Text Mining 프로젝트

The Strange Case Of Dr. Jekyll And Mr. Hyde 와 Cinderella

201701071 김하나

INDEX

1 연구배경 및 목적

스포일러?　　　　스포금지?

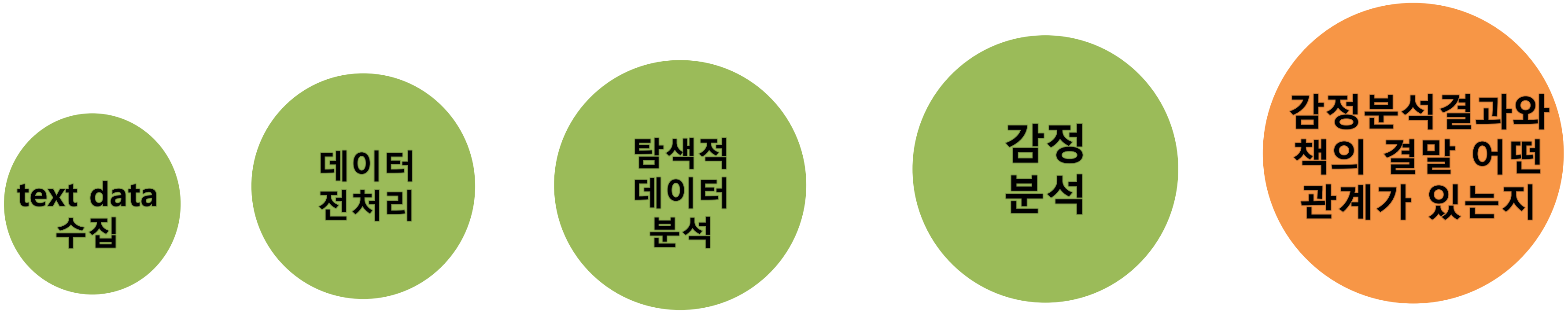
책의 결말을 보지 않고도 결말을 추측해 볼 수 있는 방법

전반적으로 긍정적인 내용 → 결말도 긍정적인 해피엔딩
전반적으로 부정적인 내용 → 결말도 부정적인 새드엔딩 되지 않을까

이런 의문을 해결해보기 위해 두 편의 e-book data로 텍스트마이닝 진행
책의 감정분석 결과와 책의 결말 사이에는 어떤 관계가 있는지 알아보고자 함

2 연구방향

연구방향



3 텍스트 데이터 수집

Project Gutenberg 에서 무료로 제공하는 e-book 데이터를 사용.

Free eBooks - Project Gutenberg

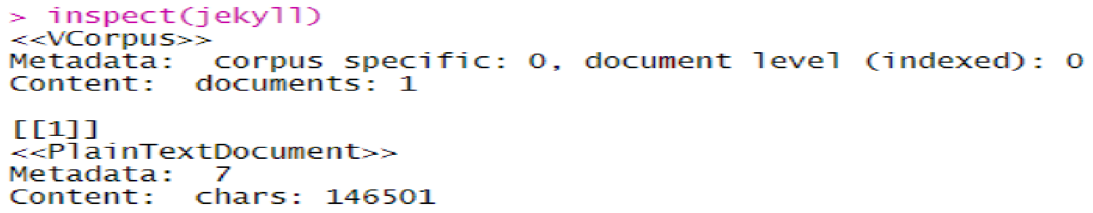
Book search · Book categories · Browse catalog · Mobile site · Report errors · Terms of use

Some of the Latest eBooks

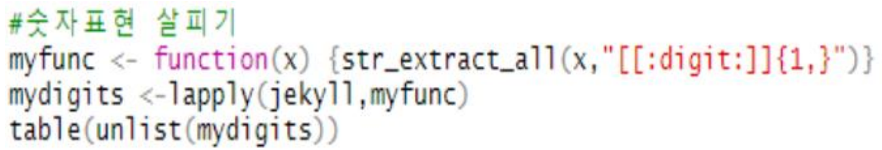
해피엔딩과 새드엔딩을 가진 책 2권의 텍스트 데이터를 사용

The Strange Case of Dr. Jekyll and Mr. Hyde by Robert Louis Stevenson
Cinderella by Henry W. Hewet

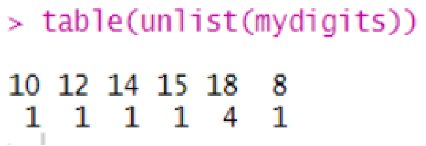
데이터 전 처리 (지킬앤하이드)



```
> inspect(jekyll)
<<VCorpus>>
Metadata:  corpus specific: 0, document level (indexed): 0
Content:  documents: 1

[[1]]
<<PlainTextDocument>>
Metadata:  7
Content:  chars: 146501
```

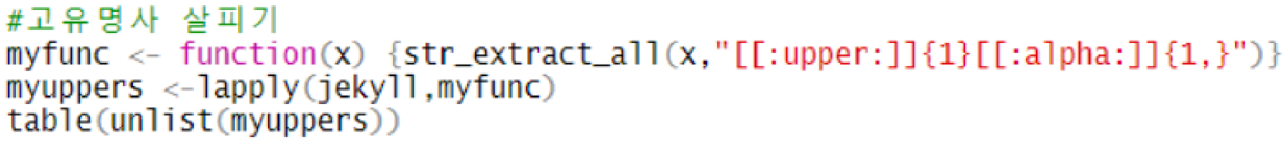


```
#숫자표현 살피기
myfunc <- function(x) {str_extract_all(x,"[[:digit:]]{1,}")}
mydigits <-lapply(jekyll,myfunc)
table(unlist(mydigits))
```



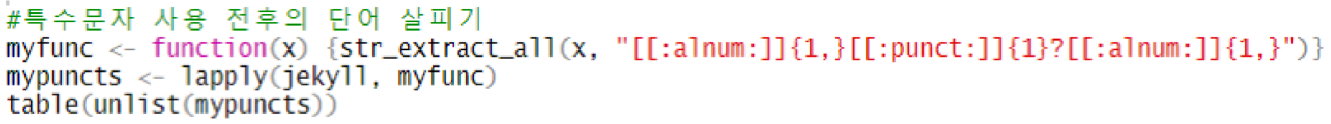
```
> table(unlist(mydigits))

10 12 14 15 18  8
 1  1  1  1  4  1
```

특별한 의미나 패턴을 갖는 숫자표현이 없다고 봐도 무방하다.



```
#고유명사 살피기
myfunc <- function(x) {str_extract_all(x,"[[:upper:]]{1}[[:alpha:]]{1,}")}
myuppers <-lapply(jekyll,myfunc)
table(unlist(myuppers))
```

