데이터 사이언스

1. 11000개 text파일 위 6줄 Table에 불러오기

```
#################################################################################################################
# 경로지정, 경로에 파일 불러오기, 개수 999999999개로 늘리고 불러오기

options(max.print=999999999)
directory<-c("C:/Users/yosub/Documents/StatewithR/Project/data")
read_file<-list.files(directory)
read_file
str(read_file)
read_file_leng<-length(read_file)
read_file_leng
```

➔ 경로설정 및 11000개 파일 불러오기

```
#################################################################################################################
# 11000개 text 앞 6개 불러오기 및 frame ( read_file_leng = txt 개수 길이 )

for(i in 1:read_file_leng){
  dataset<-read.table(
    paste(directory, "/", read_file[i], sep=""), sep="=", header=FALSE, nrows=6, fill=T)
  dataset<-dataset[,c(1:2)]
  dataset_mid = as.data.frame(t(dataset))
  names(dataset_mid)<-c("ID", "URL", "SIZE", "DATA", "TIME", "DATASET")
  rownames(dataset_mid)<-NULL
  dataset_mid<-dataset_mid[c(2),]
  if(i==1)
    dataset_result<-dataset_mid
  else
    dataset_result<-rbind(dataset_mid, dataset_result)

  rm(dataset_mid)
  rm(dataset)
}
```

➔ ID, URL, SIZE, DATA, TIME, DATASET 불러온 후 한 Table에 불러오기

```
# 11000개 data.csv에 저장
write.csv(dataset_result, "C:/Users/yosub/Documents/StatewithR/Project/result/data.csv")
```

	ID	URL	SIZE	DATA	TIME	DATASET
2	X1000	http://www	10429	12/07/200	17:41:43	Sport
210999	X0999	http://www	5487	12/07/200	17:41:43	Sport
210998	X0998	http://www	5516	12/07/200	17:41:42	Sport
210997	X0997	http://quiz	25876	12/07/200	17:41:41	Sport
210996	X0996	http://www	52021	12/07/200	17:41:41	Sport
210995	X0995	http://www	34569	12/07/200	17:41:40	Sport
210994	X0994	http://www	16619	12/07/200	17:41:39	Sport
210993	X0993	http://www	5091	12/07/200	17:41:39	Sport
210992	X0992	http://www	12913	12/07/200	17:41:39	Sport
210991	X0991	http://www	7478	12/07/200	17:41:38	Sport
210990	X0990	http://www	12473	12/07/200	17:41:38	Sport
210989	X0989	http://www	18730	12/07/200	17:41:38	Sport

➔ csv파일로 저장

2. 11000개의 .txt 파일의 URL읽어와서 데이터 추출 및 분류

```
#################################################################################################
# url읽어오고 데이터 추출하기 ( 공백 데이터 삭제 및 추출 )

library("rvest")

# init -> a, p, script, span 태그 불러오기
i=1
html = read_html(paste0(directory, "/", read_file[i], sep=""), encoding="UTF-8")
cast <- html_nodes(html, "a") %>%  html_text()
cast2 <- html_nodes(html,"p") %>% html_text()
cast3 <- html_nodes(html,"script") %>% html_text()
cast4 <- html_nodes(html,"span") %>% html_text()

# create table
table <- data.frame(matrix(nrow=1, ncol=4))
#4개의 열과 1개의 행을 하나만들고 이름설정
names(table) = c('a','p','script','span')
cast = data.frame(length(cast),length(cast2),length(cast3),length(cast4))

# 분류 ( 한 파일내에 태그개수가 20개 이상이면은 1, 아니면 0 )
for(j in 1:4){
  if(cast[,j]>=20){
    table[j][1] =1
  }else
    table[j][1] =0
}
```

➔ **Init 설정 : 각 txt의 URL을 읽어 a, p, script, span 태그 불러와 각 태그 별 개수를 읽어 그 값을 테이블에 넣는다.**

분류 : 한 파일내 태그개수가 20개 이상이면 1, 아니면 0이다. 이 값을 table에 저장

```
# read to html tag
for(i in 2:read_file_leng){
  cast <- html_nodes(html, "a") %>%  html_text()
  cast2 <- html_nodes(html,"p") %>% html_text()
  cast3 <- html_nodes(html,"script") %>% html_text()
  cast4 <- html_nodes(html,"span") %>% html_text()

  # create table
  temp <- data.frame(matrix(nrow=1, ncol=4))
  # 4개의 열과 1개의 행을 하나만들고 이름설정,
  names(temp) = c('a','p','script','span')
  cast = data.frame(length(cast),length(cast2),length(cast3),length(cast4))

  # 분류 ( 한 파일내에 태그개수가 15개 이상이면은 1, 아니면 0 )
  for(j in 1:4){
    if(cast[,j]>=15){
      temp[j][1] =1
    }else
      temp[j][1] =0
  }

  # rbind
  table = rbind(table,temp)
}
```

➔ **나머지 2부터 11000개 설정 : 초기를 제외한 2~11000개의 txt의 URL을 읽어 a, p, script, span 태그 불러와 각 태그 별 개수를 읽어 그 값을 테이블에 넣는다.**

분류 : 한 파일내 태그개수가 20개 이상이면 1, 아니면 0이다. 이 값을 temp에 저장

Temp와 초기값인 table을 합쳐 table에 저장

```
train = (table$a == 1) # a의 값이 1인것들을 train로 설정한다.
test = (!train) # 그 외의 것을 테스트로 설정한다.
```

➔ **train : a의 값이 1인것들을 training data 로 설정한다.**

test : 그 외의 것들을 test data 로 설정한다.

➔ **Linear regression**

```
#linear regression
lm.fit=lm(a~p+span+script, data=table, subset=train)
lm.fit
```

```
> lm.fit=lm(a~p+span+script, data=table, subset=train)
> lm.fit

Call:
lm(formula = a ~ p + span + script, data = table, subset = train)

Coefficients:
(Intercept)            p         span
  1.000e+00           NA   -5.333e-12
     script
         NA
```

: p태그와 script태그는 판별 계수가 NA가 나오며 span은 -5.333e-12의 값이 나온다.

➔ **logistic regression**

```
#logistic regression
glm.fit=glm(as.factor(a)~p+span+script, data=table ,family=binomial)
summary(glm.fit)
```

```
Call:
glm(formula = as.factor(a) ~ p + span + script, family = binomial,
    data = table)

Deviance Residuals:
       Min          1Q      Median
-2.409e-06  -2.409e-06  -2.409e-06
        3Q         Max
-2.409e-06  -2.409e-06

Coefficients: (2 not defined because of singularities)
              Estimate Std. Error z value
(Intercept) -2.657e+01  3.561e+05       0
p                   NA         NA      NA
span        -2.308e-10  3.561e+05       0
script              NA         NA      NA
            Pr(>|z|)
(Intercept)        1
p                 NA
span               1
script            NA

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 0.0000e+00  on 10999  degrees of freedom
Residual deviance: 6.3817e-08  on 10998  degrees of freedom
AIC: 4

Number of Fisher Scoring iterations: 25
```

: 각각의 Deviance Residuals, 판별 계수 및 null 편차, 잔여 편차의 자유도 나온다.