L12. Textmining (1)

Sim, Min Kyu, Ph.D., mksim@seoultech.ac.kr

SEOULTECH 서울과학기술대학교 데이터사이언스학과

1. I. Textbook and Dataset
2. II. Tidy Text

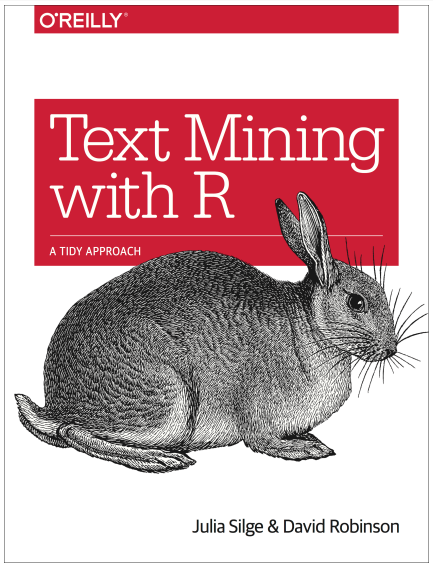
Section 1

I. Textbook and Dataset

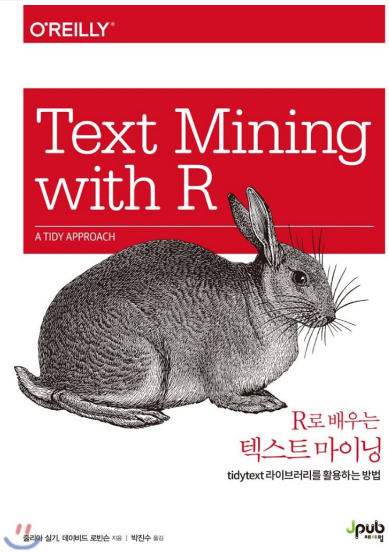
Textbook for this module

- Text Mining with R written by Julia Silge & David Robinson

```
knitr::include_graphics("fig/cover.png")
```




```
knitr::include_graphics("fig/cover_kr.jpg")
```



- Access
  - Available free online: https://www.tidytextmining.com/
  - 'Published with **bookdown**'
  - GitHub repository for this site: https://github.com/dgrtwo/tidy-text-mining

Github repository

```
knitr::include_graphics("fig/github.png")
```

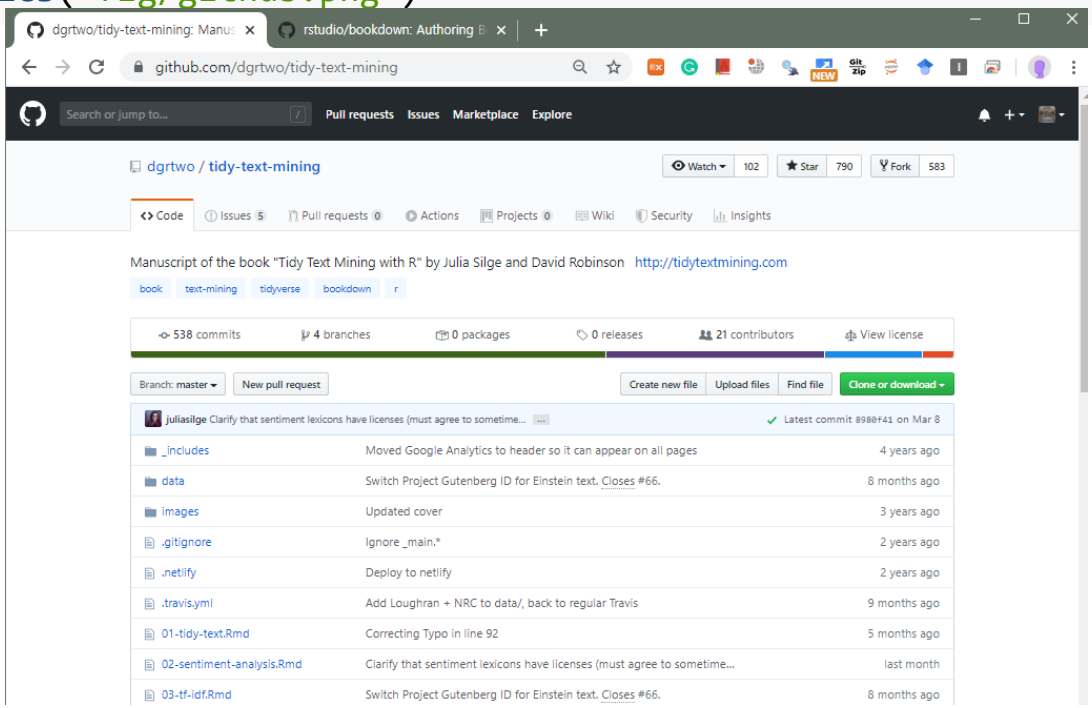


Figure 1: GitHub 스크린샷

- `01-tidy-text.Rmd`은 챕터 1의 소스파일
- Online acess가 아닌 로컬에 다운 받는 방법은?
- ~~pdf 파일로 제작해서 보유할 수도 있을까?~~

