
fig, ax = plt.subplots()
sns.barplot(data=sorted_df, y='source_name', x='score',
            ax=ax, hue='phrase', errorbar=None)

# Iterate over bars and dataframe rows to add sentiment
for p, (_, row) in zip(ax.patches, sorted_df.iterrows()):
    plt.text(p.get_width(), p.get_y() + p.get_height()/2,
             f' {row["sentiment_label"]}',
             ha='left', va='center', fontsize='smaller')

ax.set_title('Political Phrase TF-IDF Scores by Source (w/Sentiment)')
ax.set_xlabel('Importance (score)')
ax.set_ylabel('Source')

plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)

plt.show()
```



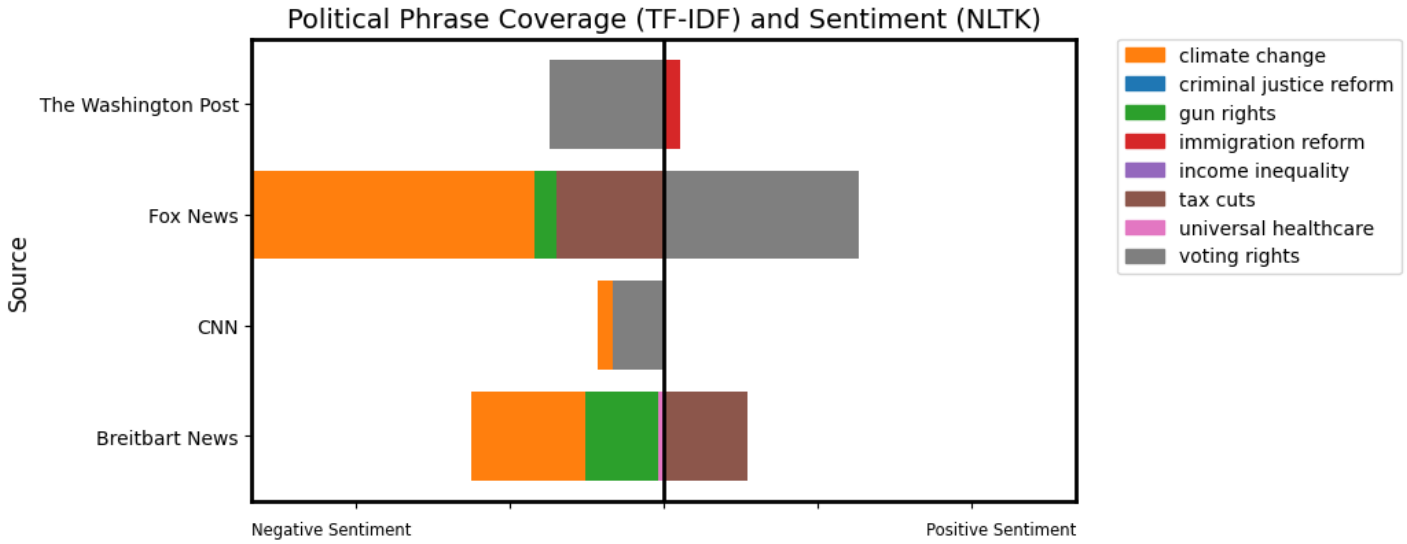
In [23]:

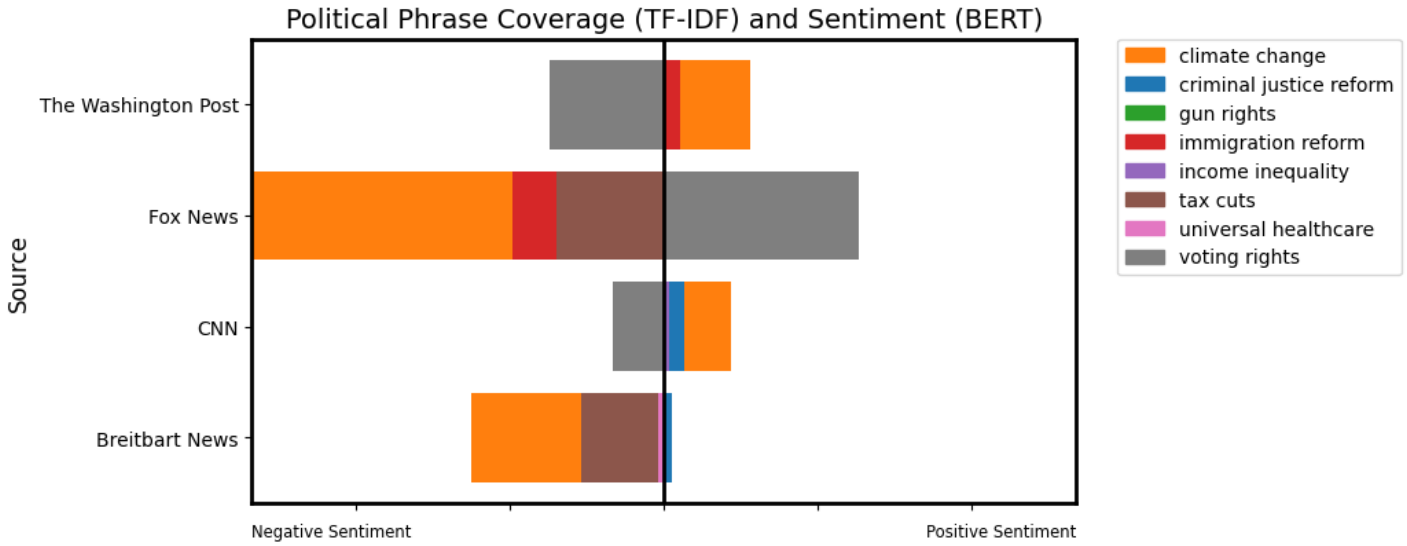
```
# Sentiment comparision plot function
def create_plot(df, sentiment_label, title):
    # Sort DataFrame by 'source' and 'phrase' to match order of bars in plot
    sorted_df = df.sort_values(['source_name', 'phrase'])

    # Create plot
```

```
    fig, ax = plt.subplots()

    # Define a custom color palette
    custom_palette = ['#FF7F0E', '#1F77B4', '#2CA02C', '#D62728', '#9467BD',
                      '#8C564B', '#E377C2', '#7F7F7F', '#BCBD22', '#17BECF']

    # Ensure the colors are consistent
    hue_order = sorted_df['phrase'].unique()
    palette = [custom_palette[i % len(custom_palette)] for i in range(len(hue_order))]

    # List to store the created patches
    patches = []

    # Iterate over dataframe rows to add bars
    for i, (_, row) in enumerate(sorted_df.iterrows()):
        # Adjust score so that 'negative' sentiments display to the left of the y-axis
        score = row['score'] if row[sentiment_label] != 'negative' else row['score'] * -1

        color = palette[np.where(hue_order == row['phrase'])[0][0]]
        patch = ax.barh(row['source_name'], score, color=color)
        patches.append(patch)

    ax.set_title(title)
    ax.set_ylabel('Source')

    # Create legend
    handles = [plt.Rectangle((0,0),1,1, color=palette[i]) for i in range(len(hue_order))]
    plt.legend(handles, hue_order, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)

    # Center x-axis
    max_abs_score = abs(sorted_df['score']).max()
    ax.set_xlim([-max_abs_score, max_abs_score])
    ax.axvline(0, color='black')  # draw y-axis line

    # Remove x-axis labels and add custom labels
    ax.xaxis.set_ticklabels([])

    ax.annotate('Negative Sentiment', xy=(-max_abs_score, 0), xytext=(0,-10),
                xycoords=('data', 'axes fraction'), textcoords='offset points',
                ha='left', va='top', fontsize='smaller')

    ax.annotate('Positive Sentiment', xy=(max_abs_score, 0), xytext=(0,-10),
                xycoords=('data', 'axes fraction'), textcoords='offset points',
                ha='right', va='top', fontsize='smaller')

    plt.show()
```

In [24]:

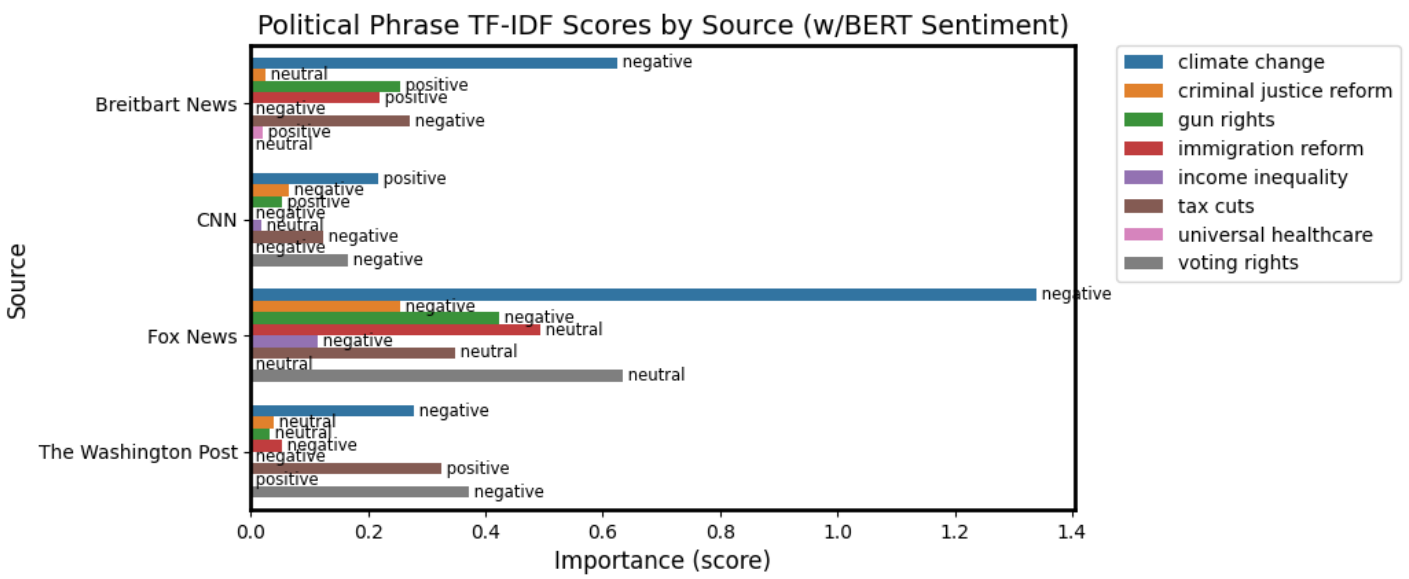
```
create_plot(results_df, 'sentiment_label', 'Political Phrase Coverage (TF-IDF) and Sentime
create_plot(results_df, 'sentiment_label_T', 'Political Phrase Coverage (TF-IDF) and Senti
```





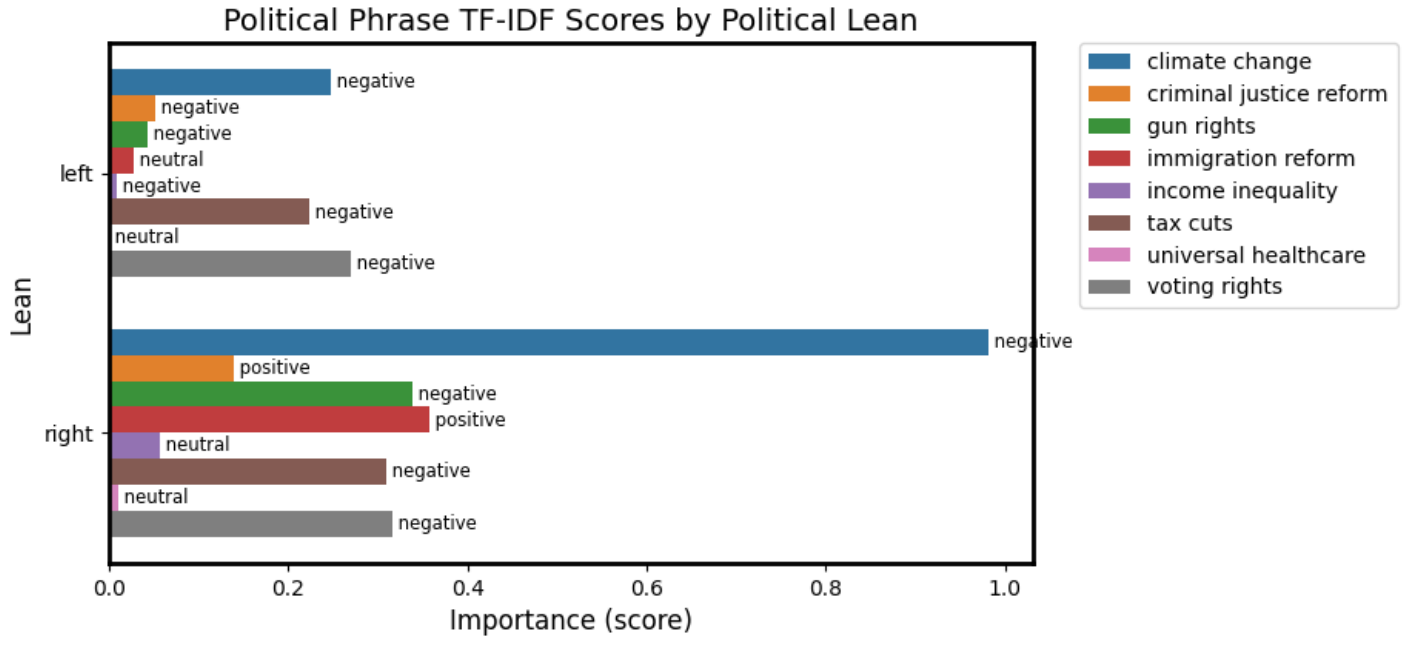
In [25]:

```
# Sort DataFrame by 'source' and 'phrase' to match order of bars in plot
sorted_df = results_df.sort_values(['source_name', 'phrase'])

# Create barplot
fig, ax = plt.subplots()
sns.barplot(data=sorted_df, y='source_name', x='score',
            ax=ax, hue='phrase', errorbar=None)

# Iterate over bars and dataframe rows to add sentiment
for p, (_, row) in zip(ax.patches, sorted_df.iterrows()):
    plt.text(p.get_width(), p.get_y() + p.get_height()/2,
             f' {row["sentiment_label_T"]}',
             ha='left', va='center', fontsize='smaller')

ax.set_title('Political Phrase TF-IDF Scores by Source (w/BERT Sentiment)')
ax.set_xlabel('Importance (score)')
ax.set_ylabel('Source')

plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)

plt.show()
```



In [26]:

```
# Sort DataFrame by political 'lean' and 'phrase' to match order of bars in plot
lean_df = train_df[['source_name', 'lean']].drop_duplicates()
results_df = pd.merge(results_df, lean_df, on='source_name', how='left')
sorted_df = results_df.sort_values(['lean', 'phrase'])

# Create barplot
fig, ax = plt.subplots()
sns.barplot(data=sorted_df, y='lean', x='score',
            ax=ax, hue='phrase', errorbar=None)

# Iterate over bars and dataframe rows to add sentiment
for p, (_, row) in zip(ax.patches, sorted_df.iterrows()):
    plt.text(p.get_width(), p.get_y() + p.get_height()/2,
             f' {row["sentiment_label"]}',
             ha='left', va='center', fontsize='smaller')

ax.set_title('Political Phrase TF-IDF Scores by Political Lean')
ax.set_xlabel('Importance (score)')
ax.set_ylabel('Lean')

plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)

plt.show()
```



Modeling

In [27]:
```
import warnings
warnings.filterwarnings('ignore')

# Define labels
target_labels = master_df[target_cls_col].unique().tolist()

# Dummy function to leverage existing tokenization; and tokenize
def identity_tokenizer(tokens):
    return tokens
tfidf_vectorizer = TfidfVectorizer(tokenizer=identity_tokenizer, preprocessor=lambda x: x,
                                   lowercase=False, ngram_range=(1, 3))

# Define train/val/test per original split
X_train = tfidf_vectorizer.fit_transform(train_df['article_text_tokenized'])
y_train = train_df[target_cls_col]
X_val = tfidf_vectorizer.transform(val_df['article_text_tokenized'])
y_val = val_df[target_cls_col]
X_test = tfidf_vectorizer.transform(test_df['article_text_tokenized'])
y_test = test_df[target_cls_col]

# Finally, convert sparse matrices to dataframes as required by ModelProcess class
X_train = pd.DataFrame.sparse.from_spmatrix(X_train, columns=tfidf_vectorizer.get_feature_
X_val = pd.DataFrame.sparse.from_spmatrix(X_val, columns=tfidf_vectorizer.get_feature_name
X_test = pd.DataFrame.sparse.from_spmatrix(X_test, columns=tfidf_vectorizer.get_feature_na
```

In [28]:
```
# Set model list
mp_queue = (
#     (LogisticRegression(), {'C': 0.01, 'class_weight': 'balanced', 'random_state': 42}),
    (NearestCentroid(), {}),
)
```

In [29]:
```
import copy

ModelProcess.show_progress = True

# Iterate models (note use of 'copy' is to preserve mutable elements
#   of model_queue tuple for possible later use)
mp_df = pd.DataFrame(mp_queue, columns=['algorithm', 'params'])
mp_df['mp'] = mp_df.apply(
    lambda mp: ModelProcess(copy.deepcopy(mp['algorithm']), None,
                            copy.copy(mp['params']),
                            X_train, y_train,
                            X_val, y_val,
                            X_test, y_test,
                            labels=target_labels,
                            cat_cols=None, num_cols=None,
                            balance_target=True).train_validate_test(), axis=1)

# Compile, sort, and display results
mp_df[['train_acc', 'train_f1', 'train_time',
       'val_acc', 'val_f1', 'val_time',
       'test_acc', 'test_f1', 'test_time']] =\
    mp_df['mp'].apply(
        lambda mp: sum(list(map(
            lambda dataset: mp.score[dataset] + [mp.time[dataset]], ['train', 'val', 'test
mp_df.sort_values(by=['train_f1', 'val_f1', 'test_f1'],
                  ascending=[False, False, False], inplace=True)
mp_df.loc[:, mp_df.columns != 'mp'].to_csv('results_table.csv')
```

```
print()
mp_df
```

```
NearestCentroid: train... done in 7.64s.
NearestCentroid: val... done in 0.82s.
NearestCentroid: test... done in 0.68s.
```

Out[29]:

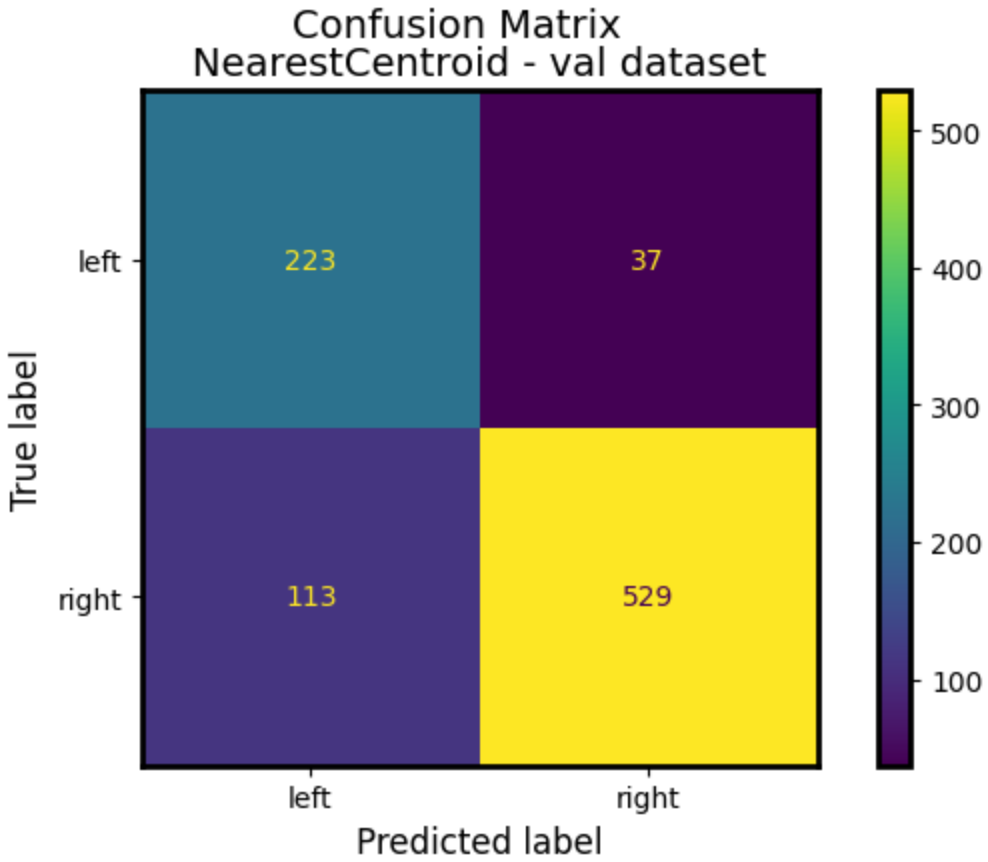
	algorithm	params	mp	train_acc	train_f1	train_time	val_acc	val_f1	va
0	NearestCentroid()	{}	<model_process.ModelProcess object at 0x7fd439...	0.84	0.76	7.64	0.83	0.75	

In [30]:

```
# Select model (process) for focus and tuning
select_mp = 0
```

In [31]:

```
# Show confusion matrix and summary for top Neural Network
mp_df.loc[select_mp]['mp'].confusion_matrix('val')
mp_df.loc[select_mp]['mp'].summary('val')
```




```
NearestCentroid - val dataset

              precision    recall  f1-score   support

        left       0.66      0.86      0.75       260
       right       0.93      0.82      0.88       642

    accuracy                           0.83       902
   macro avg       0.80      0.84      0.81       902
weighted avg       0.86      0.83      0.84       902
```

In [ ]:

509 Final Project

- Import Libraries
- Define Functions
- Load MASTER data and/or Business Data
- Pre-Processing
- EDA
- Linear SVC Train/Test
- SKLEARN Linear SVC Train/Test
- Linear SVC Business Data Results

Globally import libraries

In [385…

```
#! pip install pyLDavis

import pyLDAvis

pyLDAvis.enable_notebook()
from tqdm.auto import tqdm
import spacy
import pyLDAvis.lda_model
import pyLDAvis.gensim_models

import numpy as np
import pandas as pd
import pymysql as mysql
import matplotlib.pyplot as plt
import os
import shutil
import re
import logging
import time
import zipfile
import requests
from bs4 import BeautifulSoup
import datetime
import re
import regex as rex
from collections import defaultdict, Counter
import random
import requests
from bs4 import BeautifulSoup
import datetime
import json
from wordcloud import WordCloud
from tabulate import tabulate
from sklearn.svm import SVC
import textacy.preprocessing as tprep
from textacy.extract import keyword_in_context

import sqlite3
import nltk
from string import punctuation
from nltk.corpus import stopwords
import re
import emoji
from nltk.metrics import ConfusionMatrix
import itertools
import collections
import pickle

from sklearn.model_selection import train_test_split
```

```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.decomposition import NMF
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, f1_score, recall_score, precision_recall_curve
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.model_selection import GridSearchCV

## Depracated:
# from sklearn.metrics import plot_confusion_matrix
## New version:
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.svm import LinearSVC
from sklearn.decomposition import LatentDirichletAllocation

#import mysql.connector

# Set pandas global options
pd.options.display.max_rows = 17

import warnings
warnings.filterwarnings('ignore')
warnings.simplefilter('ignore')
```

Functions:

Data pre-processing:

REGEX and NORMALIZE FUNCTIONS

In [211…

```
rex_sep = rex.compile(r' ')
rex_ucode = rex.compile(r'[\\]u20*')

'''re.sub lambda citation:
https://chat.openai.com/share/402ec66e-2802-4cda-af8c-6f9f5b097d85
'''
sep_lst = []
ucode_lst = []
# Add leading and trailing space to URLs
def rex_replace(text):
    #txt = str(text)
    #print(lambda x: x.replace(' ', ' '))
    #sep_lst.append(rex_sep.findall(txt))
    #ucode_lst.append(rex_ucode.findall(txt))
    text = text.replace(r' ', ' ').replace(r'-', ' ')\
    .replace(r'\n', ' ').replace('\u2063', ' ').replace('\u2066', ' ')\
    .replace('\u2069', ' ').replace('\u200b', ' ').replace('\u200d', ' ')\
    .replace('(click to view)', ' ')\
    .replace('a post shared by', ' ')\
    .replace('app users click here', ' ')\
    .replace('app users: click here', ' ')\
    .replace('app users, click here:', ' ')\
    .replace('click here.', ' ')\
    .replace('click here for more cartoons', ' ')\
    .replace('click here for more', ' ')\
    .replace('click here for more sports coverage on foxnews.com', ' ')\
    .replace('click here for other fox news digital adoptable pets stories', ' ')\
    .replace('click here for the fox news app', ' ')\
    .replace('click here for the latest fox news reporting', ' ')\
    .replace('click here for topline and cross tabs conducted', ' ')\
    .replace('click here to hear more', ' ')\
    .replace('click here to ge the fox news app', ' ')\
    .replace('click here to get the fox news app', ' ')\
    .replace('click here to get the opinion newsletter', ' ')\
    .replace('click here to learn more', ' ')\
    .replace('click here to read more', ' ')\
    .replace('click here to sign up for our health newsletter', ' ')\
    .replace('click here to sign up for our lifestyle newsletter', ' ')\
    .replace('click here to sign up for our opinion newsletter', ' ')\
    .replace('click here to sign up for the entertainment newsletter', ' ')\
    .replace('click here to subscribe and get your first year of fox nation free of charge', ' ')\
```

```
    .replace('click here to view', ' ')\
    .replace("click to get kurt's cyberguy newsletter with quick tips, tech reviews, security alerts and ea
    .replace("click to get kurt's cyberguy newsletter with security alerts, quick tips, tech reviews, secur
    .replace("click to get kurt's free cyberguy newsletter with quick tips, tech reviews, security alerts a
    .replace("click to get kurt's free cyberguy newsletter with security alerts, quick tips, tech reviews,
    .replace('click to get the fox news app', ' ')\
    .replace('fox news digital', ' ')\
    .replace('request for comment', ' ')\
    .replace('the ap ', ' ')\
    .replace('copyright © 2023 breitbart', ' ')\
    .replace('all rights reserved', ' ')\
    .replace('copyright 2023 cyberguy.com', ' ')\
    .replace('copyright 2023 fox news network', ' ')\
    .replace('copyright 2023 viq media transcription', ' ')\
    .replace("please let us know if you're having issues with commenting", ' ')\
    .replace('view this post on instagram', ' ')
    #txt = txt
    #text = text.replace(r'200b', 'd171c')
    #text = rex_ucode.sub('', text)
    return text

def normalize(text):
    text = tprep.normalize.hyphenated_words(text)
    text = tprep.normalize.quotation_marks(text)
    text = tprep.normalize.unicode(text)
    text = tprep.remove.accents(text)
    return text
```

LEMMATIZATION FUNCTION

In [212…

```
nlp_trans = spacy.load('en_core_web_sm')

def lemma(text):
    trans_txt = nlp_trans(text)
    tokens = [t.lemma_ for t in trans_txt]
    return tokens
```

- CASE LOAD
- STOPWORD REMOVAL
- URL REMOVAL
- EMOJI REMOVAL
- PUNCTUATION REMOVAL
- MESSY TEXT REMOVAL
- TOKENIZE

In [217…

```
# CASE LOAD, REMOVE STOPWORDS,
# EMOJI and PUNCTUATION REMAL,
# URL REMOVAL
# TOKENIZE
# REMOVE MESSY text

punctuation = set(punctuation) # speeds up comparison
tw_punct = punctuation - {"#"}

sw = stopwords.words("english")
sw = sw + ['nan']
sw = sw + ['said'] + ['news'] + ['us'] + ['reuters'] + ['ap'] \
    + ['fox'] + ['cnn'] + ['breitbart'] + ['digital'] + ['follow'] \
    + ['associated press'] + ['press contributed'] + ['press'] \
    + ['dont'] + ['2023'] + ['told digital'] + ['associated contributed']\
    + ['contributed report'] + ['associated'] + ['contributed'] +\
    ['report'] + ['continued'] + ['reportedly'] + ['im']

# Two useful regex
whitespace_pattern = re.compile(r"\s+")
hashtag_pattern = re.compile(r"^#[0-9a-zA-Z]+")

def emoji_split(text):
    return("".join([' ' + ch + ' ' if emoji.is_emoji(ch) else ch for ch in text]))

def remove_stop(tokens) :
```

```
    # modify this function to remove stopwords

    return[t for t in tokens if t not in sw]

def remove_punctuation(text, punct_set=tw_punct) :
    return("".join([ch for ch in text if ch not in punct_set]))

def tokenize(text) :
    """ Splitting on whitespace rather than the book's tokenize function. That
        function will drop tokens like '#hashtag' or '2A', which we need for Twitter. """

    return([item.lower() for item in whitespace_pattern.split(text)])

def remove_url(text):
    return(re.sub(r'http\S+', '', text))

def remove_messy(text): # remove words that give away the source
    text1=re.sub(r'cnn', '', text)
    text2=re.sub(r'fox', '', text1)
    text3=re.sub(r' — ', '', text2)
    text4=re.sub(r'breitbart', '', text3)
    return(re.sub(r'\\n', '', text4))

# two pipelines to either tokenize or simply remove punctuation
# and lowercase as we will need to extract feature words:

full_pipeline = [str.lower, remove_url, rex_replace, emoji_split, remove_messy,
                 remove_punctuation, tokenize, remove_stop]
first_pipeline = [str.lower, remove_url, rex_replace, emoji_split, remove_messy,
                  remove_punctuation]


def prepare(text, pipeline) :
    tokens = str(text)

    for transform in pipeline :
        tokens = transform(tokens)

    return(tokens)
```

Feature extraction Function:

In [7]:

```
def conv_features(text,fw) :
     feature_set=dict()
     for word in text.split():
          if word in fw:
               feature_set[word]=True
     return(feature_set)
```

EDA functions:

- GET PATTERNS
- WORD COUNTS
- WORDCLOUD
- TOPIC MODELING

In [8]:

```
def get_patterns(text_analyze, num_words, T):
    if(len(text_analyze)==0):
        raise ValueError("Can't work with empty text object")
    total_tokens = 1
    unique_tokens = 0
    avg_token_len = 0.0
    lexical_diversityP = 0.0
    top_words = []

    # Only applying the token_normal, which takes only alphanumeric values
    # to twitter data:
    if T ==1:
        text_analyze=token_normal(text_analyze)

    total_tokens = len(text_analyze)
```

```
    unique_tokens = len(set(text_analyze))
    lexical_diversityP = unique_tokens/total_tokens
    avg_token_len = np.mean([len(ta) for ta in text_analyze])

    top_words_1 = collections.Counter(text_analyze)
    top_words = top_words_1.most_common(num_words)

    results={'tokens': total_tokens,
             'unique_tokens': unique_tokens,
             'avg_token_length': avg_token_len,
             'lexical_diversity': lexical_diversityP,
             'top_words': top_words}
    return(results)
```

In [9]:

```
def wordcloud(word_freq, title=None, max_words=200, stopwords=None):

    wc = WordCloud(width=800, height=400,
                   background_color= "black", colormap="Paired",
                   max_font_size=150, max_words=max_words)

    # convert data frame into dict
    if type(word_freq) == pd.Series:
        counter = Counter(word_freq.fillna(0).to_dict())
    else:
        counter = word_freq

    # filter stop words in frequency counter
    if stopwords is not None:

        counter = {token:freq for (token, freq) in counter.items()
                              if token not in stopwords}
    wc.generate_from_frequencies(counter)

    plt.title(title)

    plt.imshow(wc, interpolation='bilinear')
    plt.axis("off")

# Here, we only apply splitting to the lyrics data due to the difference
# in dataframe/data ingestion between twitter and lycis data:

#def count_words(df, column='tokens', preprocess=None, min_freq=2, split=0):
def count_words(x, preprocess=None, min_freq=2, split=0):

    # process tokens and update counter
    def update(doc):
        tokens = doc if preprocess is None else preprocess(doc)
        counter.update(tokens)

    # create counter and run through all data
    #counter = collections.Counter()
    #top_words_1 = collections.Counter(text_analyze)
    #top_words = top_words_1.most_common(num_words)
    if split == 0:
        counter = collections.Counter(x)
    else:
        counter = collections.Counter(x.split())

    #df[column].map(update)

    # transform counter into data frame
    freq_df = pd.DataFrame.from_dict(counter, orient='index', columns=['freq'])
    freq_df = freq_df.query('freq >= @min_freq')
    freq_df.index.name = 'token'

    return freq_df.sort_values('freq', ascending=False)
```

In [10]:

```
def display_topics(model, features, no_top_words=5):
    for topic, words in enumerate(model.components_):
        total = words.sum()
        largest = words.argsort()[::-1] # invert sort order
        print("\nTopic %02d" % topic)
        for i in range(0, no_top_words):
```

