Workshop: Data science with R (ZEW)

Session #5: Sentiment analysis and graphics

Obryan Poyser
2019-03-27

Outline

- Sentiment analysis
  - Twitter API
  - Text files
  - Regular expressions
- Graphs with `ggplot2`

Sentiment analysis

1. Information gathering involves finding what other people think
2. Growing availability of opinion rich content in the Internet opened new possibilities
3. Sentiment analysis, (also called opinion mining)
    1. Field of study that analyzes: people's opinions, sentiments, appraisals, attitudes, and emotions toward entities and their attributes expressed in written text.
    2. Entities could be: products, services, organizations, individuals, events, issues, or topics
    3. Computational treatment of the opinion, sentiment, and subjectivity

Sentiment analysis

1. It has been an active research field of natural language processing (NLP).
    1. Data mining
    2. Web mining
2. In acamedia is strongly related to **reputation mechanisms**
    1. Online reputation mechanisms builds **trust** inside marketplaces.
    2. Reputation mechanism promote cooperation and reduce asymmetric information.
    3. Lemons? Anybody?

Source: https://medium.com/@avtarsehra/icos-and-economics-of-lemon-markets-96638e86b3b2

Sentiment analysis

1. Creating systems that can process subjective information effectively requires overcoming a number of novel challenges.

2. Complex structure of the languange

    1. Contradictory: Some phrases might express contradictory polarity conditional on the context:
        1. *This camera sucks* vs *this vaccum cleaner really sucks*
    2. Sentences can express no sentiment at all
        1. *Can you tell me which Sony camera is good?*
        2. *Does anyone know how to repair this terrible printer?* (ever more complex)
    3. Sarcasm: there is a great amoun of people who do not get it, imagine a machine!
        1. Not common in product reviews, but there's plenty of them in politics.
    4. Implicit sentiment
        1. *This washer uses a lot of water* (bad)

Sentiment analysis

1. Sentiment words can be divided into:
    1. base type: as expressed before
    2. comparative type: better, worse, best, worst, etc.

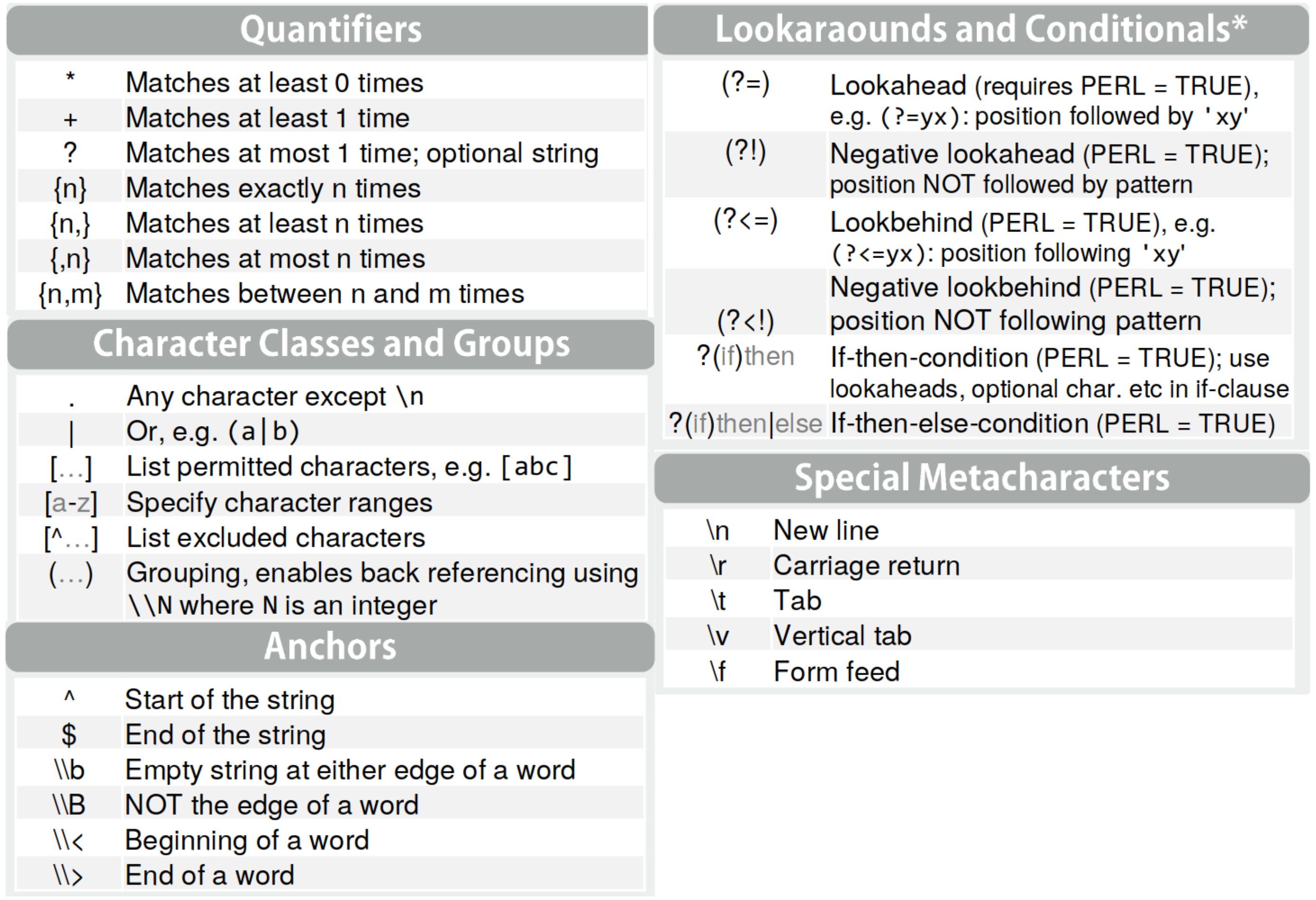
1. There three main approaches compile sentiment words:
    1. Manual: time consuming
    2. Dictionary-based: Process of the algorithm: seeds' sentiment words, search for synonyms/antonyms, propagate polarity and ends with manual cleaning. There are more complex algorithms as well.
    3. Corpus-based: given a seed of list of known sentiment words, discover other sentiment words and their orientations from a domain corpus. Adapt the new sentiment lexicon using the domain corpus.

