

Data Cleaning & Sentiment Score

Minyoung Do

March 07, 2020

Data Prep

Read tweets from the files & Find the overlapping time frame

```
# Read data from csv files
bernie_original <- read_csv("bernie_tweets.csv")
warren_original <- read_csv("warren_tweets.csv")
pete_original <- read_csv("buttigieg_tweets.csv")
bloomberg_original <- read_csv("bloomberg_tweets_updated.csv")
amy_original <- read_csv("klobuchar_tweets.csv")

# Convert ID to character from numeric.
bernie_original$user_id <- as.character(bernie_original$user_id)
warren_original$user_id <- as.character(warren_original$user_id)
pete_original$user_id <- as.character(pete_original$user_id)
bloomberg_original$user_id <- as.character(bloomberg_original$user_id)
amy_original$user_id <- as.character(amy_original$user_id)

# check the time frame of each data
date_check <- function(df){
  df %>%
    group_by(1) %>%
    summarise(max = max(created), min = min(created))
}

# apply the new function to each data frame
bernie_date <- date_check(bernie_original)
warren_date <- date_check(warren_original)
pete_date <- date_check(pete_original)
bloomberg_date <- date_check(bloomberg_original)
amy_date <- date_check(amy_original)

# find overlapping time period
date_frame <- bind_rows(bernie_date, warren_date, pete_date, bloomberg_date, amy_date, .id = "id")[, -2]
```

Preparing/cleaning the data

The first step of data cleaning process entails removing:

1. *urls*
2. character strings between “<” and “>” to deal with smileys and other encoded text.
3. retweet marks, @RT
4. quotation marks and apostrophes
5. any @userid
6. punctuation and blank spaces
7. stopwords and single letters

```
clean_tweets <- function(df) {  
  # Remove URLs  
  df$text <- gsub("http[[:space:]]*", "", df$text)  
  # Remove retweet entities  
  df$text <- gsub("(RT|via)((?:\\b\\W*@\\w+)+)", " ", df$text)  
  # Remove quotes  
  df$text <- gsub("'s'|'s|'", "", df$text)  
  # Remove at people  
  df$text <- gsub("@\\w+", " ", df$text)  
  # Remove punctuation  
  df$text <- gsub("[[:punct:]]", " ", df$text)  
  # Remove single letters.  
  df$text <- gsub("*\\b[[:alpha:]]{1}\\b *", "", df$text)  
  # Remove unnecessary spaces  
  df$text <- gsub("[ \\t]{2,}", " ", df$text)  
  # Remove leading and trailing whitespaces  
  df$text <- gsub("^\\s+|\\s+$", "", df$text)  
  
  ## parsing, tokenizing, and re-grouping text column  
  df <- df %>%  
    unnest_tokens(output = word, input = text) %>%  
    anti_join(stop_words) %>%  
    filter(!str_detect(word, "^[0-9]*$")) %>%  
    group_by(username, created) %>%  
    summarize(text = str_c(word, collapse = " ")) %>%  
    ungroup()  
  # creating a date column without time  
  df$date <- as.Date(df$created, "%Y-%m-%d")  
  # setting date objects  
  start <- as.Date("2020-01-02")  
  end <- as.Date("2020-02-26")  
  # subsetting by date range  
  df <- df %>%  
    subset(date >= start & date <= end) %>%  
    select(-created)  
}
```

```
bernie_clean <- clean_tweets(bernie_original)
warren_clean <- clean_tweets(warren_original)
pete_clean <- clean_tweets(pete_original)
bloomberg_clean <- clean_tweets(bloomberg_original)
amy_clean <- clean_tweets(amy_original)
```

Check for duplicates (Sample Data: Bernie Sanders)

```
# creating a data frame only containing the text column of the clean data
bernie_text <- as.tibble(bernie_clean$text)

# groupingn by frequency of the same texts/words
grouped_bernie_text <- aggregate(bernie_text, by = list(bernie_clean$text), FUN = length);
colnames(grouped_bernie_text) <- c("Text", "TweetCount")

# reordering by frequency rate
grouped_bernie_text <- arrange(grouped_bernie_text, desc(TweetCount))

# duplicated words across the tweets
```

```

bernie_text_duplicates <- subset(grouped_bernie_text, grouped_bernie_text$TweetCount > 1)

# un-comment the code below to take a look
# head(bernie_text_duplicates, n = 20)

# finding any tweet that contains 4 words below as it's likely to be a meaningless tweet
# check Tweetcount for how many times they appear in the dataset
duplicates_bernie <- bernie_text_duplicates[grep("stock|retweet|follow|update", bernie_text_duplicates$Text)]

```

Still looking for a way to remove the duplicates in the data in an efficient manner.

Term Frequency

```

clean_frequency <- function(df){
  term_freq_df <- df %>%
    unnest_tokens(output = word, input = text) %>%
    anti_join(stop_words) %>%
    count(word, sort = TRUE) %>%
    arrange(desc(n)) %>%
    drop_na()

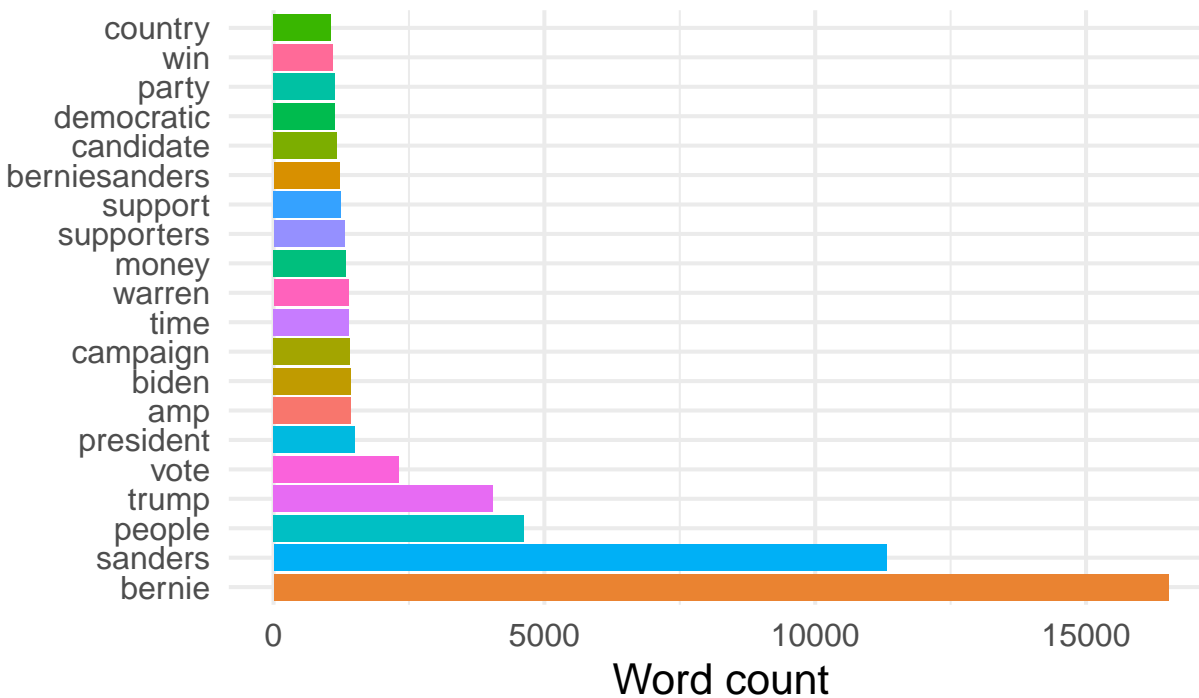
  term_freq_df %>%
  top_n(20) %>%
  ggplot(aes(x = reorder(word, -n), y = n, fill = word)) +
    geom_bar(stat = "identity") +
    scale_x_reordered() +
    labs(x = NULL,
         y = "Word count") +
    coord_flip() +
    theme_minimal(base_size = rcfss::base_size * .65) +
    theme(legend.position = "none")
}

clean_frequency(bernie_clean) +
  labs(title = "Word Frequency",
       subtitle = "Tweets Containing Bernie Sanders",
       x = NULL,
       y = "Word count")