```
            print("  %s (%2.2f)" % (features[largest[i]],
                                    abs(words[largest[i]]*100.0/total)))
```

Load Data from CSV's:

Full Dataset compiled for Training/Test of Classifier

- From CNN, Breitbart, Fox, and Washington Post:

In [154…
```
api_data_complete_df=pd.read_csv('../data/master.csv')
```

Client Dataset to collect Classifier Results for Client

- From The Hill: (already tokenized)

In [13]:
```
api_data_complete_df_business=pd.read_csv('master_business_TheHill.csv')
```

combining articles into one (no need to run again)

In [ ]:
```
api_data_complete_df=pd.read_csv('News_API_TheHill_wordcount3.csv')
api_data_complete_df.rename (columns = {'article_parsed':'article_text'}, inplace=True)
```

In [364…
```
api_data_complete_df_og=pd.read_csv('News_API_TheHill_wordcount.csv')
api_data_complete_df3=pd.read_csv('News_API_TheHill_wordcount3.csv')
Business_frames = [api_data_complete_df_og, api_data_complete_df3]

api_data_complete_df = pd.concat(Business_frames)
api_data_complete_df.rename (columns = {'article_parsed':'article_text'}, inplace=True)
```

In [366…
```
api_data_complete_df.head()
```

Out[366…

	Unnamed: 0	Source	Author	Title	URL	date	content	article_text
0	0	The Hill	Zach Schonfeld	Ketanji Brown Jackson issues solo dissent in r...	https://thehill.com/regulation/court-battles/4...	2023-06-01T15:38:33Z	Skip to content\r\nLiberal Justice Ketanji Bro...	\n\nLiberal Justice Ketanji Brown Jackson issu...
1	1	The Hill	Brett Samuels	How Biden pulled it off...	https://thehill.com/homenews/administration/40...	2023-06-01T06:31:36Z	Skip to content\r\nIn late March, the prospect...	\n\nIn late March, the prospects of President ...
2	2	The Hill	the hill	How Christie could be wildcard in 2024 race...	https://thehill.com/homenews/campaign/4029170-...	2023-06-01T12:00:04Z	The Memo: How Chris Christie could be a wildca...	\n\nFormer New Jersey Gov. Chris Christie is v...
3	3	The Hill	Alexander Bolton	Schumer announces agreement to pass debt ceili...	https://thehill.com/homenews/senate/4031054-sc...	2023-06-01T23:42:11Z	Senate Majority Leader Chuck Schumer (D-N.Y.) ...	\n\nSenate Majority Leader Chuck Schumer (D-N....
4	4	The Hill	Zack Budryk	Kaine introduces amendment to strip Manchin-ba...	https://thehill.com/policy/energy-environment/...	2023-06-01T16:43:29Z	Sen. Tim Kaine (D-Va.) added an amendment to t...	\n\nSen. Tim Kaine (D-Va.) added an amendment ...

In [365…

```
api_data_complete_df.shape
```

Out[365…
```
(181, 9)
```

Pre-Process Data:

(same process done to MASTER and THE_HILL_MASTER):

CHOOSE 1:

Business Data (The Hill)

In [354…
```
# If Business data loaded - this data is already
# tokenized, so no need to pre-process:

#api_data_complete_df=api_data_complete_df_business.copy()
```

Train/Test master Data (CNN, Fox, Breitbart, Washington Post)

In [155…
```
# Train/Test data loaded:
api_data_complete_df=api_data_complete_df.copy()
```

TOKENIZE COLUMN added:

In [218…
```
# Tokenize text:

api_data_complete_df['tokens']= api_data_complete_df['article_text'].apply(prepare,
                                                pipeline=full_pipeline)
```

In [241…
```
api_data_complete_df_business['tokens']= api_data_complete_df_business['article_text'].apply(prepare,
                                                pipeline=full_pipeline)
```

CLEAN DATA (without tokenizing) COLUMN added:

In [177…
```
# Clean data into lowercase/no punctuation:

api_data_complete_df['cleaner_text']= api_data_complete_df['article_text'].apply(prepare,
                                                pipeline=first_pipeline)
```

In [242…
```
api_data_complete_df_business['cleaner_text']= api_data_complete_df_business['article_text'].apply(prepare,
                                                pipeline=first_pipeline)
```

In [158…
```
# These results change depending on the data loaded that is being worked on:

api_data_complete_df.shape
```

Out[158…
```
(4509, 9)
```

If TRAIN data used: LABEL target variable:

In [159…
```
# Add target variable values:

api_data_complete_df['Political_Lean'] = np.where((api_data_complete_df['source_name'] == "CNN") | \
                                                  (api_data_complete_df['source_name'] == \
                                                   "The Washington Post"), 'Left', 'Right')
```

Saving new MASTER CSV;s (tokenized/labeled/etc) for the Business and training datasets

In [369…
```
#api_data_complete_df.to_csv("master_business_TheHill.csv", sep=',')
```

EDA

Add Word Counts Columns:

In [363…
```
api_data_complete_df=pd.read_csv('master_tokenized_labeled_af.csv')
```

In [153…
```
api_data_complete_df.head()
```

Out[153…

	source_name	author	title	url	publish_date	content	article_
0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/...	2023-05-30T16:31:36Z	Travelers in Alabama driving on Interstate 65 ...	Travele Alab drivin Inter
1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0...	2023-05-30T19:56:33Z	Comment on this story\r\nComment\r\nA federal ...	A fe prosec ma neari decis
2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/...	2023-05-30T23:52:34Z	Comment on this story\r\nComment\r\nA federal ...	A fe app Tue clearec
3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0...	2023-05-30T18:30:47Z	Comment on this story\r\nComment\r\nSpeaking i...	Speaki Orlan Nover 2 Repul
5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0...	2023-05-30T10:00:00Z	Comment on this story\r\nComment\r\nLook, we k...	"Lool know cour goi the w

In [178…
```
api_data_complete_df['word_count_tokens'] = api_data_complete_df['tokens'].apply(lambda x: \
                                                    len(str(x).split(" ")))
#api_data_complete_df['word_count'] = api_data_complete_df['article_text'].apply(lambda x: len(str(x).split
```

In [179…
```
api_data_complete_df['word_count_cleaner'] = api_data_complete_df['cleaner_text'].apply(lambda x:\
                                                    len(str(x).split("
```

In [180…
```
api_data_complete_df['word_count_tokens'].mean()
```

Out[180…
```
428.96517302573204
```

In [181…
```
api_data_complete_df = api_data_complete_df[api_data_complete_df['word_count_tokens']>1]
```

Remove "centered" authors from Training/Test Data:

In [182…
```
# Removing Reuters, ms, and Associated Press authors from data:

api_data_complete_df2 = api_data_complete_df[~api_data_complete_df['author'].isin(['msn', 'Associated Press

display(api_data_complete_df2.head())

display(api_data_complete_df2['Political_Lean'].value_counts())
```

	source_name	author	title	url	publish_date	content	article_
0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/...	2023-05-30T16:31:36Z	Travelers in Alabama driving on Interstate 65 ...	Travele Alab drivin Inter
1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0...	2023-05-30T19:56:33Z	Comment on this story\r\nComment\r\nA federal ...	A fe prosec ma neari decis
2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/...	2023-05-30T23:52:34Z	Comment on this story\r\nComment\r\nA federal ...	A fe app Tue cleare
3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0...	2023-05-30T18:30:47Z	Comment on this story\r\nComment\r\nSpeaking i...	Speaki Orlan Nover 2 Repul
5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0...	2023-05-30T10:00:00Z	Comment on this story\r\nComment\r\nLook, we k...	"Lool know cour goi the w

```
Political_Lean
Right    2758
Left     1268
Name: count, dtype: int64
```

In [183…

```
api_data_complete_df2.to_csv("master_tokenized_labeled_af.csv", sep=',')
```

Word Clouds

In [366…

```
# Separate Left and Right Lean articles to analyze as groups:

Left_lean_articles_df=api_data_complete_df2[api_data_complete_df2['Political_Lean']=="Left"]
Right_lean_articles_df=api_data_complete_df2[api_data_complete_df2['Political_Lean']=="Right"]
```

In [367…

```
# Generate list from Left an Right dataframes:

Left_text=[token for sublist in
           Left_lean_articles_df['tokens']
           for token in sublist]
Right_text=[token for sublist in
            Right_lean_articles_df['tokens']
            for token in sublist]
```

In [195…

```
#Left_text
```

In [15]:

```
# Left lean vs Right lean word clouds:

Left_Lean_counts = collections.Counter(Left_text)


Right_Lean_counts = collections.Counter(Right_text)



print("\nLeft Leaning Article's top 5 words:\n")
for HT, count in Left_Lean_counts.most_common(5):
    print(f"{HT}: {count}")
```

```
print("\nRight Leaning Article's top 5 words:\n")
for HT, count in Right_Lean_counts.most_common(5):
    print(f"{HT}: {count}")
```

```
Left Leaning Article's top 5 words:

would: 3591
trump: 3442
also: 2774
house: 2566
people: 2507

Right Leaning Article's top 5 words:

biden: 4453
president: 3767
also: 3621
would: 3512
people: 3375
```

In [18]:

```
Left_Lean_counts = count_words(Left_text, split=0)
Right_Lean_counts = count_words(Right_text, split=0)
display(Left_Lean_counts)
```

token	freq
would	3591
trump	3442
also	2774
house	2566
people	2507
...	...
paints	2
houghton	2
912	2
k5	2
andreeva	2

27475 rows × 1 columns

In [ ]:

```
#her_df=lyrics_data_df.loc[0,:]
#robyn_df=lyrics_data_df.loc[1,:]
```

In [19]:

```
wordcloud(Left_Lean_counts['freq'], title="Left Leaning Article WordCloud", max_words=500)
```

In [20]:

```
wordcloud(Right_Lean_counts['freq'], title="Right Leaning Article WordCloud", max_words=500)
```

Word Clouds reveal that both Left and Right leaning articles tend to have a high word count for their opposition; Left focusing on Trump while the Right focuses on the current president Biden.

Topic Modeling NMF:

In [39]:
```
Left_tfidf_topic = TfidfVectorizer(stop_words=list(sw), min_df=5, max_df=0.7, ngram_range=(1,2))
Left_topic_modeling_input = Left_tfidf_topic.fit_transform(Left_lean_articles_df['cleaner_text'])
```

In [40]:
```
Right_tfidf_topic = TfidfVectorizer(stop_words=list(sw), min_df=5, max_df=0.7, ngram_range=(1,2))
Right_topic_modeling_input = Right_tfidf_topic.fit_transform(Right_lean_articles_df['cleaner_text'])
```

In [49]:
```
nmf_text_model_newsL = NMF(n_components=3, random_state=314)
Left_text_matrix = nmf_text_model_newsL.fit_transform(Left_topic_modeling_input)
HLeft_text_matrix = nmf_text_model_newsL.components_
```

In [50]:
```
nmf_text_model_newsR = NMF(n_components=3, random_state=314)
Right_text_matrix = nmf_text_model_newsR.fit_transform(Right_topic_modeling_input)
HRight_text_matrix = nmf_text_model_newsR.components_
```

Display Topics for Left vs Right:

In [51]:
```
display_topics(nmf_text_model_newsL, Left_tfidf_topic.get_feature_names_out())
```

```
Topic 00
  trump (1.75)
  desantis (1.14)
  president (0.32)
  campaign (0.32)
  former (0.31)

Topic 01
  debt (0.97)
  house (0.63)
  mccarthy (0.61)
  biden (0.60)
  debt ceiling (0.57)

Topic 02
  court (0.21)
  state (0.21)
  police (0.15)
  people (0.15)
  abortion (0.14)
```

In [52]:
```
display_topics(nmf_text_model_newsR, Right_tfidf_topic.get_feature_names_out())
```

```
Topic 00
  biden (0.22)
  debt (0.17)
  house (0.16)
  ai (0.14)
  bill (0.13)

Topic 01
  border (1.72)
  migrants (1.12)
```

```
  title (0.71)
  title 42 (0.65)
  42 (0.64)

Topic 02
  desantis (1.29)
  trump (1.24)
  percent (0.59)
  president (0.41)
  florida (0.39)
```

Topic Modeling using NMF reveals that the top three topics depending on political lean are:

Left:

- Trump/Desantis relationship
- US debt ceiling
- Abortion Laws

Right:

- Bills to change Bill of Rights and regulate AI
- Title 42/Migrants
- Relationship DeSantis and Trump

LDA topic modeling for visualization:

In [368…
```
Lcount_text_vectorizer = CountVectorizer(stop_words=list(sw), min_df=5, max_df=0.7)
Left_count_text_vectors = Lcount_text_vectorizer.fit_transform(Left_lean_articles_df['cleaner_text'])
Left_count_text_vectors.shape

Rcount_text_vectorizer = CountVectorizer(stop_words=list(sw), min_df=5, max_df=0.7)
Right_count_text_vectors = Rcount_text_vectorizer.fit_transform(Right_lean_articles_df['cleaner_text'])
Right_count_text_vectors.shape
```

Out[368…
```
(2758, 13356)
```

In [376…
```
Left_lda_para_model = LatentDirichletAllocation(n_components = 3, random_state=42)
W_lda_para_matrix_Left = Left_lda_para_model.fit_transform(Left_count_text_vectors)
H_lda_para_matrix_Left = Left_lda_para_model.components_
```

In [377…
```
Right_lda_para_model = LatentDirichletAllocation(n_components = 3, random_state=42)
W_lda_para_matrix_Right = Right_lda_para_model.fit_transform(Right_count_text_vectors)
H_lda_para_matrix_Right = Right_lda_para_model.components_
```

In [378…
```
display_topics(Left_lda_para_model, Lcount_text_vectorizer.get_feature_names_out())
```

```
Topic 00
  trump (1.59)
  people (0.49)
  election (0.45)
  president (0.44)
  former (0.38)

Topic 01
  state (0.69)
  people (0.57)
  health (0.44)
  new (0.42)
  abortion (0.39)

Topic 02
  biden (1.01)
  house (0.98)
  republicans (0.67)
```

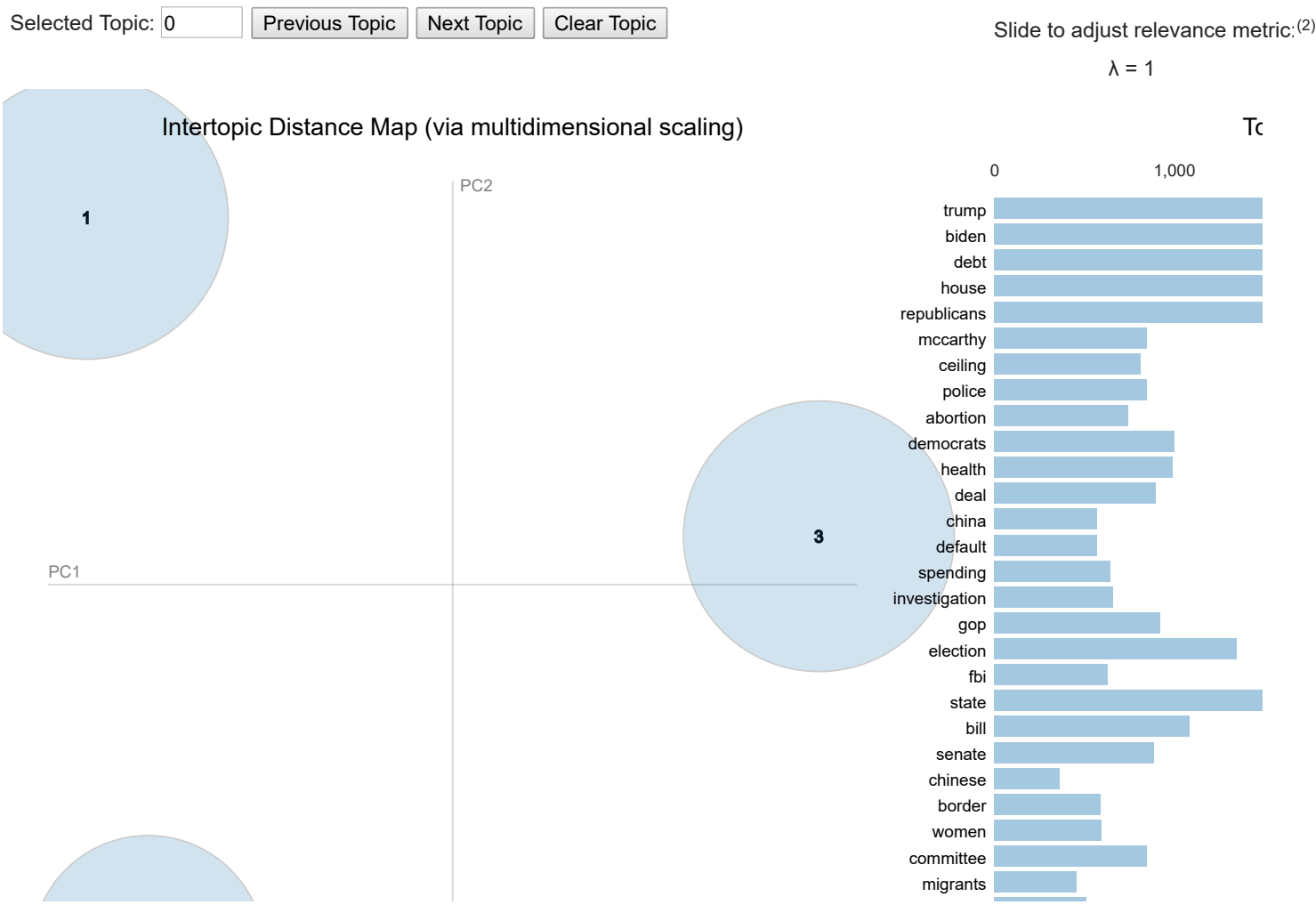
```
  debt (0.67)
  president (0.55)
```

In [379…
```
display_topics(Right_lda_para_model, Rcount_text_vectorizer.get_feature_names_out())
```

```
Topic 00
  police (0.46)
  people (0.39)
  also (0.38)
  ai (0.36)
  one (0.35)

Topic 01
  trump (1.05)
  desantis (0.65)
  president (0.50)
  former (0.50)
  also (0.41)

Topic 02
  biden (1.78)
  border (0.94)
  house (0.88)
  president (0.80)
  would (0.53)
```

In [380…
```
lda_display_left = pyLDAvis.lda_model.prepare(Left_lda_para_model, Left_count_text_vectors,
                                              Lcount_text_vectorizer, sort_topics=False)
pyLDAvis.display(lda_display_left)
```

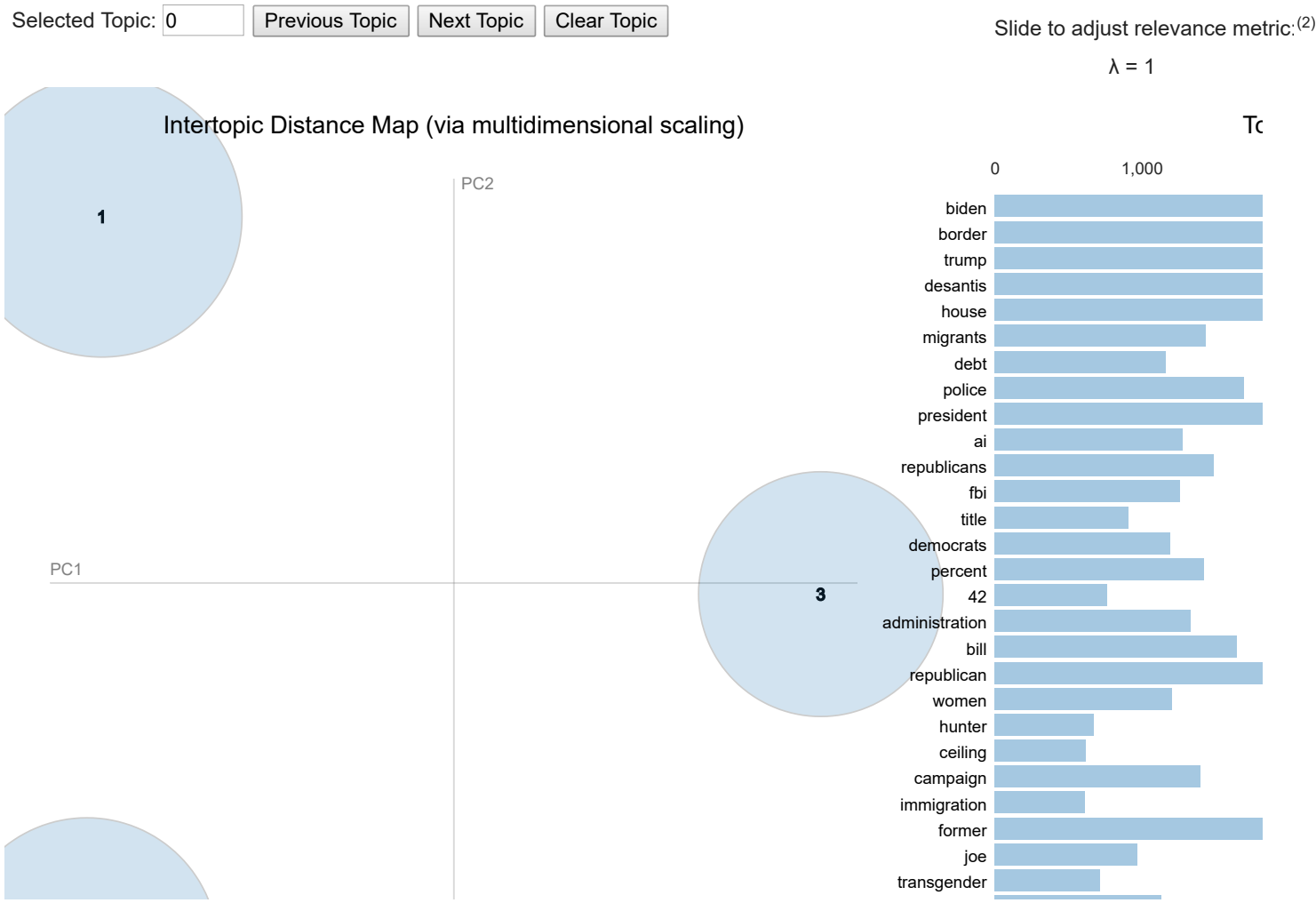
Out[380…



In [381…
```
pyLDAvis.save_html(lda_display_left, 'lda_left.html')
```

In [382…
```
lda_display_right = pyLDAvis.lda_model.prepare(Right_lda_para_model, Right_count_text_vectors,
                                               Rcount_text_vectorizer, sort_topics=False)
pyLDAvis.display(lda_display_right)
```

Out[382…



In [383…

```
pyLDAvis.save_html(lda_display_right, 'lda_right.html')
```

Topic Modeling using LDA reveals that the top three topics depending on political lean are:

Left:

- Healthcare
- McCarthy/Biden Debt Ceiling
- Trump/DeSantis relationship
- Justice system, police

Right:

- Unclear/school-related (the visualization shows this topic overlaps with the following one a bit)
- President, Biden, Trump, politics
- Justice system, fbi
- Immigration

Missing Values:

Only one missing value for training data:

In [223…

```
#api_data_complete_df2.isna().sum()
```

Modeling:

In [57]:
```
# Obtain total counts of words in the entire corpus:

word_dist=Counter([item for sublist in api_data_complete_df['tokens']
                   for item in sublist])
```

Prepare data for modeling (Target defined, clean text as X)

In [46]:
```
# generate the dict data to then be filtered through
# the filtered_words list:

news_data=dict()


news_data=[(api_data_complete_df2.at[index,'cleaner_text'],
            api_data_complete_df2.at[index,'Political_Lean']) \
           for (index,row) in api_data_complete_df2.iterrows()]
```

Feature Words filtering:

In [47]:
```
word_cutoff=5
feature_words2=[]
for word, count in word_dist.items() :
    #for convention_data_df['word_dist']
    if count > word_cutoff :
        feature_words2.append(word)

print(f"With a word cutoff of {word_cutoff}, we have {len(feature_words2)} as features in the model.")
```

```
With a word cutoff of 5, we have 22181 as features in the model.
```

In [48]:
```
# Filter the data through the feature words set
# determined above:

featuresets=dict()
featuresets = [(conv_features(text,feature_words2),
                lean) for (text, lean) in news_data]
```

Rough Naive-Bayes Classifier setup and run-through:

In [49]:
```
random.seed(42)
random.shuffle(featuresets)
test_size = 20
```

In [50]:
```
test_set=dict()
train_set=dict()
test_set, train_set = featuresets[:test_size], featuresets[test_size:]
```

In [51]:
```
classifier = nltk.NaiveBayesClassifier.train(train_set)
print(nltk.classify.accuracy(classifier, test_set))
```

```
0.2
```

In [52]:
```
# Confusion Matrix:

predicted_labels = [classifier.classify(features) for features,
                    label in test_set]
gold_labels = [label for features, label in test_set]

cm = ConfusionMatrix(gold_labels, predicted_labels)
print(cm.pretty_format(sort_by_count=True, show_percents=True))
```

```
      |      R       |
      |      i       |
      |      g       |
      |      h       |
      |      t       |
------+--------------+
```

```
Right |     <.> 80.0% |
 Left |      . <20.0%>|
------+---------------+
(row = reference; col = test)
```

Linear SVC classifier:

Prepare Data for Linear SVC (SVM) implementation:

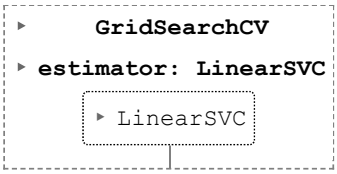
In [184…
```
X_train, X_test, Y_train, Y_test = train_test_split(api_data_complete_df2['cleaner_text'],
                                                    api_data_complete_df2['Political_Lean'],
                                                    test_size=0.2,
                                                    random_state=42,
                                                    stratify=api_data_complete_df2['Political_Lean'])

print('Size of Training Data ', X_train.shape[0])
print('Size of Test Data ', X_test.shape[0])
```

```
Size of Training Data  3220
Size of Test Data  806
```

In [185…
```
# TF-IDF Vectorization for Training/Test Data:

tfidf = TfidfVectorizer(min_df = 10, max_df=0.7, ngram_range=(1,2), stop_words=sw)
X_train_tf = tfidf.fit_transform(X_train)
X_test_tf = tfidf.transform(X_test)
```

Hyperparameter tuning:

In [361…
```
parameters = {'C':[0.1, 1, 6, 6.5, 8], 'penalty':['l1', 'l2'], 'tol':[0.0000001,
                                                                      0.00001, 0.0001, 0.001, 0.01]}
svc_grid = LinearSVC(random_state=0)
clf_grid = GridSearchCV(svc_grid, parameters)
clf_grid.fit(X_train_tf, Y_train)
```

Out[361…


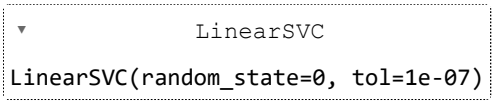
In [362…
```
clf_grid.best_params_
```

Out[362…
```
{'C': 6, 'penalty': 'l2', 'tol': 1e-07}
```

The hyperparameter that gave the best results was tolerance at 1e-07, the others brought accuracy down so eliminated them

Train and Test Linear SVC model:

In [359…
```
# Train Lineary SVC Model with Training Data:

model1 = LinearSVC(random_state=0, tol=1e-7)
model1.fit(X_train_tf, Y_train)
```

Out[359…


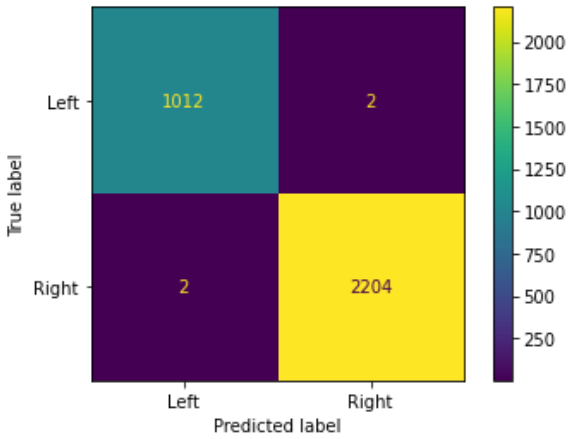
In [389…
```
Y_train_pred = model1.predict(X_train_tf)

print ('Training Accuracy Score - ', accuracy_score(Y_train, Y_train_pred))
```

```
Training Accuracy Score -  0.9987577639751553
```

In [436…
```
train_svc_pred_cm = confusion_matrix(Y_train, Y_train_pred)
```

```
print(classification_report(Y_train, Y_train_pred))
print(train_svc_pred_cm)

train_svc_pred_cm_calc = ConfusionMatrixDisplay(confusion_matrix=train_svc_pred_cm,
                                                display_labels=model1.classes_)
train_svc_pred_cm_calc.plot()
plt.show()
```

```
              precision    recall  f1-score   support

        Left       1.00      1.00      1.00      1014
       Right       1.00      1.00      1.00      2206

    accuracy                           1.00      3220
   macro avg       1.00      1.00      1.00      3220
weighted avg       1.00      1.00      1.00      3220

[[1012    2]
 [   2 2204]]
```



Test Results Linear SVC:

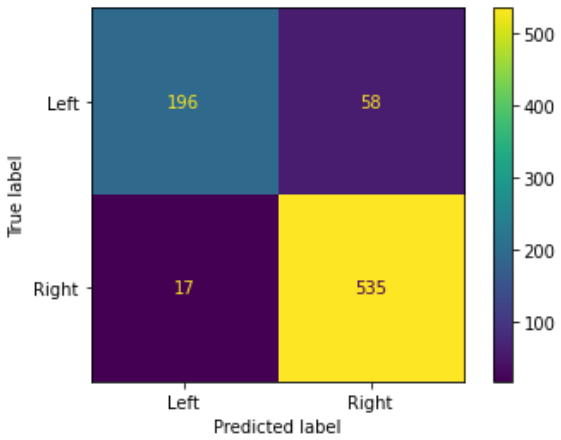
In [360…

```
Y_pred = model1.predict(X_test_tf)
print ('Accuracy Score - ', accuracy_score(Y_test, Y_pred))
#print ('F1 Score - ', recall_score(Y_test, Y_pred))
```

```
Accuracy Score -  0.9069478908188585
```

In [188…

```
Y_pred = model1.predict(X_test_tf)
confusion_matrix(Y_test, Y_pred)
cm = confusion_matrix(Y_test, Y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels = model1.classes_)
disp.plot()
plt.show()
```



In [189…

```
TNmodel1=cm[0][0]
FPmodel1=cm[0][1]
FNmodel1=cm[1][0]
TPmodel1=cm[1][1]
```

In [190…

```
# Results:

TANmodel1=TNmodel1+FPmodel1
TAPmodel1=TPmodel1+FNmodel1
TPPmodel1=FPmodel1+TPmodel1
TPNmodel1=TNmodel1+FNmodel1
GTmodel1=TANmodel1+TAPmodel1
AccuracyM1=(TNmodel1+TPmodel1)/GTmodel1
ErrorRateM1=1-AccuracyM1
SensitivityM1=TPmodel1/(TAPmodel1)
RecallM1=SensitivityM1
SpecificityM1=TNmodel1/TANmodel1
PrecisionM1=TPmodel1/TPPmodel1
F1M1=2*PrecisionM1*RecallM1/(PrecisionM1 + RecallM1)
F2M1=5*(PrecisionM1*RecallM1)/((4*PrecisionM1)+RecallM1)
Fp5M1=(1.25)*(PrecisionM1*RecallM1)/((0.25*PrecisionM1)+RecallM1)

header = ["Accuracy", "Error Rate", "Sensitivity", "Recall", "Specificity",
          "Precision", "F1", "F2", "F0.5"]
data1 = [["Accuracy", AccuracyM1], ["Error Rate", ErrorRateM1],
         ["Sensitivity", SensitivityM1],
         ["Recall", RecallM1], ["Specificity", SpecificityM1],
         ["Precision", PrecisionM1],
         ["F1", F1M1], ["F2", F2M1], ["F0.5", Fp5M1]]

col_names=["Measurement", "Linear SVC Model"]

ModelEvaluationTable = tabulate(data1, headers=col_names,
                                tablefmt="fancy_grid")

print(ModelEvaluationTable)
```

```
╒═══════════════╤═══════════════════╕
│ Measurement   │   Linear SVC Model │
╞═══════════════╪═══════════════════╡
│ Accuracy      │          0.906948 │
├───────────────┼───────────────────┤
│ Error Rate    │          0.0930521 │
├───────────────┼───────────────────┤
│ Sensitivity   │          0.969203 │
├───────────────┼───────────────────┤
│ Recall        │          0.969203 │
├───────────────┼───────────────────┤
│ Specificity   │          0.771654 │
├───────────────┼───────────────────┤
│ Precision     │          0.902192 │
├───────────────┼───────────────────┤
│ F1            │          0.934498 │
├───────────────┼───────────────────┤
│ F2            │          0.955016 │
├───────────────┼───────────────────┤
│ F0.5          │          0.914843 │
╘═══════════════╧═══════════════════╛
```

In [299…

```
data1
```

Out[299…

```
[['Accuracy', 0.9069478908188585],
 ['Error Rate', 0.09305210918114148],
 ['Sensitivity', 0.9692028985507246],
 ['Recall', 0.9692028985507246],
 ['Specificity', 0.7716535433070866],
 ['Precision', 0.9021922428330523],
 ['F1', 0.9344978165938865],
 ['F2', 0.9550160656908249],
 ['F0.5', 0.914842681258551]]
```

In [303…

