Data Knyts Project Report

Anirban Banerjee (aba177), Ayush Raina (ara95), Tanmay Jain (tja34), Siddhartha Haldar (sha285)

Problem Definition

Our goal in this project is to analyze reader engagement for articles published in the New York Times (NYT). We used NYT's API to extract article and comment metadata for the years 2017-2021. We then performed data cleaning, ETL, and feature engineering on this raw data in order to generate our dataset. This data was then used to create visualizations to depict metrics and trends related to user engagement.

Methodology

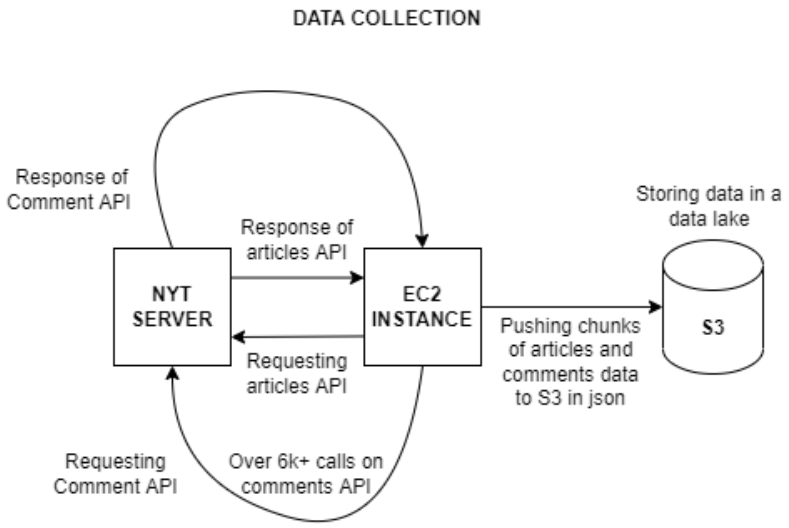
Data Collection

The data we needed for the project had to be collected using the NYT Developer API. There are two main APIs that we used to pull the data off NYT:

1. The articles archive API, which returns all the articles for a certain month, and
2. The other one being the comments API, which returns comments in chunks of 25 for a certain article.

We created two main scripts to perform the task of collecting the data from NYT and uploading it to AWS's S3 data storage. Both the scripts are deployed on EC2 and they keep running indefinitely as background processes.

1. The first script is responsible for connecting to the NYT APIs, reading the data returned, selecting the relevant fields and storing them as JSON files in the EC2 local storage.
2. The second script reads the data files from the EC2 storage, pushes it off to the S3 bucket and finally removes it from the local storage.



ETL

After generating the raw dataset, we performed ETL using PySpark on Google Cloud's Dataflow service. The first step was data cleaning: the article metadata contained multiple nested lists and structures, from which we extracted the fields of interest. Several fields required cleaning - video articles lacked an author, a majority of interactive features were not labelled properly, some articles were missing keywords, and so on. The list of article metadata fields after the ETL are as follows:

News Desk	Section Name	Type of Material	Headline
Abstract	Word Count	Keywords	Publication Date
Author	Web URL	Article ID	Total Comments
Total comment replies		Total comment recommendations	
Total Editor's Selection comments			

The comment metadata was relatively cleaner, and the only cleaning required was to convert all the datetime fields from epoch values to timestamps. The list of comment metadata fields after the ETL process are as follows:

User ID	User Location	Comment Body	Recommendations
Reply Count	Editor's Selection (Y/N)	Date Created	Date Updated
Comment ID	Article ID		

Visualization

We used Plotly with Dash to create the following visualizations from our article/comment data. Some examples of our visualizations can be seen in the 'Results' section.

- Bar chart of top authors by article count, overall and by year.
- Bar chart of top categories by article count, overall and by year.
- Pie chart of most engaging articles by news desk, overall and by year. Recommendations have been used as a measure of engagement.
- Bubble chart of total aggregated comment count versus top authors and categories, overall and by year.
- Bubble chart of total aggregated recommendations versus top authors and categories, overall and by year.
- Choropleth map of comment count versus user location in the United States, by year.
- Line chart of the number of times selected countries (Canada, US, UK, India and China) appear in articles on Covid-19 from 2019 to 2021.
- Bar chart measuring instances of queries, annoyance, emotion and excitement present in comments, overall and by year.

