36-668: Coffee Break Experiment 1

Arsh Gupta

Loading in libraries.

```
library(readtext)
library(quanteda)
library(cmu.textstat)
library(quanteda.textstats)
library(dplyr)
library(tidyr)
library(knitr)
library(ggraph)
library(igraph)
```

Use `readtext()` function from `readtext` package to create a `data.frame` object.

```
rt1 <- readtext("album_19_1/*.txt",
                docvarsfrom = "filenames",
                docvarnames = c("type", "album", "song_rank"))

rt2 <- readtext("album_21_2/*.txt",
                docvarsfrom = "filenames",
                docvarnames = c("type", "album", "song_rank"))

rt3 <- readtext("album_25_3/*.txt",
                docvarsfrom = "filenames",
                docvarnames = c("type", "album", "song_rank"))

rt4 <- readtext("album_30_4/*.txt",
                docvarsfrom = "filenames",
                docvarnames = c("type", "album", "song_rank"))
```

Make a `corpus` object.

```
album1_corpus <- corpus(rt1)
album2_corpus <- corpus(rt2)
album3_corpus <- corpus(rt3)
album4_corpus <- corpus(rt4)

rt_12 <- rbind(rt1, rt2)
rt_34 <- rbind(rt3, rt4)
rt_all <- rbind(rt1, rt2, rt3, rt4)

albums_12_corpus <- corpus(rt_12)
albums_34_corpus <- corpus(rt_34)
all_albums_corpus <- corpus(rt_all)
```

Check the result.

```
knitr::kable(head(album1_corpus %>% summary()), caption = "Partial summary of album1 corpus.")
```

Table 1: Partial summary of album1 corpus.

Text	Types	Tokens	Sentences	type	album	song_rank
song_1_1.txt	107	197	1	song	1	1
song_1_10.txt	110	370	1	song	1	10
song_1_11.txt	108	331	17	song	1	11
song_1_12.txt	100	246	3	song	1	12
song_1_2.txt	160	439	4	song	1	2
song_1_3.txt	89	325	27	song	1	3

```
knitr::kable(head(album2_corpus %>% summary()), caption = "Partial summary of album2 corpus.")
```

Table 2: Partial summary of album2 corpus.

Text	Types	Tokens	Sentences	type	album	song_rank
song_2_1.txt	147	625	1	song	2	1
song_2_10.txt	33	185	1	song	2	10
song_2_11.txt	130	384	3	song	2	11
song_2_2.txt	113	426	1	song	2	2
song_2_3.txt	91	259	1	song	2	3
song_2_4.txt	95	219	10	song	2	4

```
knitr::kable(head(album3_corpus %>% summary()), caption = "Partial summary of album3 corpus.")
```

Table 3: Partial summary of album3 corpus.

Text	Types	Tokens	Sentences	type	album	song_rank
song_3_1.txt	134	429	4	song	3	1
song_3_10.txt	119	297	5	song	3	10
song_3_11.txt	116	276	1	song	3	11
song_3_2.txt	96	442	3	song	3	2
song_3_3.txt	114	440	1	song	3	3
song_3_4.txt	133	405	4	song	3	4

```
knitr::kable(head(album4_corpus %>% summary()), caption = "Partial summary of album4 corpus.")
```

Table 4: Partial summary of album4 corpus.

Text	Types	Tokens	Sentences	type	album	song_rank
song_4_1.txt	87	128	2	song	4	1
song_4_10.txt	130	413	4	song	4	10
song_4_11.txt	130	354	2	song	4	11

Text	Types	Tokens	Sentences	type	album	song_rank
song_4_12.txt	141	396	3	song	4	12
song_4_2.txt	97	213	1	song	4	2
song_4_3.txt	190	489	9	song	4	3

