# Tweet Assignment - ngrams

Tabitha Hagen

4/19/2023

## 1 - Load the libraries

## 2 - Read-in the following datasets in R (the dataset is available on Canvas).

```
#read in the data
tweets_original <- read_csv("Twitter Data-Corona.csv")

# tidy and view data
tweets <- tweets_original %>%
    filter(nonrts.lang=="en")%>%
     select (created_at=nonrts.created_at, text=nonrts.text)

#describe new dataframe
names(tweets) # gives the column/variable names

## [1] "created_at" "text"

dim(tweets)   # gives the observation/rows count

## [1] 964   2
```

## 3 - Describe the variables in the data set and number of observations. (10 points)

There are two columns/variables called “created_at” which is the date the tweet was created and “text” which is the content of the tweet. By filtering only English tweets, the original dataframe has been reduced from 1912 observations/rows to 964 observations/rows.

## 4 - Create a tidytext dataset of Tweets - Tokenize by bigrams. (20 points)

```
#Create a tidytext dataframe
bigrams <- tweets %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2) #create bigrams of 2
words
```

```
bigrams #view tidytext dataframe
```

```
## # A tibble: 15,017 × 2
##    created_at                     bigram
##    <chr>                          <chr>
##  1 Fri Mar 27 00:48:56 +0000 2020 at least
##  2 Fri Mar 27 00:48:56 +0000 2020 least he
##  3 Fri Mar 27 00:48:56 +0000 2020 he s
##  4 Fri Mar 27 00:48:56 +0000 2020 s being
##  5 Fri Mar 27 00:48:56 +0000 2020 being honest
##  6 Fri Mar 27 00:48:56 +0000 2020 honest u
##  7 Fri Mar 27 00:48:56 +0000 2020 u 0001f937
##  8 Fri Mar 27 00:48:56 +0000 2020 0001f937 u
##  9 Fri Mar 27 00:48:56 +0000 2020 u 0001f3fd
## 10 Fri Mar 27 00:48:56 +0000 2020 0001f3fd u
## # … with 15,007 more rows
```

## 5 - Separate the bigrams. Filter stop-words, undesirable words, and words less than 3 charcters. (30 points)

```
# Pre-process tidytext dataframe
bigrams_separated <- bigrams %>%
    # separates n-gram into n columns, "word1", "word2", .., "wordn"
    separate(bigram, c("word1", "word2"), sep = " ")


undesirable_words <-
c("https","t.co","amp","rt","tco","coronavirus","corona","covid","virus",
"ass", "fucking", "bitch", "0001f525", "horny", "tdk", "bro", "aye",
"neekolul", "12ct", "toilet", "ccp", "ccp_is_terrorist", "tempestwynn_",
"stopdlow", "q3lzl2crpd", "n04kfkdfia", "kmov", "fai", "bra", "3fle7en7nu",
"south32", "ya'll", "rendon63rd", "lorenaad80", "gavinnewsom", "ab5",
"anymore", "ocasio", "yay", "lululan09", "    tomforutah", "andrewyang",
"rand", "porky", "dab", "mrecmann", "hyundai", "alyssa_milano", "bts",
"threadreaderapp", "jamierodr14", "glov", "lmfaooo", "karaharagu1",
"jopbyfl", "iiqidaigw", "esvrmbfpb", "callmeanqiiex", "cabujnhnj",
"gqkzrfegc", "ylgukcir", "rjvgemll", "eumilsi", "sexier", "lvklqhs",
"afezlralb", "liijwmzo", "meatbjhzlf", "dwuhlfelderlaw", "lxzelujpz",
"coscklyxyr", "cvtjyjjkh", "kgqleuchk", "igwofruyc", "ofgihsj", "safjltdy",
"realcandaceo", "neqbz", "ivhfxjcp", "snoqxiwq", "euca", "oqsgopcm",
"couwmwxap", "ogxnxctmc", "mrtzgbshth", "dlvdbsxj", "ccpisterrorist",
"tempestwynn", "sgsuwbnuec", "pyxvjiprch", "aczzaq", "gxyjuhnuv", "qlzlcrpd",
"fboidbfbas", "kvlhnuue", "")

bigrams_separated$word1 <- gsub("[^a-zA-Z]","", bigrams_separated$word1) #
only use alpha characters
bigrams_separated$word2 <- gsub("[^a-zA-Z]","", bigrams_separated$word2) #
only use alpha characters
```

```
bigrams_separated$word1 <- gsub("\\s+","", bigrams_separated$word1) # get rid
of whitespace
bigrams_separated$word2 <- gsub("\\s+","", bigrams_separated$word2) # get rid
of whitespace

bigrams_filtered <- bigrams_separated %>%
  # remove undesirable_words
  filter(!word1 %in% undesirable_words) %>%
  filter(!word2 %in% undesirable_words) %>%
  # remove stop_words
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

bigrams_filtered #view separated tidytext dataframe
```

```
## # A tibble: 2,162 × 3
##    created_at                     word1     word2
##    <chr>                          <chr>     <chr>
##  1 Fri Mar 27 00:48:56 +0000 2020 septa     yall
##  2 Fri Mar 27 00:48:56 +0000 2020 america   smh
##  3 Fri Mar 27 00:48:56 +0000 2020 estimated figures
##  4 Fri Mar 27 00:48:56 +0000 2020 lab       confirmed
##  5 Fri Mar 27 00:48:56 +0000 2020 confirmed deaths
##  6 Fri Mar 27 00:48:56 +0000 2020 sharp     rise
##  7 Fri Mar 27 00:48:57 +0000 2020 lwn       repdancrenshaw
##  8 Fri Mar 27 00:48:57 +0000 2020 trillion  amid
##  9 Fri Mar 27 00:48:57 +0000 2020 stories   ranging
## 10 Fri Mar 27 00:48:57 +0000 2020 affected  campus
## # … with 2,152 more rows
```

