

# TEXT MINING

## Lecture 06

### TEXT PRE-PROCESSING II

---

**KEUNGOUI KIM**

*awekim@handong.edu*

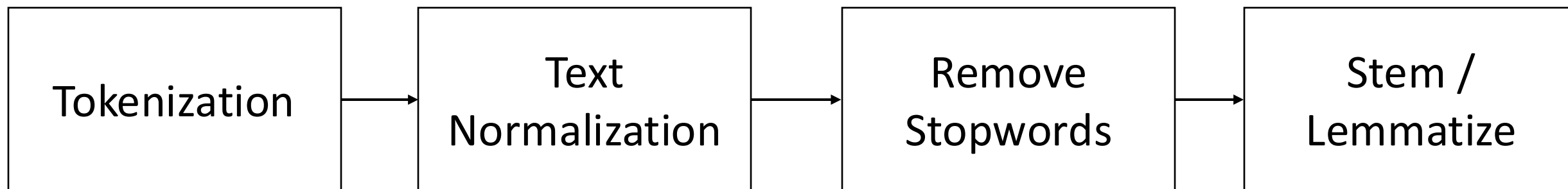


# *Text Pre-processing*

- As we know, text is unstructured data...
  - In other words, text itself cannot be used for analysis. Thus, we need to make it into a structured format.
- However, the pre-processing step is important even when conducting data analysis with structured data... why?
  - There is no perfect data. Without the pre-processing step, we may get an unexpected outcome.
- In a similar sense, text pre-processing is needed.
  - Transforming text data into a “structured” form
  - Correcting the errors and unnecessary or useless information

- In Text Mining, text pre-processing has two meanings
- 1) Creating tidy text data set
  - Text pre-processing is the process of transforming “unstructured” text data into “structured” text data.
  - Also, a unit of text data relevant to the goal of research should be maintained.
- 2) Dimension reduction
  - Text pre-processing should reduce the smaller text data set.
  - No duplicated or meaningless texts.

- Like any other typical data analysis, text pre-processing is the most time consuming and important process
- Text pre-processing includes any necessary steps needed for preparing the text data set
  - In general, it includes four major steps.
  - Any necessary steps can be added or skipped.
  - In actual practice, text pre-processing steps are repeated many times...
  - At the end, pre-processed text data becomes low dimensionality data.



# *Tokenization*

- Tokenization
  - Process of splitting text into “token”
  - Depending on the goal of text analysis, the unit of token either can be word, character, number, symbol, or n-grams.
  - String: full text
  - Word: classified character word
  - Term: normalized word (or also known as word)
  - Token: semantic unit of text
  - Type: all tokens included in the same string

- Simple way of running tokenization
  - Tokenize to 1) paragraph, 2) sentence, 3) word

son.wiki.doc <- "Son Heung-min is a South Korean professional footballer who plays as a forward for Premier League club Tottenham Hotspur and captains the South Korea national team. Considered one of the best forwards in the world and one of the greatest Asian footballers of all time, he is known for his explosive speed, finishing, two-footedness and ability to link play.

Born in Chuncheon, Gangwon Province, Son relocated to Germany to join Hamburger SV at 16, making his debut in the Bundesliga in 2010. In 2013, he moved to Bayer Leverkusen for a club record €10 million before signing for Tottenham for £22 million two years later, becoming the most expensive Asian player in history. While at Tottenham, Son became the top Asian goalscorer in both Premier League and Champions League history, and surpassed Cha Bum-kun's record for most goals scored by a Korean player in European competition. In 2019, he became the second Asian in history to reach and start a UEFA Champions League final after compatriot Park Ji-sung. In the 2021–22 season, he won, shared alongside Mohamed Salah, the Premier League Golden Boot award with 23 goals, becoming the first Asian player to win it."



- When tokenizing into a word, “ ” may not be the best solution.
  - There are some words whose meanings are better understood with other words or words themselves are originally written with other words.
  - The simplest way of identifying such cases is thinking of the “frequency” → If terms are more likely to appear within a combination of words, then analyzing the number of words as a unit of token can be more preferred.

```
> son.wiki.word[[1]]
```

[1]	"Son"	"Heung-min"	"is"	"a"	"South"
[6]	"Korean"	"professional"	"footballer"	"who"	"plays"
[11]	"as"	"a"	"forward"	"for"	"Premier"
[16]	"League"	"club"	"Tottenham"	"Hotspur"	"and"
[21]	"captains"	"the"	"South"	"Korea"	"national"
[26]	"team"				

