Sentiment Analysis and Dictionary Methods

Loren Collingwood

This script reviews how to use the Lexicoder sentiment dictionary within quanteda and also some common sentiment dictionaries from the tidytext package. Users can also easily create their own dictionaries (i.e., see Oskooii, Lajevardi, and Collingwood 2019). The most important part when conducting sentiment analysis is to line your text up with dictionaries that have been developed for similar types of text. Otherwise you are encountering an apples/oranges or garbage in garbage out situation.

```r
options(scipen = 999, digits = 4)

########################
#       Packages       #
########################

library(quanteda)
```

```
## Package version: 2.1.1
## Parallel computing: 2 of 8 threads used.
## See https://quanteda.io for tutorials and examples.
##
## Attaching package: 'quanteda'
## The following object is masked from 'package:utils':
##
##     View
```

```r
#install.packages("descr")
library(descr)
#install.packages("ggplot2)
library(ggplot2)

#########################
# Set Working Directory #
#########################

setwd("~/Dropbox/collingwood_research/posc_fall_20/POSC-207/lecture")
```

Step 1

Use the lexicoder dictionary of negativity and positivity, first read in the Clicks4Kass dataset.

```r
# Read in the #Clicks4Kass Corpus #
clicks <- read.csv("Clicks.csv", header=T)

# Reduce the number of possible retweets for now #
```

```r
clicks <- clicks[clicks$retweets_count < 1,]

# Convert to Corpus #
click_corp <- corpus(clicks$tweet)
```

Step 2

Take a look at the tokens, what is getting converted. Do this to ensure that what you are doing has face validity, etc.

```r
# look at tokens, nicely #
tok_look <- tokens_lookup(tokens(click_corp),
                    dictionary = data_dictionary_LSD2015,
                    exclusive = FALSE,
                    nested_scope = "dictionary")
tok_look[[3]]
```

```
##  [1] "And"              "POSITIVE"        "to"              "@JohnHolbein1"
##  [5] "for"              "getting"         "wrapped"         "into"
##  [9] "this"             "#Clicks4Kass"    "debate"          "over"
## [13] "NEGATIVE"         "refs"            "."               "@SergioGarciaRs"
## [17] "or"               "@hlw_phd"        "can"             "back"
## [21] "me"               "up"              "."               "."
## [25] "."                "."               "or"              "@lorenc2"
```

Step 3

Deal with compounds – negative negatives and negative positives so you can then subtract that later (e.g., the biscuits are NOT good).

```r
# Compound neg_negative and neg_positive tokens before creating a dfm object
toks <- tokens_compound(tokens(click_corp),
                        data_dictionary_LSD2015)
```

Step 4

Generate a document term matrix that is just based on sentiment scores

```r
# Create the DFM #
c_dfm <- dfm_lookup(dfm(toks),
                    data_dictionary_LSD2015)

# Convert the Document Term Matrix to Sentiment data.frame() object #
sent_data <- convert(c_dfm, "data.frame")
```