We'll use **quanteda** to tokenize. And after tokenization, we'll convert them to lower case. As a next step, we'll being combining tokens like *a* and *lot* into single units. And we'll be using a list of expressions that isn't case sensitive.

```
album1_tokens <- tokens(album1_corpus,
                        include_docvars = TRUE,
                        remove_punct = TRUE,
                        remove_numbers = TRUE,
                        remove_symbols = TRUE,
                        what = "word")

album1_tokens <- tokens_tolower(album1_tokens)

album2_tokens <- tokens(album2_corpus,
                        include_docvars = TRUE,
                        remove_punct = TRUE,
                        remove_numbers = TRUE,
                        remove_symbols = TRUE,
                        what = "word")

album2_tokens <- tokens_tolower(album2_tokens)

album3_tokens <- tokens(album3_corpus,
                        include_docvars = TRUE,
                        remove_punct = TRUE,
                        remove_numbers = TRUE,
                        remove_symbols = TRUE,
                        what = "word")

album3_tokens <- tokens_tolower(album3_tokens)

album4_tokens <- tokens(album4_corpus,
                        include_docvars = TRUE,
                        remove_punct = TRUE,
                        remove_numbers = TRUE,
                        remove_symbols = TRUE, what = "word")

album4_tokens <- tokens_tolower(album4_tokens)

all_albums_tokens <- tokens(all_albums_corpus,
                        include_docvars=TRUE,
                        remove_punct = TRUE,
                        remove_numbers = TRUE,
                        remove_symbols = TRUE,
                        what = "word")

albums_12_tokens <- tokens(albums_12_corpus,
                        include_docvars=TRUE,
```

```
                          remove_punct = TRUE,
                          remove_numbers = TRUE,
                          remove_symbols = TRUE,
                          what = "word")

albums_34_tokens <- tokens(albums_34_corpus,
                          include_docvars=TRUE,
                          remove_punct = TRUE,
                          remove_numbers = TRUE,
                          remove_symbols = TRUE,
                          what = "word")
```

An issue that we run into frequently with corpus analysis is what to do with multi-word expressions. For example, consider a common English quantifier: "a lot". Typical tokenization rules will split this into two tokens: *a* and *lot*. But counting *a lot* as a single unit might be important depending on our task. We have a way of telling **quanteda** to account for these tokens.

All that we need is a list of multi-word expressions.

The **cmu.textstat** comes with an example of an mwe list called **multiword_expressions**:

```
album1_tokens <- tokens_compound(album1_tokens,
                                 pattern = phrase(multiword_expressions))

album2_tokens <- tokens_compound(album2_tokens,
                                 pattern = phrase(multiword_expressions))

album3_tokens <- tokens_compound(album3_tokens,
                                 pattern = phrase(multiword_expressions))

album4_tokens <- tokens_compound(album4_tokens,
                                 pattern = phrase(multiword_expressions))

all_albums_tokens <- tokens_compound(all_albums_tokens,
                                     pattern = phrase(multiword_expressions))

albums_12_tokens <- tokens_compound(albums_12_tokens,
                                    pattern = phrase(multiword_expressions))

albums_34_tokens <- tokens_compound(albums_34_tokens,
                                    pattern = phrase(multiword_expressions))
```

With our tokens object we can now create a document-feature-matrix using the **dfm()** function. As a reminder, a **dfm** is table with one row per document in the corpus, and one column per unique token in the corpus. Each cell contains a count of how many times a token shows up in that document.

```
album1_dfm <- dfm(album1_tokens)
album2_dfm <- dfm(album2_tokens)
album3_dfm <- dfm(album3_tokens)
album4_dfm <- dfm(album4_tokens)

all_albums_dfm <- dfm(all_albums_tokens)
albums_12_dfm <- dfm(albums_12_tokens)
albums_34_dfm <- dfm(albums_34_tokens)
```

Next we'll create a **dfm** with proportionally weighted counts. We will create another corpus for all the albums.

```
prop_album1_dfm <- dfm_weight(album1_dfm, scheme = "prop")
prop_album2_dfm <- dfm_weight(album2_dfm, scheme = "prop")
prop_album3_dfm <- dfm_weight(album3_dfm, scheme = "prop")
prop_album4_dfm <- dfm_weight(album4_dfm, scheme = "prop")

prop_all_albums_dfm <- dfm_weight(all_albums_dfm, scheme = "prop")
prop_albums_12_dfm <- dfm_weight(albums_12_dfm, scheme = "prop")
prop_albums_34_dfm <- dfm_weight(albums_34_dfm, scheme = "prop")
```