*"professional footballer"*

*"South Korea" or "South", "Korea"*

*"Tottenham Hotspur" or "Tottenham", "Hotspur"*

- `ngram::ngram(x, n)`
  - Produces ngram object
  - `x`: string / `n`: number of grams

```
> son.wiki <- "Son Heung-min is a South Korean professional footballer who plays as a forward for Premier League club Tottenham Hotspur and captains the South Korea national team."
```

```
>  
> son.wiki.ng <- son.wiki %>%  
+   ngram(n=3)  
> class(son.wiki.ng)
```

```
[1] "ngram"  
attr(,"package")  
[1] "ngram"
```

```
> son.wiki.ng  
An ngram object with 24 3-grams
```

```
> str(son.wiki.ng)  
Formal class 'ngram' [package "ngram"] with 6 slots  
..@ str_ptr:<externalptr>  
..@ strlen : int 1  
..@ n      : int 3  
..@ ngl_ptr:<externalptr>  
..@ ngsz   : int 24  
..@ sl_ptr :<externalptr>
```

son.wiki <- "Son Heung-min is a South Korean professional footballer who plays as a forward for Premier League club Tottenham Hotspur and captains the South Korea national team."

```
> son.wiki.ng %>%  
+   print(output="truncated") #full  
South Korean professional | 1  
footballer {1} |  
  
Heung-min is a | 1  
South {1} |  
  
professional footballer who | 1  
plays {1} |  
  
and captains the | 1  
South {1} |  
  
a forward for | 1  
Premier {1} |  
  
[[ ... results truncated ... ]]
```

*Next possible words*



- Some useful functions from ngram package

- `ngram::get.phrasetable()`
- `ngram::get.ngrams()`
- `ngram::get.strings()`

```
> son.wiki.ng %>%  
+   get.phrasetable %>% head
```

	ngrams	freq	prop
1	South Korean professional	1	0.04166667
2	Heung-min is a	1	0.04166667
3	professional footballer who	1	0.04166667
4	and captains the	1	0.04166667
5	a forward for	1	0.04166667
6	for Premier League	1	0.04166667

*Frequency and  
proportion of ngrams*

```
> son.wiki.ng %>%
```

```
+   get.ngrams %>% head
```

```
[1] "South Korean professional"  
[2] "Heung-min is a"  
[3] "professional footballer who"  
[4] "and captains the"  
[5] "a forward for"  
[6] "for Premier League"
```

*Vector of ngrams*

```
> son.wiki.ng %>%
```

```
+   get.string %>% head
```

```
[1] "Son Heung-min is a South Korean professional footballer wh  
o plays as a forward for Premier League club Tottenham Hotspur  
and captains the South Korea national team."
```