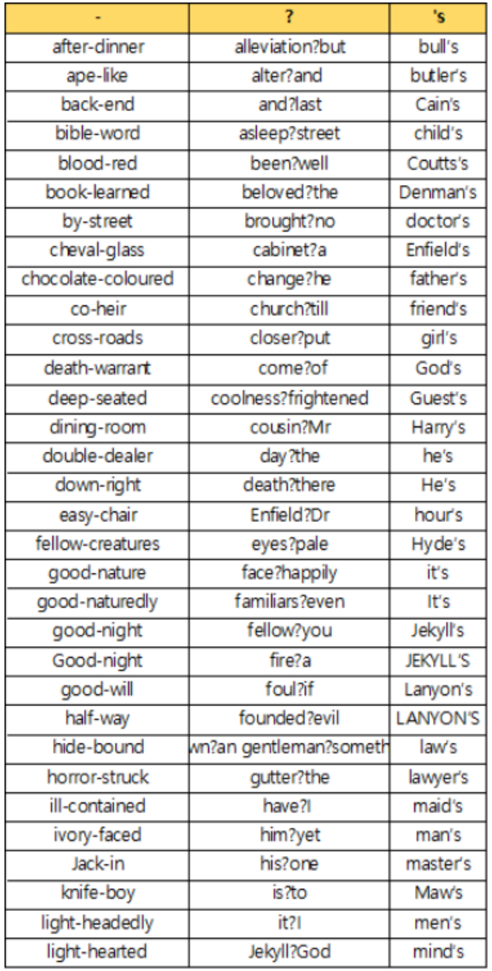
확인 결과 일반명사와 혼동 가능한 고유명사는 존재하지 않는다.



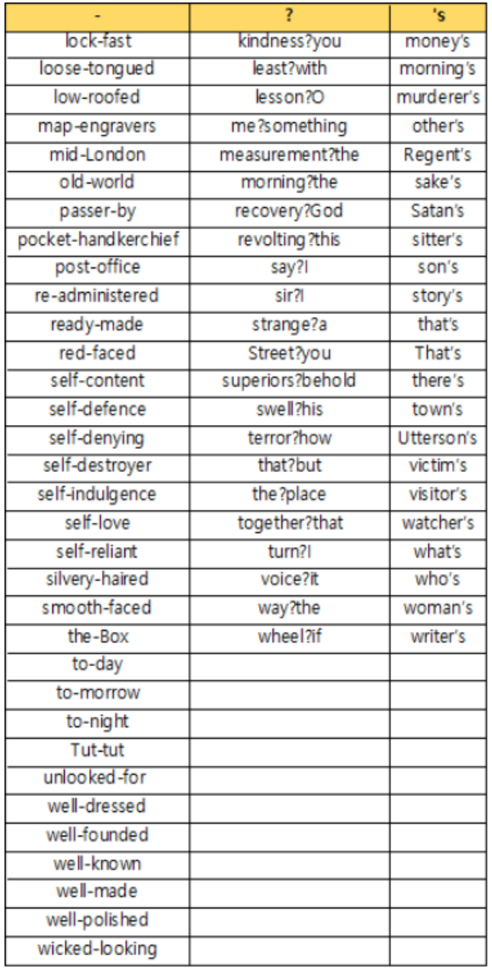
```
#특수문자 사용 전후의 단어 살피기
myfunc <- function(x) {str_extract_all(x, "[[:alnum:]]{1,}[[:punct:]]{1}?[[:alnum:]]{1,}")}
mypuncts <- lapply(jekyll, myfunc)
table(unlist(mypuncts))
```

특수문자의 경우 일괄 삭제하지 않고 목록을 살펴봐야 한다.

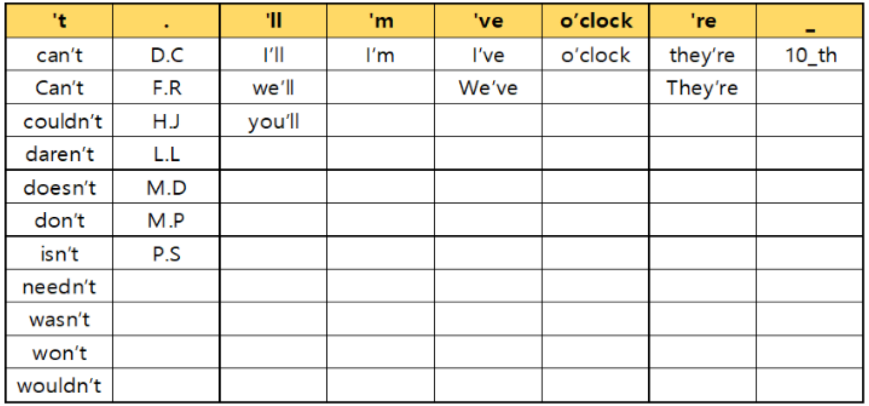
4 데이터 전 처리 (지킬앤하이드)



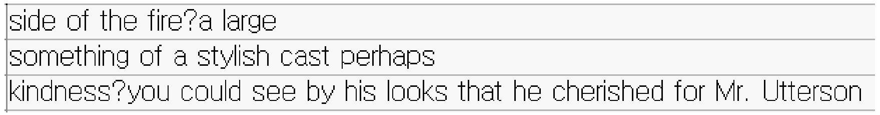
-	?	's
after-dinner	alleviation?but	bull's
ape-like	alter?and	butler's
back-end	and?last	Cain's
bible-word	asleep?street	child's
blood-red	been?well	Coutts's
book-learned	beloved?the	Denman's
by-street	brought?no	doctor's
cheval-glass	cabinet?a	Enfield's
chocolate-coloured	change?he	father's
co-heir	church?till	friend's
cross-roads	closer?put	girl's
death-warrant	come?of	God's
deep-seated	coolness?frightened	Guest's
dining-room	cousin?Mr	Harry's
double-dealer	day?the	he's
down-right	death?there	He's
easy-chair	Enfield?Dr	hour's
fellow-creatures	eyes?pale	Hyde's
good-nature	face?happily	it's
good-naturedly	familiars?even	It's
good-night	fellow?you	Jekyll's
Good-night	fire?a	JEKYLL'S
good-will	foul?if	Lanyon's
half-way	founded?evil	LANYON'S
hide-bound	wn?an gentleman?someth	law's
horror-struck	gutter?the	lawyer's
ill-contained	have?I	maid's
ivory-faced	him?yet	man's
Jack-in	his?one	master's
knife-boy	is?to	Maw's
light-headedly	it?I	men's
light-hearted	Jekyll?God	mind's