```
Data_metric_results_TheHill=pd.DataFrame(data1)
Data_metric_results_TheHill.head()
```

Out[303…

	0	1
0	Accuracy	0.906948
1	Error Rate	0.093052

	0	1
2	Sensitivity	0.969203
3	Recall	0.969203
4	Specificity	0.771654

In [304…

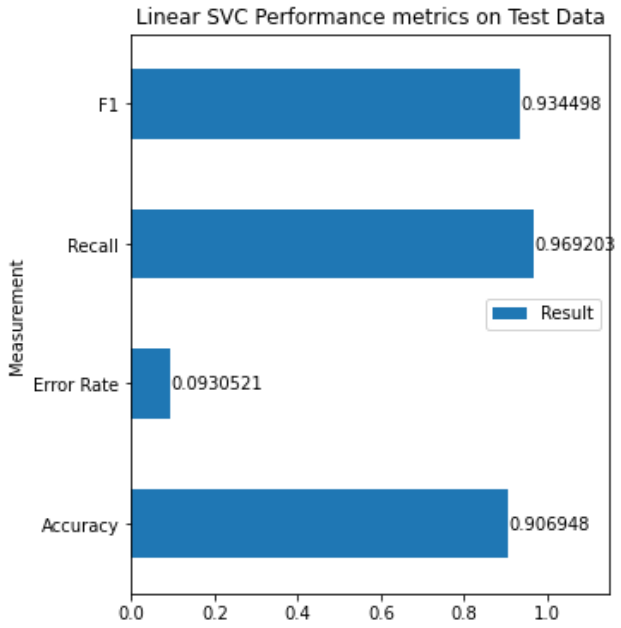
```
Data_metric_results_TheHill.rename (columns = {0:'Measurement'}, inplace=True)
Data_metric_results_TheHill.rename (columns = {1:'Result'}, inplace=True)
```

In [319…

```
#plt.bar(x=ModelEvaluationTable)


ax=Data_metric_results_TheHill[(Data_metric_results_TheHill['Measurement'] == 'Accuracy') |
                               (Data_metric_results_TheHill['Measurement'] == 'Recall') |
                               (Data_metric_results_TheHill['Measurement'] == 'F1') |
                               (Data_metric_results_TheHill['Measurement'] == \
                                'Error Rate')].plot(kind="barh",
                                                    x='Measurement',
                                    figsize=(5,6),
                                    title='Linear SVC Performance metrics on Test Data')
ax.bar_label(ax.containers[0])
ax.set_xlim(right=1.15)
```

Out[319…

```
(0.0, 1.15)
```



Because the classifier has a harder time classifying the Left class, our Accuracy is lower than Recall for the Right class. But still within acceptable range. F1 score is above .9 which was the goal. This classifier is good to try with the business case data give its speed and simpler design.

Save trained SVC model:

In [386…

```
# Path to save the pickled model
file_path = "model1_linSVC.pkl"

# Pickle the model
with open(file_path, "wb") as file:
    pickle.dump(model1, file)

print("Model pickled and saved successfully.")
```

```
Model pickled and saved successfully.
```

SKLEARN SVC Proba Model:

In [363…

```
svm = SVC(random_state=0, kernel='linear', probability=True)
```

```
svm.fit(X_train_tf, Y_train)
```

Out[363…

```
SVC(kernel='linear', probability=True, random_state=0)
```

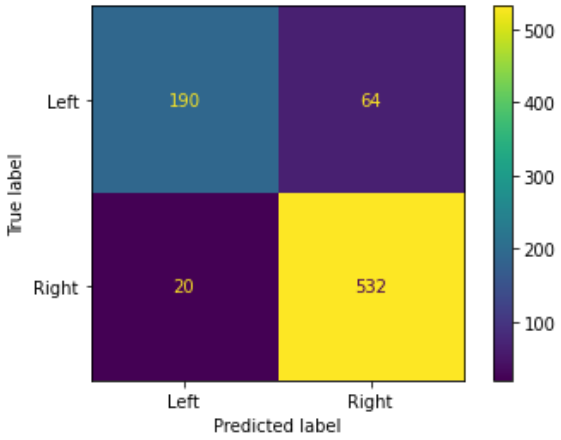
In [364…

```
Y_pred_sklearn = svm.predict(X_test_tf)
print ('Accuracy Score - ', accuracy_score(Y_test, Y_pred_sklearn))
```

```
Accuracy Score -  0.8957816377171216
```

In [203…

```
#confusion_matrix(Y_test, Y_pred_sklearn)
cm_sklearn = confusion_matrix(Y_test, Y_pred_sklearn)
disp_sklearn = ConfusionMatrixDisplay(confusion_matrix=cm_sklearn, display_labels = svm.classes_)
disp_sklearn.plot()
plt.show()
```



In [204…

```
SK_TNmodel1=cm_sklearn[0][0]
SK_FPmodel1=cm_sklearn[0][1]
SK_FNmodel1=cm_sklearn[1][0]
SK_TPmodel1=cm_sklearn[1][1]
```

In [205…

```
# Results:

SK_TANmodel1=SK_TNmodel1+SK_FPmodel1
SK_TAPmodel1=SK_TPmodel1+SK_FNmodel1
SK_TPPmodel1=SK_FPmodel1+SK_TPmodel1
SK_TPNmodel1=SK_TNmodel1+SK_FNmodel1
SK_GTmodel1=SK_TANmodel1+SK_TAPmodel1
SK_AccuracyM1=(SK_TNmodel1+SK_TPmodel1)/SK_GTmodel1
SK_ErrorRateM1=1-SK_AccuracyM1
SK_SensitivityM1=SK_TPmodel1/(SK_TAPmodel1)
SK_RecallM1=SK_SensitivityM1
SK_SpecificityM1=SK_TNmodel1/SK_TANmodel1
SK_PrecisionM1=SK_TPmodel1/SK_TPPmodel1
SK_F1M1=2*SK_PrecisionM1*SK_RecallM1/(SK_PrecisionM1 + SK_RecallM1)
SK_F2M1=5*(SK_PrecisionM1*SK_RecallM1)/((4*SK_PrecisionM1)+SK_RecallM1)
SK_Fp5M1=(1.25)*(SK_PrecisionM1*SK_RecallM1)/((0.25*SK_PrecisionM1)+SK_RecallM1)

header = ["Accuracy", "Error Rate", "Sensitivity", "Recall", "Specificity",
          "Precision", "F1", "F2", "F0.5"]
SK_data1 = [["Accuracy", SK_AccuracyM1], ["Error Rate", SK_ErrorRateM1],
         ["Sensitivity", SK_SensitivityM1],
         ["Recall", SK_RecallM1], ["Specificity", SK_SpecificityM1],
         ["Precision", SK_PrecisionM1],
         ["F1", SK_F1M1], ["F2", SK_F2M1], ["F0.5", SK_Fp5M1]]

col_names=["Measurement", "SKLEARN Linear SVC Model"]

SK_ModelEvaluationTable = tabulate(SK_data1, headers=col_names,
                                   tablefmt="fancy_grid")

print(SK_ModelEvaluationTable)
```

Measurement	SKLEARN Linear SVC Model
Accuracy	0.895782
Error Rate	0.104218
Sensitivity	0.963768
Recall	0.963768
Specificity	0.748031
Precision	0.892617
F1	0.926829
F2	0.948645
F0.5	0.905995

Features of Importance:

In [456…

```
# Get the most predictive words

feature_names = tfidf.get_feature_names_out()
#model1.coef_
top_words = sorted(zip(model1.coef_[0], feature_names), reverse=True)[:10]
#svm.coe
#top_words
```

In [457…

```
top_words_df=pd.DataFrame(top_words)

top_words_df.rename (columns = {0:'var_imp'}, inplace=True)
top_words_df.rename (columns = {1:'feature'}, inplace=True)
```

In [458…

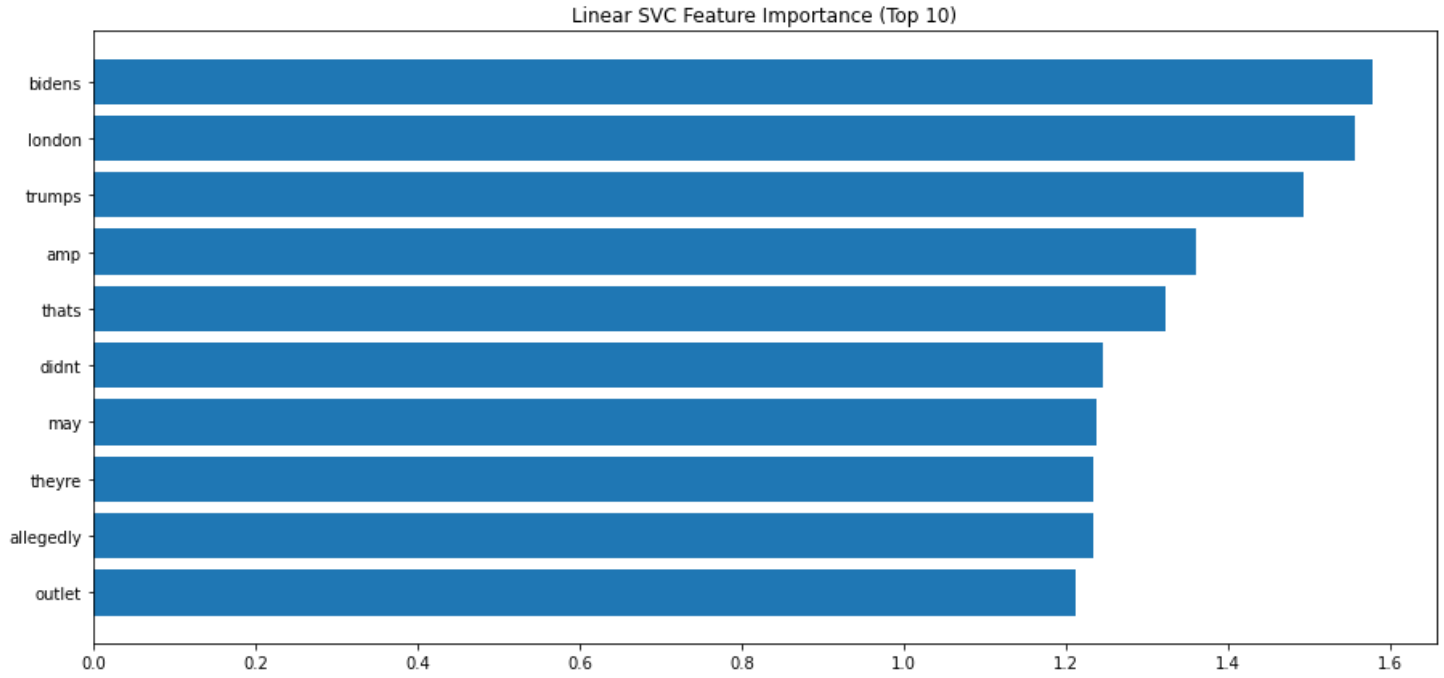
```
top_words_df.head()
```

Out[458…

	var_imp	feature
0	1.578366	bidens
1	1.556567	london
2	1.493616	trumps
3	1.359869	amp
4	1.322696	thats

In [460…

```
top_words_df.sort_values('var_imp',inplace=True)
plt.figure(figsize=(15,7))
plt.title('Linear SVC Feature Importance (Top 10)')
plt.barh([x for x in range(len(top_words_df['var_imp']))], top_words_df['var_imp'],
         tick_label=top_words_df['feature'])
plt.show()
```



These words are difficult to analyze without context. However, they help us know if there are any words that are indicative more of the journal/attributes of the news source that might give away the label instead of actual words in the articles that are relevant to political lean. We used this and found that certain phrases were indicating Fox News or other sources which might have helped drive accuracy deceptively high. This output helps us know that it is the text within articles that have helped classify the articles and develop our classifier.

Apply Classifier to Business Client Data:

In [243…
```
#Centered data Prep classification:

X_data = api_data_complete_df_business['cleaner_text']
```

In [244…
```
# TF-IDF Vectorization for Business application:

X_Data_Centered = tfidf.transform(X_data)
```

Linear SVD:

In [245…
```
Data_Classification_Centered = model1.predict(X_Data_Centered)
```

In [246…
```
print(api_data_complete_df_business['Author'])
```

```
0         Zach Schonfeld
1          Brett Samuels
2               the hill
3       Alexander Bolton
4            Zack Budryk
              ...
176         Tobias Burns
177        Julia Mueller
178       Caroline Vakil
179         Tobias Burns
180        Lauren Sforza
Name: Author, Length: 181, dtype: object
```

In [247…
```
Data_Classification_Centered
```

Out[247…
```
array(['Left', 'Left', 'Right', 'Right', 'Right', 'Right', 'Right',
       'Left', 'Left', 'Right', 'Left', 'Left', 'Right', 'Left', 'Left',
       'Left', 'Left', 'Right', 'Right', 'Right', 'Right', 'Left',
       'Right', 'Left', 'Left', 'Right', 'Left', 'Right', 'Right',
```

```
       'Right', 'Left', 'Left', 'Right', 'Left', 'Left', 'Right', 'Left',
       'Left', 'Right', 'Right', 'Left', 'Right', 'Left', 'Left', 'Left',
       'Right', 'Right', 'Right', 'Left', 'Right', 'Left', 'Right',
       'Right', 'Left', 'Left', 'Left', 'Right', 'Right', 'Left', 'Right',
       'Left', 'Right', 'Right', 'Left', 'Right', 'Left', 'Right', 'Left',
       'Left', 'Left', 'Right', 'Right', 'Left', 'Right', 'Left', 'Right',
       'Right', 'Right', 'Right', 'Right', 'Right', 'Left', 'Right',
       'Right', 'Left', 'Left', 'Left', 'Left', 'Right', 'Right', 'Left',
       'Right', 'Right', 'Right', 'Right', 'Left', 'Left', 'Left',
       'Right', 'Left', 'Left', 'Right', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Right', 'Left', 'Right', 'Right', 'Right', 'Left',
       'Right', 'Left', 'Right', 'Left', 'Right', 'Left', 'Right', 'Left',
       'Right', 'Left', 'Right', 'Left', 'Right', 'Right', 'Left',
       'Right', 'Right', 'Right', 'Right', 'Right', 'Right', 'Right',
       'Right', 'Right', 'Left', 'Left', 'Right', 'Left', 'Left', 'Left',
       'Left', 'Right', 'Right', 'Right', 'Right', 'Right', 'Left',
       'Right', 'Left', 'Right', 'Right', 'Right', 'Left', 'Right',
       'Left', 'Left', 'Right', 'Right', 'Right', 'Left', 'Right',
       'Right', 'Right', 'Left', 'Left', 'Right', 'Left', 'Left', 'Right',
       'Right', 'Right', 'Left', 'Left', 'Right', 'Right', 'Right',
       'Right', 'Right'], dtype=object)
```

In [248…

```
left=0
right=0
for ii in range(0,len(Data_Classification_Centered)):
    if (Data_Classification_Centered[ii] == 'Left'):
        left=left+1
    elif (Data_Classification_Centered[ii] == 'Right'):
        right=right+1

print('Number of Left Articles: ', left, '\nNumber of Right Articles: ',
      right)
```

```
Number of Left Articles:  81
Number of Right Articles:  100
```

In [249…

```
Data_centered_results=pd.DataFrame(Data_Classification_Centered)
```

In [250…

```
Data_centered_results.rename (columns = {0:'Prediction'}, inplace=True)
```

In [251…

```
Data_centered_results.head()
```

Out[251…

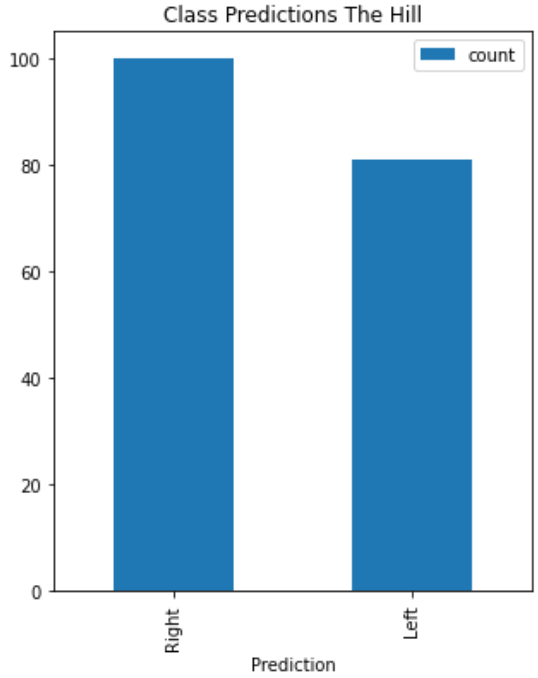
	Prediction
0	Left
1	Left
2	Right
3	Right
4	Right

In [252…

```
Data_centered_results['Prediction'].value_counts().plot(kind="bar",
                                    legend=True,
                                    figsize=(5,6),
                                    title='Class Predictions The Hill')
```

Out[252…

```
<AxesSubplot:title={'center':'Class Predictions The Hill'}, xlabel='Prediction'>
```



Even with a small dataset of 181 articles, we see almost an even distribution of Right and Left leaning classifications for The Hill. We are interested in seeing the confidence (probability ratings) for these classifications below.

SKLEARN SVC for probability testing:

In [253…
```
AP_predictions_proba = svm.predict_proba(X_Data_Centered)
```

In [254…
```
AP_predictions_proba[0][0]
```

Out[254…
```
0.9503953866731141
```

In [208…
```
len(AP_predictions_proba)
```

Out[208…
```
181
```

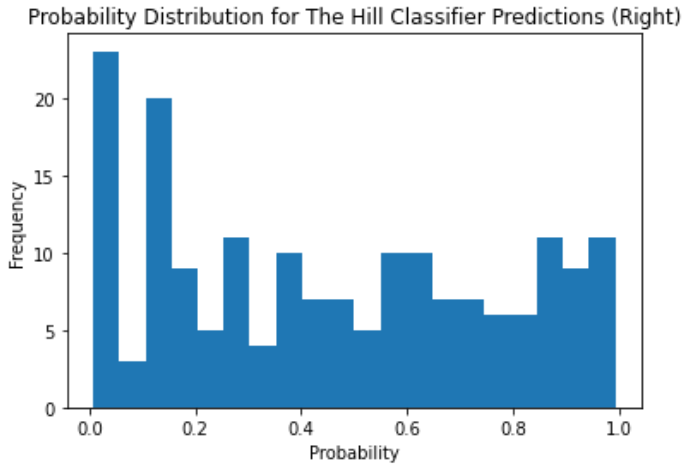
In [255…
```
#from sklearn.svm import SVC
#svm = SVC(kernel='linear', probability=True)
#svm.fit(X_train_tf, Y_train)

#AP_predictions_proba = svm.predict_proba(X_Data_Centered)

for pa in range(0,len(AP_predictions_proba)):
    for pa2 in range(0,1):
        if (AP_predictions_proba[pa][pa2] <= 0.65) & (AP_predictions_proba[pa][pa2] >= 0.5):
            print(pa, ':', AP_predictions_proba[pa][pa2], Data_Classification_Centered[pa])
```

```
2 : 0.5973377294016616 Right
5 : 0.505540954908968 Right
18 : 0.5137707984203023 Right
27 : 0.5645007945373783 Right
38 : 0.5801257602633187 Right
39 : 0.5756885039402224 Right
46 : 0.5418546159064842 Right
75 : 0.6216635081105005 Right
77 : 0.6216635081105005 Right
84 : 0.6413191667751802 Left
105 : 0.6412982042489684 Left
106 : 0.6412982042489684 Left
115 : 0.5 Right
119 : 0.5 Right
136 : 0.5857159103879271 Right
147 : 0.5893703782219359 Right
150 : 0.5547001230087428 Right
```

```
157 : 0.6448261346071565 Left
166 : 0.6448261346071565 Left
168 : 0.6242723322311122 Right
172 : 0.616816668944659 Right
175 : 0.611971953429539 Left
176 : 0.5 Right
179 : 0.5 Right
```

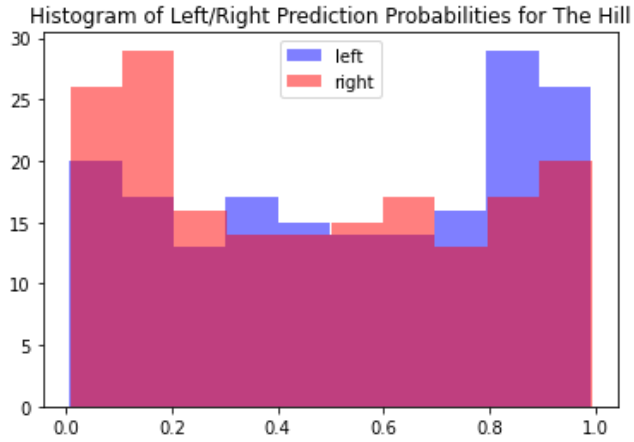
In [296…

```
#AP_predictions_proba

plt.hist(AP_predictions_proba[:, 1], bins=20)
plt.xlabel('Probability')
plt.ylabel('Frequency')
plt.title('Probability Distribution for The Hill Classifier Predictions (Right)')
plt.show()
```



In [481…

```
# Plotting histograms
plt.hist(AP_predictions_proba[:, 0], bins=10,
         alpha=0.5, color='blue', label='left')
plt.hist(AP_predictions_proba[:, 1], bins=10,
         alpha=0.5, color='red', label='right')

# Adding legend and title
plt.legend()
plt.title('Histogram of Left/Right Prediction Probabilities for The Hill')

# Displaying the plot
plt.show()
```



As we had hoped, The Hill appears to be much more balanced in their political lean, as is confirmed by the AllSides data. We did find a slight increase in Right leaning articles, but upon observation of the probability distribution, it is apparant that there are many low probability (low confidence) classifications for the lean. This distribution reveals that even when there is a lean classification, there is not an overwhelming number of confident classifications, which means The Hill is probably still a Centered online publication.

Apply Classifier to AP data:

Collect data saved in CSV and preprocess:

In [261…
```
api_data_complete_df_AP=pd.read_csv('News_API_AP_wordcount.csv')
```

In [262…
```
# Tokenize text:

api_data_complete_df_AP['tokens']= api_data_complete_df_AP['article_parsed'].apply(prepare,
                                                    pipeline=full_pipeline)
# Clean data into lowercase/no punctuation:

api_data_complete_df_AP['cleaner_text']= api_data_complete_df_AP['article_parsed'].apply(prepare,
                                                    pipeline=first_pipeline)
api_data_complete_df_AP['word_count_tokens'] = api_data_complete_df_AP['tokens'].apply(lambda x:\
                                                                                       len(str(x).\
                                                                                           split(" ")))

api_data_complete_df_AP = api_data_complete_df_AP[api_data_complete_df_AP['word_count_tokens']>1]

api_data_complete_df_AP.to_csv("master_tokenized_AP.csv", sep=',')
```

In [269…
```
#Centered data Prep classification:

X_data_AP = api_data_complete_df_AP['cleaner_text']
```

In [270…
```
# TF-IDF Vectorization for Business application:

X_Data_Centered_AP = tfidf.transform(X_data_AP)
```

Linear SVD:

In [271…
```
Data_Classification_Centered_AP = model1.predict(X_Data_Centered_AP)
```

In [273…
```
print(api_data_complete_df_AP['Author'])
```

```
0              By ALANNA DURKIN RICHER - Associated Press
1                        By SEUNG MIN KIM Associated Press
2                                           Bernard Condon
3      By KIMBERLEE KRUESI, SARAH RANKIN and DENISE L...
4                                      Brendan Farrington
                              ...
183    By LISA MASCARO, KEVIN FREKING and STEPHEN GRO...
184                  By FARNOUSH AMIRI - Associated Press
185               By RADUL RADOVANOVIC - Associated Press
186                  By FARNOUSH AMIRI - Associated Press
187    By LISA MASCARO, KEVIN FREKING and STEPHEN GRO...
Name: Author, Length: 188, dtype: object
```

In [274…
```
Data_Classification_Centered_AP
```

Out[274…
```
array(['Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Right',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Right', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Right', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Right', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Right', 'Left', 'Left', 'Left', 'Right', 'Right',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Right', 'Left', 'Right', 'Left', 'Right',
       'Left', 'Left', 'Left', 'Right', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Right', 'Left', 'Left', 'Left', 'Left',
       'Right', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Right', 'Left', 'Right', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Right', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Right', 'Left', 'Left', 'Left',
```

```
       'Left', 'Left', 'Left', 'Right', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left', 'Left',
       'Right', 'Right', 'Left', 'Right', 'Left', 'Left', 'Right', 'Left',
       'Left', 'Left', 'Left', 'Right', 'Left', 'Left', 'Right', 'Left',
       'Left', 'Left', 'Left', 'Left'], dtype=object)
```

In [275…

```
left=0
right=0
for ii in range(0,len(Data_Classification_Centered_AP)):
    if (Data_Classification_Centered_AP[ii] == 'Left'):
        left=left+1
    elif (Data_Classification_Centered_AP[ii] == 'Right'):
        right=right+1

print('Number of Left Articles: ', left, '\nNumber of Right Articles: ',
      right)
```

```
Number of Left Articles:  164
Number of Right Articles:  24
```

In [276…

```
Data_centered_results_AP=pd.DataFrame(Data_Classification_Centered_AP)
```

In [277…

```
Data_centered_results_AP.rename (columns = {0:'Prediction'}, inplace=True)
```

In [278…

```
Data_centered_results_AP.head()
```

Out[278…

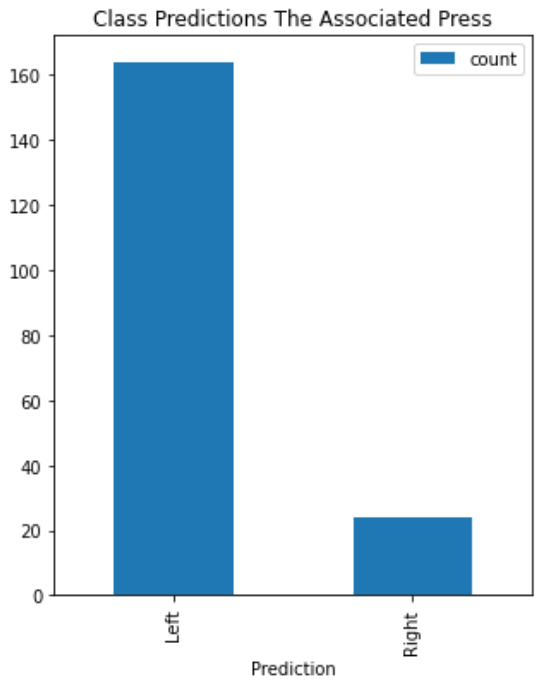
	Prediction
0	Left
1	Left
2	Left
3	Left
4	Left

In [279…

```
Data_centered_results_AP['Prediction'].value_counts().plot(kind="bar",
                                    legend=True,
                                    figsize=(5,6),
                                    title='Class Predictions The Associated Press')
```

Out[279…

```
<AxesSubplot:title={'center':'Class Predictions The Associated Press'}, xlabel='Prediction'>
```



An overwhelming majority of Associated press articles were categorized as Left Leaning.

SKLEARN SVC for probability testing:

In [280…
```
AP_predictions_proba_AP = svm.predict_proba(X_Data_Centered_AP)
```

In [281…
```
AP_predictions_proba_AP[0][0]
```

Out[281…
```
0.9508323188246903
```

In [292…
```
AP_predictions_proba_AP[1][0]
```

Out[292…
```
0.981768860823333
```

In [294…
```
for x in range(0,len(AP_predictions_proba_AP)):
    print(AP_predictions_proba_AP[x][0])
    print(Data_Classification_Centered_AP[x])
```

```
0.9508323188246903
Left
0.981768860823333
Left
0.9614810136661492
Left
0.98661746903831
Left
0.891259423999621
Left
0.9906278481561918
Left
0.9427318115573183
Left
0.5336534447481074
Right
0.98661746903831
Left
0.936648044221054
Left
0.981768860823333
Left
0.9630019060086964
Left
0.6940622912199284
Left
0.803178366915479
Left
0.9508323188246903
Left
0.784522092358475
Left
0.9808996379009253
Left
0.26829051965364115
Right
0.9972611023492843
Left
0.9183765606311858
Left
0.9972611023492843
Left
0.9902424311393712
Left
0.9972611023492843
Left
0.9183765606311858
Left
0.8461805772804466
Left
0.9902424311393712
Left
```

0.08359528360983542
Right
0.9183765606311858
Left
0.7147570033831585
Left
0.7832130005086422
Left
0.7147570033831585
Left
0.9865357100592266
Left
0.7832130005086422
Left
0.7147570033831585
Left
0.9865357100592266
Left
0.278292939996084
Right
0.7096360056069901
Left
0.9611433318272665
Left
0.95338868265243
Left
0.9822489551465043
Left
0.7462156028150145
Left
0.9728761475398888
Left
0.9611433318272665
Left
0.8752703633993194
Left
0.9822489551465043
Left
0.9420684108893576
Left
0.95338868265243
Left
0.7096360056069901
Left
0.6855629559539462
Left
0.7462156028150145
Left
0.9966482796096178
Left
0.9420684108893576
Left
0.9822489551465043
Left
0.9148257284353466
Left
0.9254239127575978
Left
0.84331649453353
Left
0.9931531162628318
Left
0.9894081760478759
Left
0.44982791103942854
Right
0.9148257284353466
Left
0.9254239127575978
Left
0.9894081760478759
Left
0.46262772324766915
Right
0.44982791103942854
Right
0.9917164439949772

Left
0.9937696130487768
Left
0.7890964292773458
Left
0.9338817375901171
Left
0.9359899071544867
Left
0.9917164439949772
Left
0.9209507441746894
Left
0.9937696130487768
Left
0.8195357680267555
Left
0.8787682426295289
Left
0.9338817375901171
Left
0.35881962996488825
Right
0.8787682426295289
Left
0.39819564396189217
Right
0.9684504456091689
Left
0.37121384584459777
Right
0.9936183822776451
Left
0.9743476710605173
Left
0.9899141547920952
Left
0.6174728227984922
Right
0.8021810684713792
Left
0.9899141547920952
Left
0.9684504456091689
Left
0.9402702376634287
Left
0.9936183822776451
Left
0.5516811229058828
Left
0.9196391841976358
Left
0.37121384584459777
Right
0.7540110657838195
Left
0.9402702376634287
Left
0.9743476710605173
Left
0.9062226514282471
Left
0.3820067013624183
Right
0.9772896476217972
Left
0.9359789272722828
Left
0.9885503134330745
Left
0.9859512746016549
Left
0.918172484567357
Left
0.926695814388605
Left

0.9772896476217972
Left
0.9885503134330745
Left
0.4904534100495921
Right
0.9859512746016549
Left
0.4904534100495921
Right
0.9840803960697755
Left
0.9830953251625819
Left
0.8246320108724589
Left
0.9339851192219057
Left
0.9742210946142087
Left
0.9340226402633245
Left
0.9830953251625819
Left
0.9840803960697755
Left
0.9877542937101121
Left
0.8369809868732206
Left
0.9529389556665637
Left
0.9339851192219057
Left
0.9165376832636434
Left
0.5888748501334558
Left
0.9742210946142087
Left
0.8246320108724589
Left
0.9340226402633245
Left
0.8369809868732206
Left
0.5888748501334558
Left
0.9610896091745622
Left
0.8440574965883695
Left
0.9789949454217545
Left
0.8187578291283613
Left
0.9950459775743616
Left
0.7701341569045096
Left
0.053833868875404915
Right
0.9610896091745622
Left
0.9789949454217545
Left
0.9950459775743616
Left
0.7701341569045096
Left
0.8440574965883695
Left
0.9789949454217545
Left
0.15366794586119084
Right
0.9986985071050143

Left
0.9087452087801907
Left
0.9986985071050143
Left
0.9087452087801907
Left
0.9892408164327222
Left
0.9087452087801907
Left
0.18927868289788397
Right
0.9892408164327222
Left
0.7060289973370131
Left
0.8516094144446876
Left
0.905086157170679
Left
0.8516094144446876
Left
0.9432652689556511
Left
0.6177850658464895
Left
0.4646506935547059
Left
0.990994520082568
Left
0.8820320456271037
Left
0.8916856427246771
Left
0.990994520082568
Left
0.8916856427246771
Left
0.990994520082568
Left
0.9603541164657194
Left
0.9426983425095306
Left
0.9426983425095306
Left
0.9603541164657194
Left
0.963316826150573
Left
0.9426983425095306
Left
0.5747207232364414
Right
0.4093535342776969
Right
0.989238652072837
Left
0.5747207232364414
Right
0.989238652072837
Left
0.5757840366788696
Left
0.4234077367859415
Right
0.9615758479910338
Left
0.5757840366788696
Left
0.9615758479910338
Left
0.8313556036673061
Left
0.4320092771823249
Right

```
0.5750196473917903
Left
0.9851058336025954
Left
0.4320092771823249
Right
0.9770739568593986
Left
0.5750196473917903
Left
0.9677301593531965
Left
0.5750196473917903
Left
0.9770739568593986
Left
```

In [282…]

```
len(AP_predictions_proba_AP)
```

Out[282…]

