TAD Week 4 Assignment

Tommy Klein

Working Directory

```
setwd('/Users/tklein/Desktop/Desktop_tpk/JHU_Classes/text_as_data/week4')
```

Libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.6
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(quanteda)
```

```
## Package version: 3.2.0
## Unicode version: 13.0
## ICU version: 69.1

## Parallel computing: 4 of 4 threads used.

## See https://quanteda.io for tutorials and examples.
```

```
library(readtext)
library(jsonlite)
```

```
## Warning: package 'jsonlite' was built under R version 4.1.2

##
## Attaching package: 'jsonlite'

## The following object is masked from 'package:purrr':
##
##     flatten
```

```
library(nytimes)
library(httr)
```

Functions

Loading some functions I wrote that I might use - I'll show the functions at the end of this document in the appendix.

```
source('../functions/helper_functions.R')
```

Research Question

The New York Times is the paper of record in America, and because of this its content can be viewed as a microcosm of American thought and opinions. We can use the New York Times data to get an understanding of what topics Americans care about, what specific things they care about regarding that topic, and the sentiment towards that topic. For example, you could query the New York Times articles to see what articles had been written about, or associated with, the city you live in. You could then analyze the text to understand any common themes are sentiment in the articles, which would give you an idea of how the rest of America views your city. This data could then be used by city leaders to inform city development and branding, to ensure the city is successful in the future.This method is very interesting because unlike conducting a survey we can easily do this analysis for historical time periods, it is easily reproducible, and it is cheap.

Collecting Article Data

First, I will read in my api key which I saved in a file in another folder.

```
my_nyt_key <- read.csv('../resources/nyt_api_key', header = F, stringsAsFactors = F)[[1]]
```

Now I will search for articles that are tagged as being about or from Portland, Oregon.

```
portland_articles <- nyt_search(
  'portland oregon',
  n = 100,
  end_date = '20220101',
  apikey = my_nyt_key
)


nyt_df = as.data.frame(portland_articles)

glimpse(nyt_df)
```

```
## Rows: 100
## Columns: 19
## $ id               <chr> "nyt://article/afd7d5d2-a13f-5fe8-8d8a-b03c6e150ef7",~
## $ abstract         <chr> "HOLLYWOOD, Dec. 30 (AP)--Tim Hardin, the singer-song~
## $ byline           <chr> NA, "Special to The New York Times", NA, "ADA LOUISE ~
## $ document_type    <chr> "article", "article", "article", "article", "article"~
## $ headline         <chr> "Tim Hardin, Songwriter, 39, Dies; A Romantic Compose~
## $ keywords         <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ lead_paragraph   <chr> "", "", "", "", "", "", "", "", "", "", "", "", "", "~
## $ multimedia       <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ news_desk        <chr> "None", "None", "None", "None", "None", "None", "None~
## $ print_page       <chr> "6", "12", "8", "25", "1", "17", "8", "2", "7", "84",~
## $ print_section    <chr> "M", "A", "S", "A", "S", "A", "M", "S", "S", "F", "M"~
## $ pub_date         <dttm> 1980-12-31 05:00:00, 1980-12-29 05:00:00, 1980-12-29~
## $ section_name     <chr> "Archives", "Archives", "Archives", "Archives", "Arch~
## $ snippet          <chr> "", "", "", "", "", "", "", "", "", "", "", "", "", "~
## $ source           <chr> "The New York Times", "The New York Times", "The New ~
## $ type_of_material <chr> "Archives", "Archives", "Archives", "Archives", "Arch~
## $ uri              <chr> "nyt://article/afd7d5d2-a13f-5fe8-8d8a-b03c6e150ef7",~
## $ web_url          <chr> "https://www.nytimes.com/1980/12/31/archives/tim-hard~
## $ word_count       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
```

```
## have to wait for about 6 seconds
## to not run afoul of NYT API call limits

