# Week2_milestone_report

Tom Matsuno

2023-02-17

## Load data

I set the three dataset under the “Data_Science_Capstone” folder. Using the R base such as file connection and readLines, I connected each database and read every lines, and saved them to each variables.

```
setwd("/home/tommat/ドキュメント/Coursera勉強/Johns_Hopkins DATA SCIENCE/Data_Science_Capstone")
blogs_file_name <- "final/en_US/en_US.blogs.txt"
con <- file(blogs_file_name, open="r")
blogs <- readLines(con, encoding="UTF-8", skipNul=T)
close(con)

news_file_name <- "final/en_US/en_US.news.txt"
con <- file(news_file_name, open="r")
news <- readLines(con, encoding="UTF-8", skipNul=T)
close(con)

twitter_file_name <- "final/en_US/en_US.twitter.txt"
con <- file(twitter_file_name, open="r")
twitter <- readLines(con, encoding="UTF-8", skipNul=T)
close(con)
```

## Exploratory Data Analysis

At the beginning, I summarized the basic analysis of three data set. I calculated the each basic summary and combined them into “total” variables.

```
library(kableExtra)
library(quanteda)
```

```
## Package version: 3.2.4
## Unicode version: 14.0
## ICU version: 70.1
```

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

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

```
library(tokenizers)

num_words <- c(sum(count_words(blogs)),
              sum(count_words(news)),
              sum(count_words(twitter)))
num_lines <- c(length(blogs),
               length(news),
               length(twitter))
longest_line <- c(max(count_characters(blogs)),
                 max(count_characters(news)),
                 max(count_characters(twitter)))
mean_length <- c(round(mean(count_characters(blogs)),1),
                round(mean(count_characters(news)),1),
                round(mean(count_characters(twitter)),1))

total <- data.frame(num_words, num_lines, longest_line, mean_length)
colnames(total) <- c("Word Count","Line Count","Longest Line","Mean Length")
total %>% kable
```

| Word Count | Line Count | Longest Line | Mean Length |
|---:|---:|---:|---:|
| 37546250 | 899288 | 40833 | 230.0 |
| 34762395 | 1010242 | 11384 | 201.2 |
| 30093413 | 2360148 | 140 | 68.7 |

# Sampling and create corpus

Because of the limitation of my computer capacity, I restrict the number of row of data set. I randomly chose 10000 lines from three data set. Next, I create the corpus by using “quanteda” package.

```
sample_blogs <- sample(blogs, size=10000)
sample_news <- sample(news, size=10000)
sample_twitter <- sample(twitter, size=10000)

corpus_blogs <- corpus(sample_blogs)
corpus_news <- corpus(sample_news)
corpus_twitter <- corpus(sample_twitter)
```

# Create histograms of each corpus

By using “tokenizers” package, I clean each corpus with removing puctuations, numbers, symbols and url.

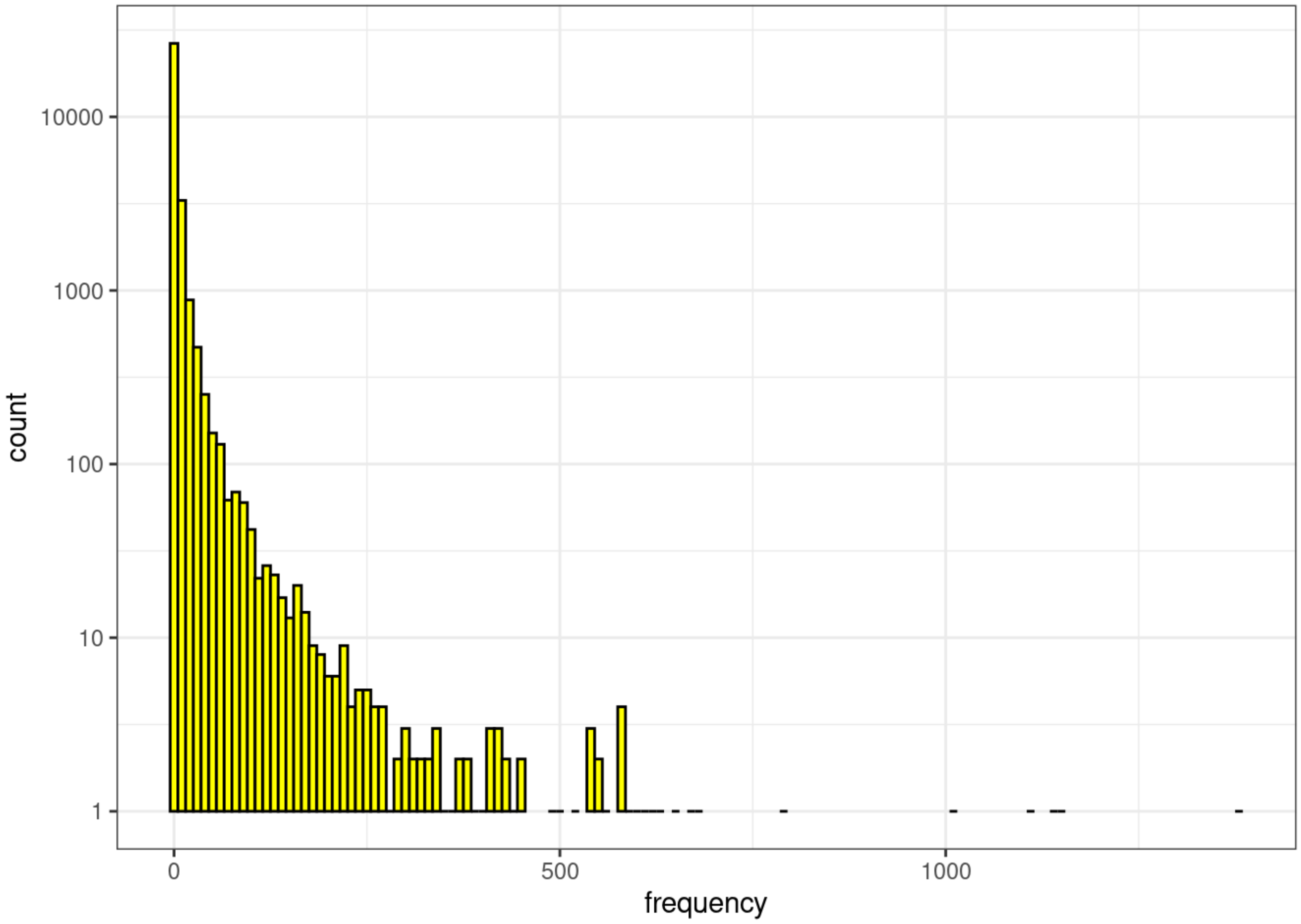
```
library(tokenizers)
suppressMessages(library(dplyr))
suppressMessages(library(ggplot2))
token_blogs <- tokens(corpus_blogs, remove_punct=T, remove_numbers=T,
                      remove_symbols=T, remove_url=T)
token_news <- tokens(corpus_news, remove_punct=T, remove_numbers=T,
                      remove_symbols=T, remove_url=T)
token_twitter <- tokens(corpus_twitter, remove_punct=T, remove_numbers=T,
                      remove_symbols=T, remove_url=T)
```