1. Sentiment analysis is a whole are of NLP, therefore there is vas amount of details and approaches, in this session we will focus on dictionary lexicon-based method for sentiment analysis.

Sentiment analysis [in practice]

1. My must include in our technical toolbox **regular expressions (REGEX)**
2. Then again, is a whole universe of rules.
    1. There will be always one easier way to solve one problem. Practice matters a lot.
3. REGEX reference: https://www.regular-expressions.info/refcapture.html
4. REGEX gymnasium: https://regex101.com/
5. REGEX in R cheatsheet: https://www.rstudio.com/wp-content/uploads/2016/09/RegExCheatsheet.pdf
6. Main difference: R uses two escaping backslashes `\\`
7. Is an iterative process. The cleaning process demands a lot of time.

Sentiment analysis [in practice]



Quantifiers	
*	Matches at least 0 times
+	Matches at least 1 time
?	Matches at most 1 time; optional string
{n}	Matches exactly n times
{n,}	Matches at least n times
{,n}	Matches at most n times
{n,m}	Matches between n and m times

Character Classes and Groups	
.	Any character except \n
|	Or, e.g. (a|b)
[…]	List permitted characters, e.g. [abc]
[a-z]	Specify character ranges
[^…]	List excluded characters
(…)	Grouping, enables back referencing using \\N where N is an integer

Anchors	
^	Start of the string
$	End of the string
\\b	Empty string at either edge of a word
\\B	NOT the edge of a word
\\<	Beginning of a word
\\>	End of a word

Lookaraounds and Conditionals*	
(?=)	Lookahead (requires PERL = TRUE), e.g. (?=yx): position followed by 'xy'
(?!)	Negative lookahead (PERL = TRUE); position NOT followed by pattern
(?<=)	Lookbehind (PERL = TRUE), e.g. (?<=yx): position following 'xy'
(?<!)	Negative lookbehind (PERL = TRUE); position NOT following pattern
?(if)then	If-then-condition (PERL = TRUE); use lookaheads, optional char. etc in if-clause
?(if)then|else	If-then-else-condition (PERL = TRUE)

Special Metacharacters	
\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
\f	Form feed

Sentiment analysis [in practice]

Packages

```r
library("twitteR")
library("tidytext")
library(qdap)
library(tm)
library(epubr)
```

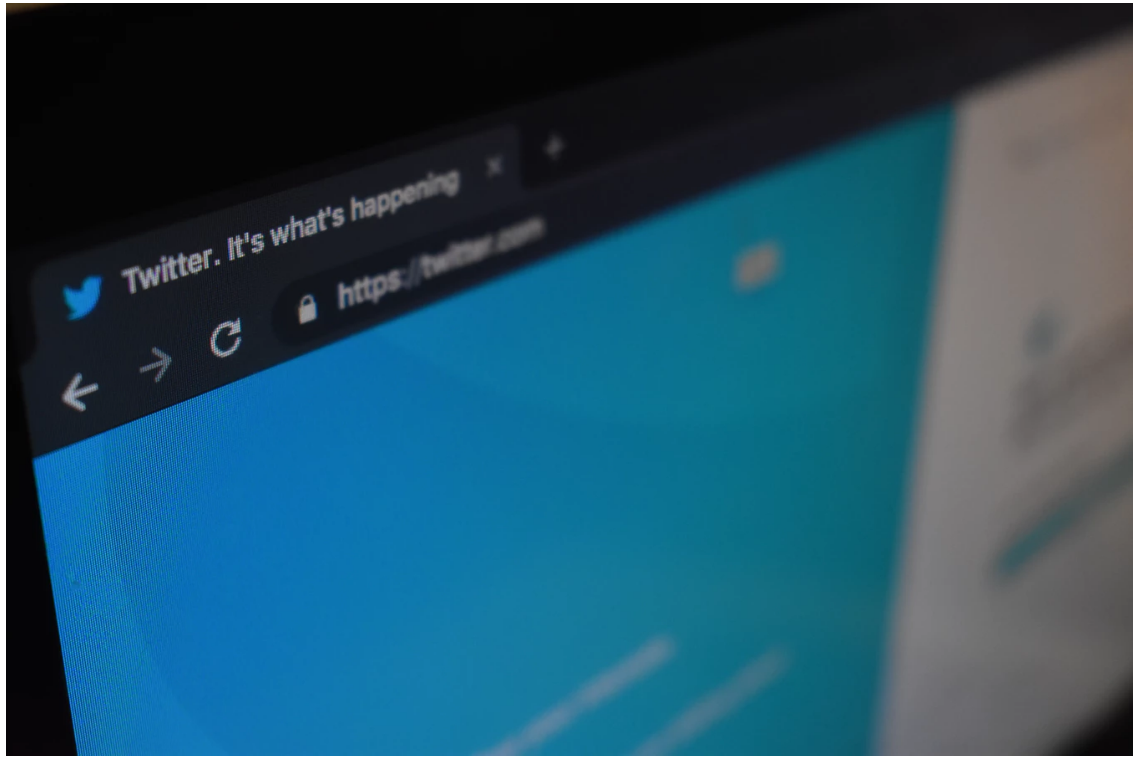
Relevant functions

```r
tm::removePunctuation()
tm::removeNumbers()
tm::stripWhitespace()
tm::removeWords()
tm::stopwords()
tm::stemDocument() # reduce words to unify documents eg
# computers, computation -> comput -> computer
tm::TermDocumentMatrix()
tm::DocumentTermMatrix()
tm::stemCompletion() # complete the word taking as an input a
qdap::bracketX(): # "It's (so) cool" -> "It's cool"
qdap::replace_number(): # "2" -> "two"
qdap::replace_abbreviation(): # "Sr" -> "Senior"
qdap::replace_contraction(): # "shouldn't" -> "should not"
qdap::replace_symbol() # "$" -> "dollar"
```

