Chapter7. Case study: comparing Twitter archives

Geonwoo Ban

이번 장에서 다룰 데이터는 트위터를 통해 온라인으로 공유되는 텍스트입니다. 이 책에 사용된 lexicon 중 일부는 트윗과 함께 사용해 트윗의 유효성을 검증하도록 설계되었습니다.

이 책의 저자인 Julia (https://twitter.com/juliasilge)와 David (https://twitter.com/juliasilge)는 모두 트위터를 하며, 일반적인 사용자이므로 이번 사례 연구에서는 두 저자의 전체 트위터 아카이브를 비교해보고자합니다.

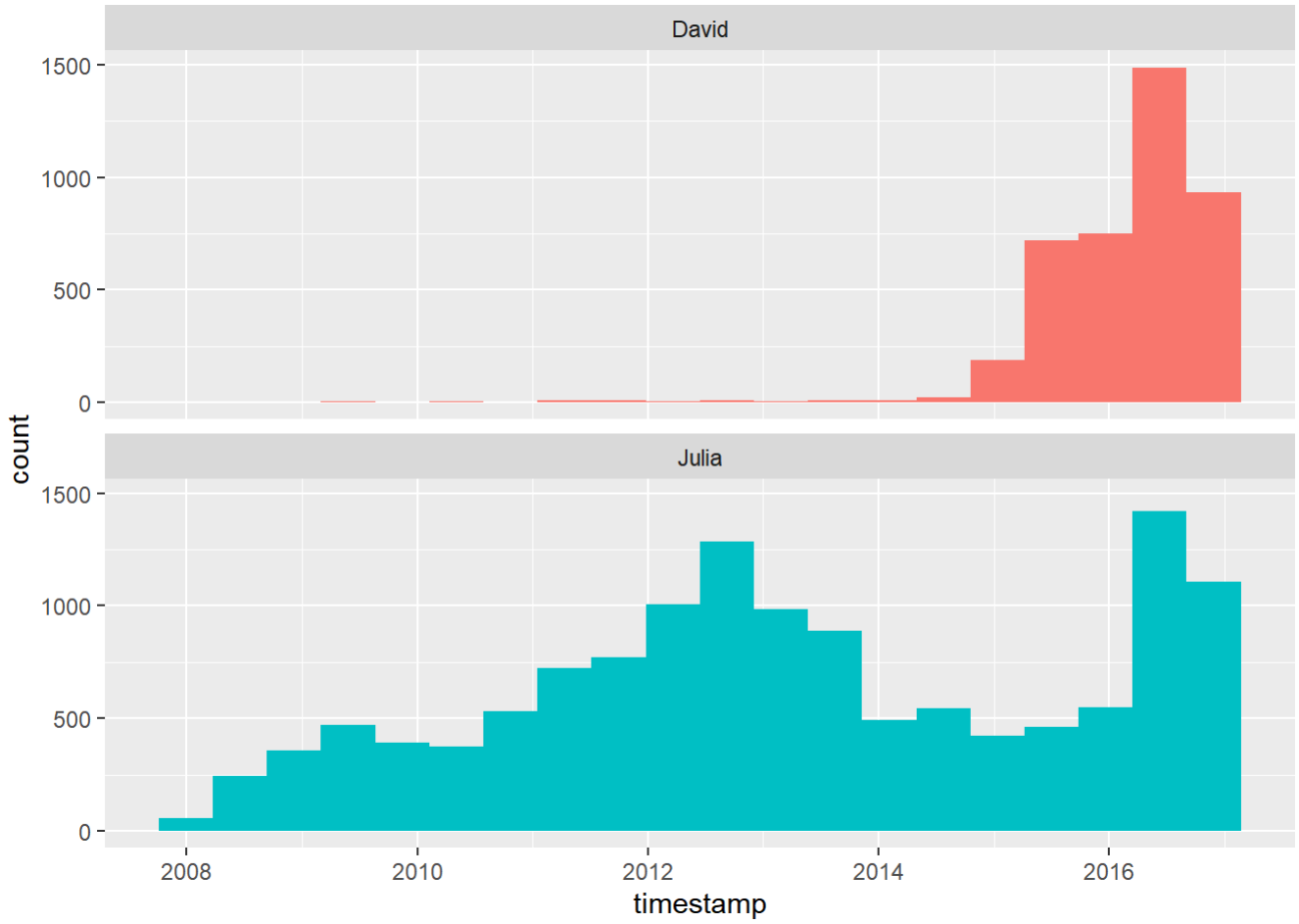
7.1 Getting the data and distribution of tweets

Julia와 David의 트위터 아카이브를 다운하여 lubridate 패키지를 사용하여 문자열 timestamp를 date 형식으로변환하고 전반적으로 트윗 패턴 전체를 살펴볼 것입니다.

```
library(lubridate)
library(ggplot2)
library(dplyr)
library(readr)

tweets_julia <- read_csv("data/tweets_julia.csv")
tweets_dave <- read_csv("data/tweets_dave.csv")
tweets <- bind_rows(tweets_julia %>%
                      mutate(person = "Julia"),
                    tweets_dave %>%
                      mutate(person = "David")) %>%
  mutate(timestamp = ymd_hms(timestamp))

tweets %>% select(text) %>% head(5)
```

```
## # A tibble: 5 x 1
##   text
##   <chr>
## 1 @Jowanza Hooooooooo boy, that's tough in so many ways.
## 2 I love many things about living here, but the relative lack of diversity is n~
## 3 BREAKING NEWS, EVERYBODY!!! Populations with more white people experience les~
## 4 So much side eye to this nonsensical article: Salt Lake tech workers experien~
## 5 @astropixie She was my external committee member.
```

```
ggplot(tweets, aes(x = timestamp, fill = person)) +
  geom_histogram(position = "identity", bins = 20, show.legend = FALSE) +
  facet_wrap(~person, ncol = 1)
```



최근시점에 대해서는 David와 Julia가 유사한 속도로 트위터에 글을 올리고 있지만, 전체적인 양을 보면 Julia가 David보다 매우 많이 트윗을 하였습니다.

7.2 Word frequencies

`unnest_tokens()` 를 사용해 트위터의 모든 단어에 대해 tidy data frame을 만들고 stop word를 삭제해 보았습니다.

먼저 이 데이터셋에서 리트윗을 제거해 우리가 직접 작성한 트윗만 남게 할 생각이며 그런 다음 `mutate()` 줄이 링크를 제거하고 & 등과 같이 원하지 않는 일부 문자를 지웁니다.

일부 문자를 처리하기 위해서 `anti_join()` 을 사용하여 불용어를 제거하기보다는 `str_detect()` 를 사용하여 `filter()` 로 제거할 수 있습니다.

```
library(tidytext)
library(stringr)

remove_reg <- "&amp;|&lt;|&gt;" # Remove "&"
tidy_tweets <- tweets %>%
  filter(!str_detect(text, "^RT")) %>% # Remove retweets
  mutate(text = str_remove_all(text, remove_reg)) %>%
  unnest_tokens(word, text, token = "tweets") %>%
  filter(!word %in% stop_words$word, # Remove stop words
         !word %in% str_remove_all(stop_words$word, "'"), # Remove "'"
         str_detect(word, "[a-z]")) # Filter word

tidy_tweets %>% select(timestamp,person, word) %>% head(15)
```

```
## # A tibble: 15 x 3
##    timestamp           person word
##    <dttm>              <chr>  <chr>
##  1 2017-01-01 21:48:41 Julia  @jowanza
##  2 2017-01-01 21:48:41 Julia  hooooooooo
##  3 2017-01-01 21:48:41 Julia  boy
##  4 2017-01-01 21:48:41 Julia  tough
##  5 2017-01-01 21:16:16 Julia  love
##  6 2017-01-01 21:16:16 Julia  living
##  7 2017-01-01 21:16:16 Julia  relative
##  8 2017-01-01 21:16:16 Julia  lack
##  9 2017-01-01 21:16:16 Julia  diversity
## 10 2017-01-01 21:13:45 Julia  breaking
## 11 2017-01-01 21:13:45 Julia  news
## 12 2017-01-01 21:13:45 Julia  populations
## 13 2017-01-01 21:13:45 Julia  white
## 14 2017-01-01 21:13:45 Julia  people
## 15 2017-01-01 21:13:45 Julia  experience
```

