# Final Project: Trend of Covid Vaccines, and Associations of American Presidents throughout the Pandemic

Emily Zeng (with Julia Keating as corpus / code collaborator)

## Introduction

Covid-19 drastically changed the way we function as a society, barring us from normal day to day activity that allow us to go outside and experience real life interaction. With everyone having to be indoors more, social media has given us the opportunity to still connect with others via the internet. The Covid-19 vaccines that were developed all throughout the world, and specifically Moderna and Pfizer in the U.S., were a huge issue of whether one should receive the vaccine or not. Complaints about body autonomy, governmental restrictions, and using Covid as a bioweapon were rampant in the U.S. as soon as talks about a vaccine were in the work. Additionally, throughout the pandemic, American has gone through 2 presidents: Trump and Biden. The two presidents have had distinctly unique ways of handling Covid-19 and their thoughts towards the Covid vaccines. In the following analysis, we try to answer the following guiding research questions.

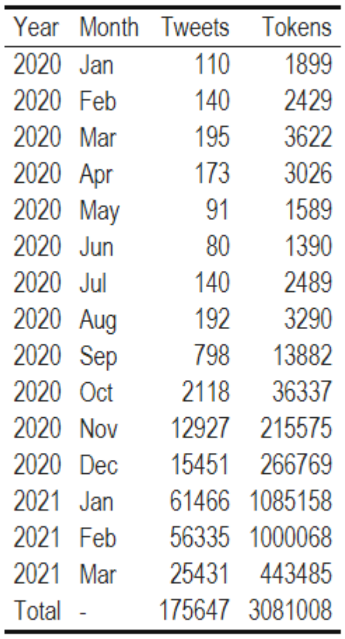
a) How frequent is the mentioning of the two most popular American vaccines, i.e. Moderna and Pfizer, and the two most recent American presidents, Biden and Trumps? Additionally, what are the most common associated words with these vaccines and presidents that are seen on Twitter's discourse?
b) What is the trend of these vaccines and presidents, and do the trends align with significant events from throughout the pandemic era, i.e. what are the changing contexts in which these words appear, and what are the major shifts in discourse around these vaccines and presidents?

## Data

The data is a corpus of about 300,000 tweets found under the *#covidvaccine* hashtag spanning from January 2020 to November 2021 that were compiled by Kaggle user, Kash (Suresh, 2020). This corpus is relevant to our research questions because we are trying to measure trends of our tokens of interest found in tweets over time. After initial data cleaning and reformatting (described later in this section), the corpus ended up being exactly 175,648 tweets spanning from January 2020 to March 2021.

In terms of data cleaning and reformatting, there were several issues with the original CSV file directly downloaded from Kaggle. Firstly, the date column was a character and not sorted chronologically. Secondly, many rows of data had missing dates (N/A values), making diachronic analysis impossible. Thirdly, many of unnecessary punctuation symbols were added. For instance, the word "impossible" shows up as "â€˜Impossibleâ€™". Lastly, for the first nine months of data, there was only one day's worth of tweets, as opposed to an entire month's worth of data.

In *Table 1*, we see a summary of the corpus by month and year, with columns for number of tweets, year, month, and the number of tokens for that month and year. As mentioned before, there is a significant difference in the number of tweets between tweets from January 2020 to August 2020 and tweets from September 2020 to March 2021, which is evident in the Tweets column of *Table 1*. For clarity, the corpus of interest was tokenized by splitting on the space character.



| Year | Month | Tweets | Tokens |
|---|---|---|---|
| 2020 | Jan | 110 | 1899 |
| 2020 | Feb | 140 | 2429 |
| 2020 | Mar | 195 | 3622 |
| 2020 | Apr | 173 | 3026 |
| 2020 | May | 91 | 1589 |
| 2020 | Jun | 80 | 1390 |
| 2020 | Jul | 140 | 2489 |
| 2020 | Aug | 192 | 3290 |
| 2020 | Sep | 798 | 13882 |
| 2020 | Oct | 2118 | 36337 |
| 2020 | Nov | 12927 | 215575 |
| 2020 | Dec | 15451 | 266769 |
| 2021 | Jan | 61466 | 1085158 |
| 2021 | Feb | 56335 | 1000068 |
| 2021 | Mar | 25431 | 443485 |
| Total | - | 175647 | 3081008 |

*Table 1: Summary of Twitter corpus.*

## Methods

The tokens we are interested in measuring the changing contexts in which they appear through our corpus are *moderna*, *pfizer*, *trump*, and *biden*. However, prior to being able to measure when and what the major shifts in discourse occur in regard to our tokens of interest, we first perform exploratory data analysis by looking at the absolute frequencies and visualizing the distributions of these frequencies over time to answer research question one.

We measured absolute frequency by using hot encoding, which assigns a tweet with 1 if one of our tokens of interest appears within that tweet, and 0 otherwise. We then calculated the combined frequency of our token of interest throughout a certain month to get the absolute frequency. The reason why we chose absolute frequency as opposed to relative frequency is because as aforementioned, the initial eight months of our data has a lot of missing data.

For the second part of EDA, we did part of speech tagging to a portion of the dataset, as tagging the entire corpus of 300,000 rows of data is computationally exhaustive. Specifically, we look at proper noun frequencies to account for our tokens of interest, meaning we want to see if *moderna*, *pfizer*, *trump*, and *biden* were some of the most frequent proper noun tokens throughout the subset of our corpus.

Lastly, we looked at collocations for our entire corpus that were the most associated with our tokens of interest by using a measure called Mutual Information (MI). Collocations represent how often a word co-occurs a word and another word. In that vein, MI scores are commonly used to measure how much information shared between x and y, so in our case, how much information is shared between our token of interest and other tokens that are commonly associated with one another. If two tokens have high MI scores, this indicates that they are highly associated.

For our analysis of whether there are changing contexts for our tokens of interest to answer research question 2, we follow the method described in "Ireland in British parlimenatry debates" (Baker, Brezina, McEnery, 2017), without fitting a non-parametric general additive model. This includes 4 steps: 1) create chunks of rolling windows in the data 2) find the MI scores for each chunk 3) for each pair of chunks of rolling windows, find the Gwet's AC , and 4) plot the Gwet's AC values against each rolling window chunk, and then smooth the scatterplot to see the overall trend. Each step will be explained in detail below.

The first step is to create rolling window chunks in the data. This means to create a chunk of data that slides through the corpus approximately 4 months at a time. We ended up with 11 chunks data, with each chunk's time period described below. The reason why we chose a chunk to be 4 months is because any period shorter would yield chunks with identical data due to issues of missing data for the first 8 months and any period longer would yield a limited number of chunks to plot later on.

- Chunk 1: January 9, 2020 – May 1, 2020
- Chunk 2: February 1, 2020 – June 1, 2020
- Chunk 3: March 1, 2020 – July 1, 2020
- Chunk 4: April 1, 2020 – August 1, 2020
- Chunk 5: May 1, 2020 – September 1, 2020
- Chunk 6: June 1, 2020 – October 1, 2020
- Chunk 7: July 1, 2020 – November 1, 2020
- Chunk 8: August 1, 2020 – December 1, 2020
- Chunk 9: September 1, 2020 – January 1, 2021
- Chunk 10: October 1, 2020 – February 1, 2021
- Chunk 11: November 1, 2020 – March 21, 2021

Then, for each token that we are interested in, we do the following 3 steps:

1) For each chunk, calculate the MI scores and only keep the tokens that have a minimum threshold MI score of 5. Then, depending on the chunk, we decided on the threshold for the collocate frequency (if it's a chunk with very little data, all the collocate tokens were kept, but if it were chunk 8-11, collocates with a collocate frequency of at least 2, 3, or 5 were kept). We also used a span of 5 left and 5 right, meaning a token would be considered a collocate if it were at most 5 words away from our token of interest.
2) For each pair of chunks, i.e. chunk 1 and chunk 2, chunk 2 and chunk 3, … chunk 10 and chunk 11, find Gwet's AC. Gwet's AC is an inter-rater measure that indicates

how similar or different the collocates for each pair of chunks are (Baker, Brezina, McEnery, 2017). The smaller Gwet's AC is, the less similar the collocates are for a pair of chunks.