bookdown

- Yihui Xie가 제작한 R package (Software Engineer at RStudio)
- `bookdown` allows to combine `Rmd` files into a book, seemlessly
- 공식 페이지
  - https://bookdown.org/home/about/ (방문해볼만함!)
  - https://github.com/rstudio/bookdown
- (Selected) Authored books
  - bookdown: Authoring Books and Technical Documents with R Markdown https://bookdown.org/yihui/bookdown/
  - R Markdown: The Definitive Guide https://bookdown.org/yihui/rmarkdown/

`bookdown`으로 제작된 책을 local에 저장하는 방법

- Authoring process with `bookdown`
  0. `install.packages("bookdown")`
  1. Initiate a project that will generate `xxxx.Rproj`
  2. Compose `index.Rmd` that contains YAML meta-data
  3. Compose `01-xxxx.Rmd` for Chapter 1
  4. Compose `02-xxxx.Rmd` for Chapter 2
  5. …(continue writing all chapters)…
  6. …(All `Rmd` files must be in same folder)…
  7. Compile the `xxxx.Rproj` then boo-yah!
- Recovering from github
  0. 저자가 source를 공개한 경우에 github로 가서 'Download ZIP'
  1. 각 챕터가 필요하다면 단일 챕터 `Rmd`를 Knit하면 됨
  2. 전체 book을 원한다면 `install.packages("bookdown")`을 한 후에
  3. Rstudio를 닫고 폴더의 `xxxx.Rproj`를 더블클릭해서 Rstudio 실행
  4. Environment 가 포함된 패널에서 'Build' 탭을 이용해서 Render
  5. 에러메시지를 확인하면서 추가적인 패키지 설치
  6. `index.Rmd` 파일의 메타 데이터를 수정하면서 customizing 가능

Dataset

Consumer Reviews of Amazon Products

- A list of over 34,000 reviews of Amazon products like the Kindle, Fire TV, etc
- https://www.kaggle.com/datafiniti/consumer-reviews-of-amazon-products
- This is a list of over 34,000 consumer reviews for Amazon products like the Kindle, Fire TV Stick, and more provided by Datafiniti's Product Database. The dataset includes basic product information, rating, review text, and more for each product.
- Note that this is a sample of a large dataset. The full dataset is available through Datafiniti.

```
library(tidyverse)
amz <- read_csv("data/Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.csv")
```

Exploration

```
options(tibble.width = Inf) # show full columns
amz <- amz %>%
  select(id, name, asins, brand, primaryCategories, manufacturer,
         reviews.doRecommend, reviews.numHelpful, reviews.rating,
         reviews.text, reviews.title, reviews.username)
amz[1:2,1:9]
```

```
## # A tibble: 2 x 9
##   id                     name
##   <chr>                  <chr>
## 1 AVqVGZNvQMlgsOJE6eUY "Amazon Kindle E-Reader 6\" Wifi (8th Generation, 2016)"
## 2 AVqVGZNvQMlgsOJE6eUY "Amazon Kindle E-Reader 6\" Wifi (8th Generation, 2016)"
##   asins      brand  primaryCategories manufacturer reviews.doRecommend
##   <chr>      <chr>  <chr>             <chr>        <lgl>
## 1 B00ZV9PXP2 Amazon Electronics       Amazon       FALSE
## 2 B00ZV9PXP2 Amazon Electronics       Amazon       TRUE
##   reviews.numHelpful reviews.rating
##                <dbl>          <dbl>
## 1                  0              3
## 2                  0              5
```

```
amz[1:2,10:12]
```

```
## # A tibble: 2 x 3
##   reviews.text                          reviews.title           reviews.username
##   <chr>                                 <chr>                   <chr>
## 1 I thought it would be as big as sma~ Too small               llyyue
## 2 This kindle is light and easy to us~ Great light reader. Eas~ Charmi
```