```
188
```

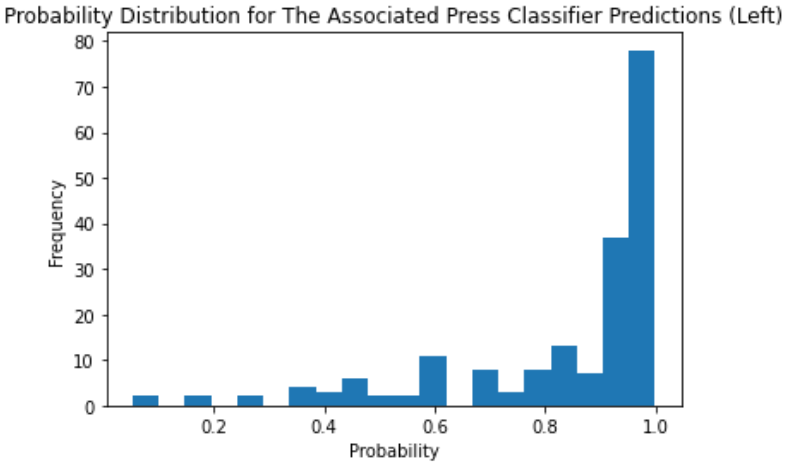
In [283…]

```
#from sklearn.svm import SVC
#svm = SVC(kernel='linear', probability=True)
#svm.fit(X_train_tf, Y_train)

#AP_predictions_proba = svm.predict_proba(X_Data_Centered)

for paAP in range(0,len(AP_predictions_proba_AP)):
    for pa2AP in range(0,1):
        if (AP_predictions_proba_AP[paAP][pa2AP] <= 0.65) & \
        (AP_predictions_proba_AP[paAP][pa2AP] >= 0.5):
            print(pa, ':', AP_predictions_proba_AP[paAP][pa2AP],
                  Data_Classification_Centered_AP[paAP])
```

```
180 : 0.5336534447481074 Right
180 : 0.6174728227984922 Right
180 : 0.5516811229058828 Left
180 : 0.5888748501334558 Left
180 : 0.5888748501334558 Left
180 : 0.6177850658464895 Left
180 : 0.5747207232364414 Right
180 : 0.5747207232364414 Right
180 : 0.5757840366788696 Left
180 : 0.5757840366788696 Left
180 : 0.5750196473917903 Left
180 : 0.5750196473917903 Left
180 : 0.5750196473917903 Left
```

In [286…]

```
#AP_predictions_proba

plt.hist(AP_predictions_proba_AP[:, 0], bins=20)
plt.xlabel('Probability')
plt.ylabel('Frequency')
plt.title('Probability Distribution for The Associated Press Classifier Predictions (Left)')
plt.show()
```



As could be expected from the AllSides experts, the Associated Press online publication is, in fact more confidently left leaning. Since all our left leaning data was first order left lean, this makes sense. The distribution for left leaning data is pretty confident when it is leaning and most articles are categorized as left leaning."

FOX vs MSNBC political audience disparities (presentation data)

In [ ]:
```
MSNBC_political_demographic = [['MSN',0.95, 0.05],['FOX',0.07, 0.93]]
FOX_political_demographic = []
News_political_demographic=pd.DataFrame(MSNBC_political_demographic)
```

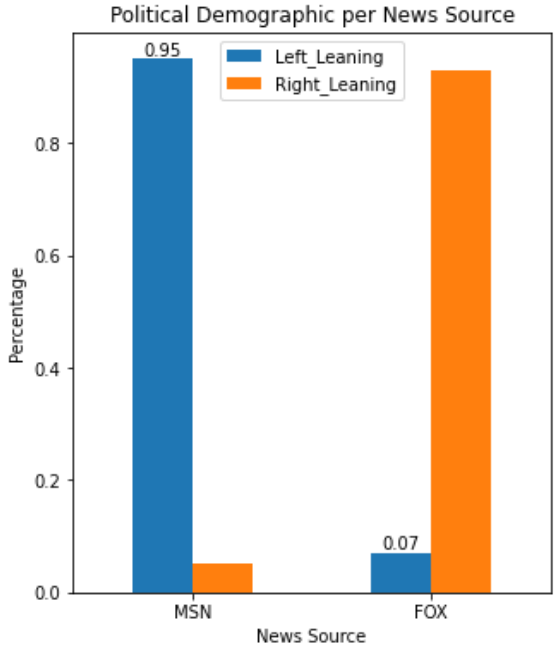
In [ ]:
```
News_political_demographic.head()
```

	0	1	2
0	MSN	0.95	0.05
1	FOX	0.07	0.93

In [ ]:
```
News_political_demographic.rename (columns = {0:'News_Source'}, inplace=True)
News_political_demographic.rename (columns = {1:'Left_Leaning'}, inplace=True)
News_political_demographic.rename (columns = {2:'Right_Leaning'}, inplace=True)
```

In [474…
```
ax2=News_political_demographic.plot(kind="bar", x='News_Source',rot=0,
                                    legend=True,
                                    figsize=(5,6),
                                    xlabel='News Source',
                                    ylabel='Percentage',
                                    title='Political Demographic per News Source')
ax2.bar_label(ax2.containers[0])
```

Out[474…
```
[Text(0, 0, '0.95'), Text(0, 0, '0.07')]
```



Concentration ratio Charts (presentation data)

In [475…

```
Concentration_Ratios=[['Right','nyc',17.49],['Right','reparations',16.89],['Right','aliens',12.7],
                      ['Right','ccp',11.5],['Right','locker',11.07],['Right','comey',9.58],
                      ['Right','lula',9.2],['Right','busch',8.9],['Right','fidelity',8.77],
                      ['Left','anonymity',17.39],['Left','thomnass',16.14],['Left','willis',14.19],
                      ['Left','mehta',13.6],['Left','docket',13.08],['Left','ginni',12.24],
                      ['Left','pork',11.59],
                      ['Left','uncertainty',9.54],['Left','hush',9.74],['Left','gorsuch',9.74]]

Concentration_Ratios_df=pd.DataFrame(Concentration_Ratios)
```

In [476…

```
Concentration_Ratios_df.rename (columns = {0:'Political_Lean'}, inplace=True)
Concentration_Ratios_df.rename (columns = {1:'Word'}, inplace=True)
Concentration_Ratios_df.rename (columns = {2:'Factor_of_Concentration'}, inplace=True)
```

In [477…

```
Concentration_Ratios_df.head()
```

Out[477…

	Political_Lean	Word	Factor_of_Concentration
0	Right	nyc	17.49
1	Right	reparations	16.89
2	Right	aliens	12.70
3	Right	ccp	11.50
4	Right	locker	11.07

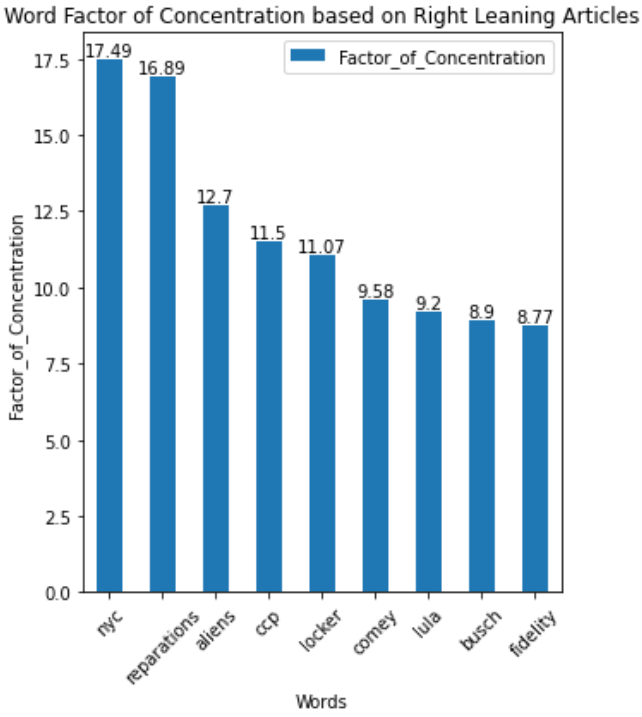
In [478…

```
ax3=Concentration_Ratios_df[Concentration_Ratios_df['Political_Lean']=='Right'].plot(kind="bar",
                                                                                     x='Word',rot=45,
                                    legend=True,
                                    figsize=(5,6),
                                    xlabel='Words',
                                    ylabel='Factor_of_Concentration',
                                    #='Political_Lean',
                                    title='Word Factor of Concentration based on Right Leaning Articles')
ax3.bar_label(ax3.containers[0])
```

Out[478…

```
[Text(0, 0, '17.49'),
 Text(0, 0, '16.89'),
 Text(0, 0, '12.7'),
 Text(0, 0, '11.5'),
 Text(0, 0, '11.07'),
 Text(0, 0, '9.58'),
```

```
 Text(0, 0, '9.2'),
 Text(0, 0, '8.9'),
 Text(0, 0, '8.77')]
```

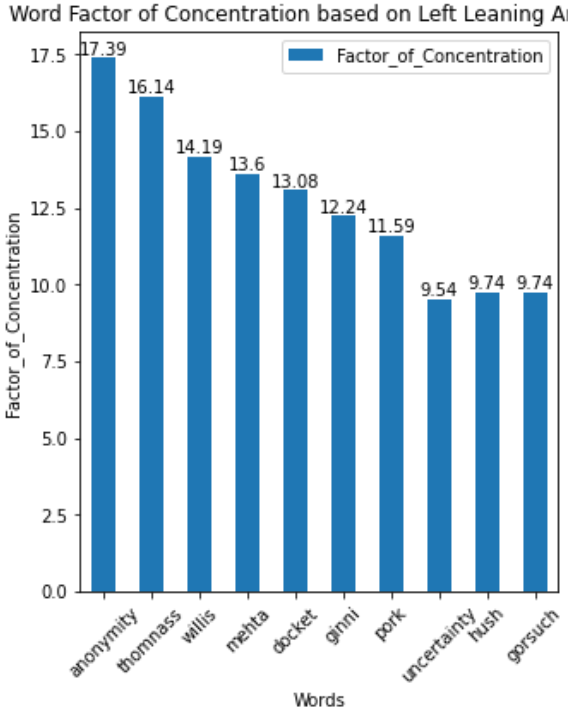


In [479…

```
ax3=Concentration_Ratios_df[Concentration_Ratios_df['Political_Lean']=='Left'].plot(kind="bar",
                                                                                    x='Word',rot=45,
                                    legend=True,
                                    figsize=(5,6),
                                    xlabel='Words',
                                    ylabel='Factor_of_Concentration',
                                    #='Political_Lean',
                                    title='Word Factor of Concentration based on Left Leaning Articles')
ax3.bar_label(ax3.containers[0])
```

Out[479…

```
[Text(0, 0, '17.39'),
 Text(0, 0, '16.14'),
 Text(0, 0, '14.19'),
 Text(0, 0, '13.6'),
 Text(0, 0, '13.08'),
 Text(0, 0, '12.24'),
 Text(0, 0, '11.59'),
 Text(0, 0, '9.54'),
 Text(0, 0, '9.74'),
 Text(0, 0, '9.74')]
```


Word Factor of Concentration based on Left Leaning Articles
Factor_of_Concentration
17.39
16.14
14.19
13.6
13.08
12.24
11.59
9.54
9.74
9.74
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Factor_of_Concentration
anonymity
thomnass
willis
mehta
docket
ginni
pork
uncertainty
hush
gorsuch
Words

In [ ]:

509 Final Project

The notebook is for Exploratory Data Analysis (EDA), text data preprocessing, modeling, and evaluation.

Globally import libraries

In [1]:
```
from bs4 import BeautifulSoup
from collections import defaultdict, Counter
import datetime as dt
import emoji
import itertools
import json
import logging
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd
import pickle
import pymysql as mysql
import random
import re
import regex as rex
import requests
import shutil
from string import punctuation
import time
from tqdm import tqdm
import zipfile

import nltk
from nltk.corpus import stopwords
import spacy

from skopt import BayesSearchCV
from skopt.space import Real, Categorical, Integer

from sklearn.feature_extraction.text import TfidfTransformer, \
CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.pipeline import make_pipeline, Pipeline
from sklearn import metrics
from sklearn.metrics import make_scorer, f1_score, classification_report, \
confusion_matrix, ConfusionMatrixDisplay, RocCurveDisplay

import textacy.preprocessing as tprep
from textacy.extract import keyword_in_context

# Set pandas global options
```

```
pd.options.display.max_rows = 17
pd.options.display.precision = 4
np.set_printoptions(suppress=True, precision=4)

%matplotlib inline
```

Upload data from CSV

In [2]:
```
'''Dir nav citation:
https://softhints.com/python-change-directory-parent/'''
curr_dir = os.path.abspath(os.curdir)
print(curr_dir)
os.chdir("..")
up1_dir = os.path.abspath(os.curdir)
print(up1_dir)
```

```
C:\Users\acarr\Documents\GitHub\ADS509_Final_project\deliverables
C:\Users\acarr\Documents\GitHub\ADS509_Final_project
```

In [3]:
```
# change `data_location` to the location of the folder on your machine.
data_location = 'data'

file_in_name01 = 'master.csv'
file_in_name02 = 'master_business_TheHill.csv'

file_in_path01 = os.path.join(up1_dir, data_location, file_in_name01)
file_in_path02 = os.path.join(up1_dir, data_location, file_in_name02)

print(f'CSV file 1 in path: {file_in_path01}')
print(f'CSV file 2 in path: {file_in_path02}')
```

```
CSV file 1 in path: C:\Users\acarr\Documents\GitHub\ADS509_Final_project\data\maste
r.csv
CSV file 2 in path: C:\Users\acarr\Documents\GitHub\ADS509_Final_project\data\master
_business_TheHill.csv
```

Review dataframe

In [4]:
```
slct_tbl_full_df01 = pd.read_csv(file_in_path01)
print(f'Dataframe shape: {slct_tbl_full_df01.shape}')
display(slct_tbl_full_df01.head())
```

```
Dataframe shape: (4509, 7)
```

	source_name	author	title	url	publis
0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/...	20 30T16:
1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0...	20 30T19:
2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/...	20 30T23:
3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0...	20 30T18:
4	The Washington Post	NaN	The revolt of Christian home-schoolers...	https://www.washingtonpost.com/education/inter...	20 30T18:

Exploratory Data Analysis (EDA)

Count missing `article_text` feature

The majority of null values appear in the `content` column. There are also several in `author` and one in `article_text`. Neither `content` nor `author` will be used for current modeling efforts, therefore they are not a factor. The one instance with missing article text will be removed.

In [5]:
```
count_nan = slct_tbl_full_df01.isnull().sum()

# printing the number of values present
# in the column
print('Number of NaN values present: ' + str(count_nan))
```

```
Number of NaN values present: source_name        0
author            37
title              0
url                0
publish_date       0
content         3351
article_text       1
dtype: int64
```

Count blank `article_text` feature

In [6]:
```
print(len(slct_tbl_full_df01[slct_tbl_full_df01['article_text']=='']))
display(slct_tbl_full_df01[slct_tbl_full_df01['article_text']=='].head(20))
```

0

source_name	author	title	url	publish_date	content	article_text

Remove missing `article_text` row(s)

In [7]:
```
'''Drop missing citation:
https://pandas.pydata.org/pandas-docs/stable/reference
/api/pandas.DataFrame.dropna.html#pandas.DataFrame.dropna'''
slct_tbl_full_df02 = slct_tbl_full_df01.dropna(subset=['article_text'])
print(f'Dataframe shape: {slct_tbl_full_df02.shape}')
display(slct_tbl_full_df02.head())
```

```
Dataframe shape: (4508, 7)
```

	source_name	author	title	url	pub
0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/...	30T
1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0...	30T
2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/...	30T
3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0...	30T
5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0...	30T

Count characters and words for initial review

In [8]:
```
tqdm.pandas(ncols=50)  # can use tqdm_gui, optional kwargs, etc
# Now you can use `progress_apply` instead of `apply`

# Raw text character and word counts
slct_tbl_full_df02['char_cnt'] = slct_tbl_full_df02['article_text']\
.progress_apply(len)
slct_tbl_full_df02['word_cnt'] = slct_tbl_full_df02['article_text']\
.progress_apply(lambda x: len(x.split()))
display(slct_tbl_full_df02.head())
```

```
100%|██████| 4508/4508 [00:00<00:00, 322023.34it/s]
C:\Users\acarr\AppData\Local\Temp\ipykernel_23812\2936833956.py:5: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  slct_tbl_full_df02['char_cnt'] = slct_tbl_full_df02['article_text']\
100%|██████| 4508/4508 [00:00<00:00, 11534.89it/s]
C:\Users\acarr\AppData\Local\Temp\ipykernel_23812\2936833956.py:7: SettingWithCopyWa
rning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  slct_tbl_full_df02['word_cnt'] = slct_tbl_full_df02['article_text']\
```

	source_name	author	title	url	pub
0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/...	30T
1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0...	30T
2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/...	30T
3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0...	30T
5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0...	30T

Descriptive statistics

Stats are displayed for both categorical and numerical columns. As expected "Fox News" is the most frequent value in `source_name` as the most articles were collected from that news site. The inclusion of "Associated Press" as the mode for `author` identified it as a potential source for skew in the final results, as AP was rated as a "center" source in the AllSide Media Bias Chart. As a result, all articles with an `author` value of "Associated Press" were removed; similarily, articles by "msn" and "Reuters" were also removed.
For the numerical values, there was a very large range for both character and word counts (80,454 and 14,306, respectively), but also a large delta between the 75% percentile and max (74,920.5 and 13,433, respectively), indicating a distribution with a very long right tail with a very small amount of some extremely long (outlier) articles. As a result, the standard deviation was also quite large relative to the mean. For the current analyses, no additional efforts will be performed to account for outliers, but this will be an examination factor for future expansion/comparative studies.

In [9]:
```
slct_tbl_full_df02[['source_name',
                    'author',
                    'publish_date',
                    'article_text']].describe(include="O").T
```

Out[9]:

	count	unique	top	freq
source_name	4508	4	Fox News	2192
author	4472	956	Associated Press	450
publish_date	4508	4486	2023-05-13T11:00:00Z	3
article_text	4508	4508	Travelers in Alabama driving on Interstate 65 ...	1

In [10]:
```
slct_tbl_full_df02.describe().T
```

Out[10]:

	count	mean	std	min	25%	50%	75%	max
char_cnt	4508.0	4655.5011	3137.3650	131.0	2832.0	3951.5	5664.5	80585.0
word_cnt	4508.0	731.0315	518.5765	16.0	432.0	607.0	889.0	14322.0

Display Source counts

In [11]:
```
slct_tbl_full_df02['source_name'].value_counts()
```

Out[11]:
```
Fox News               2192
Breitbart News         1017
CNN                     773
The Washington Post     526
Name: source_name, dtype: int64
```

Examine inclusion of "centrist" sources indicated by `author` feature

In [12]:
```
slct_tbl_full_df02a = slct_tbl_full_df02[slct_tbl_full_df02['author']\
                                         .isin(['msn',
                                                'Associated Press',
                                                'Reuters'])]

display(slct_tbl_full_df02a[slct_tbl_full_df02a['author']=='msn'])

display(slct_tbl_full_df02a.groupby(by=['source_name', 'author']).count())
```

	source_name	author	title	url	pu
17	The Washington Post	msn	State Dept seeks to expand space diplomacy...	https://www.washingtonpost.com/technology/2023...	30
18	The Washington Post	msn	SHOCK IN RUSSIAN CAPITAL	https://www.washingtonpost.com/world/2023/05/3...	30
22	The Washington Post	msn	Debate over whether AI will destroy us is divi...	https://www.washingtonpost.com/technology/2023...	20
81	The Washington Post	msn	Corporate bankruptcies creeping up as pressure...	https://www.washingtonpost.com/business/2023/0...	23
84	The Washington Post	msn	The looming existential crisis for cable news...	https://www.washingtonpost.com/media/2023/05/2...	23
...	...	...	...	...	
492	The Washington Post	msn	Biden shows growing appetite to cross Putin's ...	https://www.washingtonpost.com/national-securi...	01
502	The Washington Post	msn	Behind-the-scenes videos of Tucker Carlson wer...	https://www.washingtonpost.com/media/2023/06/0...	02
503	The Washington Post	msn	Georgia probe of Trump broadens to activities ...	https://www.washingtonpost.com/nation/2023/06/...	02
506	The Washington Post	msn	DRAMA: Couple, both nurses, save man's life mi...	https://www.washingtonpost.com/lifestyle/2023/...	02

	source_name	author	title	url	pu
509	The Washington Post	msn	'DRAG RACE' queen says cancellation of militar...	https://www.washingtonpost.com/nation/2023/06/...	02

25 rows × 9 columns

		title	url	publish_date	content	article_text	char_cnt	word_cnt
source_name	**author**							
CNN	**Reuters**	6	6	6	1	6	6	6
Fox News	**Associated Press**	450	450	450	73	450	450	450
	Reuters	1	1	1	0	1	1	1
The Washington Post	**msn**	25	25	25	25	25	25	25

In [13]:
```
counter = Counter(slct_tbl_full_df02['author'])

word_cutoff = 5
con_feature_words = set()

for word, count in counter.items():
    if count > word_cutoff:
        con_feature_words.add(word)

print(f'''With a word cutoff of {word_cutoff}, we have
{len(con_feature_words)} words as features in the model.''')
print(con_feature_words)
```

```
With a word cutoff of 5, we have
151 words as features in the model.
{nan, 'Greg Gutfeld', 'Julia Musto', 'Ryan Morik', 'Devlin Barrett', 'Associated Pre
ss', 'Hanna Panreck', 'Howard Kurtz', 'Lawrence Richard', 'Amy B Wang', 'Gabriel Hay
s', 'Breitbart London, Breitbart London', 'Bradford Betz', 'Timothy Nerozzi', 'Steph
en Collinson', 'Pam Key, Pam Key', 'Peter Aitken', 'Elaine Mallon, Elaine Mallon',
'Neil Munro, Neil Munro', 'Robert Barnes', 'Chad Pergram', 'Chris Pandolfo', 'Deirdr
e Reilly', 'Katherine Hamilton, Katherine Hamilton', 'Brian Fung', 'Matt Egan', 'Joe
l B. Pollak, Joel B. Pollak', 'Elizabeth Elkind', 'Greg Norman', 'Peter Caddle, Pete
r Caddle', 'Frances Martel, Frances Martel', 'Haley Chi-Sing', 'Madeline Coggins',
'Kurt Zindulka, Kurt Zindulka', 'Rebecca Rosenberg', 'Houston Keene', 'Paulina Deda
j', 'Christian K. Caruzo, Christian K. Caruzo', 'Hannah Ray Lambert', 'AWR Hawkins,
AWR Hawkins', 'Andrew Miller', 'Nadeen Ebrahim', 'Danielle Wallace', 'Eric Bradner',
'Alana Mastrangelo, Alana Mastrangelo', 'Paul Steinhauser', 'Jordan Dixon-Hamilton,
Jordan Dixon-Hamilton', 'Paul Kane', 'Sean Moran, Sean Moran', 'Brianna Herlihy', 'G
lenn Kessler', 'Adam Shaw', 'Wendell Husebø, Wendell Husebø', 'Elizabeth Heckman',
'Hannah Rabinowitz', 'Joshua Nelson', 'Jessica Chasmar', 'John Nolte, John Nolte',
'Lindsay Kornick', 'Ryan Gaydos', 'Tierney Sneed', 'John Hayward, John Hayward', 'Az
i Paybarah', 'Zachary B. Wolf', 'Hannah Bleau, Hannah Bleau', 'Ashley Oliver, Ashley
Oliver', 'Joe Schoffstall', 'Nicole Goodkind', 'Emma Colton', 'Lucas Nolan, Lucas No
lan', 'Jeffrey Clark', 'Kevin Liptak', 'Melissa Rudy', 'Oliver Darcy', 'Chris Eberha
rt', 'Alexandra Meeks', 'Audrey Conklin', 'Fox News', 'Allum Bokhari, Allum Bokhar
i', 'Philip Bump', 'Steve Contorno', 'Kassy Dillon', 'Taylor Penley', 'Spencer S. Hs
u', 'Jon Brown', 'Kendall Tietz', 'Bailee Hill', 'Kurt Knutsson, CyberGuy Report',
'Adam Sabes', 'Paul Waldman', 'Ariane de Vogue', 'Michael Lee', 'Bob Price, Bob Pric
e', 'Reuters', 'Alisha Ebrahimji', 'Maeve Reston', 'Ashley Carnahan', 'Charles Creit
z', 'Peter Kasperowicz', 'Andrea Vacchiano', 'Angelica Stabile', 'Michael Ruiz', 'ms
n', 'Tony Romm', 'Louis Casiano', 'Anders Hagstrom', 'Kristine Parks', 'Joshua Klei
n, Joshua Klein', 'Warner Todd Huston, Warner Todd Huston', 'Sarah Rumpf-Whitten',
'David Ng, David Ng', 'Jacob Bliss, Jacob Bliss', 'Kyle Morris', 'Oliver JJ Lane, Ol
iver JJ Lane', 'Caitlin McFall', 'Yael Halon', 'John Binder, John Binder', 'Jeff Ste
in', 'Tami Luhby', 'Sean Lyngaas', 'Chantz Martin', 'Aaron Blake', 'John Wagner', 'P
atrick Hauf', 'Fox News Staff', 'Kerry Byrne', 'Brian Flood', 'Brie Stimson', 'Matth
ew Boyle, Matthew Boyle', 'Brandon Gillespie', 'Amy Furr, Amy Furr', 'Stephen Sorac
e', 'Aubrie Spady', 'Jeff Poor, Jeff Poor', 'Simon Kent, Simon Kent', 'Jennifer Rubi
n', 'Brooke Singman', 'Kristina Wong, Kristina Wong', 'Thomas Catenacci', 'Mariana A
lfaro', 'Joseph Wulfsohn', 'Ian Hanchett, Ian Hanchett', 'Aaron Kliegman', 'Greg Weh
ner', 'Nick Gilbertson, Nick Gilbertson', 'Hannah Grossman', 'Landon Mion', 'Dylan G
winn, Dylan Gwinn', 'Alexander Hall', 'Hannah Knowles', 'Paul Bois, Paul Bois'}
```

Assign class based on `source_name` and AllSides Media Bias Chart

In [14]:

```
slct_tbl_full_df03 = slct_tbl_full_df02[~slct_tbl_full_df02['author']\
                                        .isin(['msn',
                                               'Associated Press',
                                               'Reuters'])]
slct_tbl_full_df03 = slct_tbl_full_df03.reset_index()
slct_tbl_full_df03['political_lean'] = 'right'
print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())
slct_tbl_full_df03.loc[(slct_tbl_full_df03['source_name'] \
                        == 'The Washington Post') \
                       | (slct_tbl_full_df03['source_name'] \
                          == 'CNN'), 'political_lean'] = 'left'
```

```
display(slct_tbl_full_df03.head())

display(slct_tbl_full_df03['political_lean'].value_counts())
```

```
(4026, 11)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

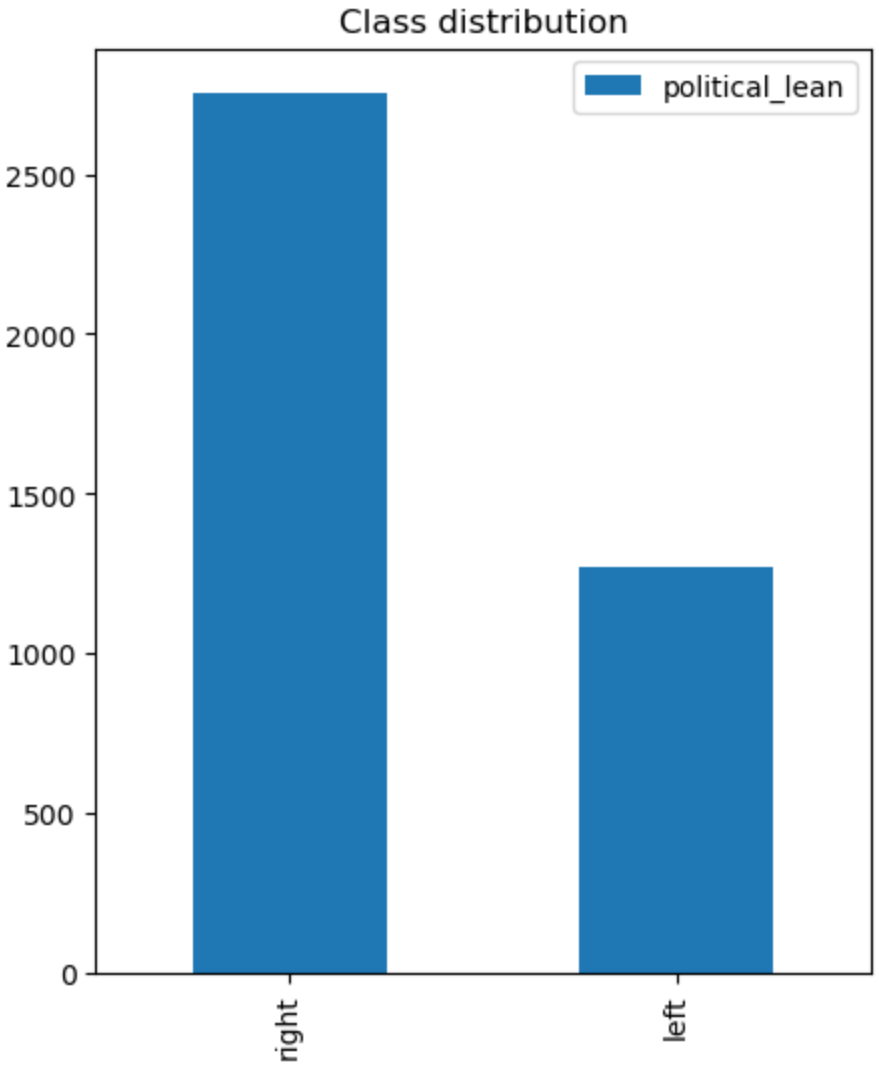
```
right    2758
left     1268
```

Visualize class distribution

There is definitely an imbalance in the number of instances in each class. This is due to Fox News being the most prolific source, whether because they put out a lot more articles or their sites were more consistently available for scraping. This imbalance is not considered extreme and will not be adjusted for within the scope of the current study.

In [15]:
```
slct_tbl_full_df03['political_lean'].value_counts().plot(kind="bar",
                                                         legend=True,
                                                         figsize=(5,6),
                                                        title='Class distribution')
```

Out[15]:
```
<Axes: title={'center': 'Class distribution'}>
```

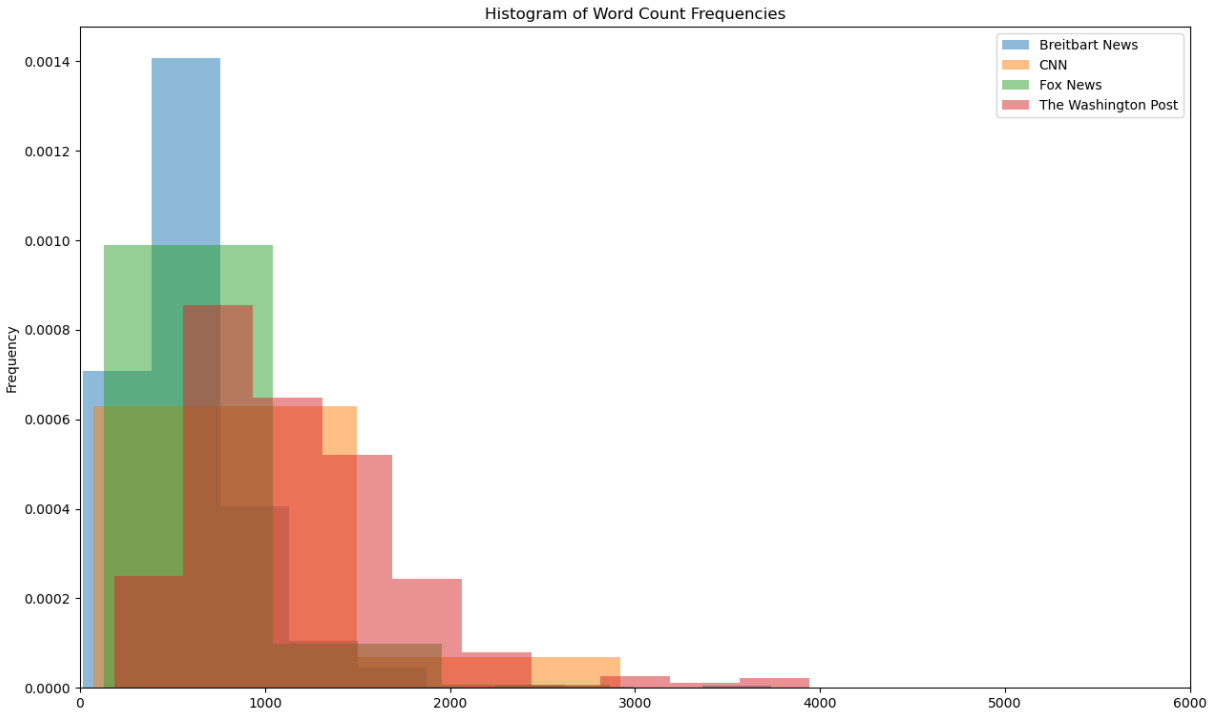


Plot word counts

All sources seem to have very similar consolidation of most frequent word counts between 0 and 2,000. However, the two "left" sources (CNN and The Washington Post) seem to be the significant source of the outliers, with a small amount of articles each that have extremely large word counts (*note:* the x-axis range was truncated at 6,000 to make it more readabla--as noted above, there were some articles with word counts greater than 14,000). Given the simialirties between the sources within each class, the differences may correlate to intentional word limitation based on perceived audience desires, but in the very least do add evidence that the sources have been grouped together appropriately.