3) Lastly, we plot the Gwet's AC value found for each chunk against the chunk, and then smooth the resulting scatterplot to see the overall trend better. We examine where the peaks and troughs of the Gwet's AC values appear over time because we want to see where a set of shared collocates converge (peak) or diverge (trough). Simply put, we want to see if the change in discourse where there are peaks / troughs align with current events during that rolling period chunk.

## Results

To answer research question 1, we employed EDA methods to measure frequencies and distributions of our tokens of interest over time. In *Figure 1*, we see bar plots showing the absolute frequency of each token from January 2020 to March 2021. As noted before, the number of tweets prior to September 2020 was extremely limited, which is evident in the low frequencies of each token before October 2020. During January 2021, there is a peak in the frequency of *moderna*, *pfizer*, and *biden*, which makes sense because Biden took office as the American president during this time and contributed to a culture of getting America getting back on track with beating Covid by encouraging people to get the Covid vaccines. There is a peak in frequency for the *trump* token during November 2020, which aligns with the 2020 U.S. Presidential elections, where Trump was running again to be America's president during a second term. Trump has been avidly promoting that Covid is a scam, and that vaccines are not a good way to combat the pandemic.

Next, we looked at the most frequent proper noun tokens for a subset of our corpus (10,000 rows instead of 300,000). In *Table 2*, we see the most frequent proper noun tokens and their absolute frequencies. As expected, we see that the tokens *moderna* and *pfizer* appear within the top 10 most frequent proper nouns, which makes sense and confirms our motive to conduct the following analysis on changing contexts with the appearance of these tokens. It's interesting to note that the other American Covid vaccine, JohnsonJohnson, is missing from this table. Moderna and Pfizer happen to the more popular vaccines in the U.S, so it is unsurprising that those tweeting about the Covid vaccine would mention Johnson-Johnson. It should also be noted that JnJ was recalled for a time during the earlier part of 2021. Lastly, both *covaxin* (India's vaccine) and *india* show up in the most frequent proper noun tokens. This also makes sense because most recently with the onset of the Delta variant, India faced a huge second wave crisis across its nation. Suddenly, a nation that had "beat Covid" suddenly became an example of what other

Lastly, for EDA, we looked at some of the most frequent and highly associated collocates for each token of interest, as shown in *Table 3*. We take note of the tokens 94.5 and 92 as collocates for *moderna*. This shows that there was a lot of discourse on Twitter regarding the efficacy of the Moderna vaccine protecting against Covid. On the other hand, we see that the most frequent and highly associated collocates for the pfizer token have to do with the company and leadership that created the Pfizer vaccine. Specifically, these are *albert*, *boula*, and variations of *biotech*. Discourse on the Pfizer vaccine on Twitter seems to be more centered on who created the Pfizer vaccine and the leadership behind the company that spearheaded the development of the Pfizer vaccine, as opposed to efficacy of the vaccine itself. For the *trump* token, we see the *eos* token, which is specifically about President Trump's executive order on ensuring that Americans have priority access to Covid vaccines, which was published December 2020 (Ensuring Access to United States Government COVID-19 Vaccines, 2020). Additionally, the *sieg* token refers to Trump's association with the alt-right, as the sieg heil is a Nazi salute often used by Trump supporters. This checks out with kind of audience that Trump has garnered even prior to his becoming president. Lastly, the top collocate to the trump token is *climatejustice/climateaction*. These groups are advocates for pro-climate change action, or more generally, pro-science. It can be inferenced that these tokens are associated with the *trump* token because Trump was actively advocating that Covid-19 is a hoax, and has said similar things about climate change; both are ways in which he denies science. On the other hand, President Biden's standpoint on Covid and vaccination has been a complete 180 to Trump's. The most frequent and most highly associated collocate with biden is *eos*, again referring to executive orders. This most likely refers to President Biden's September 2021 executive order that required federal workers to be vaccinated (Requiring Coronavirus Disease 2019 Vaccination for Federal Employees, 2021). What's interesting is in direct retaliation to this executive order, we see the second most frequent and highly associated collocate to be *fired/suspended*. Many employees were fired or suspended for refusing to be vaccinated.

The results from the above sections for our EDA lead into analyzing the results for the second part of our analysis. In *Figure 2*, we see the convergence and divergence of each token of interest over time. We are only interested in troughs, as these periods show when there is a shift in discourse over a certain token, and how there is changing context within the appearance of our tokens.

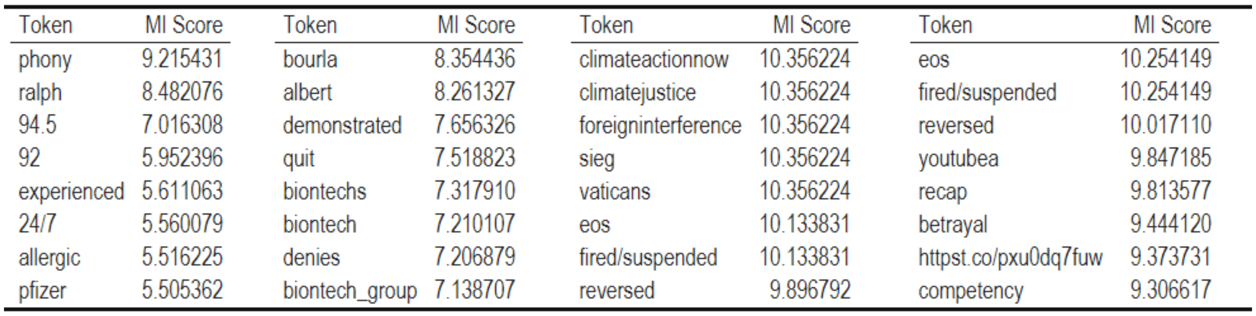
For the *moderna* token, we see a trough indicated by the blue line at around chunk 4, which is April 1, 2020 to August 1, 2020. Since there is a trough, this indicates that there was a shift in discourse on *moderna* throughout Twitter. Coincidentally, during this time period, around May 2020, Moderna reported positive results for their clinical trials for their vaccine. Similar commentary can be said about the *pfizer* token. We see a trough at around chunk 4 as well, indicating that again, there is a change in discourse on *pfizer* throughout Twitter. During early May of 2020, Pfizer also announced that they were conducting clinical trials to test the efficacy of their vaccine against Covid-19. The bottom line is that these major current events affected a shift in discussion of Covid vaccines on Twitter.

