New York Times API

John Cruz

2023-03-24

Import Libraries

```
library(tidyverse)
library(glue)
library(jsonlite)
library(lubridate)
library(ggrepel)
```

Introduction

The New York Times (NYT), provides access to their data through the Times API. With it, data analysis and visualizations can be performed on trends or decisions made within their articles published. This report will be a general use of pulling data and performing some exploratory analysis of what the NYT published.

Connect to New York Times API

Using the *httr* library, this will connect to the New York Times API to query information that was published on the NY Times website. To keep my API key private I incorporated an R Studio API *askForPassword* function.

```
nyt_data_pull <- function(year, month){
  api_cnxn <- fromJSON(glue("https://api.nytimes.com/svc/archive/v1/{year}/{month}.json?api-key={rstudi
}
```

Transform JSON data into Data Frame

This loop will query the API for each month in 2022 using the function connection I previously made.

```
year <- 2022
months <- c(1:12)

last3_month_df <- tibble()

for(month in months){
  last3_month_df <-
    as.data.frame(nyt_data_pull(year, month)) |>
    janitor::clean_names() |>
    rbind(last3_month_df, .data)
}
```

Datetime and Columns of Interest

With the data pulled from the API, a few columns such as published date, section name and keywords will be used to analyze potential trends. The *keywords* column contains a nested list and will be unnested to obtain the keywords. The published date also needs to be transformed from a string to datetime format. The *lubridate* library will assist in parsing the datetime properly. A new column, *hourly_date*, will contain the hour the article was published for easier binning later on.

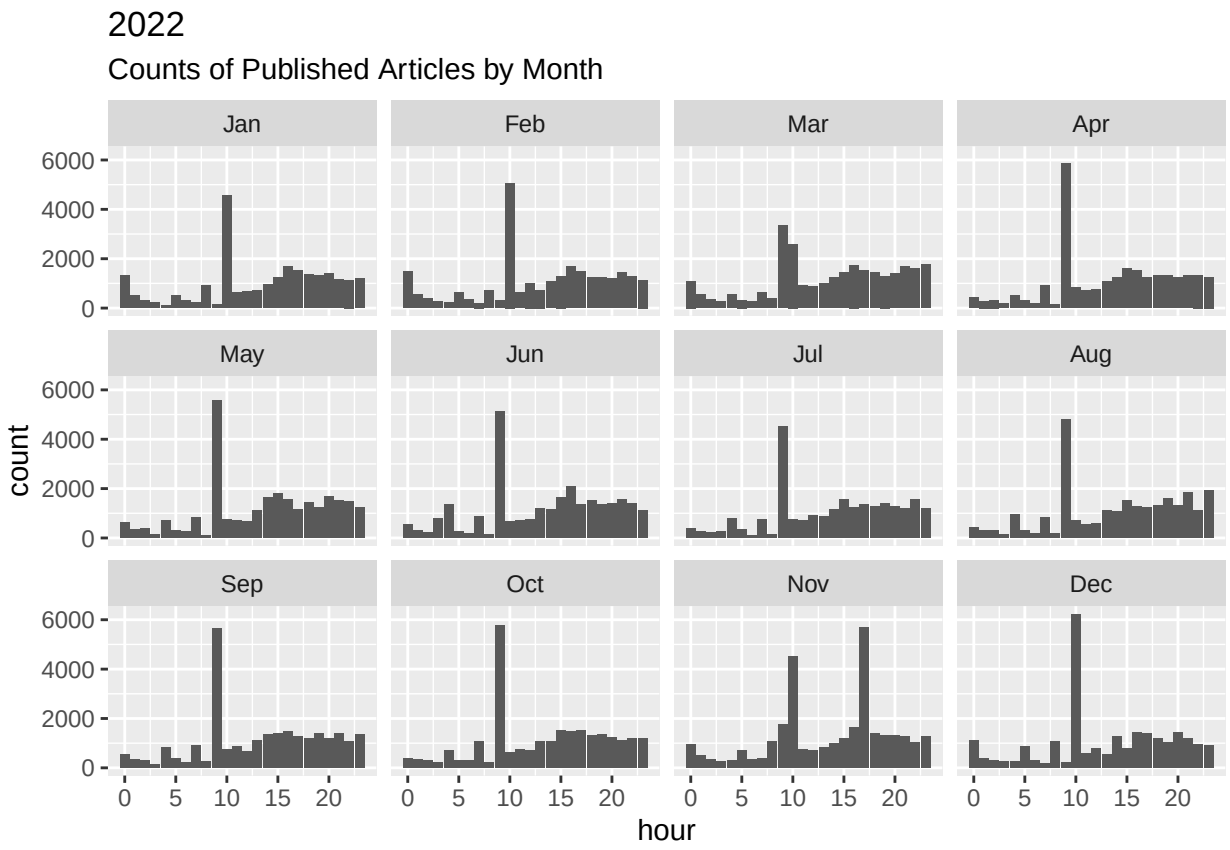
```
last3_month_df_unnest <-
  last3_month_df |>
  select(response_docs_pub_date, response_docs_section_name, response_docs_keywords) |>
  unnest(response_docs_keywords, keep_empty = TRUE, names_sep = '_') |>
  mutate(response_docs_pub_date = str_extract(response_docs_pub_date, "[:graph:]*(?=\\+)"))

last3_month_df_unnest$response_docs_pub_date <-
  last3_month_df_unnest$response_docs_pub_date |>
  ymd_hms()

last3_month_df_unnest <-
  last3_month_df_unnest |>
  mutate(hourly_date = floor_date(response_docs_pub_date, unit = 'hour'))
```