```

Word Frequency

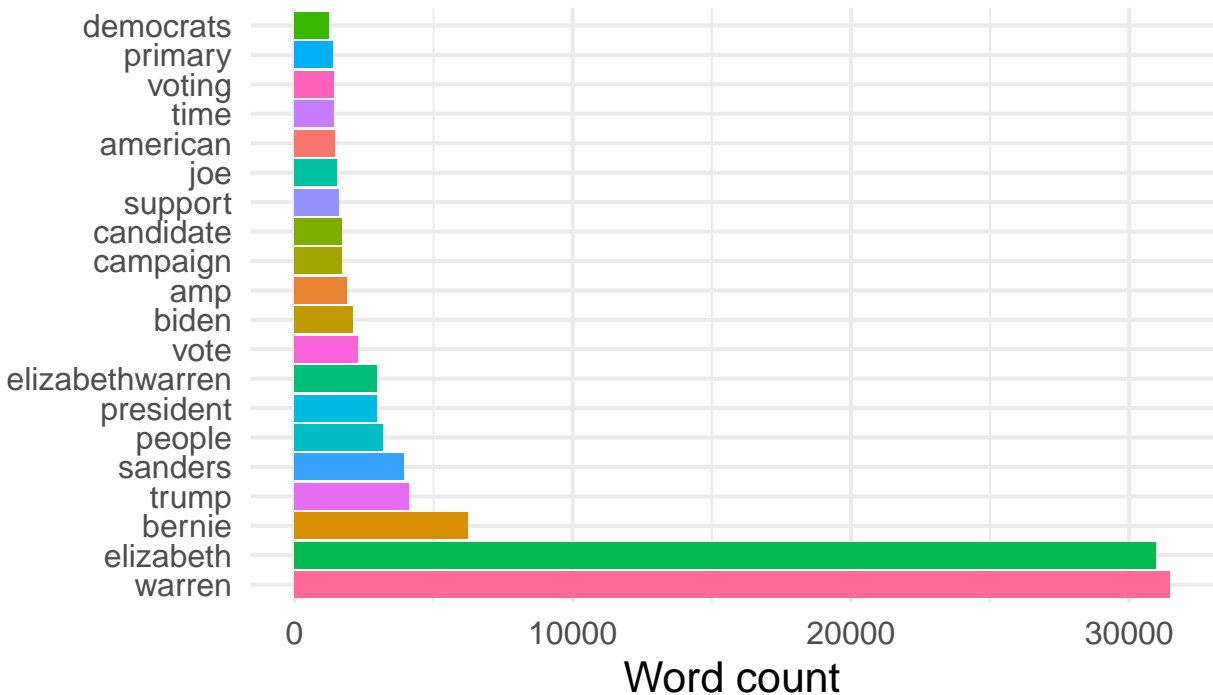
Tweets Containing Bernie Sanders



```
clean_frequency(warren_clean) +  
  labs(title = "Word Frequency",  
        subtitle = "Tweets Containing Elizabeth Warren",  
        x = NULL,  
        y = "Word count")
```

Word Frequency

Tweets Containing Elizabeth Warren

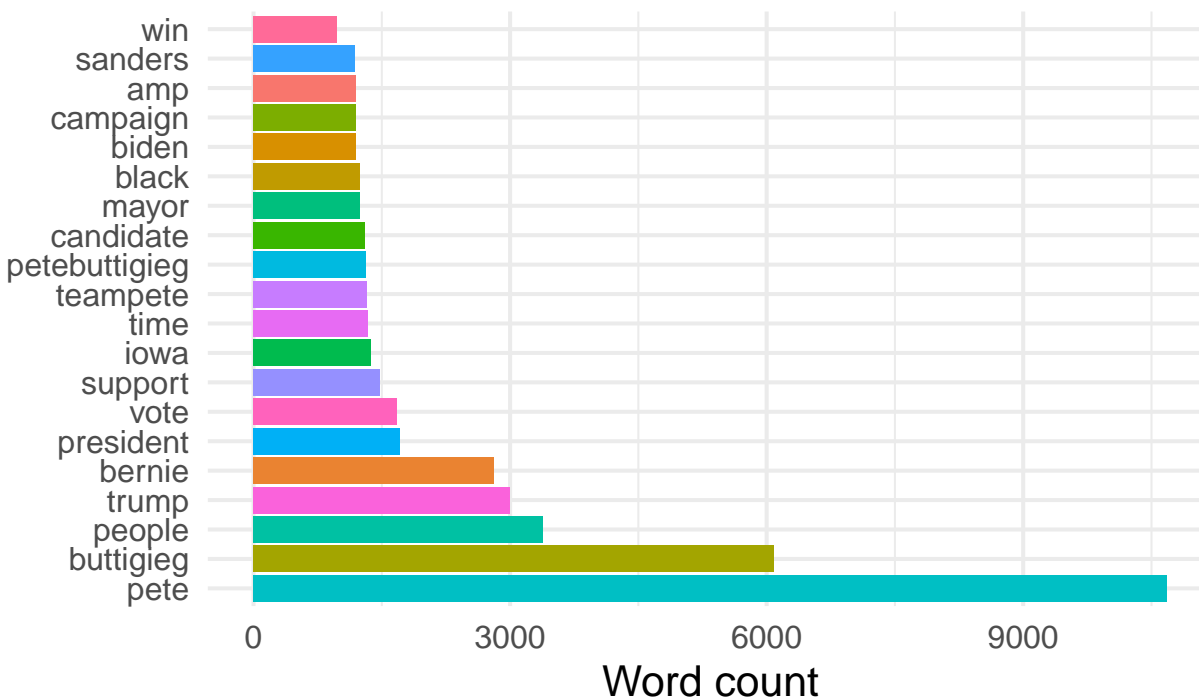


```
clean_frequency(pete_clean) +  
  labs(title = "Word Frequency",  
        subtitle = "Tweets Containing Pete Buttigieg",
```

```
x = NULL,  
y = "Word count")
```

Word Frequency

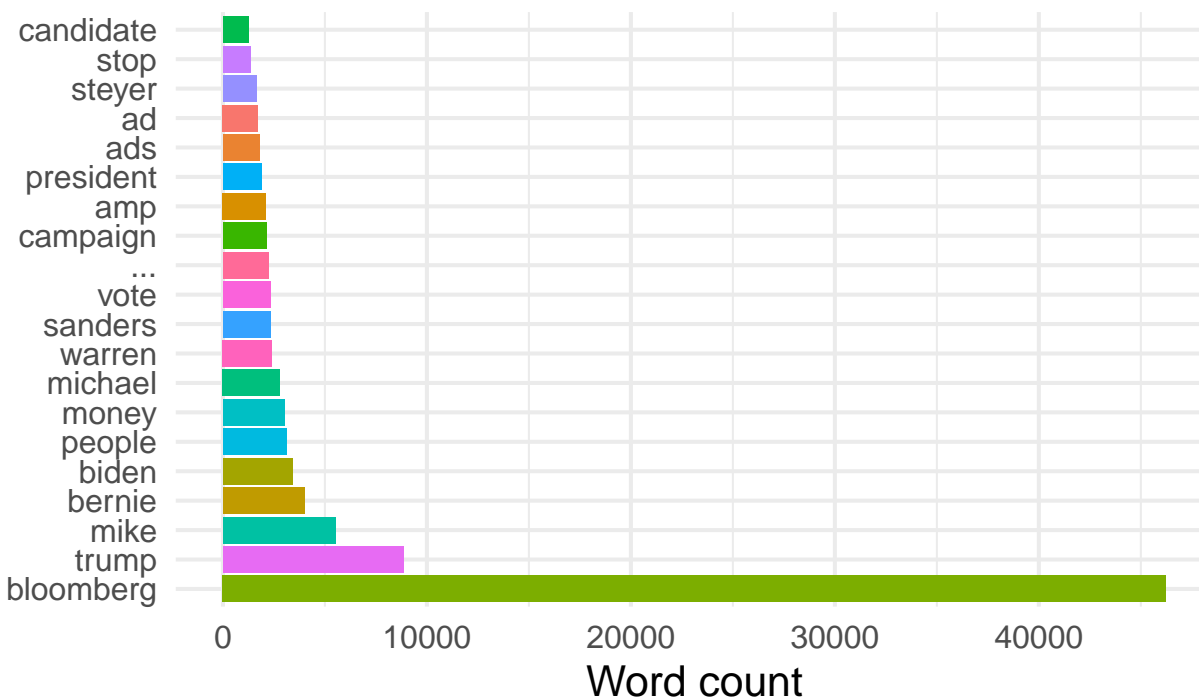
Tweets Containing Pete Buttigieg



```
clean_frequency(bloomberg_clean) +  
  labs(title = "Word Frequency",  
        subtitle = "Tweets Containing Mike Bloomberg",  
        x = NULL,  
        y = "Word count")
```

