텍스트 자료의 처리

Jinseog Kim
Dongguk University
jskim1986@gmail.com

2018-03-07

Contents

1 문자열 처리

1.1 문자열의 처리 패키지 및 함수

1. 패키지: stringr, base

2. 함수

Function	설명	기본 함수
str_c()	문자열 결합	paste()
str_length()	문자의 개수	nchar()
str_sub()	부분문자열 추출	substring()
str_dup()	문자의 중복	
str_trim()	문자열의 앞/뒤 공백 제거	
word()	문자열에서 단어 추출	

1.2 문자열 조립 - 이어 붙이기

- paste(x, sep, collapse = NULL)

```
paste("I", "love", "R", sep = "-")
```

```
## [1] "I-love-R"
```

```
# paste with objects of different lengths
paste("X", 1:5, sep = ".")
```

```
## [1] "X.1" "X.2" "X.3" "X.4" "X.5"
```

```
# paste with collapsing
paste(1:3, c("!", "?", "+"), sep = "", collapse = "")
```

```
## [1] "1!2?3+"
```

1.3 문자열의 길이 계산

 nchar(x) 함수

```
texts <- c("텍스트마이닝", "Text Mining","자연어처리",
           "Natural Language Processing")
texts
```

```
## [1] "텍스트마이닝"                "Text Mining"
## [3] "자연어처리"                  "Natural Language Processing"
```

```
nchar(texts)
```

```
## [1]  6 11  5 27
```

1.4 문자열 추출 (Substring) : 문자열(string)에서 일부 문자열 추출

- substr(x, start, stop)
  - x: 문자열
  - start : 시작위치
  - stop : 끝위치

```
texts <- c("텍스트마이닝", "Text Mining","자연어처리",
           "Natural Language Processing")
substr(x = texts, start = 1, stop = 3)
```

```
## [1] "텍스트" "Tex"    "자연어" "Nat"
```

1.5 문자열의 앞/뒤 공백 제거

- sub_trim(x, side)

```
bad_text = c("This", " example ", "has several ", " whitespaces ")
str_trim(bad_text, side = "left")
```

```
## [1] "This"         "example "     "has several " "whitespaces "
```

1.6 문자열에서 특정 위치의 단어 추출

- word()

```
word(string, start = 1L, end = start, sep = fixed(" "))
```

```
# some sentence
change = c("Be the change", "you want to be")
# extract first word
word(change, 1)
```

```
## [1] "Be"  "you"
```

```
word(change, 2)
```

```
## [1] "the"  "want"
```

1.7 문자열에서 패턴 탐색

- grep(pattern, x, value, fixed, ...) 함수
  - pattern: 찾으려는 패턴(문자열 또는 정규표현식)
  - x: 문자열
  - value : TRUE|FALSE (FALSE(default)이면 패턴이 있는 위치, TRUE면 해당 문자열 반환)
  - fixed : TRUE|FALSE (FALSE(default), TRUE면 패턴은 문자열)

- 예1

```
texts <- c("텍스트마이닝", "Text Mining","자연어처리", "텍스트 처리",
           "Natural Language Processing", "텍스트 데이터")
grep("텍스트", x = texts)
```

```
## [1] 1 4 6
```

```
grep("텍스트", x = texts, value=TRUE)
```

```
## [1] "텍스트마이닝"  "텍스트 처리"   "텍스트 데이터"
```

 예2: 텍스트|Text : “텍스트” or “Text”가 포함된 패턴

```
grep("텍스트|Text", x = texts, value=TRUE)
```

```
## [1] "텍스트마이닝"  "Text Mining"   "텍스트 처리"   "텍스트 데이터"
```

1.8 문자열에서 패턴 탐색: 정규표현식(Regular expression)

1. [abc] : 대괄호안의 문자가 적어도 하나 포함된 패턴

```
i <- grep("[텍T]", texts)
i
```

```
## [1] 1 2 4 6
```

```
texts[i]
```

```
## [1] "텍스트마이닝"  "Text Mining"   "텍스트 처리"   "텍스트 데이터"
```

2. [0-9] : match any digit

```
numerics = c("123", "17-April", "I-II-III", "R 3.0.1")
grep(pattern = "[0-9]", numerics, value = TRUE)
```

```
## [1] "123"      "17-April" "R 3.0.1"
```

