Forbes 'MOST POPULAR'

Data Mining

https://github.com/Trigger21

Forbes Site

Forbes

MOST POPULAR SEE WHAT'S TRENDING ON FORBES

Editors' Picks

New Post 604,926 views

Department Of Homeland Security Compiling Database Of Journalists And 'Media Influencers'

New Post 22,508 views

Five Questions You Can Ask Instead Of 'How Are You?'

New Post 6,928 views

A-Rod In Paradise: Swinging For Redemption Through Baseball And Business

From the issue

#SportsMoney

6,928 views

A-Rod In Paradise: Swinging For Redemption Through Baseball And Business

From the issue

45,104 views

John Cena Overtakes Brock Lesnar As WWE's Highest-Paid Wrestler

8,852 views

Zidane Says No Guard-Of-Honor But Real Madrid Do Barcelona A Huge Favor Anyway

#NewTech

2,018 views

Industry 4.0 Explained By A Leading Venture Capitalist

21,258 views

Spotify's Market Cap Shows Just How Powerful Apple Really Is

15,224 views

Best Home Printers To Buy: Inkjet Vs Laser

- Index

- Crawling
- Data Refined
- WordCloud

1. Crawling

- **Selenium을 이용한 Crawling**
- **Selenium을 선택한 이유**

  : Forbes site는 동적인 주소를 사용하고 있고, 각 페이지 이동 시 버튼을 눌러, 이동해야 하기 때문에 Selenium 패키지를 사용함.

- **사용 방법 : 다음 슬라이드에서 안내**

1. Crawling(Selenium)

```
library(RSelenium)                                          #라이브러리 실행
Ch=wdman::chrome(port=4567L)                                #변수에 크롬포트 설정

remDr=remoteDriver(port=4567L, browserName='chrome')        #리모트 드라이브 실행
remDr$open()                                                #크롬드라이버 실행
remDr$navigate(https://www.forbes.com)                      #Forbes Site 이동

webElem0 <- remDr$findElement(using = "css selector", "div.continue-button")
webElem0$clickElement()                                     #객체(버튼) 찾은 후 클릭

source<-remDr$getPageSource()[[1]]                          #페이지 소스 가져오기
html <- read_html(source)                                   #html  Read
url <- html_nodes(html, css ="a.ng-scope > h2 > span")%>%    #내용 부분 찾기
html_text()
```

2. Data Refined

- 크롤링한 데이터를 정제
  1. 특수기호 삭제
  2. 특정문제 치환
  3. 불용어 추출

```
issue1 <- VCorpus(VectorSource(x))
issue2 <- TermDocumentMatrix(issue1)
m1 <- as.matrix(issue2)
rowSums(m1)

issue3 <- tm_map(issue1, stripWhitespace)  #공백제거
issue3 <- tm_map(issue3, tolower)              #모든문자 소문자 변환
issue3 <- tm_map(issue3, removeNumbers) #숫자제거
issue3 <- tm_map(issue3, removePunctuation) #구두점제거
issue3 <- tm_map(issue3, PlainTextDocument) #문자 빈도수 추출 후 행렬에 저장

tostring <- content_transformer(function(x,from,to) gsub(from,to,x))
issue3 <- tm_map(issue3, tostring, "”", "")
issue3 <- tm_map(issue3, tostring, "“", "")
issue3 <- tm_map(issue3, tostring, "’s", "")
issue3 <- tm_map(issue3, tostring, "’ll", "")
issue3 <- tm_map(issue3, tostring, "’ve", "")
issue3 <- tm_map(issue3, tostring, "’re", "")

sword2 <- c(stopwords("en"), "end", "but", "not")
issue3 <- tm_map(issue3, removeWords, sword2)  # 불용어 제거(전치사, 관사...)
issue4 <- TermDocumentMatrix(issue3)

m2 <- as.matrix(issue4)
issue5 <- sort(rowSums(m2), decreasing = T)
```

