

Introduction to Data Wrangling IV

Summer Institute in Data Science

Rolando J. Acosta



HARVARD
SCHOOL OF PUBLIC HEALTH

July 01, 2020



@RJANunez

What to expected today

- Today we will use some of the functions we have learned this week in a case study
 - Join functions
 - *regex*
- Specifically, we are going to conduct a *sentiment analysis* on Trump tweets
- Sentiment analysis is the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information

Case Study: Trump Tweets

- During the 2016 US presidential election, Donald Trump used his twitter account to, among other things, communicate with potential voters

Case Study: Trump Tweets

- During the 2016 US presidential election, Donald Trump used his twitter account to, among other things, communicate with potential voters
- On August 2016, Todd Vaziri tweeted:



Todd Vaziri ✓
@tvaziri

Every non-hyperbolic tweet is from iPhone (his staff).

Every hyperbolic tweet is from Android (from him).



Donald J. Trump @realDonaldTrump
Good luck #TeamUSA
#OpeningCeremony #Rio2016
pic.twitter.com/mS8qsQpJPh

27,391 Likes

8,392 Retwee

Aug 5, 2016 at 8:59 PM

via **Twitter for iPhone**



Donald J. Trump @realDonaldTrump
Heading to New Hampshire - will be
talking about Hillary saying her brain
SHORT CIRCUITED, and other thing!

4,451 Likes

1,480 Retwee

Aug 6, 2016 at 11:11 AM

via **Twitter for Android**



Case Study: Trump Tweets

- David Robinson, a data scientist, conducted an analysis to check if the data supports this claim

Case Study: Trump Tweets

- David Robinson, a data scientist, conducted an analysis to check if the data supports this claim
- Today, we will go through David's analysis

Case Study: Trump Tweets

- David Robinson, a data scientist, conducted an analysis to check if the data supports this claim
- Today, we will go through David's analysis
- Here is the set up

```
library(tidyverse)
library(lubridate)
library(tidytext)
library(textdata)
library(dslabs)
library(scales)

data("trump_tweets")
```

	source	id_str	text	created_at	retweet_count	in_reply_to_user_id_str	favorite_count	is_retweet
1	Twitter Web Client	6971079756	From Donald Trump: Wishing everyone a wonderful h...	2009-12-23 12:38:18	28	NA	12	FALSE
2	Twitter Web Client	6312794445	Trump International Tower in Chicago ranked 6th tall...	2009-12-03 14:39:09	33	NA	6	FALSE
3	Twitter Web Client	6090839867	Wishing you and yours a very Happy and Bountiful Th...	2009-11-26 14:55:38	13	NA	11	FALSE
4	Twitter Web Client	5775731054	Donald Trump Partners with TV1 on New Reality Serie...	2009-11-16 16:06:10	5	NA	3	FALSE
5	Twitter Web Client	5364614040	--Work has begun, ahead of schedule, to build the gr...	2009-11-02 09:57:56	7	NA	6	FALSE

Case Study: Trump Tweets

- We are interested in tweets made from an iPhone or an Android during the presidential campaign

Case Study: Trump Tweets

- We are interested in tweets made from an iPhone or an Android during the presidential campaign
- Thus, we have to do some wrangling:

```
campaign_tweets <- trump_tweets %>%  
  extract(source, "source", "Twitter for (.*)") %>%  
  filter(source %in% c("Android", "iPhone") &  
         created_at >= ymd("2015-06-17") &  
         created_at < ymd("2016-11-08")) %>%  
  filter(!is_retweet) %>%  
  arrange(created_at) %>%  
  as_tibble()
```

Case Study: Trump Tweets

- We are interested in tweets made from an iPhone or an Android during the presidential campaign
- Thus, we have to do some wrangling:

```
campaign_tweets <- trump_tweets %>%  
  extract(source, "source", "Twitter for (.*)") %>%  
  filter(source %in% c("Android", "iPhone") &  
         created_at >= ymd("2015-06-17") &  
         created_at < ymd("2016-11-08")) %>%  
  filter(!is_retweet) %>%  
  arrange(created_at) %>%  
  as_tibble()
```

- Note that the `extract` function is similar to the separate function that permits the use of *regex*

Case Study: Trump Tweets

- We are interested in tweets made from an iPhone or an Android during the presidential campaign
- Thus, we have to do some wrangling:

```
campaign_tweets <- trump_tweets %>%  
  extract(source, "source", "Twitter for (.*)") %>%  
  filter(source %in% c("Android", "iPhone") &  
         created_at >= ymd("2015-06-17") &  
         created_at < ymd("2016-11-08")) %>%  
  filter(!is_retweet) %>%  
  arrange(created_at) %>%  
  as_tibble()
```

- Note that the `extract` function is similar to the separate function that permits the use of *regex*
- `ymd` is a date format function from the *lubridate* package

Case Study: Trump Tweets

- Now we can employ our visualization skills!