이제 각 단어를 분리시킨 구조로 만들었으면 각 단어에 대한 빈도를 계산할 수 있습니다. 먼저 사람별로 그룹화하고 각 사람이 각 단어를 몇번이나 사용했는지 계산합니다. 그런 다음 `left_join()` 을 사용해 각 사람이 사용하는 총 단어 개수 열을 추가합니다. 마지막으로 각 사람과 각 단어의 빈도수를 계산합니다.

```
frequency <- tidy_tweets %>%
  group_by(person) %>%
  count(word, sort = TRUE) %>%
  left_join(tidy_tweets %>%
              group_by(person) %>%
              summarise(total = n())) %>%
  mutate(freq = n/total)

frequency
```

```
## # A tibble: 24,067 x 5
## # Groups:   person [2]
##    person word               n total    freq
##    <chr>  <chr>          <int> <int>   <dbl>
##  1 Julia  @selkie1970      570 74152 0.00769
##  2 Julia  time             557 74152 0.00751
##  3 Julia  @skedman         531 74152 0.00716
##  4 Julia  day              437 74152 0.00589
##  5 Julia  baby             392 74152 0.00529
##  6 David  @hadleywickham   308 20699 0.0149
##  7 Julia  love             302 74152 0.00407
##  8 Julia  @haleynburke     298 74152 0.00402
##  9 Julia  house            283 74152 0.00382
## 10 Julia  morning          278 74152 0.00375
## # ... with 24,057 more rows
```

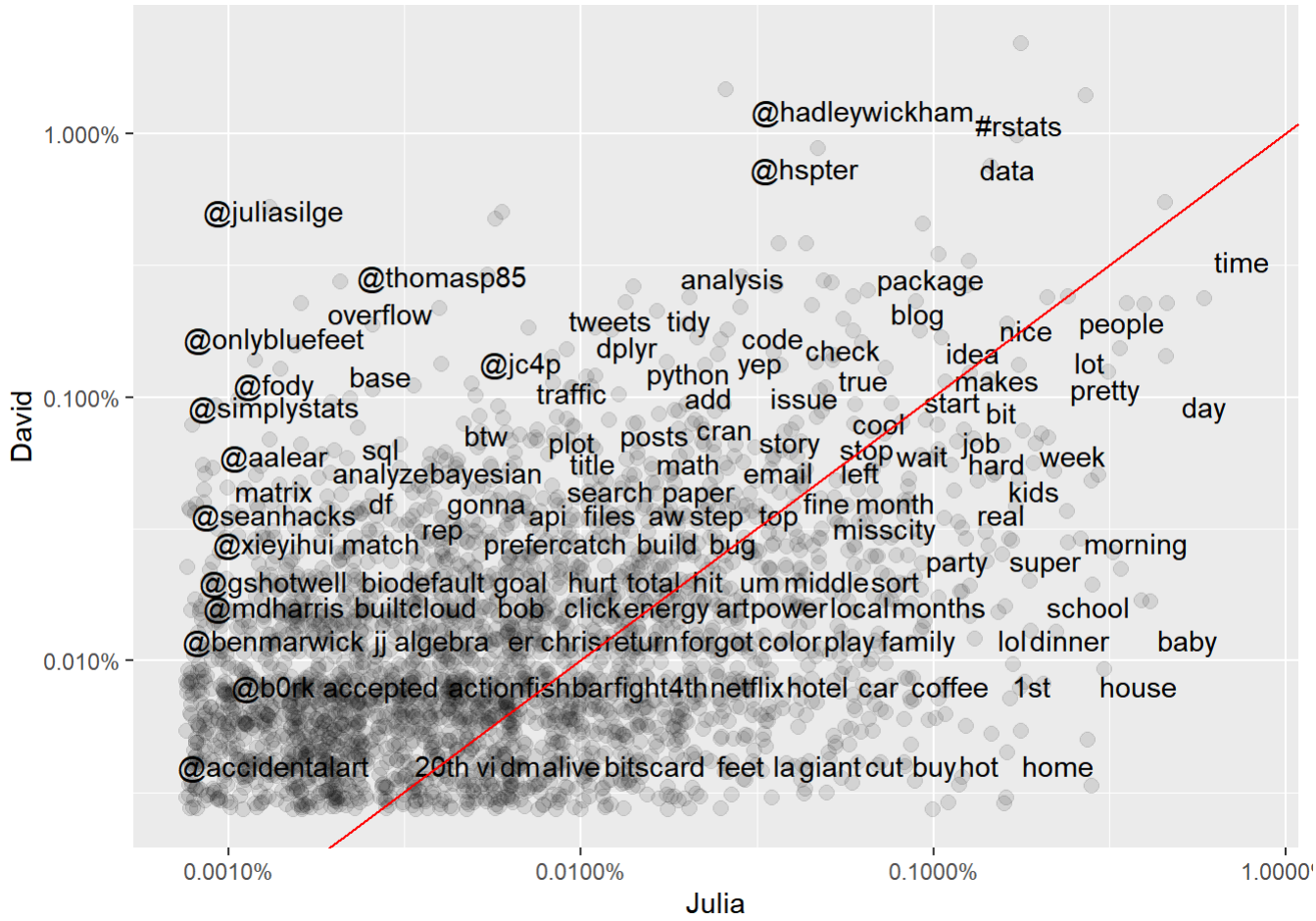
이제 두 사람간의 사용하는 단어들의 공통점을 찾아보기 위해 각 사람별 단어의 비율을 시각화해보고자합니다. 하지만 그러기 위해서는 단어에 대해 개인의 비율이 열로 표현이 되어야 하기 때문에 `spread()` 를 사용하여 바꿔줘야 합니다.

```
library(tidyr)

frequency <- frequency %>%
  select(person, word, freq) %>%
  pivot_wider(names_from = person, values_from = freq) %>%
  arrange(Julia, David)

frequency
```

```
## # A tibble: 21,071 x 3
##    word               Julia     David
##    <chr>              <dbl>     <dbl>
##  1 @accidentalart 0.0000135 0.0000483
##  2 @alicedata     0.0000135 0.0000483
##  3 @alistaire     0.0000135 0.0000483
##  4 @corynissen    0.0000135 0.0000483
##  5 @jennybryans   0.0000135 0.0000483
##  6 @jsvine        0.0000135 0.0000483
##  7 @lewislab      0.0000135 0.0000483
##  8 @lizasperling  0.0000135 0.0000483
##  9 @ognyanova     0.0000135 0.0000483
## 10 @rbloggers     0.0000135 0.0000483
## # ... with 21,061 more rows
```

이제 그래프를 그릴 준비가 되었다. `geom_jitter()` 를 사용하여 그림을 그리고 `check_overlap = TRUE` 로 설정하여 일부 텍스트 레이블이 겹쳐서 표시되지 않도록 하자.

```
library(scales)

ggplot(frequency, aes(Julia, David)) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.25, height = 0.25) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = percent_format()) +
  scale_y_log10(labels = percent_format()) +
  geom_abline(color = "red")
```



선을 기준으로 선 근처의 단어들은 두 사람에 대해 서로 동일한 빈도로 사용한 단어이고, 선 위의 단어들은 David가 Julia보다 더 많이 사용한 단어이며 선 아래는 Julia가 David보다 더 많이 사용한 단어들입니다.

