

M31 - Text Mining

LearningSpoonsR

2018-07-08

Contents

- Part 2. 분석 & 시각화 - `ggplot2`
 - Part 1. Text 전처리 - `tm`, `KONLP`
 - Part 3. 일반화 - Writing a function
 - (NEXT - Part. 일반화) - `M32-flexdashboard`
-
- `ggplot2`의 원리와 다른 용법에 대해서 알아봅니다.
 - Text 문서를 입력으로 하여 분석하는 `text-mining` 도구들을 알아봅니다.
 - 함수를 작성하면서, 1) 공통 기능을 일반화 하고, 2) 다른 기능은 분기하여 범용적인 도구로 진화시킵니다.
 - (Next - 다양한 input형태를 모두 처리할 수 있는 도구로 완성시킵니다. (`shiny`))

Part 2. - 분석 & 시각화 - ggplot2

GG - Grammar of Graphics

- Motivation

1. 그래픽스에 대한 원리가 없다면, 그래픽 관련 패키지와 함수는 단지 특수 경우의 모음일 뿐
2. 요리 백과사전을 다 읽는 것 vs. 물과 기름과 불의 작용에 대해서 익히고 백과사전을 **찾아가면서** 요리하는 것

- Advantage

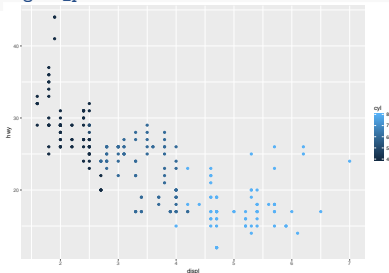
1. 새로운 package나 함수의 등장을 빠르게 흡수
2. 새로운 graphics를 만들어 내는 아이디어가 체계적이 됨

- Features

1. 독립적이고 더할 수 있는 구성 요소들로 그래픽을 표현
2. 개발과정에서 그래프의 특징을 한 가지 씩, 반복적으로 바꾸면서 그래프를 만들어 감
3. 생각의 흐름, 스토리텔링의 흐름과 연계시킬 수 있기에 interactive graphics와 잘 조화됨

구성 요소

```
library(ggplot2)
ggplot(mpg) +
  aes(x = displ, y = hwy, color = cyl) +
  geom_point()
```



- Aesthetics

1. position
2. size
3. color
4. shape

- Geometric Object (geom_)

1. Scatterplot - point
2. Bubblechart - point *Size 비*
3. Barchart - bar *비*
4. Box-and-whisker plot - boxplot *비*
5. Line chart - line

Behind the scene

data

```
head(mpg[,c("displ", "hwy", "cyl")],10)
```

```
## # A tibble: 10 x 3
##   displ hwy  cyl
##   <dbl> <int> <int>
## 1  1.8    29    4
## 2  1.8    29    4
## 3  2.0    31    4
## 4  2.0    30    4
## 5  2.8    26    6
## 6  2.8    26    6
## 7  3.1    27    6
## 8  1.8    26    4
## 9  1.8    25    4
## 10 2.0    28    4
```

aes

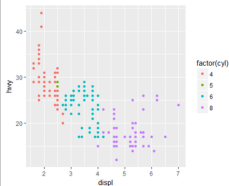
x	y	colour
1.8	29	4
1.8	29	4
2.0	31	4
2.0	30	4
2.8	26	6
2.8	26	6
3.1	27	6
1.8	26	4

geom

x	y	colour	size	shape
0.037	0.531	#F8766D	1	19
0.037	0.531	#F8766D	1	19
0.074	0.594	#F8766D	1	19
0.074	0.562	#F8766D	1	19
0.222	0.438	#00BFC4	1	19
0.222	0.438	#00BFC4	1	19
0.278	0.469	#00BFC4	1	19
0.037	0.438	#F8766D	1	19