### Count the most common bigrams.(10 points)

```
# new bigram counts
bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)
```

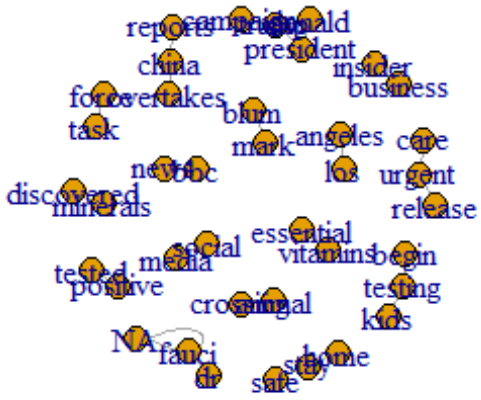
## 6 - Load the package igraph:

```
#local library to be used
library(igraph)
```

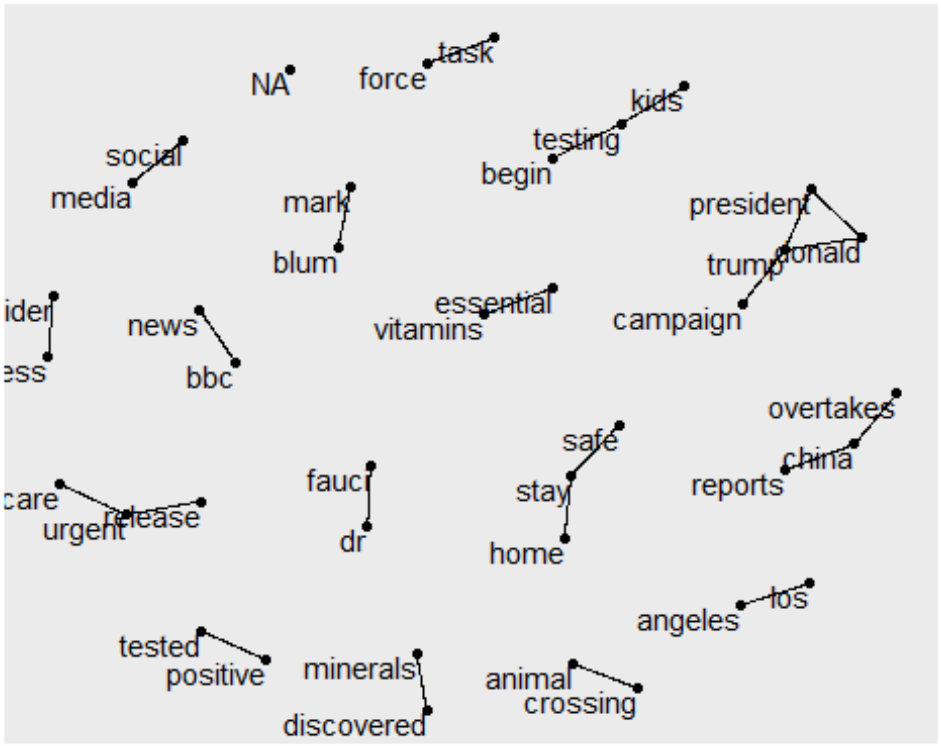
## 7 - Use the output from step 5 and build a network of common bigrams [filter for only relatively common combinations (based on n - (n>2) – use lines instead of directed arrows between nodes (graph_from_data_frame (directed = FALSE))]. (10 points)

```
# filter for only relatively common combinations
bigram_graph_common <- bigram_counts %>%
```

```
  filter(n > 2) %>% #include only repeated words
  graph_from_data_frame((directed = FALSE))

## Warning in graph_from_data_frame(., (directed = FALSE)): In `d' `NA'
elements
## were replaced with string "NA"

plot(bigram_graph_common) # view simple graph
```




```
#create a better network graph
library(ggraph)
set.seed(2016)

ggraph(bigram_graph_common, layout = "fr") +
     #add edge_alpha to make links transparent based on how common or rare
the bigram is
  geom_edge_link() +
  geom_node_point() +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1)

## Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2
3.4.0.
## ℹ Please use `linewidth` in the `default_aes` field and elsewhere instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




## 8 - Load the package ggraph

