Insights on the Road to 2020

Measuring the Effect of a Fox News Appearance for 2020 Democratic Candidates based on Google Trends

Aleszu Bajak & Ryan Kriegbaum

Final Report

ALY 6110: Data Management and Big Data

Northeastern University

Professor Janos Mako

Spring 2019 CPS Sec 02, 6/27/19

Insights on the Road to 2020

Summary

There is currently contentious debate among the many candidates vying to be the Democratic nominee for president of the United States as to whether or not they should appear on town halls or debates hosted by Fox News (Grunbaum & Ember, 2019).

This report examines Google Trends search data on three 2020 candidates who have already appeared on the network – Senator Bernie Sanders, Mayor Pete Buttigieg and Senator Kirsten Gillibrand – to understand whether there are any discernible changes or spikes in internet searches surrounding these three candidates before or after their Fox News appearances. (Bernie Sanders appeared on Fox News on April 15, 2019; Pete Buttigieg appeared on May 21, 2019; Kirsten Gillibrand appeared on June 2, 2019.)

This kind of analysis, which visualizes the results in an interactive R Shiny app, might be used by political strategists to infer how such an appearance effects regional interest in a candidate to tune their advertising, voter turnout, and fundraising operations, or to make better informed decisions regarding future town hall appearances and campaign stops.

Introduction

The field of Democratic candidates running for U.S. president in 2020 is sprawling. With 24 and counting, American voters are overwhelmed by the sheer size of the field, and many of the candidates struggle to achieve any kind of name recognition. Most candidates are currently polling around 1%, and likely won't make it onto the debate stages at the end of June (2019).

That's what's made the offers from Fox News for candidates to join their anchors onstage for a town hall discussion increasingly appealing. While some Democrats, like Elizabeth Warren and Kamala Harris, have flat out rejected their invitations, others have gladly accepted. Those appearing are likely hoping that the highly popular right-leaning television network will help boost their name recognition and drum up interest across a broader spectrum of Americans.

So the question we have is simple: Are these Fox News appearances effective in generating interest? If they are, how did candidates benefit, and who benefited the most?

We posit that by tapping into the vast amounts of data generated by Google searches, we might be able to gauge the interest of Google-using Americans towards the candidates. Using Google Trends, we pulled historical search data for three candidates that had already appeared on the network and began our search for indicators that might help us gauge any changes in interest. We settled on using related search terms for each candidate, the associated geographical information from searches, and search interest over time in order to form our understanding about the potential value of a Fox News appearance.

Analysis

We chose three 2020 candidates, Bernie Sanders, Pete Buttigieg and Kirsten Gillibrand, who attended a Town Hall style forum on Fox News in April, May and June, respectively. Using the whole names for each candidate as the main search term, or 'keyword,' we used the R package, "gtrendsR," to create a master record of all search data across those months. (Massicotte, 2019) The master record that is created by the gtrendsR package is originally formatted as a list, and contains many potentially useful data points, however as we had decided to focus on search terms, regional and historic information, we next created cleaned and trimmed data frames that contained only the information related to these aspects (Figure 1).

```r
# Data manipulation
library(dplyr)
library(tidyr)
library(stringr)
library(lubridate)


### Master Data Frame ###
# Create a master Data Frame, to limit the number of pulls from google trends
# gTrends pulls as a list of tables, so we will need to pull tables of interest
Master_gTrends <- gtrends(c("Bernie Sanders", "Pete Buttigieg", "Kirsten Gillibrand"),
                          time = "2019-04-04 2019-06-08",
                          geo = c("US"))

### For the Time Series ###

Master_interest <- Master_gTrends$interest_over_time %>%
    as_tibble() %>%
    mutate(date = ymd(date), hits = as.numeric(hits))

# If we want a csv
write.csv(Master_interest, file = "candidate_interest.csv")
```

Figure 1

For the word clouds using related search terms, we had to take an extra intermediate step and save the results as text files in order to properly follow the best practices guide for Natural Language Processing on Apache Spark (Talby, 2017). Once the text files had been uploaded to our local Spark instance, we were able to run basic textual analysis - including sentence segmentation, regular expression character replacement, tokenization, and stop word replacement - all with the aid of the "sparklyr" and "dplyr" R packages. (Luraschi, et. al, & Wickham, et. al) Finally, using the Spark explode method, we converted the text into a table of key-value pairs in order to count the number of times a related search term appeared in our data set (Figure 2).

```
sc <- spark_connect(master = "local", spark_home = "/opt/spark")


# Loading the texts into Spark
Spark_Bernie <- spark_read_text(sc, "bernie_text", "bernie_related.txt", overwrite = TRUE)
Spark_Pete <- spark_read_text(sc, "pete_text", "pete_related.txt", overwrite = TRUE)


### Here is a complete run through for the processing for Bernie

bernie_words <- Spark_Bernie %>%                                        # Selecting candidate
    mutate(regarding = "bernie") %>%                                    # adding a vector for later
    filter(nchar(line) > 0) %>%                                         # filtering out empty segments
    mutate(line = regexp_replace(line, "[_\"\'():;,.!?\\-]", " ")) %>%  # regular expression character replacement
    ft_tokenizer(input_col = "line",
                 output_col = "word_list") %>%                          # tokenization
    ft_stop_words_remover(input_col = "word_list",
                          output_col = "wo_stop_words") %>%             # stop word removal
    mutate(word = explode(wo_stop_words)) %>%                           # Spark NLP explode
    select(word, regarding) %>%
    filter(nchar(word) > 2) %>%
    compute("bernie_words")                                                # Spark NLP compute

# To see what we can see (examine the word list)
glimpse(bernie_words)
```