Case Study: Trump Tweets

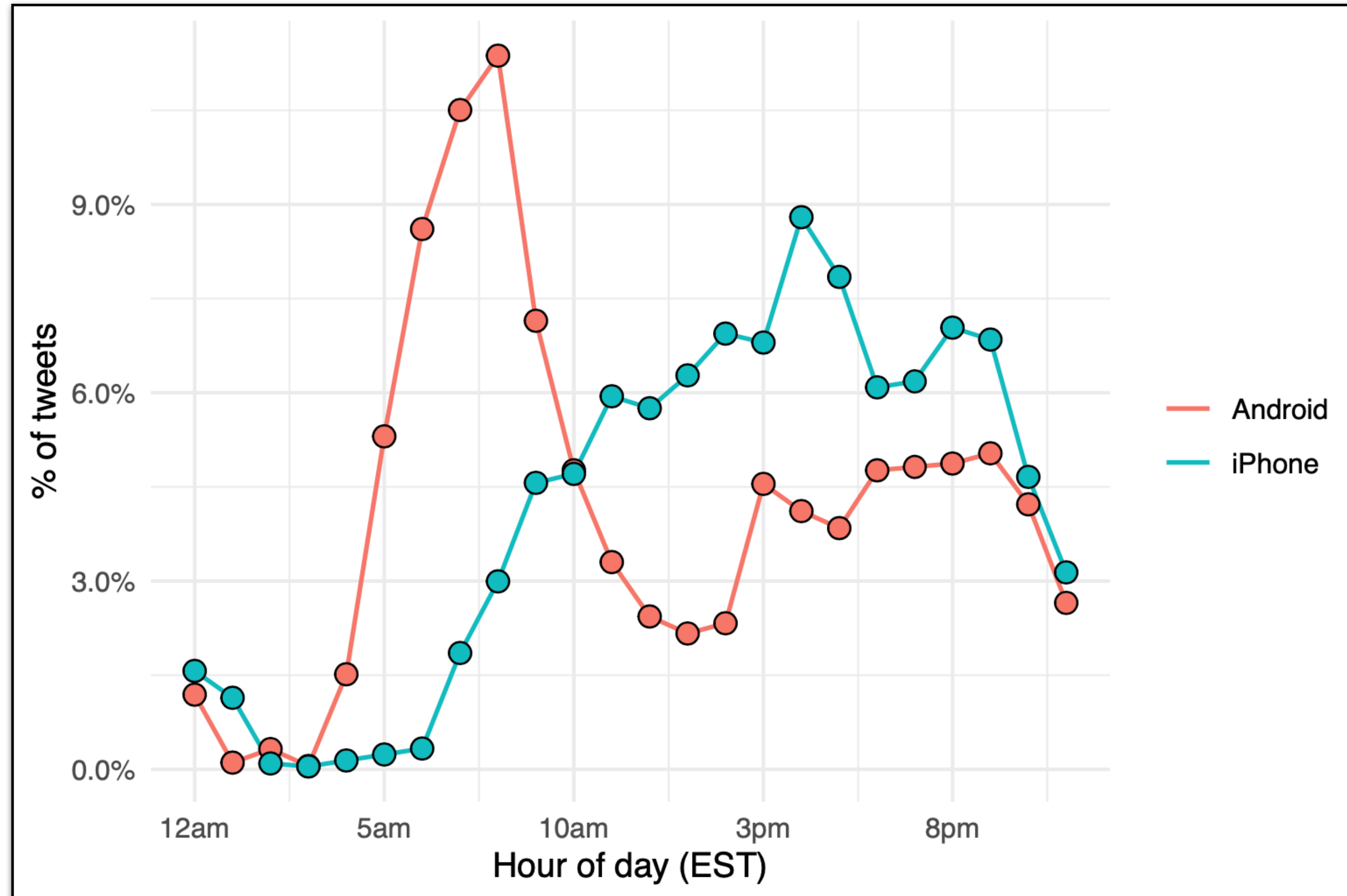
- Now we can employ our visualization skills!
- For each tweet, let us extract the hour it was tweeted and then plot proportion of tweets by device

Case Study: Trump Tweets

- Now we can employ our visualization skills!
- For each tweet, let us extract the hour it was tweeted and then plot proportion of tweets by device

```
campaign_tweets %>%  
  mutate(hour = hour(with_tz(created_at, "EST"))) %>%  
  count(source, hour) %>%  
  group_by(source) %>%  
  mutate(percent = n / sum(n)) %>%  
  ungroup() %>%  
  ggplot(aes(hour, percent, fill = source, color = source)) +  
  geom_line(size = 0.70) +  
  geom_point(shape = 21,  
             color = "black",  
             size = 3,  
             show.legend = FALSE) +  
  scale_y_continuous(labels = percent_format()) +  
  scale_x_continuous(breaks = seq(0, 20, by = 5),  
                    labels = c("12am", "5am", "10am", "3pm", "8pm")) +  
  labs(x = "Hour of day (EST)", y = "% of tweets", color = "") +  
  theme_minimal()
```

Case Study: Trump Tweets



- Comments?

Quick detour: The `unnest_tokens` function

- The *tidytext* package helps us convert text into a tidy table

Quick detour: The `unnest_tokens` function

- The *tidytext* package helps us convert text into a tidy table
- The function to do this is the `unnest_tokens` function

Quick detour: The `unnest_tokens` function

- The *tidytext* package helps us convert text into a tidy table
- The function to do this is the `unnest_tokens` function
- Here is a quick example:

```
poem <- c("Roses are red,", "Violets are blue,",  
         "Sugar is sweet,", "And so are you.")  
  
example <- tibble(line = c(1, 2, 3, 4),  
                  text = poem)  
  
example_token <- example %>% unnest_tokens(word, text)
```

Quick detour: The unnest_tokens function

	line	text
1	1	Roses are red,
2	2	Violets are blue,
3	3	Sugar is sweet,
4	4	And so are you.

