R 肉<89>苑<8c> 뜚釉뮉 沅<8e>뜖諭벥 苑<95>뵠甕<84> 뤀뮞몴 럲寃뺞에끇彛<85>(뮞寃<95>삋釉) 釉뮉 獄쎻뫖苡

hkim

2018-11-24

***preface*** (last update: 2018.11.24) 이번 포스트에서는 R을 이용하여 네이버 뉴스에서 원하는 키워드의 검색 결과를 웹크롤링(스크래핑) 하는 방법에 대하여 설명합니다.

# Naver News Web Scraping using Keywords in R

다음 자료를 참고하였습니다:  
- <https://blog.naver.com/knowch/221060289410>

먼저 필요한 패키지를 설치합니다. 이번 가이드에서 사용하는 패키지는 rvest 와 dplyr 입니다.

## 패키지 설치하기   
install.packages("rvest")  
install.packages("dplyr")

필요한 패키지를 불러옵시다.

## 패키지 불러오기  
library(rvest)

## Loading required package: xml2

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

## 웹크롤링을 위한 웹주소 리스트 만들기

웹크롤링 하기 위해서는 웹크롤링의 대상이 되는 웹페이지 주소의 구조를 이해해야 합니다. 네이버 뉴스에서 ’미국 기준금리’라는 키워드로 검색하는 경우를 예로 들어 보겠습니다.

먼저 [네이버 뉴스](https://news.naver.com/) 우측 상단에 있는 뉴스 검색에서 ’미국+기준금리’를 입력합니다. 날짜도 지정해줍시다. 검색시작일자와 검색종료일자를 각각 2018년 11월 19일로 설정합니다. 뉴스들이 1페이지에서 끝나지 않습니다. 맨 아래에 있는 페이지 숫자를 이리저리 클릭해봅시다. 주소창에서 &start= 뒤에 있는 숫자가 10단위로 움직이는 것을 확인할 수 있습니다. 다시 1페이지로 돌아오면 다음과 같은 주소를 얻을 수 있습니다.

https://search.naver.com/search.naver?&where=news&query=미국%2B기준금리&sm=tab\_pge&sort=0&photo=0&field=0&reporter\_article=&pd=3&ds=2018.11.19&de=2018.11.19&docid=&nso=so:r,p:from20181119to20181119,a:all&mynews=0&cluster\_rank=67&start=1&refresh\_start=0

위 웹 주소를 하나씩 뜯어보면 아래와 같이 구성되어 있다는 것을 확인할 수 있습니다.

https://search.naver.com/search.naver?  
&where=news  
&query=미국%2B기준금리  
&sm=tab\_pge  
&sort=0  
&photo=0  
&field=0  
&reporter\_article=  
&pd=3  
&ds=2018.11.19  
&de=2018.11.19  
&docid=  
&nso=so:r,p:from20181119to20181119,a:all&mynews=0  
&cluster\_rank=67  
&start=1  
&refresh\_start=0

우리에게 필요한 부분은 &query=미국+기준금리 , &ds=2018.11.19 , &de=2018.11.19 , &start=1 입니다. 이 부분들을 포함해서 검색 결과가 표시되는 웹 주소를 만들어 봅시다. 저는 여러번의 try and error 를 거쳐 아래 주소를 입력하면 우리가 원하는 결과를 얻을 수 있다는 것을 확인했습니다.

https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=1

다시 말해, 아래와 같은 주소를 생성하면 원하는 결과를 얻을 수 있습니다.

https://search.naver.com/search.naver?&where=news&query= [검색키워드] &pd=3&ds= [검색시작날짜] &de= [검색종료날짜] &start= [게시물번호]

날짜·쿼리 등을 원하는대로 바꾸거나, 게시물번호를 11 이나 21 로 바꿔도 원하는 결과를 얻을 수 있습니다.

이러한 구조를 이용하여, 문자열(string)을 결합하여 R 에서 사용할 수 있는 형태로 웹 주소를 만들어 봅시다. 여기에서는 R 에서 문자열을 결합하는 함수로 paste0() 을 사용합니다.