Sys.sleep(30)

portland_articles_pre2000 <- nyt_search(
  'portland oregon',
  n = 100,
  end_date = '19991231',
  apikey = my_nyt_key
)


nyt_df_pre2000 = as.data.frame(portland_articles_pre2000)

glimpse(nyt_df_pre2000)
```

```
## Rows: 100
## Columns: 19
## $ id               <chr> "nyt://article/afd7d5d2-a13f-5fe8-8d8a-b03c6e150ef7",~
## $ abstract         <chr> "HOLLYWOOD, Dec. 30 (AP)--Tim Hardin, the singer-song~
## $ byline           <chr> NA, "Special to The New York Times", NA, "ADA LOUISE ~
## $ document_type    <chr> "article", "article", "article", "article", "article"~
## $ headline         <chr> "Tim Hardin, Songwriter, 39, Dies; A Romantic Compose~
## $ keywords         <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ lead_paragraph   <chr> "", "", "", "", "", "", "", "", "", "", "", "", "", "~
## $ multimedia       <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ news_desk        <chr> "None", "None", "None", "None", "None", "None", "None~
## $ print_page       <chr> "6", "12", "8", "25", "1", "17", "8", "2", "7", "84",~
## $ print_section    <chr> "M", "A", "S", "A", "S", "A", "M", "S", "S", "F", "M"~
## $ pub_date         <dttm> 1980-12-31 05:00:00, 1980-12-29 05:00:00, 1980-12-29~
## $ section_name     <chr> "Archives", "Archives", "Archives", "Archives", "Arch~
## $ snippet          <chr> "", "", "", "", "", "", "", "", "", "", "", "", "", "~
## $ source           <chr> "The New York Times", "The New York Times", "The New ~
## $ type_of_material <chr> "Archives", "Archives", "Archives", "Archives", "Arch~
## $ uri              <chr> "nyt://article/afd7d5d2-a13f-5fe8-8d8a-b03c6e150ef7",~
## $ web_url          <chr> "https://www.nytimes.com/1980/12/31/archives/tim-hard~
## $ word_count       <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
```

It looks like in both queries I returned 100 results. That makes sense, as that is the max number of results (specified by `n = 100`).

```
nyt_df$headline[1:20]
```

```
##  [1] "Tim Hardin, Songwriter, 39, Dies; A Romantic Composer 'A Legal Loophole'"
##  [2] "Bill Is Expected to Benefit Pacific Coast Fishermen; Protection of Salmon Behind on Payments O
##  [3] "All-Oregon Final in Far West; Tenn. 69, Ariz. State 53 Duke 77, New Orleans 63 Cornell 63, Hof
##  [4] "ARCHITECTURE VIEW; The Boom in Bigness Goes On ARCHITECTURE VIEW"
##  [5] "The Pros Aren't Passing on Lomax; The Pros Aren't Passing On Portland State's Lomax "
##  [6] "Oregon Stores Seek Restitution From Shoplifters; One Man Files Suit "
##  [7] "Oregon Man Gets 20-Year Term In $250,000 Gem Extortion Plot"
##  [8] "Sports World Specials; Monday Night Live Casey and the Wolf Hall of Records Aerial Game Poll P
##  [9] "U.C.L.A., in a 94-81 Victory, Impresses Losing Notre Dame Coach; Sanders and Foster Stand Out
## [10] "Christmas Catalogues: No Order's Too Tall; What Is Available A Genre of Catalogues Through the
## [11] "Tim Hardin, Songwriter, 39, Dies; A Romantic Composer 'A Legal Loophole'"
## [12] "Bill Is Expected to Benefit Pacific Coast Fishermen; Protection of Salmon Behind on Payments O
## [13] "All-Oregon Final in Far West; Tenn. 69, Ariz. State 53 Duke 77, New Orleans 63 Cornell 63, Hof
## [14] "ARCHITECTURE VIEW; The Boom in Bigness Goes On ARCHITECTURE VIEW"
## [15] "The Pros Aren't Passing on Lomax; The Pros Aren't Passing On Portland State's Lomax "
## [16] "Oregon Stores Seek Restitution From Shoplifters; One Man Files Suit "
## [17] "Oregon Man Gets 20-Year Term In $250,000 Gem Extortion Plot"
## [18] "Sports World Specials; Monday Night Live Casey and the Wolf Hall of Records Aerial Game Poll P
## [19] "U.C.L.A., in a 94-81 Victory, Impresses Losing Notre Dame Coach; Sanders and Foster Stand Out
## [20] "Christmas Catalogues: No Order's Too Tall; What Is Available A Genre of Catalogues Through the
```

The headlines don't really seem to be about Portland. Also, it looks like its just the same 10 articles repeated over again.

Semantics

