Data 607 - Assignment 6 - Working with APIs

William Jasmine

2022-10-25

Introduction

This document outlines the process of retrieving data from an API so that it can be analyzed in R. In this case, we will be using the New York Times APIs as our example. More specifically, we will be pulling data from NYT's Times Newswire API, which "provides an up-to-the-minute stream of published articles." The steps outlined below will utilize the documentation provided for this specific API in order to connect and pull data.

Loading Data From API

In order to first use the NYT API, you will need to create an app with an authorization key that can be used to access their data. The steps for doing this are outlined here.

When running this `.Rmd` file, you should be prompted to enter this key so that it can be used to authenticate the api calls (make sure to run using the `Knit With Parameters...` option). You can also enter the into the file manually below by un-commenting the line mentioned in the code block below. Either way, this key will be stored as the variable `api_key`. The documentation for the specific API that is used provides the required text for the other two variables defined below, `api_url` and `num_results`. The `api_url` is merely the URL location where we will be pulling the data, which can be found once you have determined the NYT API you wish to use (listed here). The `num_results` will tell the request the number of rows that we want to return (can be any multiple of 20 up to 500).

```
# Enter NYT API Key'
api_key <- params$key
# Un-comment below line for manual entry.
#api_key <- 'dummy-api-key'

# Enter NYT API web location
api_url <- 'https://api.nytimes.com/svc/news/v3/content/all/all.json?'

# Number of results
num_results <- 500
```

The code chunk below combines the three variables above into a single API call via the instructions provided in the API documentation. It then uses the `fromJSON` function to actually carry out this API call, and the results are transformed into a R dataframe, `df`.

```
api_call <- paste(api_url, "limit=", num_results, "&api-key=", api_key, sep='')
df <- fromJSON(api_call, flatten = TRUE) %>% data.frame
glimpse(df)
```

```
## Rows: 500
```

```
## Columns: 27
## $ status                       <chr> "OK", "OK", "OK", "OK", "OK", "OK", "OK",~
## $ copyright                    <chr> "Copyright (c) 2022 The New York Times Co~
## $ num_results                  <int> 500, 500, 500, 500, 500, 500, 500, 500, 5~
## $ results.slug_name            <chr> "25pol-padebate-takeaways", "midterm-blog~
## $ results.section              <chr> "U.S.", "U.S.", "U.S.", "Today's Paper", ~
## $ results.subsection           <chr> "Politics", "Politics", "Politics", "", "~
## $ results.title                <chr> "5 Takeaways From the Pennsylvania Senate~
## $ results.abstract             <chr> "Lt. Gov. John Fetterman and Mehmet Oz tr~
## $ results.uri                  <chr> "nyt://article/69d757ba-dd74-5900-b0d2-f1~
## $ results.url                  <chr> "https://www.nytimes.com/2022/10/25/us/po~
## $ results.byline               <chr> "BY KATIE GLUECK AND TRIP GABRIEL", "BY C~
## $ results.item_type            <chr> "Article", "Article", "Article", "Article~
## $ results.source               <chr> "New York Times", "New York Times", "New ~
## $ results.updated_date         <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.created_date         <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.published_date       <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.first_published_date <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.material_type_facet  <chr> "News", "Live Blog Post", "News", "Quote"~
## $ results.kicker               <chr> "", "", "", "", "", "", "", "wordplay, th~
## $ results.subheadline          <chr> "", "", "", "", "", "", "", "", "", "", "~
## $ results.des_facet            <list> <"Midterm Elections (2022)", "United Sta~
## $ results.org_facet            <list> <"Democratic Party", "Republican Party",~
## $ results.per_facet            <list> <"Fetterman, John (1969- )", "Oz, Mehmet~
## $ results.geo_facet            <list> "Pennsylvania", <NULL>, "Pennsylvania", ~
## $ results.related_urls         <list> <NULL>, <NULL>, <NULL>, <NULL>, <NULL>, ~
## $ results.multimedia           <list> <NULL>, [<data.frame[4 x 8]>], [<data.fr~
## $ results.thumbnail_standard   <chr> NA, "https://static01.nyt.com/images/2022~
```

The results shown above show that our API call has produced the desired results.