Next, using “quanteda.textstats” package, I counted the frequency of each words, then draw three histogram of each data set.

```
library(quanteda.textstats)
token_blogs %>%
  tokens_remove(pattern = stopwords('en')) %>%
  dfm(tolower=T) %>%
  textstat_frequency() %>%
  ggplot(aes(x=frequency)) +
  geom_histogram(binwidth=10,fill="yellow", color="black") +
  scale_y_log10() +
  theme_bw()
```

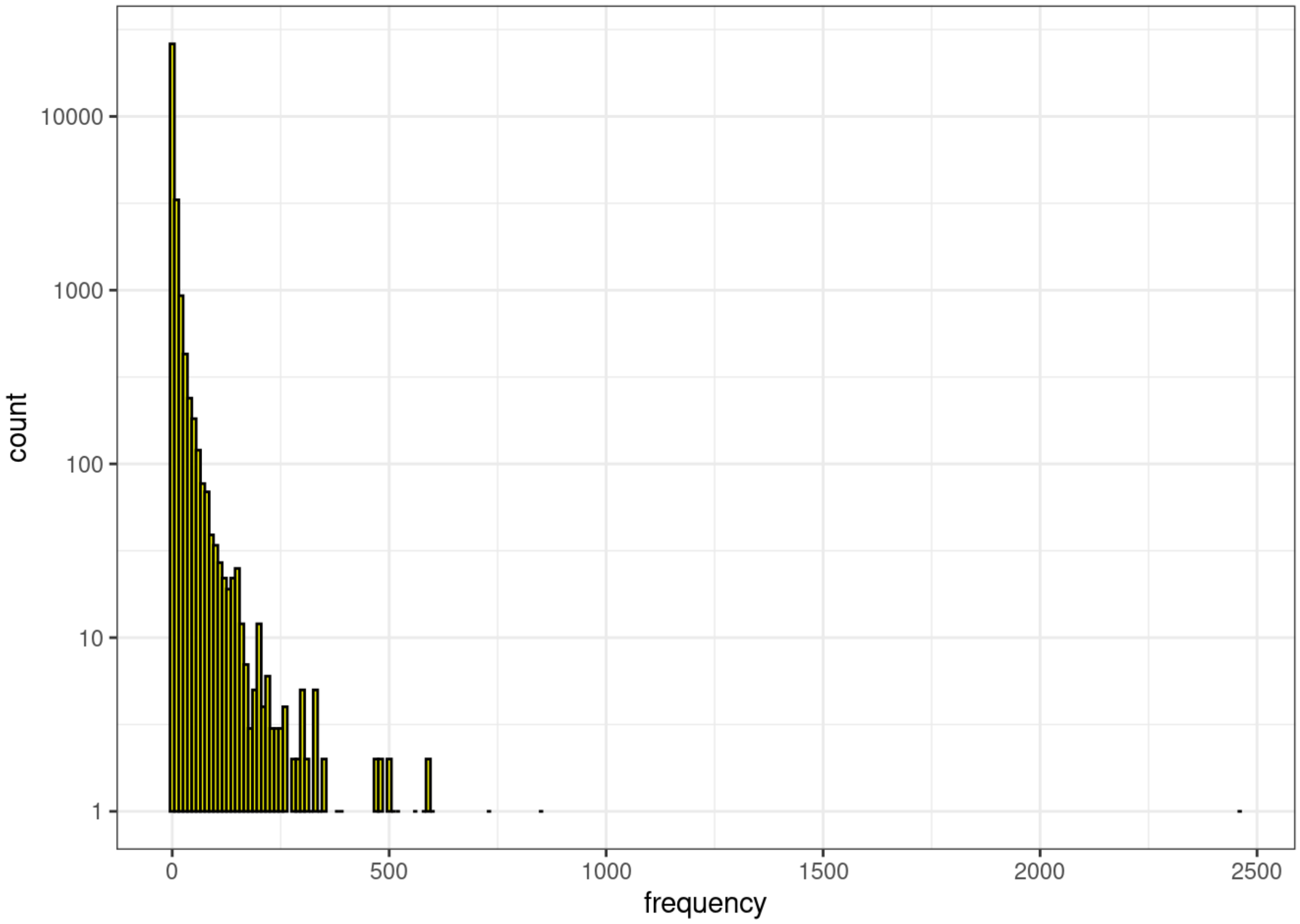
```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 72 rows containing missing values (`geom_bar()`).
```



```
token_news %>%
  tokens_remove(pattern = stopwords('en')) %>%
  dfm(tolower=T) %>%
  textstat_frequency() %>%
  ggplot(aes(x=frequency)) +
  geom_histogram(binwidth=10,fill="yellow", color="black") +
  scale_y_log10() +
  theme_bw()
```