```r
semantic_url <- paste0(
  'http://api.nytimes.com/svc/semantic/v2/concept/name/nytd_geo/Oregon?fields=all&api-key=',
  my_nyt_key
  )


r <- GET(semantic_url)

Sys.sleep(10)

res = content(r, 'parsed')

res$results[[1]]$geocodes
```

```
## [[1]]
## [[1]]$admin_code1
## [1] "OR"
##
## [[1]]$admin_code2
## NULL
##
## [[1]]$admin_code3
## NULL
##
## [[1]]$admin_code4
## NULL
##
## [[1]]$admin_name1
## [1] "Oregon"
##
## [[1]]$admin_name2
## NULL
```

```
##
## [[1]]$admin_name3
## NULL
##
## [[1]]$admin_name4
## NULL
##
## [[1]]$concept_id
## [1] 27820
##
## [[1]]$concept_name
## [1] "Oregon"
##
## [[1]]$concept_status
## [1] "Active"
##
## [[1]]$concepts_geocodes_created
## [1] "\"2013-02-25 15:10:12-05:00\""
##
## [[1]]$concepts_geocodes_updated
## [1] "\"2013-02-25 15:10:12-05:00\""
##
## [[1]]$country_code
## [1] "US"
##
## [[1]]$country_name
## [1] "United States"
##
## [[1]]$dst_offset
## [1] "\"1969-12-31 18:59:53-05:00\""
##
## [[1]]$elevation
## [1] 1460
##
## [[1]]$feature_class
## [1] "A"
##
## [[1]]$feature_code
## [1] "ADM1"
##
## [[1]]$feature_code_name
## [1] "first-order administrative division"
##
## [[1]]$geocode_id
## [1] 416
##
## [[1]]$geoname_id
## [1] 5744337
##
## [[1]]$gmt_offset
## [1] "\"1969-12-31 18:59:52-05:00\""
##
## [[1]]$is_times_tag
## [1] 1
```

```
##
## [[1]]$latitude
## [1] 44.00013
##
## [[1]]$longitude
## [1] -120.5014
##
## [[1]]$name
## [1] "Oregon"
##
## [[1]]$time_zone_id
## [1] "America/Los_Angeles"
```

The semantic API returns info about how the NYT defines the term, some information about the term, and some examples of how its used. For example, I searched for Oregon, and it returned articles where Oregon was mentioned, some geographic info about Oregon, and some links to further information about Oregon. This is different than the article API, which returns articles and some meta-data about those articles.

Books API

```r
book_url <- paste0(
  'https://api.nytimes.com/svc/books/v3/reviews.json?author=Celeste+Ng&api-key=',
  my_nyt_key
  )


book_api_results <- GET(book_url)

Sys.sleep(10)

book_results = content(book_api_results, 'parsed')



book_data <- tibble()
for(i in 1:length(book_results$results)){

  temp_data <- tibble(
    'title' = book_results$results[[i]]$book_title,
    'summary' = book_results$results[[i]]$summary
    )

  book_data <- bind_rows(book_data, temp_data)

}



book_corpus <- quanteda::corpus(book_data, docid_field = 'title', text_field = 'summary')


book_corpus
```

```
## Corpus consisting of 2 documents.
## Everything I Never Told You :
## "A tragedy tears away at a mixed-race family in 1970s Ohio."
##
## Little Fires Everywhere :
## "Celeste Ng's "Little Fires Everywhere" witnesses the mysteri..."
```

```
corp_to_dfm(book_corpus)
```

```
## Document-feature matrix of: 2 documents, 20 features (50.00% sparse) and 0 docvars.
##                                features
## docs                             tragedy tears away mixed-race family 1970s ohio
##   Everything I Never Told You          1     1    1          1      1     1    1
##   Little Fires Everywhere              0     0    0          0      0     0    0
##                                features
## docs                             celeste ng's little
##   Everything I Never Told You          0    0      0
##   Little Fires Everywhere              1    1      1
## [ reached max_nfeat ... 10 more features ]
```

Most Popular API