동시에 사용하는 단어들 보다도 눈에 띄는 것은 David의 경우엔 누군가를 태그하는 의미의 @를 Julia보다 더 많이 사용하였고 반면에 Julia는 “home”, “House”, “freidens”, “lol”등의 일상과 관련된 단어들을 보다 더 많이 사용함을 알 수 있습니다.

이는 두 사람이 트위터를 사용하는 목적이 다르기 때문에 이와 같이 나왔다고 볼 수 있습니다. David는 자신의 트위터 계정을 거의 전문적인 목적으로만 사용했지만 Julia의 경우엔 David보다 사적인 용도로 더 많이 사용했다고 볼 수 있습니다.

7.3 Comparing word usage

앞선 그림에서는 두 사람의 단어 빈도를 비교하였습니다. 이번에는 로그 오즈비를 사용하여 각자의 계정에서 어느 단어가 더 나올지 또는 덜 나올지를 알아보겠습니다.

사용할 데이터는 2016년에 보낸 트윗으로만 제한하자. 2016년도에는 David는 활발히 활동했고, Julia는 데이터 과학 분야로 전환하여 경력을 쌓았습니다.

```
tidy_tweets <- tidy_tweets %>%
  filter(timestamp >= as.Date("2016-01-01"),
         timestamp < as.Date("2017-01-01"))
```

다음으로 `str_detect()` 를 사용하여 word열에서 트위터 사용자 이름을 제거하고자합니다. 이를 하지 않으면 결과가 오직 두 사람 개인이 아는 사람에 의해서만 결과가 편향되고, 그렇지 않은 사람들은 결과에 영향을 끼치지 못하게 되기 때문입니다. 이름을 제거한 후에는 각 사람이 각 단어를 몇 번이나 사용했는지 계산하고 10회 이상 사용한 단어만 유지합니다.

$$logoddsratio = ln(\frac{[\frac{n+1}{total+1}]_{David}}{[\frac{n+1}{total+1}]_{Julia}})$$

여기서 n은 각 사람이 해당 단어를 사용한 횟수이고, total은 각 사람의 총 단어를 나타낸다. 로그오즈비가 0보다 크다면 David가 더 많이 사용한 단어이고, 0보다 작다면 Julia가 더 많이 사용한 단어입니다.

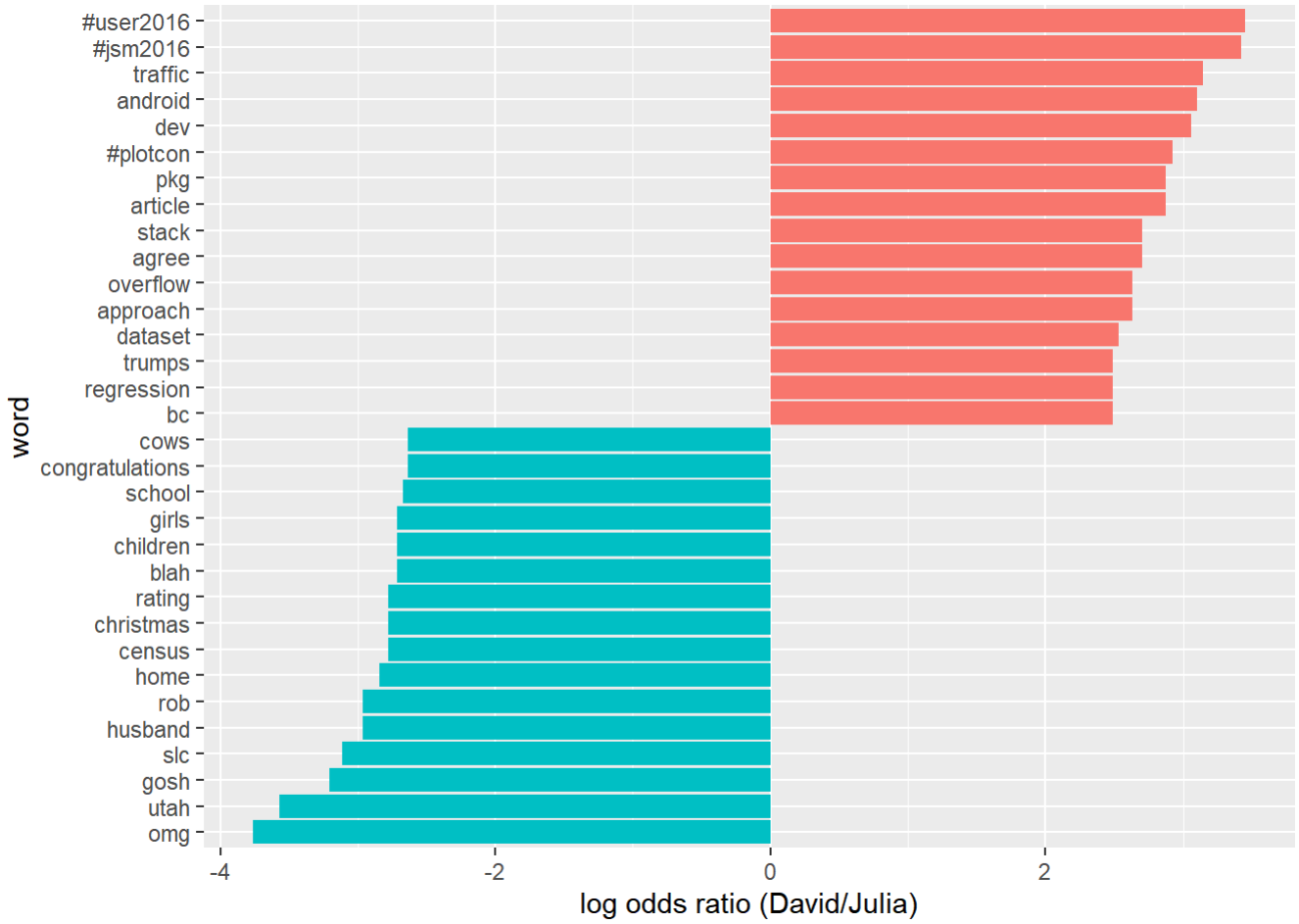
```
word_ratios <- tidy_tweets %>%
  filter(!str_detect(word, "^@")) %>% # Remove People id
  count(word, person) %>%
  group_by(word) %>%
  filter(sum(n) >= 10) %>%
  ungroup() %>%
  pivot_wider(names_from = person, values_from = n, values_fill = 0) %>%
  mutate_if(is.numeric, list(~(. + 1) / (sum(.) + 1))) %>% # Odds
  mutate(logratio = log(David / Julia)) %>% # log odds ratio
  arrange(desc(logratio))

word_ratios %>%
  arrange(abs(logratio))
```

```
## # A tibble: 351 x 4
##    word      David   Julia logratio
##    <chr>     <dbl>   <dbl>    <dbl>
##  1 words   0.00377 0.00378 -0.00334
##  2 science 0.00653 0.00648  0.00771
##  3 idea    0.00577 0.00594 -0.0279
##  4 email   0.00251 0.00243  0.0330
##  5 file    0.00251 0.00243  0.0330
##  6 purrr   0.00251 0.00243  0.0330
##  7 test    0.00226 0.00216  0.0454
##  8 account 0.00201 0.00189  0.0612
##  9 api     0.00201 0.00189  0.0612
## 10 sad     0.00201 0.00189  0.0612
## # ... with 341 more rows
```

log ratio가 0에 가까운 단어들을 뽑아 2016년에 David와 Julia의 계정에서 똑같이 나올 가능성이 있는 단어는 무엇인가 확인해볼 수 도 있고, 이를 확인해본 결과 science, idea, file, api에 관해서는 서로 같이 트윗할 가능성이 있음을 볼 수 있습니다.

