

Pre- and Post-Debate Democratic Primary Data: Twitter, Google Trends, and Polls

Fundamentals of Computing and Data Display, Fall 2019

Zoe Padgett and Fatou Thiam

2019-12-11

Contents

Introduction	2
Data	2
Results	7
Discussion	22
References	23

Introduction

Twitter and Google Trends are becoming increasingly popular tools for social science researchers. They represent an easily accessible source of large amounts of data, which has become advantageous as survey response rates decline and costs rise. Researchers interested in predicting election results have begun looking to Twitter data to replace or supplement traditional election polls, with mixed results (Gayo-Avello 2013).

Recently, there have been studies using sentiment analysis of Twitter data to predict election outcomes in India (Salunkhe and Deshmukh 2017), to predict state-level polling results in the U.S. (Beauchamp 2017), and to predict the winners of three presidential elections in Latin America (Gaurav et al. 2013). Beauchamp (2017) found that Twitter data may be useful in making state-level campaign strategy decisions. Additionally, Kassraie, Modirshanechi, and Aghajan (2017) used Google Trends and Twitter data to predict the 2016 U.S. election outcomes with only 1% error.

We are interested in whether Twitter data and Google trends data could be used to supplement polling results, by providing real-time information to candidates while they wait for polling data to come in. For example, candidates may be interested in understanding how public opinion has shifted immediately after a debate in order to run a more agile campaign. This project is an exploratory analysis of Twitter and Google Trends data to see if pre- and post-debate polling data during the 2020 U.S. Democratic primary election aligns with these real-time sources.

Data

All data used to create the tables, figures, and shiny app in this paper can be found on our Github repository at https://github.com/znpadgett/surv727_padgett_thiam.

Twitter

Using the `rtweet` package and the `search_tweet` function, we were able to gather sufficient tweets for our project and separated in two datasets; pre- and post- debate. The initial dataset contains a little over 1.3 million tweets which was rich enough to proceed with analysis. After removing irrelevant tweets such as tweets containing “Ukraine”, we were left with over 1.2 million tweets. This was used to create the pre- and post- debate datasets. The pre-debate dataset has 226,799 tweets and 998,780 tweets for the post-debate dataset. As expected, there was a lot more tweets after the debate than before. We used the Twitter location field to perform analysis at the state level. However, the location column is an open text and not regulated. We selected several states such New York, Virginia, New Hampshire, Nevada, Illinois, Iowa, Ohio, California and DC to focus our analysis on.

Below is our token to access the Twitter API and start collecting tweets. Note that these codes are fake, for security purposes.

```
create_token(app = "fcdd-course", consumer_key = "XXXXXXXXXXXXXXXXXXXX",
  consumer_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
  access_token = "XXXXXXXXXXXXXXXXXXXX-XXXXXXXXXXXXXXXXXXXX",
  access_secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX")
```

Quering Tweets

Using the `search_tweet` function in the `rtweet` package, we will be quering tweets with the specified keywords below. We limited our tweets pull to November 20th and November 21st. The reasoning behind this is to get a sense on how Twitter users feel pre- and post- debate.

```
dem <- search_tweets("#democrats OR #candidates OR #election2020 OR #BIDEN
  OR #sanders OR #warren OR #harris
  OR #buttigieg OR #steyer OR #yang OR #booker
  OR #klobuchar OR #gabbard
  OR @KamalaHarris OR @JoeBiden OR @BernieSanders
  OR @ewarren OR @PeteButtigieg
  OR @TomSteyer OR @AndrewYang OR @BookerCory
  OR @amyklobuchar OR @TulsiGabbard ",
  since = "2019-11-20", until = "2019-11-21", n = 5e+05, retryonratelimit = TRUE,
  verbose = TRUE)
```

Data Cleaning

Even with specified keywords, irrelevant tweets were also made it to the initial dataset mainly with the Ukraine issue at the time of pulling. We removed all tweets that had the word “Ukraine” in order to focus completely on tweets regarding the debate.

```
vars <- c("text", "location", "created_at")
df1 <- dem[vars]
df2 <- dem2[vars]

df1 <- df1 %>% mutate(ukraine = (str_detect(df1$text, regex("Ukraine",
  ignore_case = TRUE)))) %>% filter(ukraine == "FALSE") %>%
  select(-ukraine)

df2 <- df2 %>% mutate(ukraine = (str_detect(df2$text, regex("Ukraine",
  ignore_case = TRUE)))) %>% filter(ukraine == "FALSE") %>%
  select(-ukraine)

tweets <- rbind(df1, df2)
```

Next, we separate the tweets by candidates in order to do analysis at the candidate level. It’s important to note that one tweets can be addressed to multiple candidates. In that case,

the tweet will be found in the each of those candidates dataset. The code below exemplifies this step of the cleaning process.

```
tweets$BS <- (str_detect(tweets$text, regex("#Sanders|@BernieSanders",
  ignore_case = TRUE)))
tweets$KH <- (str_detect(tweets$text, regex("#harris |@KamalaHarris",
  ignore_case = TRUE)))
tweets$JB <- (str_detect(tweets$text, regex("#biden |@JoeBiden",
  ignore_case = TRUE)))
tweets$EW <- (str_detect(tweets$text, regex("#warren |@ewarren",
  ignore_case = TRUE)))
tweets$PB <- (str_detect(tweets$text, regex("#buttigieg|@PeteButtigieg",
  ignore_case = TRUE)))
tweets$TS <- (str_detect(tweets$text, regex("#steyer | @TomSteyer",
  ignore_case = TRUE)))
tweets$AY <- (str_detect(tweets$text, regex("#yang| @AndrewYang",
  ignore_case = TRUE)))
tweets$BC <- (str_detect(tweets$text, regex("#booker | @BookerCory",
  ignore_case = TRUE)))
tweets$AK <- (str_detect(tweets$text, regex("#klobuchar | @amyklobuchar",
  ignore_case = TRUE)))
tweets$TG <- (str_detect(tweets$text, regex("#gabbard |@TulsiGabbard",
  ignore_case = TRUE)))
```

Here we reformat the created_at column as a date & time variable in order to separate the tweets in two groups : pre- and post- debate.

```
tweets <- tweets %>% mutate(date = as.POSIXct(created_at, tryFormats = c("%Y-%m-%d %H:%M:%S", "%Y-%m-%d %H:%M:%OS")))
df1 <- df1 %>% mutate(date = as.POSIXct(created_at, tryFormats = c("%Y-%m-%d %H:%M:%S", "%Y-%m-%d %H:%M:%OS")))
df2 <- df2 %>% mutate(date = as.POSIXct(created_at, tryFormats = c("%Y-%m-%d %H:%M:%S", "%Y-%m-%d %H:%M:%OS")))
```

All the tweets received before 9 p.m. on November 20th were accounted for in the pre-debate dataset and all tweets from 11 p.m. on November 20th to the next day are in the post-debate dataset. Note that tweets during the debate (9 - 11 p.m.) were removed.

```
pre_debate <- tweets %>% filter(date(date) == "2019-11-20" &
  hour(date) < 21)

post_debate <- tweets %>% filter(date(date) == "2019-11-20" &
  hour(date) >= 23 | date(date) == "2019-11-21")
```

```
pre_debate <- pre_debate %>% select(-date, -created_at)
post_debate <- post_debate %>% select(-date, -created_at)
```

Sentiment Analysis

Sentiment analysis of the tweets will not be performed for all candidates but only 5 : Harris, Sanders, Biden, Warren & Buttigieg.