example

	line	word
1	1	roses
2	1	are
3	1	red
4	2	violets
5	2	are
6	2	blue
7	3	sugar
8	3	is
9	3	sweet
10	4	and
11	4	so
12	4	are
13	4	you

example_token

Quick detour: The unnest_tokens function

- Now let us look at an example from the tweets

```
i <- 3008
campaign_tweets$text[i] %>% str_wrap(width = 65) %>% cat()
Great to be back in Iowa! #TBT with @JerryJrFalwell joining me in
Davenport- this past winter. #MAGA https://t.co/A5IF0QHnic

campaign_tweets[i,] %>%
  unnest_tokens(word, text) %>%
  pull(word)
[1] "great"      "to"         "be"         "back"       "in"
[6] "iowa"       "tbt"        "with"       "jerryjrfalwell" "joining"
[11] "me"         "in"         "davenport"  "this"       "past"
[16] "winter"     "maga"       "https"      "t.co"       "a5if0qhnic"
```

Quick detour: The unnest_tokens function

- Now let us look at an example from the tweets

```
i <- 3008
campaign_tweets$text[i] %>% str_wrap(width = 65) %>% cat()
Great to be back in Iowa! #TBT with @JerryJrFalwell joining me in
Davenport- this past winter. #MAGA https://t.co/A5IF0QHnic

campaign_tweets[i,] %>%
  unnest_tokens(word, text) %>%
  pull(word)
[1] "great"      "to"         "be"         "back"       "in"
[6] "iowa"       "tbt"        "with"       "jerryjrfalwell" "joining"
[11] "me"         "in"         "davenport"  "this"       "past"
[16] "winter"     "maga"       "https"      "t.co"       "a5if0qhnic"
```

- Note that the function strips characters that are important in the context of Twitter (e.g., @JerryJrFalwell)

Quick detour: The unnest_tokens function

- Now let us look at an example from the tweets

```
i <- 3008
campaign_tweets$text[i] %>% str_wrap(width = 65) %>% cat()
Great to be back in Iowa! #TBT with @JerryJrFalwell joining me in
Davenport- this past winter. #MAGA https://t.co/A5IF0QHnic

campaign_tweets[i,] %>%
  unnest_tokens(word, text) %>%
  pull(word)
[1] "great"      "to"         "be"         "back"       "in"
[6] "iowa"       "tbt"        "with"       "jerryjrfalwell" "joining"
[11] "me"         "in"         "davenport"  "this"       "past"
[16] "winter"     "maga"       "https"      "t.co"       "a5if0qhnic"
```

- Note that the function strips characters that are important in the context of Twitter (e.g., @JerryJrFalwell)
- A *token* in the context of Twitter is not the same as in the context of spoken or written English

Quick detour: The unnest_tokens function

- Let us fix this:

```
campaign_tweets[i,] %>%  
  unnest_tokens(word, text, token = "tweets") %>%  
  pull(word)  
[1] "great"      "to"          "be"  
[4] "back"       "in"          "iowa"  
[7] "#tbt"       "with"        "@jerryjrfalwell"  
[10] "joining"    "me"          "in"  
[13] "davenport" "this"        "past"  
[16] "winter"     "#maga"       "https://t.co/a5if0qhnic"
```

Quick detour: The unnest_tokens function

- Let us fix this:

```
campaign_tweets[i,] %>%  
  unnest_tokens(word, text, token = "tweets") %>%  
  pull(word)  
[1] "great"          "to"              "be"  
[4] "back"           "in"              "iowa"  
[7] "#tbt"           "with"            "@jerryjrfalwell"  
[10] "joining"        "me"              "in"  
[13] "davenport"      "this"            "past"  
[16] "winter"         "#maga"           "https://t.co/a5if0qhnic"
```

- Another minor adjustment, let us remove the links to photos

```
links <- "https://t.co/[A-Za-z\\d]+!&";  
campaign_tweets[i,] %>%  
  mutate(text = str_replace_all(text, links, "")) %>%  
  unnest_tokens(word, text, token = "tweets") %>%  
  pull(word)
```

Case Study: Trump Tweets

- Now we are ready to extract the words for all our tweets

```
tweet_words <- campaign_tweets %>%  
  mutate(text = str_replace_all(text, links, "")) %>%  
  unnest_tokens(word, text, token = "tweets")
```

	source	id_str	created_at	retweet_count	in_reply_to_user_id_str	favorite_count	is_retweet	word
1	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	why
2	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	did
3	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	@danaperino
4	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	beg
5	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	me

Case Study: Trump Tweets

- Now we are ready to extract the words for all our tweets

```
tweet_words <- campaign_tweets %>%  
  mutate(text = str_replace_all(text, links, "")) %>%  
  unnest_tokens(word, text, token = "tweets")
```

	source	id_str	created_at	retweet_count	in_reply_to_user_id_str	favorite_count	is_retweet	word
1	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	why
2	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	did
3	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	@danaperino
4	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	beg
5	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	me

- What are the most commonly used words?

Case Study: Trump Tweets

- Now we are ready to extract the words for all our tweets

```
tweet_words <- campaign_tweets %>%  
  mutate(text = str_replace_all(text, links, "")) %>%  
  unnest_tokens(word, text, token = "tweets")
```

	source	id_str	created_at	retweet_count	in_reply_to_user_id_str	favorite_count	is_retweet	word
1	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	why
2	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	did
3	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	@danaperino
4	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	beg
5	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	me

- What are the most commonly used words?

```
tweet_words %>%  
  count(word) %>%  
  arrange(desc(n))
```

Case Study: Trump Tweets

```
tweet_words %>%  
  count(word) %>%  
  arrange(desc(n))
```

	word	n
1	the	2329
2	to	1410
3	and	1239
4	in	1185
5	i	1143
6	a	1112
7	you	999
8	of	982
9	is	942
10	on	874

- Comments?