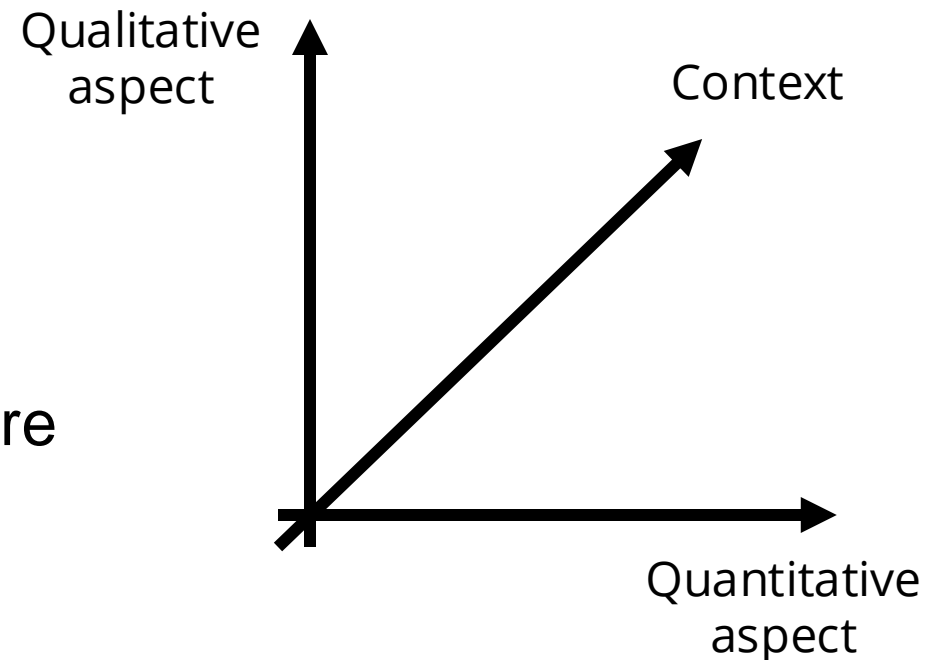
*Vector of string*

- `RWeka::NgramTokenizer(x, Weka_control)`
  - Returns tokenized strings
  - `x`: string
  - `Weka_control(min=i, max=j) →` Generate 2-grams and 3-grams

```
> son.wiki %>%
+   NGramTokenizer(
+     Weka_control(min=2, max=3))
[1] "Son Heung-min is"      "Heung-min is a"      "is a South"
[4] "a South Korean"        "South Korean professional" "Korean professional footballer"
[7] "professional footballer who" "footballer who plays" "who plays as"
[10] "plays as a"            "as a forward"        "a forward for"
[13] "forward for Premier"   "for Premier League"   "Premier League club"
[16] "League club Tottenham" "club Tottenham Hotspur" "Tottenham Hotspur and"
[19] "Hotspur and captains"  "and captains the"     "captains the South"
[22] "the South Korea"       "South Korea national" "Korea national team"
[25] "Son Heung-min"         "Heung-min is"         "is a"
[28] "a South"               "South Korean"         "Korean professional"
[31] "professional footballer" "footballer who"       "who plays"
[34] "plays as"              "as a"                 "a forward"
[37] "forward for"           "for Premier"          "Premier League"
[40] "League club"           "club Tottenham"       "Tottenham Hotspur"
[43] "Hotspur and"           "and captains"         "captains the"
[46] "the South"             "South Korea"          "Korea national"
[49] "national team"
```

# *Text Normalization I*

- Normalization
  - Organization of data to appear similar across all records and fields
- Text normalization
  - Process of reducing the complexity of text
  - Create a “simplified text”
- Three dimension of complex text
  - Quantitative aspect: length, frequency
  - Qualitative aspect: duplicated meaning, structure
  - Context: background knowledge, keywords



<http://www.nysed.gov/bilingual-ed/topic-brief-3-de-mystifying-complex-texts-what-are-complex-texts-and-how-can-we-ensure>

- Text normalization
  - Process of getting rid of meaningless text
  - Issue of digits → remove meaningless numbers
  - Issue of spaces → remove spaces
  - Issue of capital letters and small letters → tolower() or toupper()
  - Issue of symbols → remove punctuation
  - Issue of stopwords → remove stopwords

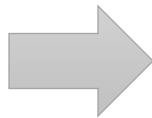
- Issue of spaces
  - In language, space is used to differentiate the words
  - Using space, we can easily create a vector of words. (using space as a separator)
  - As there is no perfect data, there is no perfect text data → error of multiple spaces

“God is good”



“God” “is” “good”

“God is good”



“God” “” “is” “good”

“God ” “is” “good”

“God” “ is” “good”



- Issue of spaces → handling multiple spaces

- stringr::str\_replace\_all()
- stringr::str\_squish()
- tm::stripWhitespace()

## Handling spaces in a vector

```
> hgu <- "God is good"
> hgu
[1] "God is good"
> hgu %>%
+ str_replace_all("[:space:]", " ")
[1] "God is good"
> hgu %>%
+ str_replace_all("[:space:]{1,}", " ")
[1] "God is good"
> hgu %>%
+ str_replace_all("[:space:]", "")
[1] "Godisgood"
> hgu %>%
+ str_squish
[1] "God is good"
```

## Handling spaces in a list

```
> hgu <- "God is good"
> hgu.word <-
+ hgu %>% str_split(" ")
> hgu.word
[[1]]
 [1] "God" "" "" "" "" "" "" "" "" ""
[11] "" "" "" "is" "" "" "" "" "" ""
[21] "good"

> hgu.word[[1]] %>%
+ str_replace_all("[:space:]", " ")
 [1] "God" "" "" "" "" "" "" "" "" ""
[11] "" "" "" "is" "" "" "" "" "" ""
[21] "good"

> hgu.word[[1]] %>%
+ str_squish()
 [1] "God" "" "" "" "" "" "" "" "" ""
[11] "" "" "" "is" "" "" "" "" "" ""
[21] "good"

> hgu.word[[1]][hgu.word[[1]]!=""]
[1] "God" "is" "good"
```

## \* Combinations

```
> hgu <- "God is good"
> hgu.word.not <-
+ hgu %>%
+ str_squish %>%
+ str_split(" ")
> hgu.word.not
[[1]]
 [1] "God" "is" "good"

> hgu.word.not %>% unlist
[1] "God" "is" "good"
```

- Issue of digits
  - Text can contain “numbers”
  - In general, numbers indicate quantity or time. But, if you are interested in names (ex. company name), such cases should be treated separately.

No. 7

No. Seven

Lucky Seven

7 people

Seven people

Seven Eleven

```
> numb.set <-  
+   c("7 players drank 7 bottles of wine on the 7th week")  
> numb.set %>%  
+   str_replace_all("[:digit:]", "")  
[1] " players drank  bottles of wine on the th week"  
> numb.set.word <-  
+   numb.set %>%  
+   str_split(" ")  
> numb.set.word[[1]] %>%  
+   str_replace_all("[:digit:]", "")  
[1] ""      "players" "drank"  ""      "bottles" "of"     "wine"   "on"  
[9] "the"    "th"     "week"
```

- Issue of symbols

- In text data, symbols or punctuation is often used to denote grammatical or symbolic meanings.
- The way punctuation is used differs by domain field.

“I love BTS, Bong Jun Ho, Son Heung Min and HGU.”