Highest Active Time for Published Articles

Overall in 2022, the most active time for articles to be published was around 0900 to 1000 by a wide margin. However, you can notice that in November, that changes with bi-modal peaks at 1000 and again at 1700. Given the strong nature of elections during the month of November in the United States, this could be a potential factor.

```
last3_month_df_unnest |>
  mutate(hour = hour(response_docs_pub_date), month = month(response_docs_pub_date, label = TRUE)) |>
  group_by(month, hour) |>
  summarize(count = n()) |>
  ggplot(aes(x = hour, y = count)) +
  geom_bar(stat = 'identity') +
```

```
  facet_wrap(vars(month)) +
  labs(title = '2022', subtitle = 'Counts of Published Articles by Month')
```




```
hour_table <-
  last3_month_df_unnest |>
  mutate(hour = hour(response_docs_pub_date), month = month(response_docs_pub_date, label = TRUE)) |>
  group_by(month, hour) |>
  summarize(count = n()) |>
  arrange(month, hour, desc(count)) |>
  group_by(month) |>
  mutate(rank = dense_rank(desc(count))) |>
  filter(rank %in% c(1)) |>
  select(month:count)

knitr::kable(hour_table, caption = "Table: Highest Active Hour for Publishing")
```

Table 1: Table: Highest Active Hour for Publishing

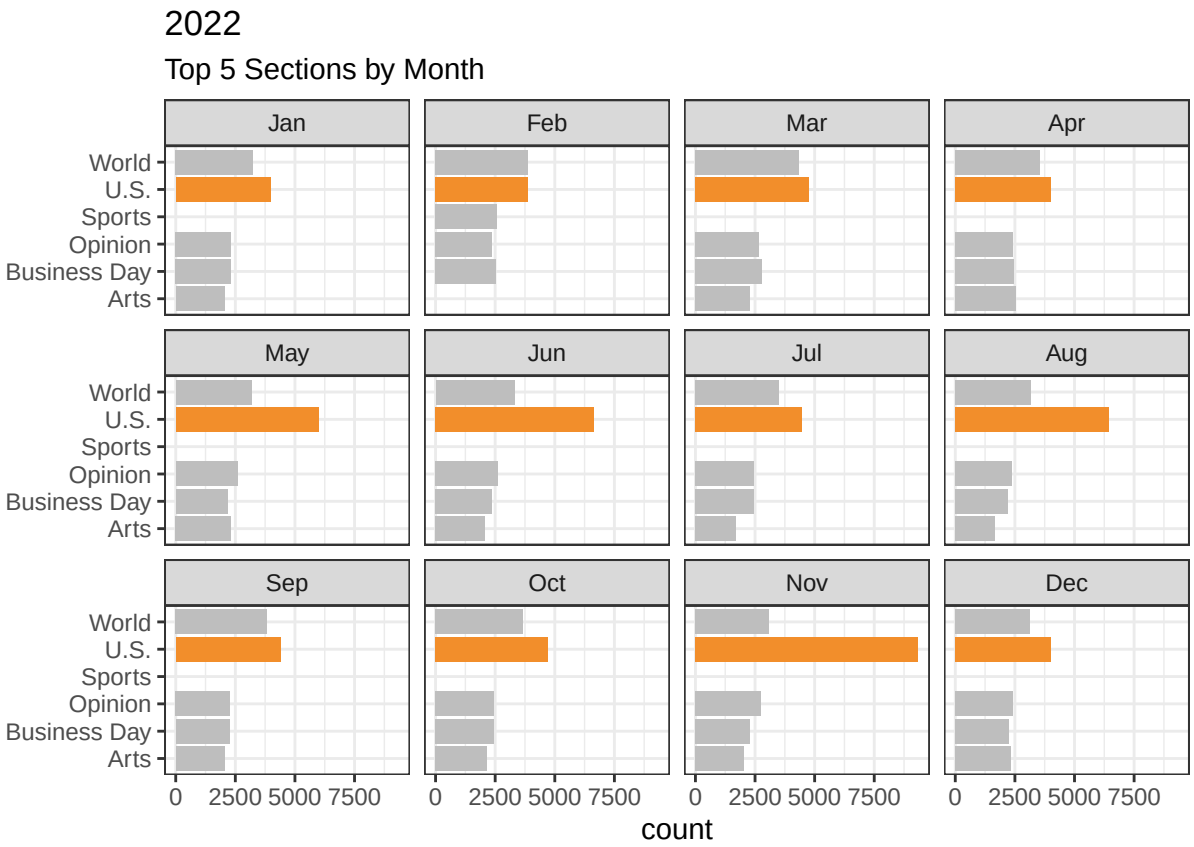
month	hour	count
Jan	10	4548
Feb	10	5069
Mar	9	3347
Apr	9	5850