For the *trump* token, there is a trough at around chunk 5. Around this time period (May 1, 2020 – September 1, 2020), there was a shift in discourse on the appearance of Trump on Twitter, which aligns with a lot of different events: many Americans were finally getting access to the Covid vaccines, Trump and his public health officials had various opinions on the discussion of Covid and the vaccines, etc. Overall, it can be said that there was a lot of discussion about the Covid vaccine that occurred throughout Twitter, both positive and negative, when it came to Trump and the vaccines. Lastly, there seems to be 2 different troughs for the *biden* token, one at chunk 6 (June 1, 2020 – October 1, 2020) and one at chunk 7 (July 1, 2020 – November 1, 2020). Since there are too many historical events that occurred during these periods, we outline two important events with regards to President

Biden and the Covid vaccines include the following: his plan of action to beat Covid during his presidential campaign and his becoming president elect in November 2020 and encouraging that vaccines are safe, trustworthy, and necessary to help American fight the pandemic. These events would definitely be a cause for shift in discourse in the appearance of the *biden* token on Twitter.

| Token | Frequency | Rank |
|---|---|---|
| covidvaccine | 333 | 1 |
| covid | 80 | 2 |
| #covidvaccine | 41 | 3 |
| pfizer | 37 | 4 |
| slots | 34 | 5 |
| vaccine | 31 | 6 |
| india | 31 | 6 |
| covid19 | 30 | 8 |
| moderna | 21 | 9 |
| covaxin | 19 | 10 |

*Table 2: (Partial) most frequent proper noun tokens for Twitter corpus.*



| Token | MI Score | Token | MI Score | Token | MI Score | Token | MI Score |
|---|---|---|---|---|---|---|---|
| phony | 9.215431 | bourla | 8.354436 | climateactionnow | 10.356224 | eos | 10.254149 |
| ralph | 8.482076 | albert | 8.261327 | climatejustice | 10.356224 | fired/suspended | 10.254149 |
| 94.5 | 7.016308 | demonstrated | 7.656326 | foreigninterference | 10.356224 | reversed | 10.017110 |
| 92 | 5.952396 | quit | 7.518823 | sieg | 10.356224 | youtubea | 9.847185 |
| experienced | 5.611063 | biontechs | 7.317910 | vaticans | 10.356224 | recap | 9.813577 |
| 24/7 | 5.560079 | biontech | 7.210107 | eos | 10.133831 | betrayal | 9.444120 |
| allergic | 5.516225 | denies | 7.206879 | fired/suspended | 10.133831 | httpst.co/pxu0dq7fuw | 9.373731 |
| pfizer | 5.505362 | biontech_group | 7.138707 | reversed | 9.896792 | competency | 9.306617 |

*Table 3: Top tokens associated with moderna, pfizer, trump, and biden (left to right) tokens and their MI scores, respectively.*

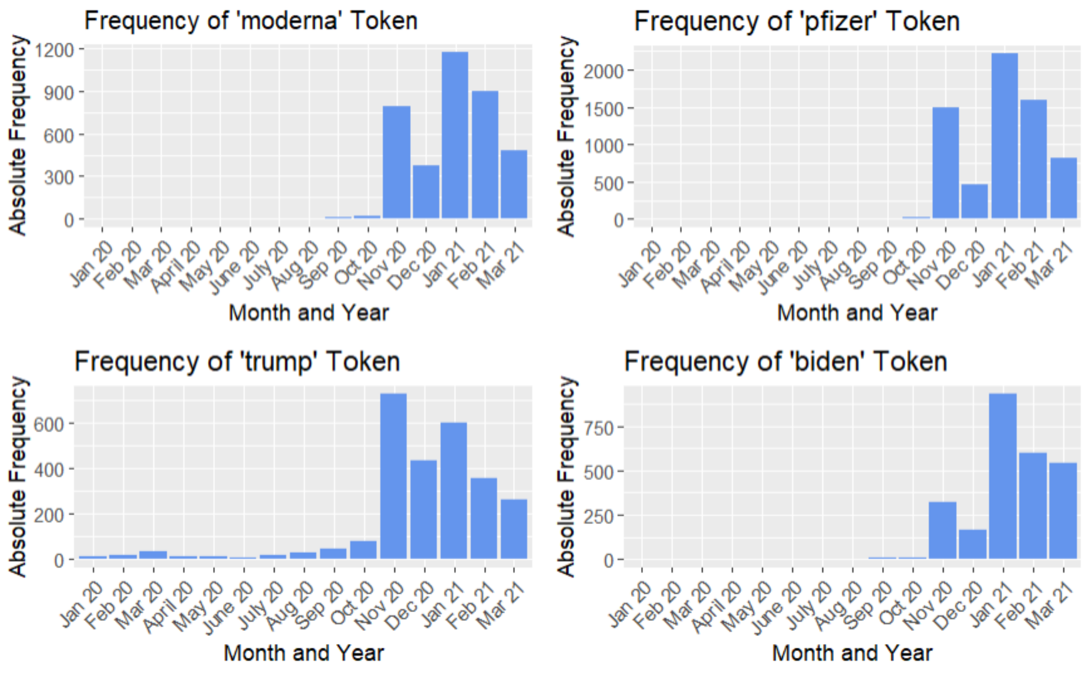


*Figure 1: Absolute frequencies of moderna, pfizer, trump, and biden tokens from January 2020 to March 2021.*

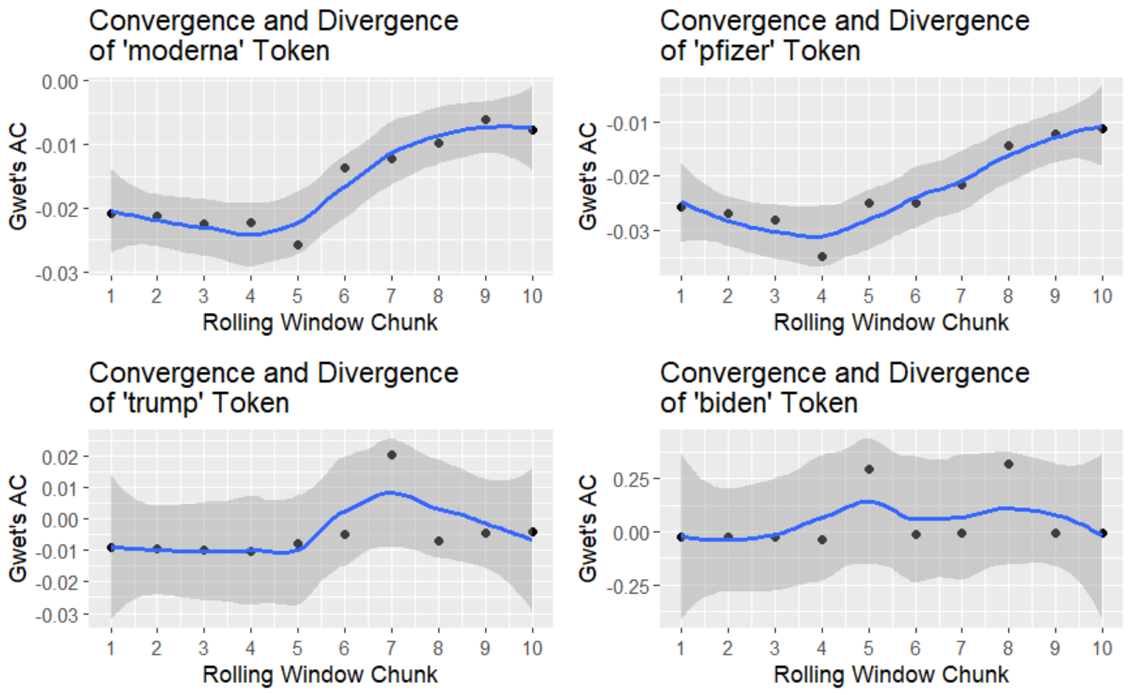


*Figure 2: Convergence / divergence of moderna, pfizer, trump, and biden tokens over each rolling window chunk.*

## Discussion

There is a plethora of limitations that this analysis suffers from. The conclusions in which we arrived at in terms of the second research question merely show when there are shifts in discourse for a certain token and which events align with the shift. However, there is no information on what exactly is being said during these shifts, how intense these shifts are, or whether the shifts in discourse have are long-lasting or ephemeral. Future steps to address these issues include more comprehensive investigation by analyzing the corpus at these specified time periods when there are shifts in discourse to see if the change in discourse being linked to current events is real, or purely coincidental.