그렇다면 어떤 단어가 각 계정별로 독특하게 나타나는지를 확인해볼 수 도 있습니다.

```
word_ratios %>%
  group_by(logratio < 0) %>%
  slice_max(abs(logratio), n = 15) %>%
  ungroup() %>%
  mutate(word = reorder(word, logratio)) %>%
  ggplot(aes(word, logratio, fill = logratio < 0)) +
  geom_col(show.legend = FALSE) +
  coord_flip() +
  ylab("log odds ratio (David/Julia)") +
  scale_fill_discrete(name = "", labels = c("David", "Julia"))
```



위 그림을 보면 David의 경우엔 자신이 참석한 특정 회의나 Stack Overflow에 대해 트윗을 했고, Julia의 경우엔 Utah, Census data, 그리고 그녀의 가족들에 대해 트윗을 했다는 점을 알 수 있습니다.

7.4 Changes in word use

이번에는 트위터 피드에서 어떤 단어의 시간별 트윗 비율이 커지거나 작아지는지, 변화율이 큰 단어들을 확인해보고자 합니다. 이를 보기 위해서 각 트윗이 게시된 시간 단위를 정의하는 새로운 시간 변수를 데이터 프레임에서 정의할 것 입니다. lubridate 패키지의 `floor_date()` 를 사용하면 원하는 목적에 맞춰 분석을 진행할 수 있습니다. 단위를 한달로 나누어 각 나누어진 time bins에서 단어의 빈도를 확인해볼 수 있습니다.

```
words_by_time <- tidy_tweets %>%
  filter(!str_detect(word, "^@")) %>% # Remove name
  mutate(time_floor = floor_date(timestamp, unit = "1 month")) %>% # Seperate by 1 month
  count(time_floor, person, word) %>%
  group_by(person, time_floor) %>%
  mutate(time_total = sum(n)) %>% # how many words that person used during that time bin
  group_by(person, word) %>%
  mutate(word_total = sum(n)) %>% # how many times that person used that word over the whole year
  ungroup() %>%
  rename(count = n) %>%
  filter(word_total > 30)

words_by_time
```

```
## # A tibble: 326 x 6
##    time_floor          person word    count time_total word_total
##    <dttm>              <chr>  <chr>   <int>      <int>      <int>
##  1 2016-01-01 00:00:00 David  #rstats     2        315        205
##  2 2016-01-01 00:00:00 David  broom       2        315         34
##  3 2016-01-01 00:00:00 David  data        2        315        148
##  4 2016-01-01 00:00:00 David  ggplot2     1        315         37
##  5 2016-01-01 00:00:00 David  time        2        315         56
##  6 2016-01-01 00:00:00 David  tweets      1        315         46
##  7 2016-01-01 00:00:00 Julia  #rstats    10        437        116
##  8 2016-01-01 00:00:00 Julia  blog        2        437         33
##  9 2016-01-01 00:00:00 Julia  data        5        437        105
## 10 2016-01-01 00:00:00 Julia  day         1        437         43
## # ... with 316 more rows
```

이 데이터 프레임의 각 행은 주어진 Time bin(1 month)에서 한 단어를 사용하는 한 개인에 해당합니다. `count` 변수는 해당 time bin에서 해당 단어를 사용한 횟수를 나타내고, `time_total` 은 해당 time bin에서 사용한 단어의 개수를 의미하며, `word_total` 은 해당 단어를 1년 내내 사용한 횟수를 나타내고 있습니다.

이제 이 데이터셋을 tidyr의 `nest()` 를 사용하여 각 단어에 대한 소형 데이터 프레임들이 들어있는 1개 list 열로 구성된 데이터 프레임을 만들 수 있습니다.

```
nested_data <- words_by_time %>%
  nest(-word, -person)
```

```
## Warning: All elements of `...` must be named.
## Did you want `data = c(time_floor, count, time_total, word_total)`?
```

```
nested_data
```

```
## # A tibble: 32 x 3
##    person word    data
##    <chr>  <chr>   <list>
##  1 David  #rstats <tibble [12 x 4]>
##  2 David  broom   <tibble [10 x 4]>
##  3 David  data    <tibble [12 x 4]>
##  4 David  ggplot2 <tibble [10 x 4]>
##  5 David  time    <tibble [12 x 4]>
##  6 David  tweets  <tibble [8 x 4]>
##  7 Julia  #rstats <tibble [12 x 4]>
##  8 Julia  blog    <tibble [10 x 4]>
##  9 Julia  data    <tibble [12 x 4]>
## 10 Julia  day     <tibble [12 x 4]>
## # ... with 22 more rows
```

```
nested_data$data[[1]]
```

```
## # A tibble: 12 x 4
##    time_floor          count time_total word_total
##    <dttm>              <int>      <int>      <int>
##  1 2016-01-01 00:00:00     2        315        205
##  2 2016-02-01 00:00:00    17       1037        205
##  3 2016-03-01 00:00:00    21       1058        205
##  4 2016-04-01 00:00:00    24       1110        205
##  5 2016-05-01 00:00:00    15        601        205
##  6 2016-06-01 00:00:00    24        820        205
##  7 2016-07-01 00:00:00    16        738        205
##  8 2016-08-01 00:00:00    26       1956        205
##  9 2016-09-01 00:00:00     9       1265        205
## 10 2016-10-01 00:00:00    18        846        205
## 11 2016-11-01 00:00:00    17       1282        205
## 12 2016-12-01 00:00:00    16        691        205
```

이 데이터 프레임은 각 사람-단어 조합에 대해 하나의 행을 가지며, data 변수는 데이터 프레임이 포함된 리스트 열인데, 각 개인과 단어의 조합마다 한 개씩 있습니다.

purrr 패키지의 `map()` 을 사용하여 큰 데이터 프레임 내의 작은 데이터 프레임에 모델링 절차를 적용해 볼 수 있습니다.

특정 time bin에서 특정 단어가 언급되었는가? “예”인가 아니면 “아니오”인가? 단어에 대한 개수가 시간에 따라 어떻게 달라지는가? 에 대한 질문들에 답하는 모델링 절차를 생각해 볼 수 있습니다.

여기서는 개수를 세는 데이터이므로 모델링을 위해 glm(family=“binomial”)을 사용하여 모델링을 하였습니다.(?)

```
library(purrr)

nested_models <- nested_data %>%
  mutate(models = map(data, ~ glm(cbind(count, time_total-count) ~ time_floor, .,
                                  family = "binomial")))

nested_models
```

```
## # A tibble: 32 x 4
##    person word    data              models
##    <chr>  <chr>   <list>            <list>
##  1 David  #rstats <tibble [12 x 4]> <glm>
##  2 David  broom   <tibble [10 x 4]> <glm>
##  3 David  data    <tibble [12 x 4]> <glm>
##  4 David  ggplot2 <tibble [10 x 4]> <glm>
##  5 David  time    <tibble [12 x 4]> <glm>
##  6 David  tweets  <tibble [8 x 4]>  <glm>
##  7 Julia  #rstats <tibble [12 x 4]> <glm>
##  8 Julia  blog    <tibble [10 x 4]> <glm>
##  9 Julia  data    <tibble [12 x 4]> <glm>
## 10 Julia  day     <tibble [12 x 4]> <glm>
## # ... with 22 more rows
```

```
nested_models$data[[1]]
```

```
## # A tibble: 12 x 4
##    time_floor          count time_total word_total
##    <dttm>              <int>      <int>      <int>
##  1 2016-01-01 00:00:00     2        315        205
##  2 2016-02-01 00:00:00    17       1037        205
##  3 2016-03-01 00:00:00    21       1058        205
##  4 2016-04-01 00:00:00    24       1110        205
##  5 2016-05-01 00:00:00    15        601        205
##  6 2016-06-01 00:00:00    24        820        205
##  7 2016-07-01 00:00:00    16        738        205
##  8 2016-08-01 00:00:00    26       1956        205
##  9 2016-09-01 00:00:00     9       1265        205
## 10 2016-10-01 00:00:00    18        846        205
## 11 2016-11-01 00:00:00    17       1282        205
## 12 2016-12-01 00:00:00    16        691        205
```

```
nested_models$models[[1]]
```

```
##
## Call:  glm(formula = cbind(count, time_total - count) ~ time_floor,
##     family = "binomial", data = .)
##
## Coefficients:
## (Intercept)   time_floor
##   7.592e+00   -7.922e-09
##
## Degrees of Freedom: 11 Total (i.e. Null);  10 Residual
## Null Deviance:       28.04
## Residual Deviance: 27.14     AIC: 85.53
```

```
predict(nested_models$models[[1]], type="response")
```

```
##         1         2         3         4         5         6         7
## 0.01969779 0.01929223 0.01892025 0.01853040 0.01816064 0.01778615 0.01743097
##         8         9        10        11        12
## 0.01707127 0.01671887 0.01638465 0.01604619 0.01572520
```

이제 모델링 결과를 담는 새로운 열이 생겼다는 점에 주목해볼 수 있습니다. 이 열 또한 리스트 형식으로 된 열이며 glm object를 포함합니다.