Step 5

Do some addition and subtraction, and also add on user name from corpus

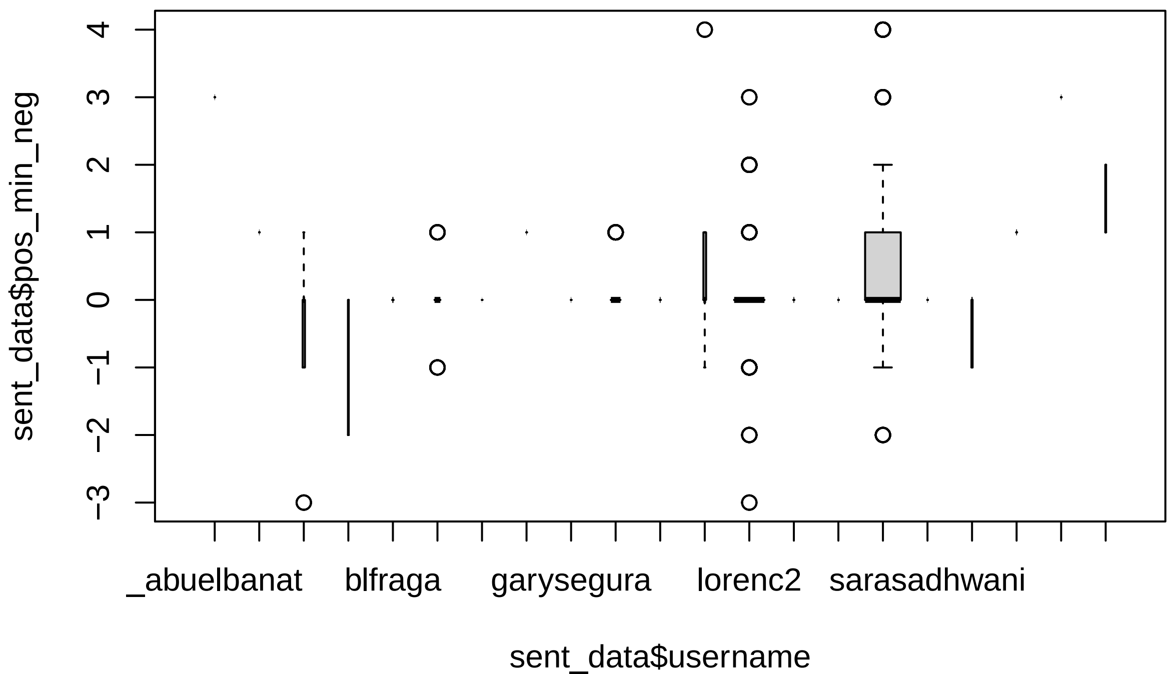
```r
sent_data$pos_final <- with(sent_data, positive - neg_positive)
sent_data$neg_final <- with(sent_data, negative - neg_negative)
sent_data$pos_min_neg <- with(sent_data, pos_final - neg_final)
sent_data$username <- clicks$username
```

Step 6

Look at sentiment by user, or any other grouping of interest

```r
out <- as.data.frame(compmeans(sent_data$pos_min_neg, sent_data$username))
```

```
## Warning in compmeans(sent_data$pos_min_neg, sent_data$username): Warning:
## "sent_data$username" was converted into factor!
```




```r
# Order it real good #
out <- out[order(out$Mean),]
out
```

```
##                      Mean   N Std. Dev.
## b_a_fitzgerald   -1.00000   6    1.0954
## angelaxocampo    -0.50000  18    1.2948
## sergiogarciars   -0.33333   9    0.5000
## blfraga           0.00000   9    0.0000
## buzznet           0.00000   1       NaN
## garysegura        0.00000   3    0.0000
## karamdana         0.00000   6    0.0000
## nazitalajevardi   0.00000   6    0.0000
## quicopedraza      0.00000   3    0.0000
## sarasadhwani      0.00000   3    0.0000
## bryanmwilcox      0.06667  45    0.5800
## lorenc2           0.13924 237    0.9261
## hlw_phd           0.24000  75    0.4300
```

```
## Total             0.29799 745    1.0548
## realmabarreto     0.48387 279    1.2080
## kassrao           0.57143  21    1.5353
## almjr80           1.00000   3    0.0000
## fabianneuner      1.00000   3    0.0000
## shortle           1.00000   6    0.0000
## wearepriec        1.50000   6    0.5477
## _abuelbanat       3.00000   3    0.0000
## skdreier24        3.00000   3    0.0000
```

Now take a look at the tidytext textdata example:

```
#################################################################
# Using the tidytext and textdata package to analyze sentiment #
#################################################################

#install.packages("tidytext")
#install.packages("textdata")
#install.packages("dplyr")

library(tidytext); library(textdata)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
nrc <- get_sentiments("nrc")

# Look at the sentiment-type words #
table(nrc$sentiment)
```

```
##
##        anger anticipation      disgust         fear          joy     negative
##         1247          839         1058         1476          689         3324
##     positive      sadness     surprise        trust
##         2312         1191          534         1231
```

```
# Plotting it out #
tidy_kass_tweets<- clicks %>%
    select(id, date, user_id, tweet) %>%
    unnest_tokens("word", tweet)

# Negative #
kass_sentiment_plot <-
    tidy_kass_tweets %>%
    inner_join(get_sentiments("nrc")) %>%
    filter(sentiment=="negative") %>%
    count(date, sentiment)
```

```
## Joining, by = "word"
```