Additionally, other limitations within the analysis arise when we think about the limited amount of data we were provided with. The original study we replicated used roughly 200 years of data, whereas our dataset only consisted of 14 months of data. The convergence and divergence of collocates to our tokens of interest were difficult to visualize due to the lack of data. There were only 10 chunks of rolling window data, rather than 166 that the original study used. It made for visualization and modeling much better. Future work can be done to increase the amount of data used to show more apparent divergences in shared collocates with our tokens of interest.

## References

Baker, H., Brezina, V., & McEnery, T. (2017). Ireland in British parliamentary debates 1803–2004. *Advances in Historical Sociolinguistics*, 83–107. https://doi.org/10.1075/ahs.7.04bak

Ensuring Access to United States Government COVID-19 Vaccines Fed. Reg. (December 11, 2020).

Requiring Coronavirus Disease 2019 Vaccination for Federal Employees Fed. Reg. (September 14, 2021).

Gabrielatos, C., McEnery, T., Diggle, P. J., & Baker, P. (2012). The peaks and troughs of corpus-based contextual analysis. *International Journal of Corpus Linguistics*, *17*(2), 151–175. https://doi.org/10.1075/ijcl.17.2.01gab

Suresh, K. (2020, August). Covid Vaccine Tweets, Version 37. Retrieved October 31, 20201 from https://www.kaggle.com/kaushiksuresh147/covidvaccine-tweets

## Technical Appendix

```
## load in libraries
library(tidyr)
library(tidyverse)
library(dplyr)
library(rvest)
library(robotstxt)
library(janitor)
```

```r
library(cmu.textstat)
library(quanteda)
library(quanteda.textstats)
library(udpipe)
library(future.apply)
library(purrr)
library(syuzhet)
library(irrCAC)
library(lubridate)
library(mgcv)
library(tidymv)
library(voxel)
library(kableExtra)
```

### Read in data and data cleaning

```r
tweets <- read.csv("../data/covidvaccine.csv")

##
all_tweets <- tweets %>%
  mutate(text = str_remove_all(text, '‰¥˜³@â€™¦ðÿ‡šï¸œž')) %>%
  mutate(doc_id = paste0("tweet", "_", seq(1:nrow(tweets)))) %>%
  dplyr::select(doc_id, text, date)

## remove weird symbols
all_tweets$text <- lapply(all_tweets$text, str_remove_all, '[:‰¥˜³@â€™¦ðÿ‡šï¸
œž“ˆ]')

## make all tweets lowercase
all_tweets$text <- lapply(all_tweets$text, tolower)

## replace tokens with tokens of interest
all_tweets$text <- str_replace(all_tweets$text, "covid-19", "covid_19")
all_tweets$text <- str_replace(all_tweets$text, "bidenharris", "biden")
all_tweets$text <- str_replace(all_tweets$text, "biden2020", "biden")
all_tweets$text <- str_replace(all_tweets$text, "joebiden", "biden")
all_tweets$text <- str_replace(all_tweets$text, "bidens", "biden")
all_tweets$text <- str_replace(all_tweets$text, "realdonaldtrump", "trump")
all_tweets$text <- str_replace(all_tweets$text, "modernas", "moderna")
all_tweets$text <- str_replace(all_tweets$text, "moderna_tx", "moderna")
all_tweets$text <- str_replace(all_tweets$text, "donaldtrump", "trump")
all_tweets$text <- str_replace(all_tweets$text, "trumps", "trump")
all_tweets$text <- str_replace(all_tweets$text, "pfizers", "pfizer")

## create new col to convert date from character to posixct form
all_tweets <- all_tweets %>% mutate(Time = mdy_hm(all_tweets$date))

## create new col for month of tweet
all_tweets$month <- month(all_tweets$Time, label = T)

## create new col for year of tweet
all_tweets$year <- year(all_tweets$Time)
```

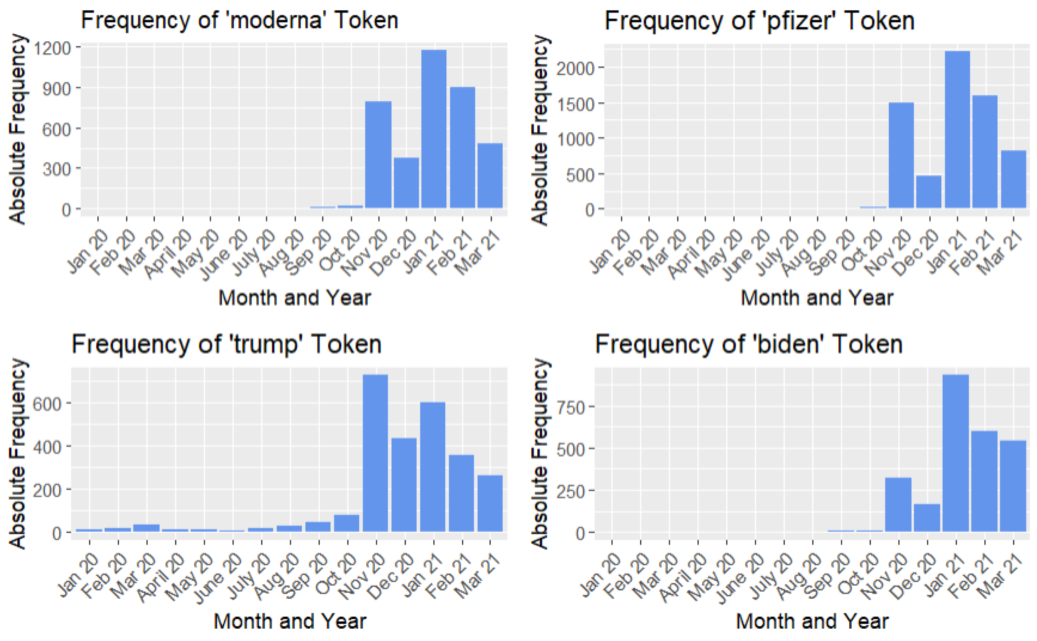
```r
## create corpus of entire corpus
tweets_corpus <- all_tweets %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

## create dfm
tweets_dfm <- dfm(tweets_corpus)

## corpus composition table
tweets_corpus_comp <- ntoken(tweets_dfm) %>%
  data.frame(Tokens = .,Month = all_tweets$month,Year = all_tweets$year) %>%
  group_by(Year,Month) %>%
  summarize(Tweets = n(),
    Tokens = sum(Tokens)) %>%
  janitor::adorn_totals()
```

**Hot encoding of interested tokens, then create columns of where token appears in the tweet**

```r
moderna <- "moderna"
pfizer <- "pfizer"
biden <- "biden"
trump <- "trump"

if_moderna <- rep(0, nrow(all_tweets))
for (i in 1:nrow(all_tweets)) {
  if_moderna[i] <- grepl(moderna, all_tweets$text[i])
}
sum(if_moderna)

## [1] 3767

if_pfizer <- rep(0, nrow(all_tweets))
for (i in 1:nrow(all_tweets)) {
  if_pfizer[i] <- grepl(pfizer, all_tweets$text[i])
}
sum(if_pfizer)

## [1] 6619

if_biden <- rep(0, nrow(all_tweets))
for (i in 1:nrow(all_tweets)) {
  if_biden[i] <- grepl(biden, all_tweets$text[i])
}
sum(if_biden)

## [1] 2602

if_trump <- rep(0, nrow(all_tweets))
for (i in 1:nrow(all_tweets)) {
  if_trump[i] <- grepl(trump, all_tweets$text[i])
}
sum(if_trump)

## [1] 2657

which(if_pfizer == 1)

175443
## [6611] 175455 175521 175544 175555 175569 175570 175592 175595 175604

all_tweets <- all_tweets %>%
  mutate(if_moderna, if_pfizer, if_biden, if_trump)
```

