# TAD Week 6 Assignment

Tommy Klein

3/8/2022

**Working Directory**

```
setwd('/Users/tklein/Desktop/Desktop_tpk/JHU_Classes/text_as_data/week5')
```

**Library**

```
library(ndjson)
library(SentimentAnalysis)
```

```
##
## Attaching package: 'SentimentAnalysis'
```

```
## The following object is masked from 'package:base':
##
##     write
```

```
library(RedditExtractoR)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.6
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()  masks stats::filter()
## x purrr::flatten() masks ndjson::flatten()
## x dplyr::lag()     masks stats::lag()
```

```
source('../functions/helper_functions.R')
```

```
## Package version: 3.2.0
## Unicode version: 13.0
## ICU version: 69.1
```

```
## Parallel computing: 4 of 4 threads used.
```

```
## See https://quanteda.io for tutorials and examples.
```

## Question 1

Dictionary methods assign values to words or phrases by using pre-determined labels. For example, a dictionary that attempts to classify text as positive or negative may assign a value of 1 to the word "awesome" because it is associated with positive sentiment, and a value of 0 to the word "terrible" because it is associated with negative sentiment. In this case a higher score would mean that the text is more positive, and a lower score would mean that the text is more negative. The most obvious downside of this method is that it lacks any sort of context - the words and phrases have the same values in every text, regardless of how they were used in each text. Using the above example, the sentences "It was terrible." and "I feel terrible for you if you missed it." would both receive a negative rating for the word "terrible", even though the two phrases have opposite meanings.

This flaw can sometimes be overcome by curating a dictionary that is based on text with similar context. For example, public company financial filings are filled with text disclosures that could be analyzed via dictionaries. However, the language used in these disclosures is often vary different than colloquial language - there is a much different context. For example, a "liability" colloquially is not a good thing, but with regards to financial disclosures it is just a common part of the balance sheet. Using a dictionary method on financial disclosure text may produce unexpected results, unless the dictionary was crafted with this context in mind. However, no matter how good the dictionary, it is impossible to capture the full context of every text - even among financial disclosures some words/phrases may be positive in some text and negative in another.

While this risk cannot be eliminated, it can be controlled by validating the dictionary. To validate a dictionary you would first need to score some documents manually. Then, you could use the dictionary to score the same documents automatically. Comparing the two scores would then shed light on how well the dictionary performs at scoring documents. This process can also be iteratively completed through training and testing sets. Using the training set of documents that have been manually scored, a dictionary can be created the assigns values to words/phrases derived from the scores of the documents. Then, these values can be applied to another set of documents that have also been manually reviewed. This process can be repeated to create a dictionary that minimizes the difference between the human scores and the dictionary scores of the testing documents. The benefit of this is that you would be able to quantify how different, on average, the dictionary method is from the human grades, which you could use to guide classifications from the dictionary. For example, something graded as positive by the dictionary, but within the average dictionary error, might instead be rated as "neutral".

### Question 2

First, I will pull in some data from Reddit.

```
# I ran these initially, but they take quite a bit so I saved the data frames
#top_crypto_urls <- find_thread_urls(subreddit="crypto", sort_by="top", period = 'year')
#
#
#crypto_contents <- get_thread_content(top_crypto_urls$url)
#
#write.csv(crypto_contents$threads, file = 'reddit_crypto_threads.csv')
#
#write.csv(crypto_contents$comments, file = 'reddit_crypto_comments.csv')

crypto_comments_df <- read_csv('reddit_crypto_comments.csv')
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
```

```
##   X1 = col_double(),
##   url = col_character(),
##   author = col_character(),
##   date = col_date(format = ""),
##   score = col_double(),
##   upvotes = col_double(),
##   downvotes = col_double(),
##   golds = col_double(),
##   comment = col_character(),
##   comment_id = col_character()
## )
```

```
crypto_comments <- csv_to_corpus('reddit_crypto_comments.csv', 'comment')

crypto_comments[1:2]
```

```
## Corpus consisting of 2 documents and 9 docvars.
## 1 :
## "There's a lot of context missing here"
##
## 2 :
## "I simply asked:  what are the best attacks against AES?"
```

```
crypto_comments_df$comment[1:2]
```

```
## [1] "There's a lot of context missing here"
## [2] "I simply asked:  what are the best attacks against AES?"
```

Then I will use the Sentiment Analysis package to add in the sentiment.