-	?	's
lock-fast	kindness?you	money's
loose-tongued	least?with	morning's
low-roofed	lesson?O	murderer's
map-engravers	me?something	other's
mid-London	measurement?the	Regent's
old-world	morning?the	sake's
passer-by	recovery?God	Satan's
pocket-handkerchief	revolting?this	sitter's
post-office	say?I	son's
re-administered	sir?I	story's
ready-made	strange?a	that's
red-faced	Street?you	That's
self-content	superiors?behold	there's
self-defence	swell?his	town's
self-denying	terror?how	Utterson's
self-destroyer	that?but	victim's
self-indulgence	the?place	visitor's
self-love	together?that	watcher's
self-reliant	turn?I	what's
silvery-haired	voice?it	who's
smooth-faced	way?the	woman's
the-Box	wheel?if	writer's
to-day		
to-morrow		
to-night		
Tut-tut		
unlooked-for		
well-dressed		
well-founded		
well-known		
well-made		
well-polished		
wicked-looking		



't	.	'll	'm	've	o'clock	're	_
can't	D.C	I'll	I'm	I've	o'clock	they're	10_th
Can't	F.R	we'll		We've		They're	
couldn't	H.J	you'll					
daren't	L.L						
doesn't	M.D						
don't	M.P						
isn't	P.S						
needn't							
wasn't							
won't							
wouldn't							



side of the fire?a large
something of a stylish cast perhaps
kindness?you could see by his looks that he cherished for Mr. Utterson

4 데이터 전 처리 (지킬앤하이드)

1. 말뭉치에 등장한 숫자표현들은 모두 삭제한다.
2. 말뭉치에 사용된 특수문자들의 경우 두 단계를 통해 사전처리 한다.
   ㄱ. 표에 따라 몇몇 표현들을 교체한다.
   ㄴ. 2-ㄱ에 해당되지 않는 특수문자들은 일괄 삭제한다.
3. 2번 이상 연이어 나타난 공란들은 하나의 스페이스 공란(" ")으로 바꾼다.
4. 대문자로 나타난 텍스트는 모두 소문자로 전환한다.
5. tm 라이브러리에 탑재된 SMART 불용문자 목록에 포함되어 있는 단어들을 모두 삭제한다.
6. 어근이 동일하지만 문법적으로 변용된 단어들을 통합하는 어근 동일화 알고리즘을 적용한다.

4 데이터 전 처리 (지킬앤하이드)

지정된 표현	교체된 표현	지정된 표현	교체된 표현
-like	like	post-	post
-glass	glass	re-	re
co-	co	ready-	ready
cross-	cross	to-	to
double-	double	-faced	faced
down-	down	tut-	tut
easy-	easy	-for	for
-will	will	jack-	jack
half-	half	O'clock	Oclock
hide-	hide	_th	th
horror-	horror	?	공백
knife-	knife	's	제거
light-	light	't	not
lock-	lock	'll	will
loose-	loose	'm	제거
old-	old	've	have
passer-	passer	're	제거

```r
#특수문자 교체

trans <- content_transformer(function(x,from,to) gsub(from,to,x))
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(co-)","co")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "-like","like")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "-glass","glass")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(cross-)","cross")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(double-)","double")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(down-)","down")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(easy-)","easy")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "-will","will")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(half-)","half")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(hide-)","hide")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(horror-)","horror")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(knife-)","knife")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(light-)","light")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(lock-)","lock")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(loose-)","loose")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(old-)","old")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(passer-)","passer")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(post-)","post")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(re-)","re")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(ready-)","ready")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(to-)","to")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "-faced","faced")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(tut-)","tut")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "-for","for")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\b(jack-)","jack")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "O'clock","Oclock")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "_th","th")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\?","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\'s","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\'t","not")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\'ll","will")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\'m","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\'ve","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\\'re","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "-"," ")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\"","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\'","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\‘[[:alnum:]]*","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\’[[:alnum:]]*","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\“[[:alnum:]]*","")
jekyllcorpus <-tm_map(jekyllcorpus, trans, "\”","")
#특수문자 제거
jekyllcorpus <-tm_map(jekyllcorpus, removePunctuation)
```

4 데이터 전 처리 (지킬앤하이드)

```r
#공란 제거
jekyllcorpus <-tm_map(jekyllcorpus, stripWhitespace)
```

```r
#대소문자 통합
jekyllcorpus <-tm_map(jekyllcorpus, content_transformer(tolower))
```

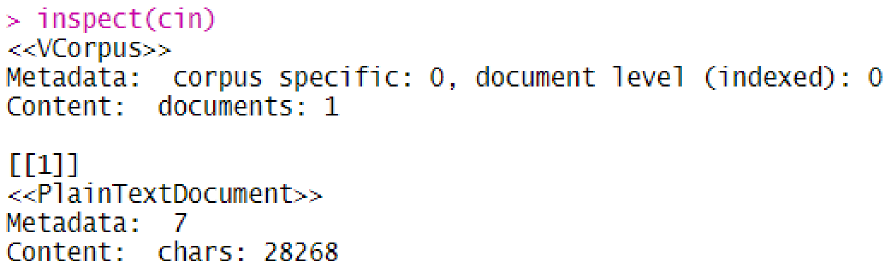
```r
#불용단어 제거
jekyllcorpus <-tm_map(jekyllcorpus, removeWords, words=stopwords("SMART"))
```

```r
#어근 동일화 처리
jekyllcorpus <-tm_map(jekyllcorpus, stemDocument, language="en")
```