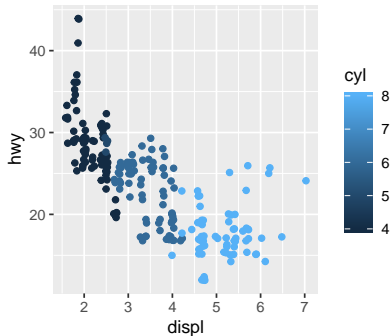
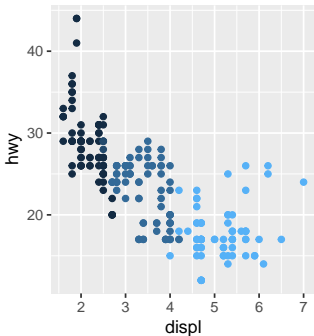
plot

```
library(ggplot2)
ggplot(data = mpg, aes(x = displ, y = hwy, color = factor(cyl))) +
  geom_point()
```



중첩된 관찰값에 노이즈를: position = "jitter"

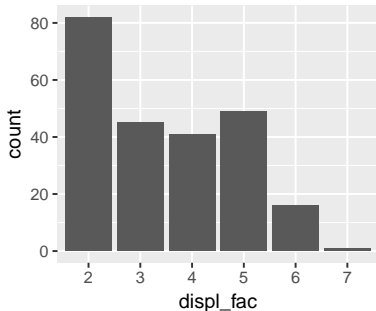
```
library(gridExtra)
a <- ggplot(mpg) + geom_point(aes(displ, hwy, color = cyl))
b <- ggplot(mpg) + geom_point(aes(displ, hwy, color = cyl), position = "jitter")
grid.arrange(a, b, nrow = 1, ncol = 2)
```



Barchart

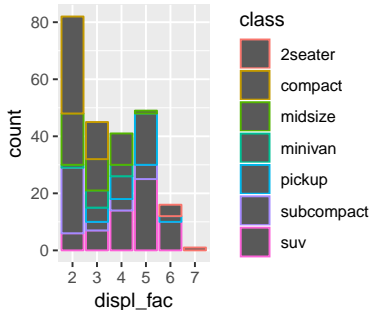
- x축에 이산변수 (discrete value)를 넣고 변수 x의 각각의 값에 대해서 몇 개의 관찰값이 있는지를 보여줌.
- #count #density #distribution

```
mpg$displ_fac <-  
  as.factor(round(mpg$displ,0))  
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac))
```



- x변수 외에도 이산 변수를 추가할 수 있음.

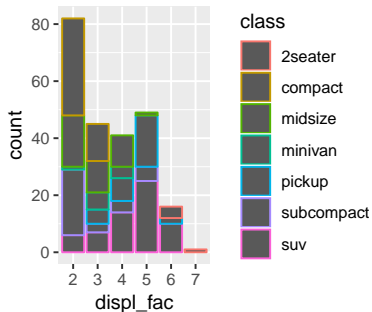
```
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac, color = class))
```



Barchart (Color and Fill)

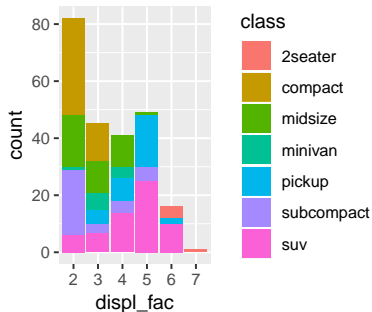
- **color** - 테두리만 됨

```
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac, color = class))
```



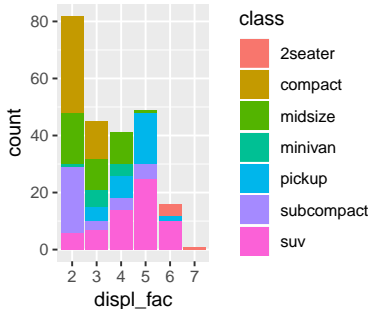
- **fill** - 채워짐

```
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac, fill = class))
```



Barchart(position = "dodge")

```
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac, fill = class))
```