Use textstat_frequency to calculate the frequencies for the entire 4 albums, for albums 1 and 2, and for albums 3 and 4.

```
freq_df <- textstat_frequency(all_albums_dfm) %>%
  data.frame(stringsAsFactors = F) %>%
  select(feature, frequency) %>%
  rename("Token" = "feature", "Frequency" = "frequency")
```

```
## as(<dgCMatrix>, "dgTMatrix") is deprecated since Matrix 1.5-0; do as(., "TsparseMatrix") instead
```

```
kable(head(freq_df))
```

Token	Frequency
i	749
you	569
the	427
me	370
to	360
it	305

```
freq_df_12 <- textstat_frequency(albums_12_dfm) %>%
  data.frame(stringsAsFactors = F) %>%
  select(feature, frequency) %>%
  rename("Token" = "feature", "Frequency" = "frequency")

kable(head(freq_df_12))
```

Token	Frequency
i	340
you	319
the	186
it	177
and	158
to	157

```
freq_df_34 <- textstat_frequency(albums_34_dfm) %>%
  data.frame(stringsAsFactors = F) %>%
  select(feature, frequency) %>%
  rename("Token" = "feature", "Frequency" = "frequency")

kable(head(freq_df_34))
```

Token	Frequency
i	409
you	250
the	241
me	213
to	203
my	129

Now, we calculate dispersion tokens for our two dfm's

```
dispersion_all_albums <- all_albums_dfm %>% dispersions_all()
dispersion_albums_12 <- albums_12_dfm %>% dispersions_all()
dispersion_albums_34 <- albums_34_dfm %>% dispersions_all()

kable(head(dispersion_all_albums))
```

Token	AF	Per_10.4	Carrolls_D2	Rosengrens_S	Lynes_D3	DC	Juillands_D	DP	DP_norm
i	749	519.4175	0.9540649	0.9195357	0.9137127	0.9087672	0.9212488	0.2225919	0.2243971
you	569	394.5908	0.9318796	0.8564299	0.8855290	0.8432097	0.8919461	0.2702648	0.2724565
the	427	296.1165	0.9262236	0.8653860	0.8553535	0.8349821	0.8983312	0.2610946	0.2632119
me	370	256.5881	0.8906300	0.7926759	0.7841198	0.7538687	0.8748004	0.3466788	0.3494902
to	360	249.6533	0.9434388	0.9029249	0.8998534	0.8768071	0.9095448	0.2387463	0.2406825
it	305	211.5118	0.8216798	0.6733847	0.6137141	0.6002417	0.8064244	0.4209579	0.4243717

```
kable(head(dispersion_albums_12))
```

Token	AF	Per_10.3	Carrolls_D2	Rosengrens_S	Lynes_D3	DC	Juillands_D	DP	DP_norm
i	340	50.18450	0.9474056	0.9118498	0.9194247	0.9151329	0.8780098	0.2316171	0.2373270
you	319	47.08487	0.9349497	0.8909239	0.9062804	0.8895045	0.8576650	0.2503515	0.2565232
the	186	27.45387	0.8755934	0.8035047	0.8112210	0.7588310	0.8304722	0.2944705	0.3017298
it	177	26.12546	0.7408726	0.6028532	0.5659612	0.5196397	0.7068907	0.4640207	0.4754598
and	158	23.32103	0.9359418	0.8850839	0.9077271	0.8649630	0.8709134	0.2179028	0.2232745
to	157	23.17343	0.9101962	0.8765525	0.8676620	0.8375918	0.8497680	0.2693435	0.2759834

```
kable(head(dispersion_albums_34))
```