Sentiment analysis [Twitter API]

As humans, what are some things that we want that technology might help us to get?

1. We want to be heard.
2. We want to satisfy our curiosity.
3. We want it easy.
4. We want it now.



How to create a Twitter API: https://towardsdatascience.com/access-data-from-twitter-api-using-r-and-or-python-b8ac342d3efe

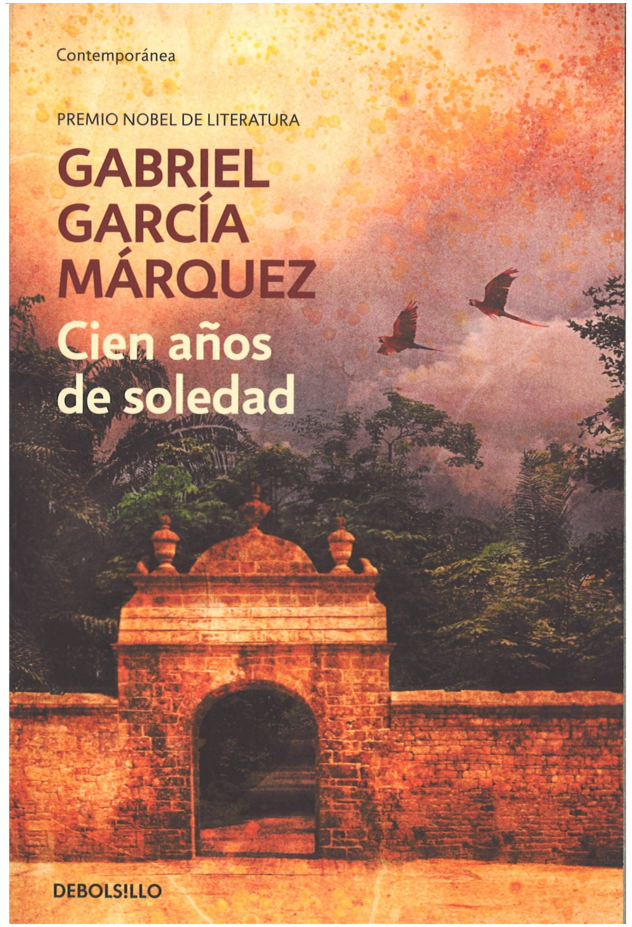
Sentiment analysis [Twitter API and Brexit]

```r
tw = searchTwitter(searchString = "#Brexit OR Brexit", n = 500, lang = "en") %>%
    twListToDF() %>% # Converts raw tweets to dataframe
    as_tibble()
#write_rds(tw, path = "datasets/twitter.rds")
```

```r
tw <- read_rds(path = "datasets/twitter.rds")
tw$text %>%
  head(10)
```

```
##  [1] "This ‘second vote’ debate reminds me of my 3 year old niece when she doesn’t get her own way. #brexit"
##  [2] "RT @euronews: #RawPolitics | \"Bored to death.\"\n\nAre you sick of Brexit? Well, German MEP Jens Geier echoed how some of
##  [3] "A kid no older than four has walked in to Addenbrookes with a massive \"Z\" shaved in to his head and now I absolutel… htt
##  [4] "RT @rtenews: EU cannot betray 6m people who signed petition to revoke Article 50 : @eucopresident #brexit https://t.co/wlS
##  [5] "@Trump_ton Many would leave with Brexit looming."
##  [6] "RT @AyoCaesar: Today in ‘Too Remain-y for Lexiters, Too Lexit-y for Remainers’: denying the relationship of Brexit to an e
##  [7] "RT @LeaveEUOfficial: WATCH | Unbelievable! The President of the European Council Donald Tusk says \"you cannot ignore the
##  [8] "RT @mikegalsworthy: We need just 180K to get this over 💥6 MILLION💥 today.\n\nThe Government’s dismissive response is irr
##  [9] "I've just asked my MP to support the Beckett Amendment for a People's Vote on the Brexit deal - tells yours here:… https:/
## [10] "RT @remain_central: Retweet if you want the Brexit default to be: Revoke Article 50. https://t.co/caCHzrdWnu"
```

Sentiment analysis [Case: One hundred years of solitude]



Cien años de soledad - Gabriel García Marquez (eng. version)