다음 단계는 broom 패키지의 `map()` 과 `tidy()` 를 사용해 각 모델의 추정된 기울기를 유도함으로써 중요한 모델을 찾을 수 있습니다.

다양한 추정된 기울기들에 대해 다중 비교를 하기 위해 p값을 adjusted p-value로 바꾸어 계산해야 합니다.

```
library(broom)

slopes <- nested_models %>%
  mutate(models = map(models, tidy)) %>%
  unnest(cols = c(models)) %>%
  filter(term == "time_floor") %>%
  mutate(adjusted.p.value = p.adjust(p.value))

slopes
```

```
## # A tibble: 32 x 9
##    person word    data          term     estimate   std.error statistic  p.value
##    <chr>  <chr>   <list>        <chr>       <dbl>       <dbl>     <dbl>    <dbl>
##  1 David  #rstats <tibble [12 ~ time_fl~ -7.92e-9     8.33e-9    -0.951  3.42e-1
##  2 David  broom   <tibble [10 ~ time_fl~ -2.12e-8     2.00e-8    -1.06   2.90e-1
##  3 David  data    <tibble [12 ~ time_fl~  2.17e-8     1.01e-8     2.16   3.10e-2
##  4 David  ggplot2 <tibble [10 ~ time_fl~ -8.10e-8     2.00e-8    -4.06   4.90e-5
##  5 David  time    <tibble [12 ~ time_fl~ -5.94e-9     1.58e-8    -0.375  7.08e-1
##  6 David  tweets  <tibble [8 x~ time_fl~  1.16e-8     2.00e-8     0.581  5.61e-1
##  7 Julia  #rstats <tibble [12 ~ time_fl~ -4.55e-8     1.12e-8    -4.08   4.41e-5
##  8 Julia  blog    <tibble [10 ~ time_fl~ -2.17e-8     2.25e-8    -0.967  3.33e-1
##  9 Julia  data    <tibble [12 ~ time_fl~  1.50e-8     1.17e-8     1.29   1.99e-1
## 10 Julia  day     <tibble [12 ~ time_fl~ -3.12e-8     1.80e-8    -1.74   8.27e-2
## # ... with 22 more rows, and 1 more variable: adjusted.p.value <dbl>
```

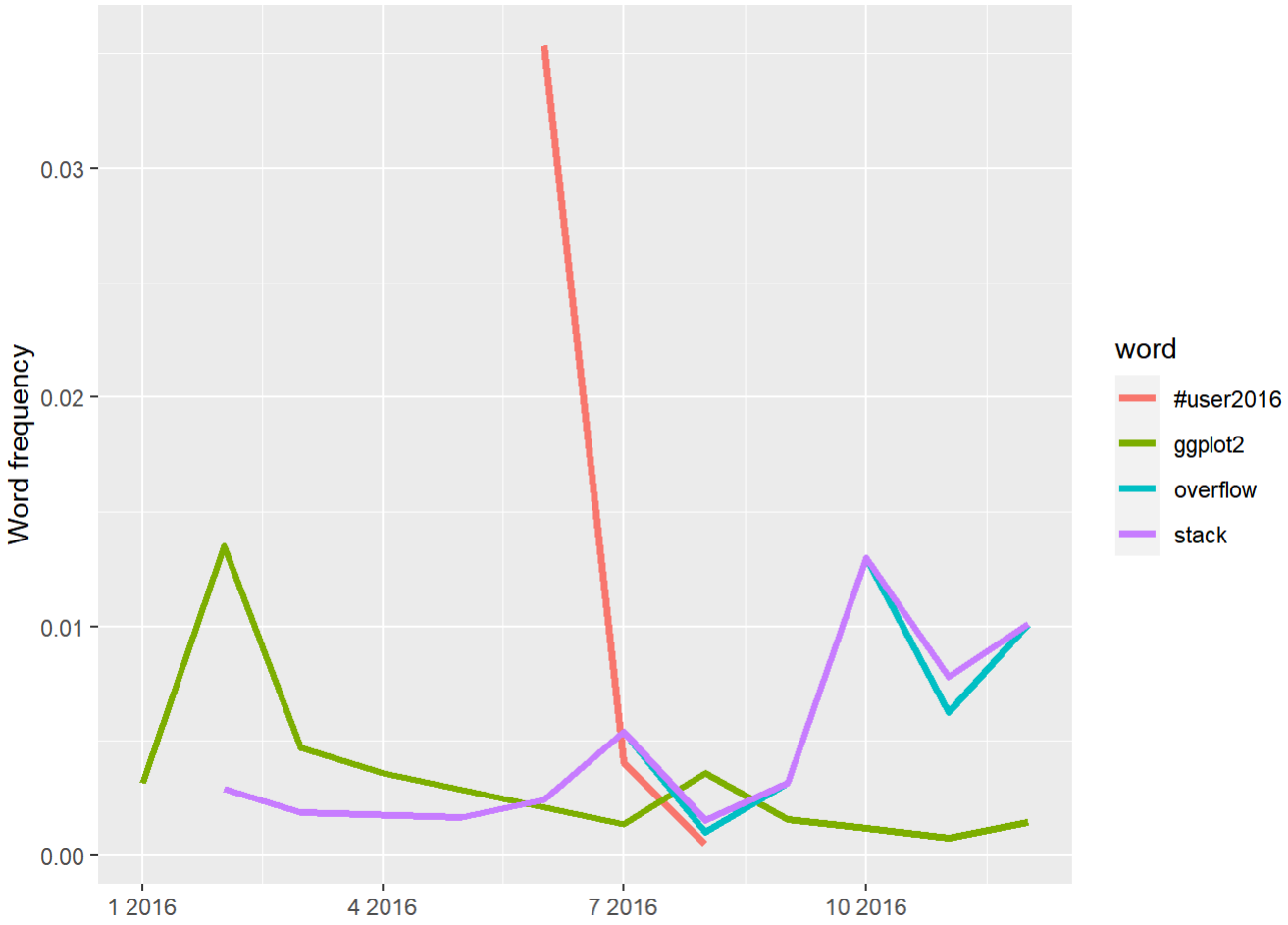
이제 가장 유의한 기울기를 찾아볼 수 있습니다. 어떤 단어가 각 두명의 트위터에서 적당히 중요한 수준의 빈도로 변경되었는지 볼 수 있습니다.