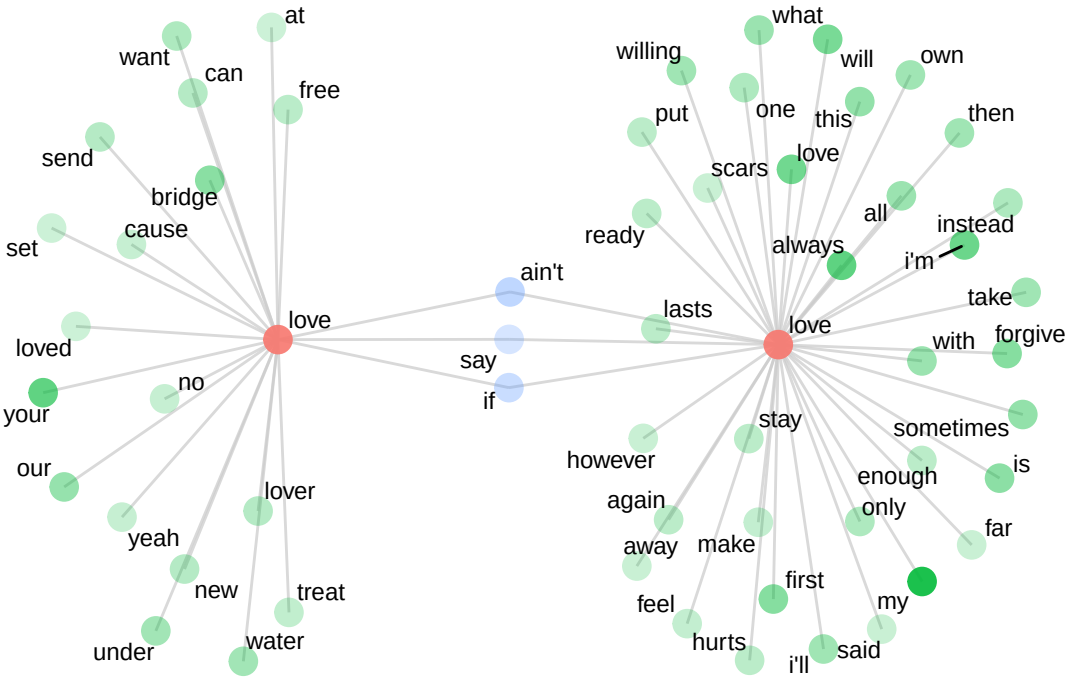
Token	AF	Per_10.3	Carrolls_D2	Rosengrens_S	Lynes_D3	DC	Juillands_D	DP	DP_norm
i	409	53.49902	0.9434864	0.9265943	0.9145510	0.9066743	0.8945252	0.2136491	0.2169408
you	250	32.70111	0.8990380	0.8349220	0.8656240	0.7988573	0.8466382	0.2850804	0.2894727
the	241	31.52387	0.9411030	0.9202259	0.8919182	0.9117209	0.8746136	0.2409157	0.2446276
me	213	27.86135	0.8742405	0.8230701	0.7908263	0.7757417	0.8330632	0.3248857	0.3298912
to	203	26.55330	0.9516044	0.9269732	0.9280982	0.9189583	0.8885167	0.2185472	0.2219143
my	129	16.87377	0.9104023	0.8592294	0.8619674	0.8414555	0.8548804	0.2827151	0.2870709

Now we create collocates!

```
albums_12_love_collocates <- collocates_by_MI(albums_12_tokens, "love") %>%
  filter(col_freq >= 4 & MI_1 >= 4)

albums_12_heart_collocates <- collocates_by_MI(albums_12_tokens, "heart") %>%
  filter(col_freq >= 4 & MI_1 >= 4)

albums_34_love_collocates <- collocates_by_MI(albums_34_tokens, "love") %>%
  filter(col_freq >= 4 & MI_1 >= 4)

albums_34_heart_collocates <- collocates_by_MI(albums_34_tokens, "heart") %>%
  filter(col_freq >= 4 & MI_1 >= 4)
```