Sentiment analysis

Case: One hundred years of solitude

```
#cas <- epub("datasets/garcia_one_hundred.epub")
#write_rds(cas, path = "datasets/cas.rds")
(cas <- read_rds("datasets/cas.rds") %>%
  dplyr::select(title:date, data))
```

```
## # A tibble: 1 x 4
##   title                            creator              date  data
##   <chr>                            <chr>                <chr> <list>
## 1 1967 - One Hundred Years of So… Gabriel Garcia Marq… 1967  <tibble [22 ×…
```

```
cas$data[[1]] %>%
  head(5)
```

```
## # A tibble: 5 x 4
##   section    text                                                   nword nchar
##   <chr>      <chr>                                                  <int> <int>
## 1 Cover      ""                                                         0     0
## 2 Heading000 "Gabriel García Márquez\nOne Hundred Years of Sol…   103   633
## 3 Heading001 "ONE\n\nMANY YEARS LATER as he faced the firing s…  6102 34781
## 4 Heading002 "TWO\n\nWHEN THE PIRATE Sir Francis Drake attacke…  6521 36070
## 5 Heading003 "THREE\n\nPILAR TERNERA'S son was brought to his …  7899 44139
```

```
# Work from the text
cas <- cas$data[[1]]
```

Sentiment analysis: lexicons and polarity

```r
# afinn
get_sentiments(lexicon = "afinn") %>%
  summary()
```

```
##      word               score        
##  Length:2476        Min.   :-5.0000  
##  Class :character   1st Qu.:-2.0000  
##  Mode  :character   Median :-2.0000  
##                     Mean   :-0.5889  
##                     3rd Qu.: 2.0000  
##                     Max.   : 5.0000
```

```r
# bing
get_sentiments(lexicon = "bing") %>%
  distinct(sentiment)
```

```
## # A tibble: 2 x 1
##   sentiment
##   <chr>    
## 1 negative 
## 2 positive
```

1. AFINN is a list of English words rated for valence with an integer between minus five (negative) and plus five (positive). The words have been manually labeled by Finn Årup Nielsen in 2009-2011. The file is tab-separated. Source
2. Bing proposed the Feature-Based Opinion Mining model, which is now also called Aspect-Based Opinion Mining Source

Sentiment analysis: lexicons and polarity

```r
# nrc
get_sentiments("nrc") %>%
  distinct(sentiment) %>%
  pull(sentiment)
```

```
##  [1] "trust"        "fear"         "negative"     "sadness"
##  [5] "anger"        "surprise"     "positive"     "disgust"
##  [9] "joy"          "anticipation"
```

```r
# loughran
get_sentiments("loughran") %>%
  distinct(sentiment) %>%
  pull(sentiment)
```

```
## [1] "negative"     "positive"     "uncertainty"  "litigious"
## [5] "constraining" "superfluous"
```

1. The NRC Emotion Lexicon is a list of English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive). The annotations were manually done by crowdsourcing. Source
2. Loughran contains tools useful for textual analysis in financial applications and data from some of the textual-related publications. Source

Sentiment analysis: lexicons and polarity

1. Polarity is calculated with according to the following variables
    1. Polarized term: negative or positive
    2. Neutral term: no emotion
    3. Negator: inverted polarized eg "not good"
    4. Valence shifters: words that affect the emotional context
        1. Amplifiers: increase emotional context (eg very good)
        2. De-amplifiers: decrease emotional context (eg not bad)

```r
string <- c("The view from east side of the bridge is very good")

(polarity <- qdap::polarity(string))
```

```
##   all total.sentences total.words ave.polarity sd.polarity stan.mean.polarity
## 1 all               1          11        0.543          NA                 NA
```

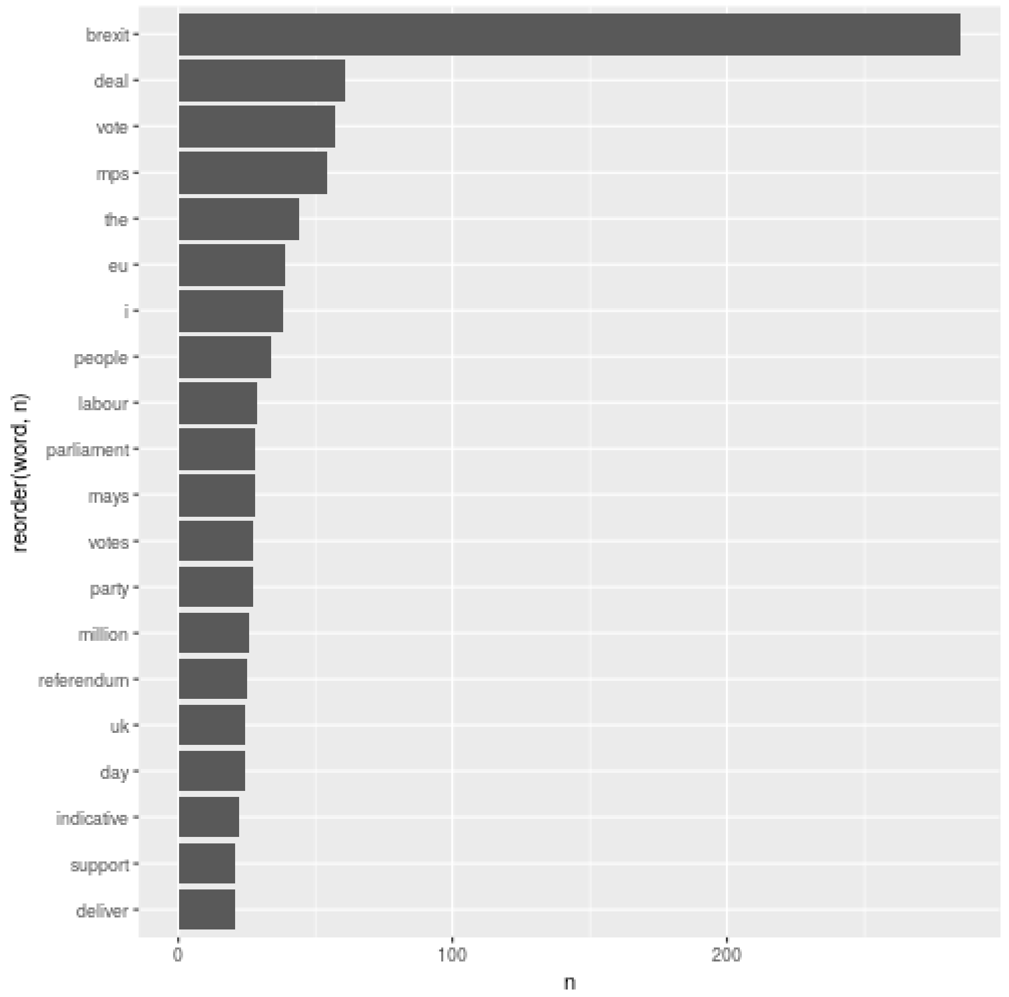
$$\frac{term + amplifier}{\sqrt{n}} \quad \frac{1 + 0.8}{\sqrt{11}} = 0.54$$