First we need to prep the tweets by removing all non-words such as emojis and use the sentiment analysis package for analysis. Although the sentiment analysis package uses 5 dictionaries, we will only look at the results from the GI dictionary. For the pre-debate tweets sentiment analysis, we used all the tweets from the dataset. However, for the post-debate sentiment analysis we sampled from the dataset due to volume and space issue. The code below exemplifies this process. The sample size for each candidate post-debate dataset is the same as the pre-debate number of tweets for consistency. For instance, the pre-debate dataset for Sanders had 31,603 tweets, the sample size for the post-debate dataset for Sanders was 31,603.

```
bs <- pre_debate %>% filter(BS == "TRUE")

usableText = str_replace_all(bs$text, "[^[:graph:]]", " ")

usableText <- tolower(bs$text)

usableText <- iconv(usableText, "UTF-8", "ASCII", sub = "byte")

sentiments_bs = analyzeSentiment(as.character(usableText))

pre_sent_pos <- data.frame(Candidate = c("Biden", "Sanders",
  "Warren", "Harris", "Buttigieg"), Positive = c((sum(sentiments_jb$PositivityGI))/274
  (sum(sentiments_bs$PositivityGI))/31603, (sum(sentiments_ew$PositivityGI))/18628,
  (sum(sentiments_kh$PositivityGI))/16362, (sum(sentiments_pb$PositivityGI))/27242))
```

Google Trends

This section describes gathering the Google Trends data using the package gtrendsR (Masicotte and Eddelbuettel 2019). First, we register our Google API key. Then, we pull data for each candidate from the 2 days preceding and two days following the debate (November 18 through 22). We have to pull the data in two separate blocks, because gtrendsR only allows us to use five search terms at a time. We limit the location of searches to the US. Please note that the key here is not a real API key, for security purposes.

```

register_google(key = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX")
res1 <- gtrends(c("Joe Biden", "Bernie Sanders", "Elizabeth Warren",
  "Kamala Harris", "Pete Buttigieg"), geo = "US", time = "2019-11-18 2019-11-22",
  low_search_volume = T)

res2 <- gtrends(c("Tom Steyer", "Andrew Yang", "Cory Booker",
  "Amy Klobuchar", "Tulsi Gabbard"), geo = "US", time = "2019-11-18 2019-11-22",
  low_search_volume = T)

```

Geocoding

Next, we compile and clean the Google Trends location data and prepare it for geocoding.

```

interest_by_location1 <- as_tibble(res1$interest_by_dma)
interest_by_location2 <- as_tibble(res2$interest_by_dma)
interest_by_location <- rbind(interest_by_location1, interest_by_location2)

locations_df <- as.data.frame(interest_by_location)
locations_df$location <- as.character(locations_df$location)

```

Then, we geocode the Google trends data.

```

gc_locations <- as_tibble(mutate_geocode(locations_df, location))

```

Data Cleaning

Next, we categorize the Google Trends data into pre-debate and post-debate data, based on the date.

```

interest_over_time1 <- as_tibble(res1$interest_over_time)
interest_over_time2 <- as_tibble(res2$interest_over_time)
interest_over_time <- rbind(interest_over_time1, interest_over_time2)

interest_over_time_pre <- interest_over_time %>% filter(date <
  "2019-11-19") %>% mutate(Pre_post = "Pre-debate")

interest_over_time_post <- interest_over_time %>% filter(date >
  "2019-11-20") %>% mutate(Pre_post = "Post-debate")

interest_over_time_all <- rbind(interest_over_time_pre, interest_over_time_post)

```

Polls

Polling data was collected from RealClearPolitics.com, which aggregates weekly polls. We created a dataset using the RealClearPolitics average before and after the November 20 debate. Because the website changes often, and we only needed a small snapshot of the data, it was more efficient to clean the data in Excel and import to R than to use web scraping.

```
github_link <- "https://github.com/znpadgett/surv727_padgett_thiam/raw/master/Data/Proje
temp_file <- tempfile(fileext = ".xlsx")
req <- GET(github_link, write_disk(path = temp_file))
polling_data <- readxl::read_excel(temp_file)
```

Results

Data Exploration

Twitter

Table 1 shows the number of mentions by candidate pre- and post- debate and Table 2 (CA only) shows similar results adding location as a factor. In both tables, we see that there were lot more tweets after the debate. This was anticipated with the debate being a headline the next day. This was also the reason why we decided to focus on pre- and post- debate analysis and not focus on the number of tweets. Sentiment analysis was performed using the GI dictionnaire in the sentiment analysis package (Feuerriegel and Proellocks 2019). Each tweet received a positive, neutral and/ or negative score. For this project, we summed the positive score of all tweets and divided by the number of tweets in each dataset. This gave us the average tweet positivity by candidate shown in Table 3. Candidate tweet positivity score pre- debate are very close with 15% for Sanders being the highest. We notice a slight increase for all candidates with Sanders in the lead still at 16%.

```
panderOptions("table.caption.prefix", "Table 1: ")
pander(table1, caption = "Count of tweets that mentionned
  candidates pre- and post- debate.")
```

Candidate	Pre_debate	Post_debate
Biden	27477	124353
Booker	19	451
Buttigieg	27242	105294
Gabbard	8515	81667
Harris	16362	148423
Klobuchar	1308	24254
Sanders	31603	140585

Candidate	Pre_debate	Post_debate
Steyer	504	10372
Warren	18628	76875
Yang	21975	139224

Table 1: Count of tweets that mentionned candidates pre- and post- debate.

```
panderOptions("table.caption.prefix", "Table 2: ")
pander(table2, caption = "Count of tweets that mentionned
  candidates in California pre- and post- debate.")
```

Candidate	Location	Pre_debate	Post_debate
Biden	California	1273	6444
Booker	California	3	42
Buttigieg	California	1847	6701
Gabbard	California	488	5842
Harris	California	1310	11559
Klobuchar	California	60	1433
Sanders	California	1936	9111
Steyer	California	44	966
Warren	California	1257	4681
Yang	California	2069	11619

Table 2: Count of tweets that mentionned candidates in California pre- and post- debate.

```
panderOptions("table.caption.prefix", "Table 3: ")
pander(table3, caption = "Sentiment Analysis average positivity
  score pre- and post- debate by candidate.")
```

Candidate	Pre_debate	Post_debate
Biden	0.1205	0.1458
Buttigieg	0.1463	0.1474
Harris	0.1293	0.136
Sanders	0.149	0.1617
Warren	0.138	0.1433

Table 3: Sentiment Analysis average positivity score pre- and post- debate by candidate.

Create functions

Create a function to tabulate the count and proportion of tweets by candidates.

```
type_var <- unlist(map(pre_debate, class))

freq_tab <- function(x) {
  # make table with count and frequency
  tab <- cbind(Count = table(x, useNA = "ifany"), Prop = round(prop.table(table(x,
    useNA = "ifany")), 2))
  # get the categories as variable and rearrange
  tab <- as.data.frame(tab) %>% tbl_df() %>% mutate(Cat = row.names(tab)) %>%
    select(Cat, Count, Prop)
}
```

```
props1 <- map(pre_debate[, type_var == "logical"], freq_tab)
props2 <- map(post_debate[, type_var == "logical"], freq_tab)
```

```
vars <- unlist(map(props1, nrow))
```

```
props_tab1 <- reduce(props1, rbind)
props_tab2 <- reduce(props2, rbind)
```

```
props_tab1 <- props_tab1 %>% mutate(Variable = rep(names(vars),
  vars), Candidate = ifelse(Variable == "BS", "Sanders", ifelse(Variable ==
  "KH", "Harris", ifelse(Variable == "JB", "Biden", ifelse(Variable ==
  "EW", "Warren", ifelse(Variable == "PB", "Buttigieg", ifelse(Variable ==
  "TS", "Steyer", ifelse(Variable == "AY", "Yang", ifelse(Variable ==
  "BC", "Booker", ifelse(Variable == "AK", "Klobuchar", ifelse(Variable ==
  "TG", "Gabbard", NA)))))))))))))
```

```
props_tab2 <- props_tab2 %>% mutate(Variable = rep(names(vars),
  vars), Candidate = ifelse(Variable == "BS", "Sanders", ifelse(Variable ==
  "KH", "Harris", ifelse(Variable == "JB", "Biden", ifelse(Variable ==
  "EW", "Warren", ifelse(Variable == "PB", "Buttigieg", ifelse(Variable ==
  "TS", "Steyer", ifelse(Variable == "AY", "Yang", ifelse(Variable ==
  "BC", "Booker", ifelse(Variable == "AK", "Klobuchar", ifelse(Variable ==
  "TG", "Gabbard", NA)))))))))))))
```