3. [^0-9] : 숫자로만 된 문자열 제외

```r
grep(pattern = "[^0-9]", numerics, value = TRUE)
```

```
## [1] "17-April" "I-II-III" "R 3.0.1"
```

 특수한 패턴의 정규표현식(Regular expression)

Class	Description
[[:lower:]]	영문: 소문자 (lower-case)
[[:upper:]]	영문: 소문자 (upper-case)
[[:alpha:]]	영문: 모든 알파벳 문자
[[:digit:]]	숫자(digits): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
[[:alnum:]]	영문 알파벳 및 숫자
[[:blank:]]	스페이스, 탭 등 공백문자
[[:cntrl:]]	제어문자
[[:punct:]]	문장부호: ! ” # % & ’ ( ) * + , - . / : ;
[[:space:]]	공백문자: tab, newline, vertical tab, form feed, carriage return, and space
[[:print:]]	인쇄가능한 문자 ([[:alpha:]], [[:punct:]] and space)

```
la_vie <- "La vie en #FFC0CB (rose);\nCes't la vie! \ttres jolie"
gsub(pattern = "[[:blank:]]", replacement = "", la_vie)
```

```
## [1] "Lavieen#FFC0CB(rose);\nCes'tlavie!tresjolie"
```

1.9 문자열 분해

 str_split, strsplit

```
# a sentence
sentence = c("R is a collaborative project with many contributors")
# split into words
strsplit(sentence, " ")
```

```
## [[1]]
## [1] "R"             "is"            "a"             "collaborative"
## [5] "project"       "with"          "many"          "contributors"
```

```
str_split(sentence, " ")
```

```
## [[1]]
## [1] "R"             "is"            "a"             "collaborative"
## [5] "project"       "with"          "many"          "contributors"
```

1.10 연습

1. 아래의 URL의 텍스트 데이터를 읽고
2. 모든 문장부호를 제거한 후
3. 공백을 기준으로 단어를 분리하시오.
4. 사용된 서로 다른 단어의 수는?
5. 단어의 길이에 대한 분포를 구하고
6. 한자짜리 단어는 삭제한 후
7. 단어의 빈도를 상위빈도순으로 구하시오

http://datamining.dongguk.ac.kr/lectures/2018-1/bigdata/자소서.txt

1.11 연습: 한글 인코딩 방법에 따라 파일 읽어오기

1. 파일이 UTF-8로 인코딩된 경우

    1. 윈도우즈에서 읽기

```
file1 <- "http://datamining.dongguk.ac.kr/lectures/2018-1/bigdata/자소서-utf.txt"
file1 <- URLencode(iconv(file1, "CP949", "UTF-8"))
texts <- readLines(file1, encoding="UTF-8"); head(texts, 5)
```

    2. LINUX에서 읽기

```
file1 <- "http://datamining.dongguk.ac.kr/lectures/2018-1/bigdata/자소서-utf.txt"
texts <- readLines(file1); head(texts, 5)
```

```
## [1] "- 지원동기 및 포부: 가장 건전한 재무구조로 외국 투자자들이 가장 선호하는 기업, 대학
## [2] "저는 앞선 기술과 최고의 서비스로 고객 만족을 위해 최선을 다하는 기업, 삼성의 일원이
## [3] "기업이 발전하기 위해서는 무엇보다도 인재 등용이 가장 중요하다고 생각합니다. 저는 실
## [4] "- 학교생활 및 특기사항: ○○대학교 법학과에 수석 입학한 저는, 학과 공부는 물론 동아리
## [5] "또한, 저는 학교 홍보도우미로 발탁되어 교내외의 각종 홍보활동은 물론 학교 홍보 자료
```

2. 파일이 CP949로 인코딩된 경우

    1. Windows에서

```
file1 <- "http://datamining.dongguk.ac.kr/lectures/2018-1/bigdata/자소서.txt"
file1 <- URLencode(iconv(file1, "CP949", "UTF-8"))
texts <- readLines(file1); head(texts, 5)
```