Case Study: Trump Tweets

- The top words are not very informative!

Case Study: Trump Tweets

- The top words are not very informative!
- This is common in text analysis. Luckily, the *tidytext* package has a database of commonly used words called *stop_words*

```
tweet_words <- campaign_tweets %>%  
  mutate(text = str_replace_all(text, links, "")) %>%  
  unnest_tokens(word, text, token = "tweets") %>%  
  filter(!word %in% stop_words$word)
```

Case Study: Trump Tweets

- The top words are not very informative!
- This is common in text analysis. Luckily, the *tidytext* package has a database of commonly used words called *stop_words*

```
tweet_words <- campaign_tweets %>%  
  mutate(text = str_replace_all(text, links, "")) %>%  
  unnest_tokens(word, text, token = "tweets") %>%  
  filter(!word %in% stop_words$word)
```

```
tweet_words %>%  
  count(word) %>%  
  top_n(10, n) %>%  
  mutate(word = reorder(word, n)) %>%  
  arrange(desc(n))
```

Case Study: Trump Tweets

	word	n
1	#trump2016	414
2	hillary	405
3	people	303
4	#makeamericagreatagain	294
5	america	254
6	clinton	237
7	poll	217
8	crooked	205
9	trump	195
10	cruz	159

- Much better!

Case Study: Trump Tweets

- After some exploration, here is the final dataset

```
tweet_words <- campaign_tweets %>%
  mutate(text = str_replace_all(text, links, "")) %>%
  unnest_tokens(word, text, token = "tweets") %>%
  filter(!word %in% stop_words$word &
         !str_detect(word, "^\\d+$")) %>%
  mutate(word = str_replace(word, "^'", ""))
```

	source	id_str	created_at	retweet_count	in_reply_to_user_id_str	favorite_count	is_retweet	word
1	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	@danaperino
2	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	beg
3	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	tweet
4	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	endorsement
5	Android	612063082186174464	2015-06-19 20:03:05	166	NA	348	FALSE	book

Sentiment analysis (SA)

- In sentiment analysis, we assign a word to one or more “sentiments” (*e.g.*, happy)

Sentiment analysis (SA)

- In sentiment analysis, we assign a word to one or more “sentiments” (*e.g.*, happy)
- The *textdata* package provides a bevy of lexicons to conduct SA

Sentiment analysis (SA)

- In sentiment analysis, we assign a word to one or more “sentiments” (e.g., happy)
- The *textdata* package provides a bevy of lexicons to conduct SA
- We will use the nrc (Mohammad and Turney, 2012) lexicon because it provides several different sentiments

Sentiment analysis (SA)

- In sentiment analysis, we assign a word to one or more “sentiments” (e.g., happy)
- The *textdata* package provides a bevy of lexicons to conduct SA
- We will use the nrc (Mohammad and Turney, 2012) lexicon because it provides several different sentiments

```
nrc <- get_sentiments("nrc") %>%  
  select(word, sentiment)
```

	word	sentiment
1	suspicious	anger
2	tobacco	negative
3	mum	fear
4	vote	anger
5	irrevocable	negative
6	exquisitely	positive
7	disappear	fear
8	instigate	negative
9	sickness	disgust
10	hag	negative

Sentiment analysis (SA)

- Let us use `inner_join` to explore a random sample of the words in the tweets and their associated sentiment

```
tweet_words %>%  
  inner_join(nrc, by = "word") %>%  
  select(source, word, sentiment) %>%  
  sample_n(10)
```

	source	word	sentiment
1	Android	friend	positive
2	iPhone	invitation	anticipation
3	iPhone	happy	anticipation
4	iPhone	budget	trust
5	iPhone	shatter	fear
6	Android	horrible	negative
7	Android	core	positive
8	Android	president	positive
9	Android	deal	anticipation
10	Android	happy	anticipation

Sentiment analysis (SA)

- We are ready to perform an analysis comparing Android v iPhone sentiments

Sentiment analysis (SA)

- We are ready to perform an analysis comparing Android v iPhone sentiments
- Each tweet has several words, hence assigning a sentiment to each tweet is challenging (but possible!)

Sentiment analysis (SA)

- We are ready to perform an analysis comparing Android v iPhone sentiments
- Each tweet has several words, hence assigning a sentiment to each tweet is challenging (but possible!)
- We will conduct a simpler analysis. Specifically, we will compare the frequencies of the sentiments appearing in each device

Sentiment analysis (SA)

- We are ready to perform an analysis comparing Android v iPhone sentiments
- Each tweet has several words, hence assigning a sentiment to each tweet is challenging (but possible!)
- We will conduct a simpler analysis. Specifically, we will compare the frequencies of the sentiments appearing in each device
- Time for some wrangling