```
pop_url <- paste0(
  'https://api.nytimes.com/svc/mostpopular/v2/viewed/30.json?api-key=',
  my_nyt_key
  )


pop_api_results <- GET(pop_url)

Sys.sleep(10)

pop_results = content(pop_api_results, 'parsed')


pop_data <- tibble()
for(i in 1:length(pop_results$results)){

  temp_data <- tibble(
    'id' = pop_results$results[[i]]$id %>% as.character(),
    'abstract' = pop_results$results[[i]]$abstract
    )

  pop_data <- bind_rows(pop_data, temp_data)

}


pop_corpus <- quanteda::corpus(pop_data, docid_field = 'id', text_field = 'abstract')


pop_corpus
```

```
## Corpus consisting of 20 documents.
## 100000008204067 :
## "Such an extensive head injury would likely have left the act..."
##
## 100000008184393 :
## "The word game, released in October, has millions of daily us..."
##
## 100000008205520 :
## "In an interview ahead of the Netflix show’s release, Ms. Sor..."
##
## 100000008192027 :
## "The Republican National Committee voted to censure Represent..."
##
## 100000008193517 :
## "The four-day rescue operation failed to save the life of Ray..."
##
## 100000008143465 :
## "The word game has gone from dozens of players to hundreds of..."
##
## [ reached max_ndoc ... 14 more documents ]
```

```
corp_to_dfm(pop_corpus)
```

```
## Document-feature matrix of: 20 documents, 253 features (94.53% sparse) and 0 docvars.
##                  features
## docs              extensive head injury likely left actor confused unconscious
##   100000008204067         1    1      1      1    1     1        1           1
##   100000008184393         0    0      0      0    0     0        0           0
##   100000008205520         0    0      0      0    0     0        0           0
##   100000008192027         0    0      0      0    0     0        0           0
##   100000008193517         0    0      0      0    0     0        0           0
##   100000008143465         0    0      0      0    0     0        0           0
##                  features
## docs              experts said
##   100000008204067       1    1
##   100000008184393       0    0
##   100000008205520       0    0
##   100000008192027       0    0
##   100000008193517       0    0
##   100000008143465       0    0
## [ reached max_ndoc ... 14 more documents, reached max_nfeat ... 243 more features ]
```

Appendix

```
# csv to corpus
csv_to_corpus
```

```
## function (file, text_col)
## {
##     temp_text = readtext(file = file, text_field = text_col)
##     temp_text$doc_id = seq.int(nrow(temp_text))
##     temp_corpus = corpus(temp_text)
##     return(temp_corpus)
## }
```

```r
# corpus to dfm
corp_to_dfm
```

```
## function (corpus, stopwords = quanteda::stopwords("english"),
##     dict = NULL, thesaur = NULL)
## {
##     if (!is.null(dict) & !is.null(thesaur)) {
##         return(print("you can't provide a dictionairy and a thesaurus"))
##     }
##     else if (!is.null(dict)) {
##         temp_dict = quanteda::dictionary(dict)
##         temp_dfm = quanteda::tokens(corpus, remove_punct = T,
##             remove_symbols = T, remove_url = T) %>% dfm(tolower = T,
##             remove_padding = T) %>% dfm_remove(pattern = stopwords) %>%
##             dfm_lookup(dictionary = temp_dict, exclusive = T,
##                 capkeys = F)
##         return(temp_dfm)
##     }
##     else if (!is.null(thesaur)) {
##         temp_thesaur = quanteda::dictionary(thesaur)
##         temp_dfm = quanteda::tokens(corpus, remove_punct = T,
##             remove_symbols = T, remove_url = T) %>% dfm(tolower = T,
##             remove_padding = T) %>% dfm_remove(pattern = stopwords) %>%
##             dfm_lookup(dictionary = temp_thesaur, exclusive = F,
##                 capkeys = F)
##         return(temp_dfm)
##     }
##     else {
##         temp_dfm = quanteda::tokens(corpus, remove_punct = T,
##             remove_symbols = T, remove_url = T) %>% dfm(tolower = T,
##             remove_padding = T) %>% dfm_remove(pattern = stopwords)
##         return(temp_dfm)
##     }
## }
## <bytecode: 0x7fa1169a64c0>
```