## 변수 입력하기  
QUERY <- "미국+기준금리" # 검색키워드  
DATE <- as.Date(as.character(20181119),format="%Y%m%d") # 검색시작날짜 & 검색종료날짜  
DATE <- format(DATE, "%Y.%m.%d")  
PAGE <- 1  
  
naver\_url\_1 <- "https://search.naver.com/search.naver?&where=news&query="  
naver\_url\_2 <- "&pd=3&ds="  
naver\_url\_3 <- "&de="  
naver\_url\_4 <- "&start="  
  
naver\_url <- paste0(naver\_url\_1,QUERY,naver\_url\_2,DATE,naver\_url\_3,DATE,naver\_url\_4,PAGE)  
naver\_url

## [1] "https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=1"

그럼 원하는 주소의 리스트를 만들어 봅시다. 날짜는 11월 19일부터 21일까지 3일간, 게시물은 첫 5 페이지를 살펴보는 것으로 제한하겠습니다.

먼저 날짜 리스트를 만들어 보겠습니다.

## 날짜 리스트 만들기  
DATE\_START <- as.Date(as.character(20181119), format="%Y%m%d") # 시작일자  
DATE\_END <- as.Date(as.character(20181121), format="%Y%m%d") # 종료일자  
DATE <- DATE\_START:DATE\_END  
DATE <- as.Date(DATE,origin="1970-01-01")  
DATE

## [1] "2018-11-19" "2018-11-20" "2018-11-21"

다음은 게시물 번호 리스트를 만들어 보겠습니다. 게시물 번호는 1, 2, 3 으로 증가하는 것이 아니라, 1, 11, 21 로 10씩 증가해야합니다. seq() 함수를 이용하면 이와 같은 연산을 실행할 수 있습니다.

## 게시물 번호 리스트 만들기  
PAGE <- seq(from=1,to=41,by=10) # 시작값과 종료값을 지정해줄 수 있습니다.  
PAGE <- seq(from=1,by=10,length.out=5) # 시작값과 원하는 갯수를 지정할 수도 있습니다.  
PAGE

## [1] 1 11 21 31 41

이제 for 문을 이용하여 이를 한꺼번에 리스트로 만들겠습니다.

## 네이버 검색결과 url 리스트 만들기  
naver\_url\_list <- c()  
for (date\_i in DATE){  
 for (page\_i in PAGE){  
 dt <- format(as.Date(date\_i,origin="1970-01-01"), "%Y.%m.%d")  
 naver\_url <- paste0(naver\_url\_1,QUERY,naver\_url\_2,dt,naver\_url\_3,dt,naver\_url\_4,page\_i)  
 naver\_url\_list <- c(naver\_url\_list, naver\_url)  
 }  
}  
head(naver\_url\_list,5)

## [1] "https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=1"   
## [2] "https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=11"  
## [3] "https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=21"  
## [4] "https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=31"  
## [5] "https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=41"

우리가 웹크롤링하고자 하는 주소 리스트가 생성되었습니다.

## 검색결과에 포함된 링크 목록 만들기

이제 본격적으로 웹 크롤링을 실시하겠습니다. 우선 위에서 만든 주소 리스트 가운데 하나를 가져와 분석합시다.

https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=1

우리에게 필요한 것은 이 페이지에서 나타난 네이버뉴스 링크입니다. 일반 신문사 링크를 가져오면 제각각 홈페이지 구조가 달라서 웹크롤링이 힘들어집니다.

먼저 페이지의 소스를 뜯어봅시다. 웹브라우저로 크롬을 쓰는 것을 추천합니다. F12 를 누르면 페이지 소스를 뜯어볼 수 있습니다.

(스크린샷을 동반한 상세한 설명이 필요한 부분이지만 일단 넘어가겠습니다. 필요시 참고 웹사이트를 찾아보세요.)

우리가 원하는 네이버 뉴스 링크들은 id="main\_pack" , class="news ..." , class="type01" 아래에 있으며 href= 라는 attribute 에 있는 것을 확인할 수 있습니다. html 소스를 분석하여 이러한 조건을 만족하는 값들만 추려내면 아래와 같습니다.

naver\_url <- "https://search.naver.com/search.naver?&where=news&query=미국+기준금리&pd=3&ds=2018.11.19&de=2018.11.19&start=1"  
html <- read\_html(naver\_url)  
temp <- unique(html\_nodes(html,'#main\_pack')%>% # id= 는 # 을 붙인다  
 html\_nodes(css='.news ')%>% # class= 는 css= 를 붙인다   
 html\_nodes(css='.type01')%>%  
 html\_nodes('a')%>%  
 html\_attr('href'))  
  
head(temp,5)

## [1] "http://news.einfomax.co.kr/news/articleView.html?idxno=4003244"   
## [2] "#"   
## [3] "http://www.seoulwire.com/news/articleView.html?idxno=35031"   
## [4] "http://news.mt.co.kr/mtview.php?no=2018111911542399654"   
## [5] "https://news.naver.com/main/read.nhn?mode=LSD&mid=sec&sid1=004&oid=008&aid=0004134391"

news.naver.com 으로 시작하는 웹사이트 외에도 많은 사이트들이 들어가있는 것을 확인할 수 있습니다. 이것들을 지우는 것은 나중에 한꺼번에 처리하겠습니다.