In [16]:
```
slct_tbl_full_df03.groupby('source_name')['word_cnt'].plot(kind="hist",
                                                           density=True,
                                                           alpha=0.5,
                                                           legend=True,
                                                           figsize=(15,9),
                                    title='Histogram of Word Count Frequencies',
                                                           xlim=(0,6000))
```

Out[16]:
```
source_name
Breitbart News         Axes(0.125,0.11;0.775x0.77)
CNN                    Axes(0.125,0.11;0.775x0.77)
Fox News               Axes(0.125,0.11;0.775x0.77)
The Washington Post    Axes(0.125,0.11;0.775x0.77)
Name: word_cnt, dtype: object
```



Data preprocessing

In [17]:
```
def uniq_tok(df_col=None):
    '''Display all unique tokens across all instances'''
    df_cols1 = pd.Series(df_col)

    all_tokens_lst01 = []

    [all_tokens_lst01.append(f) for f in df_cols1]
    all_tokens_lst01 = list(itertools.chain.from_iterable(all_tokens_lst01))
    all_tokens_set01 = set(all_tokens_lst01)
    print(len(sorted(all_tokens_set01)))
    print(sorted(all_tokens_set01))
```

In [18]:
```
slct_tbl_full_df04 = slct_tbl_full_df03.copy()
```

Case-loading

In [19]:
```
slct_tbl_full_df03['lower'] = slct_tbl_full_df03['article_text']\
.apply(str.lower)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())
```

```
(4026, 12)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

Text normalization

Create function

In [20]:
```
def normalize(text):
    text = tprep.normalize.hyphenated_words(text)
    text = tprep.normalize.quotation_marks(text)
    text = tprep.normalize.unicode(text)
    text = tprep.remove.accents(text)
    return text
```

Call function

In [21]:
```
slct_tbl_full_df03['norm'] = slct_tbl_full_df03['lower'].apply(normalize)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())
```

```
for c in range(0,1):
    try:
        print(slct_tbl_full_df03['norm'][c], '\n')
    except:
        print(f'Skip {c}')
```

```
(4026, 13)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

travelers in alabama driving on interstate 65 to parties and barbecues on memorial d
ay might have seen messages on digital road signs honoring veterans who died fightin
g for the united states. but that's not what some drivers near clanton, ala., saw on
monday. instead, motorists reported seeing a sign that was apparently hacked to disp
lay the words "reclaim america," a white nationalist slogan, and "patriot front us,"
referencing the white supremacist group that was involved in the deadly 2017 unite t
he right rally in charlottesville. "how does this come about?" wrote sarah hughes, a
motorist who captured photos of the sign and posted them on twitter. "weird as hel
l." a contractor's portable message board was hacked on i-65 in chilton county, al
a., on monday afternoon, john mcwilliams, a spokesman for the alabama department of
transportation (aldot) west central region, told the washington post in a statement.
"a citizen alerted a nearby state trooper about the message, who then contacted aldo
t," mcwilliams said tuesday. "aldot personnel immediately responded and turned the m
essage board off. no other message boards on i-65 were affected." mcwilliams added
that aldot is investigating how the white supremacist language appeared on the sign
near clanton, about 40 miles northwest of montgomery, ala. officials have given no i
mmediate indication of who is responsible for apparently hacking the interstate sig
n. the news was first reported by al.com. hughes told the post that she was driving
home to birmingham from a weekend at alabama's gulf coast when she saw the white sup
remacist messages that have recently popped up around her home city from supporters
of patriot front. "when i saw it, i thought, 'oh, it's the same guys,' " said hughe
s, a 31-year-old attorney. "i was kind of shocked." the hacked alabama road sign com
es at a time when president biden has declared white supremacy "the most dangerous t
errorist threat" to the country. during his commencement address at howard universit
y this month, biden told the graduating class at the historically black university t
hat he pledged "to stand up against the poison of white supremacy, as i did in my in
augural address — to single it out as the most dangerous terrorist threat to our hom
eland is white supremacy." "i don't have to tell you that progress toward justice of
ten meets ferocious pushback from the oldest and most sinister of forces," biden sai
d in the may 13 address, after quoting donald trump's equivocating response to the 2
017 rally in charlottesville that killed 32-year-old heather heyer and injured 19 ot
hers. "that's because hate never goes away." biden calls white supremacy greatest te
rrorism threat as 2024 race heats up the southern poverty law center (splc) tracked
at least 13 hate groups in alabama in 2021, including the proud boys. the discussion
surrounding white supremacists and white nationalists in alabama intensified this mo
nth after sen. tommy tuberville (r-ala.) said that people identified as "white extre
mists" and white nationalists should be allowed to serve in the u.s. armed forces. w
hen asked by a reporter with wbhm in birmingham whether white nationalists should be
allowed to serve in the military, tuberville replied, "well, they call them that. i
call them americans." after tuberville was criticized, a spokesman told the post tha
t the senator "resents the implication that the people in our military are anything
but patriots and heroes." gop senator says of white nationalists in the military, 'i
call them americans' patriot front, the white supremacist group whose name was displ
ayed on the interstate sign, is a texas-based hate group that broke off from vanguar
d america and formed after the charlottesville rally, the splc says. its members hav
e chanted "reclaim america" at rallies in coeur d'alene, idaho, washington and bosto
n in recent years, according to news reports. patriot front is responsible for "the
vast majority of white supremacist propaganda distributed in the united states" sinc
e 2019, according to the anti-defamation league. it's not the first time that langua
ge promoting patriot front has made its way into a public space in alabama. in july,
graffiti beneath a birmingham bridge appeared with "patriot front us" spray-painted
in red and blue letters, al.com reported. other patriot front graffiti has also been
spotted in birmingham, a city with a population that's nearly 70 percent black, acco
rding to u.s. census data. a photo posted to twitter this month showed more patriot
front graffiti along the red mountain expressway in birmingham with the words, "we d
are defend our rights." the patriot front graffiti was later removed, but the messag

e left sydney duncan, the attorney director for the magic legal center in birmingha
m, saddened that hate had become so public in some parts of alabama. "white supremac
y is alive and well," duncan wrote. hughes said she was traveling north to birmingh
am when she pulled over on i-65 to take photos of the messages on the sign. she had
seen confederate monuments and flags on that drive before, but that kind of messagin
g on government-owned property was different, she said. a police officer who was alr
eady at the scene waved at her to keep driving, hughes added. when she returned hom
e, hughes said she felt compelled to share the images due to the ongoing conversatio
n happening among birmingham residents about the promotion of patriot front in publi
c spaces. "some people might perceive this as upsetting and scary, and a sign of the
worsening of our country," she said. "but if this is their strategy, then i'm not re
ally impressed." she added, "they're a dying breed." toluse olorunnipa and azi payba
rah contributed to this report.

In [22]:
```
text2find_rex = rex.compile(r'(click here to get the fox news app)')
test_lst = []

def test(text):
    test_lst.append(text2find_rex.findall(text))

slct_tbl_full_df03['norm'].apply(test)

display(slct_tbl_full_df03.head())
#print(test_lst)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

Remove special characters

Create function

In [23]:
```
rex_sep = rex.compile(r' ')
rex_ucode = rex.compile(r'[\\]u20*')

'''re.sub lambda citation:
https://chat.openai.com/share/402ec66e-2802-4cda-af8c-6f9f5b097d85
'''
sep_lst = []
ucode_lst = []
# Add leading and trailing space to URLs
def rex_replace(text):
    #txt = str(text)
    #print(lambda x: x.replace(' ', ' '))
    #sep_lst.append(rex_sep.findall(txt))
    #ucode_lst.append(rex_ucode.findall(txt))
```

```
    text = text.replace(r' ', ' ').replace(r'-', ' ')\
    .replace(r'\n', ' ').replace('\u2063', ' ').replace('\u2066', ' ')\
    .replace('\u2069', ' ').replace('\u200b', ' ').replace('\u200d', ' ')\
    .replace('(click to view)', ' ')\
    .replace('a post shared by', ' ')\
    .replace('app users click here', ' ')\
    .replace('app users: click here', ' ')\
    .replace('app users, click here:', ' ')\
    .replace('click here.', ' ')\
    .replace('click here for more cartoons', ' ')\
    .replace('click here for more', ' ')\
    .replace('click here for more sports coverage on foxnews.com', ' ')\
    .replace('click here for other fox news digital adoptable pets stories', ' ')\
    .replace('click here for the fox news app', ' ')\
    .replace('click here for the latest fox news reporting', ' ')\
    .replace('click here for topline and cross tabs conducted', ' ')\
    .replace('click here to hear more', ' ')\
    .replace('click here to ge the fox news app', ' ')\
    .replace('click here to get the fox news app', ' ')\
    .replace('click here to get the opinion newsletter', ' ')\
    .replace('click here to learn more', ' ')\
    .replace('click here to read more', ' ')\
    .replace('click here to sign up for our health newsletter', ' ')\
    .replace('click here to sign up for our lifestyle newsletter', ' ')\
    .replace('click here to sign up for our opinion newsletter', ' ')\
    .replace('click here to sign up for the entertainment newsletter', ' ')\
    .replace('click here to subscribe and get your first year of fox nation free of
    .replace('click here to view', ' ')\
    .replace("click to get kurt's cyberguy newsletter with quick tips, tech reviews
    .replace("click to get kurt's cyberguy newsletter with security alerts, quick t
    .replace("click to get kurt's free cyberguy newsletter with quick tips, tech re
    .replace("click to get kurt's free cyberguy newsletter with security alerts, qu
    .replace('click to get the fox news app', ' ')\
    .replace('fox news digital', ' ')\
    .replace('request for comment', ' ')\
    .replace('the ap ', ' ')\
    .replace('copyright © 2023 breitbart', ' ')\
    .replace('all rights reserved', ' ')\
    .replace('copyright 2023 cyberguy.com', ' ')\
    .replace('copyright 2023 fox news network', ' ')\
    .replace('copyright 2023 viq media transcription', ' ')\
    .replace("please let us know if you're having issues with commenting", ' ')\
    .replace('view this post on instagram', ' ')
    #txt = txt
    #text = text.replace(r'200b', 'd171c')
    #text = rex_ucode.sub('', text)
    return text

    #.replace('philip bump', ' ')\
    #.replace('paul kane', ' ')\
    #.replace('&amp', ' ')\
```

Call function

In [24]:
```
slct_tbl_full_df03['replace'] = slct_tbl_full_df03['norm'].apply(rex_replace)

#print(ucode_lst)
#print(sep_lst)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())
```

```
(4026, 14)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

'''Complex citation (add lambda): https://chat.openai.com/share/a135754c-c38c-47ea-8f83-54d41d5397ab ''' slct_tbl_full_df03['replace'] = slct_tbl_full_df03['norm'].apply(lambda x: x.replace(' ', ' ').replace(r'\n', ' ').replace('\u2063', ' ').replace('\u2066', ' ').replace('\u2069', ' ').replace('\u200b', ' ').replace('\u200d', ' '))

URL RegEx find

Create function

In [25]:
```
rex_url_c = rex.compile(r'http[s]?:[\/]+[\S]*\s')

'''re.sub lambda citation:
https://chat.openai.com/share/402ec66e-2802-4cda-af8c-6f9f5b097d85
'''
# Add leading and trailing space to URLs
def rex_url(text):
    text = rex_url_c.sub(lambda match: ' ' + match.group(0) + ' ', text)
    return text
```

Call function

In [26]:
```
slct_tbl_full_df03['rex_urls'] = slct_tbl_full_df03['replace'].apply(rex_url)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())
```

```
(4026, 15)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

Separate emojis as individual tokens

Create function

In [27]:
```
def emoji_split(text):
    return("".join([' ' + c + ' ' if emoji.is_emoji(c) else c for c in text]))
```

Call function

In [28]:
```
slct_tbl_full_df03['emoji_split'] = slct_tbl_full_df03['rex_urls']\
.apply(emoji_split)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())

for c in range(0,1):
    try:
        print(slct_tbl_full_df03['emoji_split'][c], '\n')
    except:
        print(f'Skip {c}')
```

```
(4026, 16)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

travelers in alabama driving on interstate 65 to parties and barbecues on memorial d
ay might have seen messages on digital road signs honoring veterans who died fightin
g for the united states. but that's not what some drivers near clanton, ala., saw on
monday. instead, motorists reported seeing a sign that was apparently hacked to disp
lay the words "reclaim america," a white nationalist slogan, and "patriot front us,"
referencing the white supremacist group that was involved in the deadly 2017 unite t
he right rally in charlottesville. "how does this come about?" wrote sarah hughes, a
motorist who captured photos of the sign and posted them on twitter. "weird as hel
l." a contractor's portable message board was hacked on i 65 in chilton county, al
a., on monday afternoon, john mcwilliams, a spokesman for the alabama department of
transportation (aldot) west central region, told the washington post in a statement.
"a citizen alerted a nearby state trooper about the message, who then contacted aldo
t," mcwilliams said tuesday. "aldot personnel immediately responded and turned the m
essage board off. no other message boards on i 65 were affected." mcwilliams added
that aldot is investigating how the white supremacist language appeared on the sign
near clanton, about 40 miles northwest of montgomery, ala. officials have given no i
mmediate indication of who is responsible for apparently hacking the interstate sig
n. the news was first reported by al.com. hughes told the post that she was driving
home to birmingham from a weekend at alabama's gulf coast when she saw the white sup
remacist messages that have recently popped up around her home city from supporters
of patriot front. "when i saw it, i thought, 'oh, it's the same guys,' " said hughe
s, a 31 year old attorney. "i was kind of shocked." the hacked alabama road sign com
es at a time when president biden has declared white supremacy "the most dangerous t
errorist threat" to the country. during his commencement address at howard universit
y this month, biden told the graduating class at the historically black university t
hat he pledged "to stand up against the poison of white supremacy, as i did in my in
augural address — to single it out as the most dangerous terrorist threat to our hom
eland is white supremacy." "i don't have to tell you that progress toward justice of
ten meets ferocious pushback from the oldest and most sinister of forces," biden sai
d in the may 13 address, after quoting donald trump's equivocating response to the 2
017 rally in charlottesville that killed 32 year old heather heyer and injured 19 ot
hers. "that's because hate never goes away." biden calls white supremacy greatest te
rrorism threat as 2024 race heats up the southern poverty law center (splc) tracked
at least 13 hate groups in alabama in 2021, including the proud boys. the discussion
surrounding white supremacists and white nationalists in alabama intensified this mo
nth after sen. tommy tuberville (r ala.) said that people identified as "white extre
mists" and white nationalists should be allowed to serve in the u.s. armed forces. w
hen asked by a reporter with wbhm in birmingham whether white nationalists should be
allowed to serve in the military, tuberville replied, "well, they call them that. i
call them americans." after tuberville was criticized, a spokesman told the post tha
t the senator "resents the implication that the people in our military are anything
but patriots and heroes." gop senator says of white nationalists in the military, 'i
call them americans' patriot front, the white supremacist group whose name was displ
ayed on the interstate sign, is a texas based hate group that broke off from vanguar
d america and formed after the charlottesville rally, the splc says. its members hav
e chanted "reclaim america" at rallies in coeur d'alene, idaho, washington and bosto
n in recent years, according to news reports. patriot front is responsible for "the
vast majority of white supremacist propaganda distributed in the united states" sinc
e 2019, according to the anti defamation league. it's not the first time that langua
ge promoting patriot front has made its way into a public space in alabama. in july,
graffiti beneath a birmingham bridge appeared with "patriot front us" spray painted
in red and blue letters, al.com reported. other patriot front graffiti has also been
spotted in birmingham, a city with a population that's nearly 70 percent black, acco
rding to u.s. census data. a photo posted to twitter this month showed more patriot
front graffiti along the red mountain expressway in birmingham with the words, "we d
are defend our rights." the patriot front graffiti was later removed, but the messag

```
e left sydney duncan, the attorney director for the magic legal center in birmingha
m, saddened that hate had become so public in some parts of alabama. "white supremac
y is alive and well," duncan wrote.  hughes said she was traveling north to birmingh
am when she pulled over on i 65 to take photos of the messages on the sign. she had
seen confederate monuments and flags on that drive before, but that kind of messagin
g on government owned property was different, she said. a police officer who was alr
eady at the scene waved at her to keep driving, hughes added. when she returned hom
e, hughes said she felt compelled to share the images due to the ongoing conversatio
n happening among birmingham residents about the promotion of patriot front in publi
c spaces. "some people might perceive this as upsetting and scary, and a sign of the
worsening of our country," she said. "but if this is their strategy, then i'm not re
ally impressed." she added, "they're a dying breed." toluse olorunnipa and azi payba
rah contributed to this report.
```

Lemmatization using spaCY

In [29]:
```
nlp_trans = spacy.load('en_core_web_sm')

def lemma(text):
    trans_txt = nlp_trans(text)
    tokens = [t.lemma_ for t in trans_txt]
    return tokens
```

slct_tbl_full_df03['lemma'] = slct_tbl_full_df03['replace'].progress_apply(lemma) print(slct_tbl_full_df03.shape) display(slct_tbl_full_df03.head()) for c in range(0,1): try: print(slct_tbl_full_df03['lemma'][c], '\n') except: print(f'Skip {c}')

Display globally unique tokens on lemmas

In [30]:
```
#uniq_tok(df_col=slct_tbl_full_df03['lemma'])
```

Split text

Apply

In [31]:
```
slct_tbl_full_df03['split'] = slct_tbl_full_df03['emoji_split']\
.apply(str.split)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())

for c in range(0,1):
    try:
        print(slct_tbl_full_df03['split'][c], '\n')
    except:
        print(f'Skip {c}')
```

```
(4026, 17)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

['travelers', 'in', 'alabama', 'driving', 'on', 'interstate', '65', 'to', 'parties',
'and', 'barbecues', 'on', 'memorial', 'day', 'might', 'have', 'seen', 'messages', 'o
n', 'digital', 'road', 'signs', 'honoring', 'veterans', 'who', 'died', 'fighting',
'for', 'the', 'united', 'states.', 'but', "that's", 'not', 'what', 'some', 'driver
s', 'near', 'clanton,', 'ala.,', 'saw', 'on', 'monday.', 'instead,', 'motorists', 'r
eported', 'seeing', 'a', 'sign', 'that', 'was', 'apparently', 'hacked', 'to', 'displ
ay', 'the', 'words', '"reclaim', 'america,"', 'a', 'white', 'nationalist', 'sloga
n,', 'and', '"patriot', 'front', 'us,"', 'referencing', 'the', 'white', 'supremacis
t', 'group', 'that', 'was', 'involved', 'in', 'the', 'deadly', '2017', 'unite', 'th
e', 'right', 'rally', 'in', 'charlottesville.', '"how', 'does', 'this', 'come', 'abo
ut?"', 'wrote', 'sarah', 'hughes,', 'a', 'motorist', 'who', 'captured', 'photos', 'o
f', 'the', 'sign', 'and', 'posted', 'them', 'on', 'twitter.', '"weird', 'as', 'hel
l."', 'a', "contractor's", 'portable', 'message', 'board', 'was', 'hacked', 'on',
'i', '65', 'in', 'chilton', 'county,', 'ala.,', 'on', 'monday', 'afternoon,', 'joh
n', 'mcwilliams,', 'a', 'spokesman', 'for', 'the', 'alabama', 'department', 'of', 't
ransportation', '(aldot)', 'west', 'central', 'region,', 'told', 'the', 'washingto
n', 'post', 'in', 'a', 'statement.', '"a', 'citizen', 'alerted', 'a', 'nearby', 'sta
te', 'trooper', 'about', 'the', 'message,', 'who', 'then', 'contacted', 'aldot,"',
'mcwilliams', 'said', 'tuesday.', '"aldot', 'personnel', 'immediately', 'responded',
'and', 'turned', 'the', 'message', 'board', 'off.', 'no', 'other', 'message', 'board
s', 'on', 'i', '65', 'were', 'affected."', 'mcwilliams', 'added', 'that', 'aldot',
'is', 'investigating', 'how', 'the', 'white', 'supremacist', 'language', 'appeared',
'on', 'the', 'sign', 'near', 'clanton,', 'about', '40', 'miles', 'northwest', 'of',
'montgomery,', 'ala.', 'officials', 'have', 'given', 'no', 'immediate', 'indicatio
n', 'of', 'who', 'is', 'responsible', 'for', 'apparently', 'hacking', 'the', 'inters
tate', 'sign.', 'the', 'news', 'was', 'first', 'reported', 'by', 'al.com.', 'hughe
s', 'told', 'the', 'post', 'that', 'she', 'was', 'driving', 'home', 'to', 'birmingha
m', 'from', 'a', 'weekend', 'at', "alabama's", 'gulf', 'coast', 'when', 'she', 'sa
w', 'the', 'white', 'supremacist', 'messages', 'that', 'have', 'recently', 'popped',
'up', 'around', 'her', 'home', 'city', 'from', 'supporters', 'of', 'patriot', 'fron
t.', '"when', 'i', 'saw', 'it,', 'i', 'thought,', "'oh,", "it's", 'the', 'same', "gu
ys,'", '"', 'said', 'hughes,', 'a', '31', 'year', 'old', 'attorney.', '"i', 'was',
'kind', 'of', 'shocked."', 'the', 'hacked', 'alabama', 'road', 'sign', 'comes', 'a
t', 'a', 'time', 'when', 'president', 'biden', 'has', 'declared', 'white', 'supremac
y', '"the', 'most', 'dangerous', 'terrorist', 'threat"', 'to', 'the', 'country.', 'd
uring', 'his', 'commencement', 'address', 'at', 'howard', 'university', 'this', 'mon
th,', 'biden', 'told', 'the', 'graduating', 'class', 'at', 'the', 'historically', 'b
lack', 'university', 'that', 'he', 'pledged', '"to', 'stand', 'up', 'against', 'th
e', 'poison', 'of', 'white', 'supremacy,', 'as', 'i', 'did', 'in', 'my', 'inaugura
l', 'address', '—', 'to', 'single', 'it', 'out', 'as', 'the', 'most', 'dangerous',
'terrorist', 'threat', 'to', 'our', 'homeland', 'is', 'white', 'supremacy."', '"i',
"don't", 'have', 'to', 'tell', 'you', 'that', 'progress', 'toward', 'justice', 'ofte
n', 'meets', 'ferocious', 'pushback', 'from', 'the', 'oldest', 'and', 'most', 'sinis
ter', 'of', 'forces,"', 'biden', 'said', 'in', 'the', 'may', '13', 'address,', 'afte
r', 'quoting', 'donald', "trump's", 'equivocating', 'response', 'to', 'the', '2017',
'rally', 'in', 'charlottesville', 'that', 'killed', '32', 'year', 'old', 'heather',
'heyer', 'and', 'injured', '19', 'others.', '"that\'s', 'because', 'hate', 'never',
'goes', 'away."', 'biden', 'calls', 'white', 'supremacy', 'greatest', 'terrorism',
'threat', 'as', '2024', 'race', 'heats', 'up', 'the', 'southern', 'poverty', 'law',
'center', '(splc)', 'tracked', 'at', 'least', '13', 'hate', 'groups', 'in', 'alabam
a', 'in', '2021,', 'including', 'the', 'proud', 'boys.', 'the', 'discussion', 'surro
unding', 'white', 'supremacists', 'and', 'white', 'nationalists', 'in', 'alabama',
'intensified', 'this', 'month', 'after', 'sen.', 'tommy', 'tuberville', '(r', 'al
a.)', 'said', 'that', 'people', 'identified', 'as', '"white', 'extremists"', 'and',
'white', 'nationalists', 'should', 'be', 'allowed', 'to', 'serve', 'in', 'the', 'u.
s.', 'armed', 'forces.', 'when', 'asked', 'by', 'a', 'reporter', 'with', 'wbhm', 'i

```
n', 'birmingham', 'whether', 'white', 'nationalists', 'should', 'be', 'allowed', 't
o', 'serve', 'in', 'the', 'military,', 'tuberville', 'replied,', '"well,', 'they',
'call', 'them', 'that.', 'i', 'call', 'them', 'americans."', 'after', 'tuberville',
'was', 'criticized,', 'a', 'spokesman', 'told', 'the', 'post', 'that', 'the', 'senat
or', '"resents', 'the', 'implication', 'that', 'the', 'people', 'in', 'our', 'milita
ry', 'are', 'anything', 'but', 'patriots', 'and', 'heroes."', 'gop', 'senator', 'say
s', 'of', 'white', 'nationalists', 'in', 'the', 'military,', "'i", 'call', 'them',
"americans'", 'patriot', 'front,', 'the', 'white', 'supremacist', 'group', 'whose',
'name', 'was', 'displayed', 'on', 'the', 'interstate', 'sign,', 'is', 'a', 'texas',
'based', 'hate', 'group', 'that', 'broke', 'off', 'from', 'vanguard', 'america', 'an
d', 'formed', 'after', 'the', 'charlottesville', 'rally,', 'the', 'splc', 'says.',
'its', 'members', 'have', 'chanted', '"reclaim', 'america"', 'at', 'rallies', 'in',
'coeur', "d'alene,", 'idaho,', 'washington', 'and', 'boston', 'in', 'recent', 'year
s,', 'according', 'to', 'news', 'reports.', 'patriot', 'front', 'is', 'responsible',
'for', '"the', 'vast', 'majority', 'of', 'white', 'supremacist', 'propaganda', 'dist
ributed', 'in', 'the', 'united', 'states"', 'since', '2019,', 'according', 'to', 'th
e', 'anti', 'defamation', 'league.', "it's", 'not', 'the', 'first', 'time', 'that',
'language', 'promoting', 'patriot', 'front', 'has', 'made', 'its', 'way', 'into',
'a', 'public', 'space', 'in', 'alabama.', 'in', 'july,', 'graffiti', 'beneath', 'a',
'birmingham', 'bridge', 'appeared', 'with', '"patriot', 'front', 'us"', 'spray', 'pa
inted', 'in', 'red', 'and', 'blue', 'letters,', 'al.com', 'reported.', 'other', 'pat
riot', 'front', 'graffiti', 'has', 'also', 'been', 'spotted', 'in', 'birmingham,',
'a', 'city', 'with', 'a', 'population', "that's", 'nearly', '70', 'percent', 'blac
k,', 'according', 'to', 'u.s.', 'census', 'data.', 'a', 'photo', 'posted', 'to', 'tw
itter', 'this', 'month', 'showed', 'more', 'patriot', 'front', 'graffiti', 'along',
'the', 'red', 'mountain', 'expressway', 'in', 'birmingham', 'with', 'the', 'words,',
'"we', 'dare', 'defend', 'our', 'rights."', 'the', 'patriot', 'front', 'graffiti',
'was', 'later', 'removed,', 'but', 'the', 'message', 'left', 'sydney', 'duncan,', 't
he', 'attorney', 'director', 'for', 'the', 'magic', 'legal', 'center', 'in', 'birmin
gham,', 'saddened', 'that', 'hate', 'had', 'become', 'so', 'public', 'in', 'some',
'parts', 'of', 'alabama.', '"white', 'supremacy', 'is', 'alive', 'and', 'well,"', 'd
uncan', 'wrote.', 'hughes', 'said', 'she', 'was', 'traveling', 'north', 'to', 'birmi
ngham', 'when', 'she', 'pulled', 'over', 'on', 'i', '65', 'to', 'take', 'photos', 'o
f', 'the', 'messages', 'on', 'the', 'sign.', 'she', 'had', 'seen', 'confederate', 'm
onuments', 'and', 'flags', 'on', 'that', 'drive', 'before,', 'but', 'that', 'kind',
'of', 'messaging', 'on', 'government', 'owned', 'property', 'was', 'different,', 'sh
e', 'said.', 'a', 'police', 'officer', 'who', 'was', 'already', 'at', 'the', 'scen
e', 'waved', 'at', 'her', 'to', 'keep', 'driving,', 'hughes', 'added.', 'when', 'sh
e', 'returned', 'home,', 'hughes', 'said', 'she', 'felt', 'compelled', 'to', 'shar
e', 'the', 'images', 'due', 'to', 'the', 'ongoing', 'conversation', 'happening', 'am
ong', 'birmingham', 'residents', 'about', 'the', 'promotion', 'of', 'patriot', 'fron
t', 'in', 'public', 'spaces.', '"some', 'people', 'might', 'perceive', 'this', 'as',
'upsetting', 'and', 'scary,', 'and', 'a', 'sign', 'of', 'the', 'worsening', 'of', 'o
ur', 'country,"', 'she', 'said.', '"but', 'if', 'this', 'is', 'their', 'strategy,',
'then', "i'm", 'not', 'really', 'impressed."', 'she', 'added,', '"they\'re', 'a', 'd
ying', 'breed."', 'toluse', 'olorunnipa', 'and', 'azi', 'paybarah', 'contributed',
'to', 'this', 'report.']
```

Display globally unique tokens on first split

In [32]:
```
#uniq_tok(df_col=slct_tbl_full_df03['split'])
```

Remove stop words

In [33]:
```
sw = stopwords.words("english")

# Add additional stop words
sw.extend(['',
           '',
           'arent',
           'cannot',
           'cant',
           'couldnt',
           'couldve',
           'didnt',
           'doesnt',
           'dont',
           'hadnt',
           'hasnt',
           'havent',
           'hes',
           'im',
           "i'm",
           'isnt',
           'it’s',
           'ive',
           'of',
           'mightnt',
           'mustnt',
           'neednt',
           'shant',
           'shes',
           'shouldnt',
           'shouldve',
           'thatll',
           'theyll',
           'theyve',
           'wasnt',
           'werent',
           'whats',
           'weve',
           'wont',
           'wouldnt',
           'wouldve',
           'yall',
           'youd',
           'youll',
           'youre',
           'youve',
           "we'll",
           "you’re",
           "you’ve",
           "you’ll",
           "you’d",
           "she’s",
           "it’s",
           "that’ll",
           "don’t",
           "should’ve",
```