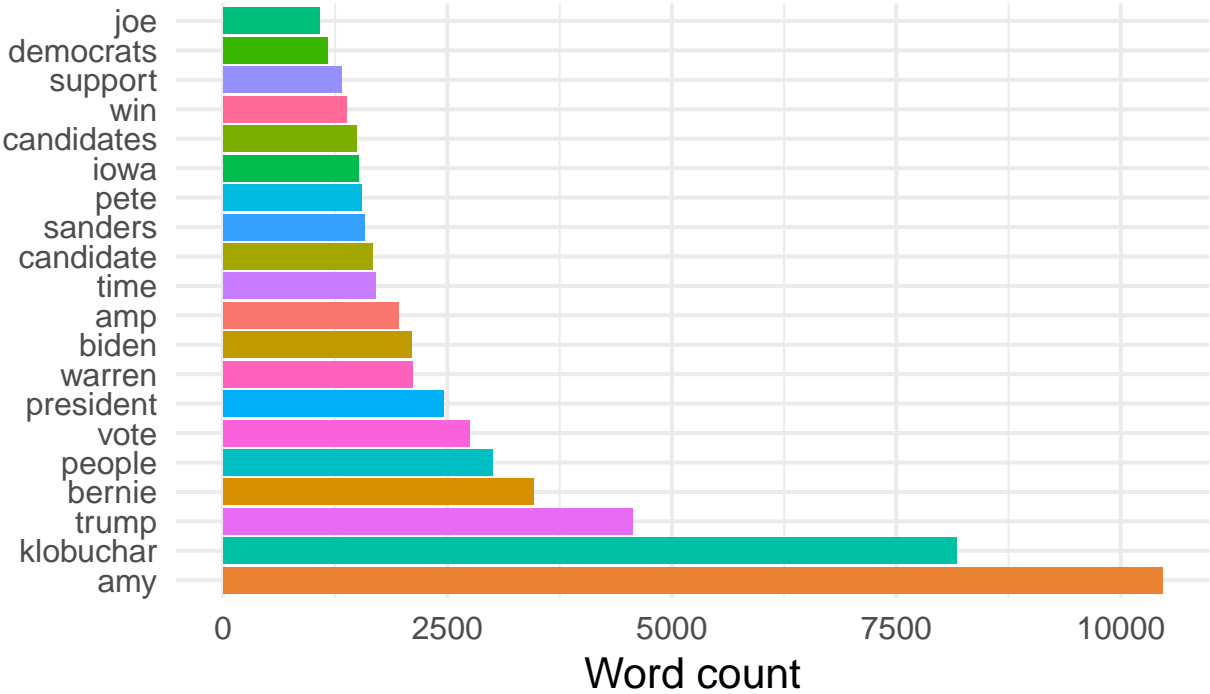
Word Frequency

Tweets Containing Mike Bloomberg



```
clean_frequency(amy_clean) +
  labs(title = "Word Frequency",
        subtitle = "Tweets Containing Amy Klobuchar",
        x = NULL,
        y = "Word count")
```

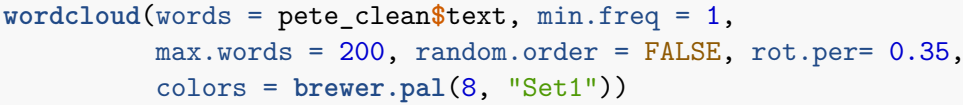
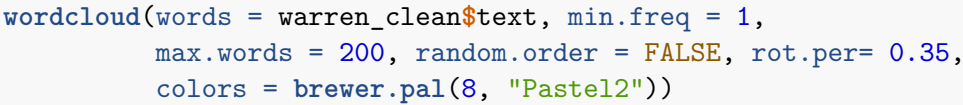
Word Frequency
Tweets Containing Amy Klobuchar



Word Cloud

```
set.seed(1234)

wordcloud(words = bernie_clean$text, min.freq = 1,
          max.words = 200, random.order = FALSE, rot.per= 0.35,
          colors = brewer.pal(8, "Pastel1"))
```



Get sentiment scores

```
# function to calculate the sentiment score
sentiment_score <- function(df) {
  df %>%
    unnest %>%
    get_sentences() %>%
    # get sentiment score for each tweet
    sentiment() %>%
    mutate(characters = nchar(stripWhitespace(text))) %>%
    filter(characters > 1)
    # same number of obs, hence no error
}

# get sentiment scores for all our data frames
bernie_sent <- sentiment_score(bernie_clean)
warren_sent <- sentiment_score(warren_clean)
pete_sent <- sentiment_score(pete_clean)
bloomberg_sent <- sentiment_score(bloomberg_clean)
amy_sent <- sentiment_score(amy_clean)

# quick summary of the result
skim(bernie_sent$sentiment)
```

Table 1: Data summary

Name	bernie_sent\$sentiment
Number of rows	51567
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	-0.03	0.35	-2.3	-0.22	0	0.17	2.87	

```
skim(warren_sent$sentiment)
```

Table 3: Data summary

Name	warren_sent\$sentiment
Number of rows	52736
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	-0.03	0.34	-2.76	-0.22	0	0.18	1.88	

```
skim(pete_sent$sentiment)
```

Table 5: Data summary

Name	pete_sent\$sentiment
Number of rows	50601
Number of columns	1
Column type frequency: numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	0.01	0.36	-2.65	-0.15	0	0.21	1.96	

```
skim(bloomberg_sent$sentiment)
```

Table 7: Data summary

Name	bloomberg_sent\$sentiment
Number of rows	55134
Number of columns	1
Column type frequency: numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	-0.03	0.31	-1.92	-0.19	0	0.13	1.53	

```
skim(amy_sent$sentiment)
```

Table 9: Data summary

Name	amy_sent\$sentiment
Number of rows	51568
Number of columns	1
Column type frequency:	

Table 9: Data summary

numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	0.02	0.35	-2.3	-0.14	0	0.21	2.06	