## Plot frequency of moderna token over time

```
moderna_tweets <- all_tweets %>%
  mutate(date_col = floor_date(all_tweets$Time,"month")) %>%
  group_by(date_col) %>%
  summarise(freq = sum(if_moderna))

months <- c("Jan 20","Feb 20","Mar 20","April 20","May 20","June 20","July 20
","Aug 20","Sep 20","Oct 20","Nov 20","Dec 20","Jan 21","Feb 21","Mar 21")

ggplot(moderna_tweets, aes(factor(months, levels = months),freq)) +
  geom_col(fill = "cornflowerblue") +
  ggtitle("Frequency of 'moderna' Token Over Time") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ylab("Absolute Frequency") +
  xlab("Month and Year")

pfizer_tweets <- all_tweets %>%
  mutate(date_col = floor_date(all_tweets$Time,"month")) %>%
  group_by(date_col) %>%
  summarise(freq = sum(if_pfizer))

ggplot(pfizer_tweets, aes(factor(months, levels = months),freq)) +
  geom_col(fill = "cornflowerblue") +
  ggtitle("Frequency of 'pfizer' Token over Time") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ylab("Absolute Frequency") +
  xlab("Month and Year")

trump_tweets <- all_tweets %>%
  mutate(date_col = floor_date(all_tweets$Time,"month")) %>%
  group_by(date_col) %>%
  summarise(freq = sum(if_trump))

ggplot(trump_tweets, aes(factor(months, levels = months),freq)) +
  geom_col(fill = "cornflowerblue") +
  ggtitle("Frequency of 'trump' Token Over Time") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ylab("Absolute Frequency") +
  xlab("Month and Year")

biden_tweets <- all_tweets %>%
  mutate(date_col = floor_date(all_tweets$Time,"month")) %>%
  group_by(date_col) %>%
  summarise(freq = sum(if_biden))

ggplot(biden_tweets, aes(factor(months, levels = months),freq)) +
  geom_col(fill = "cornflowerblue") +
  ggtitle("Frequency of 'biden' Token Over Time") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  ylab("Absolute Frequency") +
  xlab("Month and Year")

grid.arrange(a, b, c, d, ncol=2)
```



## Create 10 chunks of rolling windows

```
## create chunks of rolling window

chunk1 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-01-09 0
0:04:00"), ymd_hms("2020-05-01 23:59:00")),]
chunk2 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-02-01 0
0:00:00"), ymd_hms("2020-06-01 23:59:00")),]
chunk3 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-03-01 0
0:00:00"), ymd_hms("2020-07-01 23:59:00")),]
chunk4 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-04-01 0
0:00:00"), ymd_hms("2020-08-01 23:59:00")),]
chunk5 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-05-01 0
0:00:00"), ymd_hms("2020-09-01 23:59:00")),]
chunk6 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-06-01 0
0:00:00"), ymd_hms("2020-10-01 23:59:00")),]
chunk7 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-07-01 0
0:00:00"), ymd_hms("2020-11-01 23:59:00")),]
chunk8 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-08-01 0
0:00:00"), ymd_hms("2020-12-01 23:59:00")),]
chunk9 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-09-01 0
0:00:00"), ymd_hms("2021-01-01 23:59:00")),]
chunk.10 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-10-01
00:00:00"), ymd_hms("2021-02-01 23:59:00")),]
chunk.11 <- all_tweets[all_tweets$Time %within% interval(ymd_hms("2020-11-01
00:00:00"), ymd_hms("2021-03-21 23:59:00")),

## create tokens for each rolling window

chunk1_tkn <- chunk1 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk2_tkn <- chunk2 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk3_tkn <- chunk3 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk4_tkn <- chunk4 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk5_tkn <- chunk5 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk6_tkn <- chunk6 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk7_tkn <- chunk7 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")
```

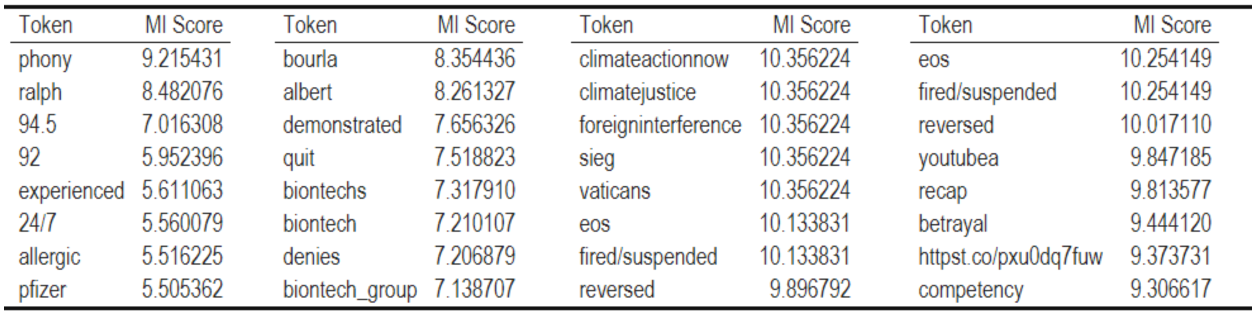
```
chunk8_tkn <- chunk8 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk9_tkn <- chunk9 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk10_tkn <- chunk.10 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword")

chunk11_tkn <- chunk.11 %>%
  mutate(text = preprocess_text(text)) %>%
  corpus() %>%
  tokens(what="fastestword"
```