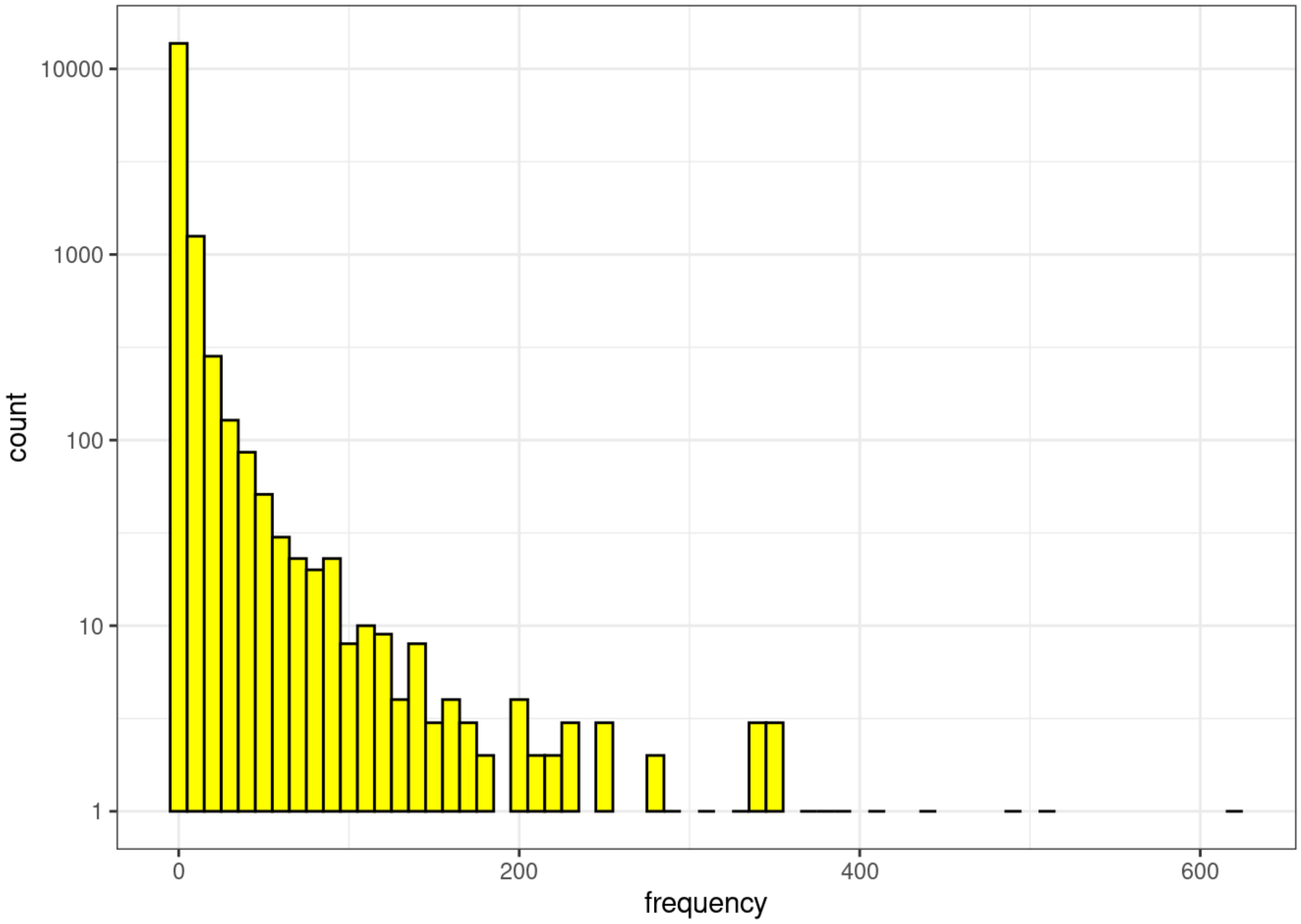
```
## Warning: Transformation introduced infinite values in continuous y-axis
## Warning: Removed 198 rows containing missing values (`geom_bar()`).
```



```
token_twitter %>%
  tokens_remove(pattern = stopwords('en')) %>%
  dfm(tolower=T) %>%
  textstat_frequency() %>%
  ggplot(aes(x=frequency)) +
  geom_histogram(binwidth=10,fill="yellow", color="black") +
  scale_y_log10() +
  theme_bw()