Sentiment analysis (SA)

```
sentiment_counts <- tweet_words %>%  
  left_join(nrc, by = "word") %>%  
  count(source, sentiment) %>%  
  pivot_wider(names_from = "source", values_from = "n") %>%  
  mutate(sentiment = replace_na(sentiment, replace = "none"))
```

	sentiment	Android	iPhone
1	anger	958	528
2	anticipation	910	715
3	disgust	638	322
4	fear	795	486
5	joy	688	535
6	negative	1641	929
7	positive	1806	1473
8	sadness	894	515
9	surprise	518	365
10	trust	1236	990
11	none	11904	10766

Sentiment analysis (SA)

```
sentiment_counts <- tweet_words %>%  
  left_join(nrc, by = "word") %>%  
  count(source, sentiment) %>%  
  pivot_wider(names_from = "source", values_from = "n") %>%  
  mutate(sentiment = replace_na(sentiment, replace = "none"))
```

	sentiment	Android	iPhone
1	anger	958	528
2	anticipation	910	715
3	disgust	638	322
4	fear	795	486
5	joy	688	535
6	negative	1641	929
7	positive	1806	1473
8	sadness	894	515
9	surprise	518	365
10	trust	1236	990
11	none	11904	10766

- Comments? Thoughts?

Sentiment analysis (SA)

```
sentiment_counts <- tweet_words %>%  
  left_join(nrc, by = "word") %>%  
  count(source, sentiment) %>%  
  pivot_wider(names_from = "source", values_from = "n") %>%  
  mutate(sentiment = replace_na(sentiment, replace = "none"))
```

	sentiment	Android	iPhone
1	anger	958	528
2	anticipation	910	715
3	disgust	638	322
4	fear	795	486
5	joy	688	535
6	negative	1641	929
7	positive	1806	1473
8	sadness	894	515
9	surprise	518	365
10	trust	1236	990
11	none	11904	10766

- Comments? Thoughts?
- Should we compare these counts? Is this a “fair” comparison?

Sentiment analysis (SA)

```
sentiment_counts <- tweet_words %>%  
  left_join(nrc, by = "word") %>%  
  count(source, sentiment) %>%  
  pivot_wider(names_from = "source", values_from = "n") %>%  
  mutate(sentiment = replace_na(sentiment, replace = "none"))
```

	sentiment	Android	iPhone
1	anger	958	528
2	anticipation	910	715
3	disgust	638	322
4	fear	795	486
5	joy	688	535
6	negative	1641	929
7	positive	1806	1473
8	sadness	894	515
9	surprise	518	365
10	trust	1236	990
11	none	11904	10766

- Comments? Thoughts?
- Should we compare these counts? Is this a “fair” comparison?
- The count for Android is 21,988, whereas the count for iPhone is 17,624

Sentiment analysis (SA)

```
sentiment_counts <- tweet_words %>%  
  left_join(nrc, by = "word") %>%  
  count(source, sentiment) %>%  
  pivot_wider(names_from = "source", values_from = "n") %>%  
  mutate(sentiment = replace_na(sentiment, replace = "none"))
```

	sentiment	Android	iPhone
1	anger	958	528
2	anticipation	910	715
3	disgust	638	322
4	fear	795	486
5	joy	688	535
6	negative	1641	929
7	positive	1806	1473
8	sadness	894	515
9	surprise	518	365
10	trust	1236	990
11	none	11904	10766

- Comments? Thoughts?
- Should we compare these counts? Is this a “fair” comparison?
- The count for Android is 21,988, whereas the count for iPhone is 17,624
- We need to use measure of association to compare the two devices.
- Let us use odds ratio

Brief introduction to odds ratio

- Suppose you are interested in two events **A** and **B**

Brief introduction to odds ratio

- Suppose you are interested in two events **A** and **B**
- For example, **A** can be the event that it rains tomorrow and **B** the event that its sunny tomorrow

Brief introduction to odds ratio

- Suppose you are interested in two events **A** and **B**
- For example, **A** can be the event that it rains tomorrow and **B** the event that its sunny tomorrow
- We define the odds of **A** with:

$$\text{Odds of A} = \frac{\text{Chances that A happens}}{\text{Chances that A does not happen}}$$

Brief introduction to odds ratio

- Suppose you are interested in two events **A** and **B**
- For example, **A** can be the event that it rains tomorrow and **B** the event that its sunny tomorrow
- We define the odds of **A** with:

$$\text{Odds of A} = \frac{\text{Chances that A happens}}{\text{Chances that A does not happen}}$$

- We define the odds of **B** in a similar way
- Finally, we define the odds ratio of **A** relative to **B** with:

$$\text{OR(A,B)} = \text{Odds of A} / \text{Odds of B}$$

Brief introduction to odds ratio

- Suppose you are interested in two events **A** and **B**
- For example, **A** can be the event that it rains tomorrow and **B** the event that its sunny tomorrow

- We define the odds of **A** with:

$$\text{Odds of A} = \frac{\text{Chances that A happens}}{\text{Chances that A does not happen}}$$

- We define the odds of **B** in a similar way
- Finally, we define the odds ratio of **A** relative to **B** with:

$$OR(A,B) = \text{Odds of A} / \text{Odds of B}$$

- If $OR > 1$, then **A** is more frequent than **B** and if $OR < 1$ the opposite is true
- If $OR = 1$, then the frequency is the same

Sentiment analysis (SA)

```
sentiment_counts %>%
  mutate(Android = Android / (sum(Android) - Android),
         iPhone = iPhone / (sum(iPhone) - iPhone),
         or_android_iphone = Android/iPhone) %>%
  arrange(desc(or_android_iphone))
```

	sentiment	Android	iPhone	or_android_iphone
1	disgust	0.02988290	0.01861057	1.6056957
2	anger	0.04555397	0.03088442	1.4749823
3	negative	0.08065071	0.05564540	1.4493688
4	sadness	0.04238172	0.03010112	1.4079783
5	fear	0.03751239	0.02835803	1.3228133
6	surprise	0.02412669	0.02114839	1.1408288
7	joy	0.03230047	0.03130669	1.0317434
8	anticipation	0.04317298	0.04228517	1.0209956
9	trust	0.05956052	0.05951665	1.0007371
10	positive	0.08948568	0.09120178	0.9811835
11	none	1.18048393	1.56984544	0.7519746

- Comments? Thoughts?