### Calculate MI scores for entire corpus and then moderna token for each chunk

```
## create moderna collcates for entire corpus
all_moderna <- collocates_by_MI(tweets_corpus, "moderna")
all_moderna <- all_moderna %>% filter(MI_1 >= 5) %>% select(token,MI_1)

all_pfizer <- collocates_by_MI(tweets_corpus, "moderna")
all_pfizer <- all_pfizer %>% filter(MI_1 >= 5) %>% select(token,MI_1)

all_trump <- collocates_by_MI(tweets_corpus, "moderna")
all_trump <- all_trump %>% filter(MI_1 >= 5) %>% select(token,MI_1)

all_biden <- collocates_by_MI(tweets_corpus, "moderna")
all_biden <- all_biden %>% filter(MI_1 >= 5) %>% select(token,MI_1)

## create moderna collcates for chunks
moderna_collocations1 <- collocates_by_MI(chunk1_tkn, "moderna")
moderna_collocations2 <- collocates_by_MI(chunk2_tkn, "moderna")
moderna_collocations3 <- collocates_by_MI(chunk3_tkn, "moderna")
moderna_collocations4 <- collocates_by_MI(chunk4_tkn, "moderna")
moderna_collocations5 <- collocates_by_MI(chunk5_tkn, "moderna")
moderna_collocations6 <- collocates_by_MI(chunk6_tkn, "moderna")
moderna_collocations7 <- collocates_by_MI(chunk7_tkn, "moderna")
moderna_collocations8 <- collocates_by_MI(chunk8_tkn, "moderna")
moderna_collocations9 <- collocates_by_MI(chunk9_tkn, "moderna")
moderna_collocations10 <- collocates_by_MI(chunk10_tkn, "moderna")
moderna_collocations11 <- collocates_by_MI(chunk11_tkn, "moderna")

## filter out high mi's and high collocate frequency
modc1 <- moderna_collocations1 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
modc2 <- moderna_collocations2 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
```

```r
modc3 <- moderna_collocations3 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
modc4 <- moderna_collocations4 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
modc5 <- moderna_collocations5 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
modc6 <- moderna_collocations6 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
modc7 <- moderna_collocations7 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
modc8 <- moderna_collocations8 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% sele
ct(token,MI_1)
modc9 <- moderna_collocations9 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% sele
ct(token,MI_1)
modc10 <- moderna_collocations10 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% se
lect(token,MI_1)
modc11 <- moderna_collocations11 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% se
lect(token,MI_1)

## outer join to create table
modc1_2 <- merge(x = modc1, y = modc2, by = "token", all = TRUE, sort = F)
modc2_3 <- merge(x = modc2, y = modc3, by = "token", all = TRUE, sort = F)
modc3_4 <- merge(x = modc3, y = modc4, by = "token", all = TRUE, sort = F)
modc4_5 <- merge(x = modc4, y = modc5, by = "token", all = TRUE, sort = F)
modc5_6 <- merge(x = modc5, y = modc6, by = "token", all = TRUE, sort = F)
modc6_7 <- merge(x = modc6, y = modc7, by = "token", all = TRUE, sort = F)
modc7_8 <- merge(x = modc7, y = modc8, by = "token", all = TRUE, sort = F)
modc8_9 <- merge(x = modc8, y = modc9, by = "token", all = TRUE, sort = F)
modc9_10 <- merge(x = modc9, y = modc10, by = "token", all = TRUE, sort = F)
modc10_1 <- merge(x = modc10, y = modc11, by = "token", all = TRUE, sort = F

modc1_2[is.na(modc1_2)] = 0
modc2_3[is.na(modc2_3)] = 0
modc3_4[is.na(modc3_4)] = 0
modc4_5[is.na(modc4_5)] = 0
modc5_6[is.na(modc5_6)] = 0
modc6_7[is.na(modc6_7)] = 0
modc7_8[is.na(modc7_8)] = 0
modc8_9[is.na(modc8_9)] = 0
modc9_10[is.na(modc9_10)] = 0
moc10_11[is.na(modc10_11)] = 0

moderna_ac <- c()
moderna_ac[1] <- gwet.ac1.raw(modc1_2)$est$coeff.val ## gwet's ac for chunk 1
 and 2 rolling window

moderna_ac[2] <- gwet.ac1.raw(modc2_3)$est$coeff.val

moderna_ac[3] <- gwet.ac1.raw(modc3_4)$est$coeff.val

moderna_ac[4] <- gwet.ac1.raw(modc4_5)$est$coeff.val

moderna_ac[5] <- gwet.ac1.raw(modc5_6)$est$coeff.val

moderna_ac[6] <- gwet.ac1.raw(modc6_7)$est$coeff.val
```

```
moderna_ac[7] <- gwet.ac1.raw(modc7_8)$est$coeff.val

moderna_ac[8] <- gwet.ac1.raw(modc8_9)$est$coeff.val

moderna_ac[9] <- gwet.ac1.raw(modc9_10)$est$coeff.val


moderna_ac[10] <- gwet.ac1.raw(modc10_11)$est$coeff.val
```



| Token | MI Score | Token | MI Score | Token | MI Score | Token | MI Score |
|---|---|---|---|---|---|---|---|
| phony | 9.215431 | bourla | 8.354436 | climateactionnow | 10.356224 | eos | 10.254149 |
| ralph | 8.482076 | albert | 8.261327 | climatejustice | 10.356224 | fired/suspended | 10.254149 |
| 94.5 | 7.016308 | demonstrated | 7.656326 | foreigninterference | 10.356224 | reversed | 10.017110 |
| 92 | 5.952396 | quit | 7.518823 | sieg | 10.356224 | youtubea | 9.847185 |
| experienced | 5.611063 | biontechs | 7.317910 | vaticans | 10.356224 | recap | 9.813577 |
| 24/7 | 5.560079 | biontech | 7.210107 | eos | 10.133831 | betrayal | 9.444120 |
| allergic | 5.516225 | denies | 7.206879 | fired/suspended | 10.133831 | httpst.co/pxu0dq7fuw | 9.373731 |
| pfizer | 5.505362 | biontech_group | 7.138707 | reversed | 9.896792 | competency | 9.306617 |

### Plot Gwet's AC against each chunk

```
moderna_dat <- data.frame(moderna_ac, r_window = c(seq(1,10)))
ggplot(moderna_dat, aes(r_window,moderna_ac)) +
  geom_point() +
  geom_smooth() +
  ylab("Gwet's AC") +
  ggtitle("Convergence and Divergence of 'moderna' Token") +
  scale_x_continuous(name = "Rolling Window Chunk", limits = c(1,10), breaks
= seq(1,10)) +
  theme_gray()

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

### Repeat above steps for Pfizer, biden, and trump tokens

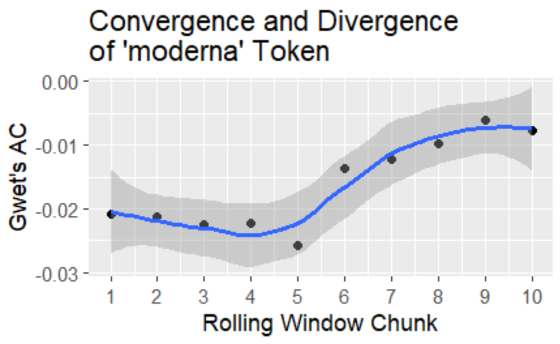
```
## create pfizer collcates for chunk 1 and chunk 2
pfizer_collocations1 <- collocates_by_MI(chunk1_tkn, "pfizer")
pfizer_collocations2 <- collocates_by_MI(chunk2_tkn, "pfizer")
pfizer_collocations3 <- collocates_by_MI(chunk3_tkn, "pfizer")
pfizer_collocations4 <- collocates_by_MI(chunk4_tkn, "pfizer")
pfizer_collocations5 <- collocates_by_MI(chunk5_tkn, "pfizer")
pfizer_collocations6 <- collocates_by_MI(chunk6_tkn, "pfizer")
pfizer_collocations7 <- collocates_by_MI(chunk7_tkn, "pfizer")
pfizer_collocations8 <- collocates_by_MI(chunk8_tkn, "pfizer")
pfizer_collocations9 <- collocates_by_MI(chunk9_tkn, "pfizer")
pfizer_collocations10 <- collocates_by_MI(chunk10_tkn, "pfizer")
pfizer_collocations11 <- collocates_by_MI(chunk11_tkn, "pfizer")

## filter out high mi's and high collocate frequency
pc1 <- pfizer_collocations1 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
pc2 <- pfizer_collocations2 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
pc3 <- pfizer_collocations3 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
```

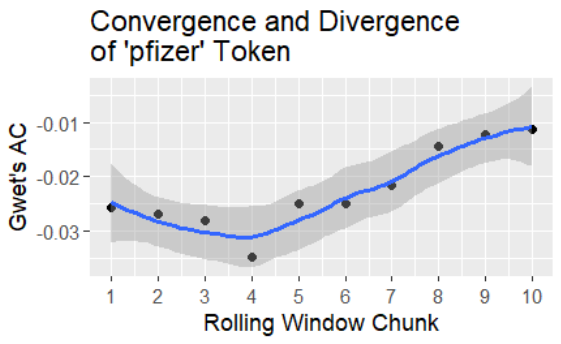
```r
pc4 <- pfizer_collocations4 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
pc5 <- pfizer_collocations5 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
pc6 <- pfizer_collocations6 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
pc7 <- pfizer_collocations7 %>% filter(col_freq >= 2 & MI_1 >= 5) %>% select
(token,MI_1)
pc8 <- pfizer_collocations8 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% select
(token,MI_1)
pc9 <- pfizer_collocations9 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% select
(token,MI_1)
pc10 <- pfizer_collocations10 %>% filter(col_freq >= 5 & MI_1 >= 5) %>% selec
t(token,MI_1)
pc11 <- pfizer_collocations11 %>% filter(col_freq >= 5 & MI_1 >= 5) %>% selec
t(token,MI_1)