month	hour	count
May	9	5589
Jun	9	5139
Jul	9	4523
Aug	9	4809
Sep	9	5637
Oct	9	5764
Nov	17	5704
Dec	10	6238

Top 5 Sections by Month

Looking further into which top topics are covered each month, we can use the sections the articles are in as parents. In each month, the U.S. section is ranked first each month in 2022. Again, we see November having a significantly larger amount of articles published for the U.S. section. Also, in February the Arts section gets dropped out of the top 5 and Sports takes its place. A potential reason can be that the NFL Super Bowl is played every February.

```
section_df <-
  last3_month_df_unnest |>
  mutate(month = month(response_docs_pub_date, label = TRUE), section = response_docs_section_name) |>
  group_by(month, section) |>
  summarise(count = n()) |>
  arrange(month, desc(count)) |>
  mutate(rank = dense_rank(desc(count))) |>
  filter(rank %in% c(1:5)) |>
  select(month, section, count, rank)

top_section <-
  section_df |>
  filter(rank == 1)

top_section_list <- top_section[['section']]

section_df <-
  section_df |> mutate(f_color = ifelse(section %in% top_section_list, "Yes", "No"))

section_df |>
  ggplot(aes(x = count, y = section, fill = f_color)) +
  geom_bar(stat = 'identity') +
  facet_wrap(vars(month)) +
  scale_fill_manual(values = c("Yes" = "#F28E2B", "No" = "gray")) +
  labs(title = '2022',
       subtitle = 'Top 5 Sections by Month',
        y = '') +
  theme_bw() +
  theme(legend.position = "none")
```




```
section_tbl <-
  section_df |>
  select(month, section, count) |>
  pivot_wider(names_from = month, values_from = count)

knitr::kable(section_tbl, caption = "Table: Top 5 Sections by Month")
```