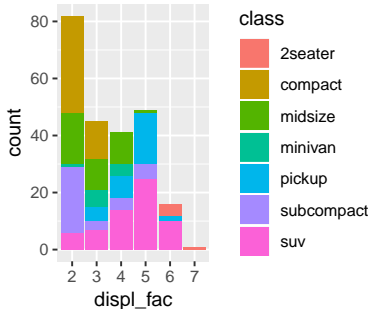
```
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac, fill = class),  
           position = "dodge")
```



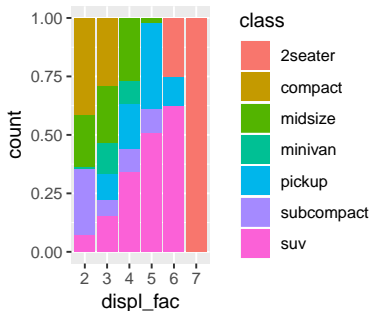
- 2개의 discrete 변수를 잘 처리하는 법?

Barchart(position = "fill")

```
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac, fill = class))
```



```
ggplot(mpg) +  
  geom_bar(aes(x = displ_fac, fill = class),  
           position = "fill")
```

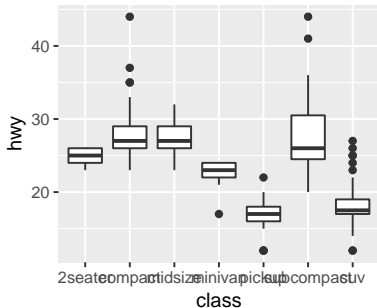


- class의 각각의 displ_fac 값에서의 분포?

Boxplot

- x 변수는 이산 변수이고 x 변수의 각각의 값에 대해서 연속 변수인 y의 분포를 보기 위함.

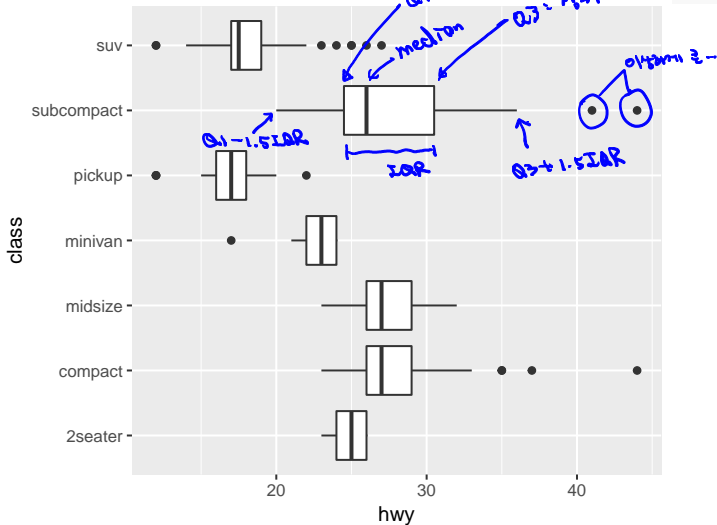
```
ggplot(mpg) +  
  geom_boxplot(aes(x = class, y = hwy))
```



- 점으로 표현된 것은 이상치(outlier)로서 이상하게 높거나 낮은 값
- ~~boxplot의 각 끝은 상위/하위 10%의 값~~
- 박스의 상단은 상위 25%, 하단은 하위 25%
- 박스의 가운데 직선은 중간값
- x변수가 이산 변수이면서 **factor**라면, 변수의 값이 **character**라서 **display**가 복잡할 가능성이 높음
- 이럴때는?

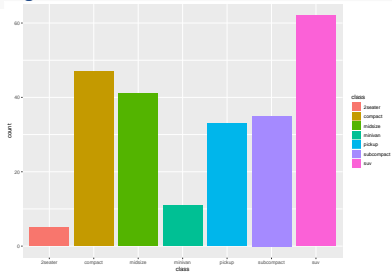
Boxplot(coord_flip())

```
ggplot(mpg) +  
  geom_boxplot(aes(x = class, y = hwy)) +  
  coord_flip()
```