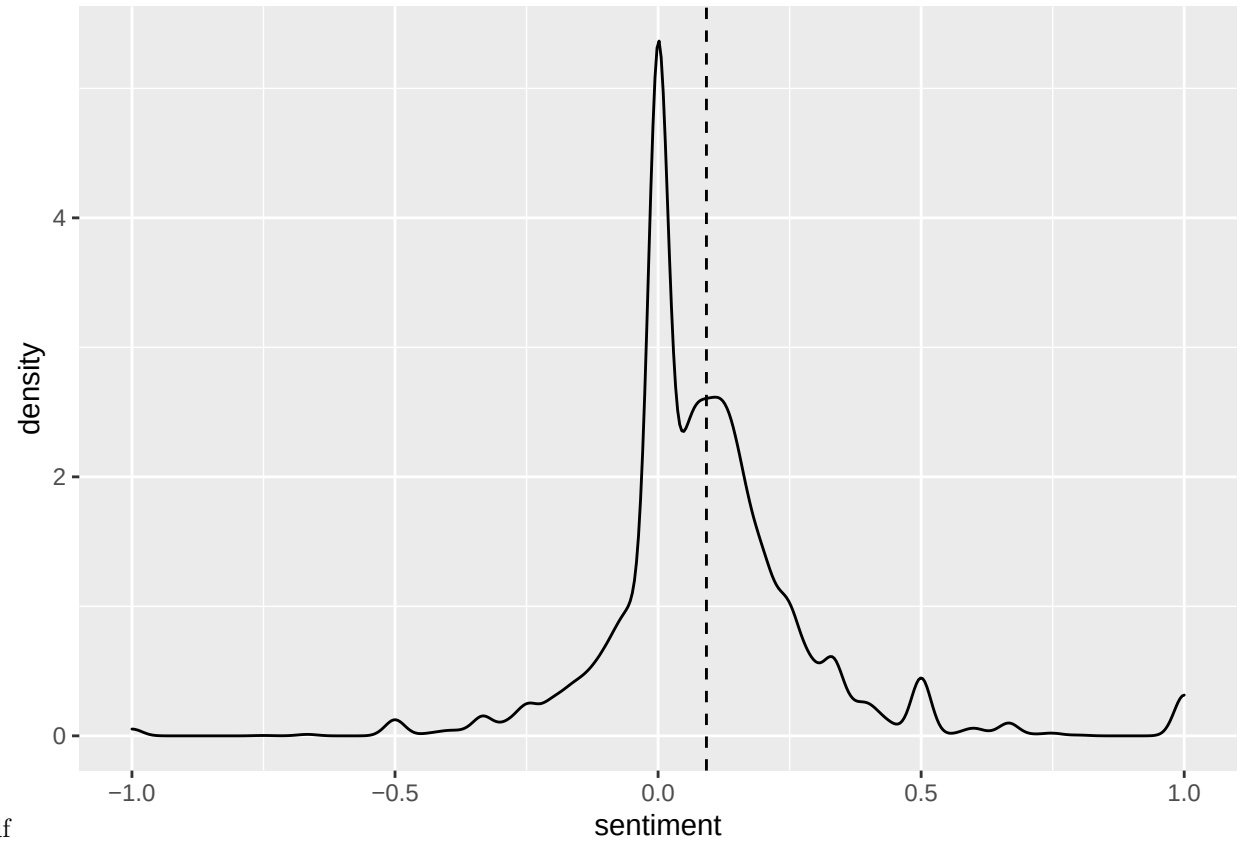
```
sentiment <- analyzeSentiment(crypto_comments)

crypto_comments_df$sentiment <- sentiment$SentimentGI
```

Now I can visualize the sentiment

```
crypto_comments_df %>%
  ggplot(aes(sentiment))+
  geom_density()+
  geom_vline(xintercept = mean(crypto_comments_df$sentiment, na.rm = T), linetype = 2)
```

```
## Warning: Removed 31 rows containing non-finite values (stat_density).
```