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

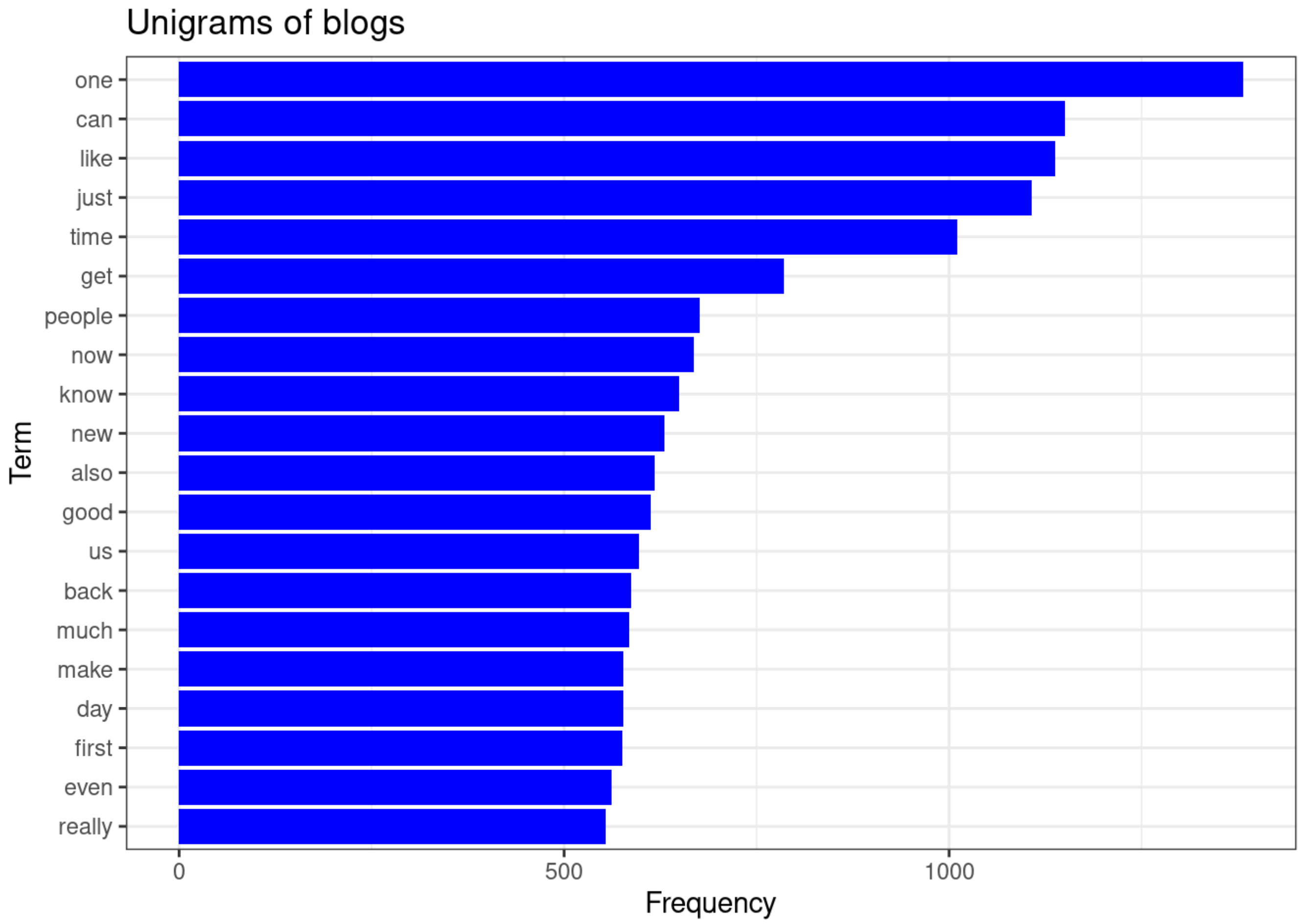
```
## Warning: Removed 25 rows containing missing values (`geom_bar()`).
```



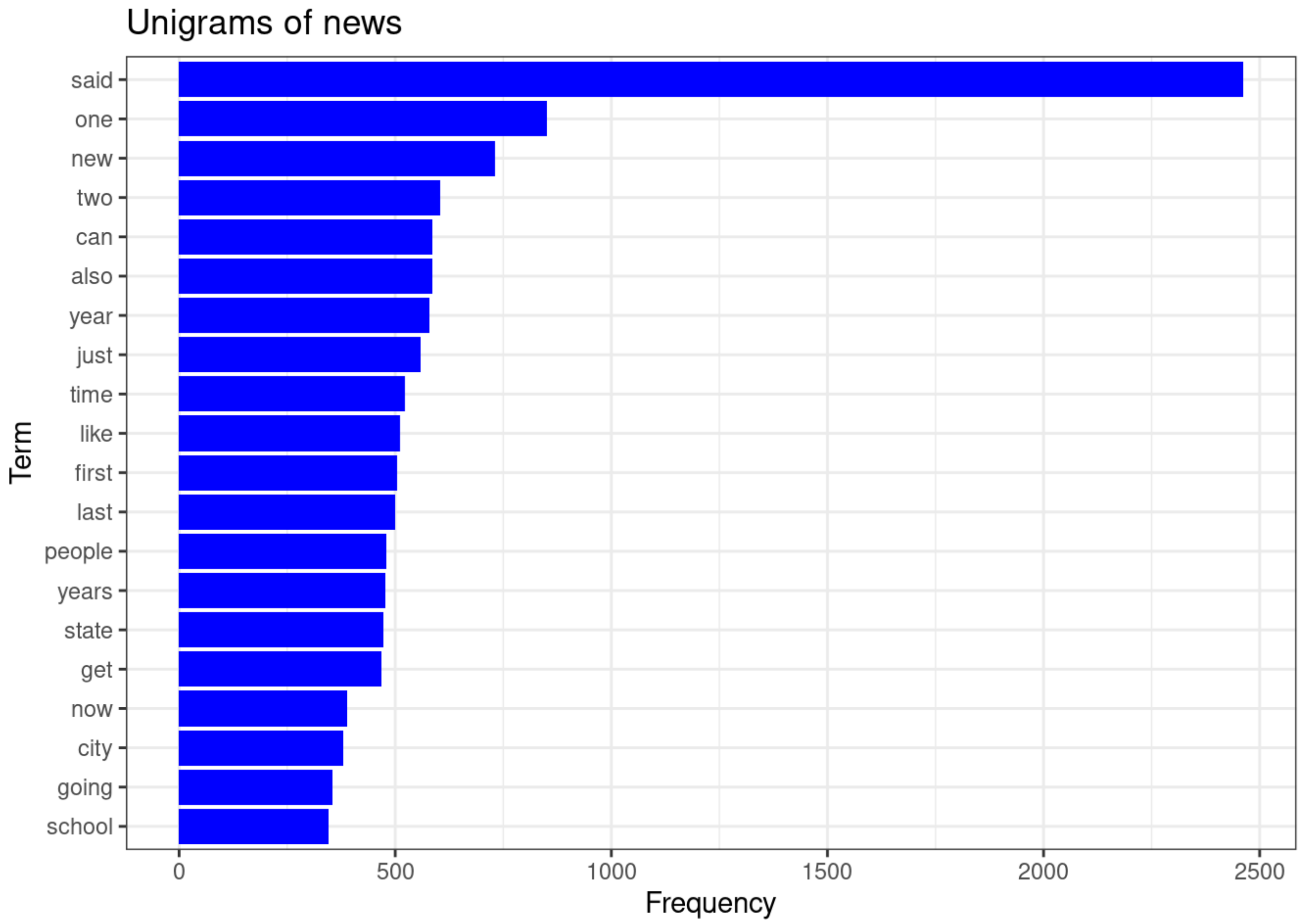
# Unigram frequency analysis of each corpus

First, I performed the unigram analysis of three corpus, by using “tokenizers” package. After that, I select top 20 most freuent words. The ranking of three corpus is quite different as you see.