Figure 2

Once we had our key-to-value pairs, as words-to- number of occurrences pairs, we then displayed these findings in a word cloud, where the number of occurrences is reflected in the size and color of the word. Repeating that process for each candidate results with word clouds much like those found in Figure 3 (below). Related Google searches for all three candidates show a strong relationship to the terms "Fox," "News," "Town" and "Hall." This shows us that the Fox News appearances were likely a significant driver for search interest during that two-month

timespan for all three candidates – and vindicates our hunch that Google searches might be a good way to measure interest generated by a Fox News appearance.

Figure 3

Next, we built a line chart that pulls the "interest_over_time" endpoint from Google Trends and plotted the results using R's "ggplot2" package. Annotations were added to more easily understand where the candidate's appearance fell within that time series. Figure 4 (below) is a sample of the function we created for interacting with the time series in the Shiny application. The complete code for the time series can be found in the Appendix: Code - time_series.R.

```
candidate_time_series <- function(selection){
    ggplot(candidate_interest, aes(date, hits, color = keyword)) +
        geom_line() +
        theme_minimal() +
        labs(title = "Search Interst over Time", x = "Date", y = "Google Trends Interest Score", color = "Search Term") +
        geom_point(data = fox_appearance_labels,
                   size = 2,
                   x = date[selection],
                   y = hits[selection],
                   color = stored_labels[selection],
                   fill = stored_labels[selection]) +
        geom_text(data = fox_appearance_labels,
                  inherit.aes = FALSE,
                  mapping = aes(x = date[selection], y = hits[selection], label = candidate[selection]),
                  nudge_y = 3,
                  fontface = "bold")
}
```