*Separating words*

*End of sentence.*

Samsung Electronics Co. Ltd.

*abbreviation*

Woo-to-the-Young-to-the-Woo

*Joining words*

#bitcoin #Teslar #dogecoin

*Hashtag - keywords*

# Text Normalization: Punctuation

- Issue of symbols
  - In general, punctuation is often removed → because when the texts are tokenized, punctuation does not have any meaning!
  - If necessary, use regular expressions to keep certain patterns of punctuation.
  - Be aware that some punctuations are not recognized by `[:punct:]`

```
> mark1.17 %>%  
+   str_view_all("[:punct:]")  
> mark1.17 <- "Come, follow me," Jesus said, "and I will send you out to fish for people."  
> mark1.17  
[1] "\"Come, follow me,\" Jesus said, \"and I will send you out to fish for people.\""  
> mark1.17 %>%  
+   str_view_all("[:punct:]")
```

*[:punct:] helps us to remove unexpected symbols generated by the program*

```
"Come," follow me," Jesus said, "and I will send you out to fish for people."
```

```
> mark1.17 %>%  
+   str_replace_all("[:punct:]", "")  
[1] "Come follow me Jesus said and I will send you out to fish for people"
```

# *Text Normalization II*

- Issue of upper & lower case letters
  - Recognizing the difference between "apple", "Apple", "APPLE"?
  - Computers recognize texts differently if upper and lowercase letters are not uniform
  - tolower(), toupper()

# Text Normalization: Upper & Lower Letters

- Problem of unifying letters
  - Check whether the text contains “names” or “pronouns”
  - If there are such cases, convert them into a unique form or remove them before conversion.

“Son Heung Min’s son  
plays football against  
Football Club Barcelona”

```
> son <- "Son Heung Min's son plays football against Football Club Barcelona"
> son %>%
+   str_extract_all(boundary("word"))
[[1]]
[1] "Son"      "Heung"    "Min's"    "son"      "plays"    "football" "against"
[8] "Football" "Club"     "Barcelona"
```

*boundary("word" or "sentence"  
or "character" or "line\_break")*

```
> son %>%
+   str_extract_all(boundary("word")) %>%
+   table
.
against Barcelona      Club football Football Heung Min's plays son
      1             1      1      1           1      1      1      1
Son
  1
```

```
> son %>%
+   tolower %>%
+   str_extract_all(boundary("word"))
[[1]]
[1] "son"      "heung"    "min's"    "son"      "plays"    "football" "against"
[8] "football" "club"     "barcelona"
```

```
> son %>%
+   tolower %>%
+   str_extract_all(boundary("word")) %>%
+   table
```

```
.
against barcelona      club football heung min's plays son
      1             1      1      2           1      1      1      2
```

*Is this correct?*

- Issue of stopwords

- List of words that have no significant meaning
- Such as pronouns (I, he, us, etc.), determiners (the, a), prepositions (in, on, etc.)
- Each source provides a different list of stopwords
- For some tasks, stopwords can be useful (ex. Getting an insight into people's personalities and behaviors (Mihalcea & Strapparava 2009))

```
> stopwords::stopwords_getsources()
[1] "snowball"      "stopwords-iso" "misc"          "smart"          "marimo"          "ancient"
[7] "nltk"          "perseus"
> stopwords::stopwords_getlanguages("nltk")
[1] "ar" "az" "da" "nl" "en" "fi" "fr" "de" "el" "hu" "id" "it" "kk" "ne" "no" "pt" "ro" "ru" "sl"
[20] "es" "sv" "tg" "tr"
> stopwords::stopwords("en", source = "nltk")
[1] "i" "me" "my" "myself" "we" "our" "ours"
[8] "ourselves" "you" "you're" "you've" "you'll" "you'd" "your"
[15] "yours" "yourself" "yourselves" "he" "him" "his" "himself"
[22] "she" "she's" "her" "hers" "herself" "it" "it's"
[29] "its" "itself" "they" "them" "their" "theirs" "themselves"
[36] "what" "which" "who" "whom" "this" "that" "that'll"
[43] "these" "those" "am" "is" "are" "was" "were"
[50] "be" "been" "being" "have" "has" "had" "having"
[57] "do" "does" "did" "doing" "a" "an" "the"
[64] "and" "but" "if" "or" "because" "as" "until"
[71] "while" "of" "at" "by" "for" "with" "about"
[78] "against" "between" "into" "through" "during" "before" "after"
[85] "above" "below" "to" "from" "up" "down" "in"
[92] "out" "on" "off" "over" "under" "again" "further"
[99] "then" "once" "here" "there" "when" "where" "why"
[106] "how" "all" "any" "both" "each" "few" "more"
[113] "most" "other" "some" "such" "no" "nor" "not"
[120] "only" "own" "same" "so" "than" "too" "very"
[127] "s" "t" "can" "will" "just" "don" "don't"
[134] "should" "should've" "now" "d" "ll" "m" "o"
[141] "re" "ve" "y" "ain" "aren" "aren't" "couldn"
[148] "couldn't" "didn" "didn't" "doesn" "doesn't" "hadn" "hadn't"
[155] "hasn" "hasn't" "haven" "haven't" "isn" "isn't" "ma"
[162] "mightn" "mightn't" "mustn" "mustn't" "needn" "needn't" "shan"
[169] "shan't" "shouldn" "shouldn't" "wasn" "wasn't" "weren" "weren't"
[176] "won" "won't" "wouldn" "wouldn't"
```