Data Cleaning

Now that the data has been loaded, we can move forward with a number of data cleaning steps. First, the actual fields we need in this case are only those with names preceded by the `results.` prefix, as this was the actual JSON element that contained the list of results. The `status`, `copyright`, and `num_results` fields were also their own JSON elements, but contain repeated information about the API call that can now be removed. This is done in the cell below:

```
df <- df %>%
  select(-status, -copyright, -num_results)
glimpse(df)
```

```
## Rows: 500
## Columns: 24
## $ results.slug_name            <chr> "25pol-padebate-takeaways", "midterm-blog~
## $ results.section              <chr> "U.S.", "U.S.", "U.S.", "Today's Paper", ~
## $ results.subsection           <chr> "Politics", "Politics", "Politics", "", "~
## $ results.title                <chr> "5 Takeaways From the Pennsylvania Senate~
## $ results.abstract             <chr> "Lt. Gov. John Fetterman and Mehmet Oz tr~
## $ results.uri                  <chr> "nyt://article/69d757ba-dd74-5900-b0d2-f1~
## $ results.url                  <chr> "https://www.nytimes.com/2022/10/25/us/po~
## $ results.byline               <chr> "BY KATIE GLUECK AND TRIP GABRIEL", "BY C~
## $ results.item_type            <chr> "Article", "Article", "Article", "Article~
```

```
## $ results.source               <chr> "New York Times", "New York Times", "New ~
## $ results.updated_date         <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.created_date         <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.published_date       <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.first_published_date <chr> "2022-10-25T23:41:42-04:00", "2022-10-25T~
## $ results.material_type_facet  <chr> "News", "Live Blog Post", "News", "Quote"~
## $ results.kicker               <chr> "", "", "", "", "", "", "", "wordplay, th~
## $ results.subheadline          <chr> "", "", "", "", "", "", "", "", "", "", "~
## $ results.des_facet            <list> <"Midterm Elections (2022)", "United Sta~
## $ results.org_facet            <list> <"Democratic Party", "Republican Party",~
## $ results.per_facet            <list> <"Fetterman, John (1969- )", "Oz, Mehmet~
## $ results.geo_facet            <list> "Pennsylvania", <NULL>, "Pennsylvania", ~
## $ results.related_urls         <list> <NULL>, <NULL>, <NULL>, <NULL>, <NULL>, ~
## $ results.multimedia           <list> <NULL>, [<data.frame[4 x 8]>], [<data.fr~
## $ results.thumbnail_standard   <chr> NA, "https://static01.nyt.com/images/2022~
```

Next, the cell below cleans the column names to remove the `results.` prefix from each column name:

```
colnames(df) <- str_replace_all(colnames(df), 'results.', '')
colnames(df)
```

```
##  [1] "slug_name"            "section"              "subsection"
##  [4] "title"                "abstract"             "uri"
##  [7] "url"                  "byline"               "item_type"
## [10] "source"               "updated_date"         "created_date"
## [13] "published_date"       "first_published_date" "material_type_facet"
## [16] "kicker"               "subheadline"          "des_facet"
## [19] "org_facet"            "per_facet"            "geo_facet"
## [22] "related_urls"         "multimedia"           "thumbnail_standard"
```

As is clear from the output above, all column names have been successfully cleaned.

Lastly, looking at the data, we can see that the elements in the `byline` column are always preceded by the string `"BY "`. To fix this we replace this column with an `authors` column that removes this extra string:

```
df <- df %>%
  mutate(author = str_extract(byline, '(?<=BY ).*'))

df <- df %>%
  select(-byline)

head(df$author)
```

```
## [1] "KATIE GLUECK AND TRIP GABRIEL" "CARL HULSE"
## [3] "SHANE GOLDMACHER"              NA
## [5] "JESSE GREEN"                   "NEWSNATION"
```

Data Analysis

The result of running the code in the above sections is that we now have a clean set of data containing information about the past 500 most recently published New York Times articles. Thus, we can now use this data to answer some interesting questions. (Note that this `.Rmd` file was first run on October 25th, 2022, and because this data is constantly being updated the following commentary may not match the results seen if the code is rerun. However, the code itself should still be able to answer the questions that follow).

Which section has been published the most?