Figure 4

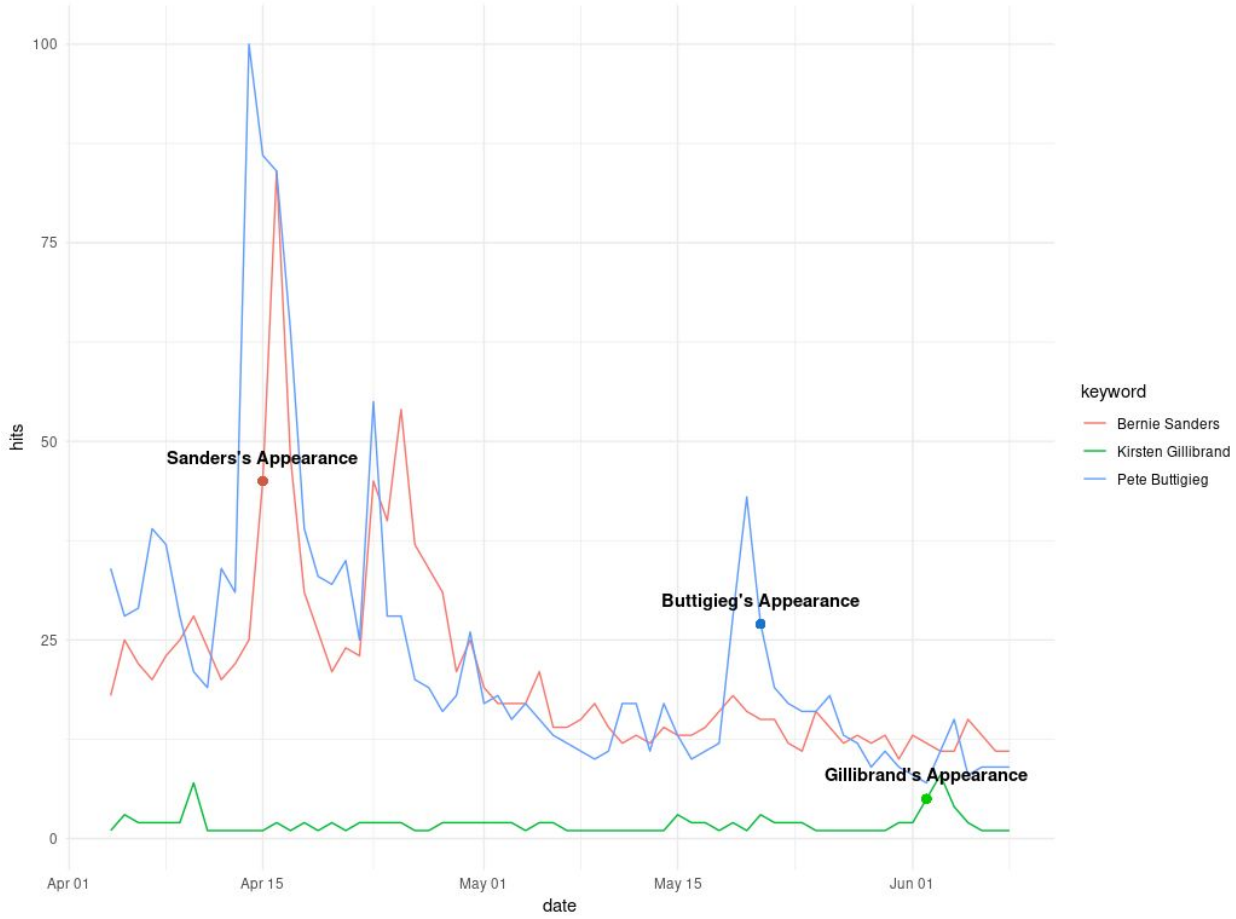

Figure 5

Figure 5 (above) displays a time series that was generated for our three candidates using the code from Figures 3 and 4. In this time series, the y scale of 'hits' represents the relative proportion of searches for these keywords on each of these days. This is then represented on a scale of 0 - 100 based on our local data, making our local maximum, which is Pete Buttigieg's spike in searches, display as our local 100.

The time series shows that there are clear spikes in interest in the direct lead up to each candidates appearance on Fox News. Interestingly, the spikes in interest for both 'Bernie

Sanders' and 'Kirsten Gillibrand' continued to grow even a few days after their town hall appearance, however the spike for 'Pete Buttigieg' appeared to die off just prior to his appearance. Also of note, is that the spikes generated by the Fox News appearance for both Sanders and Gillibrand were the largest spikes for each candidate, while Buttigieg had a massive spike much earlier (again, this is our local maximum). When reviewing the word clouds we previously generated (Figure 2), the related search terms for 'Pete Buttigieg' shows us a likely candidate for this early spike in appearances - Ellen. Further investigation confirmed that Pete Buttigieg indeed appeared on the tv show Ellen on April 12th. Regardless of that unique event, the lead up to the Fox News appearance of Buttigieg still corresponded to a significant spike in searches, just as it did for the other two candidates.

Knowing that a Fox News appearance is responsible for significant spikes in Google search interest, we decided to investigate how these spikes might differ across the United States. We built before and after maps using the "interest_by_region" endpoint of Google Trends for each candidate. Below is a quick example of the function we created to built to quickly and cleanly pull our desired information regarding the regional search interest for each candidate (Figure 6).

```
# build the US map
statesMap = ggplot2::map_data("state")

# To get the regional interest information for maping
Regional_Interest <- function(candidate, date1, date2) {
    Trend <- gtrends(candidate,
                     time = paste(date1, date2, sep = " "),
                     geo = c("US"))$interest_by_region
    Trend <- Trend %>%
        mutate(hits = as.numeric(hits),
               hits = replace_na(hits, 0),
               region = str_to_lower(location),
               date = date2)
    return(Trend)
}
```

Figure 6

The maps below show regional interest for *the day before* and *the day after* the Fox News appearance for each candidate (Figure 7). The maps seem to demonstrate that each candidate's appearance not only created search spikes, but they also appeared to create significant search volume across a broader swathe of states. A few states in particular saw gains which we found to be noteworthy. Pete Buttigieg, for example, saw a significant and notable increases in search volume in Wyoming (+43 in Google search interest) South Dakota (+91), and New Hampshire (+32). Perhaps even more dramatic, were Kirsten Gillibrand's gains in Wyoming (+100) - which appeared to radiate outward across Idaho (+31) into the Pacific Northwest. Other notable gains for Gillibrand can be seen in Nevada, Minnesota, and Arkansas. Bernie Sanders saw notable gains in Oregon (+38). Though perhaps less obvious, Sanders' gains across the so called 'Rust' and 'Bible Belts' should also be noted.