Which product has the best rating?

```
amz %>%
  group_by(name) %>%
  summarize(review_count = n(),
            avg_rating = mean(reviews.rating)) %>%
  filter(review_count >= 100) %>%
  arrange(desc(avg_rating))
```

```
## # A tibble: 11 x 3
##    name                                                review_count avg_rating
##    <chr>                                                      <int>      <dbl>
##  1 "Amazon - Echo Plus w/ Built-In Hub - Silver"                590       4.75
##  2 "Fire HD 10 Tablet, 10.1 HD Display, Wi-Fi, 16 GB - ~        106       4.67
##  3 "Amazon Echo Show Alexa-enabled Bluetooth Speaker wi~         845       4.66
##  4 "All-New Fire HD 8 Tablet, 8\" HD Display, Wi-Fi, 16~         797       4.60
##  5 "Kindle E-reader - White, 6 Glare-Free Touchscreen D~         159       4.58
##  6 "Fire Kids Edition Tablet, 7 Display, Wi-Fi, 16 GB, ~         561       4.58
##  7 "Fire Kids Edition Tablet, 7 Display, Wi-Fi, 16 GB, ~         217       4.52
##  8 "Fire Tablet with Alexa, 7\" Display, 16 GB, Magenta~         101       4.51
##  9 "Brand New Amazon Kindle Fire 16gb 7\" Ips Display T~         467       4.51
## 10 "Amazon Tap - Alexa-Enabled Portable Bluetooth Speak~         225       4.51
## 11 "Fire Tablet, 7 Display, Wi-Fi, 16 GB - Includes Spe~         371       4.46
```

```
"This lecture note is a modified version of Prof. Hyunwoo Park"
"https:\\hyunwoopark.com"
```

Section 2

II. Tidy Text

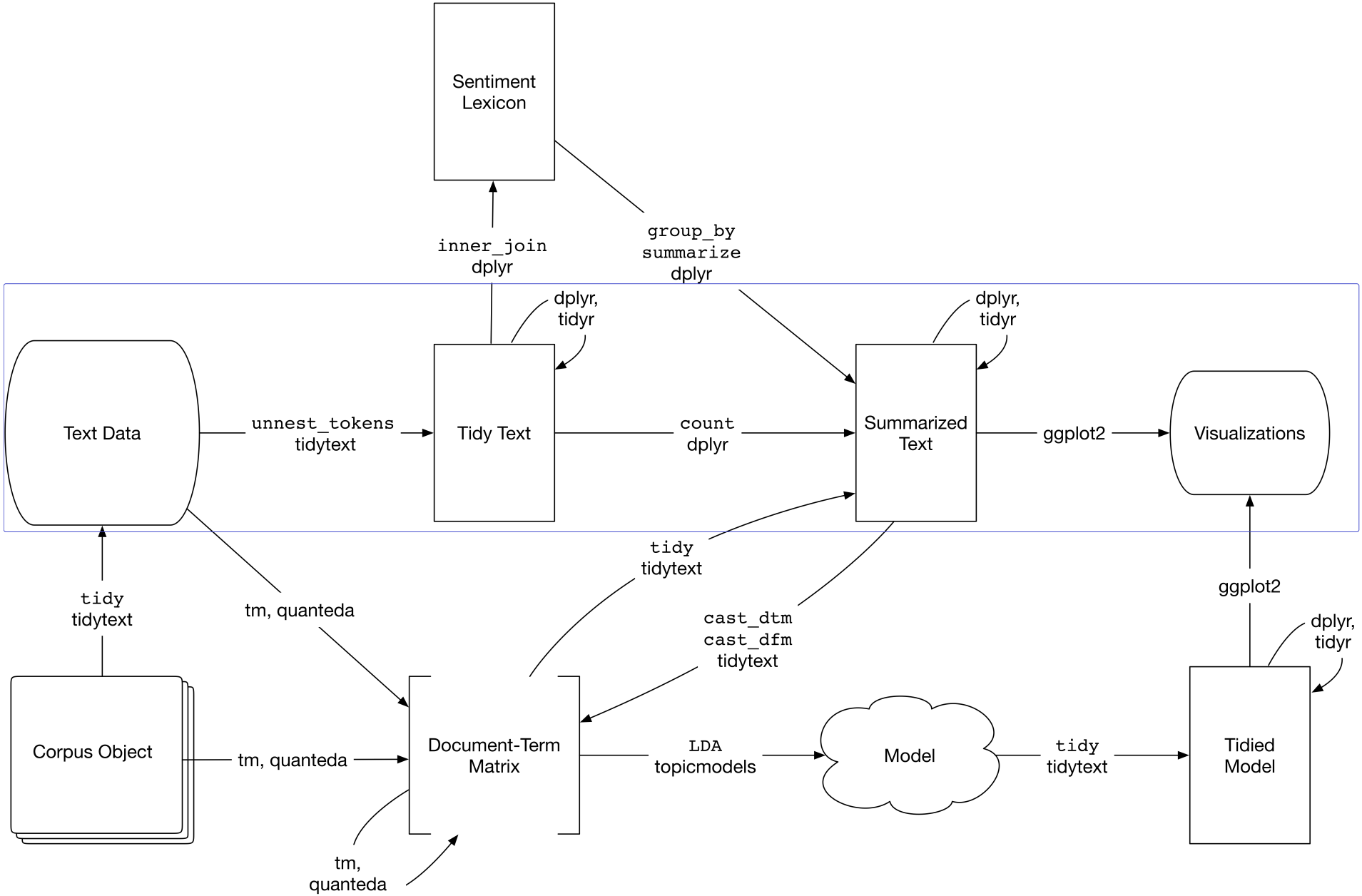
Unstructured data (비정형 data)