*List of stopwords*



- Issue of stopwords

- Exclude stopwords from the text data set using %ni% (opposite of %in%)

```
> '%ni%' <- Negate('%in%')
> steve.jobs.list <-
+   steve.jobs %>%
+   str_replace_all("[:punct:]", " ") %>%
+   str_replace_all("[:space:]", " ") %>% tolower() %>%
+   str_split(" ")
> steve.jobs.list
[[1]]
[1] "your"      "time"      "is"        "limited"    ""          "so"        "do"        "not"
[9] "waste"     "it"        "living"    "someone"   "else"      "s"         "life"      ""
```

```
> steve.jobs.list.1 <-
+   steve.jobs.list[[1]][
+     steve.jobs.list[[1]] %ni% stopwords("en", source = "nltk")]
> steve.jobs.list.1 <-
+   steve.jobs.list.1[steve.jobs.list.1 %ni% ""]
> steve.jobs.list.1
[1] "time"      "limited"    "waste"     "living"    "someone"   "else"      "life"
```

Remove  
stopwords

# Text Normalization: Stopword

- Issue of stopwords

- Based on the contextual background of the data, either modify or add to the stopwords list
- In other words, any words that are meaningless → most likely to be frequently used... (ex. Publication abstract)

abs.1 <- "The present study aims to explore how processes of local knowledge bases have been altered in this transformative environment and how these have impacted on local employment growth."

abs.2 <- "This study explores the relationship between the knowledge recombination types of exploitation and exploration and regional technical efficiency by using the empirical data sets combining EPO PATSTAT, Eurostat, and Cambridge Econometrics regional database."

abs.3 <- "For the empirical study, a structural equation modeling is employed by using survey material gathered from South Korea in the early days of the COVID-19 outbreak."

```
> abs.1.list <- abs.1 %>%  
+   str_replace_all("[:punct:]", " ") %>%  
+   str_replace_all("[:space:]", " ") %>%  
+   tolower() %>%  
+   str_split(" ")  
> abs.1.list <-  
+   abs.1.list[[1]][abs.1.list[[1]] %ni% ""]
```

```
> abs.2.list <- abs.2 %>%  
+   str_replace_all("[:punct:]", " ") %>%  
+   str_replace_all("[:space:]", " ") %>%  
+   tolower() %>%  
+   str_split(" ")  
> abs.2.list <-  
+   abs.2.list[[1]][abs.2.list[[1]] %ni% ""]
```

```
> abs.3.list <- abs.3 %>%  
+   str_replace_all("[:punct:]", " ") %>%  
+   str_replace_all("[:space:]", " ") %>%  
+   tolower() %>%  
+   str_split(" ")  
> abs.3.list <-  
+   abs.3.list[[1]][abs.3.list[[1]] %ni% ""]
```

```
> abs.all <- append(abs.1.list, abs.2.list) %>%  
+   append(abs.3.list)
```

*Combine all*

# Text Normalization: Stopword

- Issue of stopwords

## Before removing stopwords

```
> table(abs.all) %>% sort(decreasing=TRUE)
abs.all
```

the	and	of	study	by	empirical
7	4	3	3	2	2
have	how	in	knowledge	local	regional
2	2	2	2	2	2
this	using	19	a	aims	altered
2	2	1	1	1	1
bases	been	between	cambridge	combining	covid
1	1	1	1	1	1
data	database	days	early	econometrics	efficiency
1	1	1	1	1	1
employed	employment	environment	epo		
1	1	1	1		
exploitation	exploration	explore	explores		
1	1	1	1		
gathered	growth	impacted	is		
1	1	1	1		
modeling	on	outbreak	patstat		
1	1	1	1		
recombination	relationship	sets	south	st	
1	1	1	1		
technical	these	to	transformative		
1	1	1	1		

## After removing stopwords

```
> abs.all[abs.all %ni% stopwords("en", source = "nltk")] %>%
+ table %>% sort(decreasing=TRUE)
```

study	empirical	knowledge	local	regional	using
3	2	2	2	2	2
19	aims	altered	bases	cambridge	combining
1	1	1	1	1	1
covid	data	database	days	early	econometrics
1	1	1	1	1	1
efficiency	employed	employment	environment	epo	equation
1	1	1	1	1	1
eurostat	exploitation	exploration	explore	explores	gathered
1	1	1	1	1	1
growth	impacted	korea	material	modeling	outbreak
1	1	1	1	1	1
patstat	present	processes	recombination	relationship	sets
1	1	1	1	1	1
south	structural	survey	technical	transformative	types
1	1	1	1	1	1