## outer join to create table
pc1_2 <- merge(x = pc1, y = pc2, by = "token", all = TRUE, sort = F)
pc2_3 <- merge(x = pc2, y = pc3, by = "token", all = TRUE, sort = F)
pc3_4 <- merge(x = pc3, y = pc4, by = "token", all = TRUE, sort = F)
pc4_5 <- merge(x = pc4, y = pc5, by = "token", all = TRUE, sort = F)
pc5_6 <- merge(x = pc5, y = pc6, by = "token", all = TRUE, sort = F)
pc6_7 <- merge(x = pc6, y = pc7, by = "token", all = TRUE, sort = F)
pc7_8 <- merge(x = pc7, y = pc8, by = "token", all = TRUE, sort = F)
pc8_9 <- merge(x = pc8, y = pc9, by = "token", all = TRUE, sort = F)
pc9_10 <- merge(x = pc9, y = pc10, by = "token", all = TRUE, sort = F)
pc9_11 <- merge(x = pc10, y = pc11, by = "token", all = TRUE, sort = F)

pc1_2[is.na(pc1_2)] = 0
pc2_3[is.na(pc2_3)] = 0
pc3_4[is.na(pc3_4)] = 0
pc4_5[is.na(pc4_5)] = 0
pc5_6[is.na(pc5_6)] = 0
pc6_7[is.na(pc6_7)] = 0
pc7_8[is.na(pc7_8)] = 0
pc8_9[is.na(pc8_9)] = 0
pc9_10[is.na(pc9_10)] = 0
pc10_11[is.na(pc10_11)] = 0

pfizer_ac <- c()
pfizer_ac[1] <- gwet.ac1.raw(pc1_2)$est$coeff.val ## gwet's ac for chunk 1 an
d 2 rolling window

pfizer_ac[2] <- gwet.ac1.raw(pc2_3)$est$coeff.val

pfizer_ac[3] <- gwet.ac1.raw(pc3_4)$est$coeff.val

pfizer_ac[4] <- gwet.ac1.raw(pc4_5)$est$coeff.val

pfizer_ac[5] <- gwet.ac1.raw(pc5_6)$est$coeff.val

pfizer_ac[6] <- gwet.ac1.raw(pc6_7)$est$coeff.val
```

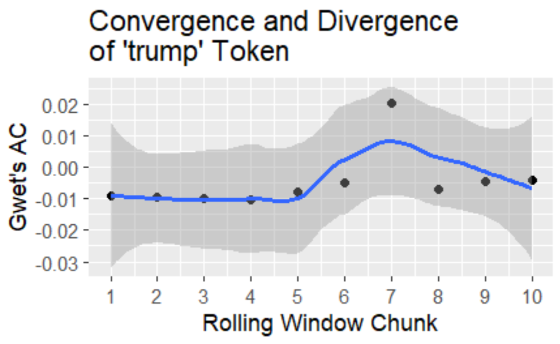
```
pfizer_ac[7] <- gwet.ac1.raw(pc7_8)$est$coeff.val

pfizer_ac[8] <- gwet.ac1.raw(pc8_9)$est$coeff.val

pfizer_ac[9] <- gwet.ac1.raw(pc9_10)$est$coeff.val

pfizer_ac[10] <- gwet.ac1.raw(pc10_1)$est$coeff.val

pfizer_data <- data.frame(pfizer_ac, r_window = c(seq(1,10)))
ggplot(pfizer_data, aes(r_window,pfizer_ac)) +
  geom_point() +
  geom_smooth() +
  ggtitle("Convergence and Divergence of 'pfizer' Token") +
  ylab("Gwet's AC") +
  scale_x_continuous(name = "Rolling Window Chunk", limits = c(1,10), breaks
= seq(1,1)) +
  theme_gray()

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## create trump collocates
trump_collocations1 <- collocates_by_MI(chunk1_tkn, "trump")
trump_collocations2 <- collocates_by_MI(chunk2_tkn, "trump")
trump_collocations3 <- collocates_by_MI(chunk3_tkn, "trump")
trump_collocations4 <- collocates_by_MI(chunk4_tkn, "trump")
trump_collocations5 <- collocates_by_MI(chunk5_tkn, "trump")
trump_collocations6 <- collocates_by_MI(chunk6_tkn, "trump")
trump_collocations7 <- collocates_by_MI(chunk7_tkn, "trump")
trump_collocations8 <- collocates_by_MI(chunk8_tkn, "trump")
trump_collocations9 <- collocates_by_MI(chunk9_tkn, "trump")
trump_collocations10 <- collocates_by_MI(chunk10_tkn, "trump")
trump_collocations11 <- collocates_by_MI(chunk11_tkn, "trump")

## filter out high mi's and high collocate frequency
tc1 <- trump_collocations1 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
tc2 <- trump_collocations2 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
tc3 <- trump_collocations3 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
tc4 <- trump_collocations4 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
tc5 <- trump_collocations5 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
tc6 <- trump_collocations6 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
tc7 <- trump_collocations7 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
tc8 <- trump_collocations8 %>% filter(col_freq >=2 & MI_1 >= 5) %>% select(to
ken,MI_1)
tc9 <- trump_collocations9 %>% filter(col_freq >= 2 & MI_1 >= 5) %>% select(t
oken,MI_1)
tc10 <- trump_collocations10 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% select
(token,MI_10)
tc11 <- trump_collocations11 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% select
(token,MI_1)
```

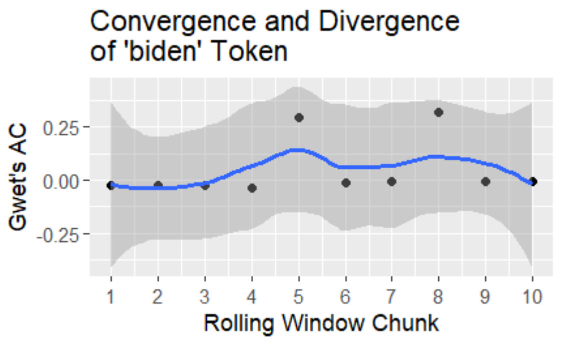
```
## outer join to create table
tc1_2 <- merge(x = tc1, y = tc2, by = "token", all = TRUE, sort = F)
tc2_3 <- merge(x = tc2, y = tc3, by = "token", all = TRUE, sort = F)
tc3_4 <- merge(x = tc3, y = tc4, by = "token", all = TRUE, sort = F)
tc4_5 <- merge(x = tc4, y = tc5, by = "token", all = TRUE, sort = F)
tc5_6 <- merge(x = tc5, y = tc6, by = "token", all = TRUE, sort = F)
tc6_7 <- merge(x = tc6, y = tc7, by = "token", all = TRUE, sort = F)
tc7_8 <- merge(x = tc7, y = tc8, by = "token", all = TRUE, sort = F)
tc8_9 <- merge(x = tc8, y = tc9, by = "token", all = TRUE, sort = F)
tc9_10 <- merge(x = tc9, y = tc10, by = "token", all = TRUE, sort = F)
tc10_11 <- merge(x = tc10, y = tc11 by = "token", all = TRUE, sort = F)