Plot the result

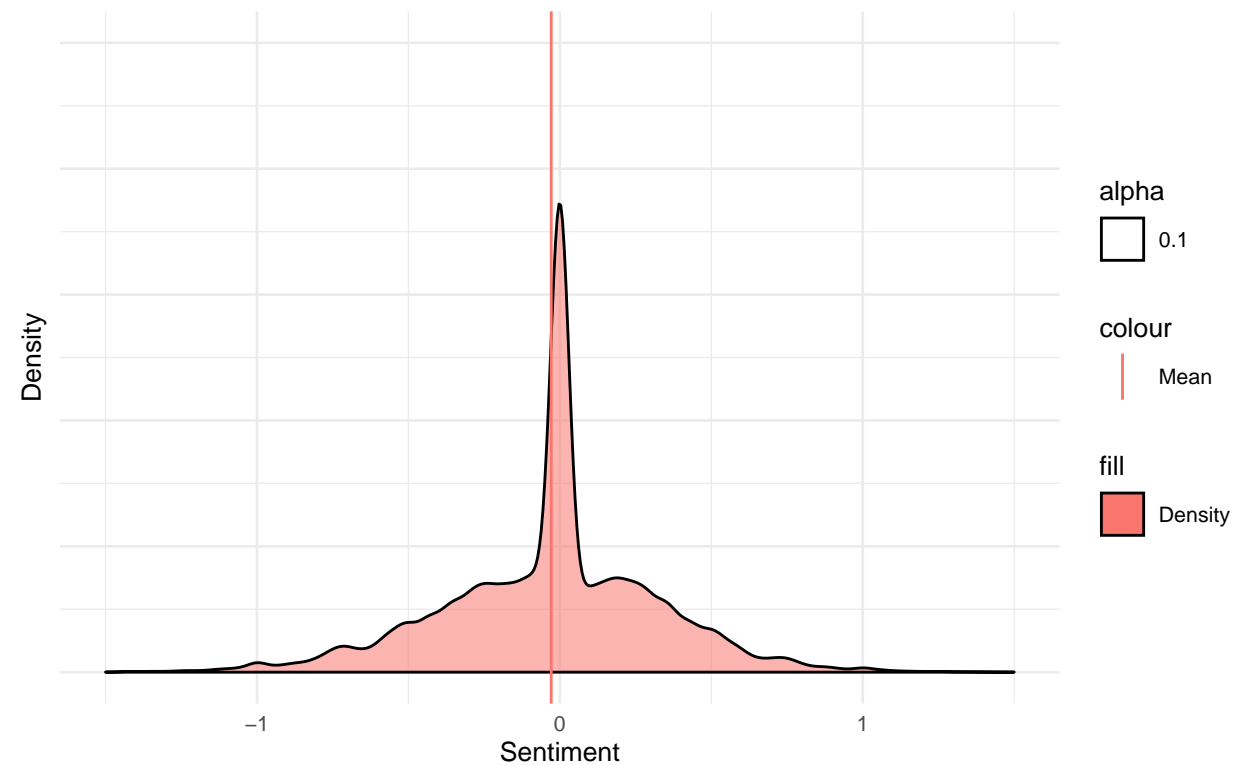
```
# function to plot the distribution of sentiment scores
sentiment_plot <- function(df){
  ggplot(df, aes(sentiment)) +
    geom_density(aes(fill = "Density", alpha = 0.1)) +
    scale_y_continuous(limits = c(0, 5)) +
    scale_x_continuous(limits = c(-1.5, 1.5)) +
    theme_minimal(base_size = 10) +
    geom_vline(aes(xintercept = mean(sentiment), color = "Mean"))
}

# customizing the labels
bernie_plot <- sentiment_plot(bernie_sent) +
  labs(x = "Sentiment",
       y = "Density",
       title = "Bernie Sanders",
       subtitle = "The Distribution of Sentiment Scores") +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
        axis.text.y=element_blank())

bernie_plot
```

Bernie Sanders

The Distribution of Sentiment Scores

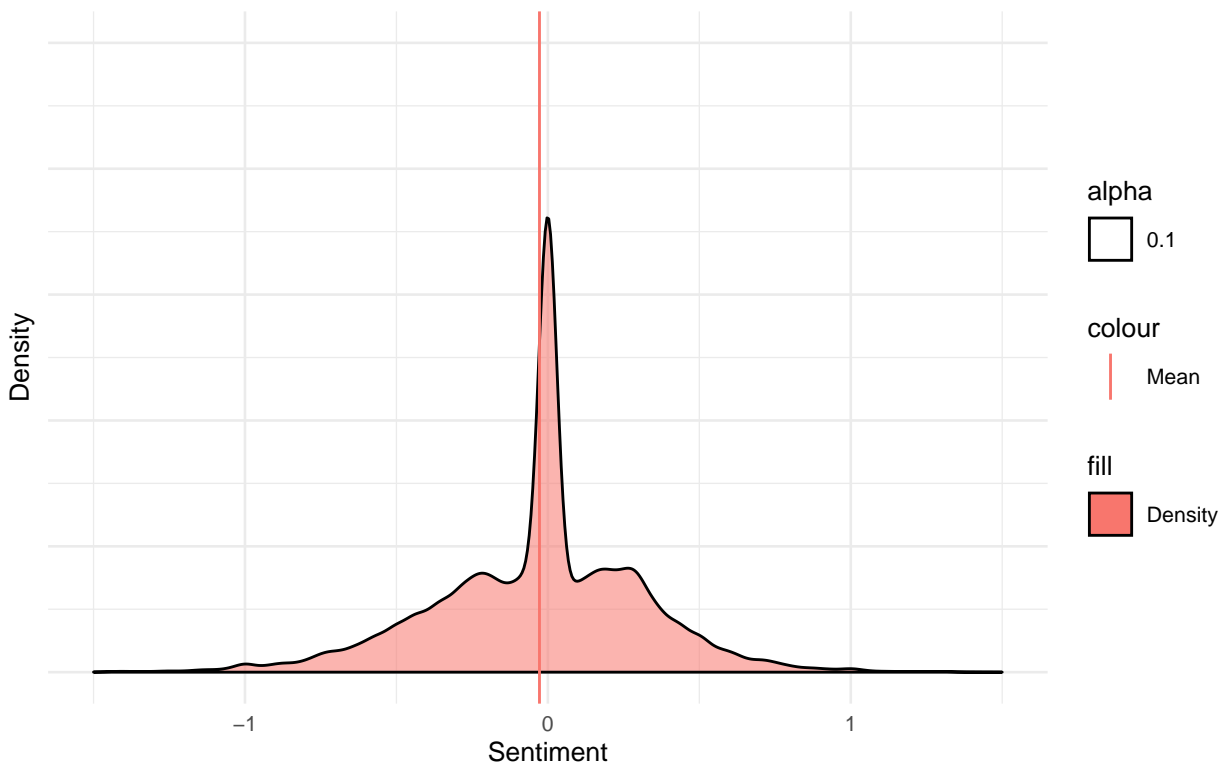


```
warren_plot <- sentiment_plot(warren_sent) +  
  labs(x = "Sentiment",  
       y = "Density",  
       title = "Elizabeth Warren",  
       subtitle = "The Distribution of Sentiment Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