Sentiment analysis (SA)

```
sentiment_counts %>%  
  mutate(Android = Android / (sum(Android) - Android),  
         iPhone = iPhone / (sum(iPhone) - iPhone),  
         or_android_iphone = Android/iPhone) %>%  
  arrange(desc(or_android_iphone))
```

	sentiment	Android	iPhone	or_android_iphone
1	disgust	0.02988290	0.01861057	1.6056957
2	anger	0.04555397	0.03088442	1.4749823
3	negative	0.08065071	0.05564540	1.4493688
4	sadness	0.04238172	0.03010112	1.4079783
5	fear	0.03751239	0.02835803	1.3228133
6	surprise	0.02412669	0.02114839	1.1408288
7	joy	0.03230047	0.03130669	1.0317434
8	anticipation	0.04317298	0.04228517	1.0209956
9	trust	0.05956052	0.05951665	1.0007371
10	positive	0.08948568	0.09120178	0.9811835
11	none	1.18048393	1.56984544	0.7519746

- Comments? Thoughts?
- Some interesting differences

Sentiment analysis (SA)

```
sentiment_counts %>%  
  mutate(Android = Android / (sum(Android) - Android),  
         iPhone = iPhone / (sum(iPhone) - iPhone),  
         or_android_iphone = Android/iPhone) %>%  
  arrange(desc(or_android_iphone))
```

	sentiment	Android	iPhone	or_android_iphone
1	disgust	0.02988290	0.01861057	1.6056957
2	anger	0.04555397	0.03088442	1.4749823
3	negative	0.08065071	0.05564540	1.4493688
4	sadness	0.04238172	0.03010112	1.4079783
5	fear	0.03751239	0.02835803	1.3228133
6	surprise	0.02412669	0.02114839	1.1408288
7	joy	0.03230047	0.03130669	1.0317434
8	anticipation	0.04317298	0.04228517	1.0209956
9	trust	0.05956052	0.05951665	1.0007371
10	positive	0.08948568	0.09120178	0.9811835
11	none	1.18048393	1.56984544	0.7519746

- Comments? Thoughts?
- Some interesting differences
- Perhaps there is meaning in the order of the sentiments

Sentiment analysis (SA)

```
sentiment_counts %>%  
  mutate(Android = Android / (sum(Android) - Android),  
         iPhone = iPhone / (sum(iPhone) - iPhone),  
         or_android_iphone = Android/iPhone) %>%  
  arrange(desc(or_android_iphone))
```

	sentiment	Android	iPhone	or_android_iphone
1	disgust	0.02988290	0.01861057	1.6056957
2	anger	0.04555397	0.03088442	1.4749823
3	negative	0.08065071	0.05564540	1.4493688
4	sadness	0.04238172	0.03010112	1.4079783
5	fear	0.03751239	0.02835803	1.3228133
6	surprise	0.02412669	0.02114839	1.1408288
7	joy	0.03230047	0.03130669	1.0317434
8	anticipation	0.04317298	0.04228517	1.0209956
9	trust	0.05956052	0.05951665	1.0007371
10	positive	0.08948568	0.09120178	0.9811835
11	none	1.18048393	1.56984544	0.7519746

- Comments? Thoughts?
- Some interesting differences
- Perhaps there is meaning in the order of the sentiments
- Are these differences due to chance? (note that we are venturing into inference land)

Sentiment analysis (SA)

```
sentiment_counts %>%  
  mutate(Android = Android / (sum(Android) - Android),  
         iPhone = iPhone / (sum(iPhone) - iPhone),  
         or_android_iphone = Android/iPhone) %>%  
  arrange(desc(or_android_iphone))
```

	sentiment	Android	iPhone	or_android_iphone
1	disgust	0.02988290	0.01861057	1.6056957
2	anger	0.04555397	0.03088442	1.4749823
3	negative	0.08065071	0.05564540	1.4493688
4	sadness	0.04238172	0.03010112	1.4079783
5	fear	0.03751239	0.02835803	1.3228133
6	surprise	0.02412669	0.02114839	1.1408288
7	joy	0.03230047	0.03130669	1.0317434
8	anticipation	0.04317298	0.04228517	1.0209956
9	trust	0.05956052	0.05951665	1.0007371
10	positive	0.08948568	0.09120178	0.9811835
11	none	1.18048393	1.56984544	0.7519746