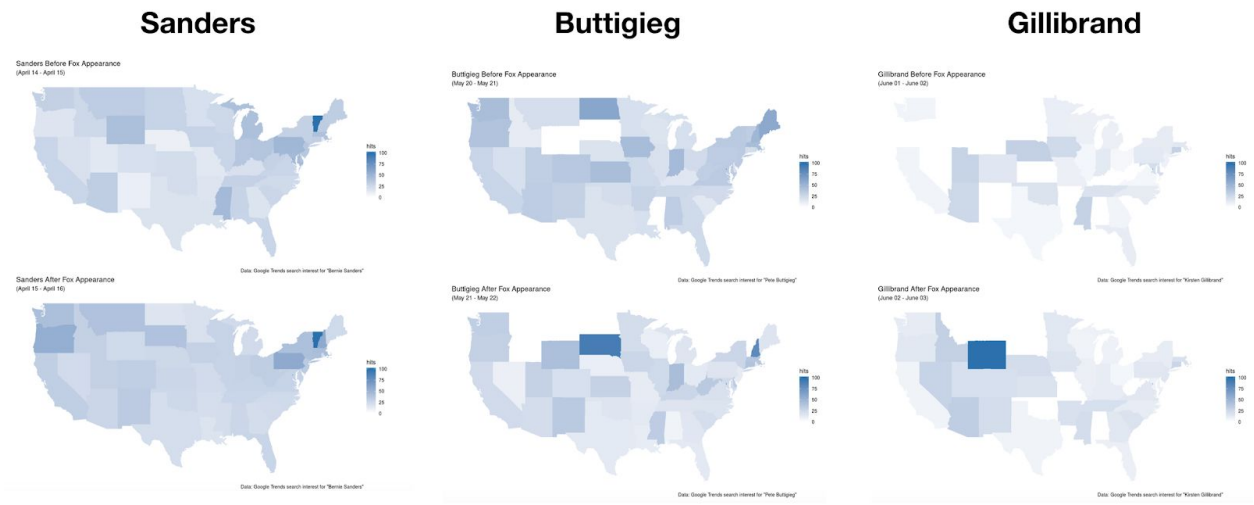

Figure 7

Shiny app

In order to make all of this data and our analysis accessible, we built a dashboard using R Shiny showing the word cloud, the before and after maps, and the line chart in separate tabs. Using a simple drop-down menu, the Shiny app allows for quick and easy comparison between our three candidates Sanders, Buttigieg and Gillibrand - and it is built to allow easy expansion for new candidates and new events.