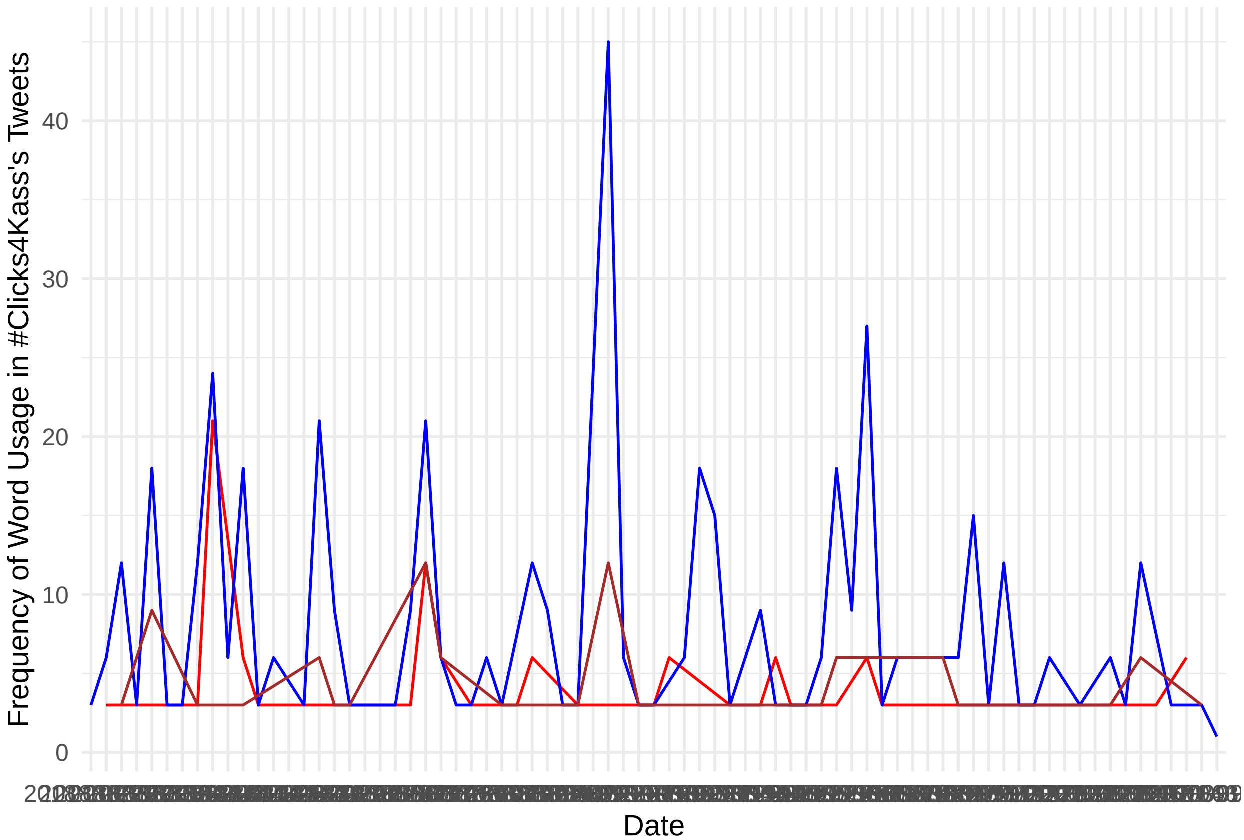
```r
# Positive #
kass_sentiment_plot_pos <-
    tidy_kass_tweets %>%
    inner_join(get_sentiments("nrc")) %>%
    filter(sentiment=="positive") %>%
    count(date, sentiment)
```

```
## Joining, by = "word"
```

```r
# Joy #
kass_sentiment_plot_joy <-
    tidy_kass_tweets %>%
    inner_join(get_sentiments("nrc")) %>%
    filter(sentiment=="joy") %>%
    count(date, sentiment)
```

```
## Joining, by = "word"
```

```r
# Plotting with ggplot #
ggplot() +
       aes(x=kass_sentiment_plot$date,
           y=kass_sentiment_plot$n, group = 1)+
    geom_line(color="red")+
    theme_minimal()+
    ylab("Frequency of Word Usage in #Clicks4Kass's Tweets") +
    xlab("Date") +
    geom_line(aes(x=kass_sentiment_plot_pos$date,
                  y = kass_sentiment_plot_pos$n, group = 1),
              color='blue') +
    geom_line(aes(x=kass_sentiment_plot_joy$date,
              y = kass_sentiment_plot_joy$n, group = 1),
          color='brown')
```