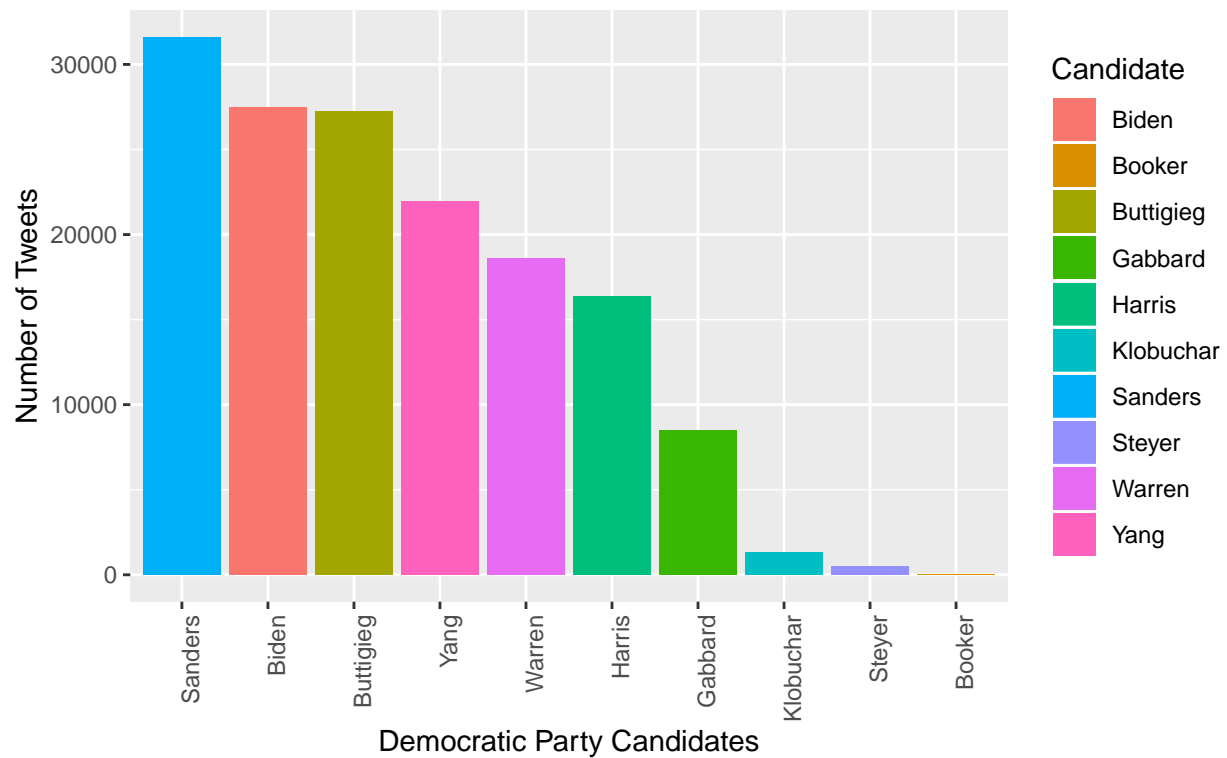
Graphing

A visual of the proportion of tweets pre and post debate

Pre-debate number of tweets

```
props_tab1 %>% filter(Cat == "TRUE") %>% ggplot() + geom_col(mapping = aes(x = reorder(
  -Count), y = Count, fill = Candidate)) + labs(x = "Democratic Party Candidates",
  y = "Number of Tweets", caption = "Source: Twitter
```

```
Figure 1") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Source: Twitter

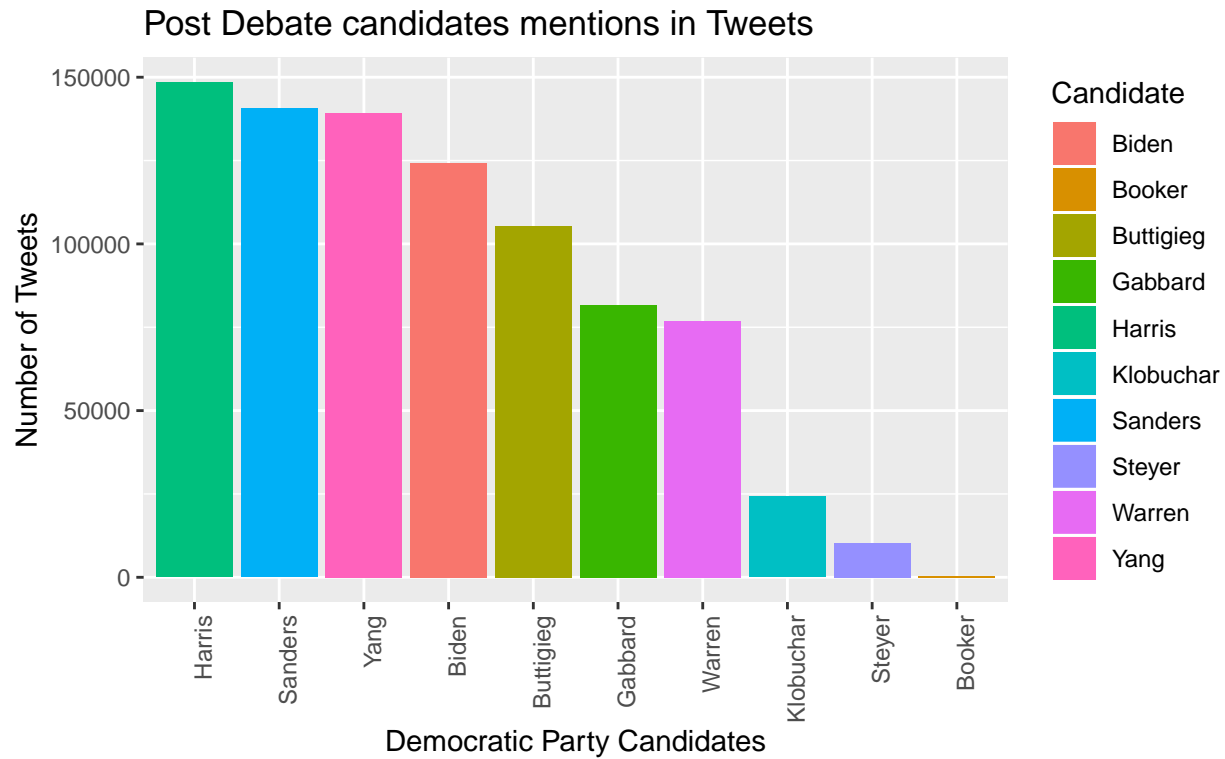
Figure 1

Figure 1 shows the number of tweets that mentioned each candidate before the debate. We noticed that Sanders, Biden and Buttigieg are in the lead with over 25,000 tweets mentions.

Post-debate number of tweets

```
props_tab2 %>% filter(Cat == "TRUE") %>% ggplot() + geom_col(mapping = aes(x = reorder(
  -Count), y = Count, fill = Candidate)) + labs(x = "Democratic Party Candidates",
  y = "Number of Tweets", title = "Post Debate candidates mentions in Tweets",
  caption = "Source: Twitter
```

```
Figure 2") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Source: Twitter

Figure 2

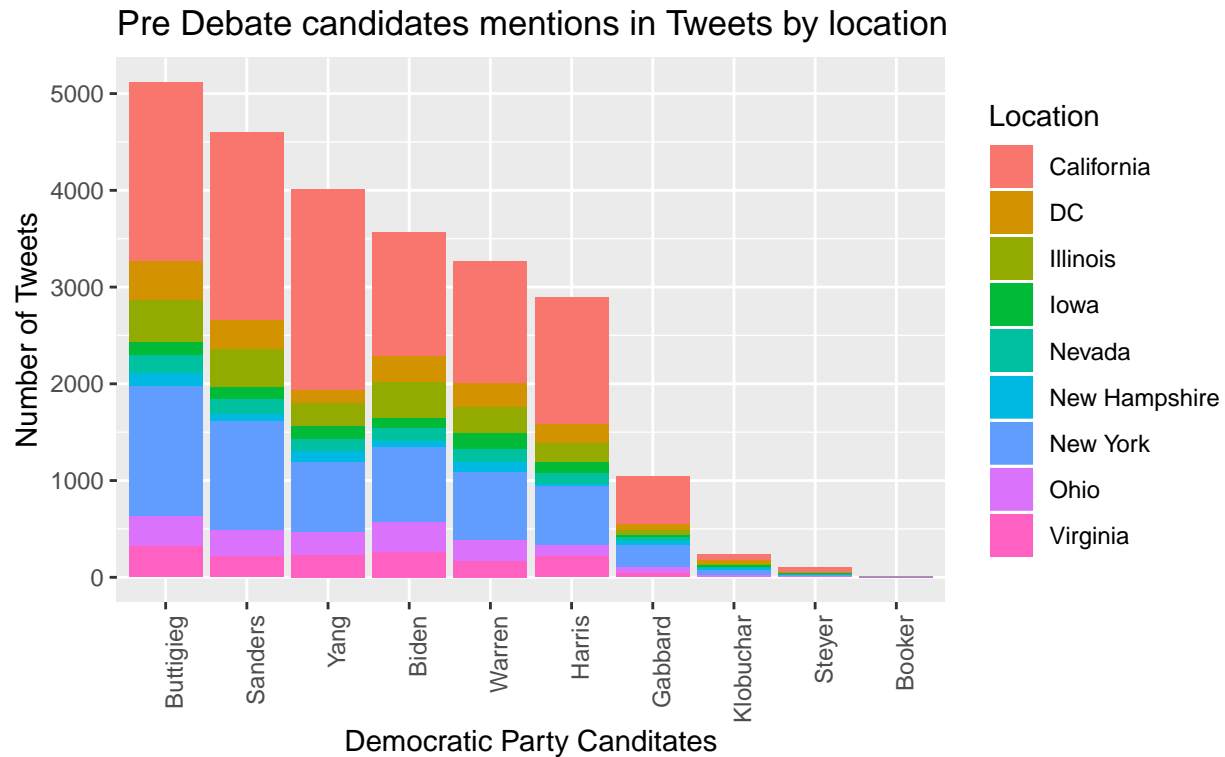
Figure 1: Post-debate Mentions in Tweets, by Candidate

Figure 2 shows the post- debate tweets mentions by candidate. As expected, lot more tweets mentions occurred after the debate, about 5 times more than pre- debate. Here we noticed that the ranking change with Harris and Yang in the lead along with Sanders and tweets mentions close to 150,000.