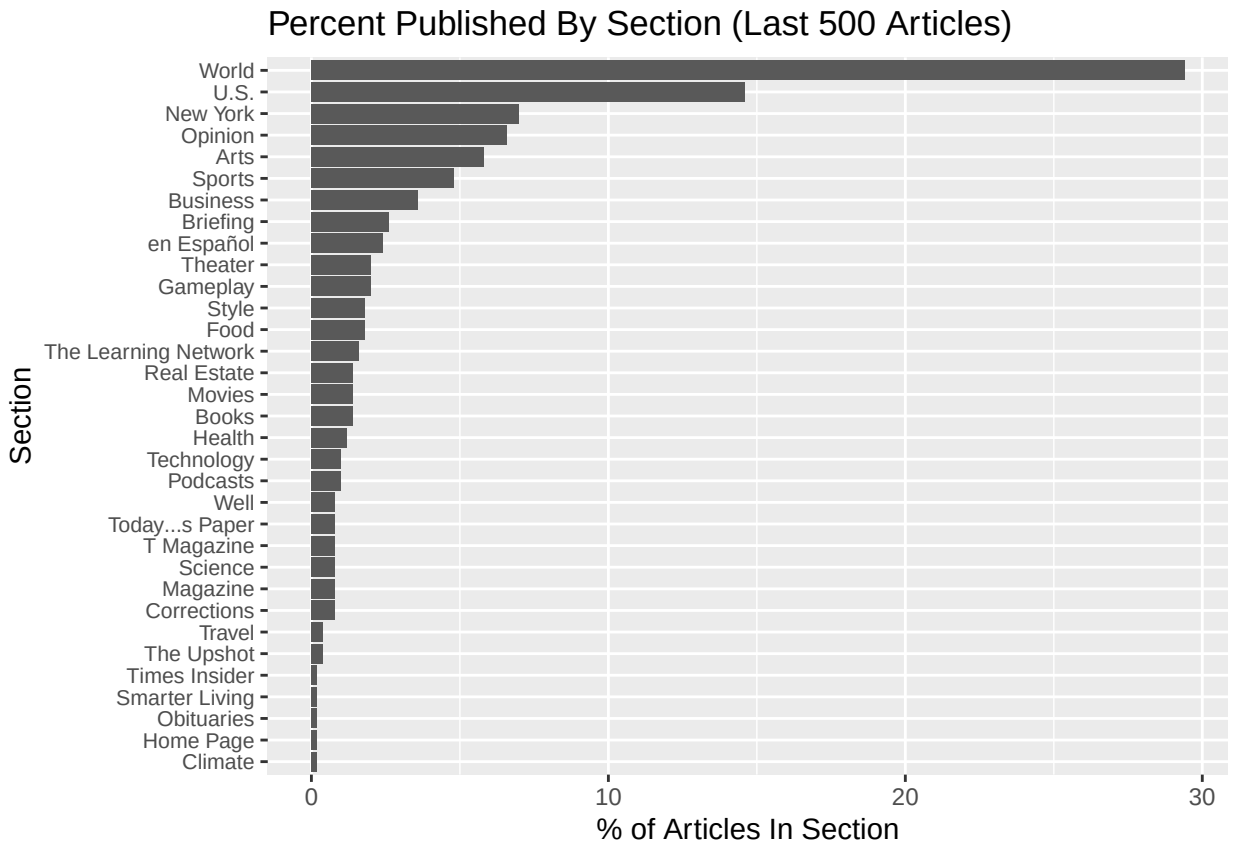
The following code breaks up the most recently published articles by section, and presents the information as a bar chart:

```
plt_df <- df %>%
  group_by(section) %>%
    summarise(
      count_articles = n(),
      pct_articles = (n() / nrow(df)) * 100
    ) %>%
      arrange(desc(count_articles))

ggplot(data=plt_df,aes(x=reorder(section, pct_articles), y=pct_articles)) + geom_bar(stat='identity') +
  coord_flip() +
  labs(
    x = 'Section',
    y = '% of Articles In Section',
    title = 'Percent Published By Section (Last 500 Articles)'
  ) +
  theme(axis.text.y = element_text(size = 8))
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today's Paper' in 'mbcsToSbcs': dot substituted for <e2>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <80>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Today’s Paper' in 'mbcsToSbcs': dot substituted for <99>
```