- 데이터가 깨끗하게 정리되서 준비되는 일은 극히 드물다
- Images, videos, and texts가 대표적인 비정형 데이터
- 비정형 데이터를 프로세싱하여 수치로 표현하는 것 자체로도 active한 연구분야
- 텍스트를 다루는 접근방법은 상당히 표준화 되있는 편이다
- 여러곳에서 많은 양의 텍스트 데이터를 저장하고 있기에, 포텐셜이 많은 영역

Tidy text data

- Tidy data
  - **Each row is an observation.**
  - Each column is a variable.
- Tidy text data
  - **Each row is a token (의미 단위) per document.**
  - Each column is a variable.

- Why still want text data to be tidy?
  - 단정해서 분석이 쉬움
  - `dplyr`와 `ggplot2`를 적용하기 쉬움
- A token?
  - Single word, n-gram, sentence, or paragraph



- This module (`II. Tidy Text`)에서는 박스안의 프로세스를 진행

- This module (**II. Tidy Text**)에서는 위 그림의 박스안의 프로세스를 진행
- Text Data → `tidytext::unnest_tokens()` → Tidy Text → `dplyr::count()` → Summarized Text → Visualization

Convert to tidy text by *tidytext::unnest_tokens()*

```
library(tidytext)
tidy_amz <- amz %>%
  unnest_tokens(word, reviews.text)
dim(amz)
```

```
## [1] 5000   12
```

```
dim(tidy_amz)
```

```
## [1] 155258     12
```

```
tidy_amz %>% select(tail(names(.), 4)) # print the last four columns
```

```
## # A tibble: 155,258 x 4
##    reviews.rating reviews.title reviews.username word
##             <dbl> <chr>         <chr>            <chr>
##  1              3 Too small     llyyue           i
##  2              3 Too small     llyyue           thought
##  3              3 Too small     llyyue           it
##  4              3 Too small     llyyue           would
##  5              3 Too small     llyyue           be
##  6              3 Too small     llyyue           as
##  7              3 Too small     llyyue           big
##  8              3 Too small     llyyue           as
##  9              3 Too small     llyyue           small
## 10              3 Too small     llyyue           paper
```

Count the occurrence of each word

```
tidy_amz %>% count(word, sort=TRUE)
```

```
## # A tibble: 5,789 x 2
##    word      n
##    <chr> <int>
##  1 the    6747
##  2 and    5028
##  3 to     5021
##  4 it     4819
##  5 i      4598
##  6 for    3717
##  7 a      3489
##  8 is     3034
##  9 my     2876
## 10 this   2643
## # ... with 5,779 more rows
```

Cleaning 1: Remove those 'stop words'

- Some tokens are obviously and intuitively not meaningful for further analysis.
- Those tokens are: a, an, the, i, you, he, she, is, are, be, etc. (전치사, 관사, be동사, 조동사, 대명사등)
- These words are called **'stop words'**. Removing them is a standard step.
- In the natural language processing (NLP) literature, there is a prepared set of words to be removed from text analysis. They come with **`tidytext`** package.

```
stop_words
```

```
## # A tibble: 1,149 x 2
##    word        lexicon
##    <chr>       <chr>
##  1 a           SMART
##  2 a's         SMART
##  3 able        SMART
##  4 about       SMART
##  5 above       SMART
##  6 according   SMART
##  7 accordingly SMART
##  8 across      SMART
##  9 actually    SMART
## 10 after       SMART
```