```
warren_plot
```

Elizabeth Warren

The Distribution of Sentiment Scores

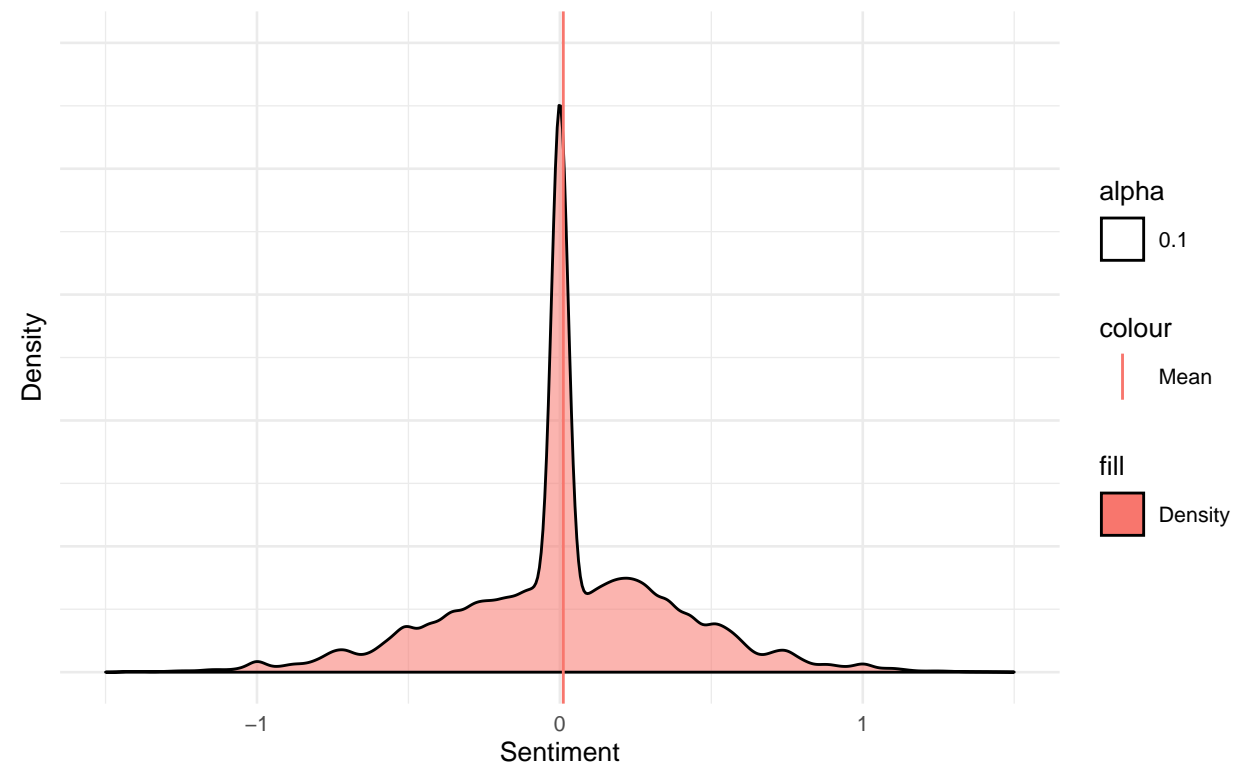


```
pete_plot <- sentiment_plot(pete_sent) +  
  labs(x = "Sentiment",  
        y = "Density",  
        title = "Pete Buttigieg",  
        subtitle = "The Distribution of Sentiment Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

pete_plot

Pete Buttigieg

The Distribution of Sentiment Scores

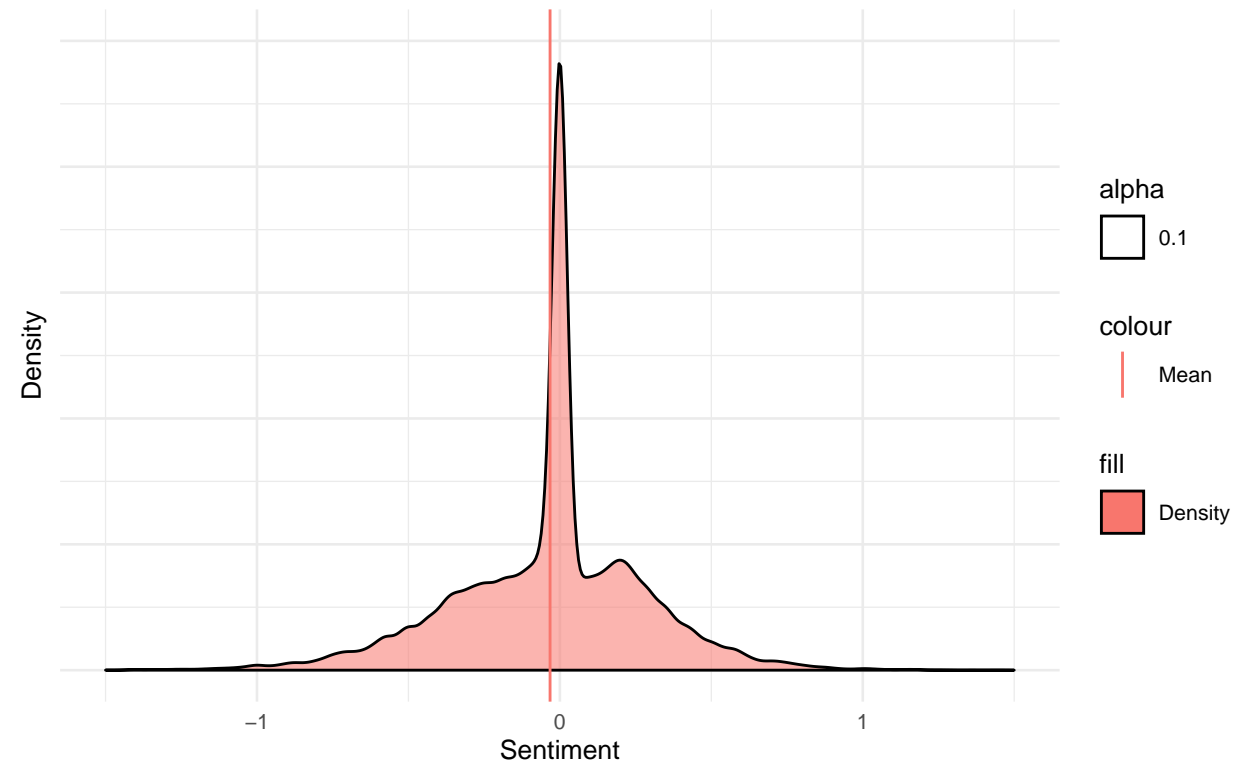


```
bloomberg_plot <- sentiment_plot(bloomberg_sent) +  
  labs(x = "Sentiment",  
       y = "Density",  
       title = "Mike Bloomberg",  
       subtitle = "The Distribution of Sentiment Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

```
bloomberg_plot
```

Mike Bloomberg

The Distribution of Sentiment Scores

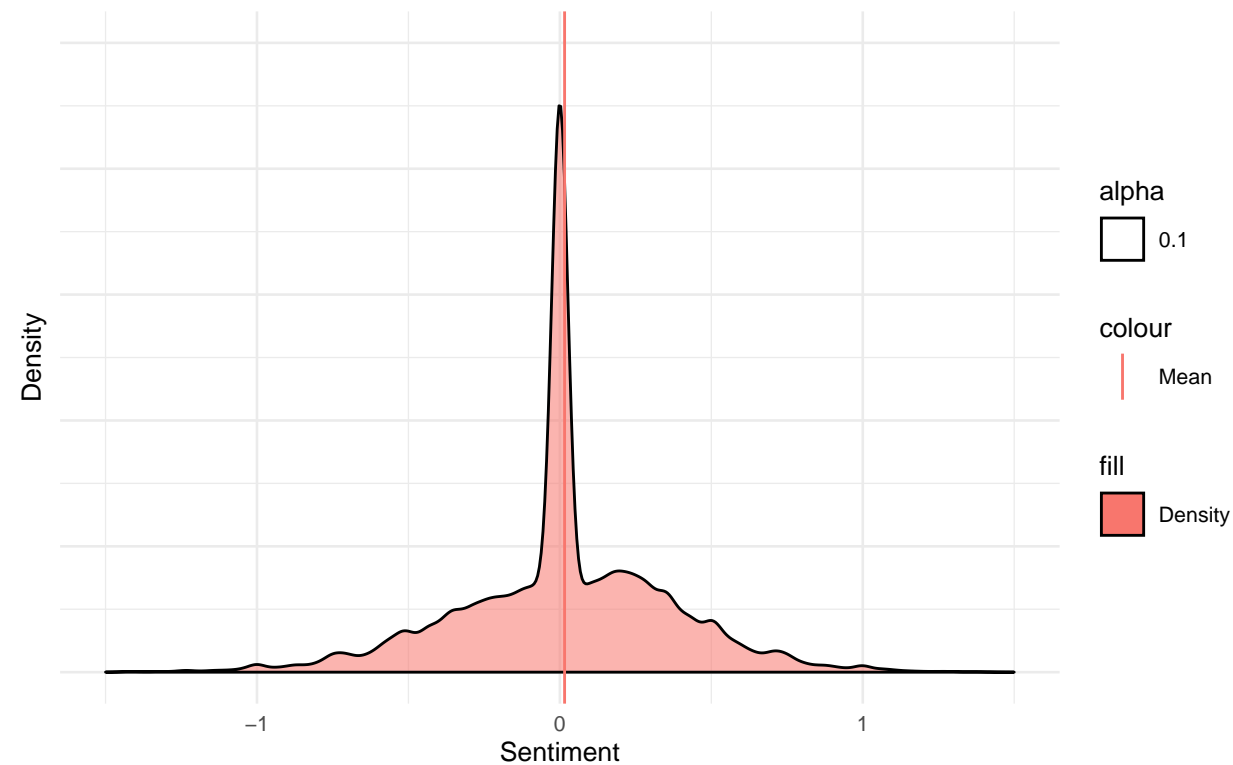