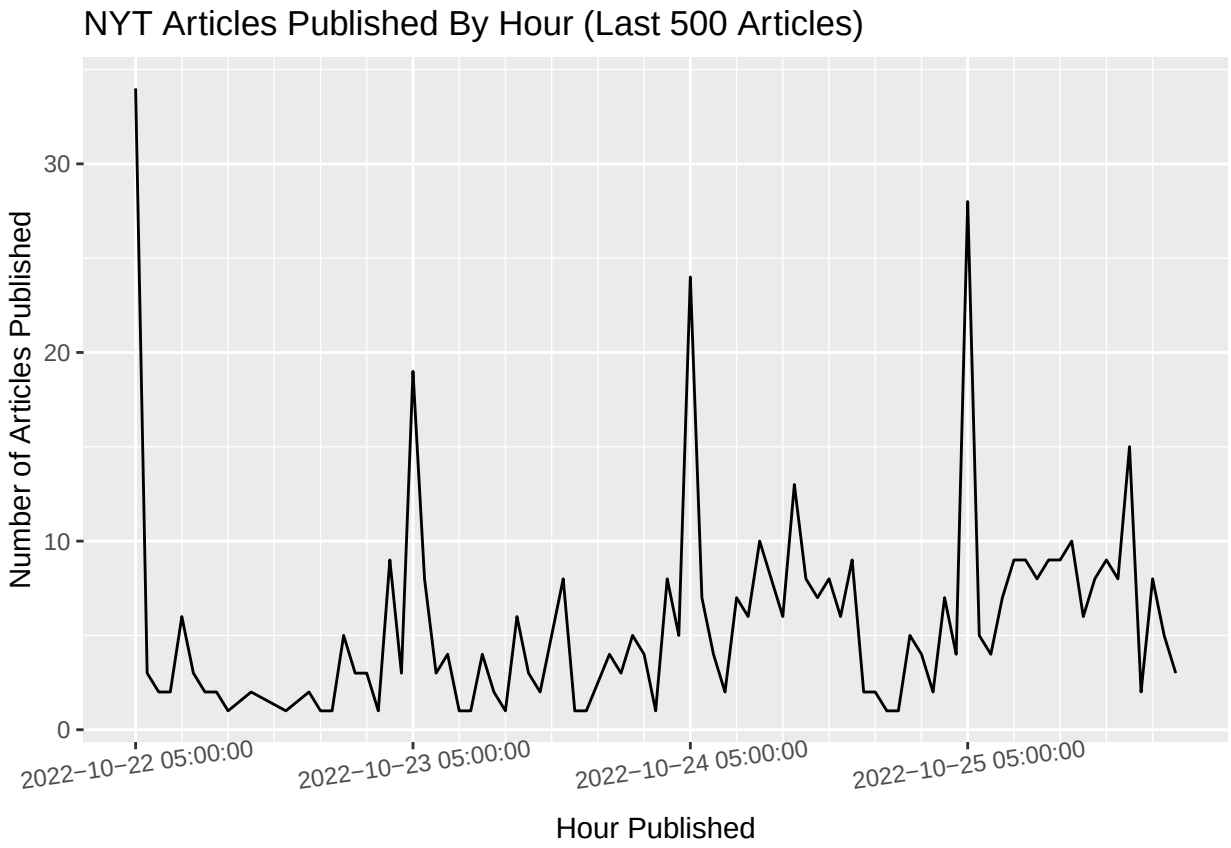
As is clear in the plot above, the most commonly published to section is `World`.

Does the # of articles published change over time?

To answer this question, the cell below determines the hour each article was first published and stores that information in the `first_published_hour` column. The data is then grouped by that column to make a plot of the number of articles published over time:

```
df <- df %>%
  mutate(first_published_hour =
    anytime(
      x = paste(str_extract(first_published_date, '^[^:]*'), ':00:00',sep=''),
      tz = Sys.timezone()
    )
  )

plt_data <- df %>%
  group_by(first_published_hour) %>%
    summarise(count_articles = n()) %>%
      arrange(desc(first_published_hour))

x <- df$first_published_hour

ggplot(data = plt_data, aes(x=first_published_hour, y=count_articles)) +
  geom_line() +
  scale_x_datetime(
```

```
  breaks = seq.POSIXt(from = min(x), to = max(x), by = "1 days"),
  minor_breaks = seq.POSIXt(from = min(x), to = max(x), by = "4 hours")
) +
theme(axis.text.x = element_text(angle=8)) +
labs(
  x = 'Hour Published',
  y = 'Number of Articles Published',
  title = 'NYT Articles Published By Hour (Last 500 Articles)'
)
```



As is to be expected, the majority of articles seem to be published in the 5:00 PM to 6:00 PM hour likely right before everyone leaves for the day.

Conclusion

This document used New York Times publicly available data to demonstrate the process of how one could use an API to load data and answer some interesting research questions. Though there will be differences in how to interact with APIs other than the ones provided by NYT, the general process flow described and carried out above should remain relatively similar across other APIs.