Tweets location pre-debate

```
loc_pre_all %>% mutate(Location = loc) %>% group_by(Candidate,
  Location) %>% summarise(total = n()) %>% ggplot() + geom_col(mapping = aes(x = reor
    -total), y = total, fill = Location)) + labs(x = "Democratic Party Canditates",
  y = "Number of Tweets", title = "Pre Debate candidates mentions in Tweets by locatio
    caption = "Source: Twitter

    Figure 3") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Source: Twitter

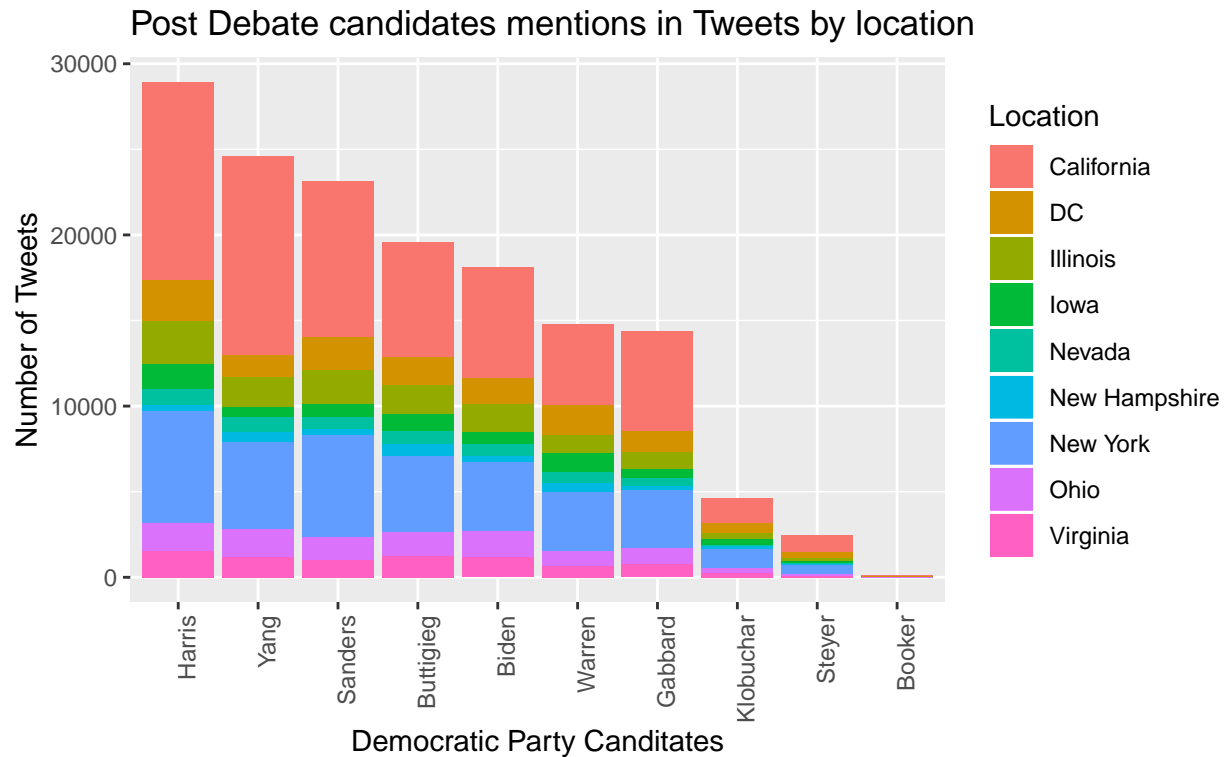
Figure 3

Figure 3 shows the number of tweets mentions by candidate by location pre- debate. Here we notice that Buttigieg, Sanders and Yang get the most tweets mentions and mainly coming from California and New York for all 3 candidates. These results are not surprising with California and New York population presence in Twitter.

Tweets location post-debate

```
loc_post_all %>% mutate(Location = loc) %>% group_by(Candidate,
  Location) %>% summarise(total = n()) %>% ggplot() + geom_col(mapping = aes(x = reord
    -total), y = total, fill = Location)) + labs(x = "Democratic Party Canditates",
  y = "Number of Tweets", title = "Post Debate candidates mentions in Tweets by locati
  caption = "Source: Twitter

  Figure 4") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Source: Twitter

Figure 4

Figure 4 is the post- debate tweets mentions by candidate by location. Just like in Figure 3 (pre-debate analysis) California and New York had the most users tweeting about the candidates. Harris, Yang and Sanders are mostly mentioned in tweets by users.

Google Trends

To explore the Google Trends data, we examined the location and density of searches for each candidate. We created a map showing our results (Kahle and Wickham 2013). Because it was difficult to see the results for each candidate overlaid on one map, we added a facet wrap to show each individual candidate.

```
# get map
us <- c(left = -125, bottom = 25.75, right = -67, top = 49)
us_map <- get_stamenmap(us, zoom = 5, maptype = "toner-lite")
ggmap(us_map)

# clean data for mapping
trends_map <- gc_locations %>% group_by(keyword, location) %>%
  mutate(Hits = sum(hits), Candidate = keyword)

# generate map
ggmap(us_map) + geom_point(data = trends_map, aes(x = lon, y = lat,
```

```
size = Hits, color = Candidate), alpha = 0.2) + theme(axis.line = element_blank(),
axis.text.x = element_blank(), axis.text.y = element_blank(),
axis.ticks = element_blank(), axis.title.x = element_blank(),
axis.title.y = element_blank()) + facet_wrap("Candidate",
shrink = FALSE)
```

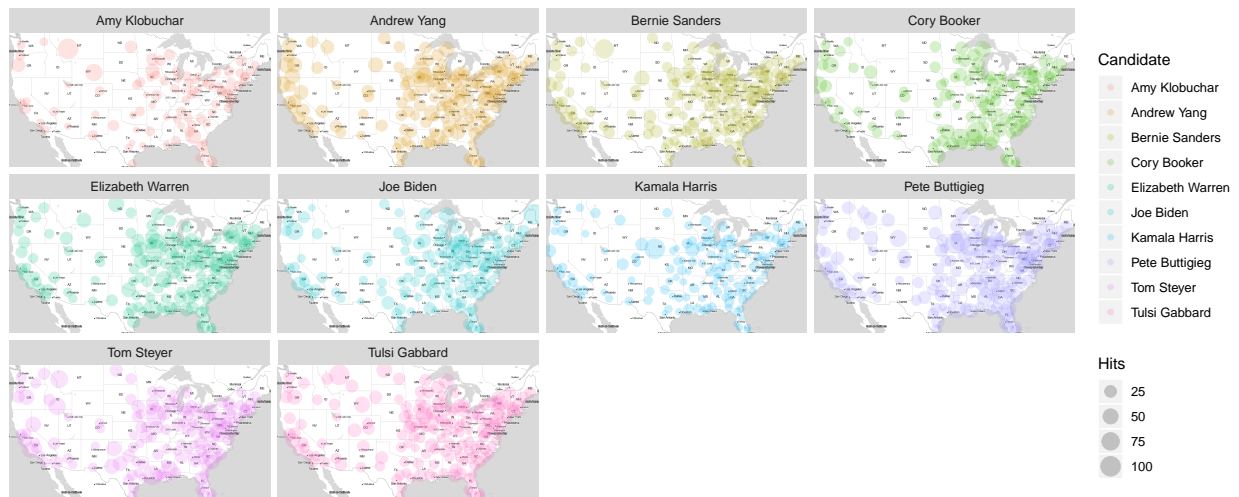


Figure 2: Map of Google Trends Hits, by Candidate

These maps show that candidates have different distributions and density of searches. Searches are more common near the coasts, and less common in the middle of the country. We also explored the number of hits pre- and post- debate for each candidate.

```
interest_over_time_all$Pre_post <- factor(interest_over_time_all$Pre_post,
  levels = c("Pre-debate", "Post-debate"))

interest_over_time_all %>% group_by(keyword, Pre_post) %>% summarise(avg_hits = mean(hi
  ggplot() + geom_col(mapping = aes(x = reorder(keyword, -avg_hits),
  y = avg_hits, fill = Pre_post), color = "black", position = "dodge") +
  xlab("Candidate") + ylab("Hits") + scale_fill_discrete(name = "Timeframe") +
  theme(axis.text.x = element_text(angle = 35))
```