Sentiment analysis [Twitter API and Brexit]

```r
## First steps
tw1 <- tw %>%
    dplyr::select(text, id) %>%
    mutate(text=str_squish(text)
             , link=str_extract_all(text, pattern = "(?<=(https:\\/\\/)).+", simplify = T) %>%
               as.vector()
             , mention=str_extract_all(text, pattern = "(?<=(@)).+(?=: )", simplify = T) %>%
               as.vector()
             , rt=ifelse(str_detect(string = text, pattern = "RT(?= @)"), yes = T, F)
             , hashtags=str_extract_all(text, "#\\S+") %>%
               map(.f = ~paste0(., collapse = ";")) %>%
               unlist()
             , text_clean=str_remove_all(text, pattern = "…|(#\\S+ )|(@\\S+ )|(htt\\S+)|(RT )") %>%
               str_replace_all(pattern = "’", replacement = "'") %>%
               str_remove_all(pattern = "![:alpha:]") %>%
               str_to_lower() %>%
               qdap::replace_abbreviation() %>%
               qdap::replace_contraction() %>%
               tm::removePunctuation() %>%
               tm::removeWords(words = c(stop_words$word, stopwords())) %>%
               str_squish() %>% str_to_lower()
               ) %>%
    mutate(word=map(text_clean, ~str_extract_all(string = .x
                                                    , pattern = "[[:alpha:]]+", simplify = T) %>%
                        as.vector() %>%
                        str_split(pattern = " ", simplify = T)
                    )
             , text_clean=str_to_lower(string = text_clean)) %>%
    unnest(word)
```

Sentiment analysis [Twitter API and Brexit]

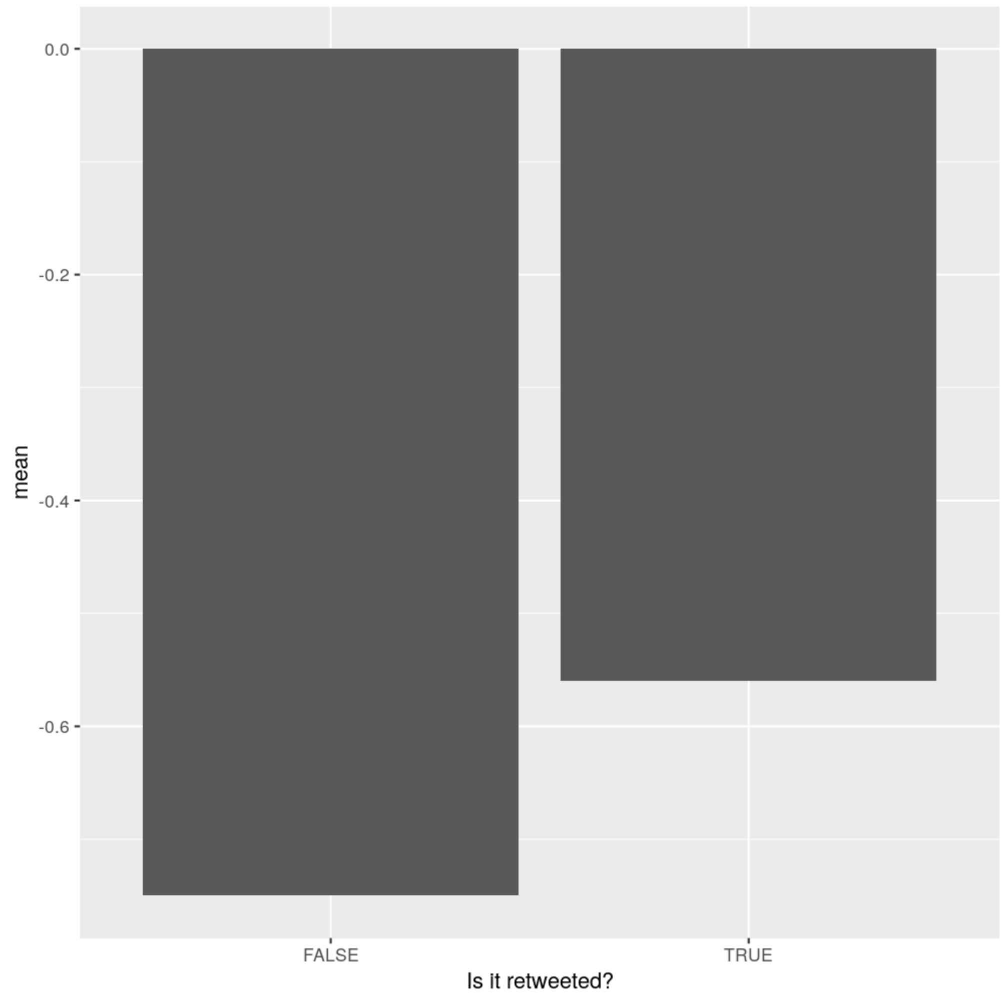
```
tw2 <- list(tw1, get_sentiments("afinn")
     , get_sentiments("bing")
     , get_sentiments("nrc")
     , get_sentiments("loughran")) %>%
  reduce(left_join, by="word") %>%
  group_by(id) %>%
  filter(!duplicated(word)) %>%
  rename(afinn=score, bing=sentiment.x, nrc=sentiment.y, lough

tw2 %>%
  group_by(word) %>%
  tally() %>%
  arrange(desc(n)) %>%
  top_n(n = 20, wt = n) %>%
  ggplot(aes(reorder(word, n), n))+
  geom_col()+
  coord_flip()
```