Now let's take a real life example of one of Dr. Collingwood's eminent publications (with Kassra Oskooii and Nazita Lajevardi). Here we are making an over time media narrative/theme argument and connecting those findings to a three-wave panel. In this article, we constructed our own dictionary (Oskooii et al 2019, reverse-alphabetical.).

```
load ( url("https://www.collingwoodresearch.com/uploads/8/3/6/0/8360930/replication_data.rdata") )

# Examine Available Datasets #
objects()
```

```
##  [1] "c_dfm"                      "click_corp"
##  [3] "clicks"                     "demos"
##  [5] "eo_add"                     "eo_add2"
##  [7] "imp_dat_w3_a"               "imp_dat_w3_b"
##  [9] "kass_sentiment_plot"        "kass_sentiment_plot_joy"
## [11] "kass_sentiment_plot_pos" "np"
## [13] "nrc"                        "out"
## [15] "seg_date_count"             "seg_date_count2"
## [17] "sent_data"                  "tidy_kass_tweets"
## [19] "tok_look"                   "toks"
## [21] "w3"
```

```
##############################################
#                  FIGURE 9                  #
##############################################

#install.packages("stringr")
library(stringr)
```

```r
# Adjust some of the \x type non-alpha characters #
np$text <- iconv(np$text,"WINDOWS-1252","UTF-8")

# Convert Text to Lower #
np$text <- tolower(np$text)

# Calculate article counts or the text patterns #
np$dem_count <- str_count(np$text,"democrat|democrats")
np$gop_count <- str_count(np$text,"republican|republicans")
np$trump_count <- str_count(np$text,"trump")
np$protests_count <- str_count(np$text,"protest|protesters|protests|airport|airports")
np$lind_count <- str_count(np$text,"graham|mccain")
np$pelosi_count <- str_count(np$text,"schumer|pelosi")
np$ai_count <- str_count(np$text,"american|unamerican|un-american|core values|religious freedom|religio

np <- np[!is.na(np$text),] # drop missing

#################################
# Function to summarize "theme" #
#################################

party_mention <- function(x){

  dem <- sum(x$dem_count)
  rep <- sum(x$gop_count)
  tru <- sum(x$trump_count)
  prot <- sum(x$protests_count)
  lg <- sum(x$lind_count)
  pel <- sum(x$pelosi_count)
  ai <- sum(x$ai_count)

  return ( c(dem, rep, tru, prot, lg, pel, ai) )
}

# Subset data to Week for first 3 months #
npweek <- np[np$month < 4,]

# Split Data by Week #
npweek_s <- split(npweek, npweek$week) ; length(npweek_s)
```

```
## [1] 13
```

```r
###################################
# Create Weekly Data Distribution #
###################################

dist_filt <- plyr::ldply(npweek_s, party_mention) # party_mention function
colnames(dist_filt) <- c("week", "dem", "rep","trump", "protest", "gm", "ps", "ai")

# Initiate Plot #
plot(dist_filt$week, dist_filt$trump, typ='l', bty='n', lwd=2, col="black", lty=9,
     ylim = c(0,1800),
     main = "Themes in Newspaper Articles Over Time\n(First 3 Months of Year)",
     xlab = "Week 1                                  Numeric Week                                  Week 1
```

```r
     ylab = "Counts of themes appearing in Ban articles")
lines(dist_filt$week, dist_filt$dem, typ='l', col="blue", lwd=2, lty=2)
lines(dist_filt$week, dist_filt$rep, typ='l', col="red", lwd=2, lty=3)
lines(dist_filt$week, dist_filt$protest, typ='l', col="green", lwd=2, lty=4)
lines(dist_filt$week, dist_filt$gm, typ='l', col="turquoise", lwd=2, lty=5)
lines(dist_filt$week, dist_filt$ps, typ='l', col="brown", lwd=2, lty=6)
lines(dist_filt$week, dist_filt$ai, typ='l', col="purple", lwd=2, lty=8)
abline(v=4.5, col="grey", lwd=2, lty=1)
text(2.8, 1600, "EO Ban\nAnnounced")

legend("topright",
       cex=.9,
       col = c("black","purple","green","blue", "red", "turquoise", "brown"),
       lwd=rep(2,7),
       lty=c(9,4,8,2,3,5,6),
       bty='n',
       title = "Theme",
       legend = c("Trump","American identity", "Protest/Airport","Democrat",
                  "Republican", "Graham/McCain","Pelosi/Schumer"))
```