Which words have changed in frequency at a moderately significant level in our tweets?

```
top_slopes <- slopes %>%
  filter(adjusted.p.value < 0.05)

top_slopes
```

```
## # A tibble: 6 x 9
##   person word  data  term  estimate std.error statistic p.value adjusted.p.value
##   <chr>  <chr> <lis> <chr>    <dbl>     <dbl>     <dbl>   <dbl>            <dbl>
## 1 David  ggpl~ <tib~ time~ -8.10e-8   2.00e-8     -4.06 4.90e-5       0.00147
## 2 Julia  #rst~ <tib~ time~ -4.55e-8   1.12e-8     -4.08 4.41e-5       0.00137
## 3 Julia  post  <tib~ time~ -5.17e-8   1.49e-8     -3.48 5.01e-4       0.0145
## 4 David  over~ <tib~ time~  7.00e-8   2.23e-8      3.13 1.73e-3       0.0467
## 5 David  stack <tib~ time~  7.42e-8   2.19e-8      3.39 7.09e-4       0.0198
## 6 David  #use~ <tib~ time~ -8.26e-7   1.55e-7     -5.31 1.07e-7       0.00000344
```

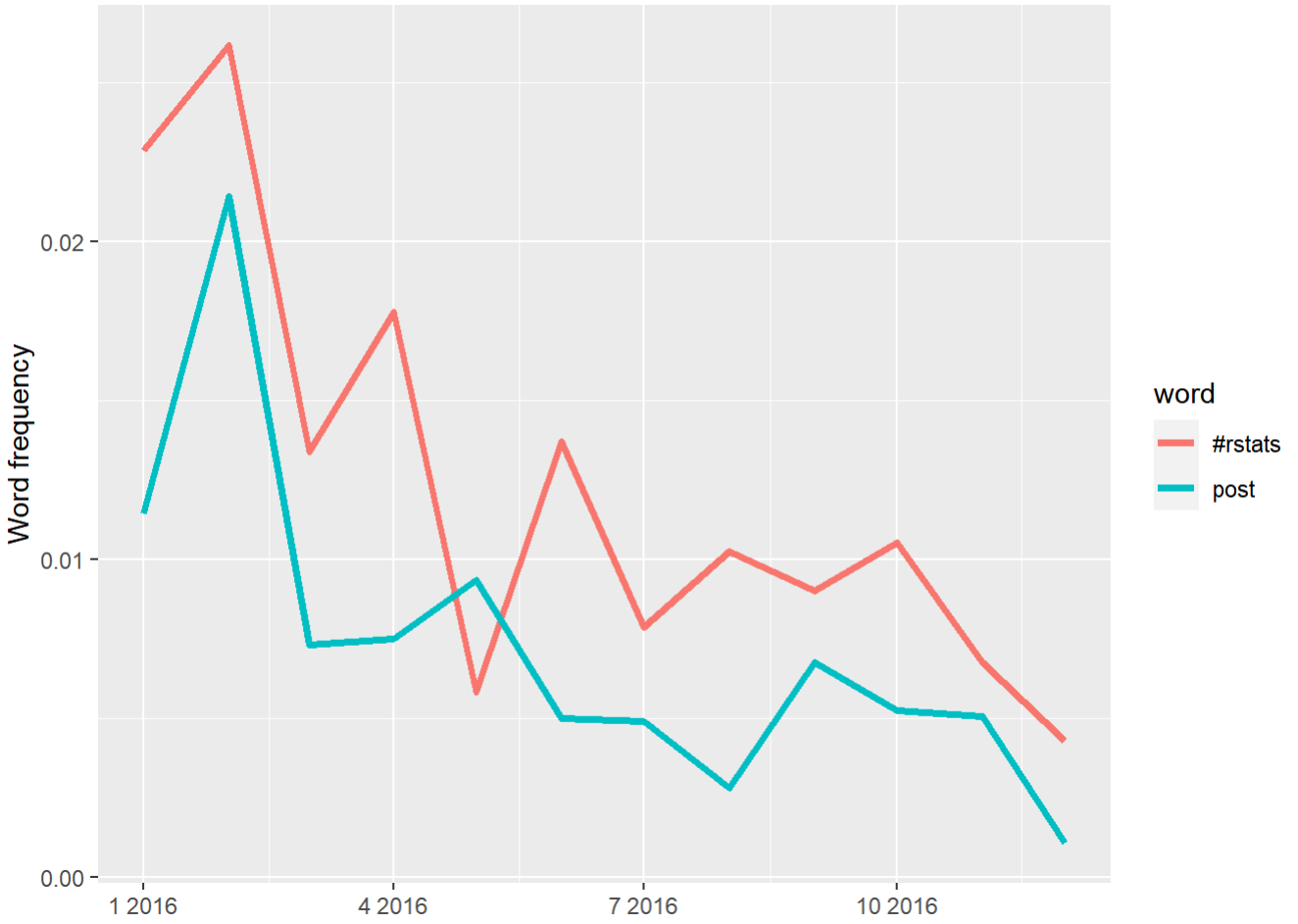
```
words_by_time %>%
  inner_join(top_slopes, by = c("word", "person")) %>%
  filter(person == "David") %>%
  ggplot(aes(time_floor, count/time_total, color = word)) +
  geom_line(size = 1.3) +
  labs(x = NULL, y = "Word frequency")
```



David가 UseR 컨퍼런스에 참여하는 동안 해당 컨퍼런스에 대해 많은 트윗을 했다가 곧 중단한 점을 볼 수 있습니다. 그리고 그는 연말까지 Stack Overflow에 대해서 더 많은 트윗을 했고, 그 해가 가기까지 ggplot2에 대해서는 더 적게 언급함을 볼 수 있습니다.

다음으로 Julia의 트윗에서 빈도가 급격하게 바뀐 단어에 대하여 시각화 해보았습니다.

```
words_by_time %>%
  inner_join(top_slopes, by = c("word", "person")) %>%
  filter(person == "Julia") %>%
  ggplot(aes(time_floor, count/time_total, color = word)) +
  geom_line(size = 1.3) +
  labs(x = NULL, y = "Word frequency")
```



Julia에 대한 유의한 기울기는 모두 음수임을 볼 수 있습니다. 이것은 그녀가 어떤 특정한 단어를 사용하여 더 높은 비율로 트윗을 하지 않고 대신 다양한 단어를 사용했다는 것을 의미하게 됩니다.

연초에는 위 그림에 나와있는 단어들이 높은 비율로 포함되어 있다가 연말로 갈수록 점차 빈도가 줄어듦을 확인할 수 있었습니다.

7.5 Favorites and retweets

트위터의 또 다른 중요한 특징은 얼마나 많은 시간 동안 사람들이 즐겨찾기를 하거나 리트윗을 하는가를 알 수 있다는 점입니다. Julia와 David의 트윗에 대해 어느 단어가 다시 쓰이거나 선호될지 확인해보고자 합니다.

사용자가 자신의 트위터 아카이브를 내려받을 때 즐겨찾기와 리트윗은 포함되지 않으므로 이 정보까지 포함되게 작성자의 트위터 데이터셋을 다시 작성하였습니다.

작성자는 트위터 API를 통해 트윗에 접근했고 각자 약 3,200개의 트윗을 내려받았습니다.

```
tweets_julia <- read_csv("data/juliasilge_tweets.csv")
tweets_dave <- read_csv("data/drob_tweets.csv")
tweets <- bind_rows(tweets_julia %>%
                      mutate(person = "Julia"),
                    tweets_dave %>%
                      mutate(person = "David")) %>%
  mutate(created_at = ymd_hms(created_at))

tweets
```

```
## # A tibble: 6,410 x 7
##         id created_at          source   retweets favorites text           person
##      <dbl> <dttm>              <chr>       <dbl>     <dbl> <chr>          <chr>
##  1 8.04e17 2016-12-01 18:09:04 Twitter~       67         0 "RT @neingeis~ Julia
##  2 8.04e17 2016-12-01 18:07:37 Twitter~        1         0 "RT @Jowanza:~ Julia
##  3 8.04e17 2016-12-01 16:44:03 Twitter~        0         0 "My score is ~ Julia
##  4 8.04e17 2016-12-01 16:42:03 Twitter~        0         9 "It's snowing~ Julia
##  5 8.04e17 2016-12-01 13:11:37 Twitter~        0         1 "@dataandme I~ Julia
##  6 8.04e17 2016-12-01 02:57:15 Twitter~        0         2 "@jkru @astro~ Julia
##  7 8.04e17 2016-12-01 02:56:10 Twitter~        0        11 "Was Julie he~ Julia
##  8 8.04e17 2016-11-30 18:55:59 Twitter~        0         2 "@JennyBryan ~ Julia
##  9 8.04e17 2016-11-30 18:41:46 Twitter~        0         2 "@Jowanza I h~ Julia
## 10 8.04e17 2016-11-30 18:40:27 Twitter~        0        17 "Am I downloa~ Julia
## # ... with 6,400 more rows
```