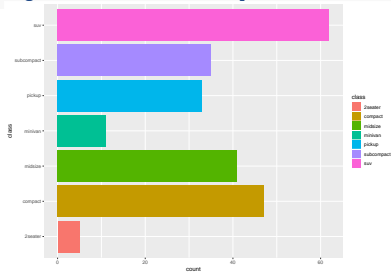


Barplot (coord_flip())

```
ggplot(mpg, aes(x = class, fill = class)) +  
  geom_bar()
```

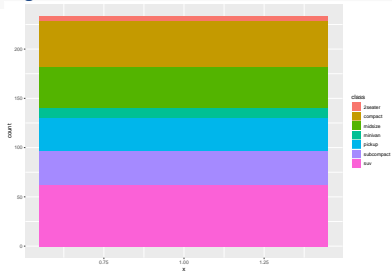


```
ggplot(mpg, aes(x = class, fill = class)) +  
  geom_bar() + coord_flip()
```

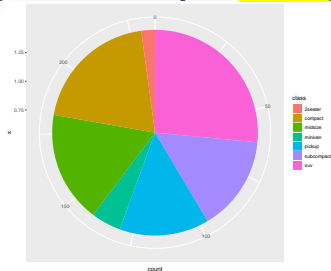


Pie-chart (coord_polar())

```
ggplot(mpg, aes(x = 1, fill = class)) +  
  geom_bar()
```



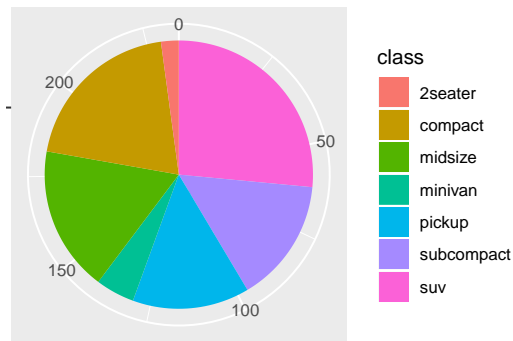
```
ggplot(mpg, aes(x = 1, fill = class)) +  
  geom_bar() + coord_polar(theta = "y")
```



Pie Chart (from M91-ggplot2-50examples.pdf)

```
ggplot(mpg, aes(x = "", fill = factor(class))) +  
  geom_bar(width = 1) +  
  theme(axis.line = element_blank(), plot.title = element_text(hjust=0.5)) +  
  labs(fill = "class", x = NULL, y = NULL,  
        title = "Pie Chart of class", caption = "Source: mpg") +  
  coord_polar(theta = "y", start=0)
```

Pie Chart of class



Source: mpg

ggplot 기타 기능 (저장 & plotly)

- ggplot객체를 png 파일로 저장

```
png("out_file.png")
ggplot(mpg) + geom_bar(aes(x=class)) +
  coord_flip()
dev.off()
```

여기
좌회전?
변경

- plotly - html에서 각종 추가 기능 제공

```
library(plotly)
a <- ggplot(mpg) + geom_bar(aes(x=class))
ggplotly(a)
```


Part 1. Text 전처리 - tm, KONLP

.

“I have a dream” - Martin Luther King, Jr.

Nulla. Source packages

```
library(tm) # for text mining
library(SnowballC) # for text stemming
library(wordcloud) # word-cloud generator
library(wordcloud2) # word-cloud generator
library(RColorBrewer) # color palettes
library(ggplot2)
library(dplyr)
```

I. Infile

```
# txt in web? no problem!
filePath <- paste0(
  "http://www.sthda.com/sthda/RDoc/",
  "example-files/martin-luther-king",
  "-i-have-a-dream-speech.txt")
text <- readLines(filePath)
docs <- Corpus(VectorSource(text))
inspect(docs)

## <<SimpleCorpus>>
## Metadata: corpus specific: 1, document lev
## Content: documents: 46
##
## [1]
## [2] And so even though we face the difficu
## [3]
## [4] I have a dream that one day this natio
## [5]
## [6] We hold these truths to be self-eviden
## [7]
## [8] I have a dream that one day on the red
## [9]
## [10] I have a dream that one day even the s
## [11]
## [12] I have a dream that my four little chi
## [13]
```

II. Cleanup text

```
toSpace <- content_transformer(  
  function(x, pattern) gsub(pattern, " ", x))  
docs <- docs %>%  
  tm_map(toSpace, "/") %>%  
  tm_map(toSpace, "@") %>%  
  tm_map(toSpace, "\\|")  
  
docs <- docs %>%  
  tm_map(content_transformer(tolower)) %>%      # Convert it to lower case  
  tm_map(removeNumbers) %>%                    # Remove numbers  
  tm_map(removeWords, stopwords("english")) %>% # Remove english common stopwords  
  tm_map(removeWords, c("blabla1", "blabla2")) %>% # Remove your own stop word  
  tm_map(removePunctuation) %>%               # Remove punctuations  
  tm_map(stripWhitespace)                     # Eliminate extra white spaces  
  
inspect(docs)  
  
## <<SimpleCorpus>>  
## Metadata: corpus specific: 1, document level (indexed): 0  
## Content: documents: 46  
##  
## [1]  
## [2] even though face difficulties today tomorrow still dream dream deeply rooted amer  
## [3]  
## [4] dream one day nation will rise live true meaning creed  
## [5]  
## [6] hold truths selfevident men created equal  
## [7]  
## [8] dream one day red hills georgia sons former slaves sons former slave owners will a  
## [9]  
## [10] dream one day even state mississippi state sweltering heat injustice sweltering h
```

III. Generate FreqTable

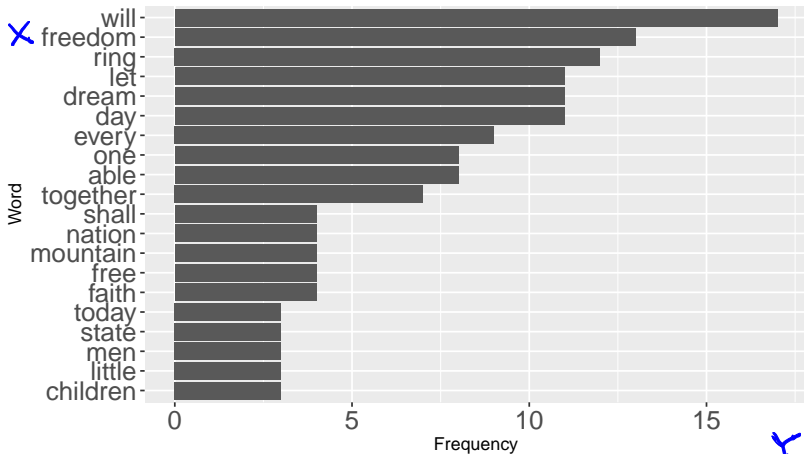
```
termMat    <- TermDocumentMatrix(docs)
termTable  <- as.matrix(termMat)
freqTable  <- data.frame(word = rownames(termTable),
                        freq = rowSums(termTable),
                        row.names = NULL)
freqTable  <- freqTable %>% arrange(desc(freq))

freqTable %>% head()
```

```
##      word freq
## 1    will   17
## 2 freedom   13
## 3    ring   12
## 4   dream   11
## 5    day   11
## 6    let   11
```

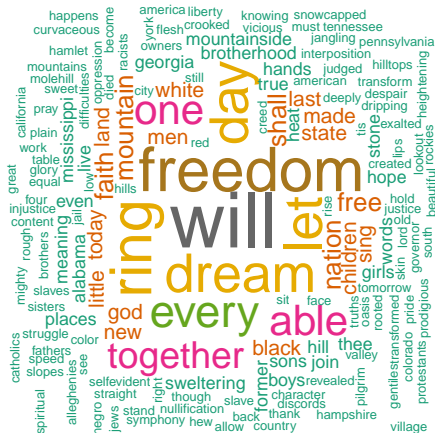
IV. Render barplot

```
ggplot(data = head(freqTable, 20)) +  
  geom_bar(aes(x = reorder(word, freq), y = freq), stat="identity") +  
  coord_flip() +  
  theme(axis.text = element_text(size = 16)) +  
  labs(x = "Word", y = "Frequency")
```



V. Render wordcloud

```
wordcloud(words = freqTable$word, freq = freqTable$freq,
  min.freq = 1, max.words=200, random.order=FALSE, rot.per=0.35,
  colors=brewer.pal(8, "Dark2"))
```



```
# for html, newer version is available!
```

```
# wordcloud2(freqTable, color = "random-light", backgroundColor = "grey")
```


Summary

0. Source packages
 1. Infile
 2. Cleanup text
 3. Generate **FreqTable**
 4. Render barplot
 5. Render wordcloud
- **Next step: sonaki.pdf**

sonaki.pdf

Nulla. Source packages

i-have-a-dream.txt

```
library(tm) # for text mining
library(SnowballC) # for text stemming
library(wordcloud) # word-cloud generator
library(wordcloud2) # word-cloud generator
library(RColorBrewer) # color palettes
library(ggplot2)
library(dplyr)
```

sonaki.pdf

```
library(tm) # for text mining
library(SnowballC) # for text stemming
library(wordcloud) # word-cloud generator
library(wordcloud2) # word-cloud generator
library(RColorBrewer) # color palettes
library(ggplot2)
library(dplyr)
```

```
library(KoNLP)
# Korean Natural Language Processing
library(pdftools)
# Extract text from pdf
```

I. Infile

i-have-a-dream.txt

```
# txt in web? no problem!
filePath <- paste0(
  "http://www.sthda.com/sthda/RDoc/",
  "example-files/", "martin-luther-",
  "king-i-have-a-dream-speech.txt")
text <- readLines(filePath)
docs <- Corpus(VectorSource(text))
inspect(docs)
```

sonaki.pdf

```
# local pdf
text <- pdf_text("../script/sonaki.pdf")
# Apply extract Noun
docs <- sapply(
  text, extractNoun, USE.NAMES = F)
docs <- unlist(docs)
# Character length >= 2
docs <- Filter(
  function(x) {nchar(x) >= 2}, docs)
head(docs): 소나기, 황순원, 소년,
개울가, 소녀, 증손녀(曾孫女)딸이라는
```

II. Cleanup text

i-have-a-dream.txt

```
toSpace <- content_transformer(  
  function(x, pattern) gsub(pattern, " ", x))  
docs <- docs %>%  
  tm_map(toSpace, "/" ) %>%  
  tm_map(toSpace, "@") %>%  
  tm_map(toSpace, "\\|")  
  
docs <- docs %>%  
  # Convert it to lower case  
  tm_map(content_transformer(tolower)) %>%  
  # Remove numbers  
  tm_map(removeNumbers) %>%  
  # Remove english common stopwords  
  tm_map(removeWords, stopwords("english")) %>%  
  # Remove your own stop word  
  tm_map(removeWords, c("blabla1", "blabla2")) %>%  
  # Remove punctuations  
  tm_map(removePunctuation) %>%  
  # Eliminate extra white spaces  
  tm_map(stripWhitespace)
```

sonaki.pdf

```
# `extractNoun` already  
# took care of...
```

III. Generate 'FreqTable'

i-have-a-dream.txt

```
termMat  <- TermDocumentMatrix(docs)
termTable <- as.matrix(termMat)
freqTable <-
  data.frame(word = rownames(termTable),
             freq = rowSums(termTable),
             row.names = NULL)
freqTable <- freqTable %>%
  arrange(desc(freq))
# head(freqTable)
```

sonaki.pdf

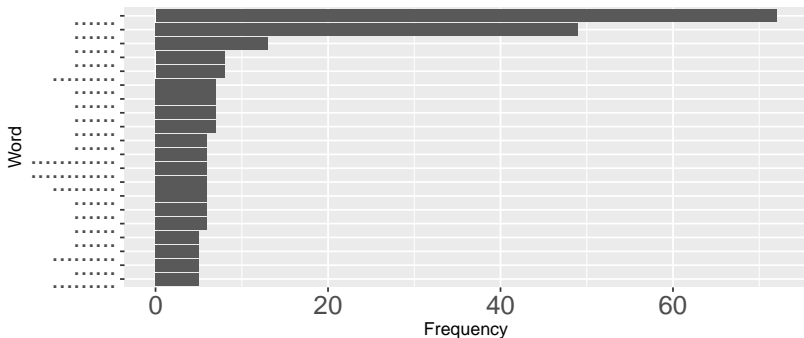
```
freqTable <- data.frame(table(docs))
names(freqTable) <- c("word", "freq")
freqTable <- freqTable %>%
  arrange(desc(freq))
# head(freqTable)
```

IV. Render barplot

i-have-a-dream.txt

sonaki.pdf

```
ggplot(data = head(freqTable, 20)) +  
  geom_bar(aes(x = reorder(word, freq), y = freq), stat="identity") +  
  coord_flip() +  
  theme(axis.text = element_text(size = 16)) +  
  labs(x = "Word", y = "Frequency")
```

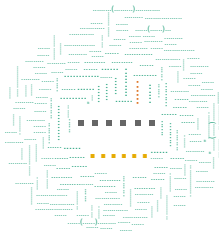


V. Render wordcloud

i-have-a-dream.txt

sonaki.pdf

```
wordcloud(words = freqTable$word, freq = freqTable$freq,  
  min.freq = 1, max.words=200, random.order=FALSE, rot.per=0.35,  
  colors=brewer.pal(8, "Dark2"))
```



Summary

- txt vs pdf
- English vs Korean

Part 3. 일반화 - Writing a function.

function getDocs2

0. Source packages & 1. Infile

```
getDocs2 <- function(fileName) {  
  # 0. Source packages  
  activate("tm", "SnowballC", "wordcloud", "KoNLP", "pdftools")  
  activate("ggplot2", "dplyr", "RColorBrewer", "cld3")  
  # 1. Infile  
  fileType <- unlist(strsplit(fileName, split = "\\.")) %>% tail(1)  
  if (fileType == "pdf") {  
    text <- pdf_text(fileName)  
  } else if (fileType == "txt") {  
    text <- readLines(fileName)  
  } else {  
    stop("We only support pdf and txt!") # defensive programming  
  }  
  isKO <- "ko" %in% detect_language(text)  
  if (isKO) { # if Korean  
    docs <- sapply(text, extractNoun, USE.NAMES = F) %>% unlist()  
    docs <- Filter(function(x) {nchar(x) >= 2}, docs)  
    attr(docs, "lang") <- "kr"  
  } else { # if English  
    docs <- Corpus(VectorSource(text))  
    attr(docs, "lang") <- "en"  
  }  
  return(docs)  
}
```

function cleanDocsGenerateFreqTable

2. Cleanup text & 3. Generate FreqTable

```
cleanDocsGenerateFreqTable <- function(docs, lang) {  
  if (lang == "kr") {  
    docs <- unlist(docs)  
    docs <- Filter(function(x) {nchar(x) >= 2}, docs) # Character length >= 2  
    freqTable <- data.frame(table(docs))  
    names(freqTable) <- c("word", "freq")  
    freqTable <- freqTable %>% arrange(desc(freq))  
  } else { # lang == "en"  
    toSpace <- content_transformer(function (x , pattern) gsub(pattern, " ", x))  
    docs <- docs %>% tm_map(toSpace, "/") %>%  
      tm_map(toSpace, "@") %>% tm_map(toSpace, "\\|")  
    docs <- docs %>%  
      tm_map(content_transformer(tolower)) %>% # Convert it to lower case  
      tm_map(removeNumbers) %>% # Remove numbers  
      tm_map(removeWords, stopwords("english")) %>% # Remove english common stopwords  
      tm_map(removeWords, c("blabla1", "blabla2")) %>% # Remove your own stop word  
      tm_map(removePunctuation) %>% # Remove punctuations  
      tm_map(stripWhitespace) %>% # Eliminate extra white spaces  
    termMat <- TermDocumentMatrix(docs)  
    termTable <- as.matrix(termMat)  
    freqTable <- data.frame(word = rownames(termTable), freq = rowSums(termTable))  
    freqTable$word <- rownames(freqTable)  
    freqTable <- freqTable %>% arrange(desc(freq))  
  }  
  return(freqTable)  
}
```

Source File 관리

- 위의 두 함수 `getDocs2`와 `cleanDocsGenerateFreqTable`를 `LSR.R`파일에 저장
- 다른 곳에서 `source("LSR.R")`을 한번 실행하면 함수가 메모리에 정의되어 있으므로 두 함수를 사용이 가능함
- 수업 중 프로그램 단위인 MXX의 폴더에서는 상위 폴더로 두 번 올라가면 `LSR.R`이 있으므로 `source("../../LSR.R")`은 경로로 실행

LS-R

파일 홈 공유 보기

← → ↶ ↷

내 PC > Windows (C:) > LS-R

이름	수정한 날짜	유형	크기
_study	2018-05-21 오후 3:42	파일 폴더	
Admin	2018-05-20 오후 11:	파일 폴더	
Ch1	2018-06-23 오후 8:19	파일 폴더	
Ch2	2018-07-01 오전 12:	파일 폴더	
Ch3	2018-07-07 오후 2:09	파일 폴더	
Ch4	2018-05-20 오후 11:	파일 폴더	
Ch5	2018-05-19 오후 11:	파일 폴더	
Ref-external	2018-05-20 오후 10:	파일 폴더	
Ref-internal	2018-06-10 오후 1:02	파일 폴더	
Ref-rmdTemplates	2018-06-10 오후 1:03	파일 폴더	
Students	2018-07-01 오전 1:16	파일 폴더	
.Rhistory	2018-07-07 오후 5:38	RHISTORY 파일	0KB
LSR	2018-05-23 오후 6:28	R 파일	9KB

★ 바로 가기

- 바탕 화면
- 다운로드
- 문서
- 사진
- 휴지통
- O_UF_draft[2]
- LS-R
- M31-ggplot_wordc
- M32-textmining
- OneDrive

activate - LSR.R내의 다른 function(1)

- 패키지가 인스톨 되어있지 않으면 인스톨 한후에,
- 패키지가 로드되어 있지 않다면 로드함.
- 주의. shiny에서는 실행하지 말 것

```
activate <- function(...) {  
  # activate("dplyr")  
  # activate("xts", "dygraphs")  
  # activate(c("ggplot2", "ISLR"))  
  packages <- unlist(list(...))  
  for (i in 1:length(packages)) {  
    package_i <- packages[i]  
    if (!package_i %in% installed.packages()[,"Package"]) {  
      print(paste("Installing", package_i))  
      suppressMessages(install.packages(package_i))  
    }  
    suppressMessages(require(package_i, character.only = TRUE))  
  }  
}
```

`rndQuote` - LSR.R내의 다른 function(2)

- `ranjith19`라는 github 사용자가 올려놓은 명언들 중에서
- 무작위로 몇개를 선택하여 돌려주는 함수
- 매일 자동 실행되는 프로그램에서 사용하기에 유용함

```
rndQuote <- function(n = 1) {  
  # rndQuote() # 2018-05-09  
  fileName <- paste0(  
    "https://raw.githubusercontent.com/ranjith19/",  
    "random-quotes-generator/master/quotes.txt")  
  activate("readr")  
  quotes <- read_file(fileName)  
  quotes <- gsub("\n.\n", "zzzzzz", quotes)  
  quotes <- unlist(strsplit(quotes, split="zzzzzz"))  
  return(quotes[sample(length(quotes), n)])  
}
```

```
cat(rndQuote(1))
```

```
## "All our knowledge merely helps us to die a more painful death  
##  than animals that know nothing." -- Maurice Maeterlinck.
```

```
cat(rndQuote(1))
```

```
## "Hofstadter's Law: It always takes longer than you expect, even  
##  when you take into account Hofstadter's Law."
```