Sentiment analysis [Twitter API and Brexit]

AFINN

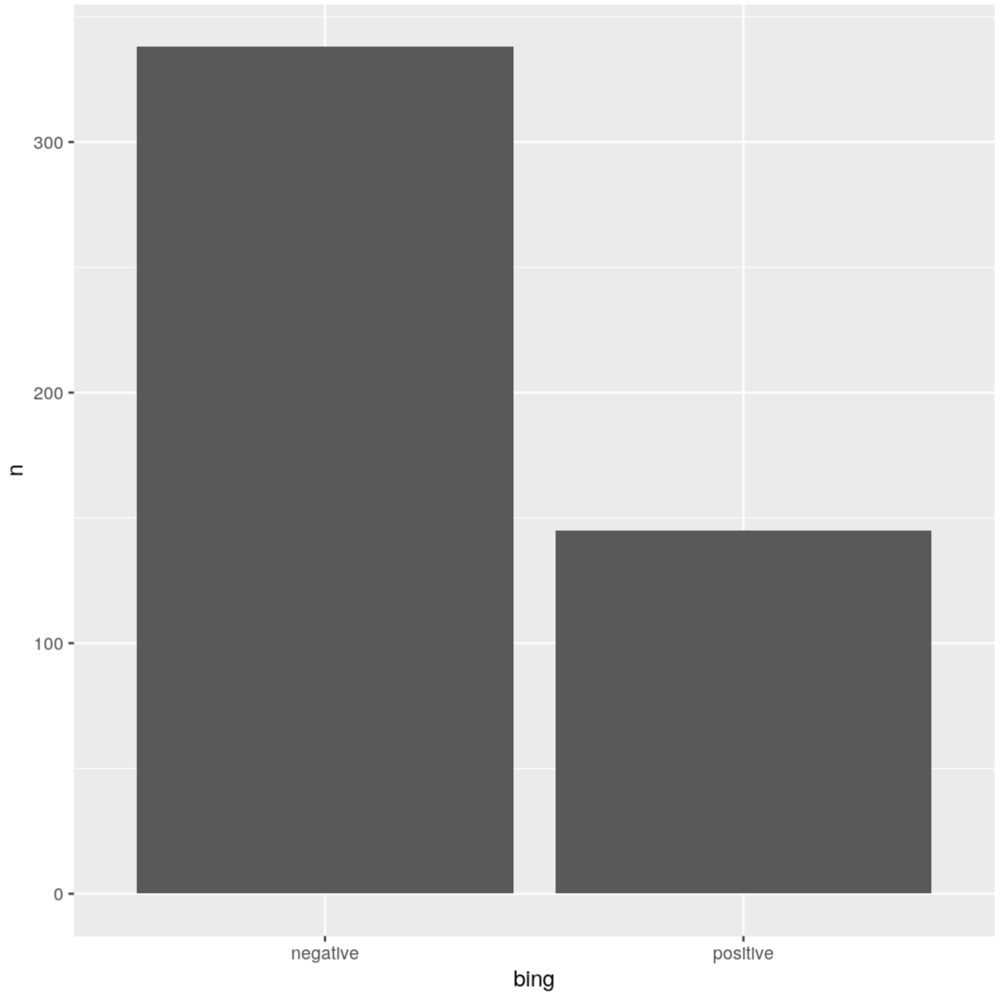
```
tw2 %>%
  filter(!is.na(afinn)) %>%
  group_by(rt) %>%
  summarise(mean=mean(afinn)) %>%
  ggplot(aes(rt, mean))+
  geom_col()+
  labs(x="Is it retweeted?")
```



Sentiment analysis [Twitter API and Brexit]