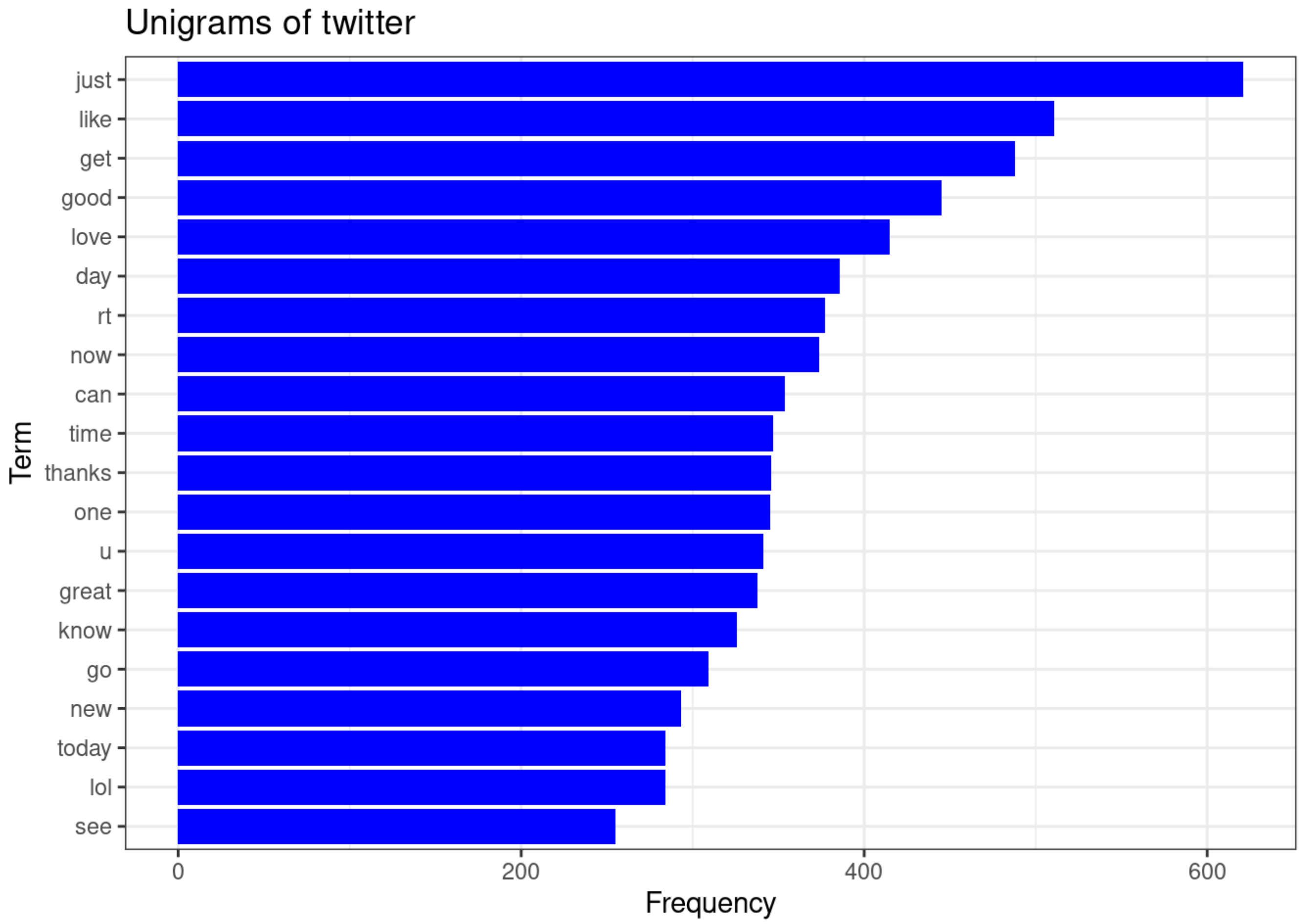
```
token_blogs %>%
  tokens_remove(pattern = stopwords('en')) %>%
  tokens_ngrams(n = 1) %>%
  dfm(tolower=T) %>%
  textstat_frequency(n=20) %>%
  arrange(desc(frequency)) %>%
  ggplot(aes(y=reorder(feature, frequency), x=frequency)) +
  geom_bar(stat="identity", fill="blue") +
  ggtitle("Unigrams of blogs") +
  ylab("Term") +
  xlab("Frequency") +
  theme_bw()
```