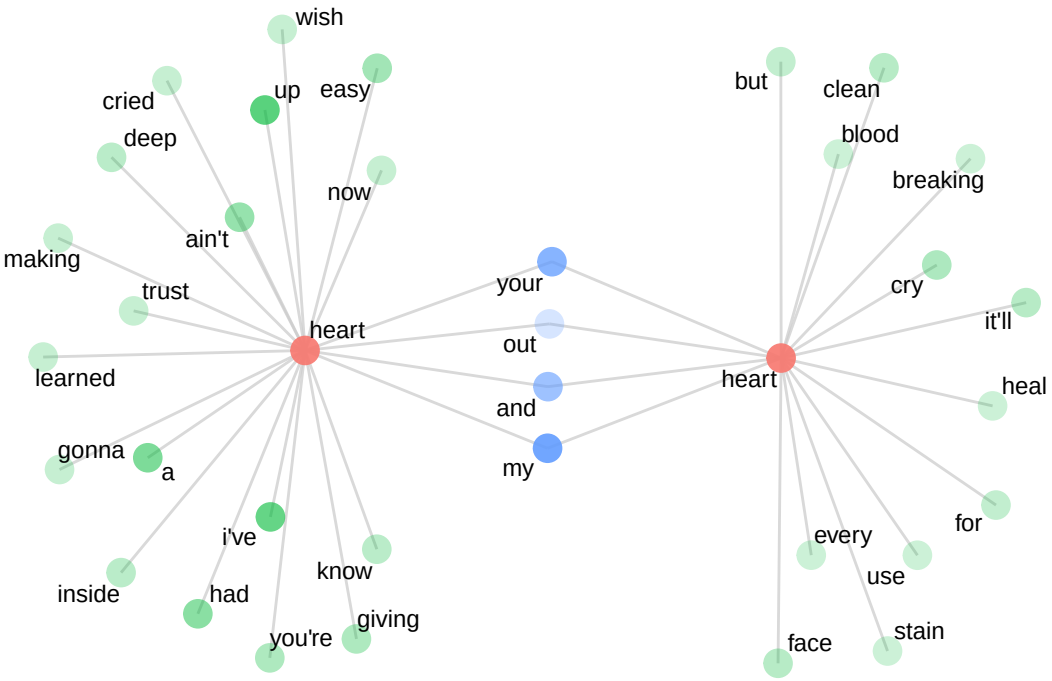
Now, we create a graph for `love`.

```
net <- col_network(albums_12_love_collocates, albums_34_love_collocates)

ggraph(net, layout = "stress") +

  geom_edge_link(color = "grey80", alpha = 0.75) +

  geom_node_point(aes(alpha = node_weight, size = 3, color = n_intersects)) +

  geom_node_text(aes(label = label), repel = T, size = 3) +

  scale_alpha(range = c(0.2, 0.9)) +

  theme_graph() +

  theme(legend.position = "none")
```



Now, we create a graph for `heart`.

```
net <- col_network(albums_12_heart_collocates, albums_34_heart_collocates)

ggraph(net, layout = "stress") +

  geom_edge_link(color = "grey80", alpha = 0.75) +

  geom_node_point(aes(alpha = node_weight, size = 3, color = n_intersects)) +

  geom_node_text(aes(label = label), repel = T, size = 3) +

  scale_alpha(range = c(0.2, 0.9)) +

  theme_graph() +

  theme(legend.position = "none")
```


wish
cried
up
easy
deep
now
ain't
making
trust
heart
learned
gonna
a
i've
inside
had
know
you're
giving
your
out
and
my
but
clean
blood
breaking
cry
it'll
heart
heal
for
every
use
face
stain

Report

Introduction

Adele is a widely revered global artist who has been the recipient of multiple awards. Her songs are characterized by a combination of soulful, deep, and heartfelt emotions, the themes of which are shared across her four albums: 19, 21, 25, and 30. Shared among these is the theme of **love**, which has evolved in the way Adele characterizes it across her songs. The purpose of this report is to analyze the was that Adele's conceptualization of the theme of love has evolved across her different albums and what insights that gives us about the manner in which she communicates about this topic.

Data

The data set used for this report includes a corpus consisting of 46 text files. Each of those text files includes lyrics to each of Adele's 46 songs spanned across her four albums. The data was manually collected from the web through Google and compiled into a corpus using the `quanteda` library in R.

Methods

There were multiple steps of data processing that were adopted before conducting the analysis which have been described in the next section. We have also obtained summary statistics of our data at multiple steps for different albums to better understand the features of each album. Tables 1, 2, 3, and 4 show a partial summary of the four albums listing the number of tokens and overall sentences in each song of that particular album.

The data is then further processed to remove punctuation, symbols, and numbers from the tokens obtained. All the tokens are also changed into lowercase to ensure consistency and avoid redundancy. As a next step, we have combined all multi-word expressions into one token using the mwe list called `multiword_expressions` as found in the `cmu.textstat` library.

At this point, we have combined all tokens from the first two albums and last two albums into a single object leaving us with a total of two token objects since we will be analyzing the conceptualization of love as found in Adele's first two albums versus the last two albums.

Results

Discussion