DTMSentimentAnalysis

Aaron Dantzler

2023-07-24

bing

```
# Set working directory and load data
dat <- read.csv("D:\\Princeton\\BSPL\\norms.csv")
dat <- dat[-nrow(dat), ]

## Sentiment Analysis
# Use a lexicon (bing, afinn,or nrc) to identify words in your corpus.
# This is a helpful vingette: https://cran.r-project.org/web/packages/tidytext/vignettes/tidying_castin
```

```
# 1. Load packages:
library(tidytext) # contains sentiment lexicons
```

```
## Warning: package 'tidytext' was built under R version 4.3.1
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.1

## Warning: package 'ggplot2' was built under R version 4.3.1

## Warning: package 'tidyr' was built under R version 4.3.1

## Warning: package 'stringr' was built under R version 4.3.1

## Warning: package 'forcats' was built under R version 4.3.1

## Warning: package 'lubridate' was built under R version 4.3.1

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
#for corpus prep
library(stringr)
library(tm)
```

```
## Warning: package 'tm' was built under R version 4.3.1
```

```
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(stm)
```

```
## stm v1.3.6 successfully loaded. See ?stm for help.
##  Papers, resources, and other materials at structuraltopicmodel.com
```

```
library(quanteda)
```

```
## Package version: 3.3.1
## Unicode version: 13.0
## ICU version: 69.1
## Parallel computing: 8 of 8 threads used.
## See https://quanteda.io for tutorials and examples.
##
## Attaching package: 'quanteda'
##
## The following object is masked from 'package:tm':
##
##     stopwords
##
## The following objects are masked from 'package:NLP':
##
##     meta, meta<-
```

```
library(textdata)
```

```
## Warning: package 'textdata' was built under R version 4.3.1
```

```
#for visualizaiton
library(ggplot2)
library(psych)
```

```
## Warning: package 'psych' was built under R version 4.3.1
```

```
##
## Attaching package: 'psych'
##
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(clipr)
```

```
## Welcome to clipr. See ?write_clip for advisories on writing to the clipboard in R.
```

```
# 2. Load lexicons:
#   Inspect each individually to decide which one to use for your analysis
afinn <- get_sentiments("afinn")
bing <- get_sentiments("bing")
# remove double words from bing
bing <- bing[!(bing$word=="envious" & bing$sentiment=="positive"),]
nrc <- get_sentiments("nrc")
```

T1

```
# 3. Prep data:
dat_t1 <- dat %>% select(prolific,frq_t1,frq_t2,frq_t3) %>%
  pivot_longer(cols = c(frq_t1,frq_t2,frq_t3),names_to = "time",values_to = "text")
dat_t1 <- dat_t1 %>% filter(time == 'frq_t1')
dat_l2 <- dat_t1[,c(1,3)]
names(dat_l2) <- c("doc_id","text")
docs <- VCorpus(DataframeSource(dat_l2))
```

```
#inspect the corpus
inspect(docs[[2]]) ##example, should return second document
```

```
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 396
##
## Improving access to mental healthcare is the most important to me because it affects me and people I
```

```
#clean--> use what is relevant for your analysis
# tm_map is buggy when you run it multiple times, so be sure to always save as a new object
docs1 <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs1, content_transformer(tolower))
docs3 <- tm_map(docs2, removeWords, stopwords("english"))
docs4 <- tm_map(docs3, removePunctuation)
# docs5 <- tm_map(docs4, removeNumbers) # Keep numbers because we want to see if they use statistics
# docs6 <- tm_map(docs4, content_transformer(gsub), pattern = "climate change", replacement = # "climat
# docs7 <- tm_map(docs6, content_transformer(gsub), pattern = "global warming", replacement = # "global
docs8 <- tm_map(docs4, content_transformer(gsub), pattern = "\"", replacement = "", docs1)