*Frequently used  
words in  
publication...*

# Text Normalization: Stopword

- Issue of stopwords

## New stopwords list

```
> new.stopwords <- stopwords("en", source = "nltk") %>%  
+   append(c("study", "empirical"))
```

```
> new.stopwords
```

[1]	"i"	"me"	"my"	"myself"	"we"
[6]	"our"	"ours"	"ourselves"	"you"	"you're"
[11]	"you've"	"you'll"	"you'd"	"your"	"yours"
[16]	"yourself"	"yourselves"	"he"	"him"	"his"
[21]	"himself"	"she"	"she's"	"her"	"hers"
[26]	"herself"	"it"	"it's"	"its"	"itself"
[31]	"thou"	"them"	"their"	"theirs"	"themselves"
[151]	"doesn't"	"hadn't"	"hasn't"	"isn't"	"mustn't"
[156]	"hasn't"	"haven't"	"isn't"	"isn't"	"isn't"
[161]	"ma"	"mightn't"	"mustn't"	"mustn't"	"mustn't"
[166]	"needn't"	"needn't"	"shan't"	"shan't"	"shouldn't"
[171]	"shouldn't"	"wasn't"	"wasn't"	"weren't"	"weren't"
[176]	"won't"	"won't"	"wouldn't"	"wouldn't"	"wouldn't"
[181]	"empirical"				"study"

*New stopwords*

## Final list without added stopwords

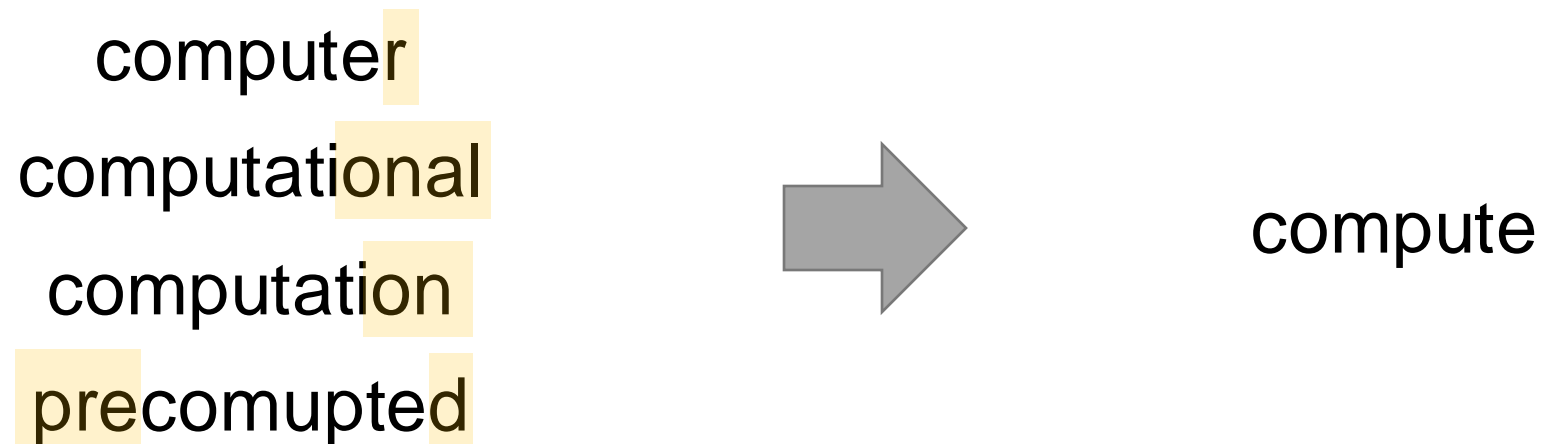
```
> abs.all[abs.all %ni% new.stopwords] %>%  
+   table %>% sort(decreasing=TRUE)
```

knowledge	local	regional	using	19
2	2	2	2	1
aims	altered	bases	cambridge	combining
1	1	1	1	1
covid	data	database	days	early
1	1	1	1	1
econometrics	efficiency	employed	employment	environment
1	1	1	1	1
epo	equation	eurostat	exploitation	exploration
1	1	1	1	1
explore	explores	gathered	growth	impacted
1	1	1	1	1
korea	material	modeling	outbreak	patstat
1	1	1	1	1
present	processes	recombination	relationship	sets
1	1	1	1	1
south	structural	survey	technical	transformative
1	1	1	1	1
types				
1				

# *Stem / Lemmatize*

- Stemming 어간 추출

- Process of identifying the common stem of words
- Removing suffixes and prefixes to obtain the stem



- Stemming often provides invalid words
- Stemmed text is not intended to be read by humans. It is better used in information retrieval systems.