Link: https://ryan-kriegbaum.shinyapps.io/2020_Insights/

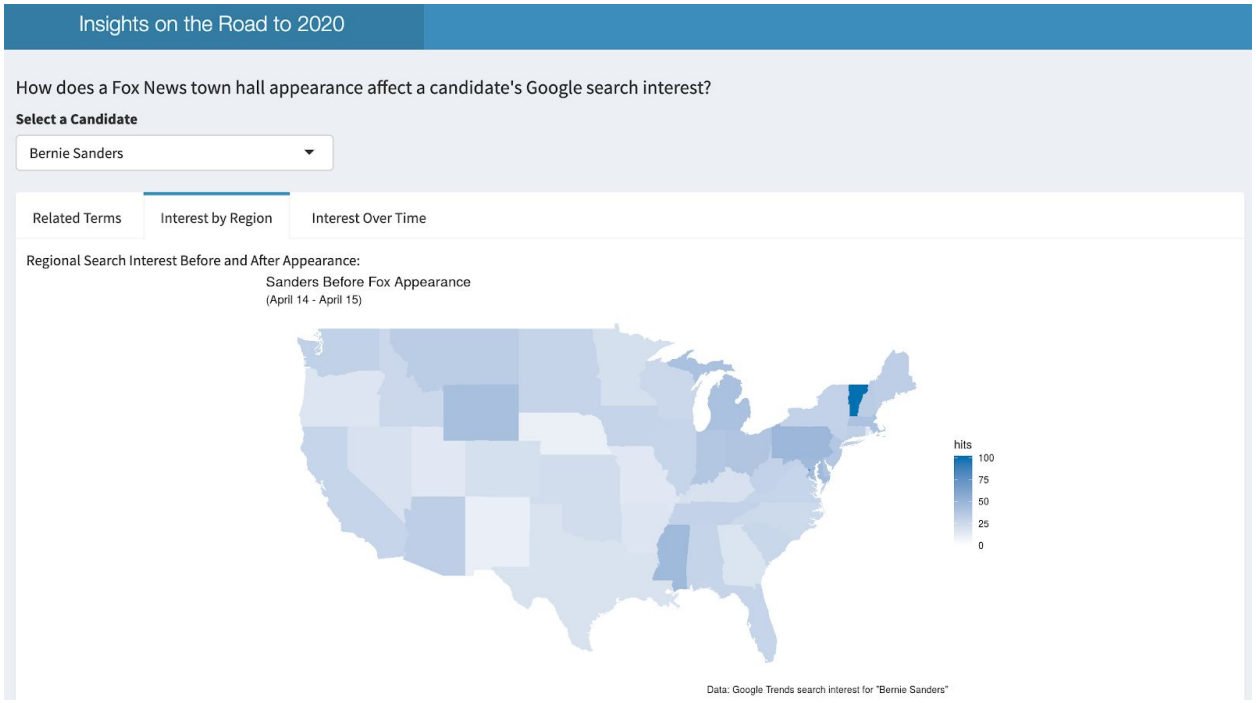

Figure 8

Results, conclusions and recommendations

Our Shiny app allows users to select a 2020 presidential candidate and explore the Google search interest they generated before and after their appearance on Fox News. Our most salient conclusion is that the Fox News town hall certainly does generate interest from Google-using Americans. These appearances appear to have paid off the most for Bernie Sanders, as his search interest seems to have begun well before and peaked after his appearance. Furthermore the interest generated for Bernie Sander's seemed to be the most consistent and evenly spread across the country. There are also several other insights we drew from the analysis that could help political strategists, researchers and journalists:

- Fox News town halls seem to have a positive impact on candidate search interest in states farther from the candidates' home state, like Gillibrand seeing more pickup in the Pacific Northwest. Interest changes also appear to be strongest in so called 'Republican stronghold' states, as seen in the search interest increases in states like the Dakotas, Montana, and Idaho (Lynn, 2016). This infers that there is a sort of geographic expansion of name recognition following an appearance on national television, and that appearing on a right-leaning television network will enhance the interest in right-leaning regions.
- Bernie Sanders saw the largest increase in search interest following his April 15 Fox News town hall, as compared with the other two candidates. This might suggest that there was a novel factor to his appearance, which could be specifically the candidate himself, or perhaps because he was the first to participate. There does appear to be a sort of diminishing returns on the interest generated for the candidates, as the third appearance generated less interest than the second. This might suggest that political campaigns are safe to reject an invitation when they will be the n^{th} candidate to participate.
- Search interest for 'Pete Buttigieg' peaked more than a month before his Fox News town hall, and he was the only candidate who did not see a peak in interest *after* his appearance. This suggests that this analysis could be helpful in finding an optimal strategy for multiple television appearances, or in recommending a strategy for striking when the "iron is hot" . Further analysis might help a specific candidate capitalize on their television generated interest, and the eventual incorporation of streaming data might increase the speed of providing these insights.

Clearly, search interest data from Google Trends is a valuable tool in a larger toolkit of real-time, historic and predictive data for understanding voter attitudes towards candidates now at the disposal of political strategists, political scientists and political journalists.

We strongly believe this Shiny app prototype might be especially valuable for Democratic strategists looking to survey interest in their respective candidates. With other data sources and methodologies from data science, we could see a similar app being used to zoom in on and help interpret regional voter interest patterns for candidate field offices across the country.

Code

ui.R

```
# Shiny
library(shiny)
library(shinydashboard)

# Graphics
library(ggplot2)

# Google Trends
library(gtrendsR)

# Data manipulation
library(dplyr)
library(tidyr)
library(stringr)
library(lubridate)

# Begin UI Definitions
ui <- fluidPage(
        titlePanel("The road to 2020"),
        tags$div(class="header", checked=NA,
                tags$p("Do Fox News town hall appearances affect Google search interest for 2020
candidates?")),
        hr(),

        sidebarLayout(
        sidebarPanel(
        selectInput("candidate_selected",
                        label = "Select a Candidate",
                        choices = c("Bernie Sanders" = 1,
                                "Pete Buttigieg"  = 2,
                                "Kirsten Gillibrand" = 3),
                        selected = 1)
        ),

        mainPanel(
        tabsetPanel(
                tabPanel("Related terms”, plotOutput("wordcloud")),
                tabPanel("Interest by region",
                                plotOutput("Before_Map"),
                                plotOutput("After_Map")),
                tabPanel("Interest over time", plotOutput("TS"))
        )
        )
        )
)
```

server.R

```
# Define Server function
shinyServer(function(input,output){

        # Word Cloud
        output$wordcloud <- renderPlot({

        selected_candidate <- as.numeric(input$candidate_selected) # default is 1 for Bernie

        # function and data all defined in time_series.R
        source(file = "word_cloud.R", local = TRUE)
        shiny_word_clouds[[selected_candidate]]

        })

        # Time Series
        output$TS <- renderPlot({

        selected_candidate <- as.numeric(input$candidate_selected) # default is 1 for Bernie

        # function and data all defined in time_series.R
        source(file = "time_series.R", local = TRUE)
        candidate_time_series(selected_candidate)

        })

        # Before Map
        output$Before_Map <- renderPlot({

        selected_candidate <- as.numeric(input$candidate_selected)

        # function and data all defined in time_series.R
        source(file = "interest_maps.R", local = TRUE)
        candidate_before_maps[[selected_candidate]]
        })

        #After map
        output$After_Map <- renderPlot({

        selected_candidate <- as.numeric(input$candidate_selected) # default is 1 for Bernie
        source(file = "interest_maps.R", local = TRUE)
        candidate_after_maps[[selected_candidate]]
        })
})
```