```
token_news %>%
  tokens_remove(pattern = stopwords('en')) %>%
  tokens_ngrams(n = 1) %>%
  dfm(tolower=T) %>%
  textstat_frequency(n=20) %>%
  arrange(desc(frequency)) %>%
  ggplot(aes(y=reorder(feature, frequency), x=frequency)) +
  geom_bar(stat="identity", fill="blue") +
  ggtitle("Unigrams of news") +
  ylab("Term") +
  xlab("Frequency") +
  theme_bw()
```



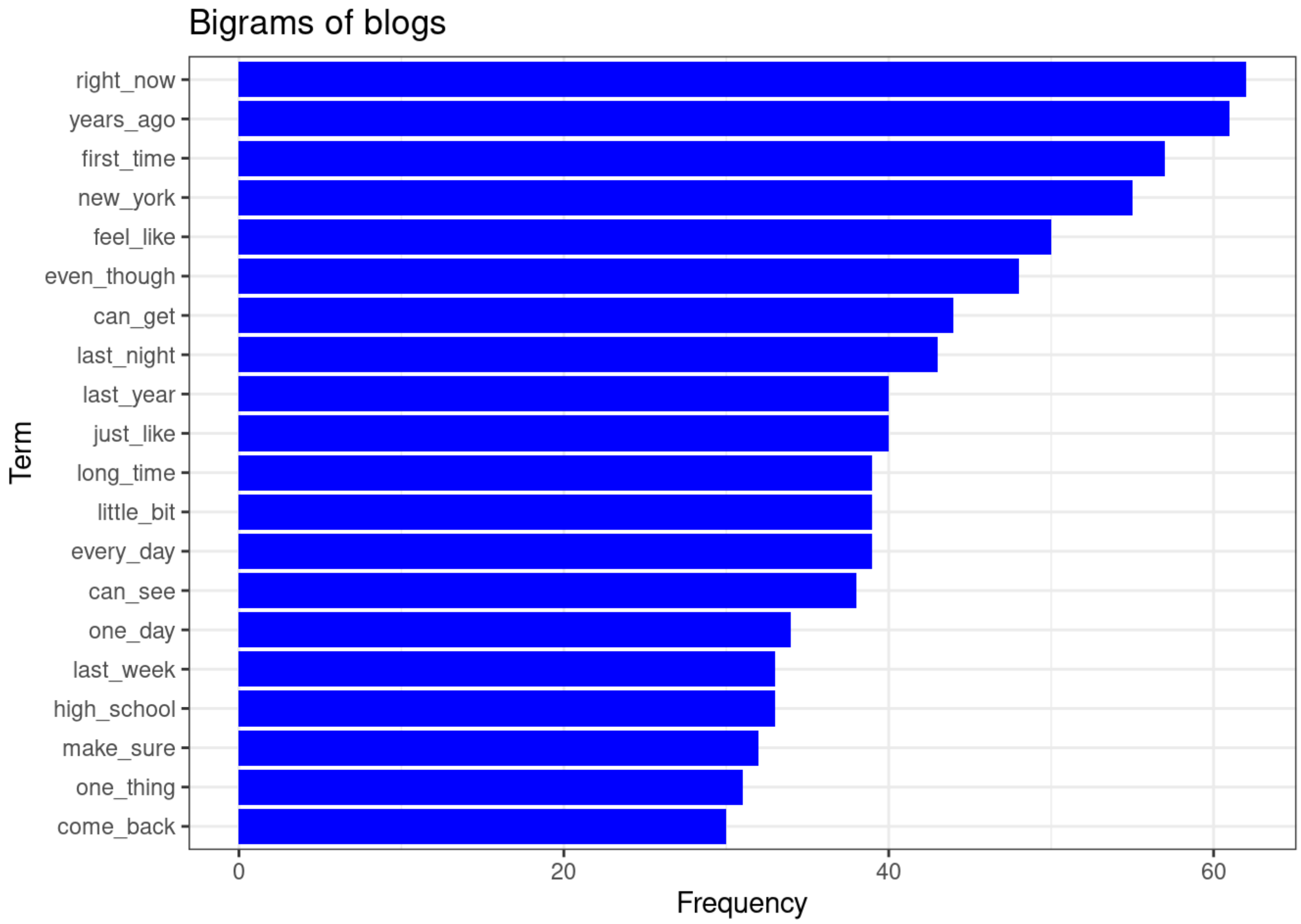
```
token_twitter %>%
  tokens_remove(pattern = stopwords('en')) %>%
  tokens_ngrams(n = 1) %>%
  dfm(tolower=T) %>%
  textstat_frequency(n=20) %>%
  arrange(desc(frequency)) %>%
  ggplot(aes(y=reorder(feature, frequency), x=frequency)) +
  geom_bar(stat="identity", fill="blue") +
  ggtitle("Unigrams of twitter") +