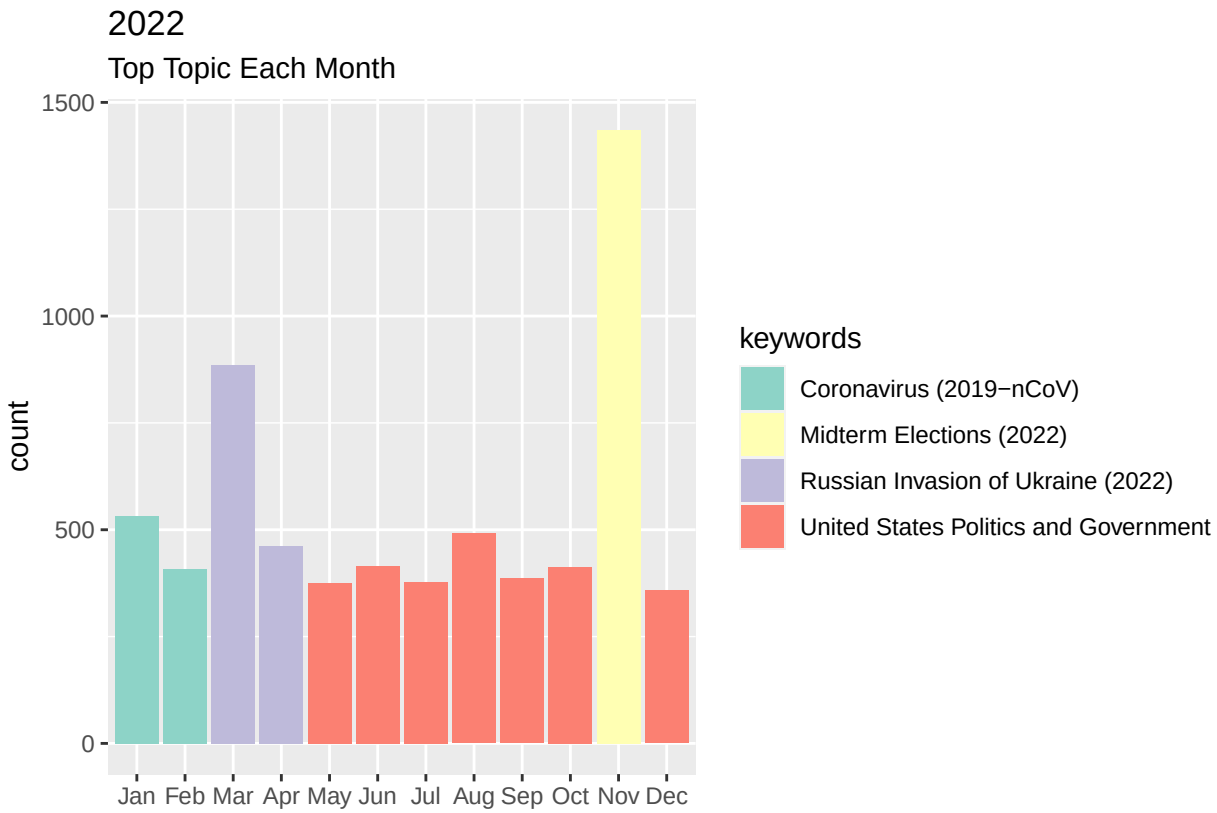
Table 2: Table: Top 5 Sections by Month

section	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
U.S.	3974	3868	4755	4019	5981	6645	4481	6443	4396	4714	9351	4002
World	3240	3864	4326	3561	3174	3308	3510	3174	3821	3683	3082	3118
Opinion	2321	2373	2666	2395	2598	2629	2459	2385	2284	2450	2750	2429
Business Day	2318	2519	2786	2445	2170	2350	2444	2185	2256	2438	2276	2252
Arts	2042	NA	2288	2534	2318	2066	1679	1666	2039	2153	2029	2308
Sports	NA	2589	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

Top Topic Each Month

In 2022, the year began with top articles covering COVID-19, then transitioning into the Russian Invasion of Ukraine. After the first third of the year, the shift into United States Politics and Government is mainly published, and as hypothesized, the US Midterm Elections is the primary topic in November.

```r
keywords_df <-
  last3_month_df_unnest |>
  filter(!is.na(response_docs_keywords_value)) |>
  mutate(month = month(response_docs_pub_date, label = TRUE), keywords = response_docs_keywords_value) |>
  group_by(month, keywords) |>
  summarise(count = n()) |>
  arrange(month, desc(count)) |>
  mutate(rank = dense_rank(desc(count))) |>
  filter(rank %in% c(1)) |>
  select(month, keywords, count)

keywords_df |>
  ggplot(aes(x = month, y = count, fill = keywords)) +
  geom_bar(stat = 'identity') +
  labs(title = '2022',
       subtitle = 'Top Topic Each Month',
       x = '') +
  scale_fill_brewer(palette = 'Set3')
```