docs9 <- tm_map(docs8, content_transformer(gsub), pattern = "'", replacement = "", docs1)
rm(docs1,docs2,docs3,docs4,docs8)
```

```
# make dtm
dtm <- DocumentTermMatrix(docs9)
```

```r
# 4. Join to sentiment lexicon:
#   Pick the lexicon that you want to use
c <- tidy(dtm) ##input: a document-term matrix
```

```r
#join the table to the bing lexicon
c_sentiments <- c %>%
  left_join(bing, by = c("term" = "word"))
```

```r
#inspect the join
c_sentiments
```

```
## # A tibble: 20,167 x 4
##    document                 term       count sentiment
##    <chr>                    <chr>      <dbl> <chr>
##  1 546e3778fdf99b2bc7ebcff6 businesses     1 <NA>
##  2 546e3778fdf99b2bc7ebcff6 even           1 <NA>
##  3 546e3778fdf99b2bc7ebcff6 federal        1 <NA>
##  4 546e3778fdf99b2bc7ebcff6 hard           1 negative
##  5 546e3778fdf99b2bc7ebcff6 heard          1 <NA>
##  6 546e3778fdf99b2bc7ebcff6 hour           3 <NA>
##  7 546e3778fdf99b2bc7ebcff6 inflation      1 <NA>
##  8 546e3778fdf99b2bc7ebcff6 irs            1 <NA>
##  9 546e3778fdf99b2bc7ebcff6 know           1 <NA>
## 10 546e3778fdf99b2bc7ebcff6 living         2 <NA>
## # i 20,157 more rows
```

```r
##aggregate sentiment to the document level
c_sent_by_doct1 <- c_sentiments %>%
  count(document, sentiment, wt = count) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative) # %>%
  # arrange(sentiment)

c_sent_by_doct1
```

```
## # A tibble: 616 x 5
##    document                 negative positive `<NA>` sentiment
##    <chr>                       <dbl>    <dbl>  <dbl>     <dbl>
##  1 546e3778fdf99b2bc7ebcff6        1        0     32        -1
##  2 55519750fdf99b7f2114cc3e        1        6     28         5
##  3 55a29659fdf99b5ff49937d3        5        2     34        -3
##  4 55b9a9b0fdf99b6906d2aba4        1        1     33         0
##  5 55c43918fdf99b080551e044       11        2     49        -9
##  6 55d0945934e9060005e57258        4        1     21        -3
##  7 55d35447da14d7000e95318f        4        3     38        -1
##  8 55e5d5fdc70c7a000b2a5bac        7        1     28        -6
##  9 56259489ed6e5a0005c80fde        0        2     28         2
## 10 565e0169998e44000a4bdab1        1        1     26         0
## # i 606 more rows
```

T2

```
# 3. Prep data:
dat_t2 <- dat %>% select(prolific,frq_t1,frq_t2,frq_t3) %>%
  pivot_longer(cols = c(frq_t1,frq_t2,frq_t3),names_to = "time",values_to = "text")
dat_t2 <- dat_t2 %>% filter(time == 'frq_t2')
dat_l2 <- dat_t2[,c(1,3)]
names(dat_l2) <- c("doc_id","text")
docs <- VCorpus(DataframeSource(dat_l2))
```

```
#inspect the corpus
inspect(docs[[2]]) ##example, should return second document
```

```
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 488
##
## Increasing the federal minimum wage would greatly help many people across the nation. Not only would
```

```
#clean--> use what is relevant for your analysis
# tm_map is buggy when you run it multiple times, so be sure to always save as a new object
docs1 <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs1, content_transformer(tolower))
docs3 <- tm_map(docs2, removeWords, stopwords("english"))
docs4 <- tm_map(docs3, removePunctuation)