BING

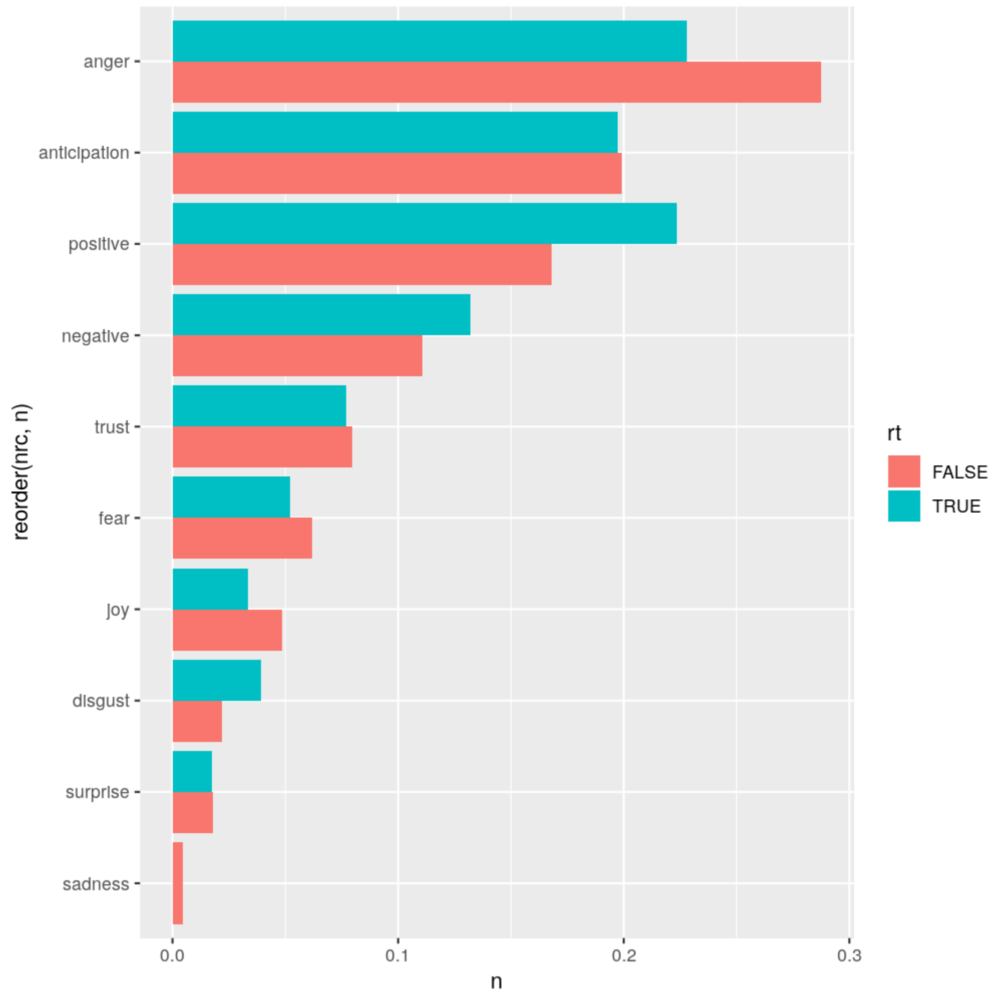
```
tw2 %>%
  filter(!is.na(bing)) %>%
  group_by(bing) %>%
  tally() %>%
  ggplot(aes(bing, n))+
  geom_col()
```



Sentiment analysis [Twitter API and Brexit]

NRC

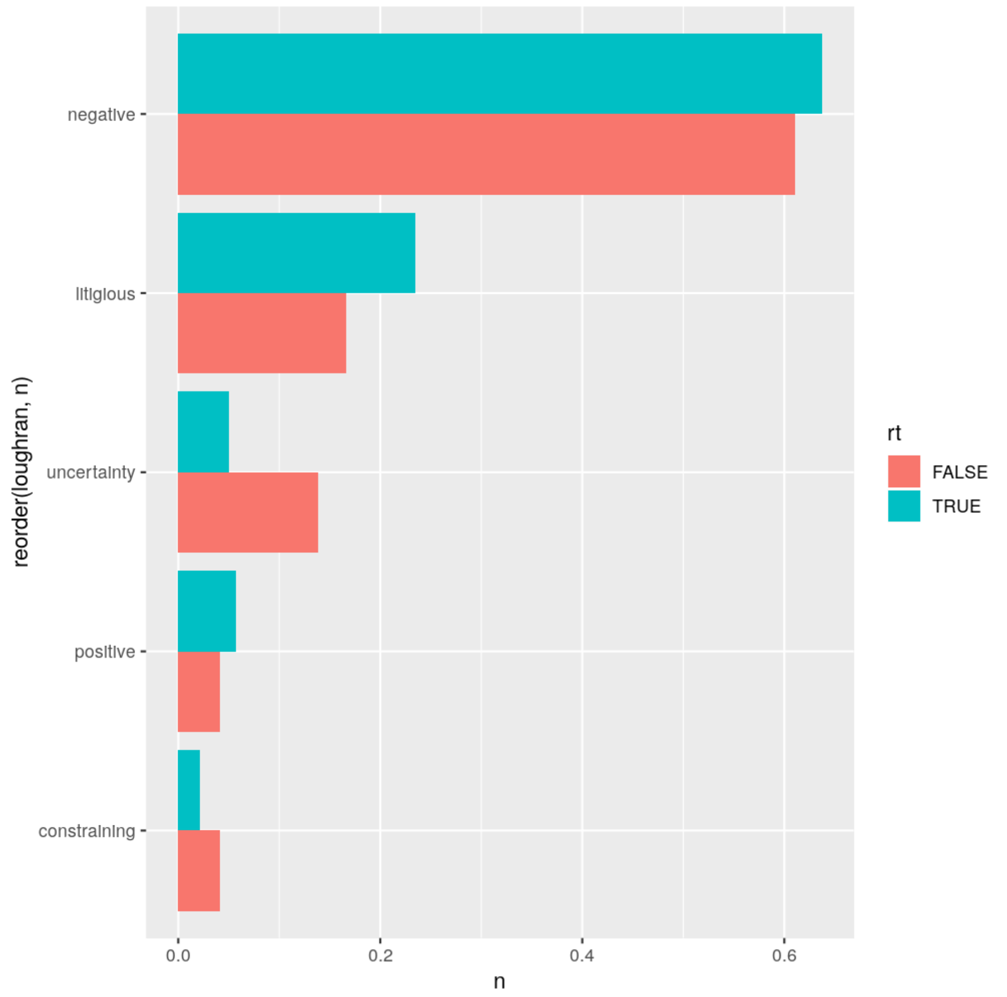
```
tw2 %>%
  filter(!is.na(nrc)) %>%
  group_by(rt, nrc) %>%
  tally() %>%
  mutate(sum=sum(n), n=n/sum) %>%
  replace_na(replace = list(sadness=0)) %>%
  ggplot(aes(reorder(nrc, n), n, fill=rt))+
  geom_col(position = "dodge")+
  coord_flip()
```