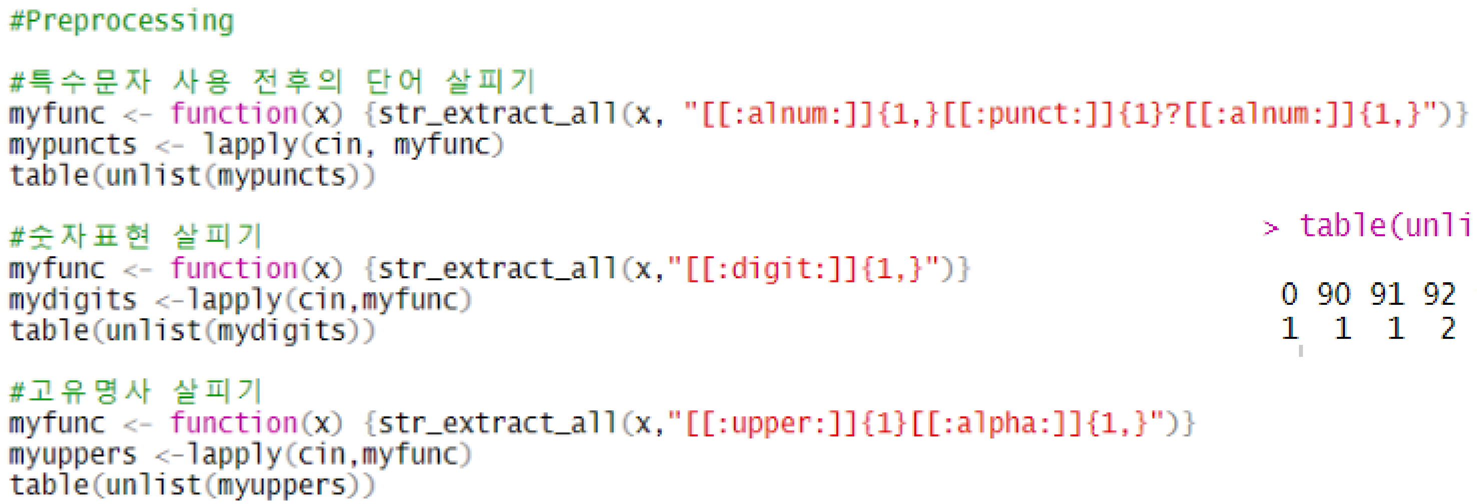
```r
#전처리 전 후 비교
mycharfunc <-function(x) {str_extract_all(x,".")}
mywordfunc <-function(x) {str_extract_all(x,boundary("word"))}

#전처리 전
mychar <- lapply(jekyll, mycharfunc)
myuniquechar0 <-length(table(unlist(mychar)))
mytotalchar0 <-sum(table(unlist(mychar)))
myword <- lapply(jekyll, mywordfunc)
myuniqueword0 <-length(table(unlist(myword)))
mytotalword0 <-sum(table(unlist(myword)))

#전처리 후
mychar<- lapply(jekyllcorpus, mycharfunc)
myuniquechar1 <-length(table(unlist(mychar)))
mytotalchar1 <-sum(table(unlist(mychar)))
myword <- lapply(jekyllcorpus, mywordfunc)
myuniqueword1 <-length(table(unlist(myword)))
mytotalword1 <-sum(table(unlist(myword)))

#전처리 전 후 비교
results.comparing <- rbind(
  +c(myuniquechar0,myuniquechar1),
  +c(mytotalchar0,mytotalchar1),
  +c(myuniqueword0,myuniqueword1),
  +c(mytotalword0,mytotalword1))

colnames(results.comparing) <- c("before","after")
rownames(results.comparing) <- c("고유문자 수","총 문자 수","고유단어 수","총 단어 수")
results.comparing
```

```
> results.comparing
            before after
고유문자 수     71    31
총 문자 수  146501 58054
고유단어 수   4183  2841
총 단어 수   25722  9355
```

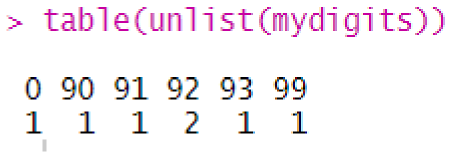
4 데이터 전 처리 (신데렐라)



```
> inspect(cin)
<<VCorpus>>
Metadata:  corpus specific: 0, document level (indexed): 0
Content:  documents: 1

[[1]]
<<PlainTextDocument>>
Metadata:  7
Content:  chars: 28268
```



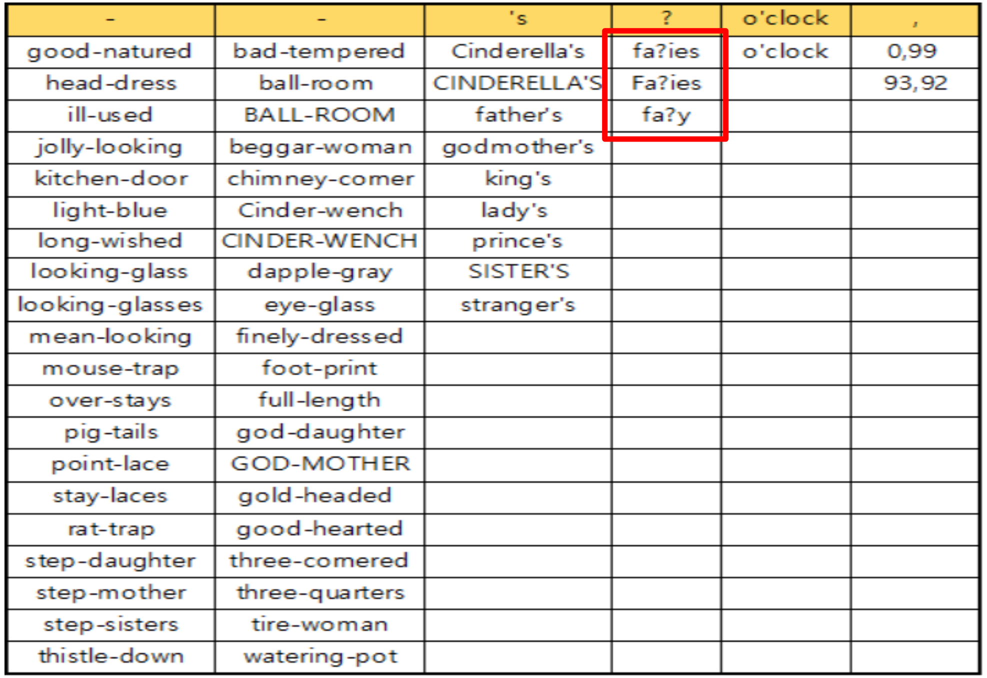
```
#Preprocessing

#특수문자 사용 전후의 단어 살피기
myfunc <- function(x) {str_extract_all(x, "[[:alnum:]]{1,}[[:punct:]]{1}?[[:alnum:]]{1,}")}
mypuncts <- lapply(cin, myfunc)
table(unlist(mypuncts))

#숫자표현 살피기
myfunc <- function(x) {str_extract_all(x,"[[:digit:]]{1,}")}
mydigits <-lapply(cin,myfunc)
table(unlist(mydigits))

#고유명사 살피기
myfunc <- function(x) {str_extract_all(x,"[[:upper:]]{1}[[:alpha:]]{1,}")}
myuppers <-lapply(cin,myfunc)
table(unlist(myuppers))
```



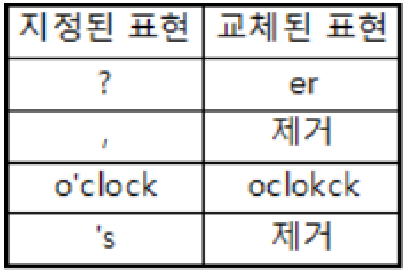
```
> table(unlist(mydigits))

 0 90 91 92 93 99
 1  1  1  2  1  1
```