# docs5 <- tm_map(docs4, removeNumbers) # Keep numbers because we want to see if they use statistics
# docs6 <- tm_map(docs4, content_transformer(gsub), pattern = "climate change", replacement = # "climat
# docs7 <- tm_map(docs6, content_transformer(gsub), pattern = "global warming", replacement = # "global
docs8 <- tm_map(docs4, content_transformer(gsub), pattern = "\"", replacement = "", docs1)
docs9 <- tm_map(docs8, content_transformer(gsub), pattern = "'", replacement = "", docs1)
rm(docs1,docs2,docs3,docs4,docs8)
```

```
# make dtm
dtm <- DocumentTermMatrix(docs9)
```

```
# 4. Join to sentiment lexicon:
#   Pick the lexicon that you want to use
c <- tidy(dtm) ##input: a document-term matrix
```

```
#join the table to the bing lexicon
c_sentiments <- c %>%
  left_join(bing, by = c("term" = "word"))
```

```
#inspect the join
c_sentiments
```

```
## # A tibble: 19,642 x 4
##    document                 term     count sentiment
##    <chr>                    <chr>    <dbl> <chr>
##  1 546e3778fdf99b2bc7ebcff6 also         2 <NA>
```

```
##  2 546e3778fdf99b2bc7ebcff6 business     1 <NA>
##  3 546e3778fdf99b2bc7ebcff6 dear         1 <NA>
##  4 546e3778fdf99b2bc7ebcff6 enough       2 positive
##  5 546e3778fdf99b2bc7ebcff6 federal      1 <NA>
##  6 546e3778fdf99b2bc7ebcff6 hard         2 negative
##  7 546e3778fdf99b2bc7ebcff6 help         1 <NA>
##  8 546e3778fdf99b2bc7ebcff6 hour         1 <NA>
##  9 546e3778fdf99b2bc7ebcff6 know         1 <NA>
## 10 546e3778fdf99b2bc7ebcff6 living       2 <NA>
## # i 19,632 more rows
```

```
##aggregate sentiment to the document level
c_sent_by_doct2 <- c_sentiments %>%
  count(document, sentiment, wt = count) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative) # %>%
  # arrange(sentiment)

c_sent_by_doct2
```

```
## # A tibble: 616 x 5
##    document                 negative positive `<NA>` sentiment
##    <chr>                       <dbl>    <dbl>  <dbl>     <dbl>
##  1 546e3778fdf99b2bc7ebcff6        2        2     36         0
##  2 55519750fdf99b7f2114cc3e        0        2     47         2
##  3 55a29659fdf99b5ff49937d3        8        0     24        -8
##  4 55b9a9b0fdf99b6906d2aba4        6        1     26        -5
##  5 55c43918fdf99b080551e044        2        8     57         6
##  6 55d0945934e9060005e57258        9        0     20        -9
##  7 55d35447da14d7000e95318f        5        2     30        -3
##  8 55e5d5fdc70c7a000b2a5bac        2        4     43         2
##  9 56259489ed6e5a0005c80fde        2        1     32        -1
## 10 565e0169998e44000a4bdab1        4        1     46        -3
## # i 606 more rows
```

T3

```
# 3. Prep data:
dat_t3 <- dat %>% select(prolific,frq_t1,frq_t2,frq_t3) %>%
  pivot_longer(cols = c(frq_t1,frq_t2,frq_t3),names_to = "time",values_to = "text")
dat_t3 <- dat_t3 %>% filter(time == 'frq_t3')
dat_l2 <- dat_t3[,c(1,3)]
names(dat_l2) <- c("doc_id","text")
docs <- VCorpus(DataframeSource(dat_l2))
```

```
#inspect the corpus
inspect(docs[[2]]) ##example, should return second document
```

```
## <<PlainTextDocument>>
## Metadata:  7
```

```
## Content:  chars: 385
##
## Our country is experiencing a mental health crisis. There are so many people out there who need menta
```

```
#clean--> use what is relevant for your analysis
# tm_map is buggy when you run it multiple times, so be sure to always save as a new object
docs1 <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs1, content_transformer(tolower))
docs3 <- tm_map(docs2, removeWords, stopwords("english"))
docs4 <- tm_map(docs3, removePunctuation)