Cleaning 2: Remove words containing 'numbers'

- Oftentimes, some tokens are only numbers (e.g., '100') or words containing numbers (e.g., '1st').
- 아래 코드는 **tidy_amz$word**에서 숫자를 포함한 token을 모두 나열함.

```
word_with_num <- tidy_amz %>%
  select(word) %>%
  filter(str_detect(word, "^[0-9]")) %>% unique()
set.seed(123)
word_with_num %>% sample_n(3)
```

```
## # A tibble: 3 x 1
##   word
##   <chr>
## 1 87
## 2 76
## 3 2nd
```

- Stop words와 마찬가지로 비정형 데이터를 정형으로 만드는 과정에서는 일정 부분 의미의 손실이 있을 수 밖에 없음.
- e.g. 'I have only used this product for 2yr' vs 'I will keep this cell phone for at least 2yr'
- e.g. 'I am 100% certain that this is good'

Cleaning 3: Stemming (어근과 어미의 분리)

- Another standard, but more optional, step is called stemming.
- Words are inflected depending on the context in a sentence.
- 'love, loves, loved' all share the same root, 'love'.
- There are several stemmers.
- 'Porter stemmer'는 오래되고 널리 쓰이는 어미제거 database
- `SnowballC::wordStem()`을 token에 적용

```
set.seed(111)
library(SnowballC)
tidy_amz %>% mutate(root = wordStem (word)) %>%
  select(word, root) %>% sample_n(10)
```

```
## # A tibble: 10 x 2
##    word       root
##    <chr>      <chr>
##  1 a          a
##  2 you        you
##  3 absolutely absolut
##  4 and        and
##  5 and        and
##  6 gift       gift
##  7 noticed    notic
```

Cleaning and Re-Count

```
tidy_amz <- tidy_amz %>%
  anti_join(stop_words) %>%     # cleaning 1
  anti_join(word_with_num) %>%  # cleaning 2
  mutate(root = wordStem(word)) # cleaning 3
```

```
word_count <- tidy_amz %>%
  count(word, sort=TRUE)
word_count
```

```
## # A tibble: 5,042 x 2
##    word       n
##    <chr>  <int>
##  1 tablet  1309
##  2 love    1090
##  3 easy     822
##  4 bought   785
##  5 kindle   764
##  6 amazon   694
##  7 echo     693
##  8 alexa    513
##  9 loves    506
## 10 screen   500
```

```
root_count <- tidy_amz %>%
  count(root, sort=TRUE)
root_count
```

```
## # A tibble: 3,645 x 2
##    root       n
##    <chr>  <int>
##  1 love    1719
##  2 tablet  1458
##  3 easi     823
##  4 kindl    821
##  5 bought   785
##  6 echo     724
##  7 amazon   701
##  8 read     649
##  9 purchas  561
## 10 product  560
```