  ylab("Term") +
  xlab("Frequency") +
  theme_bw()
```



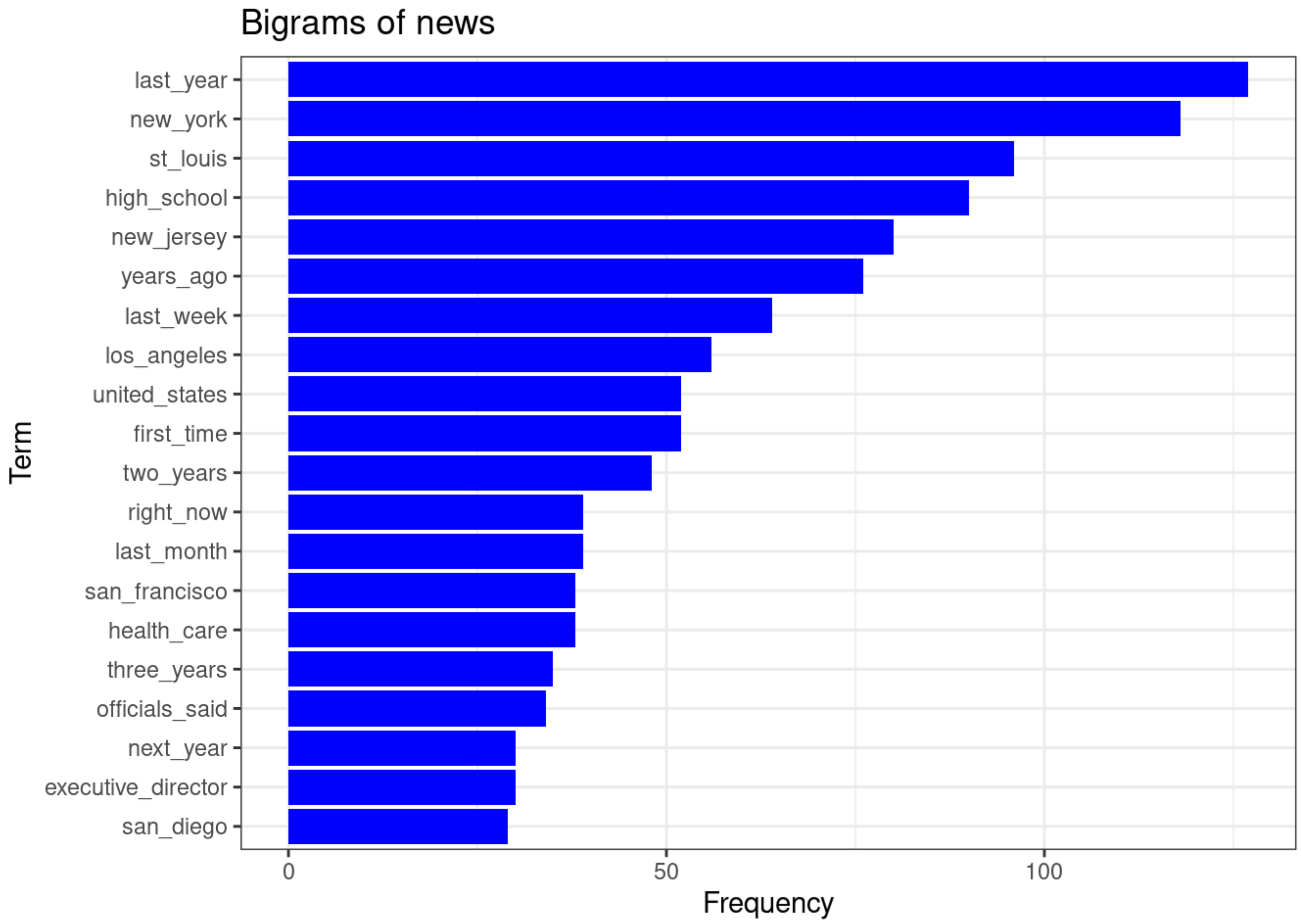
# Bigram frequency analysis of each corpus

Second, I performed the bigram analysis, by using “tokens_ngram(n=2)”. According to the result of the analysis, there are diference of two words among three corpus.

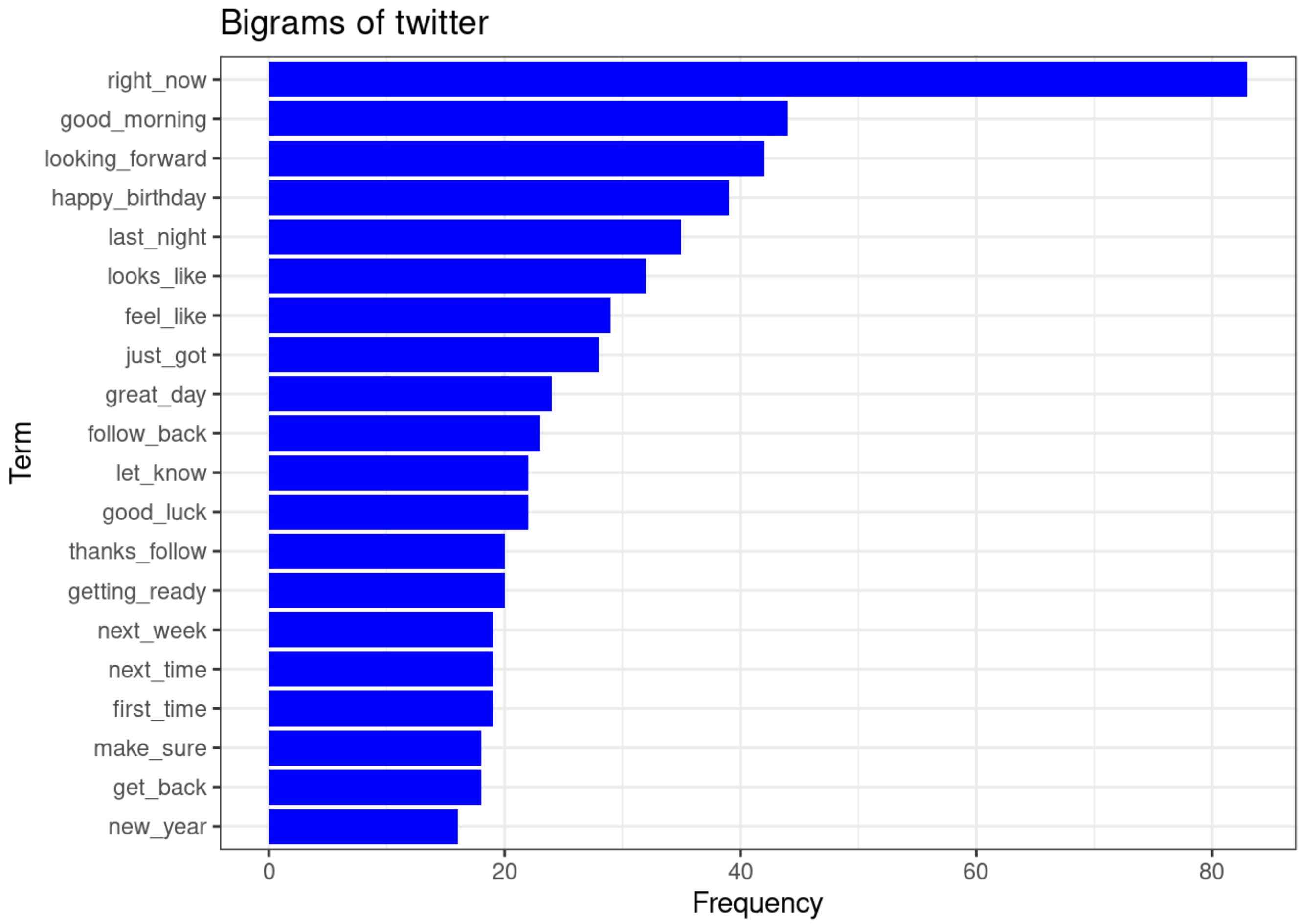
```
token_blogs %>%
  tokens_remove(pattern = stopwords('en')) %>%
  tokens_ngrams(n = 2) %>%
  dfm(tolower=T) %>%
  textstat_frequency(n=20) %>%
  arrange(desc(frequency)) %>%
  ggplot(aes(y=reorder(feature, frequency), x=frequency)) +
  geom_bar(stat="identity", fill="blue") +
  ggtitle("Bigrams of blogs") +
  ylab("Term") +
  xlab("Frequency") +
  theme_bw()
```