interest_maps.R

```
library(tidyr)
library(dplyr)
library(stringr)
library(ggplot2)
library(lubridate)
library(maps)

# build the US map
statesMap = ggplot2::map_data("state")

### Dates of Interest

# Fox Appearance Dates
bernie_fox_appearance <- ymd("2019-04-15")
pete_fox_appearance <- ymd("2019-05-21")
gilli_fox_appearance <- ymd("2019-06-02")

# Function to declare a few data types to prevent warnings or coercion later
standardised <- function(csv_file){
        standardized_csv <- csv_file %>%
        as_tibble() %>%
        mutate(date = ymd(date), hits = as.numeric(hits), region = as.character(region))
        return(standardized_csv)
}

# Step 1
# Load the data (previously constructed in data.R)
### Sanders map data
Sanders_before <- read.csv("Sanders_before.csv") %>% standardised()
Sanders_after <- read.csv("Sanders_after.csv") %>% standardised()

# Buttigieg map data
Pete_before <- read.csv("Pete_before.csv") %>% standardised()
Pete_after <- read.csv("Pete_after.csv") %>% standardised()

# Gillibrand map data
Gilli_before <- read.csv("Gilli_before.csv") %>% standardised()
Gilli_after <- read.csv("Gilli_after.csv") %>% standardised()


# For joining the data on geographic data
Map_Prep <- function(candidate_data){
        geo_data <- merge(statesMap, candidate_data, by = "region")      #merge the ggplot created map
        geo_data <- candidate_data %>%                                   #join the data on the region
        left_join(x = ., y = statesMap, by = "region")
        return(geo_data)
}

# Helper Function for displaying the dates in the subtitles
display_date <- function(date){
        return(format(date, format = "%B %d"))
}
```

```
# Map making function
# Requires several inputs
# map_info is fed from Map_Prep
# Map title is a written string
# date1 and date2 are the date bookends "From date1 to date2"
Make_Map <- function(map_info, map_title, date1, date2) {

        new_map <- ggplot(map_info, aes(x = long, y = lat)) +
        theme_void() +
        geom_polygon(aes(group = group,
                         fill = hits)) +
        labs(title = map_title,
                subtitle = paste("(", paste(display_date(date1), display_date(date2), sep = " - "),
")", sep = ""),
                caption = paste("Data: Google Trends search interest for \"", map_info$keyword[1],
"\"", sep = ""),
                color = "Interest Score")+
        coord_fixed(1.3) +
        scale_fill_gradient(low = "#ffffff", high = "#0072b2",
                         space = "Lab", na.value = "black", guide = "colourbar",
                         aesthetics = "fill", position = "left", limits = c(0,100))

        return(new_map)
}


# Make the Sanders maps
Sanders_map_before <- Make_Map(Map_Prep(Sanders_before), "Sanders Before Fox Appearance",
bernie_fox_appearance-1, bernie_fox_appearance)
Sanders_map_after <- Make_Map(Map_Prep(Sanders_after), "Sanders After Fox Appearance",
bernie_fox_appearance, bernie_fox_appearance+1)

# Make the Buttigieg maps
Pete_map_before <- Make_Map(Map_Prep(Pete_before), "Buttigieg Before Fox Appearance",
pete_fox_appearance-1, pete_fox_appearance)
Pete_map_after <- Make_Map(Map_Prep(Pete_after), "Buttigieg After Fox Appearance", pete_fox_appearance,
pete_fox_appearance+1)

# Make the Gillibrand maps
Gilli_map_before <- Make_Map(Map_Prep(Gilli_before), "Gillibrand Before Fox Appearance",
gilli_fox_appearance-1, gilli_fox_appearance)
Gilli_map_after <- Make_Map(Map_Prep(Gilli_after), "Gillibrand After Fox Appearance",
gilli_fox_appearance, gilli_fox_appearance+1)

# For selection within the Shiny App
candidate_before_maps <- list(Sanders_map_before, Pete_map_before, Gilli_map_before)
candidate_after_maps <- list(Sanders_map_after, Pete_map_after, Gilli_map_after)
```