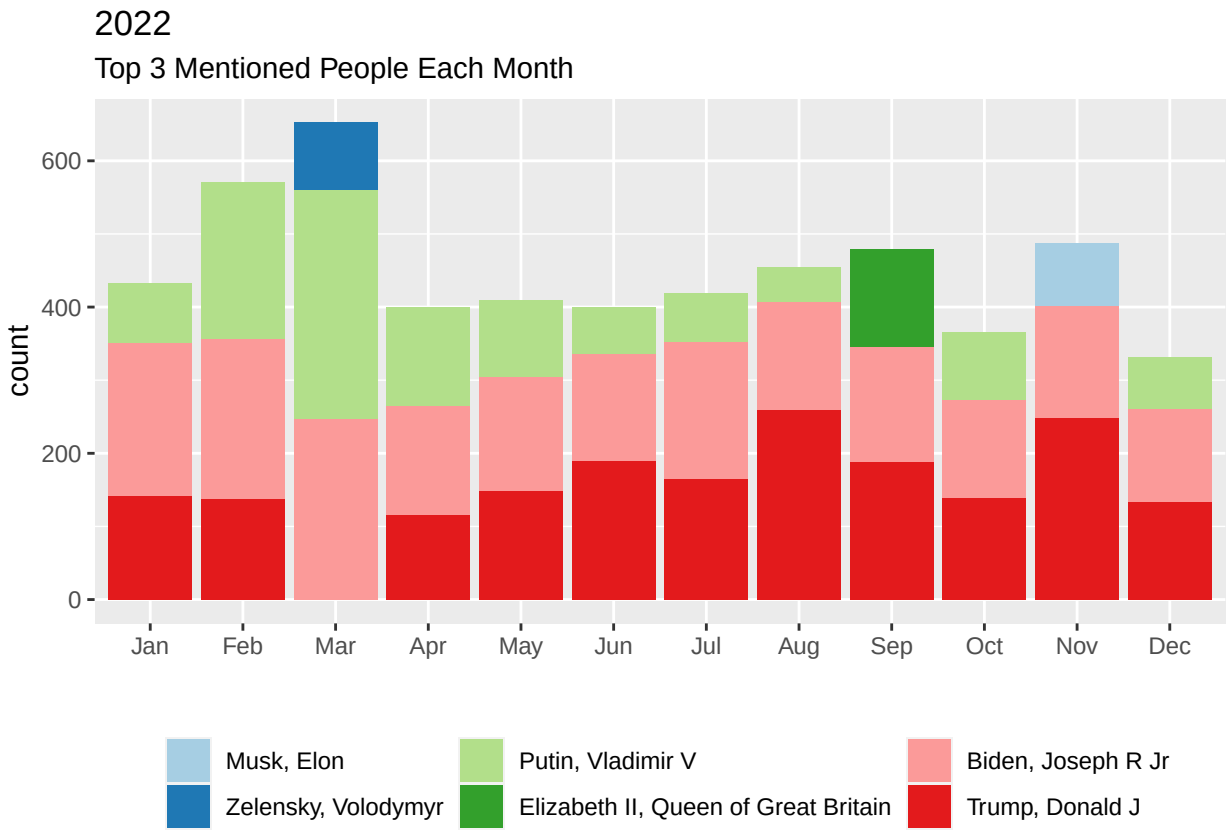


Top Person Mentioned in Article

The top mentioned person in the articles is Donald Trump followed by Joseph Biden. We can see cases in September when the Queen of England passed away and enters the top 3 mentioned along with Elon Musk

in November with the mass layoffs at Twitter.

```
person_df <-
  last3_month_df_unnest |>
  filter(!is.na(response_docs_keywords_value), response_docs_keywords_name == 'persons') |>
  mutate(month = month(response_docs_pub_date, label = TRUE), keywords = response_docs_keywords_value)
  group_by(month, keywords) |>
  summarise(count = n()) |>
  arrange(month, desc(count)) |>
  mutate(rank = dense_rank(desc(count))) |>
  filter(rank %in% c(1:3)) |>
  select(month, keywords, count, rank)

person_df |>
  ggplot(aes(x = month, y = count, fill = reorder(keywords, count))) +
  geom_bar(stat = 'identity') +
  labs(title = '2022',
       subtitle = 'Top 3 Mentioned People Each Month',
       x = '',
       fill = '') +
  theme(legend.position = 'bottom') +
  scale_fill_brewer(palette = 'Paired')
```



Conclusion

The NY Times API provides a lot of data pertaining to its published articles. It also includes things such as abstract descriptions of the article and metadata of images attached to it. Further analysis could be done with NLP and sentiment analysis or determining which articles uses larger images and screen real estate based on its topic.