# docs5 <- tm_map(docs4, removeNumbers) # Keep numbers because we want to see if they use statistics
# docs6 <- tm_map(docs4, content_transformer(gsub), pattern = "climate change", replacement = # "climat
# docs7 <- tm_map(docs6, content_transformer(gsub), pattern = "global warming", replacement = # "global
docs8 <- tm_map(docs4, content_transformer(gsub), pattern = "\"", replacement = "", docs1)
docs9 <- tm_map(docs8, content_transformer(gsub), pattern = "'", replacement = "", docs1)
rm(docs1,docs2,docs3,docs4,docs8)
```

```
# make dtm
dtm <- DocumentTermMatrix(docs9)
```

```
# 4. Join to sentiment lexicon:
#   Pick the lexicon that you want to use
c <- tidy(dtm) ##input: a document-term matrix
```

```
#join the table to the bing lexicon
c_sentiments <- c %>%
  left_join(bing, by = c("term" = "word"))
```

```
#inspect the join
c_sentiments
```

```
## # A tibble: 19,908 x 4
##    document                 term    count sentiment
##    <chr>                    <chr>   <dbl> <chr>
##  1 546e3778fdf99b2bc7ebcff6 behind      1 <NA>
##  2 546e3778fdf99b2bc7ebcff6 car         1 <NA>
##  3 546e3778fdf99b2bc7ebcff6 drive       2 <NA>
##  4 546e3778fdf99b2bc7ebcff6 driving     1 <NA>
##  5 546e3778fdf99b2bc7ebcff6 drunk       3 negative
##  6 546e3778fdf99b2bc7ebcff6 dui         1 <NA>
##  7 546e3778fdf99b2bc7ebcff6 get         1 <NA>
##  8 546e3778fdf99b2bc7ebcff6 harder      2 <NA>
##  9 546e3778fdf99b2bc7ebcff6 help        1 <NA>
## 10 546e3778fdf99b2bc7ebcff6 kills       1 negative
## # i 19,898 more rows
```

```
##aggregate sentiment to the document level
c_sent_by_doct3 <- c_sentiments %>%
  count(document, sentiment, wt = count) %>%
  spread(sentiment, n, fill = 0) %>%
  mutate(sentiment = positive - negative) # %>%
  # arrange(sentiment)
```

```
c_sent_by_doct3
```

```
## # A tibble: 616 x 5
##    document                 negative positive `<NA>` sentiment
##    <chr>                       <dbl>    <dbl>  <dbl>     <dbl>
##  1 546e3778fdf99b2bc7ebcff6        5        0     24        -5
##  2 55519750fdf99b7f2114cc3e        1        0     30        -1
##  3 55a29659fdf99b5ff49937d3        5        2     31        -3
##  4 55b9a9b0fdf99b6906d2aba4        4        1     30        -3
##  5 55c43918fdf99b080551e044        2        4     71         2
##  6 55d0945934e9060005e57258        6        0     23        -6
##  7 55d35447da14d7000e95318f        5        3     23        -2
##  8 55e5d5fdc70c7a000b2a5bac        4        1     36        -3
##  9 56259489ed6e5a0005c80fde        6        7     37         1
## 10 565e0169998e44000a4bdab1        5        4     41        -1
## # i 606 more rows
```

```
c_sent_by_doct1 <- c_sent_by_doct1 %>%
  rename(negativet1 = negative, positivet1 = positive, neutralt1 = "<NA>",
         sentimentt1 = sentiment)
c_sent_by_doct2 <- c_sent_by_doct2 %>%
  rename(negativet2 = negative, positivet2 = positive, neutralt2 = "<NA>",
         sentimentt2 = sentiment)
c_sent_by_doct3 <- c_sent_by_doct3 %>%
  rename(negativet3 = negative, positivet3 = positive, neutralt3 = "<NA>",
         sentimentt3 = sentiment)
```

```
c_sent_by_doct2 <- c_sent_by_doct2 %>% select(-document)
c_sent_by_doct3 <- c_sent_by_doct3 %>% select(-document)
```

```
sent_combined <- bind_cols(c_sent_by_doct1, c_sent_by_doct2, c_sent_by_doct3)
```

```
sent_combined <- sent_combined %>% select(-document)
```

```
dat_sent <- bind_cols(dat, sent_combined)
```

```
file_path <- "D:\\Princeton\\BSPL\\norms_sents_bing.csv"

write.csv(dat_sent, file = file_path, row.names = FALSE)
```

afinn

```
# Set working directory and load data
dat <- read.csv("D:\\Princeton\\BSPL\\norms.csv")
dat <- dat[-nrow(dat), ]