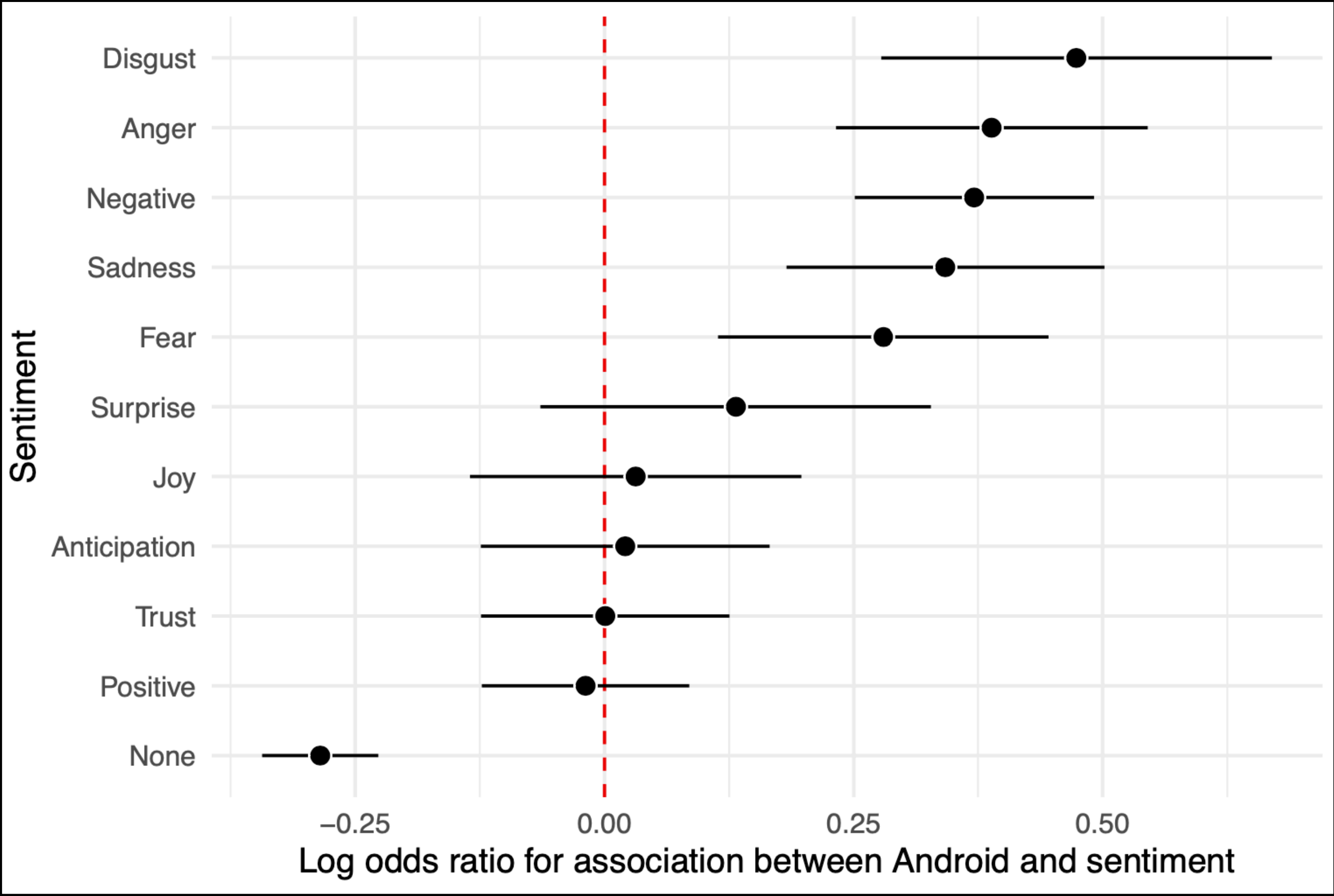
- Comments? Thoughts?
- Some interesting differences
- Perhaps there is meaning in the order of the sentiments
- Are these differences due to chance? (note that we are venturing into inference land)
- Let us compute confidence intervals for each sentiment

Sentiment analysis (SA)

```
log_or <- sentiment_counts %>%
  mutate(log_or = log((Android / (sum(Android) - Android)) /
                      (iPhone / (sum(iPhone) - iPhone))),
         se = sqrt(1/Android + 1/(sum(Android) - Android) +
                  1/iPhone + 1/(sum(iPhone) - iPhone)),
         conf.low = log_or - qnorm(0.975)*se,
         conf.high = log_or + qnorm(0.975)*se) %>%
  arrange(desc(log_or))

log_or %>%
  mutate(sentiment = str_to_title(sentiment),
         sentiment = reorder(sentiment, log_or)) %>%
  ggplot(aes(x = sentiment, ymin = conf.low, ymax = conf.high)) +
  geom_hline(yintercept = 0, color = "red2", lty = 2) +
  geom_errorbar(width = 0) +
  geom_point(aes(sentiment, log_or),
            shape = 21,
            color = "white",
            fill = "black",
            size = 3) +
  labs(x = "Sentiment", y = "Log odds ratio for association between Android and sentiment") +
  coord_flip() +
  theme_minimal()
```

Sentiment analysis (SA)