## 네이버 검색결과 url 리스트에서 관련기사 url 리스트 만들기  
news\_url <- c()  
news\_date <-c()   
  
for (date\_i in DATE){  
 for (page\_i in PAGE){  
 dt <- format(as.Date(date\_i,origin="1970-01-01"), "%Y.%m.%d")  
 naver\_url <- paste0(naver\_url\_1,QUERY,naver\_url\_2,dt,naver\_url\_3,dt,naver\_url\_4,page\_i)  
 html <- read\_html(naver\_url)  
 temp <- unique(html\_nodes(html,'#main\_pack')%>% # id= 는 # 을 붙인다  
 html\_nodes(css='.news ')%>% # class= 는 css= 를 붙인다   
 html\_nodes(css='.type01')%>%  
 html\_nodes('a')%>%  
 html\_attr('href'))  
 news\_url <- c(news\_url,temp)  
 news\_date <- c(news\_date,rep(dt,length(temp)))  
 }  
 print(dt) # 진행상황을 알기 위함이니 속도가 느려지면 제외  
}  
  
NEWS0 <- as.data.frame(cbind(date=news\_date, url=news\_url, query=QUERY))  
NEWS1 <- NEWS0[which(grepl("news.naver.com",NEWS0$url)),] # 네이버뉴스(news.naver.com)만 대상으로 한다   
NEWS1 <- NEWS1[which(!grepl("sports.news.naver.com",NEWS1$url)),] # 스포츠뉴스(sports.news.naver.com)는 제외한다   
NEWS2 <- NEWS1[!duplicated(NEWS1), ] # 중복된 링크 제거

첫번째 url <https://news.naver.com/main/read.nhn?mode=LSD&mid=sec&sid1=004&oid=008&aid=0004134391> 을 웹브라우저 주소창에 입력해봅시다. 원하는 뉴스가 정상적으로 출력되는 것을 확인할 수 있습니다.

## 뉴스 기사의 제목과 본문 추출하기

거의 다 왔습니다. 지금까지 우리는 원하는 기간 동안 원하는 키워드를 포함한 뉴스의 url 주소를 수집하는데 성공했습니다. 그럼 이 url 주소를 이용하여 기사의 제목과 본문을 추출해 봅시다.

첫번째 url <https://news.naver.com/main/read.nhn?mode=LSD&mid=sec&sid1=004&oid=008&aid=0004134391> 을 웹브라우저 주소창에 입력해봅시다. 그리고 마찬가지로 F12 를 눌러 소스를 뜯어보겠습니다.

분석 결과, 기사의 제목은 id="articleTitle" 아래에 있고 본문은 id="articleBodyContents" 아래에 있는 것을 확인했습니다. 앞에서 생성한 url 리스트를 이용하여 각 기사의 제목과 본문을 추출하고, 중복된 문자열을 제거한 최종 결과를 NEWS 에 저장하겠습니다.

## 뉴스 페이지에 있는 기사의 제목과 본문을 크롤링  
NEWS2$news\_title <- ""  
NEWS2$news\_content <- ""  
  
for (i in 1:dim(NEWS2)[1]){  
 html <- read\_html(as.character(NEWS2$url[i]))  
 temp\_news\_title <- repair\_encoding(html\_text(html\_nodes(html,'#articleTitle')),from = 'utf-8')  
 temp\_news\_content <- repair\_encoding(html\_text(html\_nodes(html,'#articleBodyContents')),from = 'utf-8')  
 if (length(temp\_news\_title)>0){  
 NEWS2$news\_title[i] <- temp\_news\_title  
 NEWS2$news\_content[i] <- temp\_news\_content  
 }  
}  
  
NEWS2$news\_content <- gsub("// flash 오류를 우회하기 위한 함수 추가\nfunction \_flash\_removeCallback()", "", NEWS2$news\_content) # 중복된 문자열 제거  
NEWS <- NEWS2 # 최종 결과 저장  
save(NEWS, file="./DATA0/NEWS.RData") # working directory 아래 subfolder "DATA0" 에 저장

## 전체 코드

이번 포스트에서 다룬 R을 이용하여 네이버 뉴스에서 원하는 키워드의 검색 결과를 웹크롤링(스크래핑) 하기 위한 전체 코드는 아래와 같습니다.