```
            "aren't",
            "couldn't",
            "didn't",
            "doesn't",
            "hadn't",
            "hasn't",
            "haven't",
            "isn't",
            "mightn't",
            "mustn't",
            "needn't",
            "shan't",
            "shouldn't",
            "wasn't",
            "weren't",
            "won't",
            "wouldn't",
            "i'm",
            "we'll",
            'said',
            'told',
            'according',
            'fox',
            'news',
            'cnn',
            'breitbart',
            'reuters',
            'reporting',
            'reported',
            #'statement',
            #'spoke',
            #'next',
            #'though',
            #'often',
            #'story',
            #'updated',
            #'additional',
            #'developments',
            #'follow',
           ])

print(sw)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "yo
u've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'h
is', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itse
lf', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'who
m', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were',
'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing',
'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of',
'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'durin
g', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'o
n', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'wh
en', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'ot
her', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'to
o', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've",
'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "cou
ldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'h
aven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'n
eedn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'were
n', "weren't", 'won', "won't", 'wouldn', "wouldn't", '', '', 'arent', 'cannot', 'can
t', 'couldnt', 'couldve', 'didnt', 'doesnt', 'dont', 'hadnt', 'hasnt', 'havent', 'he
s', 'im', "i'm", 'isnt', 'it’s', 'ive', 'of', 'mightnt', 'mustnt', 'neednt', 'shan
t', 'shes', 'shouldnt', 'shouldve', 'thatll', 'theyll', 'theyve', 'wasnt', 'werent',
'whats', 'weve', 'wont', 'wouldnt', 'wouldve', 'yall', 'youd', 'youll', 'youre', 'yo
uve', "we'll", 'you’re', 'you’ve', 'you’ll', 'you’d', 'she’s', 'it’s', 'that’ll', 'd
on’t', 'should’ve', 'aren’t', 'couldn’t', 'didn’t', 'doesn’t', 'hadn’t', 'hasn’t',
'haven’t', 'isn’t', 'mightn’t', 'mustn’t', 'needn’t', 'shan’t', 'shouldn’t', 'was
n’t', 'weren’t', 'won’t', 'wouldn’t', 'i’m', 'we’ll', 'said', 'told', 'according',
'fox', 'news', 'cnn', 'breitbart', 'reuters', 'reporting', 'reported']
```

Create function

In [34]:
```
def sw_remover(tokens):
    return [t for t in tokens if t.lower() not in sw]
```

Call function

In [35]:
```
slct_tbl_full_df03['no_sw'] = slct_tbl_full_df03['split'].apply(sw_remover)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())

for c in range(0,1):
    print(slct_tbl_full_df03['no_sw'][c])
```

```
(4026, 18)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

['travelers', 'alabama', 'driving', 'interstate', '65', 'parties', 'barbecues', 'mem
orial', 'day', 'might', 'seen', 'messages', 'digital', 'road', 'signs', 'honoring',
'veterans', 'died', 'fighting', 'united', 'states.', "that's", 'drivers', 'near', 'c
lanton,', 'ala.,', 'saw', 'monday.', 'instead,', 'motorists', 'seeing', 'sign', 'app
arently', 'hacked', 'display', 'words', '"reclaim', 'america,"', 'white', 'nationali
st', 'slogan,', '"patriot', 'front', 'us,"', 'referencing', 'white', 'supremacist',
'group', 'involved', 'deadly', '2017', 'unite', 'right', 'rally', 'charlottesvill
e.', '"how', 'come', 'about?"', 'wrote', 'sarah', 'hughes,', 'motorist', 'captured',
'photos', 'sign', 'posted', 'twitter.', '"weird', 'hell."', "contractor's", 'portabl
e', 'message', 'board', 'hacked', '65', 'chilton', 'county,', 'ala.,', 'monday', 'af
ternoon,', 'john', 'mcwilliams,', 'spokesman', 'alabama', 'department', 'transportat
ion', '(aldot)', 'west', 'central', 'region,', 'washington', 'post', 'statement.',
'"a', 'citizen', 'alerted', 'nearby', 'state', 'trooper', 'message,', 'contacted',
'aldot,"', 'mcwilliams', 'tuesday.', '"aldot', 'personnel', 'immediately', 'responde
d', 'turned', 'message', 'board', 'off.', 'message', 'boards', '65', 'affected."',
'mcwilliams', 'added', 'aldot', 'investigating', 'white', 'supremacist', 'language',
'appeared', 'sign', 'near', 'clanton,', '40', 'miles', 'northwest', 'montgomery,',
'ala.', 'officials', 'given', 'immediate', 'indication', 'responsible', 'apparentl
y', 'hacking', 'interstate', 'sign.', 'first', 'al.com.', 'hughes', 'post', 'drivin
g', 'home', 'birmingham', 'weekend', "alabama's", 'gulf', 'coast', 'saw', 'white',
'supremacist', 'messages', 'recently', 'popped', 'around', 'home', 'city', 'supporte
rs', 'patriot', 'front.', '"when', 'saw', 'it,', 'thought,', "'oh,", "guys,'", '"',
'hughes,', '31', 'year', 'old', 'attorney.', '"i', 'kind', 'shocked."', 'hacked', 'a
labama', 'road', 'sign', 'comes', 'time', 'president', 'biden', 'declared', 'white',
'supremacy', '"the', 'dangerous', 'terrorist', 'threat"', 'country.', 'commencemen
t', 'address', 'howard', 'university', 'month,', 'biden', 'graduating', 'class', 'hi
storically', 'black', 'university', 'pledged', '"to', 'stand', 'poison', 'white', 's
upremacy,', 'inaugural', 'address', '—', 'single', 'dangerous', 'terrorist', 'threa
t', 'homeland', 'white', 'supremacy."', '"i', 'tell', 'progress', 'toward', 'justic
e', 'often', 'meets', 'ferocious', 'pushback', 'oldest', 'sinister', 'forces,"', 'bi
den', 'may', '13', 'address,', 'quoting', 'donald', "trump's", 'equivocating', 'resp
onse', '2017', 'rally', 'charlottesville', 'killed', '32', 'year', 'old', 'heather',
'heyer', 'injured', '19', 'others.', '"that\'s', 'hate', 'never', 'goes', 'away."',
'biden', 'calls', 'white', 'supremacy', 'greatest', 'terrorism', 'threat', '2024',
'race', 'heats', 'southern', 'poverty', 'law', 'center', '(splc)', 'tracked', 'leas
t', '13', 'hate', 'groups', 'alabama', '2021,', 'including', 'proud', 'boys.', 'disc
ussion', 'surrounding', 'white', 'supremacists', 'white', 'nationalists', 'alabama',
'intensified', 'month', 'sen.', 'tommy', 'tuberville', '(r', 'ala.)', 'people', 'ide
ntified', '"white', 'extremists"', 'white', 'nationalists', 'allowed', 'serve', 'u.
s.', 'armed', 'forces.', 'asked', 'reporter', 'wbhm', 'birmingham', 'whether', 'whit
e', 'nationalists', 'allowed', 'serve', 'military,', 'tuberville', 'replied,', '"wel
l,', 'call', 'that.', 'call', 'americans."', 'tuberville', 'criticized,', 'spokesma
n', 'post', 'senator', '"resents', 'implication', 'people', 'military', 'anything',
'patriots', 'heroes."', 'gop', 'senator', 'says', 'white', 'nationalists', 'militar
y,', "'i", 'call', "americans'", 'patriot', 'front,', 'white', 'supremacist', 'grou
p', 'whose', 'name', 'displayed', 'interstate', 'sign,', 'texas', 'based', 'hate',
'group', 'broke', 'vanguard', 'america', 'formed', 'charlottesville', 'rally,', 'spl
c', 'says.', 'members', 'chanted', '"reclaim', 'america"', 'rallies', 'coeur', "d'al
ene,", 'idaho,', 'washington', 'boston', 'recent', 'years,', 'reports.', 'patriot',
'front', 'responsible', '"the', 'vast', 'majority', 'white', 'supremacist', 'propaga
nda', 'distributed', 'united', 'states"', 'since', '2019,', 'anti', 'defamation', 'l
eague.', 'first', 'time', 'language', 'promoting', 'patriot', 'front', 'made', 'wa
y', 'public', 'space', 'alabama.', 'july,', 'graffiti', 'beneath', 'birmingham', 'br
idge', 'appeared', '"patriot', 'front', 'us"', 'spray', 'painted', 'red', 'blue', 'l
etters,', 'al.com', 'reported.', 'patriot', 'front', 'graffiti', 'also', 'spotted',
'birmingham,', 'city', 'population', "that's", 'nearly', '70', 'percent', 'black,',

```
'u.s.', 'census', 'data.', 'photo', 'posted', 'twitter', 'month', 'showed', 'patrio
t', 'front', 'graffiti', 'along', 'red', 'mountain', 'expressway', 'birmingham', 'wo
rds,', '"we', 'dare', 'defend', 'rights."', 'patriot', 'front', 'graffiti', 'later',
'removed,', 'message', 'left', 'sydney', 'duncan,', 'attorney', 'director', 'magic',
'legal', 'center', 'birmingham,', 'saddened', 'hate', 'become', 'public', 'parts',
'alabama.', '"white', 'supremacy', 'alive', 'well,"', 'duncan', 'wrote.', 'hughes',
'traveling', 'north', 'birmingham', 'pulled', '65', 'take', 'photos', 'messages', 's
ign.', 'seen', 'confederate', 'monuments', 'flags', 'drive', 'before,', 'kind', 'mes
saging', 'government', 'owned', 'property', 'different,', 'said.', 'police', 'office
r', 'already', 'scene', 'waved', 'keep', 'driving,', 'hughes', 'added.', 'returned',
'home,', 'hughes', 'felt', 'compelled', 'share', 'images', 'due', 'ongoing', 'conver
sation', 'happening', 'among', 'birmingham', 'residents', 'promotion', 'patriot', 'f
ront', 'public', 'spaces.', '"some', 'people', 'might', 'perceive', 'upsetting', 'sc
ary,', 'sign', 'worsening', 'country,"', 'said.', '"but', 'strategy,', 'really', 'im
pressed."', 'added,', '"they\'re', 'dying', 'breed."', 'toluse', 'olorunnipa', 'az
```

Display no stop words

In [36]:
```
#uniq_tok(df_col=slct_tbl_full_df03['no_sw'])
```

Rejoin semi_processed tokens

In [37]:
```
slct_tbl_full_df03['no_sw_join'] = slct_tbl_full_df03['no_sw'].apply(" ".join)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())

for c in range(0,1):
    print(slct_tbl_full_df03['no_sw_join'][c])
```

```
(4026, 19)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

travelers alabama driving interstate 65 parties barbecues memorial day might seen me ssages digital road signs honoring veterans died fighting united states. that's driv ers near clanton, ala., saw monday. instead, motorists seeing sign apparently hacked display words "reclaim america," white nationalist slogan, "patriot front us," refer encing white supremacist group involved deadly 2017 unite right rally charlottesvill e. "how come about?" wrote sarah hughes, motorist captured photos sign posted twitte r. "weird hell." contractor's portable message board hacked 65 chilton county, ala., monday afternoon, john mcwilliams, spokesman alabama department transportation (aldo t) west central region, washington post statement. "a citizen alerted nearby state t rooper message, contacted aldot," mcwilliams tuesday. "aldot personnel immediately r esponded turned message board off. message boards 65 affected." mcwilliams added ald ot investigating white supremacist language appeared sign near clanton, 40 miles nor thwest montgomery, ala. officials given immediate indication responsible apparently hacking interstate sign. first al.com. hughes post driving home birmingham weekend a labama's gulf coast saw white supremacist messages recently popped around home city supporters patriot front. "when saw it, thought, 'oh, guys,' " hughes, 31 year old a ttorney. "i kind shocked." hacked alabama road sign comes time president biden decla red white supremacy "the dangerous terrorist threat" country. commencement address h oward university month, biden graduating class historically black university pledged "to stand poison white supremacy, inaugural address — single dangerous terrorist thr eat homeland white supremacy." "i tell progress toward justice often meets ferocious pushback oldest sinister forces," biden may 13 address, quoting donald trump's equiv ocating response 2017 rally charlottesville killed 32 year old heather heyer injured 19 others. "that's hate never goes away." biden calls white supremacy greatest terro rism threat 2024 race heats southern poverty law center (splc) tracked least 13 hate groups alabama 2021, including proud boys. discussion surrounding white supremacists white nationalists alabama intensified month sen. tommy tuberville (r ala.) people i dentified "white extremists" white nationalists allowed serve u.s. armed forces. ask ed reporter wbhm birmingham whether white nationalists allowed serve military, tuber ville replied, "well, call that. call americans." tuberville criticized, spokesman p ost senator "resents implication people military anything patriots heroes." gop sena tor says white nationalists military, 'i call americans' patriot front, white suprem acist group whose name displayed interstate sign, texas based hate group broke vangu ard america formed charlottesville rally, splc says. members chanted "reclaim americ a" rallies coeur d'alene, idaho, washington boston recent years, reports. patriot fr ont responsible "the vast majority white supremacist propaganda distributed united s tates" since 2019, anti defamation league. first time language promoting patriot fro nt made way public space alabama. july, graffiti beneath birmingham bridge appeared "patriot front us" spray painted red blue letters, al.com reported. patriot front gr affiti also spotted birmingham, city population that's nearly 70 percent black, u.s. census data. photo posted twitter month showed patriot front graffiti along red moun tain expressway birmingham words, "we dare defend rights." patriot front graffiti la ter removed, message left sydney duncan, attorney director magic legal center birmin gham, saddened hate become public parts alabama. "white supremacy alive well," dunca n wrote. hughes traveling north birmingham pulled 65 take photos messages sign. seen confederate monuments flags drive before, kind messaging government owned property d ifferent, said. police officer already scene waved keep driving, hughes added. retur ned home, hughes felt compelled share images due ongoing conversation happening amon g birmingham residents promotion patriot front public spaces. "some people might per ceive upsetting scary, sign worsening country," said. "but strategy, really impresse

Remove punctuation

In [38]:
```
punctuation = set(punctuation) # speeds up comparison
#print(punctuation)

# Add special hyphen mark
tw_punct = punctuation - {"#"}
#print(tw_punct)

# Remove hash and at symbols for later capture of hashtag info
tw_punct = tw_punct - {"@"}
tw_punct = tw_punct - {"-"}
#tw_punct = tw_punct - {"/"}
tw_punct.add("’")
tw_punct.add("‘")
tw_punct.add("”")
tw_punct.add("“")
tw_punct.add("…")
tw_punct.add("—")
tw_punct.add("...")
tw_punct.add("€")
tw_punct.add("±")
tw_punct.add("£")
tw_punct.add("¡")
tw_punct.add("§")
tw_punct.add("◉")

print(tw_punct)
```

```
{',', ':', '+', '~', '‘', ';', "'", '}', '.', '¡', '’', '€', '_', '/', ']', '§',
'?', '{', '◉', '±', '%', '(', '...', '!', '$', '[', '=', '>', '”', ')', '“', '`',
'&', '^', '\\', '—', '*', '£', '|', '"', '<', '…'}
```

Create function

In [39]:
```
def remove_punctuation(text, punct_set=tw_punct):
    return("".join([ch for ch in text if ch not in punct_set]))
```

Call function

In [40]:
```
slct_tbl_full_df03['no_sw_join_no_punc'] = slct_tbl_full_df03['no_sw_join']\
.apply(remove_punctuation, punct_set=tw_punct)

print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())

for c in range(0,1):
    try:
        print(slct_tbl_full_df03['no_sw_join_no_punc'][c], '\n')
    except:
        print(f'\nerror on {c}\n')
```

```
(4026, 20)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

travelers alabama driving interstate 65 parties barbecues memorial day might seen me ssages digital road signs honoring veterans died fighting united states thats driver s near clanton ala saw monday instead motorists seeing sign apparently hacked displa y words reclaim america white nationalist slogan patriot front us referencing white supremacist group involved deadly 2017 unite right rally charlottesville how come ab out wrote sarah hughes motorist captured photos sign posted twitter weird hell contr actors portable message board hacked 65 chilton county ala monday afternoon john mcw illiams spokesman alabama department transportation aldot west central region washin gton post statement a citizen alerted nearby state trooper message contacted aldot m cwilliams tuesday aldot personnel immediately responded turned message board off mes sage boards 65 affected mcwilliams added aldot investigating white supremacist langu age appeared sign near clanton 40 miles northwest montgomery ala officials given imm ediate indication responsible apparently hacking interstate sign first alcom hughes post driving home birmingham weekend alabamas gulf coast saw white supremacist messa ges recently popped around home city supporters patriot front when saw it thought oh guys hughes 31 year old attorney i kind shocked hacked alabama road sign comes time president biden declared white supremacy the dangerous terrorist threat country comm encement address howard university month biden graduating class historically black u niversity pledged to stand poison white supremacy inaugural address single dangerou s terrorist threat homeland white supremacy i tell progress toward justice often mee ts ferocious pushback oldest sinister forces biden may 13 address quoting donald tru mps equivocating response 2017 rally charlottesville killed 32 year old heather heye r injured 19 others thats hate never goes away biden calls white supremacy greatest terrorism threat 2024 race heats southern poverty law center splc tracked least 13 h ate groups alabama 2021 including proud boys discussion surrounding white supremacis ts white nationalists alabama intensified month sen tommy tuberville r ala people id entified white extremists white nationalists allowed serve us armed forces asked rep orter wbhm birmingham whether white nationalists allowed serve military tuberville r eplied well call that call americans tuberville criticized spokesman post senator re sents implication people military anything patriots heroes gop senator says white na tionalists military i call americans patriot front white supremacist group whose nam e displayed interstate sign texas based hate group broke vanguard america formed cha rlottesville rally splc says members chanted reclaim america rallies coeur dalene id aho washington boston recent years reports patriot front responsible the vast majori ty white supremacist propaganda distributed united states since 2019 anti defamation league first time language promoting patriot front made way public space alabama jul y graffiti beneath birmingham bridge appeared patriot front us spray painted red blu e letters alcom reported patriot front graffiti also spotted birmingham city populat ion thats nearly 70 percent black us census data photo posted twitter month showed p atriot front graffiti along red mountain expressway birmingham words we dare defend rights patriot front graffiti later removed message left sydney duncan attorney dire ctor magic legal center birmingham saddened hate become public parts alabama white s upremacy alive well duncan wrote hughes traveling north birmingham pulled 65 take ph otos messages sign seen confederate monuments flags drive before kind messaging gove rnment owned property different said police officer already scene waved keep driving hughes added returned home hughes felt compelled share images due ongoing conversati on happening among birmingham residents promotion patriot front public spaces some p eople might perceive upsetting scary sign worsening country said but strategy really impressed added theyre dying breed toluse olorunnipa azi paybarah contributed report

Tokenize

In [41]:
```
slct_tbl_full_df03['no_sw_join_no_punc_tok'] \
= slct_tbl_full_df03['no_sw_join_no_punc'].apply(str.split)
```

```
print(slct_tbl_full_df03.shape)
display(slct_tbl_full_df03.head())

for c in range(0,1):
    print(slct_tbl_full_df03['no_sw_join_no_punc_tok'][c], '\n')
```

```
(4026, 21)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

5 rows × 21 columns

['travelers', 'alabama', 'driving', 'interstate', '65', 'parties', 'barbecues', 'mem
orial', 'day', 'might', 'seen', 'messages', 'digital', 'road', 'signs', 'honoring',
'veterans', 'died', 'fighting', 'united', 'states', 'thats', 'drivers', 'near', 'cla
nton', 'ala', 'saw', 'monday', 'instead', 'motorists', 'seeing', 'sign', 'apparentl
y', 'hacked', 'display', 'words', 'reclaim', 'america', 'white', 'nationalist', 'slo
gan', 'patriot', 'front', 'us', 'referencing', 'white', 'supremacist', 'group', 'inv
olved', 'deadly', '2017', 'unite', 'right', 'rally', 'charlottesville', 'how', 'com
e', 'about', 'wrote', 'sarah', 'hughes', 'motorist', 'captured', 'photos', 'sign',
'posted', 'twitter', 'weird', 'hell', 'contractors', 'portable', 'message', 'board',
'hacked', '65', 'chilton', 'county', 'ala', 'monday', 'afternoon', 'john', 'mcwillia
ms', 'spokesman', 'alabama', 'department', 'transportation', 'aldot', 'west', 'centr
al', 'region', 'washington', 'post', 'statement', 'a', 'citizen', 'alerted', 'nearb
y', 'state', 'trooper', 'message', 'contacted', 'aldot', 'mcwilliams', 'tuesday', 'a
ldot', 'personnel', 'immediately', 'responded', 'turned', 'message', 'board', 'off',
'message', 'boards', '65', 'affected', 'mcwilliams', 'added', 'aldot', 'investigatin
g', 'white', 'supremacist', 'language', 'appeared', 'sign', 'near', 'clanton', '40',
'miles', 'northwest', 'montgomery', 'ala', 'officials', 'given', 'immediate', 'indic
ation', 'responsible', 'apparently', 'hacking', 'interstate', 'sign', 'first', 'alco
m', 'hughes', 'post', 'driving', 'home', 'birmingham', 'weekend', 'alabamas', 'gul
f', 'coast', 'saw', 'white', 'supremacist', 'messages', 'recently', 'popped', 'aroun
d', 'home', 'city', 'supporters', 'patriot', 'front', 'when', 'saw', 'it', 'though
t', 'oh', 'guys', 'hughes', '31', 'year', 'old', 'attorney', 'i', 'kind', 'shocked',
'hacked', 'alabama', 'road', 'sign', 'comes', 'time', 'president', 'biden', 'declare
d', 'white', 'supremacy', 'the', 'dangerous', 'terrorist', 'threat', 'country', 'com
mencement', 'address', 'howard', 'university', 'month', 'biden', 'graduating', 'clas
s', 'historically', 'black', 'university', 'pledged', 'to', 'stand', 'poison', 'whit
e', 'supremacy', 'inaugural', 'address', 'single', 'dangerous', 'terrorist', 'threa
t', 'homeland', 'white', 'supremacy', 'i', 'tell', 'progress', 'toward', 'justice',
'often', 'meets', 'ferocious', 'pushback', 'oldest', 'sinister', 'forces', 'biden',
'may', '13', 'address', 'quoting', 'donald', 'trumps', 'equivocating', 'response',
'2017', 'rally', 'charlottesville', 'killed', '32', 'year', 'old', 'heather', 'heye
r', 'injured', '19', 'others', 'thats', 'hate', 'never', 'goes', 'away', 'biden', 'c
alls', 'white', 'supremacy', 'greatest', 'terrorism', 'threat', '2024', 'race', 'hea
ts', 'southern', 'poverty', 'law', 'center', 'splc', 'tracked', 'least', '13', 'hat
e', 'groups', 'alabama', '2021', 'including', 'proud', 'boys', 'discussion', 'surrou
nding', 'white', 'supremacists', 'white', 'nationalists', 'alabama', 'intensified',
'month', 'sen', 'tommy', 'tuberville', 'r', 'ala', 'people', 'identified', 'white',
'extremists', 'white', 'nationalists', 'allowed', 'serve', 'us', 'armed', 'forces',
'asked', 'reporter', 'wbhm', 'birmingham', 'whether', 'white', 'nationalists', 'allo
wed', 'serve', 'military', 'tuberville', 'replied', 'well', 'call', 'that', 'call',
'americans', 'tuberville', 'criticized', 'spokesman', 'post', 'senator', 'resents',
'implication', 'people', 'military', 'anything', 'patriots', 'heroes', 'gop', 'senat
or', 'says', 'white', 'nationalists', 'military', 'i', 'call', 'americans', 'patrio
t', 'front', 'white', 'supremacist', 'group', 'whose', 'name', 'displayed', 'interst
ate', 'sign', 'texas', 'based', 'hate', 'group', 'broke', 'vanguard', 'america', 'fo
rmed', 'charlottesville', 'rally', 'splc', 'says', 'members', 'chanted', 'reclaim',
'america', 'rallies', 'coeur', 'dalene', 'idaho', 'washington', 'boston', 'recent',
'years', 'reports', 'patriot', 'front', 'responsible', 'the', 'vast', 'majority', 'w
hite', 'supremacist', 'propaganda', 'distributed', 'united', 'states', 'since', '201
9', 'anti', 'defamation', 'league', 'first', 'time', 'language', 'promoting', 'patri
ot', 'front', 'made', 'way', 'public', 'space', 'alabama', 'july', 'graffiti', 'bene
ath', 'birmingham', 'bridge', 'appeared', 'patriot', 'front', 'us', 'spray', 'painte
d', 'red', 'blue', 'letters', 'alcom', 'reported', 'patriot', 'front', 'graffiti',
'also', 'spotted', 'birmingham', 'city', 'population', 'thats', 'nearly', '70', 'per
cent', 'black', 'us', 'census', 'data', 'photo', 'posted', 'twitter', 'month', 'show
ed', 'patriot', 'front', 'graffiti', 'along', 'red', 'mountain', 'expressway', 'birm

```
ingham', 'words', 'we', 'dare', 'defend', 'rights', 'patriot', 'front', 'graffiti',
'later', 'removed', 'message', 'left', 'sydney', 'duncan', 'attorney', 'director',
'magic', 'legal', 'center', 'birmingham', 'saddened', 'hate', 'become', 'public', 'p
arts', 'alabama', 'white', 'supremacy', 'alive', 'well', 'duncan', 'wrote', 'hughe
s', 'traveling', 'north', 'birmingham', 'pulled', '65', 'take', 'photos', 'message
s', 'sign', 'seen', 'confederate', 'monuments', 'flags', 'drive', 'before', 'kind',
'messaging', 'government', 'owned', 'property', 'different', 'said', 'police', 'offi
cer', 'already', 'scene', 'waved', 'keep', 'driving', 'hughes', 'added', 'returned',
'home', 'hughes', 'felt', 'compelled', 'share', 'images', 'due', 'ongoing', 'convers
ation', 'happening', 'among', 'birmingham', 'residents', 'promotion', 'patriot', 'fr
ont', 'public', 'spaces', 'some', 'people', 'might', 'perceive', 'upsetting', 'scar
y', 'sign', 'worsening', 'country', 'said', 'but', 'strategy', 'really', 'impresse
d', 'added', 'theyre', 'dying', 'breed', 'toluse', 'olorunnipa', 'azi', 'paybarah',
'contributed', 'report']
```

Display globally unique tokens on final tokens

In [42]:
```
#uniq_tok(df_col=slct_tbl_full_df03['no_sw_join_no_punc_tok'])
```

Pipeline consolidation

Pipeline function

In [43]:
```
def prepare(text, pipeline):
    '''Run a pipeline of text processing transformers'''
    tokens = str(text)

    # Pull key and val from trans dictionaries
    for transformer in pipeline:
        trans = list(transformer.keys())[0]
        args = list(transformer.values())[0]
        #print(trans)
        #print(args)
        if args == None:
            #print(1)
            tokens = trans(tokens)
        else:
            #print('check99', trans, args)
            tokens = trans(tokens, args)

    return(tokens)
```

article_text preprocessing _ w/o lemmatization

In [44]:
```
'''Set transformer pipeline 1:
Caseloading, normalization (using textacy), special ch removal,
split on whitespace, stop word removal, rejoin,
remove custom punctuation, tokenize
'''
transformers01 = [{str.lower: None},
                  {normalize: None},
```

```
                    {rex_replace: None},
                    {rex_url: None},
                    {emoji_split: None},
                    {str.split: None},
                    {sw_remover: None},
                    {" ".join: None},
                    {remove_punctuation: tw_punct},
                    {str.split: None},
                    {" ".join: None},
                   ]

# Apply transformers to pandas dataframe, w/ new col containing tokens
slct_tbl_full_df04['processed_text'] = slct_tbl_full_df04['article_text']\
.progress_apply(prepare, pipeline=transformers01)

slct_tbl_full_df04['processed_text_split'] = slct_tbl_full_df04['processed_text']\
.progress_apply(str.split)

slct_tbl_full_df04['num_tokens'] = slct_tbl_full_df04['processed_text_split']\
.map(len)

display(slct_tbl_full_df04.head())

# Review unique tokens across entire dataset
for c in range(0,1):
    try:
        print(slct_tbl_full_df04['processed_text'][c], '\n')
    except:
        print(f'Skip {c}')
```

```
100%|████████| 4026/4026 [00:27<00:00, 143.88it/s]
100%|██████| 4026/4026 [00:00<00:00, 15319.36it/s]
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

travelers alabama driving interstate 65 parties barbecues memorial day might seen me ssages digital road signs honoring veterans died fighting united states thats driver s near clanton ala saw monday instead motorists seeing sign apparently hacked displa y words reclaim america white nationalist slogan patriot front us referencing white supremacist group involved deadly 2017 unite right rally charlottesville how come ab out wrote sarah hughes motorist captured photos sign posted twitter weird hell contr actors portable message board hacked 65 chilton county ala monday afternoon john mcw illiams spokesman alabama department transportation aldot west central region washin gton post statement a citizen alerted nearby state trooper message contacted aldot m cwilliams tuesday aldot personnel immediately responded turned message board off mes sage boards 65 affected mcwilliams added aldot investigating white supremacist langu age appeared sign near clanton 40 miles northwest montgomery ala officials given imm ediate indication responsible apparently hacking interstate sign first alcom hughes post driving home birmingham weekend alabamas gulf coast saw white supremacist messa ges recently popped around home city supporters patriot front when saw it thought oh guys hughes 31 year old attorney i kind shocked hacked alabama road sign comes time president biden declared white supremacy the dangerous terrorist threat country comm encement address howard university month biden graduating class historically black u niversity pledged to stand poison white supremacy inaugural address single dangerous terrorist threat homeland white supremacy i tell progress toward justice often meets ferocious pushback oldest sinister forces biden may 13 address quoting donald trumps equivocating response 2017 rally charlottesville killed 32 year old heather heyer in jured 19 others thats hate never goes away biden calls white supremacy greatest terr orism threat 2024 race heats southern poverty law center splc tracked least 13 hate groups alabama 2021 including proud boys discussion surrounding white supremacists w hite nationalists alabama intensified month sen tommy tuberville r ala people identi fied white extremists white nationalists allowed serve us armed forces asked reporte r wbhm birmingham whether white nationalists allowed serve military tuberville repli ed well call that call americans tuberville criticized spokesman post senator resent s implication people military anything patriots heroes gop senator says white nation alists military i call americans patriot front white supremacist group whose name di splayed interstate sign texas based hate group broke vanguard america formed charlot tesville rally splc says members chanted reclaim america rallies coeur dalene idaho washington boston recent years reports patriot front responsible the vast majority w hite supremacist propaganda distributed united states since 2019 anti defamation lea gue first time language promoting patriot front made way public space alabama july g raffiti beneath birmingham bridge appeared patriot front us spray painted red blue l etters alcom reported patriot front graffiti also spotted birmingham city population thats nearly 70 percent black us census data photo posted twitter month showed patri ot front graffiti along red mountain expressway birmingham words we dare defend righ ts patriot front graffiti later removed message left sydney duncan attorney director magic legal center birmingham saddened hate become public parts alabama white suprem acy alive well duncan wrote hughes traveling north birmingham pulled 65 take photos messages sign seen confederate monuments flags drive before kind messaging governmen t owned property different said police officer already scene waved keep driving hugh es added returned home hughes felt compelled share images due ongoing conversation h appening among birmingham residents promotion patriot front public spaces some peopl e might perceive upsetting scary sign worsening country said but strategy really imp ressed added theyre dying breed toluse olorunnipa azi paybarah contributed report

Display globally unqiue tokens on final tokens

In [45]:
```
#uniq_tok(df_col=slct_tbl_full_df04['processed_text_split'])
```

article_text preprocessing_w/ lemmatization

'''Set transformer pipeline 2: Caseloading, normalization (using textacy), special ch removal, lemmitization, stop word removal, rejoin, remove custom punctuation, tokenize ''' transformers02 = [{str.lower: None}, {normalize: None}, {rex_replace: None}, {lemma: None}, {" ".join: None}, {rex_url: None}, {emoji_split: None}, {str.split: None}, {sw_remover: None}, {" ".join: None}, {remove_punctuation: tw_punct}, {str.split: None}, {" ".join: None},] # Apply transformers to pandas dataframe, w/ new col containing tokens slct_tbl_full_df04['processed_lemmas'] = slct_tbl_full_df04['article_text']\ .progress_apply(prepare, pipeline=transformers02) slct_tbl_full_df04['processed_lemmas_split'] = slct_tbl_full_df04['processed_lemmas']\ .progress_apply(str.split) slct_tbl_full_df04['num_lemmas'] = slct_tbl_full_df04['processed_lemmas_split']\ .map(len) display(slct_tbl_full_df04.head()) # Review unique tokens across entire dataset for c in range(0,1): try: print(slct_tbl_full_df04['processed_lemmas'][c], '\n') except: print(f'Skip {c}')

Display globally unqiue tokens on final tokens

In [46]:
```
#uniq_tok(df_col=slct_tbl_full_df04['processed_lemmas_split'])
```

Calculate concentration ratio of each set of corpora

In [47]:
```
display(slct_tbl_full_df04['political_lean'].value_counts())

slct_tbl_full_df04_left = slct_tbl_full_df04[\
                                            slct_tbl_full_df04[\
                                               'political_lean'] == 'left']

print(slct_tbl_full_df04_left.shape)
#display(slct_tbl_full_df04_left.head())

slct_tbl_full_df04_right = slct_tbl_full_df04[\
                                             slct_tbl_full_df04[\
                                               'political_lean'] == 'right']

print(slct_tbl_full_df04_right.shape)
#display(slct_tbl_full_df04_right.head())

slct_tbl_full_df04_left_s1 = list(itertools.chain.from_iterable(
    list(pd.Series(slct_tbl_full_df04_left['processed_text_split']))))
print(slct_tbl_full_df04_left_s1[:10])
slct_tbl_full_df04_right_s1 = list(itertools.chain.from_iterable(
    list(pd.Series(slct_tbl_full_df04_right['processed_text_split']))))
print(slct_tbl_full_df04_right_s1[:10])
```

```
right    2758
left     1268
Name: political_lean, dtype: int64
(1268, 14)
(2758, 14)
['travelers', 'alabama', 'driving', 'interstate', '65', 'parties', 'barbecues', 'mem
orial', 'day', 'might']
['family', 'jennifer', 'farber', 'dulos', 'released', 'statement', 'wednesday', 'mar
king', 'four', 'years']
```

In [48]:
```
def concen_ratio(artist_lst=[],
                 lsts=[]):
```

```
lyr_corp_lst = []
for l in lsts:
    print(type(l))
    lyr_corp_lst.append(' '.join(l))
print(len(lyr_corp_lst))
#print(lyr_corp_lst)

cv = CountVectorizer(input='content',
                     encoding='utf-8',
                     stop_words=None,
                     token_pattern=r'\S+'
                    )

lyr_tokens_fit = cv.fit(lyr_corp_lst)

print(pd.Series(cv.get_feature_names_out()).sample(15))

lyr_tokens_sm = cv.transform(lyr_corp_lst)
display(lyr_tokens_sm)

df = pd.DataFrame(lyr_tokens_sm.toarray(),
                  columns=cv.get_feature_names_out())
#display(df)

df02 = df.copy()
df02['r_sum'] = df02.sum(axis=1)
#display(df02)

'''Filter by frequency for all columns citation:
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/;
https://pandas.pydata.org/pandas-docs/stable/reference/api/
pandas.DataFrame.ge.html'''
condition = df.ge(5).all()
#print(condition)

# Get the list of columns that satisfy the condition
columns = condition[condition].index.tolist()
#print(columns)
columns.append('r_sum')
#print(columns)

#display(df02[columns])

df03 = df02[columns].copy()
display(df03)

'''Filter by frequency for all columns & add summary row citation:
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/'''
df04 = df03.apply(lambda x: x / df03.iloc[:,-1], axis=0)
#display(df04)

# Create new rows by dividing one artist row by the second artists row
new_row01 = df04.iloc[0] / df04.iloc[1]
new_row02 = df04.iloc[1] / df04.iloc[0]

# Append the new row to the DataFrame
```

```
    df04 = df04.append(new_row01, ignore_index=True)
    df04 = df04.append(new_row02, ignore_index=True)
    display(df04)

    # Transpose dataframe
    df05 = df04.T
    df05 = df05.reset_index()
    df05.columns = ['token',
                    'c1_concen',
                    'c2_concen',
                    'c1c2_concen_ratio',
                    'c2c1_concen_ratio']
    #print(df05)

    '''Sort values citation:
    https://pandas.pydata.org/pandas-docs/stable/reference/api
    /pandas.DataFrame.sort_values.html'''
    print(artist_lst[0])
    display(df05[['token',
                  'c1c2_concen_ratio']].sort_values(by='c1c2_concen_ratio',
                                                    ascending=False).head(10))
    print(artist_lst[1])
    display(df05[['token',
                  'c2c1_concen_ratio']].sort_values(by='c2c1_concen_ratio',
                                                    ascending=False).head(10))

concen_ratio(artist_lst=['Left-Right Concentration Ratio',
                         'Right-Left Concentration Ratio'],
             lsts=[slct_tbl_full_df04_left_s1,
                   slct_tbl_full_df04_right_s1])
```

```
<class 'list'>
<class 'list'>
2
19846            feuding
20622            forgave
1809              516000
13151            coulter
6846             barroom
45411          shoplifted
48509         suspecttime
54602               wout
4016              agarro
25526             impart
3902          aeronautics
48385         supportable
14863             delores
5273           appetizing
2580     @desantiswarroom
dtype: object
<2x55704 sparse matrix of type '<class 'numpy.int64'>'
        with 79403 stored elements in Compressed Sparse Row format>
```