## Sentiment Analysis
# Use a lexicon (bing, afinn,or nrc) to identify words in your corpus.
# This is a helpful vingette: https://cran.r-project.org/web/packages/tidytext/vignettes/tidying_castin
```

```r
# 1. Load packages:
library(tidytext) # contains sentiment lexicons
library(tidyverse)

#for corpus prep
library(stringr)
library(tm)
library(stm)
library(quanteda)
library(textdata)

#for visualizaiton
library(ggplot2)
library(psych)
library(clipr)
```

```r
# 2. Load lexicons:
#   Inspect each individually to decide which one to use for your analysis
afinn <- get_sentiments("afinn")
bing <- get_sentiments("bing")
# remove double words from bing
bing <- bing[!(bing$word=="envious" & bing$sentiment=="positive"),]
nrc <- get_sentiments("nrc")
```

T1

```r
# 3. Prep data:
dat_t1 <- dat %>% select(prolific,frq_t1,frq_t2,frq_t3) %>%
  pivot_longer(cols = c(frq_t1,frq_t2,frq_t3),names_to = "time",values_to = "text")
dat_t1 <- dat_t1 %>% filter(time == 'frq_t1')
dat_l2 <- dat_t1[,c(1,3)]
names(dat_l2) <- c("doc_id","text")
docs <- VCorpus(DataframeSource(dat_l2))
```

```r
#inspect the corpus
inspect(docs[[2]]) ##example, should return second document
```

```
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 396
##
## Improving access to mental healthcare is the most important to me because it affects me and people I
```

```r
#clean--> use what is relevant for your analysis
# tm_map is buggy when you run it multiple times, so be sure to always save as a new object
docs1 <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs1, content_transformer(tolower))
docs3 <- tm_map(docs2, removeWords, stopwords("english"))
docs4 <- tm_map(docs3, removePunctuation)
```

```
# docs5 <- tm_map(docs4, removeNumbers) # Keep numbers because we want to see if they use statistics
# docs6 <- tm_map(docs4, content_transformer(gsub), pattern = "climate change", replacement = # "climat
# docs7 <- tm_map(docs6, content_transformer(gsub), pattern = "global warming", replacement = # "global
docs8 <- tm_map(docs4, content_transformer(gsub), pattern = "\"", replacement = "", docs1)
docs9 <- tm_map(docs8, content_transformer(gsub), pattern = "'", replacement = "", docs1)
rm(docs1,docs2,docs3,docs4,docs8)
```

```
# make dtm
dtm <- DocumentTermMatrix(docs9)
```

```
# 4. Join to sentiment lexicon:
#   Pick the lexicon that you want to use
c <- tidy(dtm) ##input: a document-term matrix
```

```
#join the table to the afinn lexicon
c_sentiments <- c %>%
  left_join(afinn, by = c("term" = "word"))

# Replace NA with 0 in a specific column
c_sentiments$value[is.na(c_sentiments$value)] <- 0
```

```
#inspect the join
c_sentiments
```

```
## # A tibble: 20,167 x 4
##    document                 term       count value
##    <chr>                    <chr>      <dbl> <dbl>
##  1 546e3778fdf99b2bc7ebcff6 businesses     1     0
##  2 546e3778fdf99b2bc7ebcff6 even           1     0
##  3 546e3778fdf99b2bc7ebcff6 federal        1     0
##  4 546e3778fdf99b2bc7ebcff6 hard           1    -1
##  5 546e3778fdf99b2bc7ebcff6 heard          1     0
##  6 546e3778fdf99b2bc7ebcff6 hour           3     0
##  7 546e3778fdf99b2bc7ebcff6 inflation      1     0
##  8 546e3778fdf99b2bc7ebcff6 irs            1     0
##  9 546e3778fdf99b2bc7ebcff6 know           1     0
## 10 546e3778fdf99b2bc7ebcff6 living         2     0
## # i 20,157 more rows
```

```
##aggregate sentiment to the document level
c_sent_by_doct1 <- c_sentiments %>%
  group_by(document) %>%
  summarise(combined_sentiment = sum(value))


c_sent_by_doct1
```

```
## # A tibble: 616 x 2
##    document                 combined_sentiment
##    <chr>                                 <dbl>
##  1 546e3778fdf99b2bc7ebcff6                 -2
```

```