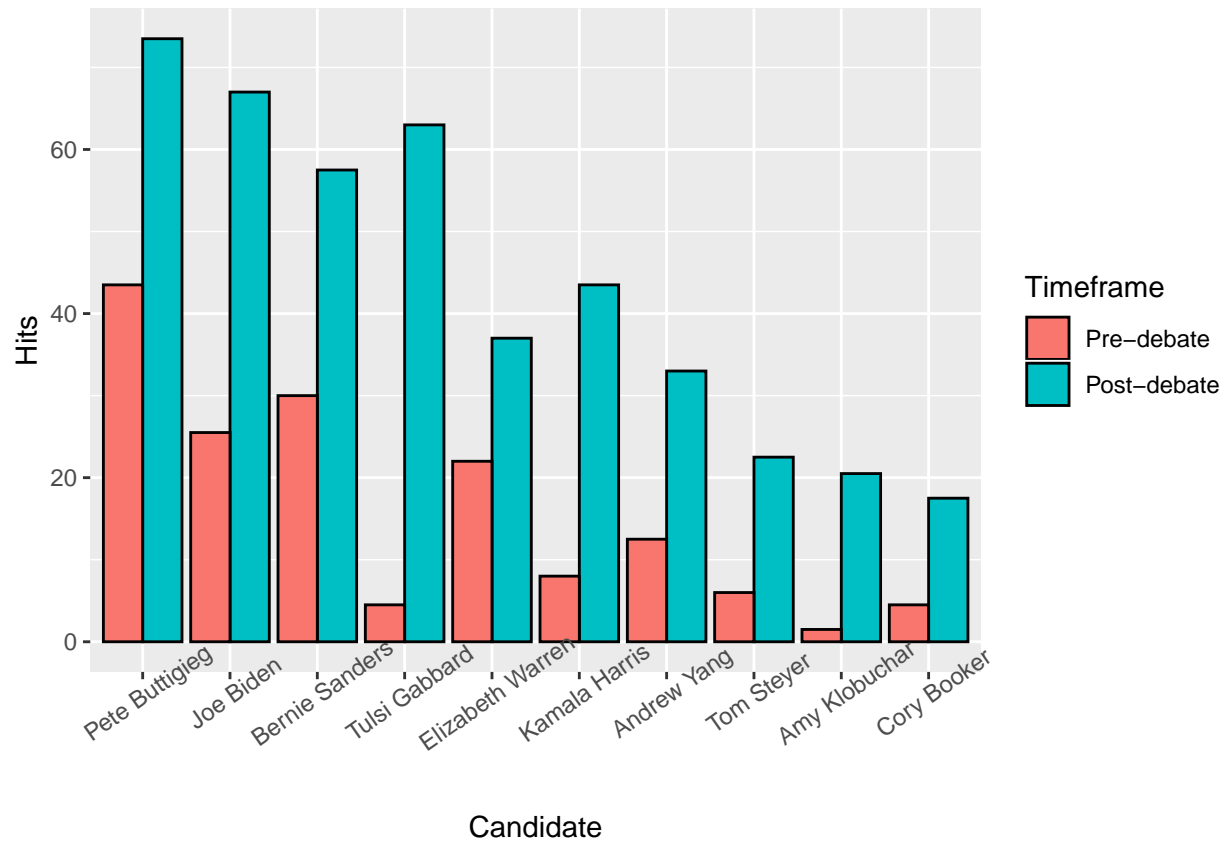


Figure 3: Pre- and Post-debate Google Trends Data, by Candidate

All candidates had higher search counts after the debate than before. This makes sense, because after a debate there is an increase in the number of news stories about candidates, which may lead to increased interest and thus, increased search activity. Additionally, users may be searching Google to learn about a candidate's debate performance if they did not watch the debate live.

Polling Data

For the polling data, we looked at the pre- and post-debate polling numbers for each candidate.

```
polling_data$Pre_post <- factor(polling_data$Pre_post, levels = c("Pre-debate",
  "Post-debate"))

polling_data %>% group_by(Candidate, Pre_post) %>% filter(Candidate !=
  "Bennet", Candidate != "Bloomberg") %>% ggplot() + geom_col(mapping = aes(x = reord
  -Percentage), y = Percentage, fill = Pre_post), color = "black",
  position = "dodge") + xlab("Candidate") + ylab("Percentage") +
  scale_fill_discrete(name = "Timeframe") + theme(axis.text.x = element_text(angle =
```

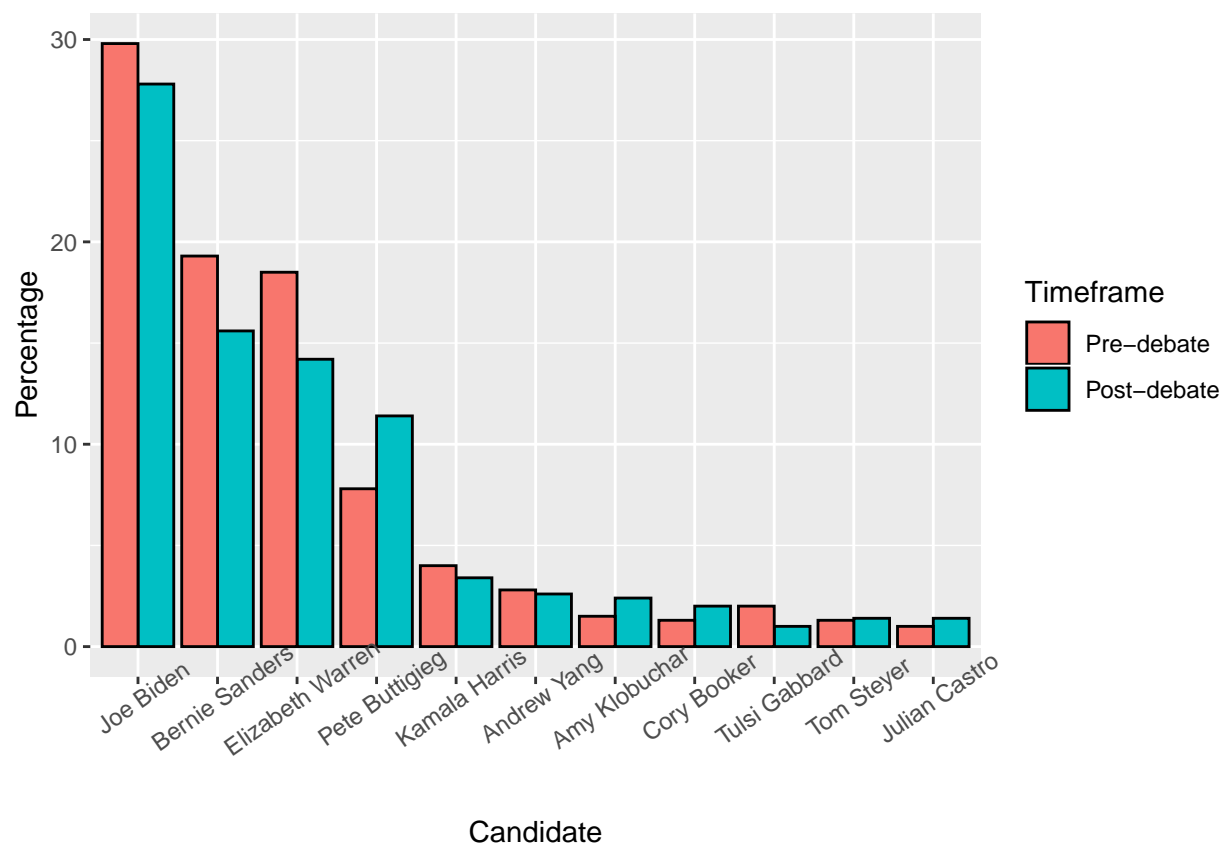


Figure 4: Pre- and Post-debate Polling Data, by Candidate

Analysis

We created a Shiny app that allows us to compare the polling, Google Trends, and Twitter sentiment data (Chang et al. 2019).