	#2	#metoo	0	07	1	10	100	1000	10000	100000	...	zero	zip	zone	zones	
0	6	18	16	5	452	397	138	65	52	45	...	67	6	38	7	
1	5	8	36	6	600	638	226	82	124	64	...	97	11	38	11	

2 rows × 10958 columns

```
C:\Users\acarr\AppData\Local\Temp\ipykernel_23812\3142308763.py:59: FutureWarning: T
he frame.append method is deprecated and will be removed from pandas in a future ver
sion. Use pandas.concat instead.
  df04 = df04.append(new_row01, ignore_index=True)
C:\Users\acarr\AppData\Local\Temp\ipykernel_23812\3142308763.py:60: FutureWarning: T
he frame.append method is deprecated and will be removed from pandas in a future ver
sion. Use pandas.concat instead.
  df04 = df04.append(new_row02, ignore_index=True)
```

	#2	#metoo	0	07	1	10	100	1000	
0	7.9407e-06	2.3822e-05	2.1175e-05	6.6172e-06	0.0006	0.0005	0.0002	8.6024e-05	6.88
1	4.7559e-06	7.6094e-06	3.4242e-05	5.7070e-06	0.0006	0.0006	0.0002	7.7996e-05	1.17
2	1.6697e+00	3.1306e+00	6.1839e-01	1.1595e+00	1.0482	0.8658	0.8496	1.1029e+00	5.83
3	5.9893e-01	3.1943e-01	1.6171e+00	8.6245e-01	0.9540	1.1550	1.1770	9.0668e-01	1.713

4 rows × 10958 columns

Left-Right Concentration Ratio

	token	c1c2_concen_ratio
723	anonymity	17.3923
9885	thomass	16.1400
10774	willis	14.1921
6201	mehta	13.6046
3115	docket	13.0790
4349	ginni	12.2442
7421	pork	11.5948
10224	uncertainty	9.9537
4873	hush	9.7397
4400	gorsuch	9.7397

Right-Left Concentration Ratio

	token	c2c1_concen_ratio
755	ap	19.4052
6725	nyc	17.4886
8200	reparations	16.8897
579	aliens	12.6972
1699	ccp	11.4994
5910	locker	11.0681
2057	comey	9.5828
5975	lula	9.1995
1530	busch	8.8940
3957	fidelity	8.7683

KWIC

In [49]:
```
def kwic(doc_series, keyword, window=35, print_samples=5):
    '''Search article text for keywords in context (KWIC)'''
    def add_kwic(text):
        kwic_list.extend(keyword_in_context(doc=text,
                                            keyword=keyword,
                                            ignore_case=True,
                                            window_width=window))
    kwic_list = []
    doc_series.map(add_kwic)

    if print_samples is None or print_samples==0:
        return kwic_list
    else:
        k = min(print_samples, len(kwic_list))
        print(f"{k} random samples out of {len(kwic_list)}" + \
              f" contexts for '{keyword}':")
        for sample in random.sample(list(kwic_list), k):
            print(re.sub(r'[\n\t]', ' ', sample[0]) + ' ' +\
                  sample[1] + ' ' +\
                  re.sub(r'[\n\t]', ' ', sample[2]))
```

In [50]:
```
#kwic(slct_tbl_full_df03['article_text'], 'amp', window=150, print_samples=120)

kwic(slct_tbl_full_df03['norm'], 'economic', window=50, print_samples=20)
```

```
20 random samples out of 947 contexts for 'economic':
government took one step back from self-inflicted  economic  disaster on tuesday. ho
use republicans avoided th
re black americans have significant political and  economic  clout. the resulting
displace
rtainty to data flows that underpin transatlantic  economic  ties, society, and our
international cooperation.
egon have experienced homelessness as a result of  economic  hardship, a shortage of
safe and affordable housi
while, trump has long polled better than biden on  economic  issues and immigration,
with americans more confi
orhood. the operation was eventually called off.   economic  crisis   the cricket le
gend-turned-politician has
ou. that's also why they've cut off access to key  economic  databases and cracked d
own on journalists. click
 facts are clear. almost every element of biden's  economic  policy has a "buy ameri
ca" component to it. its g
hold their labor is protected by the nlra even if  economic  injury results." althou
gh barrett's opinion occup
an, who is battling for a third term, buffeted by  economic  headwinds and criticism
that the impact of the fe
 areas: health care, health behaviors, social and  economic  factors, environment an
d public policy. "we hoped
we flatter condemn us to ostracism, to financial,  economic , and monetary weakness
forever," maduro
n criticized president biden on environmental and  economic  policies, has said he w
ould announce by year's en
tions to arrive at a bill that ultimately avoided  economic  disaster. through it al
l, some democrats have gru
                the two leaders called for closer  economic  and security cooperatio
n amid growing threats fro
ia.  the elections also take place amid a serious  economic  crisis and what analyst
s say is democratic erosio
f jobs.   "if they fail to do it, we will have an  economic  and financial catastrop
he that will be of our own
x news app she said she is also worried about the  economic  effects of these purcha
ses, especially in her hom
his deal, we now have a clear runway to sell that  economic  vision and implement hi
storic investments across
orkers, said elise gould, senior economist at the  economic  policy institute, a thi
nk tank that advocates for
```

Train/test split

In [51]:
```
slct_tbl_full_df04['stratifier'] = slct_tbl_full_df04['political_lean']\
.astype(str) + slct_tbl_full_df04['source_name'].astype(str)
slct_tbl_full_df04['stratifier'] = slct_tbl_full_df04['stratifier']\
.map(str.lower)
display(slct_tbl_full_df04.head())

y01a = ['stratifier']
slct_tbl_full_df04_y01_vc01a = slct_tbl_full_df04[y01a].to_numpy()
print(slct_tbl_full_df04_y01_vc01a.shape)

y01 = ['political_lean']
```

```
slct_tbl_full_df04_y01_vc01 = slct_tbl_full_df04[y01].to_numpy()
print(slct_tbl_full_df04_y01_vc01.shape)
```

	index	source_name	author	title	u
0	0	The Washington Post	NaN	Alabama Highway sign hacked with white suprema...	https://www.washingtonpost.com/nation/2023/05/
1	1	The Washington Post	Amber Phillips	Breaking down the GOP investigation into the B...	https://www.washingtonpost.com/politics/2023/0
2	2	The Washington Post	David Ovalle	Appeals court paves way for Purdue Pharma opio...	https://www.washingtonpost.com/health/2023/05/
3	3	The Washington Post	Philip Bump	Trump pledges to win an immigration fight he d...	https://www.washingtonpost.com/politics/2023/0
4	5	The Washington Post	Paul Waldman	Why fear of change will drive the GOP presiden...	https://www.washingtonpost.com/opinions/2023/0

```
(4026, 1)
(4026, 1)
```

In [52]:
```
nlm_train_x01, \
nlm_test_x01, \
nlm_train_y01, \
nlm_test_y01 = train_test_split(slct_tbl_full_df04['processed_text'],
                                slct_tbl_full_df04_y01_vc01,
                                test_size=.15,
                                random_state=1699,
                                stratify=slct_tbl_full_df04_y01_vc01a
                               )

nlm_train_y01 = nlm_train_y01.ravel()
nlm_test_y01 = nlm_test_y01.ravel()

print(f'{nlm_train_x01.shape}')
print(f'{nlm_train_y01.shape}')
print(f'\n{nlm_test_x01.shape}')
print(f'{nlm_test_y01.shape}')
```

```
(3422,)
(3422,)

(604,)
(604,)
```

lem_train_x01, \ lem_test_x01, \ lem_train_y01, \ lem_test_y01 = train_test_split(slct_tbl_full_df04['processed_lemmas'], slct_tbl_full_df04_y01_vc01, test_size=.15, random_state=1699, stratify=slct_tbl_full_df04_y01_vc01a) lem_train_y01 = lem_train_y01.ravel() lem_test_y01 = lem_test_y01.ravel() print(f'{lem_train_x01.shape}') print(f'{lem_train_y01.shape}') print(f'\n{lem_test_x01.shape}') print(f'{lem_test_y01.shape}')

TF-IDF

In [53]:
```
print(slct_tbl_full_df04['processed_text'].shape)
print(slct_tbl_full_df04['processed_text'].head())
```

```
(4026,)
0    travelers alabama driving interstate 65 partie...
1    federal prosecutor may nearing decision whethe...
2    federal appeals court tuesday cleared way drug...
3    speaking orlando november 2015 republican pres...
4    look know countrys going wrong direction flori...
Name: processed_text, dtype: object
```

In [54]:
```
nlm_tfidf = TfidfVectorizer(encoding='utf-8',
                            analyzer='word',
                            stop_words=sw,
                            token_pattern=r'(?u)\b\w\w+\b',
                            ngram_range=(1,3),
                            max_df=.7,
                            min_df=5)

nlm_train_x01_mtx = nlm_tfidf.fit_transform(nlm_train_x01)
nlm_test_x01_mtx = nlm_tfidf.transform(nlm_test_x01)

display(nlm_train_x01_mtx)
display(nlm_test_x01_mtx)
```

```
<3422x48932 sparse matrix of type '<class 'numpy.float64'>'
        with 1256997 stored elements in Compressed Sparse Row format>
<604x48932 sparse matrix of type '<class 'numpy.float64'>'
        with 215091 stored elements in Compressed Sparse Row format>
```

lem_tfidf = TfidfVectorizer(encoding='utf-8', analyzer='word', stop_words=sw, token_pattern=r'(?u)\b\w\w+\b', ngram_range=(1,3), max_df=.7, min_df=5) lem_train_x01_mtx = lem_tfidf.fit_transform(lem_train_x01) lem_test_x01_mtx = lem_tfidf.transform(lem_test_x01) display(lem_train_x01_mtx) display(lem_test_x01_mtx)

In [55]:
```
def display_samp_dwm(sm=None,
                     vec=None,
                     n=(1,1),
                     rs_tup=(1,1)):
    mtx_df01 = pd.DataFrame(sm.toarray(),
                            columns=vec.get_feature_names_out())

    mtx_df01a = mtx_df01.sample(n=n[0],
                                random_state=rs_tup[0],
```

```
                                axis=1)

    mtx_df01b = mtx_df01a.sample(n=n[1],
                                 random_state=rs_tup[1],
                                 axis=0)

    display(mtx_df01b)
    return vec.get_feature_names_out()
```

In [56]:
```
rs_tup=(1699,1699)
```

In [57]:
```
nlm_train_x01_mtx_cols = display_samp_dwm(sm=nlm_train_x01_mtx,
                                          vec=nlm_tfidf,
                                          n=(17,11),
                                          rs_tup=(5,1699))
```

	nixon	follow london	efforts block	federal employees	request meeting	sexual assaults	chief executive	almost half	senior fellow	de sant
1098	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2370	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
1091	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
3084	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2413	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
287	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
855	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2084	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
1537	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
991	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2245	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0

display_samp_dwm(sm=lem_train_x01_mtx, vec=lem_tfidf, n=(17,11), rs_tup=(5,1699))

Modeling

Algorithm setup

Gradient Boosting Classifier _ Using BayesSearchCV

In [ ]:
```
from sklearn.model_selection import RepeatedStratifiedKFold
```

```
# Start timer script
start_time = dt.datetime.today()

# Citation: Hochberg, 2018; Shanmukh, 2021
m2v1_gbc_pip = Pipeline([('gbc',
                          GradientBoostingClassifier(random_state=1699))])

loss_hparam = Categorical(['log_loss', 'exponential'])
lrate_hparam = Real(1e-3, 1e3, prior='log-uniform')
nest_hparam = Integer(1e2, 1e3, prior='log-uniform')
msamp_hparam = Real(.01, .95, prior='log-uniform')
mdepth_hparam = Integer(1, 20, prior='log-uniform')
mfeat_hparam = Categorical(['sqrt', 'log2', None])
#wstart_hparam = Categorical([True, False])
#calph_hparam = Real(0.0, 100.0, prior='log-uniform')
                #'gbc__warm_start': wstart_hparam
                #'gbc__ccp_alpha': calph_hparam

m2v1_gbc_grd = {'gbc__loss': loss_hparam,
                'gbc__learning_rate': lrate_hparam,
                'gbc__n_estimators': nest_hparam,
                'gbc__min_samples_split': msamp_hparam,
                'gbc__max_depth': mdepth_hparam,
                'gbc__max_features': mfeat_hparam
               }

'''Change GBC default scoring from accuracy to F1 score citation:
https://chat.openai.com/share/254f382b-4a8e-48e8-acd5-2918f0bbc59d
'''
f1_scorer = make_scorer(f1_score,
                        pos_label='right')

'''Customize cross-validation citation:
https://machinelearningmastery.com
/scikit-optimize-for-hyperparameter-tuning-in-machine-learning/
'''
cv = RepeatedStratifiedKFold(n_splits=10,
                             n_repeats=2,
                             random_state=1699)

m2v1_gbc = BayesSearchCV(m2v1_gbc_pip,
                         m2v1_gbc_grd,
                         n_iter=20,
                         scoring=f1_scorer,
                         cv=cv,
                         n_jobs=3,
                         refit=True,
                         verbose=4,
                         random_state=1699)

m2v1_gbc.fit(nlm_train_x01_mtx, nlm_train_y01)

# End timer script
end_time = dt.datetime.today()
time_elapse = end_time - start_time
print(f'Start Time = {start_time}')
```

```
print(f'End Time = {end_time}')
print(f'Elapsed Time = {time_elapse}')
```

Pickle best model

In [58]:
```
# Path to save the pickled model
mod_folder_name = 'trained_models'
m2v1_pkl_file_name = 'm2v2_gbc.pkl'

pkl_file_path01 = os.path.join(curr_dir, mod_folder_name, m2v1_pkl_file_name)

print(f'CSV file 1 in path: {pkl_file_path01}')
```

```
CSV file 1 in path: C:\Users\acarr\Documents\GitHub\ADS509_Final_project\deliverable
s\trained_models\m2v2_gbc.pkl
```

with open(pkl_file_path01, "wb") as file: pickle.dump(m2v1_gbc, file) print("Model pickled and saved successfully.")

Load pickled best model

In [59]:
```
with open(pkl_file_path01, 'rb') as file:
    m2v1_gbc = pickle.load(file)
```

In [60]:
```
print(f'\nBest Estimator:\n{m2v1_gbc.best_estimator_}')

print('\nCross-validaton results:')
display(pd.DataFrame(m2v1_gbc.cv_results_))

train_m2v1_gbc_y01_pred = m2v1_gbc.predict_proba(nlm_train_x01_mtx)
print(f'\nFirst 10 train set predictions:\n{train_m2v1_gbc_y01_pred[:10]}')

test_m2v1_gbc_y01_pred = m2v1_gbc.predict_proba(nlm_test_x01_mtx)
print(f'\nFirst 10 test set predictions:\n{test_m2v1_gbc_y01_pred[:10]}')

print(f'\nBest Score for "{m2v1_gbc.scorer_}" is {m2v1_gbc.best_score_}')
```

```
Best Estimator:
Pipeline(steps=[('gbc',
                 GradientBoostingClassifier(learning_rate=0.8373240042701702,
                                            loss='exponential', max_depth=11,
                                            max_features='sqrt',
                                            min_samples_split=0.582912491747238,
                                            random_state=1699))])

Cross-validaton results:
```

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_gbc__learning_rate
0	12.9224	1.2860	0.0274	0.0070	0.0162
1	10.5480	0.9983	0.0296	0.0091	0.035
2	2.9297	0.1453	0.0285	0.0057	55.3871
3	426.3148	43.3892	0.0356	0.0089	4.7311
4	5.6830	0.4884	0.0306	0.0039	25.6546
5	13.7448	1.6360	0.0610	0.0080	23.2484
6	11.9362	1.3784	0.0471	0.0051	173.3432
7	248.6976	19.5832	0.0346	0.0038	36.8407
8	7.2956	0.6681	0.0286	0.0028	274.3051
9	4.1264	0.3624	0.0399	0.0042	0.5714
10	4.9026	0.4603	0.0376	0.0042	0.8373
11	3.6591	0.2537	0.0421	0.0059	0.001
12	31.3106	3.3239	0.0366	0.0057	0.0012
13	531.8490	80.7165	0.0326	0.0094	0.001
14	2.5695	0.1020	0.0237	0.0081	0.0462

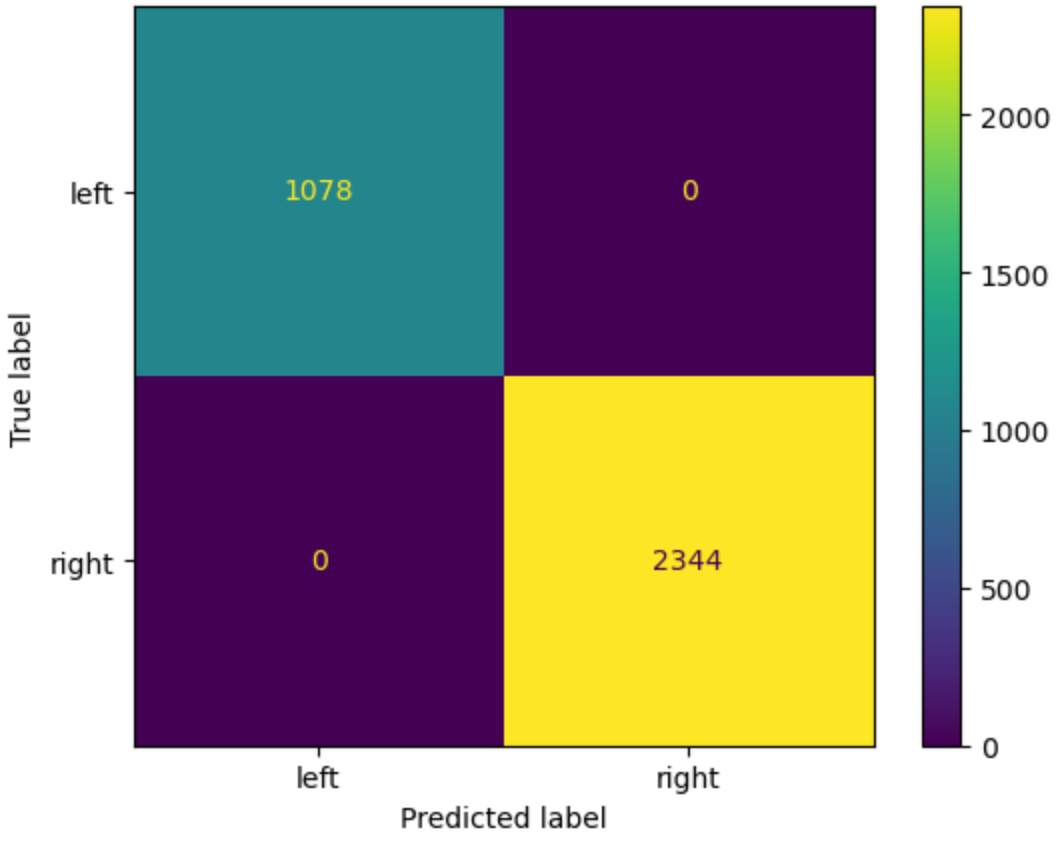
15 rows × 24 columns

```
First 10 train set predictions:
[[0.0001 0.9999]
 [0.0012 0.9988]
 [0.992  0.008 ]
 [0.     1.    ]
 [0.     1.    ]
 [0.0063 0.9937]
 [0.     1.    ]
 [0.0003 0.9997]
 [0.0023 0.9977]
 [0.9662 0.0338]]

First 10 test set predictions:
[[0.0091 0.9909]
 [0.4118 0.5882]
 [1.     0.    ]
 [0.9997 0.0003]
 [1.     0.    ]
 [0.     1.    ]
 [0.9996 0.0004]
 [0.6635 0.3365]
 [0.9917 0.0083]
 [0.0063 0.9937]]
```

Train set check

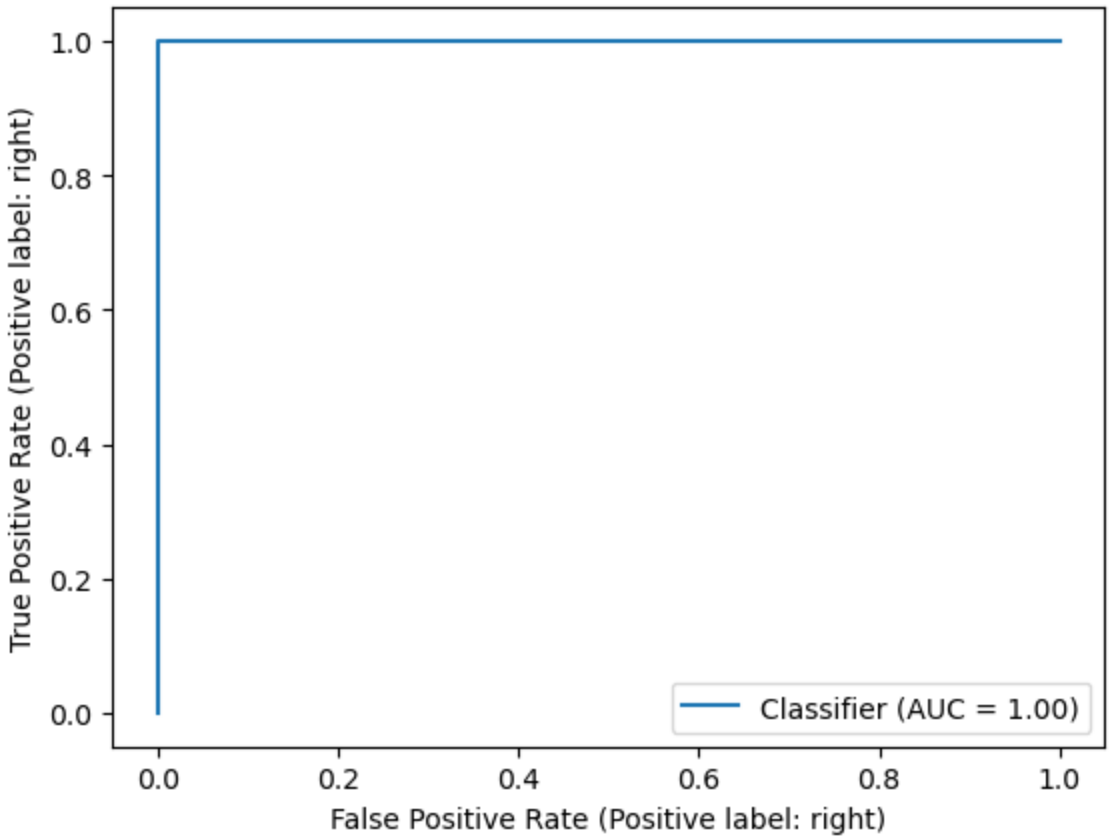
In [61]:
```
nlm_train_y01_pred = m2v1_gbc.predict(nlm_train_x01_mtx)
nlm_train_y01_pred_cm = confusion_matrix(nlm_train_y01, nlm_train_y01_pred)

print(classification_report(nlm_train_y01, nlm_train_y01_pred))
print(nlm_train_y01_pred_cm)

'''Citation:
https://scikit-learn.org/stable/modules/generated
/sklearn.metrics.ConfusionMatrixDisplay.html
#sklearn.metrics.ConfusionMatrixDisplay.plot
'''
nlm_train_cm_dsp = ConfusionMatrixDisplay(confusion_matrix=nlm_train_y01_pred_cm,
                                          display_labels=m2v1_gbc.classes_)
nlm_train_cm_dsp.plot()
plt.show()
```

```
              precision    recall  f1-score   support

        left       1.00      1.00      1.00      1078
       right       1.00      1.00      1.00      2344

    accuracy                           1.00      3422
   macro avg       1.00      1.00      1.00      3422
weighted avg       1.00      1.00      1.00      3422

[[1078    0]
 [   0 2344]]
```



ROC-AUC Curve

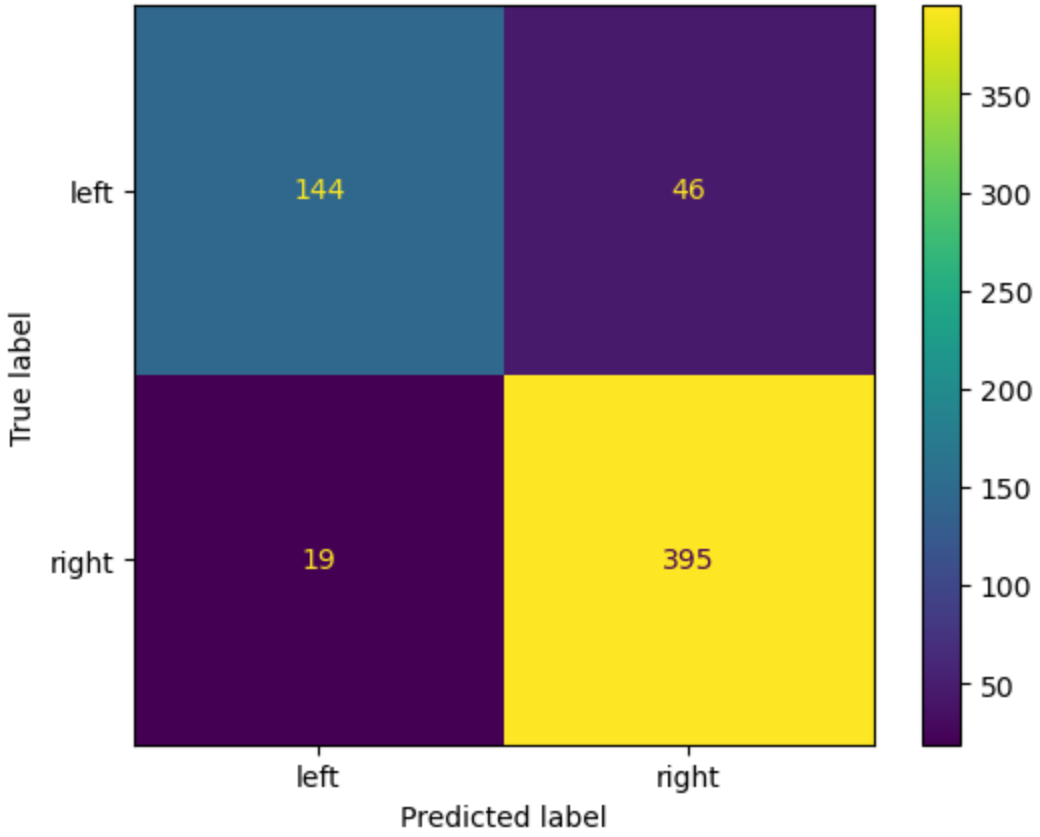
In [62]:
```
nlm_train_y01_pred_decf = m2v1_gbc.decision_function(nlm_train_x01_mtx)
RocCurveDisplay.from_predictions(nlm_train_y01, nlm_train_y01_pred_decf,
                                 pos_label='right')
plt.show()
```



Test set results

In [63]:
```
nlm_test_y01_pred = m2v1_gbc.predict(nlm_test_x01_mtx)
nlm_test_y01_pred_cm = confusion_matrix(nlm_test_y01, nlm_test_y01_pred)

print('Test Set Evaluation Metrics')
print(classification_report(nlm_test_y01, nlm_test_y01_pred))
print(nlm_test_y01_pred_cm)

'''Citation:
https://scikit-learn.org/stable/modules/generated
/sklearn.metrics.ConfusionMatrixDisplay.html
#sklearn.metrics.ConfusionMatrixDisplay.plot
'''
nlm_test_cm_dsp = ConfusionMatrixDisplay(confusion_matrix=nlm_test_y01_pred_cm,
                                         display_labels=m2v1_gbc.classes_)
nlm_test_cm_dsp.plot()
plt.show()
```

```
Test Set Evaluation Metrics
              precision    recall  f1-score   support

        left       0.88      0.76      0.82       190
       right       0.90      0.95      0.92       414

    accuracy                           0.89       604
   macro avg       0.89      0.86      0.87       604
weighted avg       0.89      0.89      0.89       604

[[144  46]
 [ 19 395]]
```



Variable importance

In [64]:

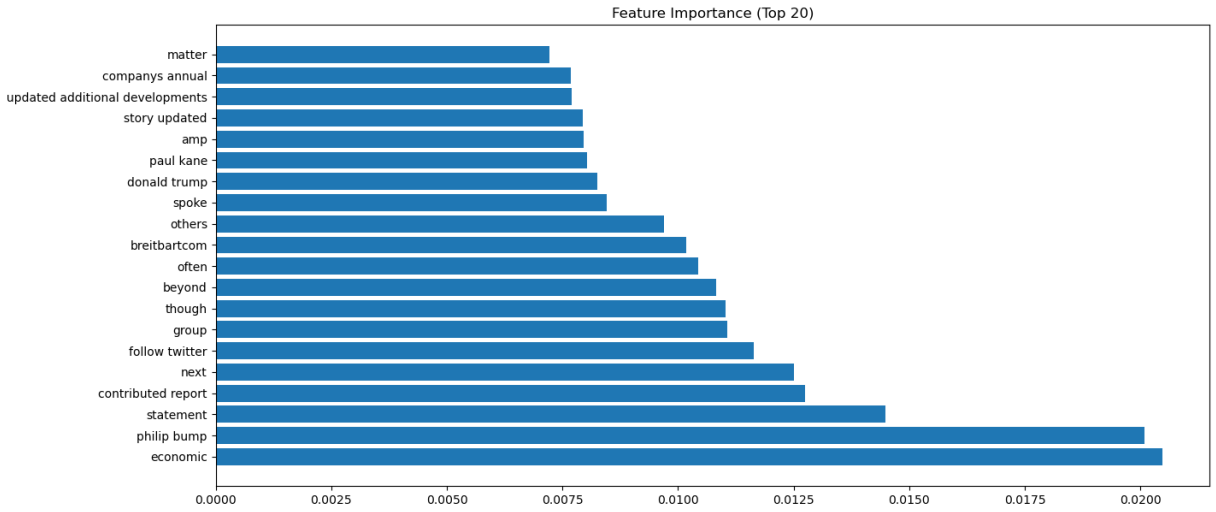
```
print(nlm_train_x01_mtx_cols)
print(type(nlm_train_x01_mtx_cols))
print(nlm_train_x01_mtx_cols.shape)

x = m2v1_gbc.best_estimator_.named_steps['gbc'].feature_importances_
x_df01 = pd.DataFrame(x, columns=['var_imp'])
x_df01['feature'] = nlm_train_x01_mtx_cols
x_df02 = x_df01.sort_values(by=['var_imp'], ascending=False)
x_df03 = x_df02.head(20)

display(x_df02.head())
print(type(x_df02))
print(x_df02.shape)
```

```
['03' '04' '05' ... 'zoos' 'zucker' 'zuckerberg']
<class 'numpy.ndarray'>
(48932,)
```

	var_imp	feature
14105	0.0205	economic
32121	0.0201	philip bump
40955	0.0145	statement
10184	0.0128	contributed report
29331	0.0125	next

```
<class 'pandas.core.frame.DataFrame'>
(48932, 2)
```

In [65]:
```
'''Citation:
https://machinelearningmastery.com/calculate-feature-importance-with-python/
'''
# plot feature importance
#figure = plt.figsize((10,9))
plt.figure(figsize=(15,7))
plt.title('Feature Importance (Top 20)')
plt.barh([x for x in range(len(x_df03['var_imp']))], x_df03['var_imp'],
         tick_label=x_df03['feature'])
plt.show()
```



In [66]:
```
TNmodel1=nlm_test_y01_pred_cm[0][0]
FPmodel1=nlm_test_y01_pred_cm[0][1]
FNmodel1=nlm_test_y01_pred_cm[1][0]
TPmodel1=nlm_test_y01_pred_cm[1][1]
```