Sentiment analysis [Twitter API and Brexit]

Loughran

```
tw2 %>%
  filter(!is.na(loughran)) %>%
  group_by(rt, loughran) %>%
  tally() %>%
  mutate(sum=sum(n), n=n/sum) %>%
  replace_na(replace = list(sadness=0)) %>%
  ggplot(aes(reorder(loughran, n), n, fill=rt))+
  geom_col(position = "dodge")+
  coord_flip()
```



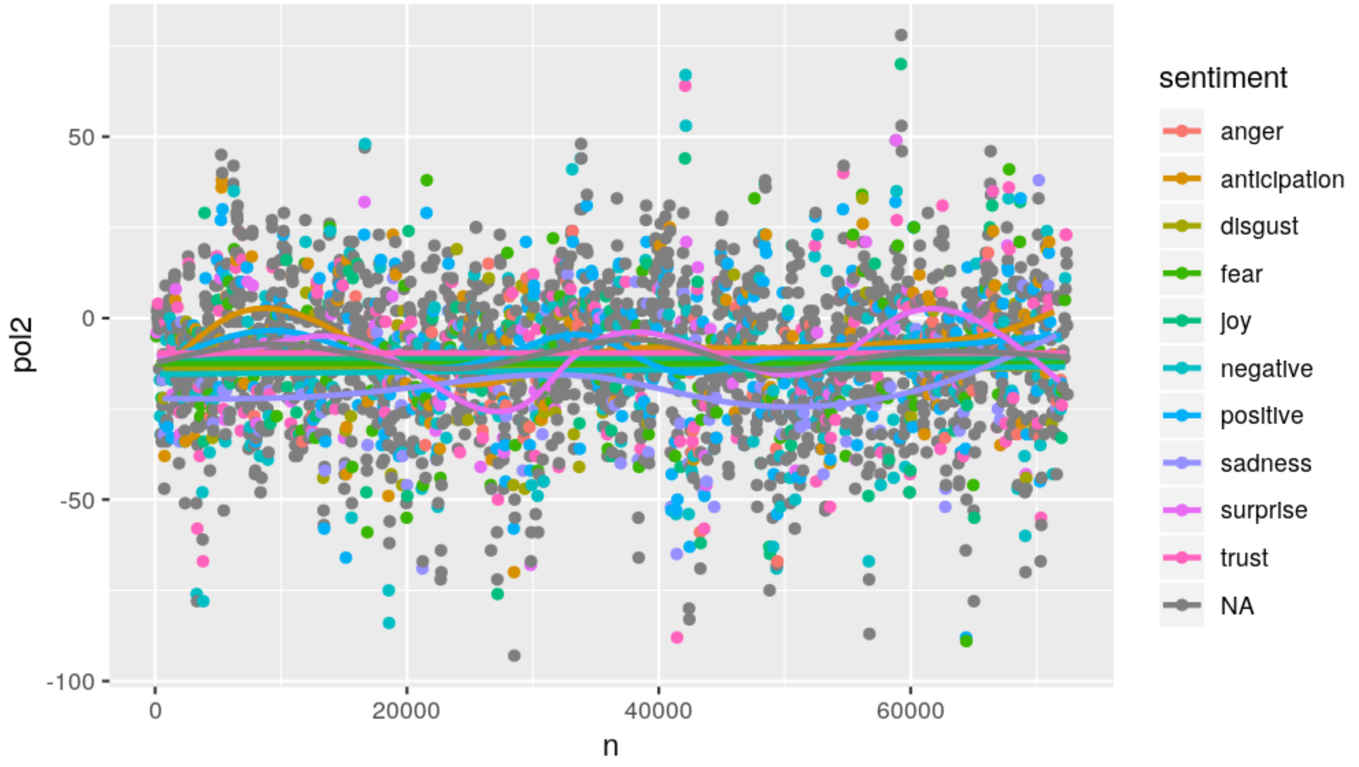
Sentiment analysis [CAS]

```r
cas1 <- cas %>%
  slice(3:n()) %>%
  mutate(text=str_to_lower(text) %>%
           tm::removePunctuation() %>%
           str_remove_all(pattern = "\\n") %>%
           tm::removeWords(words = stop_words$word) %>%
           str_squish()) %>%
    unnest_tokens(output = "word", input = "text") %>%
    left_join(get_sentiments("afinn")) %>%
    left_join(get_sentiments("nrc")) %>%
    #replace_na(replace = list(score=0)) %>%
    #group_by(section) %>%
    mutate(n=1:n(), sum=sum(score)
           , mean=mean(score, na.rm = T))
```

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

Sentiment analysis [CAS]

```
cas1 %>%
    as_tsibble(index = n) %>%
    mutate(pol2=tsibble::slide_dbl(score, ~sum(.x, na.rm = T), .size = 100, .step = 25)) %>%
    ggplot(aes(n, pol2, col=sentiment))+
    geom_point(aes(col=sentiment))+
    geom_smooth(se = F)
```



References

- Danneman, N., & Heimann, R. (2014). Social media mining with R. Packt Publishing Ltd.
- Munzert, S., Rubba, C., Meißner, P., & Nyhuis, D. (2014). Automated data collection with R: A practical guide to web scraping and text mining.
- John Wiley & Sons. Russell, M. A. (2013). Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More. " O'Reilly Media, Inc.".