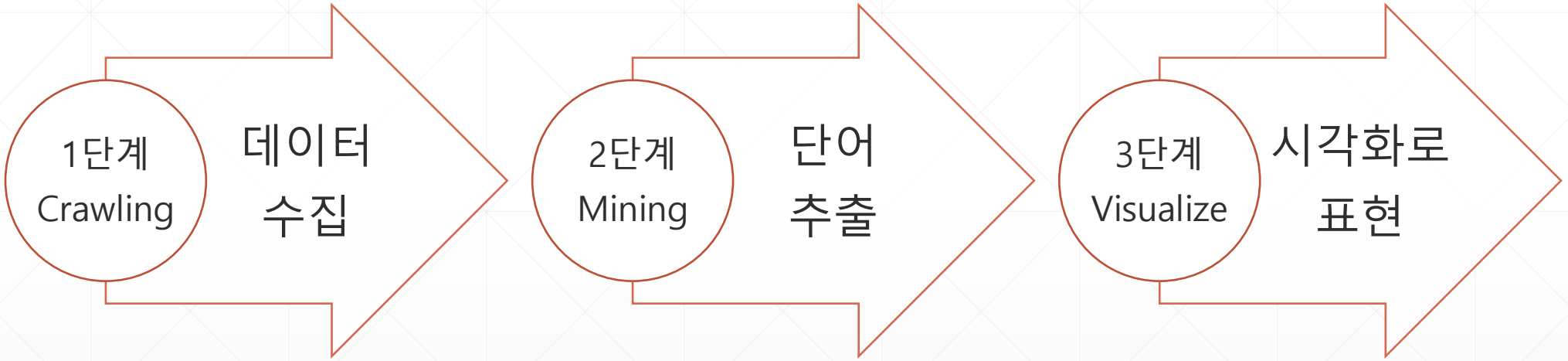
3. WordCloud

- 마이닝한 단어로 워드 클라우드 생성
- 워드 클라우드 패키지 선언

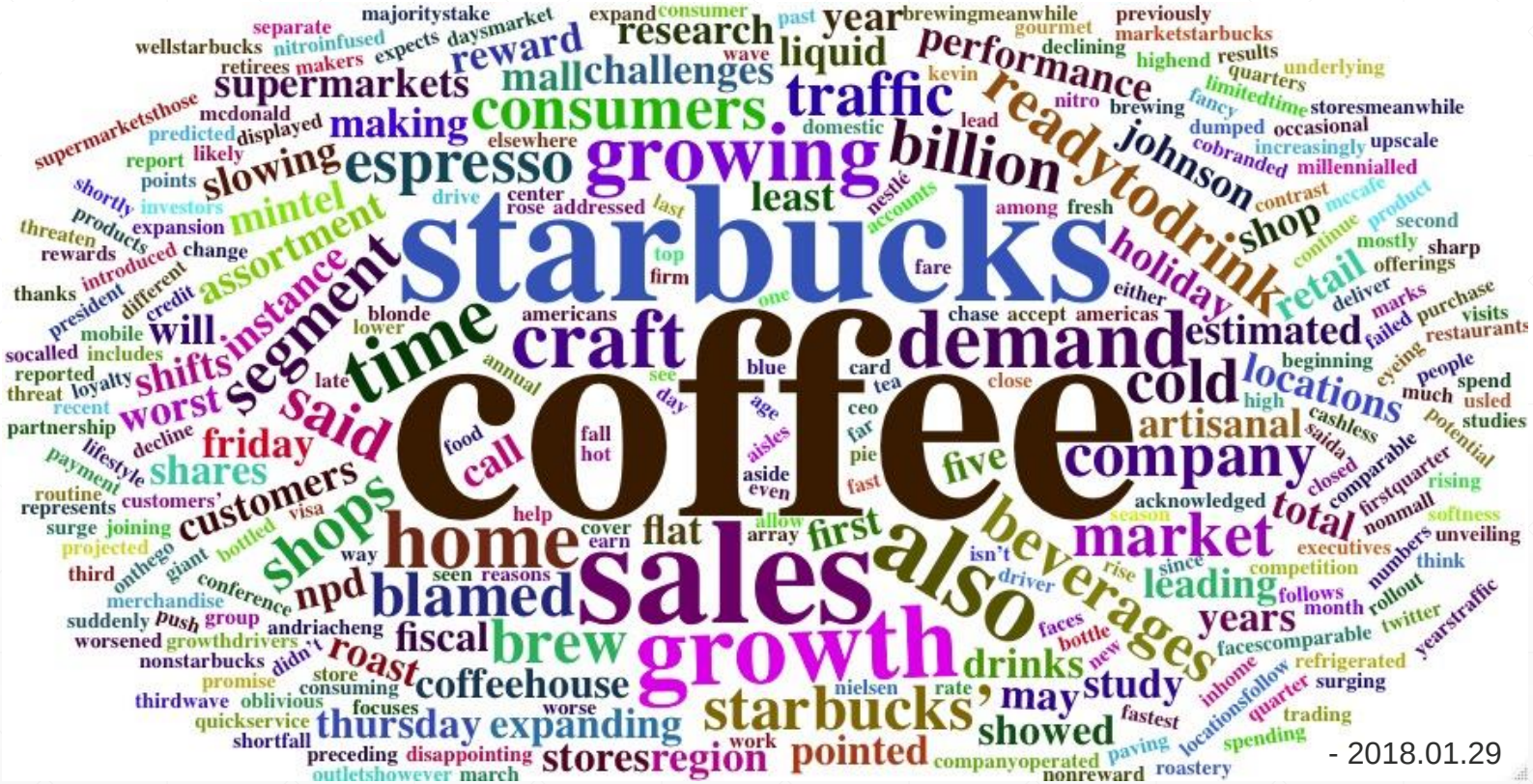
  : library(wordcloud2)
- 함수 사용

  : wordcloud2(as.table(issue5)) ※함수(테이블(정제된 데이터))

Data Maining Flow



- Wordcloud Image



감사합니다.