4

데이터 전 처리 (신데렐라)



-	-	's	?	o'clock	,
good-natured	bad-tempered	Cinderella's	fa?ies	o'clock	0,99
head-dress	ball-room	CINDERELLA'S	Fa?ies		93,92
ill-used	BALL-ROOM	father's	fa?y		
jolly-looking	beggar-woman	godmother's			
kitchen-door	chimney-comer	king's			
light-blue	Cinder-wench	lady's			
long-wished	CINDER-WENCH	prince's			
looking-glass	dapple-gray	SISTER'S			
looking-glasses	eye-glass	stranger's			
mean-looking	finely-dressed				
mouse-trap	foot-print				
over-stays	full-length				
pig-tails	god-daughter				
point-lace	GOD-MOTHER				
stay-laces	gold-headed				
rat-trap	good-hearted				
step-daughter	three-comered				
step-mother	three-quarters				
step-sisters	tire-woman				
thistle-down	watering-pot				



지정된 표현	교체된 표현
?	er
,	제거
o'clock	oclokck
's	제거

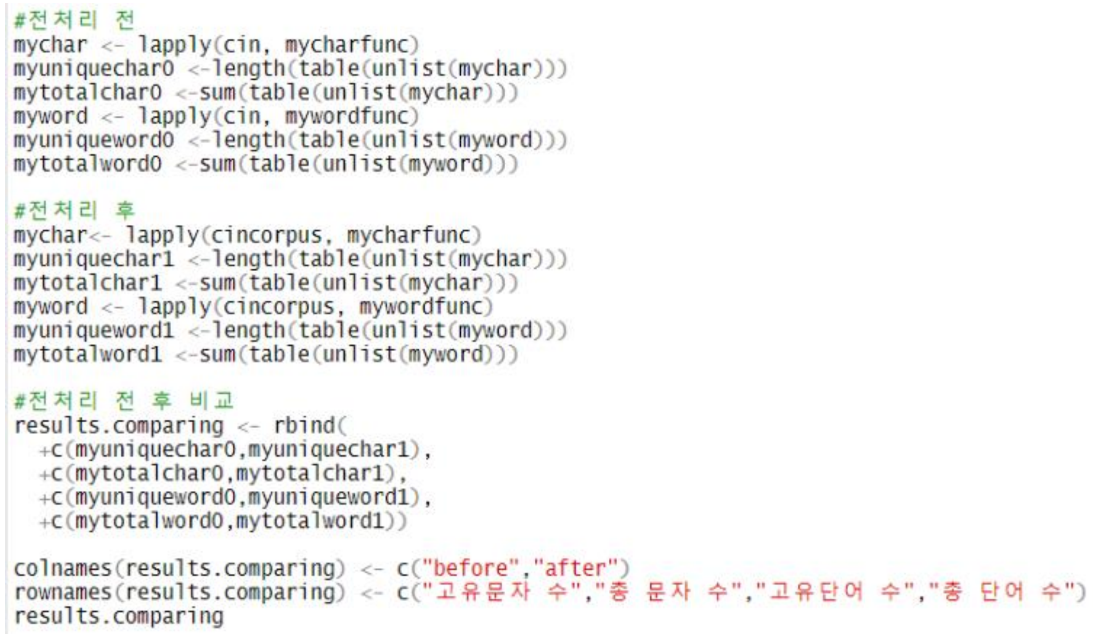
"But you *shall* go, my darling," said the old woman, "or I am not Queen of the Faëries or your Godmother. Dry up your tears like a good god-daughter and do as I bid you, and you shall have clothes and horses finer than any one."

4 데이터 전 처리 (신데렐라)

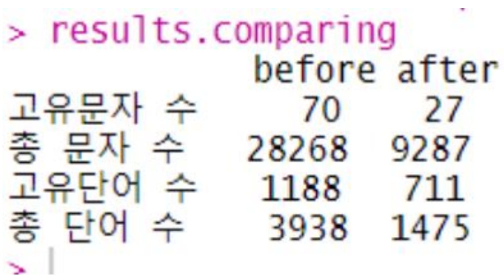
```r
#숫자표현 제거
cincorpus <- tm_map(cin, removeNumbers)

#공란 제거
cincorpus <-tm_map(cincorpus, stripWhitespace)

#대소문자 통합
cincorpus <-tm_map(cincorpus, content_transformer(tolower))

#불용단어 제거
cincorpus <-tm_map(cincorpus, removeWords, words=stopwords("SMART"))

#어근 동일화 처리
cincorpus <-tm_map(cincorpus, stemDocument, language="en")
```

```r
#특수문자 교체

trans <- content_transformer(function(x,from,to) gsub(from,to,x))

cincorpus <-tm_map(cincorpus, trans, "\\?","er")
cincorpus <-tm_map(cincorpus, trans, "\\,","")
cincorpus <-tm_map(cincorpus, trans, "o'clock","oclock")
cincorpus <-tm_map(cincorpus, trans, "\\'s","")
cincorpus <-tm_map(cincorpus, trans, "\\b(good-)","good")
cincorpus <-tm_map(cincorpus, trans, "\\b(head-)","head")
cincorpus <-tm_map(cincorpus, trans, "\\b(ill-)","ill")
cincorpus <-tm_map(cincorpus, trans, "-looking","looking")
cincorpus <-tm_map(cincorpus, trans, "\\b(light-)","light")
cincorpus <-tm_map(cincorpus, trans, "\\b(looking-)","looking")
cincorpus <-tm_map(cincorpus, trans, "\\b(point-)","point")
cincorpus <-tm_map(cincorpus, trans, "\\b(stay-)","stay")
cincorpus <-tm_map(cincorpus, trans, "\\b(step-)","step")
cincorpus <-tm_map(cincorpus, trans, "\\b(thistle-)","thistle")
cincorpus <-tm_map(cincorpus, trans, "\\b(bad-)","bad")
cincorpus <-tm_map(cincorpus, trans, "-room","room")
cincorpus <-tm_map(cincorpus, trans, "-ROOM","ROOM")
cincorpus <-tm_map(cincorpus, trans, "-woman","woman")
cincorpus <-tm_map(cincorpus, trans, "-corner","corner")
cincorpus <-tm_map(cincorpus, trans, "-gray","gray")
cincorpus <-tm_map(cincorpus, trans, "-glass","glass")
cincorpus <-tm_map(cincorpus, trans, "-dressed","dressed")
cincorpus <-tm_map(cincorpus, trans, "-print","print")
cincorpus <-tm_map(cincorpus, trans, "\\b(full-)","full")
cincorpus <-tm_map(cincorpus, trans, "\\b(god-)","god")
cincorpus <-tm_map(cincorpus, trans, "\\b(GOD-)","GOD")
cincorpus <-tm_map(cincorpus, trans, "\\b(gold-)","gold")
cincorpus <-tm_map(cincorpus, trans, "\\b(three-)","three")
cincorpus <-tm_map(cincorpus, trans, "-"," ")
cincorpus <-tm_map(cincorpus, trans, "\"","")
cincorpus <-tm_map(cincorpus, trans, "\'","")

#특수문자 제거
cincorpus <-tm_map(cincorpus, removePunctuation)
```