2. LINUX에서

```
file1 <- "http://datamining.dongguk.ac.kr/lectures/2018-1/bigdata/자소서.txt"
texts <- read.table(file1, sep = '\n', fileEncoding = "CP949", stringsAsFactors = F)[,1]
head(texts, 5)
```

```
## [1] "- 지원동기 및 포부: 가장 건전한 재무구조로 외국 투자자들이 가장 선호하는 기업, 대학
## [2] "저는 앞선 기술과 최고의 서비스로 고객 만족을 위해 최선을 다하는 기업, 삼성의 일원이
## [3] "기업이 발전하기 위해서는 무엇보다도 인재 등용이 가장 중요하다고 생각합니다. 저는 실
## [4] "- 학교생활 및 특기사항: ○○대학교 법학과에 수석 입학한 저는, 학과 공부는 물론 동아리
## [5] "또한, 저는 학교 홍보도우미로 발탁되어 교내외의 각종 홍보활동은 물론 학교 홍보 자료
```

1.12 연습: 한글텍스트 처리

```
texts <- gsub("[[:punct:]]", "", texts); head(texts, 2)
```

```
## [1] " 지원동기 및 포부 가장 건전한 재무구조로 외국 투자자들이 가장 선호하는 기업 대학생들이 가장
## [2] "저는 앞선 기술과 최고의 서비스로 고객 만족을 위해 최선을 다하는 기업 삼성의 일원이 되어 저
```

```
texts <- str_trim(texts)
texts <- str_split(texts, "[[:space:]]");  head(texts, 2)
```

```
## [[1]]
##  [1] "지원동기"     "및"           "포부"         "가장"
##  [5] "건전한"       "재무구조로"   "외국"         "투자자들이"
##  [9] "가장"         "선호하는"     "기업"         "대학생들이"
## [13] "가장"         "일하고"       "싶"           "기업"
## [17] "삼성은"       "설명이"       "필요없는"     "우리나라"
## [21] "최고의"       "기업입니다"   "조기출퇴근제" "현장근무제"
## [25] "양"           "위주의"       "관행"         "척결"
## [29] "불합리하고"   "불필요한"     "규정"         "철폐"
## [33] "신인사제도의" "추진"         "등"           "삼성만의"
```

```
## [37] "차별화된"      "경영을"        "통해"          "국내는"
## [41] "물론"          "해외에서도"    "큰"            "호응을"
## [45] "받고"          "있는"          "삼성은"        "명실공히"
## [49] "우리나라를"    "대표하는"      "21세기"        "세계"
## [53] "초일류"        "기업이라고"    "생각합니다"
##
## [[2]]
##  [1] "저는"        "앞선"        "기술과"      "최고의"      "서비스로"
##  [6] "고객"        "만족을"      "위해"        "최선을"      "다하는"
## [11] "기업"        "삼성의"      "일원이"      "되어"        "저의"
## [16] "모든"        "능력과"      "열정을"      "다해"        "일해보고"
## [21] "싶습니다"    "저는"        "그동안"      "철저한"      "자기관리와"
## [26] "시간관리를"  "통해"        "저에게"      "맡겨진"      "일에"
## [31] "대해서는"    "한치의"      "실수"        "없이"        "완벽하게"
## [36] "처리할"      "수"          "있도록"      "최선을"      "다해"
## [41] "왔으며"      "그"          "결과"        ""            "하면"
## [46] "주위에서는"  "무슨"        "일이든"      "잘"          "하고"
## [51] "어떤"        "일이든지"    "믿고"        "맡길"        "수"
## [56] "있는"        "믿음직한"    "사람이라고"  "평해"        "주시곤"
## [61] "했습니다"
```

```r
x <- do.call(c, texts) # 리스트 객체를 벡터로 변환
x <- x[nchar(x) > 2]
tx <- table(x)
sort(tx, decreasing = T)[1:20]
```

```
## x
##   있습니다     최선을 있었습니다   했습니다 생각합니다     사람이
##         22         13          9          9          8          7
##   싶습니다     저에게     주어진     지니고     최고의     컴퓨터
##          7          7          6          6          6          6
##     그리고   믿음직한     언제나   지원동기     합니다      ERP를
##          5          5          5          5          5          4
##       PI를       것으로
##          4          4
```