word_cloud.R

```
library(gtrendsR)
library(tidytext)

# Data for related searches from gTrends
# Move to data to convert to csv somehow?

Bernie_related <- gtrends(c("Bernie Sanders"), time = "2019-04-04 2019-06-08", geo =
c("US"))$related_queries
Pete_related <- gtrends(c("Pete Buttigieg"), time = "2019-04-04 2019-06-08", geo =
c("US"))$related_queries
Gilli_related <- gtrends(c("Kirsten Gillibrand"), time = "2019-04-04 2019-06-08", geo =
c("US"))$related_queries


# Tidytext function for Shiny, Spark does not work in deployed app
Word_Prep <- function(related_data){
        related_data %>%
        unnest_tokens(word, value) %>%
        count(word, sort = TRUE)
}

# Textified
Bernie_related <- Word_Prep(Bernie_related)
Pete_related <- Word_Prep(Pete_related)
Gilli_related <- Word_Prep(Gilli_related)


# Word Clouds
Bernie_Cloud <- Bernie_related %>%
        filter(!word == "bernie", !word == "sanders") %>%
        head(20) %>%
        collect() %>%
        with(ggwordcloud::ggwordcloud(
        word,
        n,
        min.freq = 1,
        colors = c("#999999", "#E69F00", "#56B4E9","#56B4E9"))+
        labs(title = "\"Bernie Sanders\" related search terms",
        subtitle = paste("(April - June)"),
        caption = paste("Date: Google Trends related search terms"))
) # coral3 is TS color

Pete_Cloud <- Pete_related %>%
        filter(!word == "pete", !word == "buttigieg") %>%
        head(20) %>%
        collect() %>%
        with(ggwordcloud::ggwordcloud(
        word,
        n,
        min.freq = 1,
        colors = c("#999999", "#E69F00", "#56B4E9","#56B4E9"))+
        labs(title = "\"Pete Buttigieg\" related search terms",
                subtitle = paste("(April - June)"),
```

```
                caption = paste("Date: Google Trends related search terms"))
        )

Gilli_Cloud <- Gilli_related %>%
        filter(!word == "kirsten", !word == "gillibrand") %>%
        head(20) %>%
        collect() %>%
        with(ggwordcloud::ggwordcloud(
        word,
        n,
        min.freq = 1,
        colors = c("#999999", "#E69F00", "#56B4E9","#56B4E9"))+
        labs(title = "\"Kirsten Gillibrand\" related search terms",
                subtitle = paste("(April - June)"),
                caption = paste("Date: Google Trends related search terms"))
        ) # green3 is TS color

shiny_word_clouds <- list(Bernie_Cloud, Pete_Cloud, Gilli_Cloud)
```

time_series.R

```
library(dplyr)
library(lubridate)
library(ggplot2)
library(ggpmisc)


### Dates of Interest

# Fox Appearance Dates
bernie_fox_appearance <- ymd("2019-04-15")
pete_fox_appearance <- ymd("2019-05-21")
gilli_fox_appearance <- ymd("2019-06-02")


### Loading the Data

# reading the csv created in data.R
candidate_interest <- read.csv("candidate_interest.csv")

# declaring data types
candidate_interest <- candidate_interest %>%
        as_tibble() %>%
        mutate(date = ymd(date), hits = as.numeric(hits))

### Appearances

# Function to find the gTrends "hits" score for any dates of interest
hits_of_interest <- function(data_set, candidate, date_of_interest){
        appearance <- data_set %>%
        filter(keyword == candidate, date == date_of_interest) %>%
        select(hits) %>%
        as.numeric()
```

```
        return(appearance)
}

# Bernie Sanders appeared on Fox News on April 15, 2019;
bernie_appearance_hits <- hits_of_interest(candidate_interest, "Bernie Sanders", bernie_fox_appearance)

#Pete Buttigieg appeared on May 21, 2019;
pete_appearance_hits <- hits_of_interest(candidate_interest, "Pete Buttigieg", pete_fox_appearance)

#Kirsten Gillibrand appeared on June 2, 2019.
gilli_appearance_hits <- hits_of_interest(candidate_interest, "Kirsten Gillibrand",
gilli_fox_appearance)

# Create labels for the points
fox_appearance_labels <- data.frame(
        candidate <- c("Sanders's Appearance", "Buttigieg's Appearance", "Gillibrand's Appearance"),
        date <- c(bernie_fox_appearance, pete_fox_appearance, gilli_fox_appearance),
        hits <- c(bernie_appearance_hits, pete_appearance_hits, gilli_appearance_hits),
        stored_labels <- c("coral3", "dodgerblue3", "green3")
)

# Plot the time series
candidate_time_series <- function(selection){
        ggplot(candidate_interest, aes(date, hits, color = keyword)) +
        geom_line() +
        theme_minimal() +
        labs(title = "Search Interest over Time", x = "Date", y = "Google Trends Interest Score", color
= "Search Term") +
        geom_point(data = fox_appearance_labels,
                size = 2,
                x = date[selection],
                y = hits[selection],
                color = stored_labels[selection],
                fill = stored_labels[selection]) +
        geom_text(data = fox_appearance_labels,
                inherit.aes = FALSE,
                mapping = aes(x = date[selection], y = hits[selection], label = candidate[selection]),
                nudge_y = 3,
                fontface = "bold")
}
```