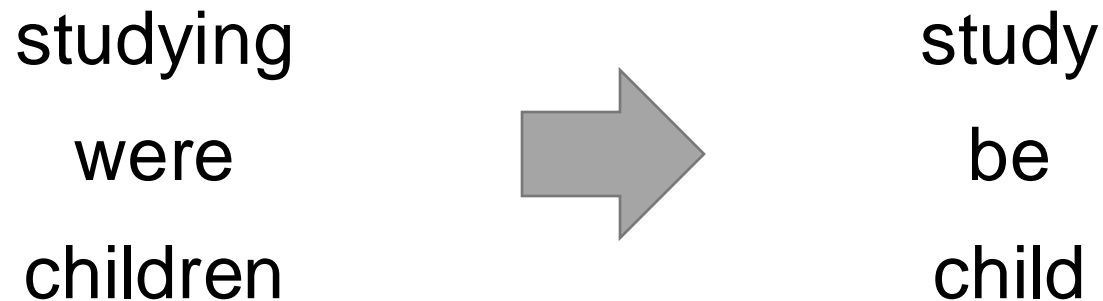
- `textstem::stem_words()`
  - Stem a vector of words

```
> abs.1 %>% str_split(" ") %>% .[[1]]
[1] "The"           "present"       "study"         "aims"          "to"
[6] "explore"       "how"           "processes"     "of"            "local"
[11] "knowledge"     "bases"        "have"          "been"          "altered"
[16] "in"            "this"          "transformative" "environment"   "and"
[21] "how"           "these"         "have"          "impacted"      "on"
[26] "local"         "employment"    "growth."

> abs.1 %>% str_split(" ") %>% .[[1]] %>%
+   stem_words()
[1] "The"           "present"       "studi"         "aim"           "to"            "explor"       "how"
[8] "process"       "of"            "local"         "knowledg"     "base"          "have"         "been"
[15] "alter"         "in"            "thi"           "transform"    "environ"       "and"          "how"
[22] "these"         "have"          "impact"        "on"           "local"         "employ"       "growth."
```

- Lemmatization 표제어 추출

- Process of reducing the inflectional forms of words to their root form.
- In other words, it removes inflectional endings only and returns the base or dictionary form of a word, which is known as the lemma.



- Unlike stemming, lemmatization provides valid words.
- However, it requires a more computationally intensive process as it requires a list of grammatical forms to handle the regular inflections as well as an extensive list of irregular words.



- `textstem::lemmatize_words()`
  - Lemmatize a vector of words








```
> abs.1 %>% str_split(" ") %>% .[[1]]
[1] "The"           "present"       "study"         "aims"          "to"
[6] "explore"       "how"           "processes"     "of"            "local"
[11] "knowledge"     "bases"         "have"          "been"          "altered"
[16] "in"            "this"          "transformative" "environment"   "and"
[21] "how"           "these"         "have"          "impacted"      "on"
[26] "local"         "employment"    "growth."

> abs.1 %>% str_split(" ") %>% .[[1]] %>%
+   lemmatize_words()
[1] "The"           "present"       "study"         "aim"           "to"
[6] "explore"       "how"           "process"       "of"            "local"
[11] "knowledge"     "base"         "have"          "be"           "alter"
[16] "in"            "this"          "transformative" "environment"   "and"
[21] "how"           "this"          "have"          "impact"        "on"
[26] "local"         "employment"    "growth."
```

# *Text Pre-processing Multiple Documents*

- In general, text mining deals with multiple documents systematically
  - For this purpose, text mining-oriented packages are often used.
  - tm package: R package specially designed for Text Mining
  - But of course, we can still do text pre-processing without the package

- Create a corpus object
  - tm::DirSource(): creates a directory source object
  - tm::VCorpus(): creates a corpus object with Source object
  - tm::as.VCorpus(): creates a corpus object with an R object
  - Useful when dealing with multiple text files

 2017_1	9/6/2022 12:48 AM	Text Document	2 KB
 2018_1	9/6/2022 12:47 AM	Text Document	1 KB
 2018_2	9/6/2022 12:47 AM	Text Document	2 KB
 2019_1	9/6/2022 12:45 AM	Text Document	2 KB
 2019_2	9/6/2022 12:46 AM	Text Document	2 KB
 2021_1	9/6/2022 12:38 AM	Text Document	2 KB
 2021_2	9/6/2022 12:39 AM	Text Document	2 KB

```
> options(encoding = "UTF-8")
> res.set <- VCorpus(DirSource("R data/awe_research"))
> summary(res.set)
```