데이터 전 처리 (신데렐라)



```
#전처리 전
mychar <- lapply(cin, mycharfunc)
myuniquechar0 <-length(table(unlist(mychar)))
mytotalchar0 <-sum(table(unlist(mychar)))
myword <- lapply(cin, mywordfunc)
myuniqueword0 <-length(table(unlist(myword)))
mytotalword0 <-sum(table(unlist(myword)))

#전처리 후
mychar<- lapply(cincorpus, mycharfunc)
myuniquechar1 <-length(table(unlist(mychar)))
mytotalchar1 <-sum(table(unlist(mychar)))
myword <- lapply(cincorpus, mywordfunc)
myuniqueword1 <-length(table(unlist(myword)))
mytotalword1 <-sum(table(unlist(myword)))

#전처리 전 후 비교
results.comparing <- rbind(
  +c(myuniquechar0,myuniquechar1),
  +c(mytotalchar0,mytotalchar1),
  +c(myuniqueword0,myuniqueword1),
  +c(mytotalword0,mytotalword1))

colnames(results.comparing) <- c("before","after")
rownames(results.comparing) <- c("고유문자 수","총 문자 수","고유단어 수","총 단어 수")
results.comparing
```



```
> results.comparing
            before after
고유문자 수     70    27
총 문자 수   28268  9287
고유단어 수   1188   711
총 단어 수    3938  1475
> 
```

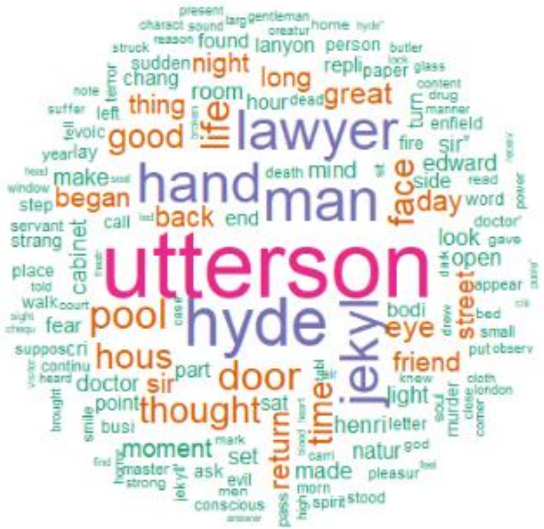
5 탐색적 데이터 분석(지킬앤하이드)

```
#DTM 구축
dtm.j <- DocumentTermMatrix(jekyllcorpus)
dtm.j
View(dtm.j)
#빈도표
word.freq <- apply(dtm.j[,],2,sum)
head(word.freq,50)
sort.word.freq <- sort(word.freq,decreasing=TRUE)
sort.word.freq[1:20]

#누적 빈도 계산
cumsum.word.freq <- cumsum(sort.word.freq)
cumsum.word.freq[1:20]
prop.word.freq <- cumsum.word.freq/cumsum.word.freq[length(cumsum.word.freq)]
prop.word.freq[1:20]
```

```
#wordcloud
display.brewer.all()
pal<- brewer.pal(4,"Dark2")
wordcloud(names(word.freq),freq=word.freq,scale=c(4,0.2),rot.per = 0.2,min.freq=10,random.order=FALSE,col=pal)
```

```
> sort.word.freq <- sort(word.freq,decreasing=TRUE)
> sort.word.freq[1:20]
utterson      hyde       man      hand    lawyer     jekyl      door      pool      life 
     116        84        74        68        68        67        50        50        48 
 thought      face      hous      good      time       eye     great    return      back 
      43        41        41        39        39        34        34        34        33 
     day      long 
      33        31 
```

```
> #누적 빈도 계산
> cumsum.word.freq <- cumsum(sort.word.freq)
> cumsum.word.freq[1:20]
utterson      hyde       man      hand    lawyer     jekyl      door      pool      life 
     116       200       274       342       410       477       527       577       625 
 thought      face      hous      good      time       eye     great    return      back 
     668       709       750       789       828       862       896       930       963 
     day      long 
     996      1027 
> prop.word.freq <- cumsum.word.freq/cumsum.word.freq[length(cumsum.word.freq)]
> prop.word.freq[1:20]
  utterson       hyde        man       hand     lawyer      jekyl       door 
0.01264305 0.02179837 0.02986376 0.03727520 0.04468665 0.05198910 0.05743869 
      pool       life    thought       face       hous       good       time 
0.06288828 0.06811989 0.07280654 0.07727520 0.08174387 0.08599455 0.09024523 
       eye      great     return       back        day       long 
0.09395095 0.09765668 0.10136240 0.10495913 0.10855586 0.11193460 
```



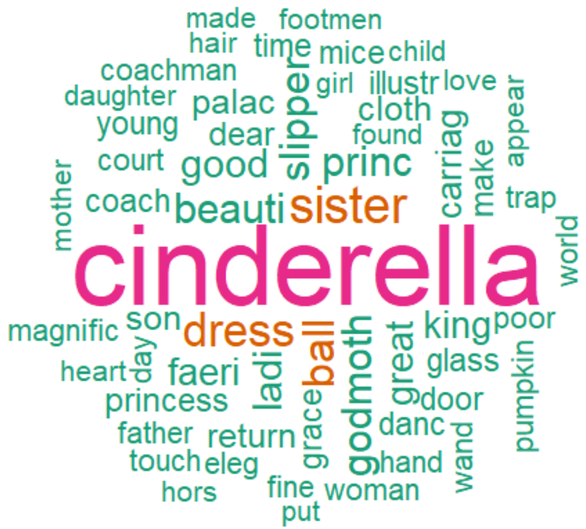
5 탐색적 데이터 분석(신데렐라)

```
#DTM 구축
dtm.c <- DocumentTermMatrix(cincorpus)
dtm.c


#빈도표
word.freq <- apply(dtm.c[,],2,sum)
head(word.freq,50)
sort.word.freq <- sort(word.freq,decreasing=TRUE)
sort.word.freq[1:20]

#누적 빈도 계산
cumsum.word.freq <- cumsum(sort.word.freq)
cumsum.word.freq[1:20]
prop.word.freq <- cumsum.word.freq/cumsum.word.freq[length(cumsum.word.freq)]
prop.word.freq[1:20]
```