```
#local library to be used
library(ggraph)
```

## 9 - Visualize the graph - Use the Fruchterman-Reingold to visualize the nodes and ties ("fr"). Apply some polishing operations to make a better looking graph.

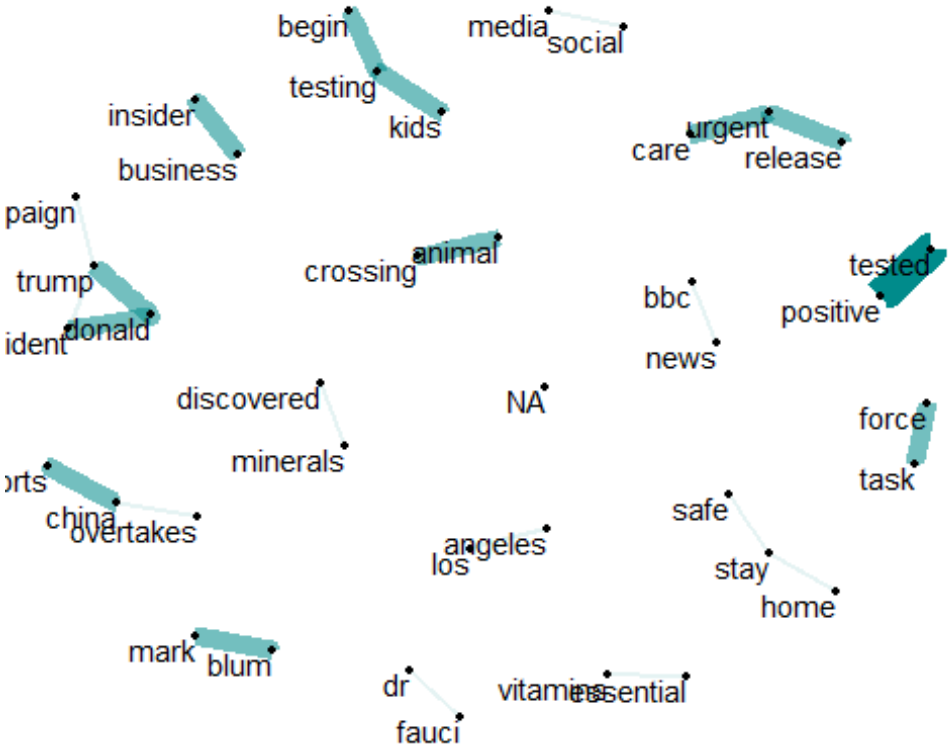
### + add the edge_alpha aesthetic to the link layer to make links transparent based on how common or rare the bigram is

### + add the edge_width aesthetic to the link layer to show the weight of the ties between bigrams

### + add a theme that's useful for plotting networks, theme_void()

```
#plot the graph of bigrams

set.seed(2017)

ggraph(bigram_graph_common, layout = "fr") +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), show.legend =
```

```
FALSE,edge_colour = "cyan4") +
  geom_node_point(size = 1) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1) +
  theme_void()
```




## 10 - Explain the results. Reflect on this assignment and provide a summary of your findings. (20 points)

The results demonstrate that the bigram “tested positive” was the most repeated bigram becaause it is in the darker, less transparent edge. Other important phrases were: " begin kids testing“, insider business”, “President Donald Trump”, “task force”, “release urgent care”, “China reports”, “Mark Blum” and surprisingly, “animal crossing” - a popular gaming app.

This assignment was truly applicable since it is hard to filter through the constant barrage of social media news. Add to the overwhlemingly amount of information given to the users, the fact that it is constantly changing is hard to track, even with machines.

Graphing using ggraph and igraph make it easier to see the connections between words or groups of words. Being able to filter out profanity in one easy step would be useful as well.