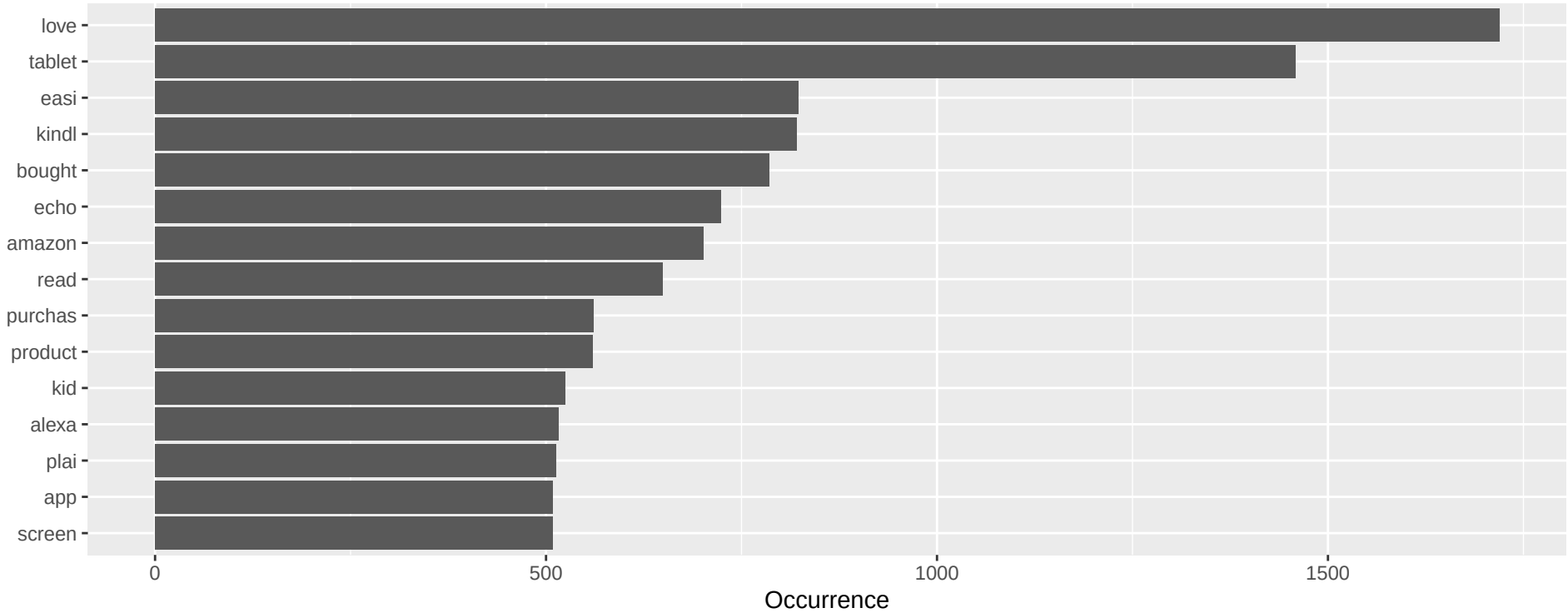
Presentation 1: Word frequency table

```
root_count
```

```
## # A tibble: 3,645 x 2
##    root        n
##    <chr>   <int>
##  1 love     1719
##  2 tablet   1458
##  3 easi      823
##  4 kindl     821
##  5 bought    785
##  6 echo      724
##  7 amazon    701
##  8 read      649
##  9 purchas   561
## 10 product   560
## # ... with 3,635 more rows
```

Presentation 2: Word frequency chart

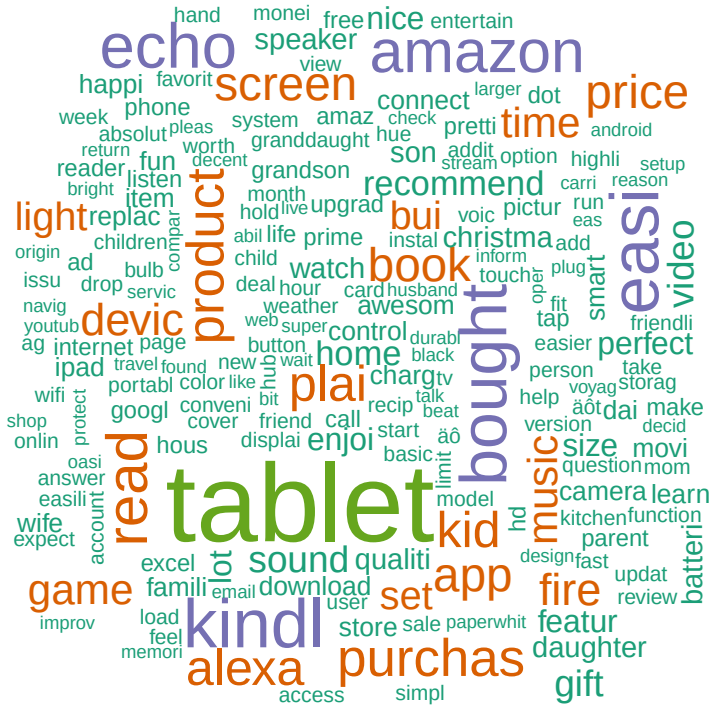
```
root_count %>% top_n(15) %>%
  ggplot() +
  geom_col(aes(x=reorder(root,n), y=n)) +
  coord_flip() +
  xlab(NULL) + ylab("Occurrence")
```



`reorder(root,n)`은 `root`변수를 `n`을 기준으로 reorder하여 사용하라는 의미

Presentation 3: Word frequency chart – Word cloud

```
library(wordcloud)
library(RColorBrewer)
wordcloud(root_count$root, root_count$n,
          min.freq=50, colors=brewer.pal(5, "Dark2"))
```



- `RColorBrewer`: Provides some sensible preset color scale
- https://www.rdocumentation.org/packages/RColorBrewer/versions/1.1-2/topics/RColorBrewer