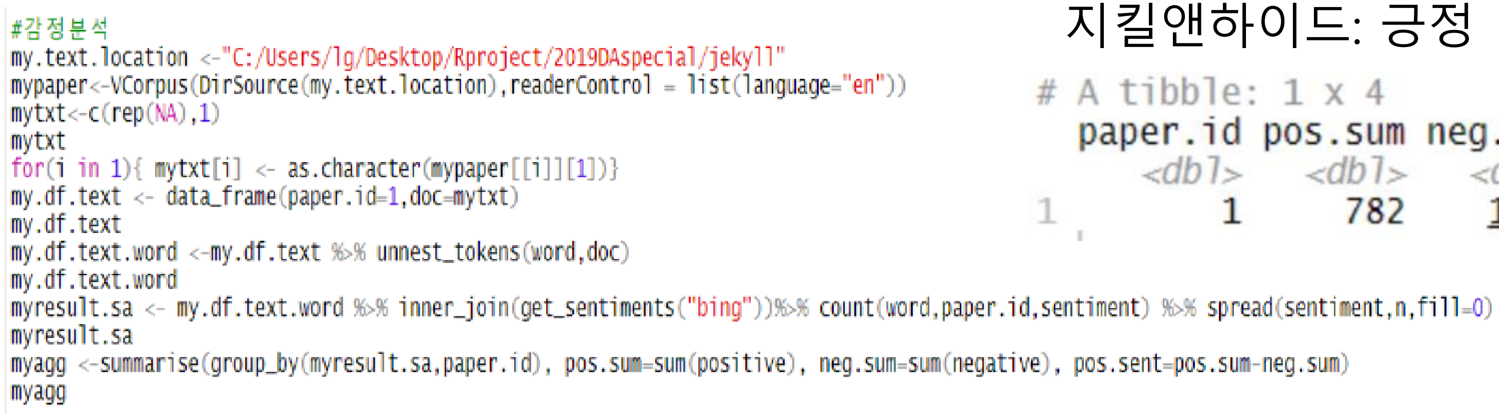
```
#wordcloud
display.brewer.all()
pal<- brewer.pal(4,"Dark2")
wordcloud(names(word.freq),freq=word.freq,scale=c(5,1),rot.per = 0.2,min.freq=5,random.order=FALSE,col=pal)
```

```
> sort.word.freq <- sort(word.freq,decreasing=TRUE)
> sort.word.freq[1:20]
cinderella     sister      dress        ball      beauti     godmoth       princ     slipper
        66         22         19          17          14          14          14          14
      good       ladi      faeri       great        king     carriag       cloth       palac
        13         13         11          11          11          10           9           9
    return        son       dear    princess
         9          9          8           8
```

```
> #누적 빈도 계산
> cumsum.word.freq <- cumsum(sort.word.freq)
> cumsum.word.freq[1:20]
cinderella     sister      dress        ball      beauti     godmoth       princ     slipper
        66         88        107         124         138         152         166         180
      good       ladi      faeri       great        king     carriag       cloth       palac
       193        206        217         228         239         249         258         267
    return        son       dear    princess
       276        285        293         301
> prop.word.freq <- cumsum.word.freq/cumsum.word.freq[length(cumsum.word.freq)]
> prop.word.freq[1:20]
cinderella     sister      dress        ball      beauti     godmoth       princ     slipper
0.04483696 0.05978261 0.07269022 0.08423913 0.09375000 0.10326087 0.11277174 0.12228261
      good       ladi      faeri       great        king     carriag       cloth       palac
0.13111413 0.13994565 0.14741848 0.15489130 0.16236413 0.16915761 0.17527174 0.18138587
    return        son       dear    princess
0.18750000 0.19361413 0.19904891 0.20448370
```

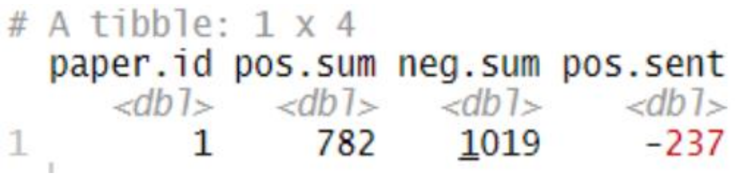


6 감정 분석

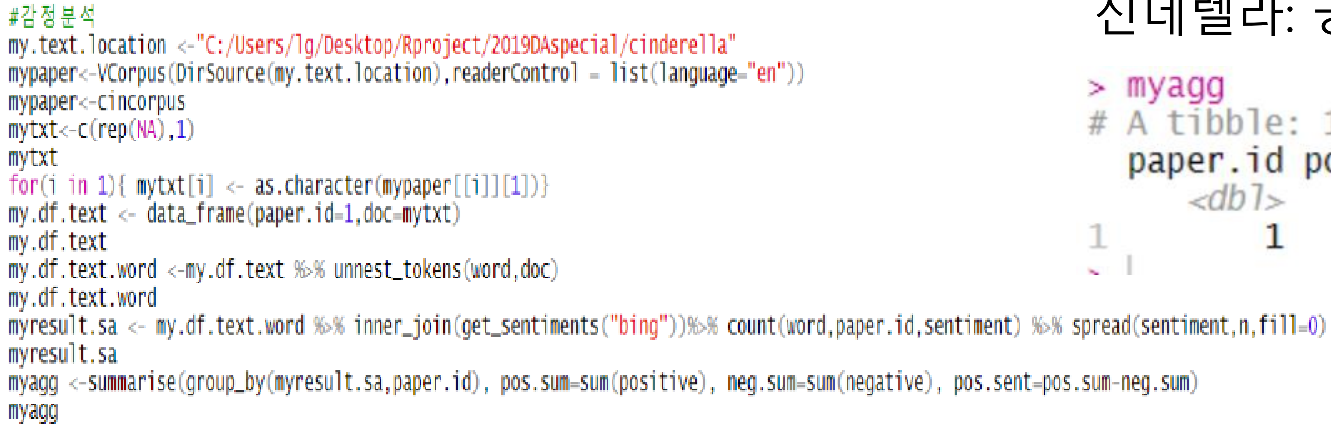


```r
#감정분석
my.text.location <-"C:/Users/lg/Desktop/Rproject/2019DAspecial/jekyll"
mypaper<-VCorpus(DirSource(my.text.location),readerControl = list(language="en"))
mytxt<-c(rep(NA),1)
mytxt
for(i in 1){ mytxt[i] <- as.character(mypaper[[i]][1])}
my.df.text <- data_frame(paper.id=1,doc=mytxt)
my.df.text
my.df.text.word <-my.df.text %>% unnest_tokens(word,doc)
my.df.text.word
myresult.sa <- my.df.text.word %>% inner_join(get_sentiments("bing"))%>% count(word,paper.id,sentiment) %>% spread(sentiment,n,fill=0)
myresult.sa
myagg <-summarise(group_by(myresult.sa,paper.id), pos.sum=sum(positive), neg.sum=sum(negative), pos.sent=pos.sum-neg.sum)
myagg
```