In [67]:
```
# Results:
from tabulate import tabulate

TANmodel1=TNmodel1+FPmodel1
TAPmodel1=TPmodel1+FNmodel1
TPPmodel1=FPmodel1+TPmodel1
```

```
TPNmodel1=TNmodel1+FNmodel1
GTmodel1=TANmodel1+TAPmodel1
AccuracyM1=(TNmodel1+TPmodel1)/GTmodel1
ErrorRateM1=1-AccuracyM1
SensitivityM1=TPmodel1/(TAPmodel1)
RecallM1=SensitivityM1
SpecificityM1=TNmodel1/TANmodel1
PrecisionM1=TPmodel1/TPPmodel1
F1M1=2*PrecisionM1*RecallM1/(PrecisionM1 + RecallM1)
F2M1=5*(PrecisionM1*RecallM1)/((4*PrecisionM1)+RecallM1)
Fp5M1=(1.25)*(PrecisionM1*RecallM1)/((0.25*PrecisionM1)+RecallM1)

header = ["Accuracy", "Error Rate", "Sensitivity", "Recall", "Specificity",
          "Precision", "F1", "F2", "F0.5"]
data1 = [["Accuracy", AccuracyM1], ["Error Rate", ErrorRateM1],
         ["Sensitivity", SensitivityM1],
         ["Recall", RecallM1], ["Specificity", SpecificityM1],
         ["Precision", PrecisionM1],
         ["F1", F1M1], ["F2", F2M1], ["F0.5", Fp5M1]]

col_names=["Measurement", "Linear SVC Model"]

ModelEvaluationTable = tabulate(data1, headers=col_names,
                                tablefmt="fancy_grid")

print(ModelEvaluationTable)
```

Measurement	Linear SVC Model
Accuracy	0.892384
Error Rate	0.107616
Sensitivity	0.954106
Recall	0.954106
Specificity	0.757895
Precision	0.895692
F1	0.923977
F2	0.941822
F0.5	0.906795

In [68]:
```
data1
```

Out[68]:
```
[['Accuracy', 0.8923841059602649],
 ['Error Rate', 0.10761589403973515],
 ['Sensitivity', 0.9541062801932367],
 ['Recall', 0.9541062801932367],
 ['Specificity', 0.7578947368421053],
 ['Precision', 0.8956916099773242],
 ['F1', 0.9239766081871345],
 ['F2', 0.9418216499761562],
 ['F0.5', 0.9067952249770431]]
```

In [69]:
```
Data_metric_results_TheHill=pd.DataFrame(data1)
Data_metric_results_TheHill.head()
```

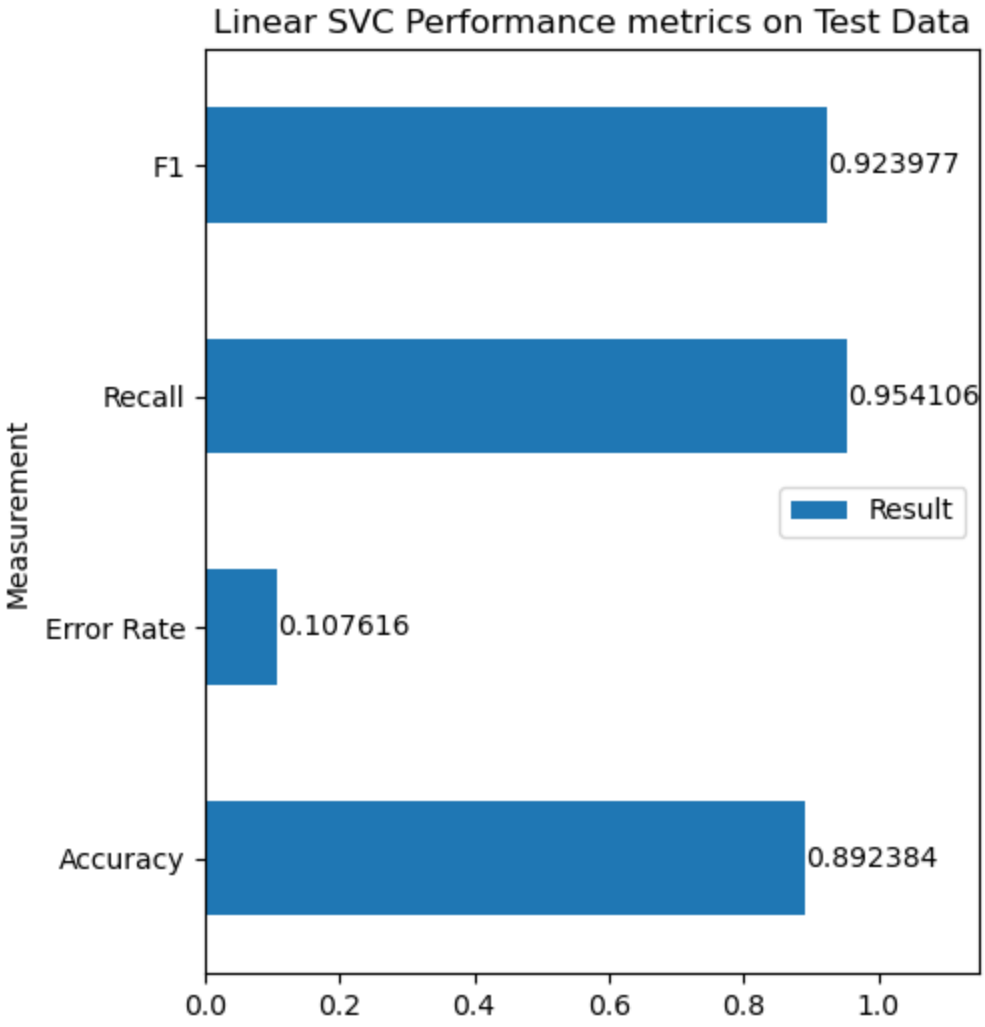
Out[69]:

	0	1
0	Accuracy	0.8924
1	Error Rate	0.1076
2	Sensitivity	0.9541
3	Recall	0.9541
4	Specificity	0.7579

In [70]:
```
Data_metric_results_TheHill.rename (columns = {0:'Measurement'}, inplace=True)
Data_metric_results_TheHill.rename (columns = {1:'Result'}, inplace=True)
```

In [71]:
```
#plt.bar(x=ModelEvaluationTable)
ax=Data_metric_results_TheHill[(Data_metric_results_TheHill['Measurement'] == 'Accu
                               (Data_metric_results_TheHill['Measurement'] == 'Recall'
                               (Data_metric_results_TheHill['Measurement'] == 'F1') |
                               (Data_metric_results_TheHill['Measurement'] == 'Error R

                                     figsize=(5,6),
                                     title='Linear SVC Performance metrics on Test Dat
ax.bar_label(ax.containers[0])
ax.set_xlim(right=1.15)
```

Out[71]:
```
(0.0, 1.15)
```



Business problem application

In [72]:
```
center_df01 = pd.read_csv(file_in_path02)

print(center_df01.shape)
display(center_df01.head())
```

```
(181, 12)
```

	Unnamed: 0.1	Unnamed: 0	Source	Author	Title	
0	0	0	The Hill	Zach Schonfeld	Ketanji Brown Jackson issues solo dissent in r...	https://thehill.com/regulation/cou
1	1	1	The Hill	Brett Samuels	How Biden pulled it off...	https://thehill.com/homenews/admir
2	2	2	The Hill	the hill	How Christie could be wildcard in 2024 race...	https://thehill.com/homenews/campaig
3	3	3	The Hill	Alexander Bolton	Schumer announces agreement to pass debt ceili...	https://thehill.com/homenews/senate/
4	4	4	The Hill	Zack Budryk	Kaine introduces amendment to strip Manchin-ba...	https://thehill.com/policy/energy-er

In [73]:
```
# Apply transformers to pandas dataframe, w/ new col containing tokens
center_df01['processed_text'] = center_df01['article_text']\
.progress_apply(prepare, pipeline=transformers01)

center_df01['processed_text_split'] = center_df01['processed_text']\
.progress_apply(str.split)

center_df01['num_tokens'] = center_df01['processed_text_split']\
.map(len)

display(center_df01.head())

# Review unique tokens across entire dataset
for c in range(0,1):
    try:
        print(center_df01['processed_text'][c], '\n')
    except:
        print(f'Skip {c}')
```

```
100%|██████████| 181/181 [00:01<00:00, 128.38it/s]
100%|██████████| 181/181 [00:00<00:00, 25704.92it/s]
```

	Unnamed: 0.1	Unnamed: 0	Source	Author	Title	
0	0	0	The Hill	Zach Schonfeld	Ketanji Brown Jackson issues solo dissent in r...	https://thehill.com/regulation/cou
1	1	1	The Hill	Brett Samuels	How Biden pulled it off...	https://thehill.com/homenews/admin
2	2	2	The Hill	the hill	How Christie could be wildcard in 2024 race...	https://thehill.com/homenews/campaig
3	3	3	The Hill	Alexander Bolton	Schumer announces agreement to pass debt ceili...	https://thehill.com/homenews/senate/
4	4	4	The Hill	Zack Budryk	Kaine introduces amendment to strip Manchin-ba...	https://thehill.com/policy/energy-en

```
liberal justice ketanji brown jackson issued first solo dissent supreme court merits
case thursday disagreeing colleagues labor dispute ruling makes easier companies sue
worker strikes 8 1 decision high court overturned lower ruling found federal union l
aws preempted concrete company glacier northwest bringing lawsuit international brot
herhood teamsters represents companys truck drivers jackson wrote courts no business
delving particular labor dispute time the majority also misapplies boards cases mann
er threatens impede boards uniform development labor law erode right strike jackson
dissented case arose union directed drivers go strike morning company mixing concret
e loading onto trucks making deliveries concrete mixed day ruined glacier sued union
damages state court 1959 supreme court precedent san diego building trades council v
garmon national labor relations act nlra federal law governs strikes collective barg
aining preempts state law two arguably conflict union got lawsuit tossed washington
supreme court garmon glacier northwest appealed nations highest court majority opini
on authored conservative justice amy coney barrett joined four colleagues court rule
d nlra preempt lawsuit strike take reasonable precautions protect companys property
foreseeable imminent danger the unions actions resulted destruction concrete glacier
prepared day also posed risk foreseeable aggravated imminent harm glaciers trucks un
ion took affirmative steps endanger glaciers property rather reasonable precautions
mitigate risk nlra arguably protect conduct barrett wrote three additional conservat
ive justices justices samuel alito clarence thomas neil gorsuch wrote separately rev
erse unions win grounds jackson hand stood alone dissenting marking first solo disse
nt merits case since joining bench last year jackson has however dissented solo outs
ide courts normal docket jpmorgan says former virgin islands first lady aided epstei
n trump indictment lays bare security risks storage mar lago noted complaint nationa
l labor relations boards general counsel filed state supreme courts ruling alleged g
lacier northwest engaged unfair labor practices relation strike majority found this
issue properly us lower courts addressed significance complaint leaving state courts
consider case proceeds the filing general counsels administrative complaint necessar
ily suffices establish unions strike conduct arguably protected within meaning garmo
n thus general counsels complaint marked end court involvement matter jackson wrote
```

In [74]:
```
nlm_apply_x01_mtx = nlm_tfidf.transform(center_df01['processed_text'])

print(nlm_apply_x01_mtx.shape)
display(nlm_apply_x01_mtx)
```

```
(181, 48932)
<181x48932 sparse matrix of type '<class 'numpy.float64'>'
        with 64886 stored elements in Compressed Sparse Row format>
```

In [75]:
```
display_samp_dwm(sm=nlm_apply_x01_mtx,
                 vec=nlm_tfidf,
                 n=(17,11),
                 rs_tup=(5,1699))
```

	nixon	follow london	efforts block	federal employees	request meeting	sexual assaults	chief executive	almost half	senior fellow	dee santi
27	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
152	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
154	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
131	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
75	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
77	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
170	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0222	0.0
121	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
43	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
48	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0
53	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0000	0.0

Out[75]:
```
array(['03', '04', '05', ..., 'zoos', 'zucker', 'zuckerberg'],
      dtype=object)
```

In [76]:
```
nlm_apply_mtx_pred_prob = m2v1_gbc.predict_proba(nlm_apply_x01_mtx)

print(nlm_apply_mtx_pred_prob.shape)
print(nlm_apply_mtx_pred_prob[:10])

nlm_apply_mtx_pred = m2v1_gbc.predict(nlm_apply_x01_mtx)

print(nlm_apply_mtx_pred.shape)
print(nlm_apply_mtx_pred)
```

```
(181, 2)
[[0.1943 0.8057]
 [0.361  0.639 ]
 [0.9765 0.0235]
 [0.0586 0.9414]
 [0.8779 0.1221]
 [0.7489 0.2511]
 [0.0426 0.9574]
 [0.8892 0.1108]
 [0.87   0.13  ]
 [0.     1.    ]]
(181,)
['right' 'right' 'left' 'right' 'left' 'left' 'right' 'left' 'left'
 'right' 'left' 'left' 'right' 'right' 'left' 'right' 'left' 'right'
 'left' 'right' 'right' 'left' 'right' 'right' 'right' 'right' 'right'
 'left' 'right' 'right' 'left' 'left' 'right' 'left' 'left' 'right'
 'right' 'right' 'right' 'right' 'left' 'left' 'right' 'left' 'left'
 'right' 'right' 'left' 'right' 'left' 'right' 'right' 'right' 'right'
 'right' 'right' 'left' 'right' 'left' 'right' 'left' 'left' 'right'
 'left' 'right' 'left' 'left' 'right' 'right' 'right' 'right' 'right'
 'right' 'right' 'right' 'right' 'right' 'right' 'right' 'right' 'right'
 'left' 'right' 'right' 'left' 'left' 'left' 'left' 'right' 'right' 'left'
 'right' 'right' 'right' 'right' 'right' 'left' 'left' 'right' 'left'
 'left' 'right' 'left' 'left' 'right' 'left' 'left' 'right' 'left' 'left'
 'right' 'right' 'right' 'right' 'left' 'right' 'right' 'right' 'left'
 'right' 'right' 'right' 'left' 'right' 'left' 'right' 'right' 'left'
 'left' 'right' 'left' 'right' 'right' 'right' 'left' 'right' 'right'
 'right' 'left' 'left' 'left' 'left' 'left' 'left' 'right' 'right' 'right'
 'right' 'right' 'left' 'right' 'left' 'right' 'right' 'right' 'left'
 'right' 'left' 'right' 'right' 'right' 'right' 'left' 'right' 'right'
 'right' 'left' 'left' 'right' 'right' 'right' 'right' 'right' 'right'
 'left' 'right' 'right' 'right' 'right' 'right' 'right']
```

In [77]:
```
# Compute the maximum values along the second dimension
max_values = np.amax(nlm_apply_mtx_pred_prob, axis=1)
max_values_df01 = pd.DataFrame(max_values,
                               columns=['decision_prob'])
max_values_df01['pred'] = nlm_apply_mtx_pred
print(max_values_df01.shape)
display(max_values_df01.head())
```

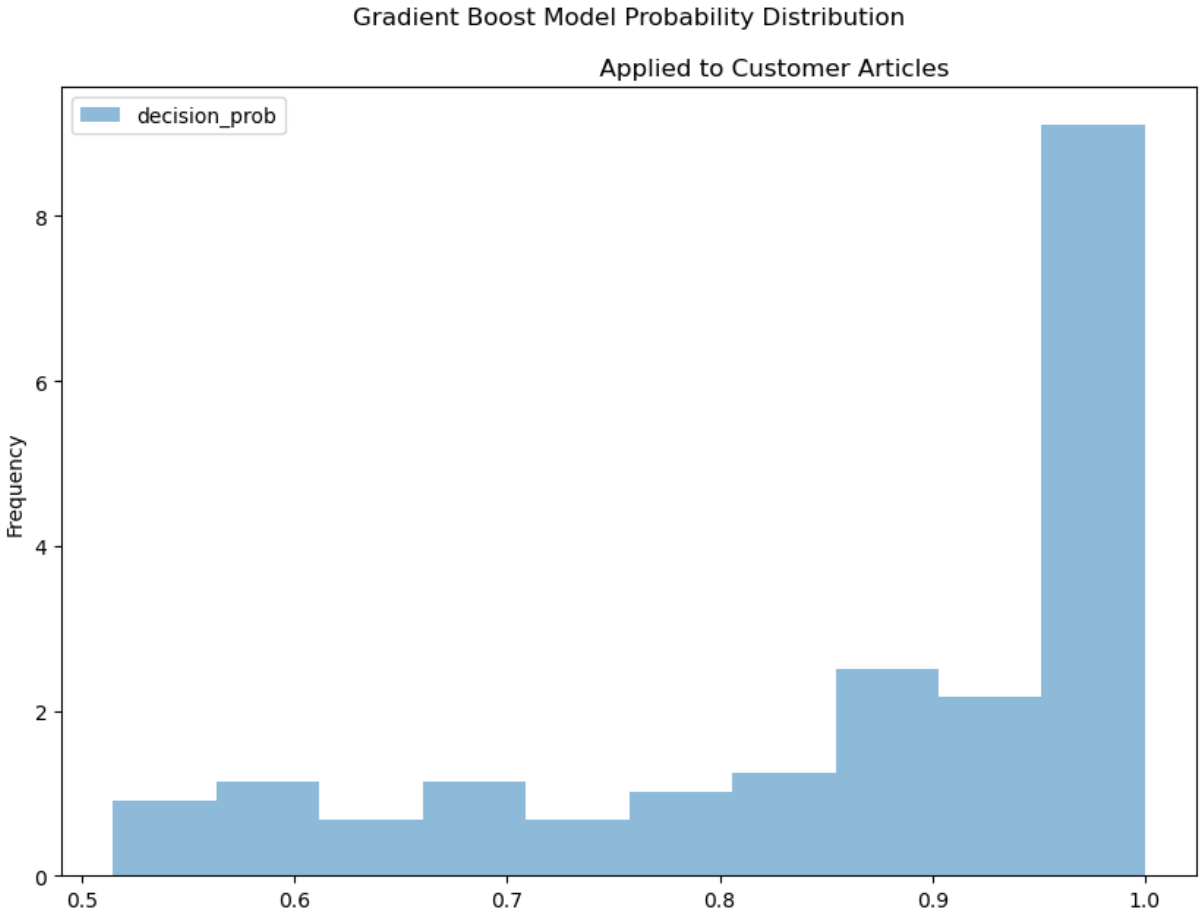
```
(181, 2)
```

	decision_prob	pred
0	0.8057	right
1	0.6390	right
2	0.9765	left
3	0.9414	right
4	0.8779	left

In [78]:
```
max_values_df01['decision_prob'].plot(kind="hist",
                                      density=True,
```

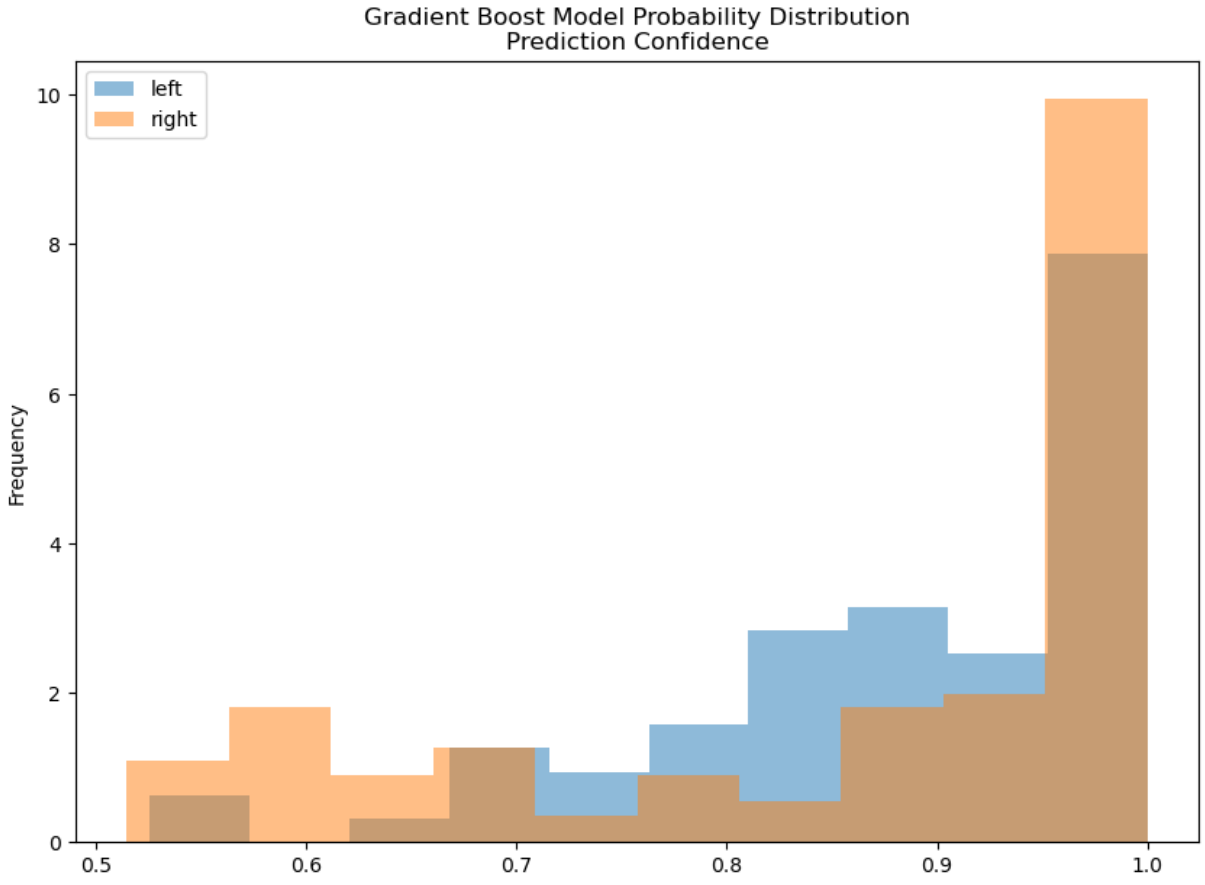
```
                                        alpha=0.5,
                                        legend=True,
                                        figsize=(10,7),
title='''Gradient Boost Model Probability Distribution\n
                                        Applied to Customer Articles''')
```

Out[78]:
```
<Axes: title={'center': 'Gradient Boost Model Probability Distribution\n\n
Applied to Customer Articles'}, ylabel='Frequency'>
```



In [79]:
```
max_values_df01.groupby('pred')['decision_prob'].plot(kind="hist",
                                                      density=True,
                                        alpha=0.5,
                                        legend=True,
                                        figsize=(10,7),
                            title='''Gradient Boost Model Probability Distribution
Prediction Confidence''')
```

Out[79]:
```
pred
left     Axes(0.125,0.11;0.775x0.77)
right    Axes(0.125,0.11;0.775x0.77)
Name: decision_prob, dtype: object
```



In [80]:
```
max_values_df02 = pd.DataFrame(nlm_apply_mtx_pred_prob.round(4),
                               columns=['left', 'right'])
max_values_df02['pred'] = nlm_apply_mtx_pred
max_values_df02
```

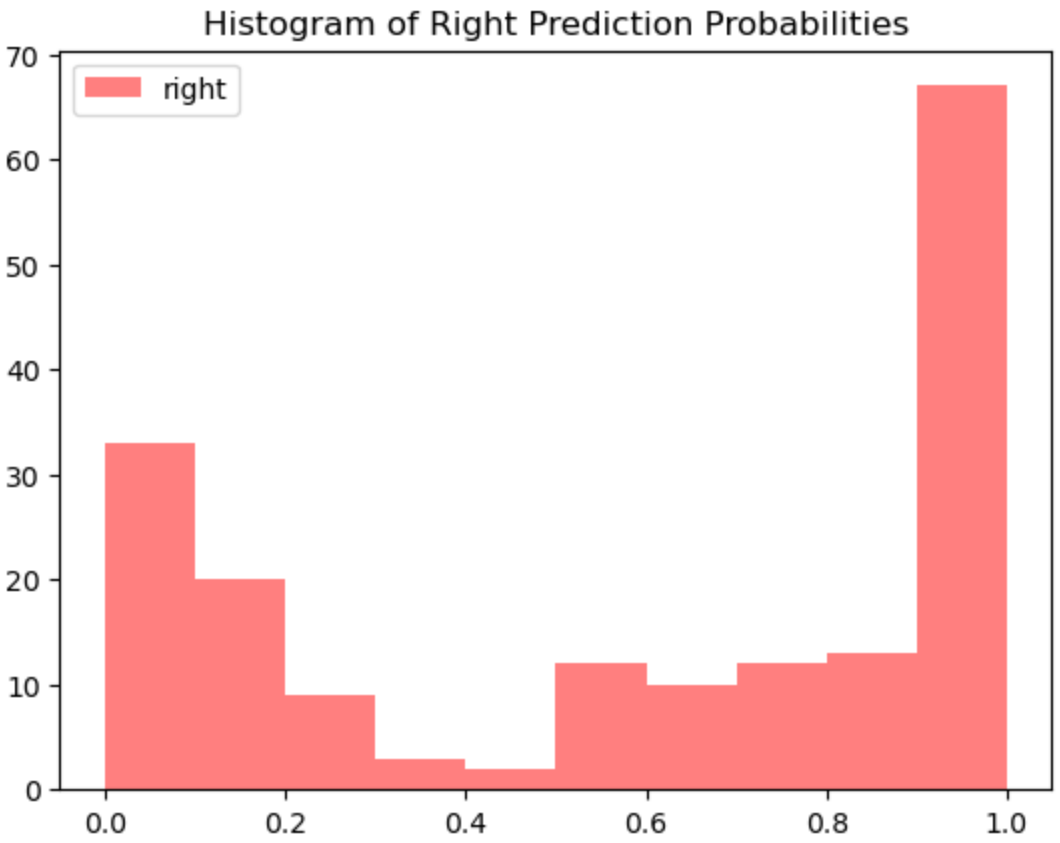
Out[80]:

	left	right	pred
0	0.1943	0.8057	right
1	0.3610	0.6390	right
2	0.9765	0.0235	left
3	0.0586	0.9414	right
4	0.8779	0.1221	left
...	...	...	...
176	0.4365	0.5635	right
177	0.0708	0.9292	right
178	0.0008	0.9992	right
179	0.4365	0.5635	right
180	0.0063	0.9937	right

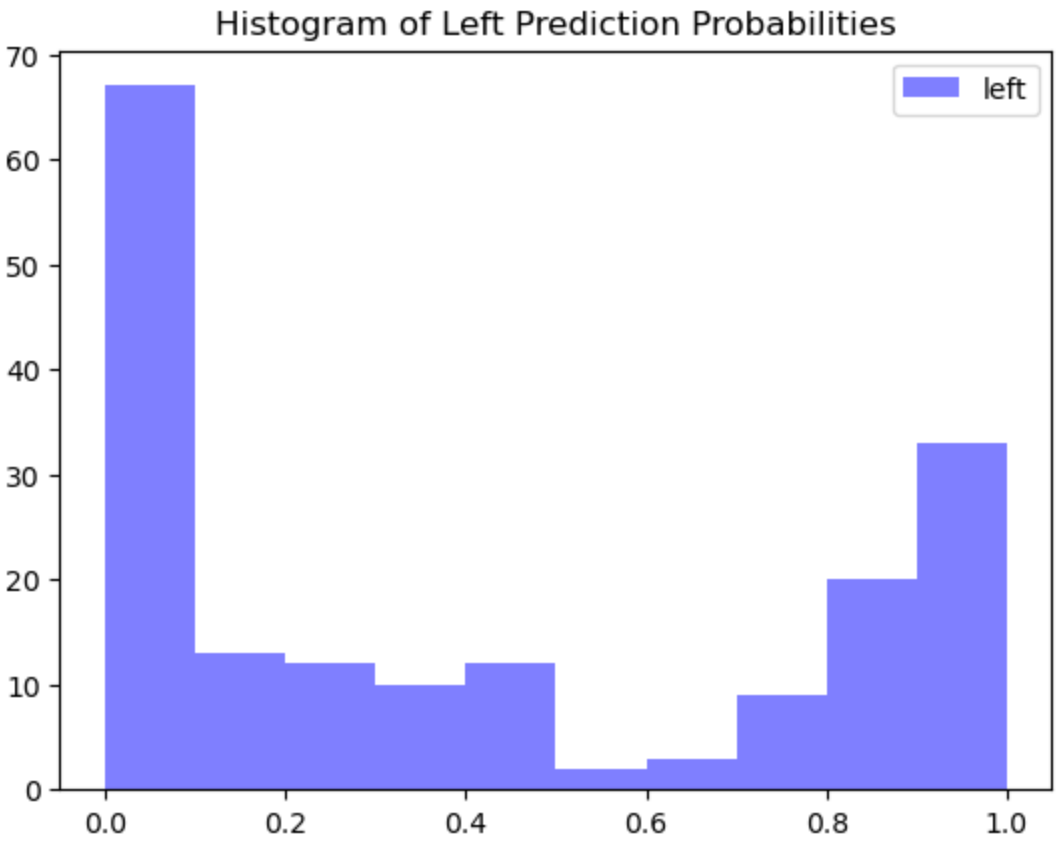
181 rows × 3 columns

In [81]:

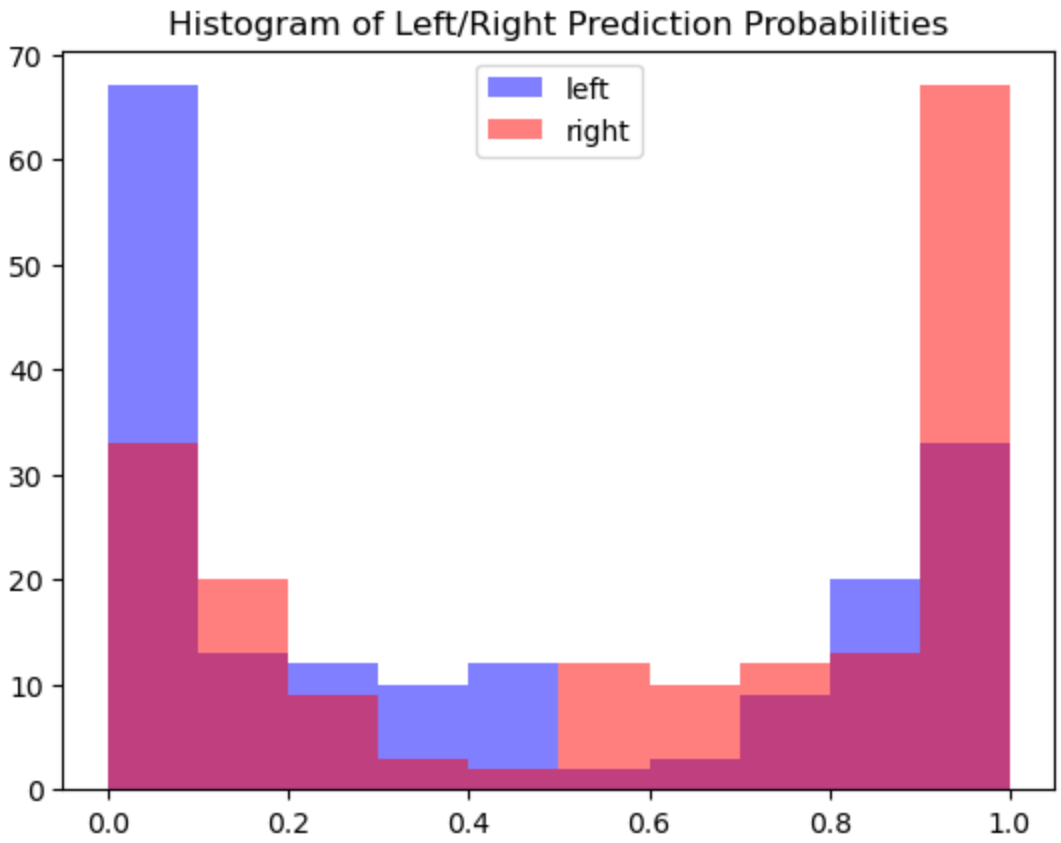
```
# Plotting histograms
#plt.hist(max_values_df02['left'], bins=10, alpha=0.5, color='blue', label='Column
plt.hist(max_values_df02['right'], bins=10, alpha=0.5, color='red',
         label='right')

# Adding legend and title
plt.legend()
plt.title('Histogram of Right Prediction Probabilities')

# Displaying the plot
plt.show()
```



In [82]:
```
# Plotting histograms
#plt.hist(max_values_df02['left'], bins=10, alpha=0.5, color='blue', label='Column
plt.hist(max_values_df02['left'], bins=10, alpha=0.5, color='blue',
         label='left')

# Adding legend and title
plt.legend()
plt.title('Histogram of Left Prediction Probabilities')

# Displaying the plot
plt.show()
```



In [83]:
```
# Plotting histograms
plt.hist(max_values_df02['left'], bins=10,
         alpha=0.5, color='blue', label='left')
plt.hist(max_values_df02['right'], bins=10,
         alpha=0.5, color='red', label='right')

# Adding legend and title
plt.legend()
plt.title('Histogram of Left/Right Prediction Probabilities')

# Displaying the plot
plt.show()
```


Histogram of Left/Right Prediction Probabilities
left
right
70
60
50
40
30
20
10
0
0.0
0.2
0.4
0.6
0.8
1.0