이제 이 데이터셋을 tidy structure로 변환해볼 수 있습니다. 이 데이터셋에서 모든 리트윗과 응답을 삭제해 David와 Julia가 직접 게시한 정식 트윗만 살펴보고자 하였습니다.

```
tidy_tweets <- tweets %>%
  filter(!str_detect(text, "^(RT|@)")) %>%
  mutate(text = str_remove_all(text, remove_reg)) %>%
  unnest_tokens(word, text, token = "tweets", strip_url = TRUE) %>%
  filter(!word %in% stop_words$word,
         !word %in% str_remove_all(stop_words$word, "'"))

tidy_tweets
```

```
## # A tibble: 11,014 x 7
##         id created_at          source        retweets favorites person word
##      <dbl> <dttm>              <chr>            <dbl>     <dbl> <chr>  <chr>
##  1 8.04e17 2016-12-01 16:44:03 Twitter Web ~        0         0 Julia  "score"
##  2 8.04e17 2016-12-01 16:44:03 Twitter Web ~        0         0 Julia  "50"
##  3 8.04e17 2016-12-01 16:42:03 Twitter Web ~        0         9 Julia  "snowing"
##  4 8.04e17 2016-12-01 16:42:03 Twitter Web ~        0         9 Julia  "WU0001f~
##  5 8.04e17 2016-12-01 16:42:03 Twitter Web ~        0         9 Julia  "drinkin~
##  6 8.04e17 2016-12-01 16:42:03 Twitter Web ~        0         9 Julia  "tea"
##  7 8.04e17 2016-12-01 16:42:03 Twitter Web ~        0         9 Julia  "WU0001f~
##  8 8.04e17 2016-12-01 16:42:03 Twitter Web ~        0         9 Julia  "#rstats"
##  9 8.04e17 2016-12-01 16:42:03 Twitter Web ~        0         9 Julia  "WU0001f~
## 10 8.04e17 2016-12-01 02:56:10 Twitter Web ~        0        11 Julia  "julie"
## # ... with 11,004 more rows
```

먼저 각 트윗이 리트윗된 횟수를 살펴볼 수 있습니다. 각 저자에 대한 전체 리트윗 횟수를 계산해보았습니다.

```
totals <- tidy_tweets %>%
  group_by(person, id) %>%
  summarise(rts = first(retweets)) %>%
  group_by(person) %>%
  summarise(total_rts = sum(rts))

totals
```

```
## # A tibble: 2 x 2
##   person total_rts
##   <chr>      <dbl>
## 1 David      13014
## 2 Julia       1750
```

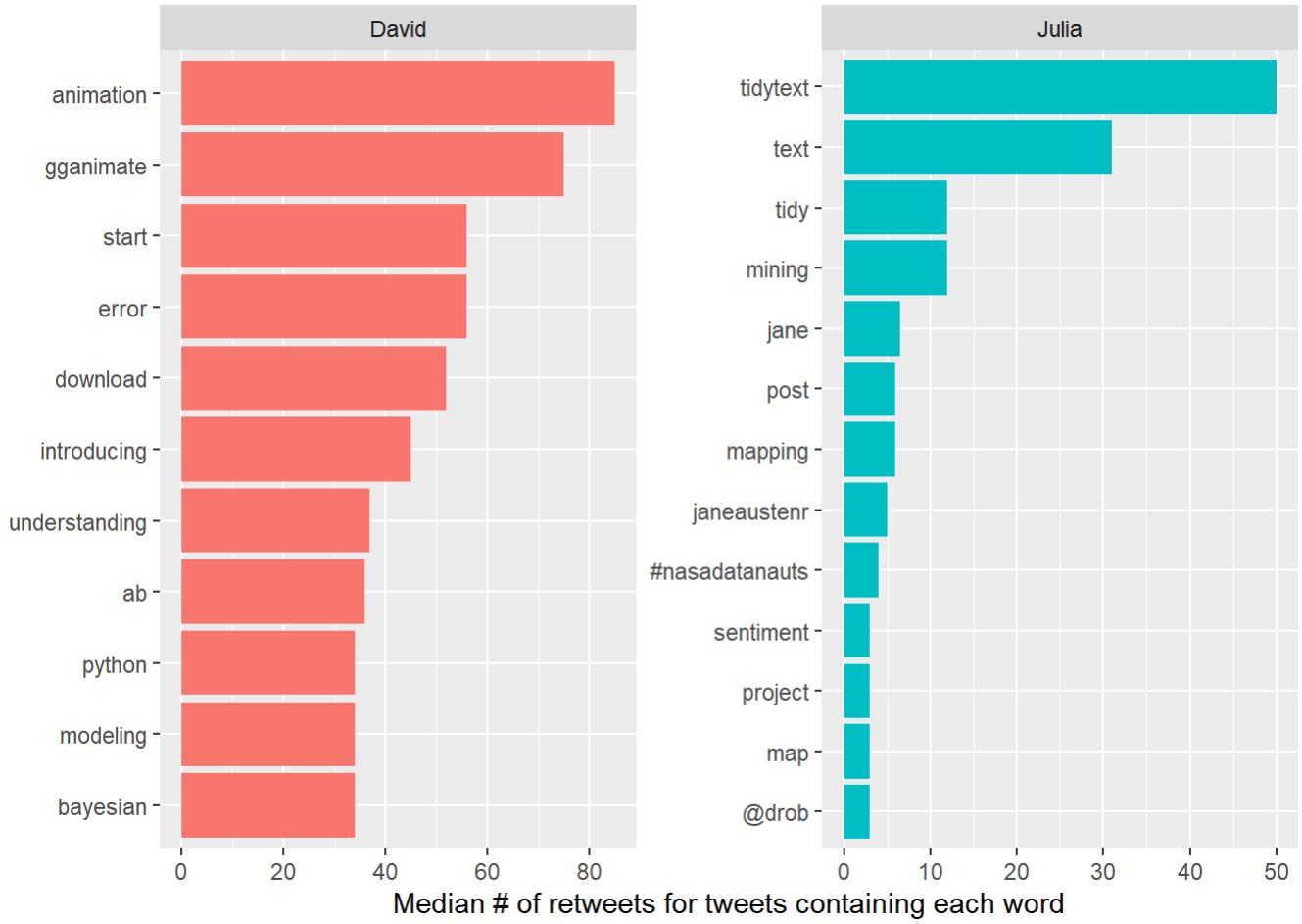
이제 각 단어와 사람에 대해 각 단어가 몇 번 리트윗이 되었는지를 세고, 각 사람과 단어에 대한 중위수 리트윗을 알아내며, 각 저자가 사용한 단어 수를 세어 데이터프레임에 연결할 수 있습니다. 적어도 다섯 번은 단어를 쓰도록 filter()를 하였습니다.

```
word_by_rts <- tidy_tweets %>%
  group_by(id, word, person) %>%
  summarise(rts = first(retweets)) %>%
  group_by(person, word) %>%
  summarise(retweets = median(rts), uses = n()) %>%
  left_join(totals) %>%
  filter(retweets != 0) %>%
  ungroup()

word_by_rts %>%
  filter(uses >= 5) %>%
  arrange(desc(retweets))
```

```
## # A tibble: 170 x 5
##    person word          retweets  uses total_rts
##    <chr>  <chr>            <dbl> <int>     <dbl>
##  1 David  animation           85     5     13014
##  2 David  gganimate           75     6     13014
##  3 David  error               56     7     13014
##  4 David  start               56     6     13014
##  5 David  download            52     5     13014
##  6 Julia  tidytext            50     7      1750
##  7 David  introducing         45     6     13014
##  8 David  understanding       37     6     13014
##  9 David  ab                  36     5     13014
## 10 David  bayesian            34     7     13014
## # ... with 160 more rows
```