## 네이버 뉴스에서 원하는 키워드의 검색 결과를 웹크롤링(스크래핑)하는 코드  
## 제작: hkim (dr-hkim.github.io)  
  
## 패키지 불러오기  
library(rvest)  
library(dplyr)  
  
## 변수 입력하기  
QUERY <- "미국+기준금리" # 검색키워드  
DATE <- as.Date(as.character(20181119),format="%Y%m%d") # 검색시작날짜 & 검색종료날짜  
DATE <- format(DATE, "%Y.%m.%d")  
PAGE <- 1  
  
naver\_url\_1 <- "https://search.naver.com/search.naver?&where=news&query="  
naver\_url\_2 <- "&pd=3&ds="  
naver\_url\_3 <- "&de="  
naver\_url\_4 <- "&start="  
  
## 날짜 리스트 만들기  
DATE\_START <- as.Date(as.character(20181119),format="%Y%m%d") # 시작일자  
DATE\_END <- as.Date(as.character(20181121), format="%Y%m%d") # 종료일자  
DATE <- DATE\_START:DATE\_END  
DATE <- as.Date(DATE,origin="1970-01-01")  
  
## 게시물 번호 리스트 만들기  
PAGE <- seq(from=1,to=41,by=10) # 시작값과 종료값을 지정해줄 수 있습니다.  
PAGE <- seq(from=1,by=10,length.out=5) # 시작값과 원하는 갯수를 지정할 수도 있습니다.  
  
## 네이버 검색결과 url 리스트에서 관련기사 url 리스트 만들기  
news\_url <- c()  
news\_date <-c()   
  
for (date\_i in DATE){  
 for (page\_i in PAGE){  
 dt <- format(as.Date(date\_i,origin="1970-01-01"), "%Y.%m.%d")  
 naver\_url <- paste0(naver\_url\_1,QUERY,naver\_url\_2,dt,naver\_url\_3,dt,naver\_url\_4,page\_i)  
 html <- read\_html(naver\_url)  
 temp <- unique(html\_nodes(html,'#main\_pack')%>% # id= 는 # 을 붙인다  
 html\_nodes(css='.news ')%>% # class= 는 css= 를 붙인다   
 html\_nodes(css='.type01')%>%  
 html\_nodes('a')%>%  
 html\_attr('href'))  
 news\_url <- c(news\_url,temp)  
 news\_date <- c(news\_date,rep(dt,length(temp)))  
 }  
 print(dt) # 진행상황을 알기 위함이니 속도가 느려지면 제외  
}  
  
NEWS0 <- as.data.frame(cbind(date=news\_date, url=news\_url, query=QUERY))  
NEWS1 <- NEWS0[which(grepl("news.naver.com",NEWS0$url)),] # 네이버뉴스(news.naver.com)만 대상으로 한다   
NEWS1 <- NEWS1[which(!grepl("sports.news.naver.com",NEWS1$url)),] # 스포츠뉴스(sports.news.naver.com)는 제외한다   
NEWS2 <- NEWS1[!duplicated(NEWS1), ] # 중복된 링크 제거   
  
  
## 뉴스 페이지에 있는 기사의 제목과 본문을 크롤링  
NEWS2$news\_title <- ""  
NEWS2$news\_content <- ""  
  
for (i in 1:dim(NEWS2)[1]){  
 html <- read\_html(as.character(NEWS2$url[i]))  
 temp\_news\_title <- repair\_encoding(html\_text(html\_nodes(html,'#articleTitle')),from = 'utf-8')  
 temp\_news\_content <- repair\_encoding(html\_text(html\_nodes(html,'#articleBodyContents')),from = 'utf-8')  
 if (length(temp\_news\_title)>0){  
 NEWS2$news\_title[i] <- temp\_news\_title  
 NEWS2$news\_content[i] <- temp\_news\_content  
 }  
}  
  
NEWS2$news\_content <- gsub("// flash 오류를 우회하기 위한 함수 추가\nfunction \_flash\_removeCallback()", "", NEWS2$news\_content)  
NEWS <- NEWS2 # 최종 결과 저장  
save(NEWS, file="./DATA0/NEWS.RData") # working directory 아래 subfolder "DATA0" 에 저장