```
token_news %>%
  tokens_remove(pattern = stopwords('en')) %>%
  tokens_ngrams(n = 2) %>%
  dfm(tolower=T) %>%
  textstat_frequency(n=20) %>%
  arrange(desc(frequency)) %>%
  ggplot(aes(y=reorder(feature, frequency), x=frequency)) +
  geom_bar(stat="identity", fill="blue") +
  ggtitle("Bigrams of news") +
  ylab("Term") +
  xlab("Frequency") +
  theme_bw()
```



```
token_twitter %>%
  tokens_remove(pattern = stopwords('en')) %>%
  tokens_ngrams(n = 2) %>%
  dfm(tolower=T) %>%
  textstat_frequency(n=20) %>%
  arrange(desc(frequency)) %>%
  ggplot(aes(y=reorder(feature, frequency), x=frequency)) +
  geom_bar(stat="identity", fill="blue") +
  ggtitle("Bigrams of twitter") +
  ylab("Term") +
  xlab("Frequency") +
  theme_bw()
```


Bigrams of twitter
Term
right_now
good_morning
looking_forward
happy_birthday
last_night
looks_like
feel_like
just_got
great_day
follow_back
let_know
good_luck
thanks_follow
getting_ready
next_week
next_time
first_time
make_sure
get_back
new_year
0
20
40
60
80
Frequency