`gganimate`, `tidytext` 같이 두 사람이 개발에 참여하고 있는 패키지에 대한 트윗이 표시됨을 볼 수 있습니다. 각 계정에 대해 최고 중위수 리트윗을 가진 단어 또한 볼 수 있습니다.

```
word_by_rts %>%
  filter(uses >= 5) %>%
  group_by(person) %>%
  slice_max(retweets, n = 10) %>%
  arrange(retweets) %>%
  ungroup() %>%
  mutate(word = factor(word, unique(word))) %>%
  ungroup() %>%
  ggplot(aes(word, retweets, fill = person)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ person, scales = "free", ncol = 2) +
  coord_flip() +
  labs(x = NULL,
       y = "Median # of retweets for tweets containing each word")->g1

g1
```

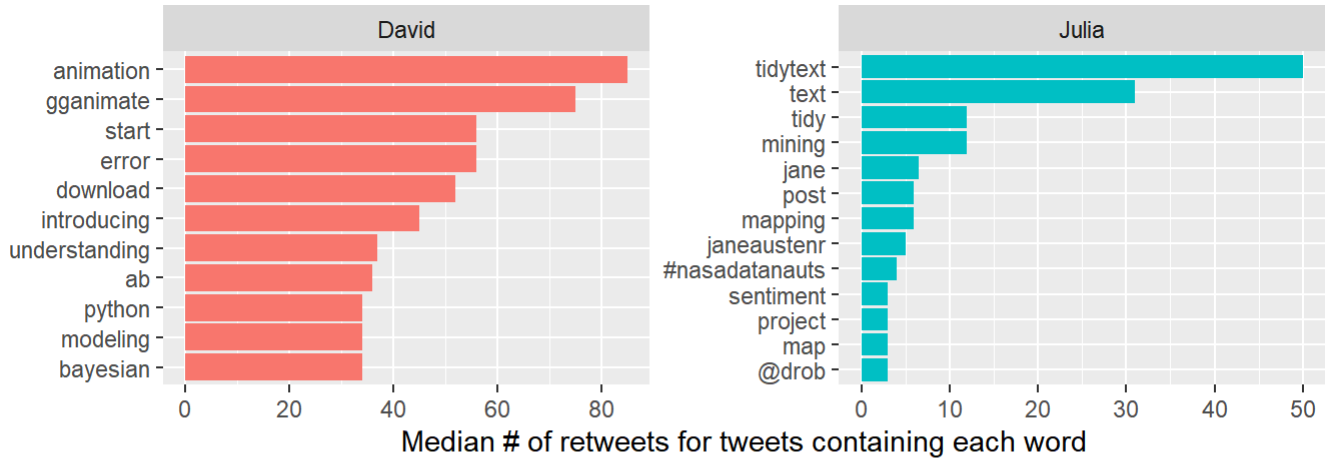


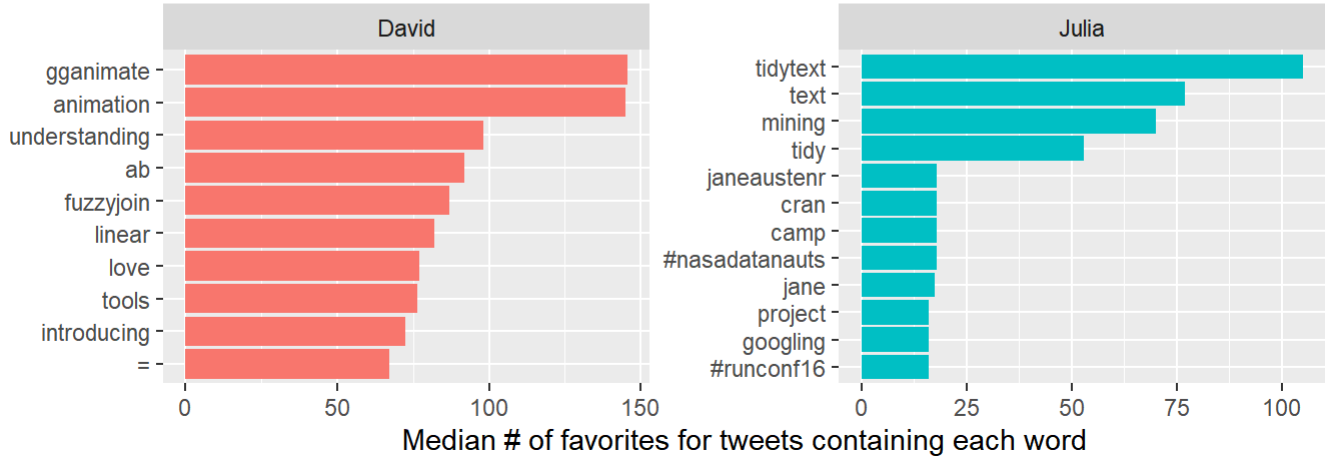
꾸준히 retweets이 많이 된 단어들을 뽑아보니 대부분 R 패캐지들임을 알 수 있습니다.

이제 이와 유사한 절차를 따라 어떤 단어가 더 많은 즐겨찾기를 이끌었는지 확인할 수 있습니다.

이후 리트윗의 관점과 즐겨찾기의 관점을 비교해볼 수 있습니다.

```
totals <- tidy_tweets %>%
  group_by(person, id) %>%
  summarise(favs = first(favorites)) %>%
  group_by(person) %>%
  summarise(total_favs = sum(favs))

word_by_favs <- tidy_tweets %>%
  group_by(id, word, person) %>%
  summarise(favs = first(favorites)) %>%
  group_by(person, word) %>%
  summarise(favorites = median(favs), uses = n()) %>%
  left_join(totals) %>%
  filter(favorites != 0) %>%
  ungroup()

library(gridExtra)

word_by_favs %>%
  filter(uses >= 5) %>%
  group_by(person) %>%
  slice_max(favorites, n = 10) %>%
  arrange(favorites) %>%
  ungroup() %>%
  mutate(word = factor(word, unique(word))) %>%
  ungroup() %>%
  ggplot(aes(word, favorites, fill = person)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ person, scales = "free", ncol = 2) +
  coord_flip() +
  labs(x = NULL,
       y = "Median # of favorites for tweets containing each word") -> g2

grid.arrange(g1,g2)
```





두 관점에 있어 차이는 크게 안보이는 것을 볼 수 있습니다. 일반적으로 리트윗을 한다면 같은 단어들이 즐겨찾기로 이어지게됩니다.

7.7 Summary

이 장은 텍스트 데이터 세트를 이해하기 위해 통합적으로 탐색해 온 개념과 코드를 통합하는 방법을 보여주는 처음부터 끝까지의 분석인 첫 번째 사례 연구였다.

단어 빈도를 비교하면 어떤 단어를 자주 트윗했는지 볼 수 있었고, 로그 오즈비로 각 계정에서 어떤 단어가 트윗될 가능성이 더 높은지 알 수 있었습니다.

glm()을 통하여 시간이 지남에 따라 변화율이 큰 단어를 찾을 수 있었으며, 마지막으로 트윗에서 어떤 단어가 리트윗과 즐겨찾기를 더 많이 이끌었는지를 찾을 수 있습니다.

이러한 모든 것은 단어를 유사하고 다른 방식으로 사용하는 방법과 트윗의 특성이 어떻게 변화하거나 서로 비교를 하는지를 측정하기 위한 접근법의 예입니다.

이는 다른 유형의 텍스트에도 적용할 수 있는 텍스트 마이닝에 대한 유연한 접근 방식입니다.