First, we cleaned up and combined the three data sources for use in the Shiny app.


```

# clean data
gtrends <- interest_over_time_all %>% mutate(Candidate = keyword,
  Data = "GTrends") %>% filter(Candidate %in% c("Joe Biden",
  "Bernie Sanders", "Elizabeth Warren", "Kamala Harris", "Pete Buttigieg")) %>%
  group_by(Candidate, Pre_post, Data) %>% arrange(Candidate,
  Pre_post) %>% summarise(Percentage = mean(hits))

poll <- polling_data %>% select(Candidate, Pre_post, Percentage) %>%
  filter(Candidate %in% c("Joe Biden", "Bernie Sanders", "Elizabeth Warren",
  "Kamala Harris", "Pete Buttigieg")) %>% mutate(Data = "Polling") %>%
  arrange(Candidate, Pre_post) %>% group_by(Candidate, Pre_post,
  Data)

twitter_data <- data.frame(Candidate = c("Joe Biden", "Bernie Sanders",
  "Elizabeth Warren", "Kamala Harris", "Pete Buttigieg", "Joe Biden",
  "Bernie Sanders", "Elizabeth Warren", "Kamala Harris", "Pete Buttigieg"),
  Pre_post = c("Pre-debate", "Pre-debate", "Pre-debate", "Pre-debate",
  "Pre-debate", "Post-debate", "Post-debate", "Post-debate",
  "Post-debate", "Post-debate"), Data = c("Twitter", "Twitter",
  "Twitter", "Twitter", "Twitter",
  "Twitter", "Twitter", "Twitter"), Percentage = c(pre_sent_pos$Positive[1],
  pre_sent_pos$Positive[2], pre_sent_pos$Positive[3], pre_sent_pos$Positive[4],
  pre_sent_pos$Positive[5], post_sent_pos$Positive[1],
  post_sent_pos$Positive[2], post_sent_pos$Positive[3],
  post_sent_pos$Positive[4], post_sent_pos$Positive[5]))

twitter <- as_tibble(twitter_data)
twitter <- twitter %>% group_by(Candidate, Pre_post, Data) %>%
  arrange(Candidate, desc(Pre_post)) %>% mutate(Percentage = Percentage *
  100)

all_data <- rbind(poll, gtrends, twitter)

all_data$Pre_post <- factor(all_data$Pre_post, levels = c("Pre-debate",
  "Post-debate"))

```

Then, we created a shiny app, which can be viewed at <https://znpadgett.shinyapps.io/surv727app/>:

```

# Define UI

ui <- fluidPage(

  # Application title

```

```

titlePanel("2020 Democratic Primary Candidate Data"),

# Sidebar with a dropdown
sidebarLayout(
  sidebarPanel(
    selectInput(inputId = "Candidate",
               label = "Candidate",
               choices = c("Joe Biden", "Pete Buttigieg", "Kamala Harris",
                           "Bernie Sanders", "Elizabeth Warren"),
               selected = "Joe Biden"),
    selectInput(inputId = "Data",
               label = "Data Type",
               choices = c("Polling", "GTrends", "Twitter"),
               selected = "Polling")
  ),

  # Show plot
  mainPanel(
    plotOutput(outputId = "graph")
  )
)

# Define server logic
server <- function(input, output) {

  output$graph <- renderPlot({
    all_data %>%
      filter(Candidate == input$Candidate, Data == input$Data) %>%
      ggplot() +
      geom_col(mapping = aes(x=Pre_post, y=Percentage, fill=input$Candidate)) +
      ylim(0,100) +
      xlab("Timeframe") + ylab("Percentage") +
      theme(legend.position = "none")
  })
}

# Run the application
shinyApp(ui = ui, server = server)

```

Finally, we examined the relationships between the polling data and the Twitter and Google Trends data post-debate, as well as the relationships between pre- and post-debate changes across data sources. We looked at these results for the 5 most popular candidates.

```
graphing_data <- cbind(twitter, poll, gtrends)
graphing_data %>% group_by(Pre_post, Candidate) %>% filter(Pre_post ==
  "Post-debate") %>% ggplot() + geom_point(mapping = aes(x = Percentage,
  y = Percentage1, color = Candidate), size = 4) + xlim(0,
  20) + ylim(0, 30) + labs(x = "Twitter Sentiment", y = "Polling Data")
```

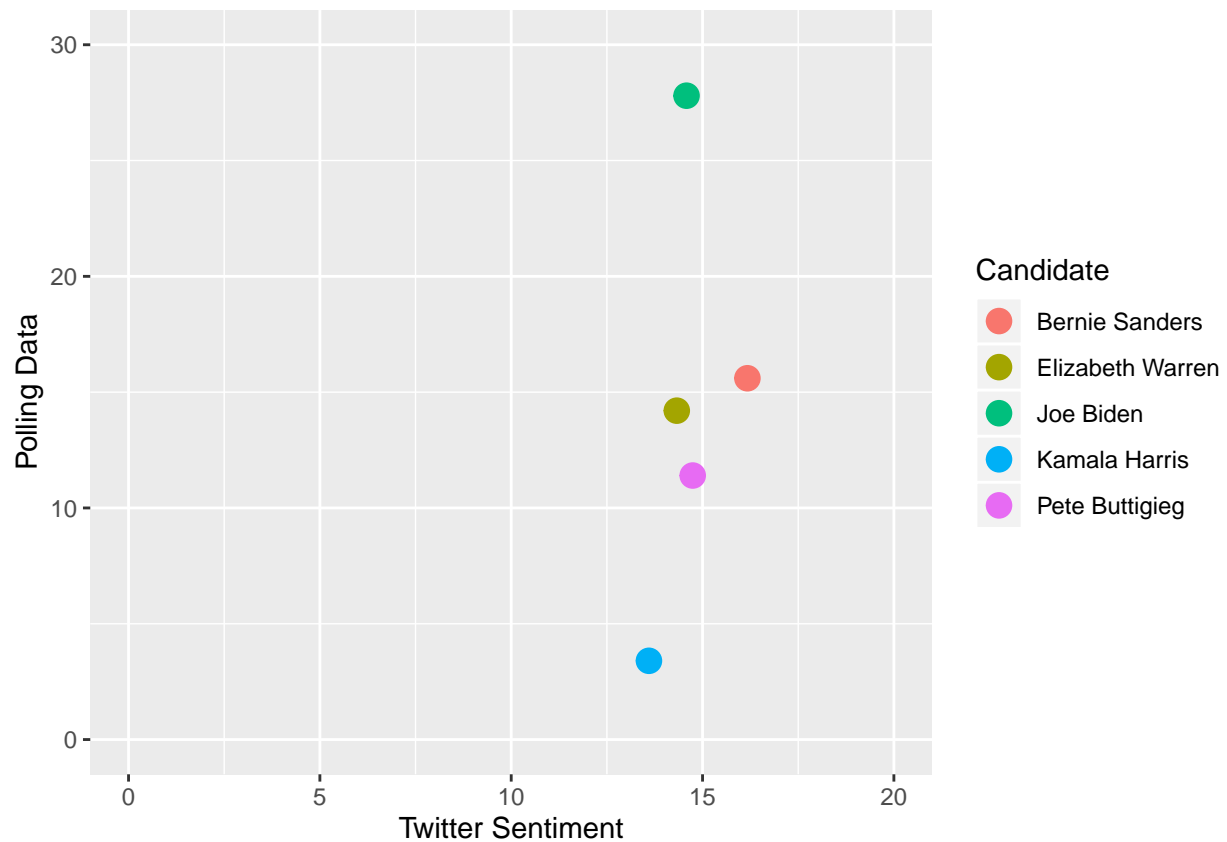


Figure 5: Comparison of Post-debate Twitter and Polling Data

Figure 5 shows that there is no clear relationship between the post-debate twitter sentiment and the polling data. All candidates have similar sentiment scores, regardless of polling percentage.

```
graphing_data %>% group_by(Pre_post, Candidate) %>% filter(Pre_post ==
  "Post-debate") %>% ggplot() + geom_point(mapping = aes(x = Percentage2,
  y = Percentage1, color = Candidate), size = 4) + xlim(0,
  100) + ylim(0, 30) + labs(x = "Google Trends Data", y = "Polling Data")
```

