ADS509_CarrA_Final_Project_01_API_v2

June 18, 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 18, 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 18, 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)

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 [ ]: