워드 클라우드

문장안에서 단어들의 중요도나 빈도수를 색상/크기 등으로 표현한 시각화 기법을 워드 클라우드라고 합니다.

문장에 포함된 단어들을 분석하기 위해서는 **의미를 갖는 최소 단위로 쪼개**는 형태소 분석이라는 처리가 필요한데, 형태소 분석에 손쉽게 사용할 수 있는 패키지로는 **KoNLP**가 있습니다.

> KoNLP의 경우 Java 1.8 환경에 대한 설치와 환경변수 설정이 선행되어야 합니다.

#01. 필요한 패키지 설치

```
# 패키지 설치
REPO_URL = "https://cran.seoul.go.kr/"
if (!require(dplyr))     install.packages("dplyr", repos=REPO_URL)
if (!require(tidyverse)) install.packages("tidyverse", repo=REPO_URL)
if (!require(stringr))   install.packages("stringr", repos=REPO_URL)
if (!require(rJava))     install.packages("rJava", repos=REPO_URL)
if (!require(memoise))   install.packages("memoise", repos=REPO_URL)
if (!require(KoNLP))     install.packages("KoNLP", repos=REPO_URL)
if (!require(wordcloud)) install.packages("wordcloud", repos=REPO_URL)
if (!require(extrafont)) install.packages("extrafont", repos=REPO_URL)

# 패키지 로드
library(dplyr)
library(readr)        # 파일 읽기 기능 제공 (tidyverse패키지에 포함됨)
library(stringr)      # 문자열 관련 기능 제공 패키지
library(rJava)        # KoNLP가 의존함 (Java기능 호출 패키지)
library(memoise)      # KoNLP가 의존함
library(KoNLP)        # 한글데이터 형태소 분석 패키지 (이름 대소문자 주의)
library(wordcloud)    # 워드클라우드 생성 패키지
library(RColorBrewer) # 색상 제어 패키지
library(extrafont)    # 폰트관리 패키지
```

#02. 사용할 폰트 확인하기

- 한글을 사용하기 위해서는 반드시 폰트를 확인하고 지정해야 한다.
- 그래프를 표시하기 위해 확인했던 폰트 이름을 그대로 사용해도 된다.

1) 폰트 스캔

- 시간이 매우 오래 걸린다. 이 작업은 각 시스템별로 1회만 수행한다.
- 만약 이전 예제들을 통해 그래프를 만드는 과정에서 아래의 코드를 수행한 적이 있다면 생략한다.

나눔고딕 폰트 가져오기

```
# 시스템의 폰트 파일들중 이름에 D2라는 단어가 포함된 폰트를 R 디렉토리 안으로 복사한다.
# --> 오랜 시간이 걸리더라도 수행할지 여부를 묻는 "y/n" 확인이 필요하면 "y"를 입력 후 엔터
font_import(pattern="NanumGothic.ttf")
# 폰트 로드 --> 운영체제에 맞게 설정하세요.
loadfonts(device="win")       # Windows
#loadfonts()  # Mac

# 이 메시지가 출력될 때 까지 다음을 진행하지 마세요.
print("----------- 폰트스캔 완료 -----------");
```

▶ 출력결과

```
Importing fonts may take a few minutes, depending on the number of fonts and the speed of the system.
Continue? [y/n]  y
```

```
Scanning ttf files in C:\Windows\Fonts ...

Extracting .afm files from .ttf files...

C:\Windows\Fonts\NanumGothic.ttf
 : NanumGothic already registered in fonts database. Skipping.

Found FontName for 0 fonts.

Scanning afm files in C:/Users/leekh/Documents/R/win-library/3.6/extrafontdb/metrics

NanumGothic already registered with windowsFonts().


[1] "---------- 폰트스캔 완료 ----------"
```

폰트 확인

```
# 폰트테이블 확인
fonts <- fonttable()
# 중복된 이름을 제거하고 출력
unique(fonts$FamilyName)
```

▶ 출력결과

```
'NanumGothic'
```

#02. 텍스트 마이닝

1) 사용할 데이터 파일 읽기

tidyverse 패키지에 포함되어 있는 **readr** 라이브러리를 사용한다.

> 2020년 01월 11일 현재 R 3.6.2 버전 이하로는 tidyverse 패키지가 설치되지 않는 문제가 있었습니다.

```
# 원격지 텍스트 파일 읽기(readr 패키지 사용)
txt <- read_file("http://itpaper.co.kr/demo/r/korea.txt")
# 500글자만 추출하여 확인
substr(txt, 0, 500)
```

▶ 출력결과

```
'대한민국헌법\r\n[시행 1988. 2. 25] [헌법 제10호, 1987. 10. 29, 전부개정]\r\n\r\n유구한 역사와 전통에 빛나는 우리 대한국민은 3
```

2) 형태소 분석

명사만 추출하기

사전 초기화

```
useNIADic()
```

▶ 출력결과

```
Backup was just finished!


WARNING: Rtools is required to build R packages, but is not currently installed.

Please download and install Rtools custom from http://cran.r-project.org/bin/windows/Rtools/.

Downloading package from url: https://github.com/haven-jeon/NIADic/releases/download/0.0.1/NIADic_0.0.1.tar



stringi (NA    -> 1.4.5) [CRAN]
cli     (1.1.0 -> 2.0.1) [CRAN]
fansi   (0.4.0 -> 0.4.1) [CRAN]
pillar  (1.4.2 -> 1.4.3) [CRAN]


Installing 4 packages: stringi, cli, fansi, pillar

Installing packages into 'C:/Users/leekh/Documents/R/win-library/3.6'
(as 'lib' is unspecified)

... 생략 ...
```

명사 추출

대한민국 헌법 전문에서 명사만 추출한다. (약간의 시간 소요됨)

추출 결과는 `character` 타입의 리스트가 된다.

```
nouns <- extractNoun(txt)
nouns
```

▶ 출력결과

```
1. '대한'
2. '민국'
3. '헌법'
4. '시행'
5. '1988.'
... 생략 ...
```

단어별 빈도표 만들기

추출 결과에 `unlist()` 함수를 적용하여 리스트를 해제하여 이를 다시 `table()` 함수로 전달하여 단어별 빈도표를 생성하고 단어와 빈도수로 구성된 데이터 프레임으로 변환한다.

데이터 프레임 변환시 문자열 데이터를 요인으로 변경하는 것이 기본값이므로 요인으로 변경을 차단하기 위해서 stringsAsFactors=FALSE 옵션이 필요하다.

```
wordcount <- table(unlist(nouns))
df_word <- as.data.frame(wordcount, stringsAsFactors = FALSE)
head(df_word, 20)
```

▶ 출력결과

A data.frame: 20 × 2

Var1	Freq
<chr>	<int>
제10호	1
1	31

Var1	Freq
<chr>	<int>
10	2
10.	2
100	1
11	2
12	2
13	1
14	1
15	3
16	1
17	1
1948	1
1987.	2
1988	1
1988.	1
1각급	1
1감사원은	1
1공무원	1
1공무원은	1

#03. 데이터 전처리

1) 데이터프레임의 컬럼명 변경

```
df_word <- rename(df_word, 단어=Var1, 빈도수=Freq)
head(df_word, 20)
```

▶ 출력결과

A data.frame: 20 × 2

단어	빈도수
<chr>	<int>
제10호	1
1	31
10	2
10.	2
100	1
11	2
12	2
13	1

단어	빈도수
<chr>	<int>
14	1
15	3
16	1
17	1
1948	1
1987.	2
1988	1
1988.	1
1각급	1
1감사원은	1
1공무원	1
1공무원은	1

2) 두 글자 이상 단어 추출하기 + 역순정렬

분석가의 주관적인 판단으로 두 글자 이상인 단어만 분석하기로 결정했다고 가정한다.

```
# 두 글자 이상 단어 추출 > 역순정렬
result_df <- df_word %>% filter(nchar(단어) >= 2) %>% arrange(desc(빈도수))
result_df
```

▶ 출력결과

A data.frame: 1210 × 2

단어	빈도수
<chr>	<int>
법률	116
국민	57
국회	53
대통령	46
헌법	42
국가	36
필요	30
기타	26
사항	23
정부	22
권리	20
2국	19
임명	19
자유	19

단어	빈도수
<chr>	<int>
의무	18
이상	18
경우	17
의결	17
보장	16
의원	16
임기	16
법원	14
재적	14
노력	13
과반수	12
보호	12
시행	12
요구	12
찬성	12
공무원	11
…	…
피고인	1
피선거권	1
피해자	1
학문	1
한계	1
한반도와	1
합리	1
항거	1
항구적	1
항해	1
해당	1
행정	1
행정부	1
혁신	1
형사피의자	1
형태	1
호선	1
확립	1
확보	1

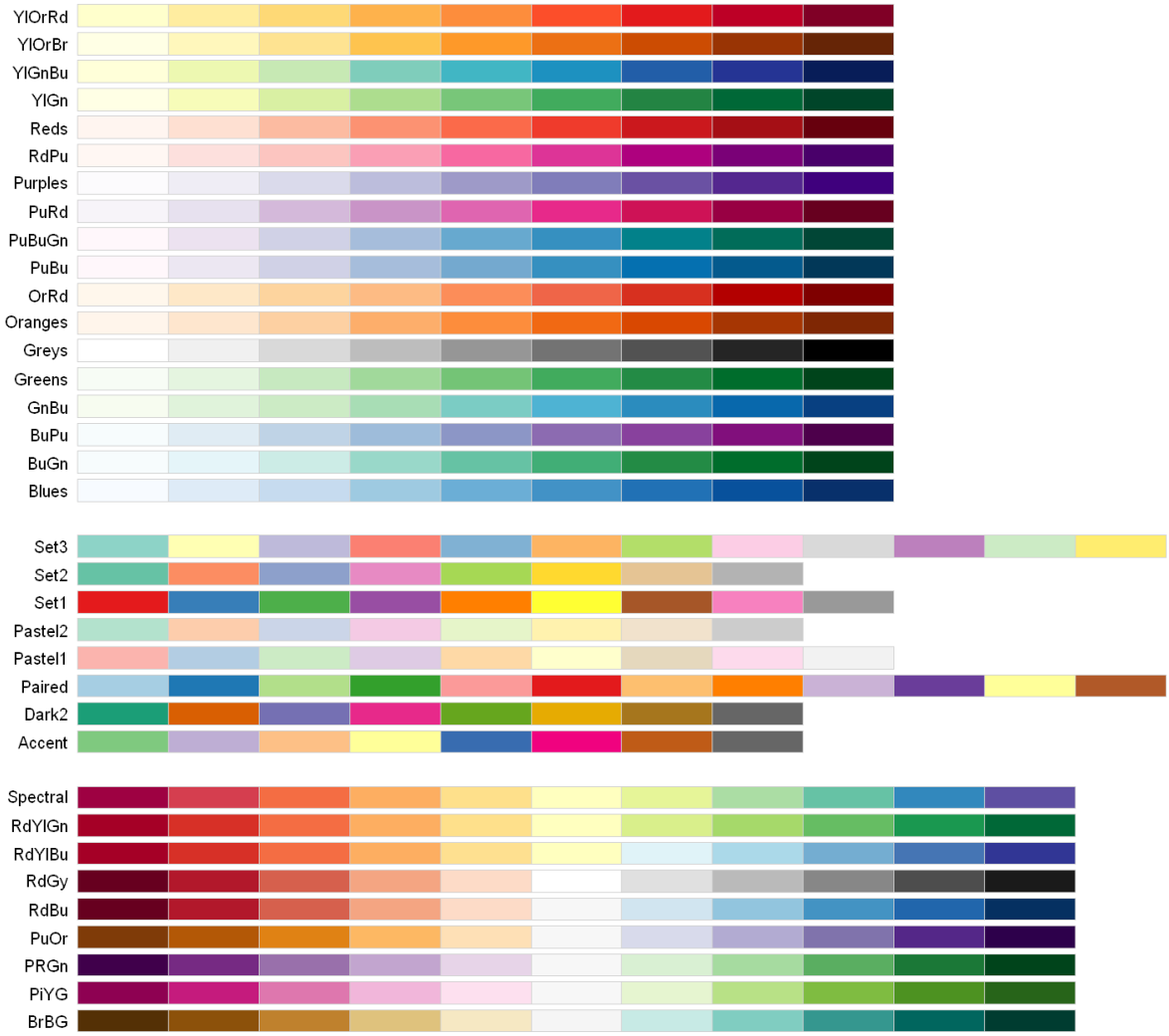
단어	빈도수
<chr>	<int>
확인	1
환경	1
환경보전	1
환부	1
회계	1
회복	1
회부	1
획정	1
효율	1
후보자	1
훈장	1

#04) 데이터 시각화

사용 가능한 칼라 팔레트 확인

```
options(repr.plot.width=13, repr.plot.height=13, warn=-1)
display.brewer.all()
```

▶ 출력결과



2) 사용할 팔레트 지정

팔레트가 지원하는 색상수 내에서 몇 개의 색상을 뽑아올지 지정.

지정된 값이 지원하는 수보다 클 경우 지원되는 색상만큼만 반환된다.

```
pal <- brewer.pal(10,"Dark2")
pal
```

▶ 출력결과

```
1. '#1B9E77'
2. '#D95F02'
3. '#7570B3'
4. '#E7298A'
5. '#66A61E'
6. '#E6AB02'
7. '#A6761D'
8. '#666666'
```

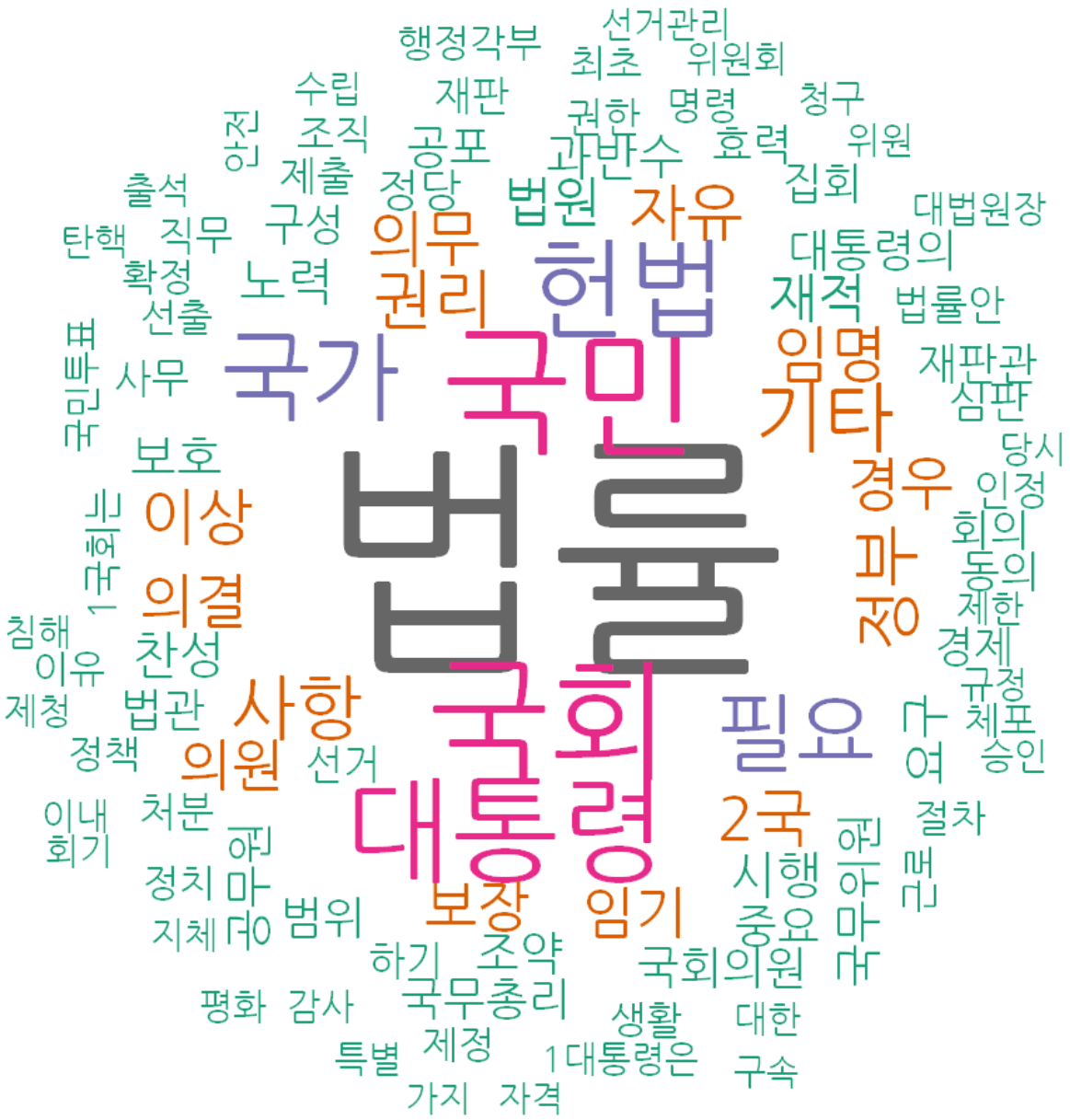
3) 워드 클라우드 만들기

```
# 그래픽 사이즈 설정
options(repr.plot.width=8, repr.plot.height=8, warn=-1)

# 랜덤값 고정 -> 실행시마다 동일한 모양으로 생성되도록 함
set.seed(1234)

# 워드클라우드 생성
wordcloud(words = result_df$단어,     # 단어
          freq = result_df$빈도,      # 빈도
          min.freq = 3,               # 최소 단어 빈도
          max.words = 100,            # 표현 단어 수
          random.order = FALSE,       # 고빈도 단어 중앙 배치
          random.color = FALSE,       # 색상으로 빈도 표현 여부
          scale = c(10, 1),           # 단어 크기 범위
          colors = pal,               # 색깔 목록
          family="NanumGothic")       # 사용할 폰트
```

▶ 출력결과



4) 팔레트 내에서 일부 색상들만 사용하기

팔레트 선정

```
blues_pal <- brewer.pal(9,"Blues")
blues_pal
```

▶ 출력결과

```
1. '#F7FBFF'
2. '#DEEBF7'
3. '#C6DBEF'
4. '#9ECAE1'
5. '#6BAED6'
6. '#4292C6'
7. '#2171B5'
8. '#08519C'
9. '#08306B'
```

선정된 팔래트에서 특정 위치의 색상값들만 추출

```
# 가져온 색상 목록에서 5번째부터 9번째 까지의 색상만 사용
mypal <- blues_pal[5:9]
mypal
```

▶ 출력결과

```
1. '#6BAED6'
2. '#4292C6'
3. '#2171B5'
4. '#08519C'
5. '#08306B'
```

워드클라우드 생성하기

> 앞 코드를 재사용하면서 색상값을 위한 colors 파라미터만 수정함

```
# 그래픽 사이즈 설정
options(repr.plot.width=8, repr.plot.height=8, warn=-1)

# 랜덤값 고정 -> 실행시마다 동일한 모양으로 생성되도록 함
set.seed(1234)

# 워드클라우드 생성
wordcloud(words = result_df$단어,     # 단어
          freq = result_df$빈도,      # 빈도
          min.freq = 3,               # 최소 단어 빈도
          max.words = 100,            # 표현 단어 수
          random.order = FALSE,       # 고빈도 단어 중앙 배치
          random.color = FALSE,       # 색상으로 빈도 표현 여부
          scale = c(10, 1),           # 단어 크기 범위
          colors = mypal,             # 색깔 목록
          family="NanumGothic")       # 사용할 폰트
```

▶ 출력결과

#05. wordcloud2 패키지 사용하기

R의 기본 wordcloud 패키지를 시각적으로 보완한 패키지.

1) 패키지 설치

```
REPO_URL = "https://cran.seoul.go.kr/"
if (!require(wordcloud2))     install.packages("wordcloud2", repos=REPO_URL)
library(wordcloud2)
```

▶ 출력결과

```
Loading required package: wordcloud2

Installing package into 'C:/Users/leekh/Documents/R/win-library/3.6'
(as 'lib' is unspecified)



package 'wordcloud2' successfully unpacked and MD5 sums checked
```

```
The downloaded binary packages are in
        C:\Users\Public\Documents\ESTsoft\CreatorTemp\RtmpqIIG1l\downloaded_packages
```

2) 워드클라우드 생성하기

기본 사용 방법

`head()` 함수를 사용하여 상위 100건만 표시하도록 지정함.

```
wordcloud2(data = head(result_df, 100), fontFamily = '나눔고딕')
```

▶ 출력결과

옵션 지정하기

```
wordcloud2(data = head(result_df,100),     # 데이터프레임
           fontFamily = '나눔고딕',         # 사용할 글꼴
           fontWeight = 'normal',          # 글꼴의 굵기 (normal or bold)
           size = 1.3,                     # 글꼴크기(기본값=1)
           minSize= 0.3,                   # 글꼴의 최소 크기
           backgroundColor = "#ffffff",    # 배경색상
           widgetsize = c(800, 600),       # 위젯의 크기 (가로,세로) 픽셀 형식의 벡터
           color=brewer.pal(11, "RdYlGn") # 색상 팔레트 적용 (단일 값인 경우 단색으로 지정됨)
          )
```

▶ 출력결과

법률
국민
국회
대통령
헌법
국가
필요
기타
사항
권리
정부
의무
이상
임명
재적
요구
노력
보장
과반수
자유
임기
국회의원
대통령의
하기
처분
탄핵
심판
중요
경우
보호
제출
규정
감사
조직
정책
출석
재판관
집회
범위
동의
제청
직무
지체
회기
시행
의결
조약
재판
국무총리
법원
찬성
대법원장
의원
절차
1대통령은
공무원
국무위원
위원
3대통령은
공개
평화
침해
인정
효력
특별
국민투표
청구
위원회
선거
이내
구성
정당
안전
대한
선출
개시
제한
생활
정치
최초수립
사무
1국회는
제정
확정
승인
2국
공포
법관
경제
명령
이유
체포
행정각부
가지
당시
회의
권한
법률안
근로