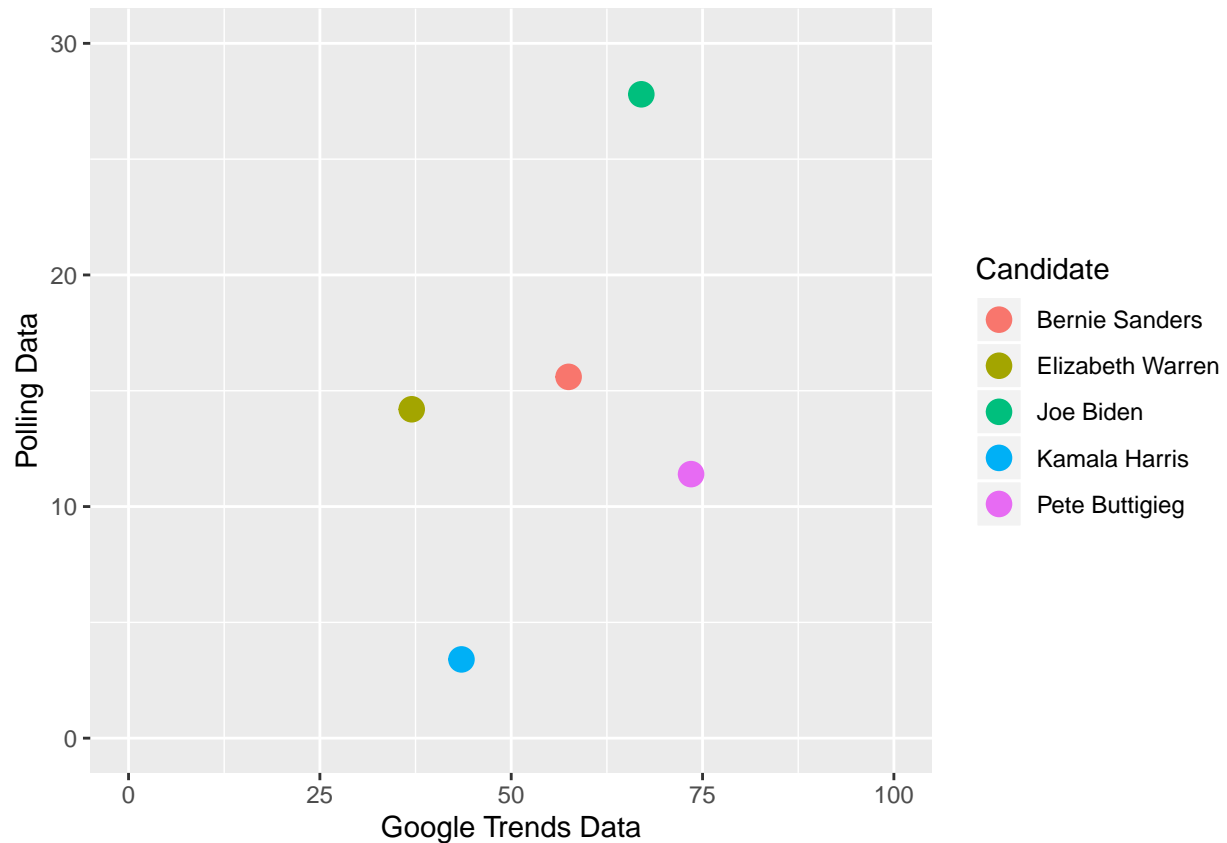


Figure 6: Comparison of Post-debate Google Trends and Polling Data

In Figure 6, we can see a very slight relationship between the post-debate Google Trends data and polling data, in that candidates polling higher generally had higher Google trends hit rates.

```
pre <- graphing_data %>% select(Pre_post, Candidate, Percentage,
  Percentage1, Percentage2) %>% group_by(Pre_post, Candidate,
  Data) %>% filter(Pre_post == "Pre-debate") %>% mutate(Pre_twitter = Percentage,
  Pre_polling = Percentage1, Pre_gtrends = Percentage2)

## Adding missing grouping variables: `Data`

post <- graphing_data %>% select(Pre_post, Candidate, Percentage,
  Percentage1, Percentage2) %>% group_by(Pre_post, Candidate,
  Data) %>% filter(Pre_post == "Post-debate") %>% mutate(Post_twitter = Percentage,
  Post_polling = Percentage1, Post_gtrends = Percentage2)

## Adding missing grouping variables: `Data`
```

```
change_data <- cbind(pre, post)

change_data %>% group_by(Pre_post, Candidate) %>% mutate(change_twitter = ((Post_twitter -
  Pre_twitter)/Pre_twitter) * 100, change_polling = ((Post_polling -
  Pre_polling)/Pre_polling) * 100, change_gtrends = ((Post_gtrends -
  Pre_gtrends)/Pre_gtrends) * 100) %>% ggplot() + geom_point(mapping = aes(x = change_
y = change_polling, color = Candidate), size = 4) + labs(x = "Percent Change in Twit
y = "Percent Change in Polling Data")
```

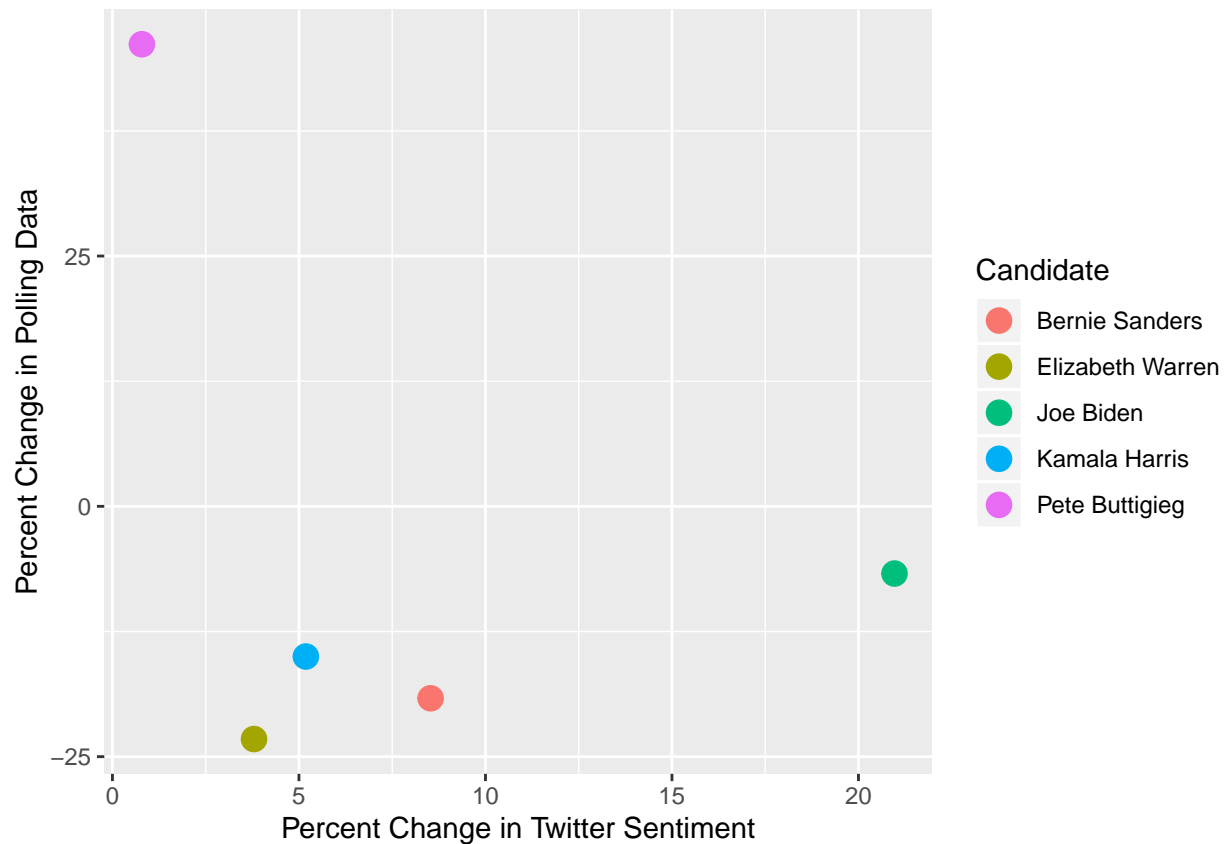


Figure 7: Comparison of Changes in Twitter and Polling Data

Figure 7 shows the percent change in Twitter sentiment and percent change in vote share between the two time periods (pre- and post-debate). There is no clear relationship between the change in vote share and the change in Twitter sentiment.

```
change_data %>% group_by(Pre_post, Candidate) %>% mutate(change_twitter = ((Post_twitter -
  Pre_twitter)/Pre_twitter) * 100, change_polling = ((Post_polling -
  Pre_polling)/Pre_polling) * 100, change_gtrends = ((Post_gtrends -
  Pre_gtrends)/Pre_gtrends) * 100) %>% ggplot() + geom_point(mapping = aes(x = change_
```

```
y = change_polling, color = Candidate), size = 4) + labs(x = "Percent Change in Goog  
y = "Percent Change in Polling Data")
```