```
amy_plot <- sentiment_plot(amy_sent) +  
  labs(x = "Sentiment",  
       y = "Density",  
       title = "Amy Klobuchar",  
       subtitle = "The Distribution of Sentiment Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

amy_plot

Amy Klobuchar

The Distribution of Sentiment Scores



Polarity score: weighted sentiment scores

```
# function to calculate the sentiment score
polarity_score <- function(df) {
  df %>%
    unnest %>%
    get_sentences() %>%
    # get sentiment score for each tweet
    sentiment(polarity_dt = lexicon::hash_sentiment_sentiword) %>%
    mutate(characters = nchar(stripWhitespace(text))) %>%
    filter(characters > 1)
    # same number of obs, hence no error
}

# get sentiment scores for all our data frames
bernie_polar <- polarity_score(bernie_clean)
warren_polar <- polarity_score(warren_clean)
pete_polar <- polarity_score(pete_clean)
bloomberg_polar <- polarity_score(bloomberg_clean)
amy_polar <- polarity_score(amy_clean)

# quick summary of the result
skim(bernie_polar$sentiment)
```

Table 11: Data summary

Name	bernie_polar\$sentiment
Number of rows	51567
Number of columns	1
Column type frequency:	

Table 11: Data summary

numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	0	0.17	-1.8	-0.07	0	0.07	1.49	

```
skim(warren_polar$sentiment)
```

Table 13: Data summary

Name	warren_polar\$sentiment
Number of rows	52736
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	0.01	0.16	-1.06	-0.06	0	0.07	1.22	

```
skim(pete_polar$sentiment)
```

Table 15: Data summary

Name	pete_polar\$sentiment
Number of rows	50601
Number of columns	1
Column type frequency:	
numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	0	0.17	-1.29	-0.06	0	0.08	1.15	

```
skim(bloomberg_polar$sentiment)
```

Table 17: Data summary

Name	bloomberg_polar\$sentiment
Number of rows	55134
Number of columns	1
Column type frequency: numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	0	0.15	-1.14	-0.06	0	0.07	0.98	

```
skim(amy_polar$sentiment)
```

Table 19: Data summary

Name	amy_polar\$sentiment
Number of rows	51568
Number of columns	1
Column type frequency: numeric	1
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	0.01	0.17	-1.83	-0.05	0	0.09	1.15	

```
# function to plot the distribution of sentiment scores
```

```
polarity_plot <- function(df){
  ggplot(df, aes(sentiment)) +
    geom_density(aes(fill = "Density", alpha = 0.1)) +
    scale_y_continuous(limits = c(0, 11)) +
    scale_x_continuous(limits = c(-1.5, 1.5)) +
    theme_minimal(base_size = 9) +
    geom_vline(aes(xintercept = mean(sentiment), color = "Mean"))
}
```

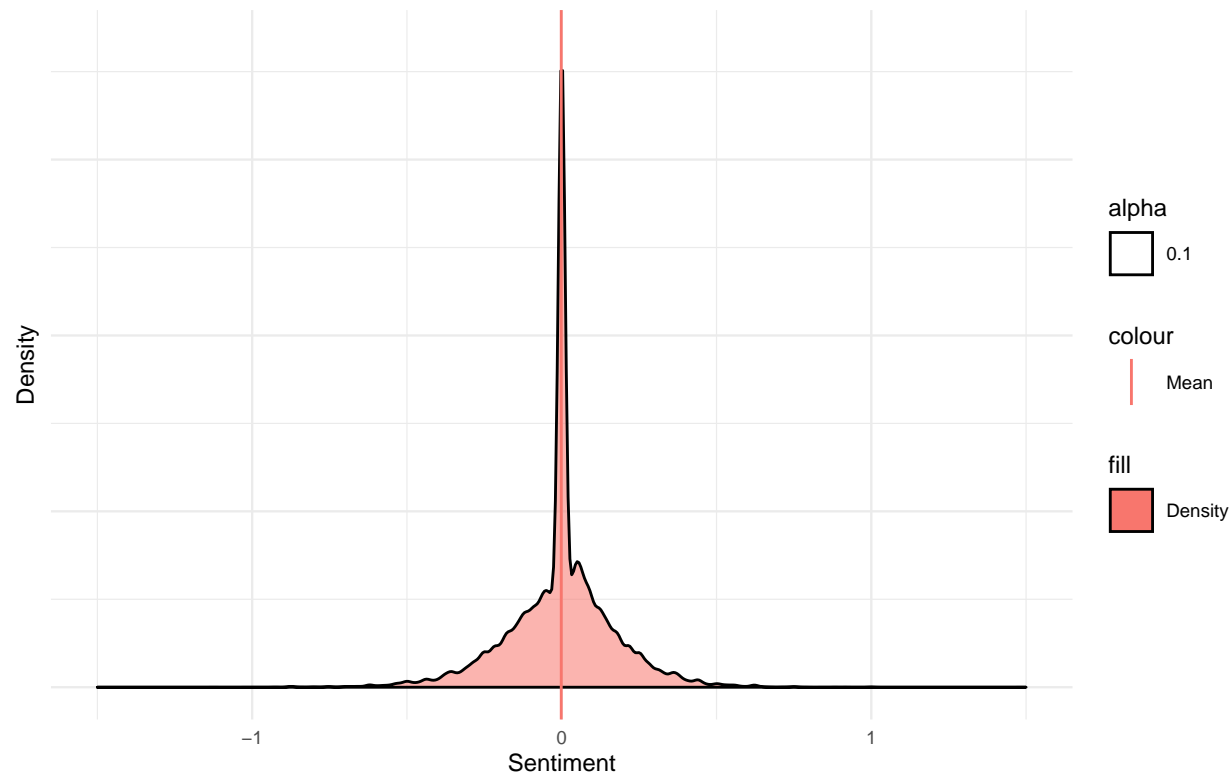
```
# customizing the labels
```

```
bernie_plot_polar <- polarity_plot(bernie_polar) +
  labs(x = "Sentiment",
       y = "Density",
       title = "Bernie Sanders",
       subtitle = "The Distribution of Polarity Scores") +
  theme(plot.title = element_text(hjust = 0.5),
        plot.subtitle = element_text(hjust = 0.5),
```

```
axis.text.y=element_blank())
```