##  2 55519750fdf99b7f2114cc3e                21
##  3 55a29659fdf99b5ff49937d3                 2
##  4 55b9a9b0fdf99b6906d2aba4                -3
##  5 55c43918fdf99b080551e044                -8
##  6 55d0945934e9060005e57258                 2
##  7 55d35447da14d7000e95318f                 0
##  8 55e5d5fdc70c7a000b2a5bac               -18
##  9 56259489ed6e5a0005c80fde                 5
## 10 565e0169998e44000a4bdab1                -2
## # i 606 more rows
```

T2

```
# 3. Prep data:
dat_t2 <- dat %>% select(prolific,frq_t1,frq_t2,frq_t3) %>%
  pivot_longer(cols = c(frq_t1,frq_t2,frq_t3),names_to = "time",values_to = "text")
dat_t2 <- dat_t2 %>% filter(time == 'frq_t2')
dat_l2 <- dat_t2[,c(1,3)]
names(dat_l2) <- c("doc_id","text")
docs <- VCorpus(DataframeSource(dat_l2))
```

```
#inspect the corpus
inspect(docs[[2]]) ##example, should return second document
```

```
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 488
##
## Increasing the federal minimum wage would greatly help many people across the nation. Not only would
```

```
#clean--> use what is relevant for your analysis
# tm_map is buggy when you run it multiple times, so be sure to always save as a new object
docs1 <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs1, content_transformer(tolower))
docs3 <- tm_map(docs2, removeWords, stopwords("english"))
docs4 <- tm_map(docs3, removePunctuation)
# docs5 <- tm_map(docs4, removeNumbers) # Keep numbers because we want to see if they use statistics
# docs6 <- tm_map(docs4, content_transformer(gsub), pattern = "climate change", replacement = # "climat
# docs7 <- tm_map(docs6, content_transformer(gsub), pattern = "global warming", replacement = # "global
docs8 <- tm_map(docs4, content_transformer(gsub), pattern = "\"", replacement = "", docs1)
docs9 <- tm_map(docs8, content_transformer(gsub), pattern = "'", replacement = "", docs1)
rm(docs1,docs2,docs3,docs4,docs8)
```

```
# make dtm
dtm <- DocumentTermMatrix(docs9)
```

```
# 4. Join to sentiment lexicon:
#   Pick the lexicon that you want to use
c <- tidy(dtm) ##input: a document-term matrix
```

```r
#join the table to the afinn lexicon
c_sentiments <- c %>%
  left_join(afinn, by = c("term" = "word"))

# Replace NA with 0 in a specific column
c_sentiments$value[is.na(c_sentiments$value)] <- 0
```

```r
#inspect the join
c_sentiments
```

```
## # A tibble: 19,642 x 4
##    document                 term     count value
##    <chr>                    <chr>    <dbl> <dbl>
##  1 546e3778fdf99b2bc7ebcff6 also         2     0
##  2 546e3778fdf99b2bc7ebcff6 business     1     0
##  3 546e3778fdf99b2bc7ebcff6 dear         1     2
##  4 546e3778fdf99b2bc7ebcff6 enough       2     0
##  5 546e3778fdf99b2bc7ebcff6 federal      1     0
##  6 546e3778fdf99b2bc7ebcff6 hard         2    -1
##  7 546e3778fdf99b2bc7ebcff6 help         1     2
##  8 546e3778fdf99b2bc7ebcff6 hour         1     0
##  9 546e3778fdf99b2bc7ebcff6 know         1     0
## 10 546e3778fdf99b2bc7ebcff6 living       2     0
## # i 19,632 more rows
```

```r
##aggregate sentiment to the document level
c_sent_by_doct2 <- c_sentiments %>%
  group_by(document) %>%
  summarise(combined_sentiment = sum(value))


c_sent_by_doct2
```

```
## # A tibble: 616 x 2
##    document                 combined_sentiment
##    <chr>                                 <dbl>
##  1 546e3778fdf99b2bc7ebcff6                  2
##  2 55519750fdf99b7f2114cc3e                  7
##  3 55a29659fdf99b5ff49937d3                -13
##  4 55b9a9b0fdf99b6906d2aba4                 -6
##  5 55c43918fdf99b080551e044                  1
##  6 55d0945934e9060005e57258                -15
##  7 55d35447da14d7000e95318f                 -8
##  8 55e5d5fdc70c7a000b2a5bac                  1
##  9 56259489ed6e5a0005c80fde                  5
## 10 565e0169998e44000a4bdab1                 -6
## # i 606 more rows
```

T3

```r
# 3. Prep data:
dat_t3 <- dat %>% select(prolific,frq_t1,frq_t2,frq_t3) %>%
  pivot_longer(cols = c(frq_t1,frq_t2,frq_t3),names_to = "time",values_to = "text")
dat_t3 <- dat_t3 %>% filter(time == 'frq_t3')
