NC STATE UNIVERSITY

Connecting to an API and Making a Word Cloud

Justin Post

Goals

- Understand very basics of APIs
- Contact an API using R
- Process returned data
- Create a word cloud

Repo with HTML slides & code

APIs

Application Programming Interfaces (APIs) - a defined method for asking for information from a computer

- List of APIs
- Important for grabbing data, often returned in JSON format
- Expand opportunities by allowing your students to get data they are interested in!
- Very few packages for contacting APIs are out there for R (as compared to say python)
- Can do it yourself using `httr` package!

Steps to obtain data

- Install packages needed for contacting APIs and handling data
  - `httr` and `jsonlite`
- (Usually) Obtain a key by registering at the API you want to contact
- Construct a **URL** to obtain data (using `GET()`)
- Process data using `jsonlite` functions

Example https://newsapi.org/

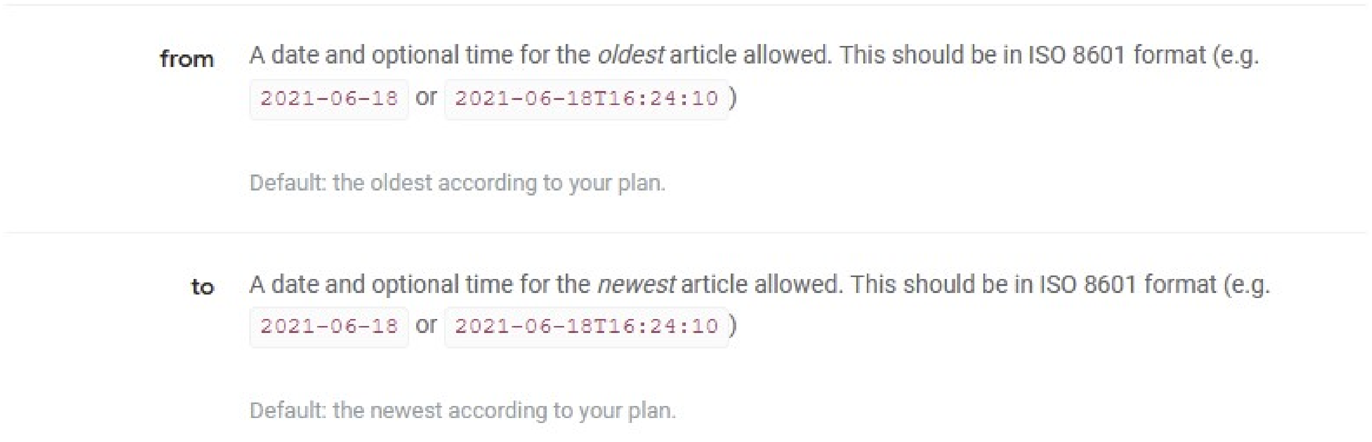
Registered for a key at newsapi.org. An API for looking at news articles

- Look at documentation for API (most have this!)
- Example URL to obtain data is given

```
https://newsapi.org/v2/everything?q=bitcoin&apiKey=myKeyGoesHere
```

Example https://newsapi.org/

- Can add in date for instance:



from A date and optional time for the *oldest* article allowed. This should be in ISO 8601 format (e.g. `2021-06-18` or `2021-06-18T16:24:10`)

Default: the oldest according to your plan.

to A date and optional time for the *newest* article allowed. This should be in ISO 8601 format (e.g. `2021-06-18` or `2021-06-18T16:24:10`)

Default: the newest according to your plan.

```
https://newsapi.org/v2/everything?q=bitcoin&from=2021-06-01&
apiKey=myKeyGoesHere
```

Using R to Obtain the Data

- Use `GET` from `httr` package (make sure to load package!)
- Modify for what you have interest in!

```
library(httr)
GET("http://newsapi.org/v2/everything?qInTitle=Juneteenth&from=2021-06-01&language=en&
    apiKey=myKeyGoesHere")
```

Returned data

- Usually what you want is stored in something like `content`

```
str(myData, max.level = 1)
```

```
## List of 10
##  $ url        : chr "http://newsapi.org/v2/everything?qInTitle=Juneteenth&from=2021-06-01&la
##  $ status_code: int 200
##  $ headers    :List of 17
##   ..- attr(*, "class")= chr [1:2] "insensitive" "list"
##  $ all_headers:List of 1
##  $ cookies    :'data.frame': 0 obs. of  7 variables:
##  $ content    : raw [1:84391] 7b 22 73 74 ...
##  $ date       : POSIXct[1:1], format: "2021-06-18 17:56:44"
##  $ times      : Named num [1:6] 0 0.00391 0.02254 0.02266 0.06164 ...
##   ..- attr(*, "names")= chr [1:6] "redirect" "namelookup" "connect" "pretransfer" ...
##  $ request    :List of 7
##   ..- attr(*, "class")= chr "request"
##  $ handle     :Class 'curl_handle' <externalptr>
##  - attr(*, "class")= chr "response"
```

Parse with `jsonlite`

Common steps:

- Grab the list element we want
- Convert it to characters (it will have a JSON structure)
- Convert it to a data frame with `fromJSON` from the `jsonlite` package

```
library(dplyr)
library(jsonlite)
parsed <- myData$content %>% rawToChar() %>% fromJSON()
str(parsed, max.level = 1)
```

```
## List of 3
##  $ status      : chr "ok"
##  $ totalResults: int 1083
##  $ articles    :'data.frame':    100 obs. of  8 variables:
```