Problems

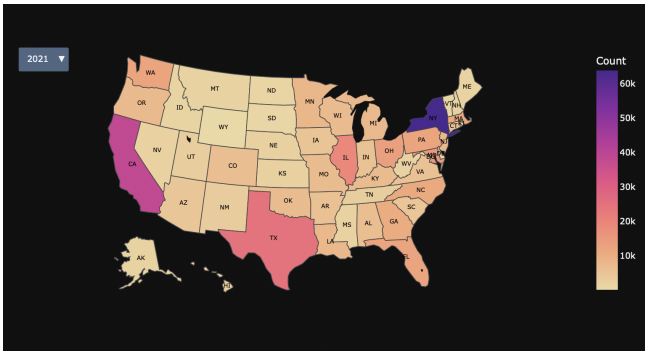
Our reason for using NYT's API instead of directly scraping the data from their site was primarily because accessing NYT articles would have required us to pay for a subscription, whereas obtaining data from their API is free. Scraping would also likely have been an extremely slow and cumbersome process compared to obtaining the data via the API.

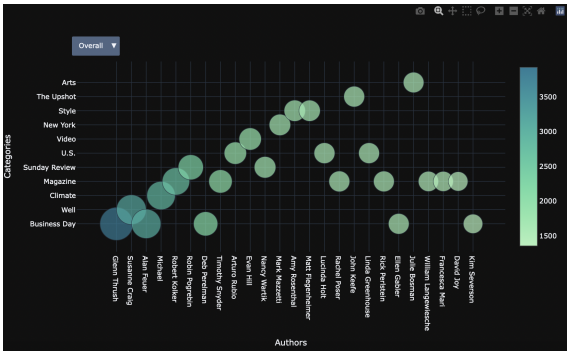
However, NYT places two call limits on its API: 4000 requests per day and 10 requests per minute. To circumvent these limits, we obtained a total of 20 API keys and placed a sleep timer of 300 milliseconds between calls to avoid hitting the requests-per-minute limit. Due to this, our data collection took considerably longer than we had initially anticipated.

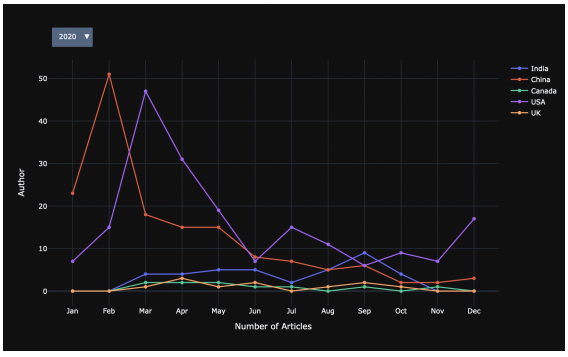
When creating the choropleth map visualization of comment count vs user location, our code would freeze indefinitely when grouping the data by state. Various attempts at optimization (by repartitioning the data on the user location, for example) did not improve performance. Ultimately, we decided to only create visualizations of per-year comment location data (most of our other visualizations include an 'overall' option in addition to 'per-year').

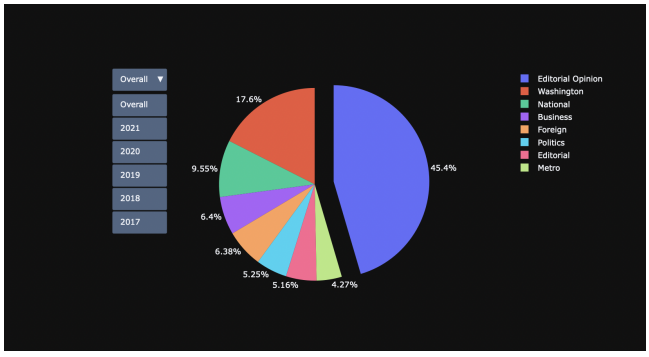
Results

Below are examples of visualizations we created using our data:









Project Summary

- Getting the data: 4
- ETL: 4
- Problem: 1
- Algorithmic Work: 1
- Bigness/Parallelization: 3
- UI: 2
- Visualization: 4
- Technologies: 1