bernie_plot_polar

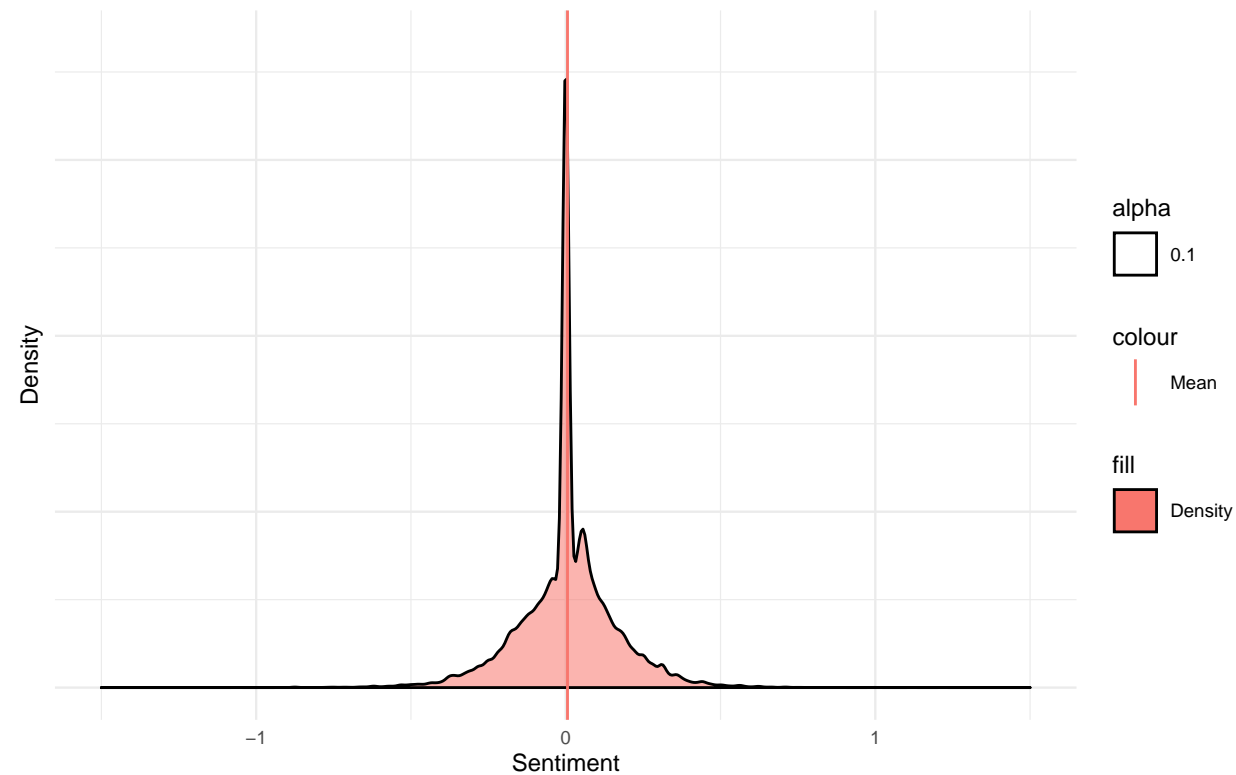
Bernie Sanders
The Distribution of Polarity Scores



```
warren_plot_polar <- polarity_plot(warren_polar) +  
  labs(x = "Sentiment",  
       y = "Density",  
       title = "Elizabeth Warren",  
       subtitle = "The Distribution of Polarity Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

warren_plot_polar

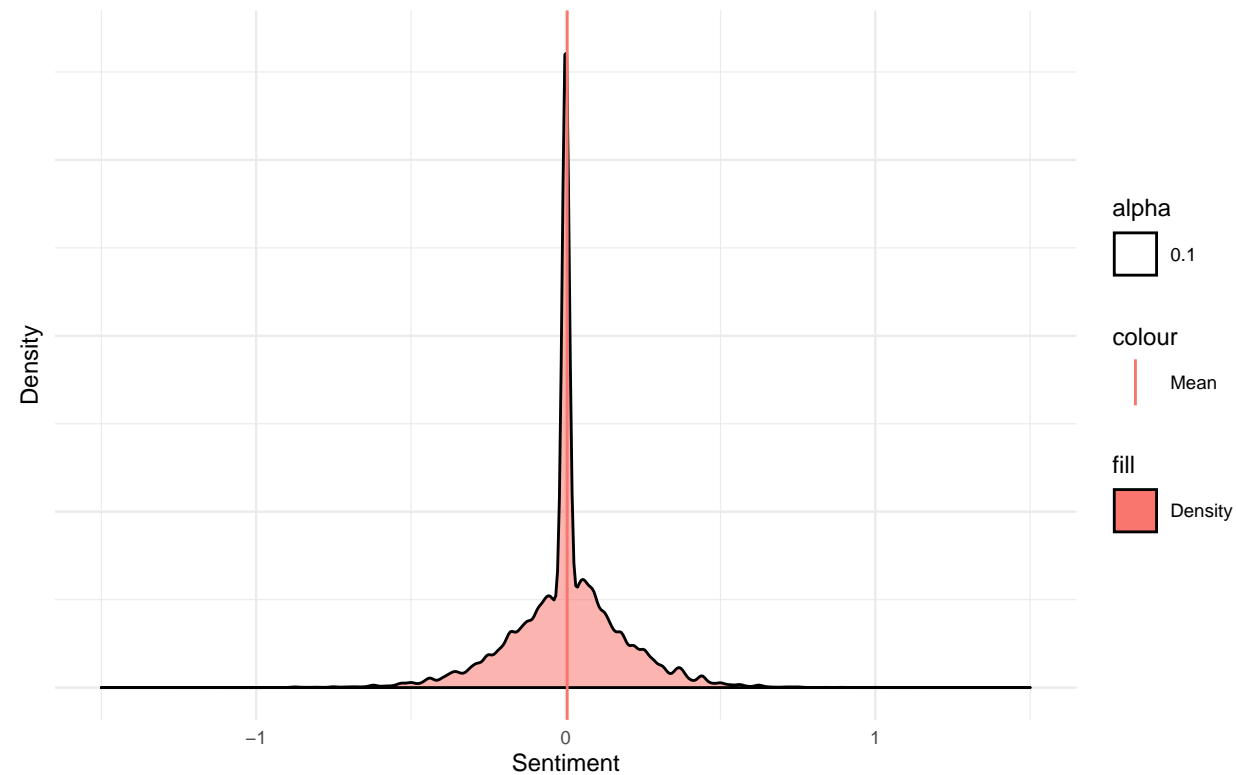
Elizabeth Warren
The Distribution of Polarity Scores



```
pete_plot_polar <- polarity_plot(pete_polar) +  
  labs(x = "Sentiment",  
        y = "Density",  
        title = "Pete Buttigieg",  
        subtitle = "The Distribution of Polarity Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

```
pete_plot_polar
```

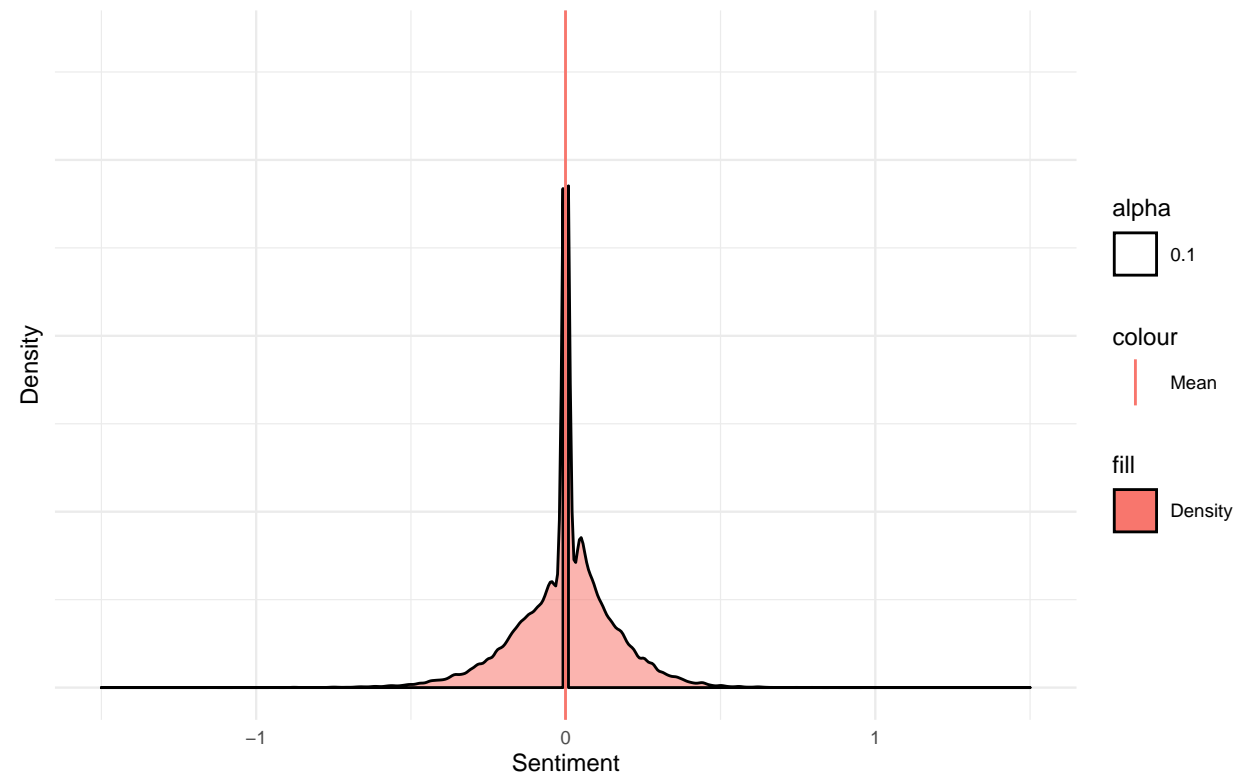
Pete Buttigieg
The Distribution of Polarity Scores



```
bloomberg_plot_polar <- polarity_plot(bloomberg_polar) +  
  labs(x = "Sentiment",  
       y = "Density",  
       title = "Mike Bloomberg",  
       subtitle = "The Distribution of Polarity Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