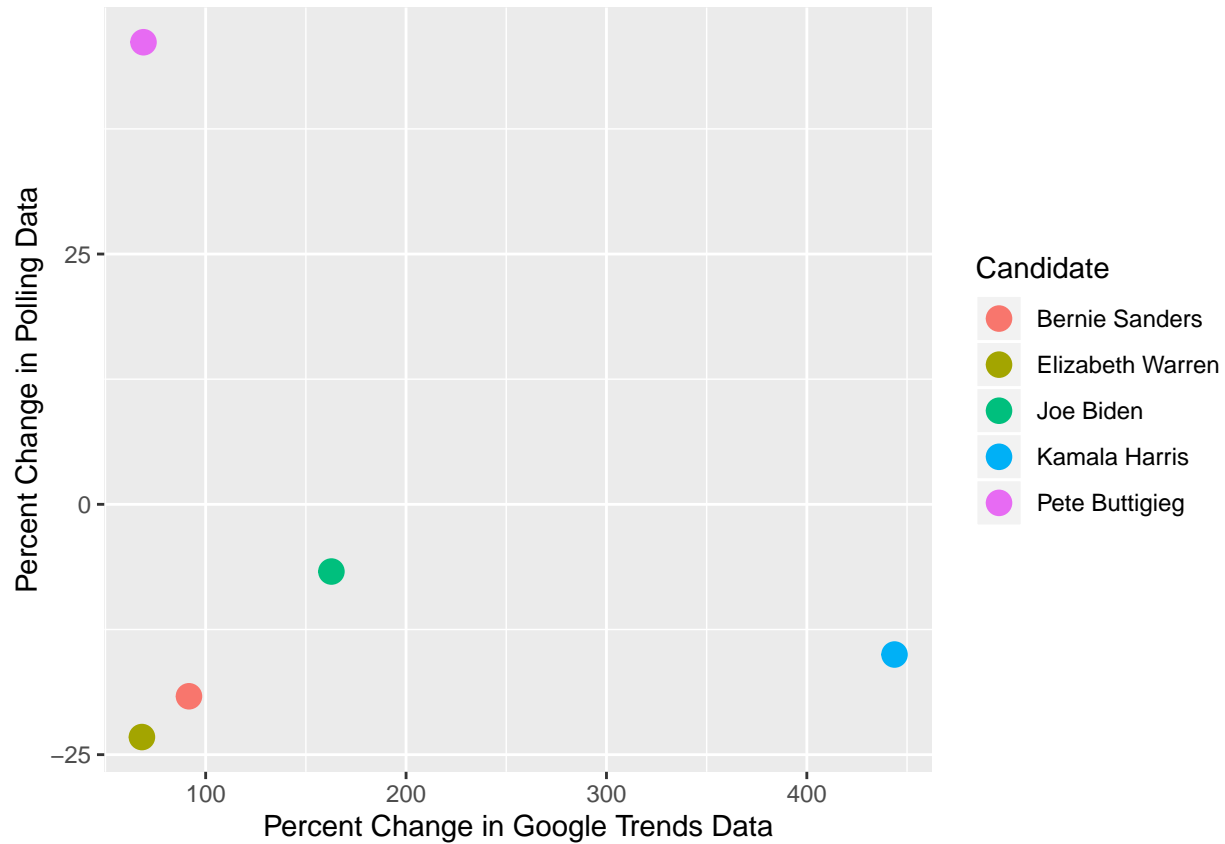


Figure 8: Comparison of Changes in Google Trends and Polling Data

Figure 8 shows the percent change in Google Trends hit rate and percent change in vote share between the two time periods. Again, there is no clear relationship between the change in the two data sources.

In conclusion, when looking at the pre- and post-debate data from each source side-by-side, we can see that, while candidates who are polling higher have higher numbers of tweets and Google searches (for the most part), there is no clear relationship between the post-debate changes in polling numbers and the post-debate changes in Twitter and Google Trends data. Regardless of polling numbers, all candidates had higher Twitter positivity ratings and higher numbers of Google searches post-debate than pre-debate.

Discussion

Our results show that neither Twitter nor Google Trends data align with pre- and post-debate changes in polling numbers. There were no discernable patterns that relate the found

data to the polling data. There may be many reasons why the found data did not align with the polling data. For one, the found data are not representative of U.S. voters, only Twitter users or Google searches, and the lack of availability of demographic data does not allow for weighting adjustments. Variable-to-variable comparison between two data sources may not be realistic. For example, a large number of searches and higher positivity of tweets may not actually mean that more people will vote for a candidate.

Additionally, there are many practical difficulties in using the found data, for example that the location field in the Twitter data corresponds to self-reported location, an open-ended field. Many users give incorrect or fake locations.

Overall, we don't think that this data should be used in place of survey data when it comes to election polling. However, this data could still be useful for candidates in understanding how social media users perceive them before and after a debate, or in understanding their search popularity. It is important to recognize that these measures are not an accurate proxy for voter opinion, but they could still be useful in their own right. Campaigns would also need to recognize that the target population of any inferences may on these data is not U.S. voters, or even all those living in the U.S., but is actually the population of tweets (for the Twitter data) or Google searches (for the Google Trends data). When these considerations are taken into account, we still think that this data could be useful for campaign strategy.

It would be advantageous to survey methodologists to be able to use sources such as Twitter and Google Trends to replace survey data. However, we need to acknowledge the limitations preventing us from doing so. We all will agree that Twitter users do not adequately represent the population consequently introducing coverage error. Validity and measurement error just to name a few limitations we encountered with Twitter data. In lieu of replacing survey data, Twitter data can and should be used as a supplemental data. It is also important to note that we collected data for a short period of time. We believed that Twitter data can be much more useful if we had a longer timeframe. With that we would be able to perform more in depth analysis with time a key variable or even fit models. For instance, the next democratic debate is scheduled for December 19th, it would be interesting to see how the sentiment differ from the November 21st debate. On that point, candidates could use Twitter data for many other purposes such as perception in the Twitter world or change over time in sentiment but definitely not for election outcomes.

References

- Beauchamp, N. 2017. "Predicting and Interpolating State-Level Polls Using Twitter Textual Data." *American Journal of Political Science* 61 (2).
- Chang, W., J. Cheng, J. Allaire, Y. Xie, and J. McPherson. 2019. "Shiny: Web Application Framework for R."
- Feuerriegel, S., and N. Proellocks. 2019. "SentimentAnalysis: Dictionary-Based Sentiment Analysis."
- Gaurav, M., A. Srivastava, A. Kumar, and S. Miller. 2013. "Leveraging Candidate Popularity on Twitter to Predict Election Outcome." *Proceedings of the 7th Workshop on Social*

Network Mining and Analysis.

Gayo-Avello, D. 2013. “A Meta-Analysis of State-of-the-Art Electoral Prediction from Twitter Data.” *Organization Studies* 31 (6): 211–48.

Kahle, D., and H. Wickham. 2013. “Ggmap: Spatial Visualization with Ggplot2.” *The R Journal* 5 (1): 144–61.

Kassraie, P., A. Modirshanechi, and H. Aghajan. 2017. “Election Vote Share Prediction Using a Sentiment-Based Fusion of Twitter Data with Google Trends and Online Polls.” *Proceedings of the 6th International Conference on Data Science, Technology and Applications.*

Massicotte, P., and D. Eddelbuettel. 2019. “gtrendsR: Perform and Display Google Trends Queries.”

Salunkhe, P., and S. Deshmukh. 2017. “Twitter Based Election Prediction and Analysis.” *International Research Journal of Engineering and Technology* 4 (10).