tc1_2[is.na(tc1_2)] = 0
tc2_3[is.na(tc2_3)] = 0
tc3_4[is.na(tc3_4)] = 0
tc4_5[is.na(tc4_5)] = 0
tc5_6[is.na(tc5_6)] = 0
tc6_7[is.na(tc6_7)] = 0
tc7_8[is.na(tc7_8)] = 0
tc8_9[is.na(tc8_9)] = 0
tc9_10[is.na(tc9_10)] = 0
tc10_11[is.na(tc10_11)] = 0

trump_ac <- c()
trump_ac[1] <- gwet.ac1.raw(tc1_2)$est$coeff.val ## gwet's ac for chunk 1 and
 2 rolling window

trump_ac[2] <- gwet.ac1.raw(tc2_3)$est$coeff.val

trump_ac[3] <- gwet.ac1.raw(tc3_4)$est$coeff.val

trump_ac[4] <- gwet.ac1.raw(tc4_5)$est$coeff.val

trump_ac[5] <- gwet.ac1.raw(tc5_6)$est$coeff.val

trump_ac[6] <- gwet.ac1.raw(tc6_7)$est$coeff.val

trump_ac[7] <- gwet.ac1.raw(tc7_8)$est$coeff.val

trump_ac[8] <- gwet.ac1.raw(tc8_9)$est$coeff.val

trump_ac[9] <- gwet.ac1.raw(tc9_10)$est$coeff.val

trump_ac[10] <- gwet.ac1.raw(tc10_11)$est$coeff.val

trump_data <- data.frame(trump_ac, r_window = c(seq(1,10)))

ggplot(trump_data, aes(r_window,trump_ac)) +
  geom_point() +
  geom_smooth() +
  ggtitle("Convergence and Divergence of 'trump' Token") +
  ylab("Gwet's AC") +
  scale_x_continuous(name = "Rolling Window Chunk", limits = c(1,10), breaks
= seq(1,10)) +
  theme_gray()

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## create biden collocates
biden_collocations1 <- collocates_by_MI(chunk1_tkn, "biden")
biden_collocations2 <- collocates_by_MI(chunk2_tkn, "biden")
biden_collocations3 <- collocates_by_MI(chunk3_tkn, "biden")
biden_collocations4 <- collocates_by_MI(chunk4_tkn, "biden")
biden_collocations5 <- collocates_by_MI(chunk5_tkn, "biden")
biden_collocations6 <- collocates_by_MI(chunk6_tkn, "biden")
biden_collocations7 <- collocates_by_MI(chunk7_tkn, "biden")
biden_collocations8 <- collocates_by_MI(chunk8_tkn, "biden")
biden_collocations9 <- collocates_by_MI(chunk9_tkn, "biden")
biden_collocations10 <- collocates_by_MI(chunk10_tkn, "biden")
biden_collocations11<- collocates_by_MI(chunk11_tkn, "biden"

## filter out high mi's and high collocate frequency
bc1 <- biden_collocations1 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
bc2 <- biden_collocations2 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
bc3 <- biden_collocations3 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
bc4 <- biden_collocations4 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
bc5 <- biden_collocations5 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
bc6 <- biden_collocations6 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
bc7 <- biden_collocations7 %>% filter(col_freq >=2 & MI_1 >= 5) %>% select(to
ken,MI_1)
bc8 <- biden_collocations8 %>% filter(MI_1 >= 5) %>% select(token,MI_1)
bc9 <- biden_collocations9 %>% filter(col_freq >= 2 & MI_1 >= 5) %>% select(t
oken,MI_1)
bc10 <- biden_collocations10 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% select
(token,MI_1)
bc1 <- biden_collocations11 %>% filter(col_freq >= 3 & MI_1 >= 5) %>% select
(token,MI_1)

## outer join to create table
bc1_2 <- merge(x = bc1, y = bc2, by = "token", all = TRUE, sort = F)
bc2_3 <- merge(x = bc2, y = bc3, by = "token", all = TRUE, sort = F)
bc3_4 <- merge(x = bc3, y = bc4, by = "token", all = TRUE, sort = F)
bc4_5 <- merge(x = bc4, y = bc5, by = "token", all = TRUE, sort = F)
bc5_6 <- merge(x = bc5, y = bc6, by = "token", all = TRUE, sort = F)
bc6_7 <- merge(x = bc6, y = bc7, by = "token", all = TRUE, sort = F)
bc7_8 <- merge(x = bc7, y = bc8, by = "token", all = TRUE, sort = F)
bc8_9 <- merge(x = bc8, y = bc9, by = "token", all = TRUE, sort = F)
bc9_10 <- merge(x = bc9, y = bc10, by = "token", all = TRUE, sort = F)
bc9_11 <- merge(x = bc10, y = bc11, by = "token", all = TRUE, sort = F)

bc1_2[is.na(bc1_2)] = 0
bc2_3[is.na(bc2_3)] = 0
```

```
bc3_4[is.na(bc3_4)] = 0
bc4_5[is.na(bc4_5)] = 0
bc5_6[is.na(bc5_6)] = 0
bc6_7[is.na(bc6_7)] = 0
bc7_8[is.na(bc7_8)] = 0
bc8_9[is.na(bc8_9)] = 0
bc9_10[is.na(bc9_10)] = 0
bc10_11[is.na(bc10_11)] = 0

biden_ac <- c()
biden_ac[1] <- gwet.ac1.raw(bc1_2)$est$coeff.val ## gwet's ac for chunk 1 and
 2 rolling window

biden_ac[2] <- gwet.ac1.raw(bc2_3)$est$coeff.val

biden_ac[3] <- gwet.ac1.raw(bc3_4)$est$coeff.val

biden_ac[4] <- gwet.ac1.raw(bc4_5)$est$coeff.val

biden_ac[5] <- gwet.ac1.raw(bc5_6)$est$coeff.val

biden_ac[6] <- gwet.ac1.raw(bc6_7)$est$coeff.val

biden_ac[7] <- gwet.ac1.raw(bc7_8)$est$coeff.val

biden_ac[8] <- gwet.ac1.raw(bc8_9)$est$coeff.val

biden_ac[9] <- gwet.ac1.raw(bc9_10)$est$coeff.val

biden_ac[10] <- gwet.ac1.raw(bc10_11)$est$coeff.val

biden_data <- data.frame(biden_ac, r_window = c(seq(1,10)))
ggplot(biden_data, aes(r_window,biden_ac)) +
  geom_point() +
  geom_smooth() +
  ggtitle("Convergence and Divergence of 'biden' Token") +
  ylab("Gwet's AC") +
  scale_x_continuous(name = "Rolling Window Chunk", limits = c(1,10), breaks
= seq(1,10)) +
  theme_gray()

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```


Convergence and Divergence
of 'moderna' Token
Gwet's AC
0.00
-0.01
-0.02
-0.03
1 2 3 4 5 6 7 8 9 10
Rolling Window Chunk


Convergence and Divergence
of 'pfizer' Token
Gwet's AC
-0.01
-0.02
-0.03
1 2 3 4 5 6 7 8 9 10
Rolling Window Chunk


Convergence and Divergence
of 'trump' Token
Gwet's AC
0.02
0.01
0.00
-0.01
-0.02
-0.03
1 2 3 4 5 6 7 8 9 10
Rolling Window Chunk


Convergence and Divergence
of 'biden' Token
Gwet's AC
0.25
0.00
-0.25
1 2 3 4 5 6 7 8 9 10
Rolling Window Chunk