Inspecting article info

```r
parsed$articles %>%
  select(author, source, title, description, everything())
```

```
##                                             author          source.id
## 1                                     Trish Bendix               <NA>
## 2               Amelia Nierenberg and David Poller               <NA>
## 3                                   Tariro Mzezewa               <NA>
## 4                            Isabella GrullÃ³n Paz               <NA>
## 5                                  Luke Broadwater               <NA>
## 6                  Annie Karni and Luke Broadwater               <NA>
## 7                                    Laura Zornosa               <NA>
## 8                                    Alyssa Lukpat               <NA>
## 9                 https://www.facebook.com/bbcnews           bbc-news
## 10       Dana Rubinstein and Luis FerrÃ©-SadurnÃ               <NA>
## 11                                    Kenny Herzog               <NA>
## 12                                         Reuters            reuters
## 13                                         Reuters            reuters
## 14                                         Reuters            reuters
## 15                                   Reuters Staff            reuters
## 16                                         Reuters            reuters
## 17                                    Makini Brice            reuters
## 18                                   Reuters Staff            reuters
## 19                  W. James Antle III, Columnist               <NA>
## 20                               Jason Weisberger               <NA>
```

Building a word cloud

Use data from titles and visualize in a word cloud

- Must ‘tokenize’ the titles and remove ‘stop words’ like “the” or “a”
- `dplyr` and `tidytext` packages makes this very easy!
- `unnest_tokens()` tokenizes the titles and creates a data frame
- A `stop_words` object is available with common words to remove (sometimes you need to add to this)
- `anti_join()` the data frame from `unnest_tokens()` and the `stop_words` data frame
- Sum up the number of times each word appears for weighting in the word cloud (`count()` works well!)

Making data for word cloud

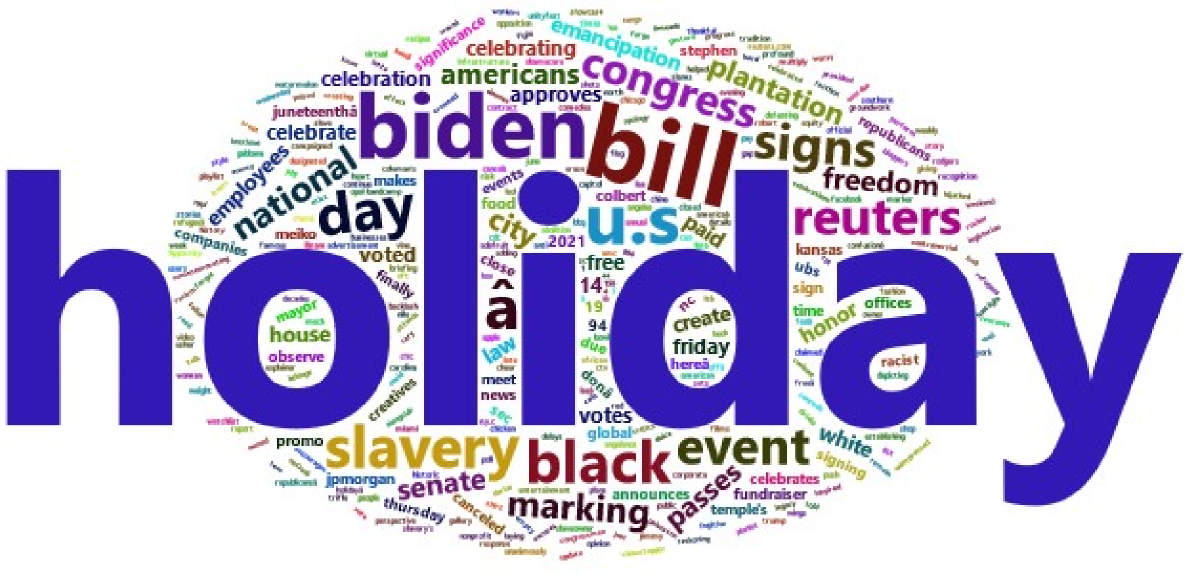
```
library(dplyr); library(tidytext)
wcData <- parsed$articles$title %>%
  as_tibble() %>%
  unnest_tokens(word, value) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE)
wcData
```

```
## # A tibble: 326 x 2
##    word           n
##    <chr>      <int>
##  1 juneteenth    99
##  2 holiday       47
##  3 federal       28
##  4 bill          14
##  5 biden         12
##  6 u.s           10
##  7 â              9
##  8 black          8
##  9 day            8
## 10 slavery        8
## # ... with 316 more rows
```

Word cloud via `wordcloud2` package

`wordcloud2` package easily creates nice looking word clouds

```
library(wordcloud2); wordcloud2(wcData[-1,])
```



Goals

- Understand very basics of APIs
- Contact an API using R
  - `httr:GET("URL")`
- Process returned data
  - Often JSON data: `jsonlite` package
  - Tokenize and remove stop words with `tidytext`
- Create a word cloud
  - `wordcloud2` package

Repo with HTML slides & code - Download the APIWordCloud.html file and open in a web browser