지킬앤하이드: 긍정 < 부정

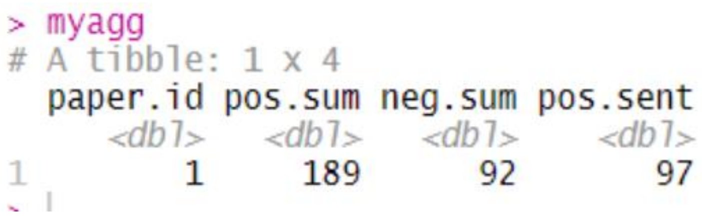


```
# A tibble: 1 x 4
  paper.id pos.sum neg.sum pos.sent
     <dbl>   <dbl>   <dbl>    <dbl>
1        1     782    1019     -237
```



```r
#감정분석
my.text.location <-"C:/Users/lg/Desktop/Rproject/2019DAspecial/cinderella"
mypaper<-VCorpus(DirSource(my.text.location),readerControl = list(language="en"))
mypaper<-cincorpus
mytxt<-c(rep(NA),1)
mytxt
for(i in 1){ mytxt[i] <- as.character(mypaper[[i]][1])}
my.df.text <- data_frame(paper.id=1,doc=mytxt)
my.df.text
my.df.text.word <-my.df.text %>% unnest_tokens(word,doc)
my.df.text.word
myresult.sa <- my.df.text.word %>% inner_join(get_sentiments("bing"))%>% count(word,paper.id,sentiment) %>% spread(sentiment,n,fill=0)
myresult.sa
myagg <-summarise(group_by(myresult.sa,paper.id), pos.sum=sum(positive), neg.sum=sum(negative), pos.sent=pos.sum-neg.sum)
myagg
```

신데렐라: 긍정 > 부정



```
> myagg
# A tibble: 1 x 4
  paper.id pos.sum neg.sum pos.sent
     <dbl>   <dbl>   <dbl>    <dbl>
1        1     189      92       97
```

7 결론

지킬앤하이드의 결말 : 주인공이 죽는 새드엔딩

신데렐라의 결말 : 주인공이 왕자와 결혼하는 해피엔딩

감정분석 결과,

지킬앤하이드 데이터는 긍정적인 단어보다 **부정적인 단어가 237개 더 많음**

신데렐라 데이터는 부정적인 단어보다 **긍정적인 단어가 97개 더 많음**

새드엔딩인 책은 전반적으로 **부정적인 단어가 많이 등장**

해피엔딩인 책은 전반적으로 **긍정적인 단어가 많이 등장**

텍스트 데이터를 두 권으로만 분석했기 때문에 더 많은 데이터로 분석해 봐야 할 필요성이 있고, 중간에 반전이 있는 책에서도 같은 결과가 나올지, 챕터 별로 감정분석을 실시하면 기승전결의 파트 중 어느 파트에 어떤 감정 어휘가 있느냐에 따라 결말이 어떻게 되는지 또한 분석해 보고 싶다.

Thank you

About a week has passed, and I am now finishing this statement under the influence of the last of the old powders. This, then, is the last time, short of a miracle, that Henry Jekyll can think his own thoughts or see his own face (now how sadly altered!) in the glass. Nor must I delay too long to bring my writing to an end; for if my narrative has hitherto escaped destruction, it has been by a combination of great prudence and great good luck. Should the throes of change take me in the act of writing it, Hyde will tear it in pieces; but if some time shall have elapsed after I have laid it by, his wonderful selfishness and circumscription to the moment will probably save it once again from the action of his ape-like spite. And indeed the doom that is closing on us both has already changed and crushed him. Half an hour from now, when I shall again and forever reindue that hated personality, I know how I shall sit shuddering and weeping in my chair, or continue, with the most strained and fearstruck ecstasy of listening, to pace up and down this room (my last earthly refuge) and give ear to every sound of menace. Will Hyde die upon the scaffold? or will he find courage to release himself at the last moment? God knows; I am careless; this is my true hour of death, and what is to follow concerns another than myself. Here then, as I lay down the pen and proceed to seal up my confession, I bring the life of that unhappy Henry Jekyll to an end.

Her two sisters now recognized her for the beautiful stranger they had seen at the ball; and, falling at her feet, implored her forgiveness for their unworthy treatment, and all the insults they had heaped upon her head. Cinderella raised them, saying, as she embraced them, that she not only forgave them with all her heart, but wished for their affection. She was then taken to the palace of the young prince, in whose eyes she appeared yet more lovely than before, and who married her shortly after.

Cinderella, who was as good as she was beautiful, allowed her sisters to lodge in the palace, and gave them in marriage, that same day, to two lords belonging to the court.

법원의 법령은 왕자가 유리 구두를 착용 한 여성과 결혼 할 것을 선언합니다. CINDERELLA는 그녀의 섬세한 발을 그녀의 가족의 위대한 놀라움에 맞출 수있는 SLIPPER를 시도합니다.

그녀의 두 자매는 이제 그들이 공에서 본 아름다운 낯선 사람으로 그녀를 인정했다. 그리고 그녀의 발에 쓰러지면서, 합당하지 않은 대우와 그들이 머리에 쌓인 모욕에 대한 용서를 간청했습니다. 신데렐라는 그들을 포용하면서 마음을 다해 그들을 용서했을뿐 아니라 그들의 애정을 바라고 있다고 말했습니다. 그 후 그녀는 어린 왕자의 궁전으로 끌려 갔으며, 그의 눈에는 이전보다 더 사랑스러워 보였고 얼마 후에 그녀와 결혼했습니다.

그녀가 아름답게 좋았던 신데렐라는 자매들이 궁전에 묵을 수있게했고, 같은 날, 법원에 속한 두 군주와 결혼했습니다.