```
bloomberg_plot_polar
```

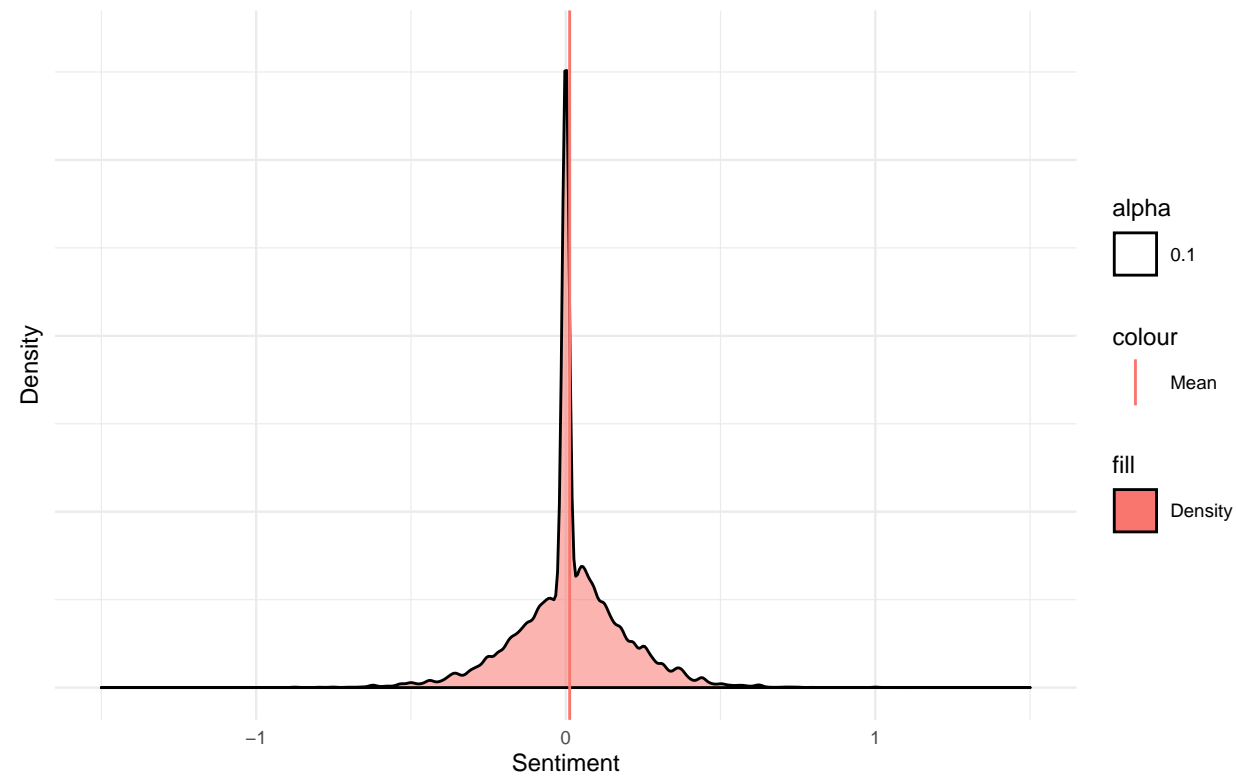
Mike Bloomberg
The Distribution of Polarity Scores



```
amy_plot_polar <- polarity_plot(amy_polar) +  
  labs(x = "Sentiment",  
        y = "Density",  
        title = "Amy Klobuchar",  
        subtitle = "The Distribution of Polarity Scores") +  
  theme(plot.title = element_text(hjust = 0.5),  
        plot.subtitle = element_text(hjust = 0.5),  
        axis.text.y=element_blank())
```

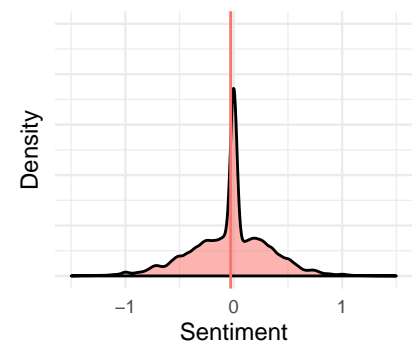
```
amy_plot_polar
```

Amy Klobuchar
The Distribution of Polarity Scores

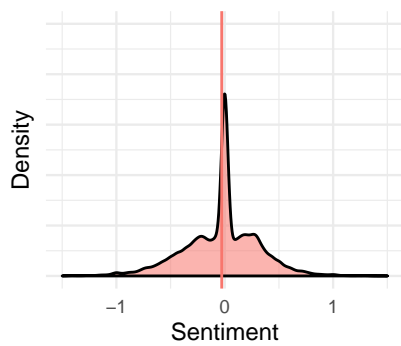


```
ggarrange(bernie_plot_polar, warren_plot_polar, pete_plot_polar, bloomberg_plot_polar, amy_plot_polar, leg
```

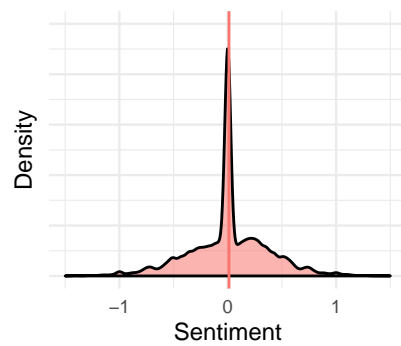
Bernie Sanders
The Distribution of Polarity Scores



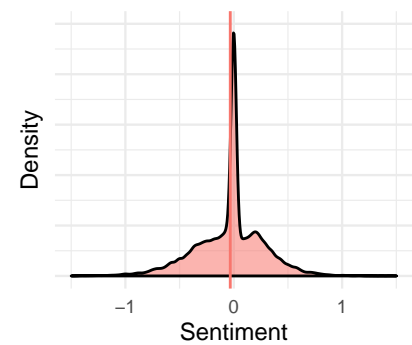
Elizabeth Warren
The Distribution of Polarity Scores



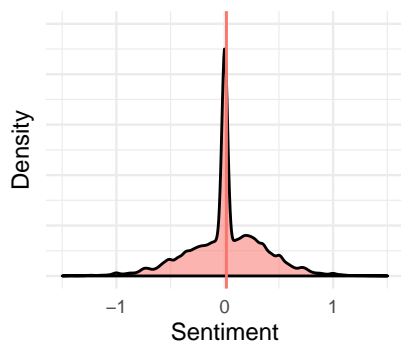
Pete Buttigieg
The Distribution of Polarity Scores



Mike Bloomberg
The Distribution of Polarity Scores



Amy Klobuchar
The Distribution of Polarity Scores



Resampling and Bootstrapping