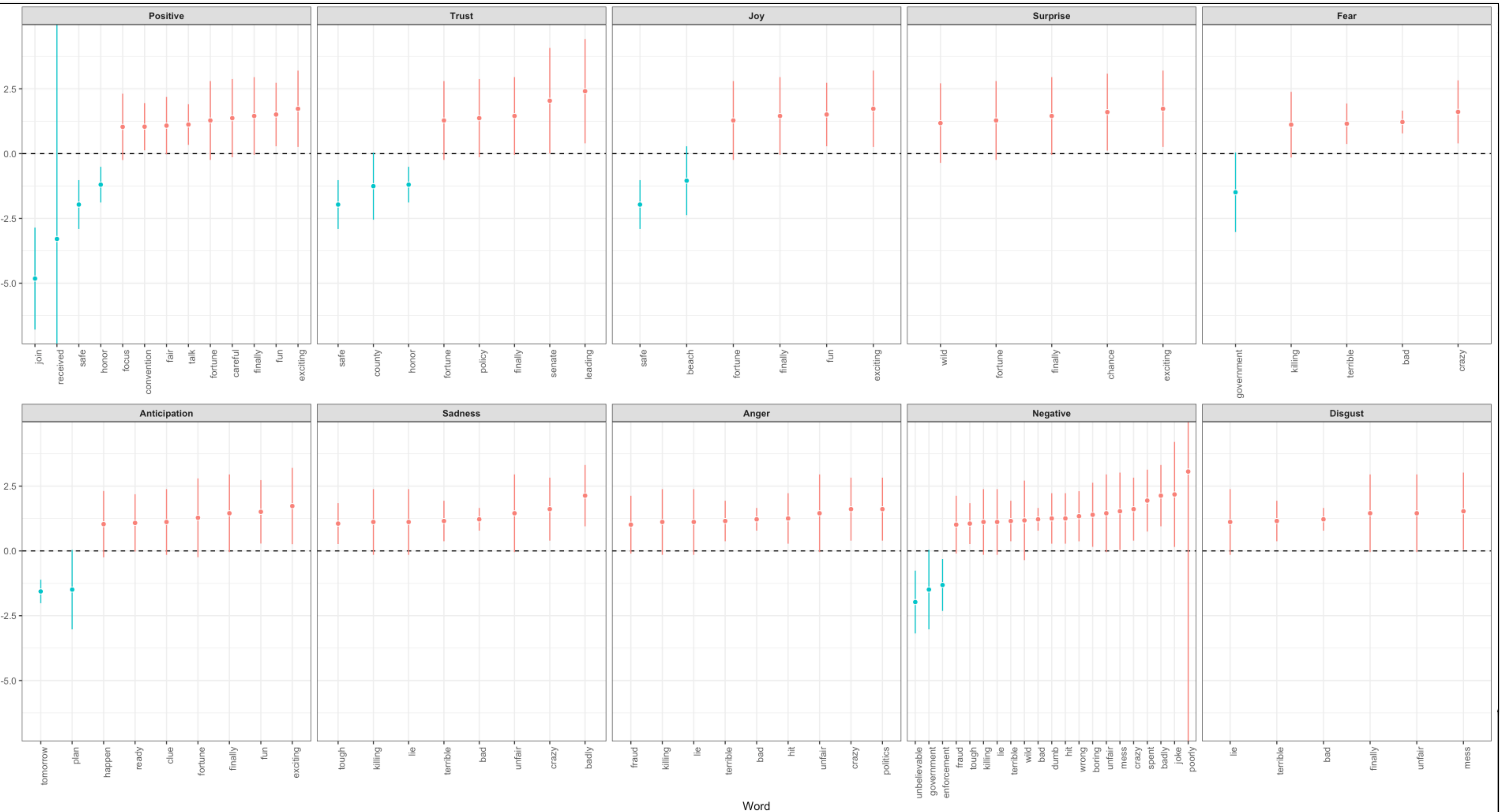


Sentiment analysis (SA)

- Finally, we can explore which specific words are driving these differences

```
tweet_words %>%
  count(word, source) %>%
  pivot_wider(names_from = "source", values_from = "n", values_fill = 0) %>%
  mutate(or = (Android + 0.5) / (sum(Android) - Android + 0.5) /
           ( (iPhone + 0.5) / (sum(iPhone) - iPhone + 0.5))) %>%
  inner_join(nrc, by = "word") %>%
  mutate(sentiment = factor(sentiment, levels = log_or$sentiment)) %>%
  mutate(log_or = log(or)) %>%
  mutate(sentiment = str_to_title(sentiment),
         sentiment = reorder(sentiment, log_or)) %>%
  filter(Android + iPhone > 10 & abs(log_or) > 1) %>%
  mutate(word = reorder(word, log_or),
         se = sqrt(1/Android + 1/(sum(Android) - Android) +
                  1/iPhone + 1/(sum(iPhone) - iPhone)),
         conf.low = log_or - qnorm(0.975)*se,
         conf.high = log_or + qnorm(0.975)*se) %>%
  ggplot(aes(word, log_or, fill = log_or < 0, color = log_or < 0, ymin = conf.low, ymax = conf.high)) +
  geom_hline(yintercept = 0, lty = 2) +
  facet_wrap(~sentiment, scales = "free_x", nrow = 2) +
  geom_errorbar(width = 0,
               show.legend = FALSE) +
  geom_point(aes(word, log_or),
            shape = 21,
            color = "white",
            size = 2,
            show.legend = FALSE) +
  labs(y = "Log OR for association between Android and sentiment",
       x = "Word") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        strip.text = element_text(face = "bold"))
```


Log OR for association between Android and sentiment



References

1. Introduction to Data Science: Data analysis and prediction algorithms with R by Rafael A. Irizarry, Chapter 26. <https://rafalab.github.io/dsbook/>

Referencias en español:

1. Introducción a la Ciencia de Datos: Análisis de datos y algoritmos de predicción con R por Rafael A. Irizarry, Capítulo 26. <https://rafalab.github.io/dslibro/>

Your turn!

[Click here for the class website](#)