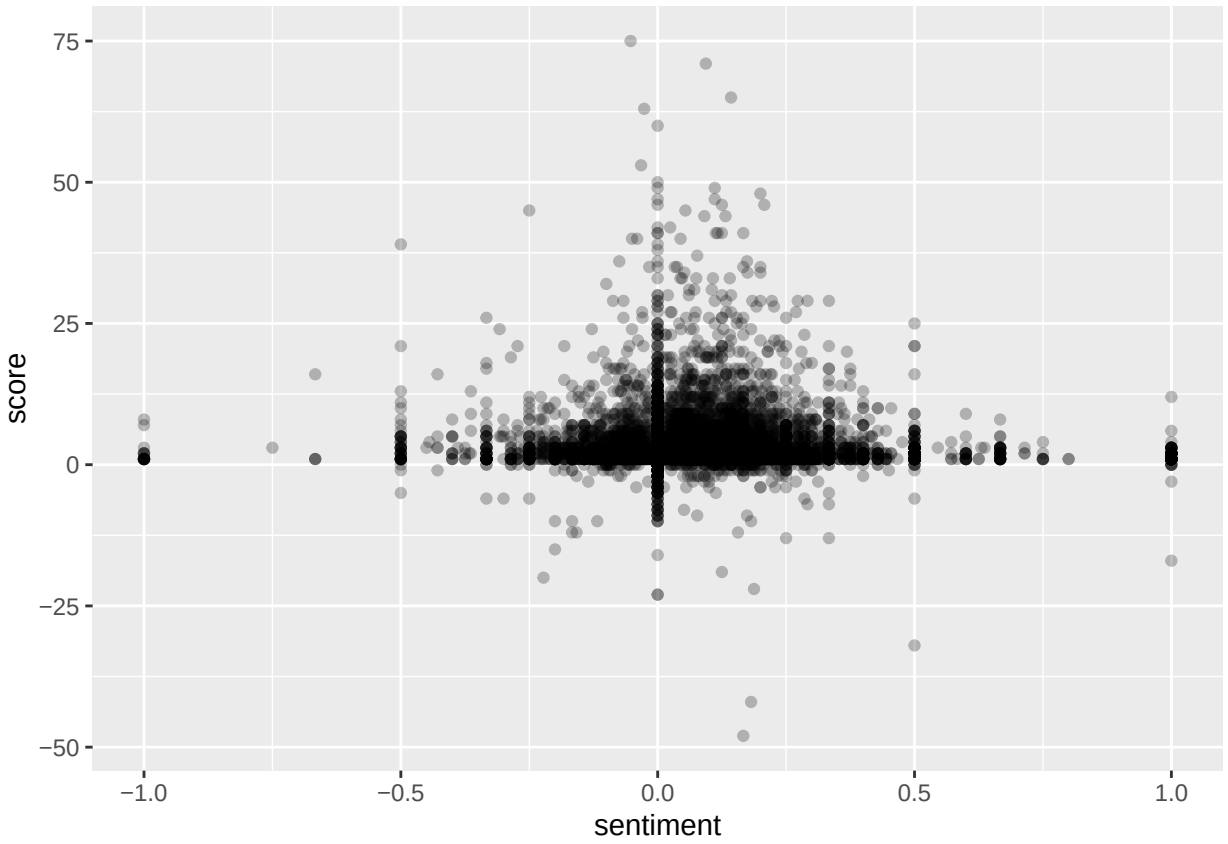

distribution-1.pdf

It looks like the sentiment of the comments definitely skews towards being more positive, but the mean score is very close to 0.

I can also visualize how sentiment is related to a comments votes.

```
crypto_comments_df %>%
  ggplot(aes(sentiment, score))+
  geom_point(alpha = .25)
```

```
## Warning: Removed 31 rows containing missing values (geom_point).
```




There really doesn't seem to be a strong relationship here, at least not a linear one. That is a bit unexpected. I would think given the enthusiasm for crypto the last few years that more positive comments would receive more upvotes, and negative comments would receive more downvotes. But instead there doesn't appear to be any correlation. However, the language people use on reddit, particularly when referring to crypto may not match the context in the dictionaries used to assign sentiment. For example, "Hold on for dear life" (often shortened to HODL), is a popular saying in the "crypto-community", and would be taken as a positive statement. But in other communities it would be used to convey turbulent times ahead, and would likely not be seen as a positive statement. So the sentiment assigned by the dictionary may not be valid for this particular use case.