data.R

```
#### This should only be run for initial setup
#### Do not use this file for Shiny
#### These data files are too large without this processing for efficient hosting

# Google Trends
```

```
library(gtrendsR)

# Data manipulation
library(dplyr)
library(tidyr)
library(stringr)
library(lubridate)


### Master Data Frame ###
# Create a master Data Frame, to limit the number of pulls from google trends
# gTrends pulls as a list of tables, so we will need to pull tables of interest
Master_gTrends <- gtrends(c("Bernie Sanders", "Pete Buttigieg", "Kirsten Gillibrand"),
                          time = "2019-04-04 2019-06-08",
                          geo = c("US"))


### For the Time Series ###

Master_interest <- Master_gTrends$interest_over_time %>%
        as_tibble() %>%
        mutate(date = ymd(date), hits = as.numeric(hits))

# csv for Shiny
write.csv(Master_interest, file = "candidate_interest.csv")



### For the Maps ###

# build the US map
statesMap = ggplot2::map_data("state")

# To get the regional interest information for mapping
Regional_Interest <- function(candidate, date1, date2) {
        Trend <- gtrends(candidate,
                         time = paste(date1, date2, sep = " "),
                         geo = c("US"))$interest_by_region
        Trend <- Trend %>%
        mutate(hits = as.numeric(hits),
                hits = replace_na(hits, 0),
                region = str_to_lower(location),
                date = date2)
        return(Trend)
}


### Smaller CSV Creation for Faster Loading in Shiny

### Sanders map data
Sanders_before <- Regional_Interest("Bernie Sanders", bernie_fox_appearance-1, bernie_fox_appearance)
write.csv(Sanders_before, file = "Sanders_before.csv")

Sanders_after <- Regional_Interest("Bernie Sanders", bernie_fox_appearance, bernie_fox_appearance+1)
write.csv(Sanders_after, file = "Sanders_after.csv")
```

```
# Buttigieg map data
Pete_before <- Regional_Interest("Pete Buttigieg", pete_fox_appearance-1, pete_fox_appearance)
write.csv(Pete_before, file = "Pete_before.csv")

Pete_after <- Regional_Interest("Pete Buttigieg", pete_fox_appearance, pete_fox_appearance+1)
write.csv(Pete_after, file = "Pete_after.csv")


#Gillibrand map data
Gilli_before <- Regional_Interest("Kirsten Gillibrand", gilli_fox_appearance-1, gilli_fox_appearance)
write.csv(Gilli_before, file = "Gilli_before.csv")

Gilli_after <- Regional_Interest("Kirsten Gillibrand", gilli_fox_appearance, gilli_fox_appearance+1)
write.csv(Gilli_after, file = "Gilli_after.csv")
```

References

Grynbaum, M. M., & Ember, S. (2019, April 17). 2020 Democrats Seek Voters in an Unusual Spot: Fox News. Retrieved June 1, 2019, from https://www.nytimes.com/2019/04/17/us/politics/fox-news-democrats-2020.html

Flood, B. (2019). Fox News announces town hall with 2020 presidential hopeful Julian Castro. Retrieved from https://www.foxnews.com/entertainment/fox-news-announces-town-hall-with-2020-presidential-hopeful-julian-castro

Luraschi, J., Kuo, K., Ushey, K., Allaire, J., Macedo, S., RStudio, & The Apache Software Foundation. (2019, May 17). sparklyr (Version 1.0.1) [Computer Software]. Retrieved from https://cran.r-project.org/web/packages/sparklyr/sparklyr.pdf

Lynn, J. (2016, March 09). Strongest Republican Party States In The U.S. Retrieved from https://www.worldatlas.com/articles/strongest-republican-party-states-in-the-u-s.html

Massicotte, P. (2019). Package 'gtrendsR'. Retrieved from https://cran.r-project.org/web/packages/gtrendsR/gtrendsR.pdf

RealClearPolitics (2019). 2020 Democratic Presidential Nomination. Retrieved from https://www.realclearpolitics.com/epolls/2020/president/us/2020_democratic_presidential_nomination-6730.html

Rosas, J. (2019). Kirsten Gillibrand's Fox News town hall tops all three CNN town halls. Washington Examiner. Retrieved from https://www.washingtonexaminer.com/news/kirsten-gillibrands-fox-news-town-hall-tops-all-three-cnn-town-halls

Talby, D. (2017). Introducing the Natural Language Processing Library for Apache Spark. Retrieved from https://databricks.com/blog/2017/10/19/introducing-natural-language-processing-library-apache-spark.html

Wickham, H., Francois, R., Henry, L., Muller, K., & RStudio. (2019, May 8). Dplyr (Version 0.8.1) [Computer software]. Retrieved from https://cran.r-project.org/web/packages/dplyr/dplyr.pdf

Wickham, H., Chang, W., Henry, L., Pederson, T. L., Takahashi, K., Wilke, C., . . . RStudio. (2019, June 16). Ggplot2 (Version 3.2.0) [Computer software]. Retrieved from https://cran.r-project.org/web/packages/ggplot2/index.html