dat_l2 <- dat_t3[,c(1,3)]
names(dat_l2) <- c("doc_id","text")
docs <- VCorpus(DataframeSource(dat_l2))
```

```r
#inspect the corpus
inspect(docs[[2]]) ##example, should return second document
```

```
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 385
##
## Our country is experiencing a mental health crisis. There are so many people out there who need menta
```

```r
#clean--> use what is relevant for your analysis
# tm_map is buggy when you run it multiple times, so be sure to always save as a new object
docs1 <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs1, content_transformer(tolower))
docs3 <- tm_map(docs2, removeWords, stopwords("english"))
docs4 <- tm_map(docs3, removePunctuation)
# docs5 <- tm_map(docs4, removeNumbers) # Keep numbers because we want to see if they use statistics
# docs6 <- tm_map(docs4, content_transformer(gsub), pattern = "climate change", replacement = # "climat
# docs7 <- tm_map(docs6, content_transformer(gsub), pattern = "global warming", replacement = # "global
docs8 <- tm_map(docs4, content_transformer(gsub), pattern = "\"", replacement = "", docs1)
docs9 <- tm_map(docs8, content_transformer(gsub), pattern = "'", replacement = "", docs1)
rm(docs1,docs2,docs3,docs4,docs8)
```

```r
# make dtm
dtm <- DocumentTermMatrix(docs9)
```

```r
# 4. Join to sentiment lexicon:
#   Pick the lexicon that you want to use
c <- tidy(dtm) ##input: a document-term matrix
```

```r
#join the table to the afinn lexicon
c_sentiments <- c %>%
  left_join(afinn, by = c("term" = "word"))

# Replace NA with 0 in a specific column
c_sentiments$value[is.na(c_sentiments$value)] <- 0
```

```r
#inspect the join
c_sentiments
```

```
## # A tibble: 19,908 x 4
##    document                 term    count value
##    <chr>                    <chr>   <dbl> <dbl>
##  1 546e3778fdf99b2bc7ebcff6 behind      1     0
```

```
##  2 546e3778fdf99b2bc7ebcff6 car         1     0
##  3 546e3778fdf99b2bc7ebcff6 drive       2     0
##  4 546e3778fdf99b2bc7ebcff6 driving     1     0
##  5 546e3778fdf99b2bc7ebcff6 drunk       3    -2
##  6 546e3778fdf99b2bc7ebcff6 dui         1     0
##  7 546e3778fdf99b2bc7ebcff6 get         1     0
##  8 546e3778fdf99b2bc7ebcff6 harder      2     0
##  9 546e3778fdf99b2bc7ebcff6 help        1     2
## 10 546e3778fdf99b2bc7ebcff6 kills       1    -3
## # i 19,898 more rows
```

```
##aggregate sentiment to the document level
c_sent_by_doct3 <- c_sentiments %>%
  group_by(document) %>%
  summarise(combined_sentiment = sum(value))


c_sent_by_doct3
```

```
## # A tibble: 616 x 2
##    document                 combined_sentiment
##    <chr>                                 <dbl>
##  1 546e3778fdf99b2bc7ebcff6                 -2
##  2 55519750fdf99b7f2114cc3e                 -2
##  3 55a29659fdf99b5ff49937d3                  5
##  4 55b9a9b0fdf99b6906d2aba4                  3
##  5 55c43918fdf99b080551e044                  2
##  6 55d0945934e9060005e57258                -12
##  7 55d35447da14d7000e95318f                 -7
##  8 55e5d5fdc70c7a000b2a5bac                 -5
##  9 56259489ed6e5a0005c80fde                  5
## 10 565e0169998e44000a4bdab1                 -4
## # i 606 more rows
```

```
c_sent_by_doct1 <- c_sent_by_doct1 %>%
  rename(combined_sentimentt1 = combined_sentiment)
c_sent_by_doct2 <- c_sent_by_doct2 %>%
  rename(combined_sentimentt2 = combined_sentiment)
c_sent_by_doct3 <- c_sent_by_doct3 %>%
  rename(combined_sentimentt3 = combined_sentiment)
```

```
c_sent_by_doct2 <- c_sent_by_doct2 %>% select(-document)
c_sent_by_doct3 <- c_sent_by_doct3 %>% select(-document)
```

```
sent_combined <- bind_cols(c_sent_by_doct1, c_sent_by_doct2, c_sent_by_doct3)
```

```
sent_combined <- sent_combined %>% select(-document)
```

```
dat_sent <- bind_cols(dat, sent_combined)
```

```
file_path <- "D:\\Princeton\\BSPL\\norms_sents_afinn.csv"

write.csv(dat_sent, file = file_path, row.names = FALSE)
```