

# 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 criteria: 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 criteria: 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]: #print(api_record_lst01)
      print(len(api_record_lst01))
```

168

```
[9]: # 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]: '''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()
```



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

```
tbl_names = pd.read_sql('SHOW TABLES', db_conn)
```

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

```
'''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''')
```

## Update news\_articles table from API

```
nat_tbl_name = 'nar_temp'
nwa_tbl_name = 'news_articles'
```

```
'''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()
```





```
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>