	Length	Class	Mode
2017_1.txt	2	PlainTextDocument	list
2018_1.txt	2	PlainTextDocument	list
2018_2.txt	2	PlainTextDocument	list
2019_1.txt	2	PlainTextDocument	list
2019_2.txt	2	PlainTextDocument	list
2021_1.txt	2	PlainTextDocument	list
2021_2.txt	2	PlainTextDocument	list
2021_3.txt	2	PlainTextDocument	list
2021_4.txt	2	PlainTextDocument	list
2021_5.txt	2	PlainTextDocument	list
2021_6.txt	2	PlainTextDocument	list
2021_7.txt	2	PlainTextDocument	list
2021_8.txt	2	PlainTextDocument	list

- Create a corpus object
  - tm package's corpus object → Can access document by using the list element
  - content: text content
  - meta: meta data. Can be edited using the meta() function

```
> res.set[[1]]$content
```

```
[1] "This study examined effect of a policy intervention that provides an upper limit for handset subsidies on users' intention to change handset and households' expenses on mobile telecommunications. The Korean government has prohibited mobile network providers from providing excessive subsidies for mobile handsets to attract subscribers since Nov. 2014 according to the mobile act. Using the exogenous variation, we estimate the impact of the policy on the intention to change handsets and expenses on handset installment, total mobile communications, and online content. The longitudinal data are from the 2014 to 2015 waves of the Korea Media Panel Survey. The mobile act lowered the predicted probability of switching handsets by 0.4% points. Moreover, the mobile act increased the predicted probability of any expense on handset installment by 7.5% points and had a significant impact on the amount of expenses on handset installment, with an increase of 7.8%. The mobile act lowered users' willingness to switch handsets and increased spending on handset installment. This increased burden in handset installment might shrink the online content market, which has a large need for government support, as well as decrease consumers' welfare. We assert that the policy intervention on handset subsidies is questionable with regard to both consumer welfare and the healthiness of the ICT ecosystem."
```

```
> res.set[[1]]$meta
```

```
author      : character(0)
timestamp   : 2022-09-05 16:04:43
description  : character(0)
heading      : character(0)
id           : 2017_1.txt
language    : en
origin       : character(0)
```

# Text Pre-Processing with tm Package

- `tm::tm_map()` → Apply transformation functions
  - Similar to `lapply()`. Use with the following functions
  - `tm_map(corpus, removeNumbers)` → remove numbers
  - `tm_map(corpus, removePunctuation)` → remove punctuations
  - `tm_map(corpus, stripWhitespace)` → remove consecutive spaces
  - `tm_map(corpus, removeWords, wordvector)` → remove words (stopwords)
  - `tm_map(corpus, stemDocument)` → stemmization
  - `tm_map(corpus, content_transformer(function))` → use `content_transformer()` to apply functions. This is not needed when dealing with corpus generated from `as.VCorpus()`
- For lemmatize, use lemmatize functions from `textstem` package
  - `textstem::lemmatize_words()` → lemmatize a vector of words
  - `textstem::lemmatize_strings()` → lemmatize a vector of strings
  - For text pre-processing multiple documents, use `lemmatize_strings()`

# Text Pre-Processing with tm Package

- Two different versions
  - Stem vs. Lemmatize
  - Still punctuations?

## Stem

```
> res.set.sted <- res.set %>%  
+   tm_map(removePunctuation) %>%  
+   tm_map(removeNumbers) %>%  
+   tm_map(stripWhitespace) %>%  
+   tm_map(stemDocument) %>%  
+   tm_map(content_transformer(tolower))  
> res.set.sted[[1]]$content  
[1] "this studi examin effect of a polici intervent that  
sers' intent to chang handset and households' expens on  
mobil network provid from provid excess subsidi for mob  
to the mobil act using the exogen variat we estim the in  
set and expens on handset instal total mobil communic an  
e to wave of the korea media panel survey the mobil act  
point moreov the mobil act increas the predict probabl  
d a signific impact on the amount of expens on handset in  
ers' willing to switch handset and increas spend on hand  
l might shrink the onlin content market which has a larg  
sumers' welfar we assert that the polici intervent on ha  
onsum welfar and the healthi of the ict ecosystem"
```

WHY?

→ 'or'

## Lemmatize

```
> res.set.led <- res.set %>%  
+   tm_map(removePunctuation) %>%  
+   tm_map(removeNumbers) %>%  
+   tm_map(stripWhitespace) %>%  
+   tm_map(content_transformer(lemmatize_strings)) %>%  
+   tm_map(content_transformer(tolower))  
> res.set.led[[1]]$content  
[1] "this study examine effect of a policy intervention that  
on user's intention to change handset and household's expens  
overnment have prohibit mobile network provider from provide  
tract subscriber since nov accord to the mobile act use the  
of the policy on the intention to change handset and expens  
nication and online content the longitudinal datum be from th  
the mobile act lower the predict probability of switch handse  
se the predict probability of any expense on handset install  
on the amount of expense on handset installment with a incre  
ness to switch handset and increase spend on handset install  
lment may shrink the online content market which have a large  
crease consumer's welfare we assert that the policy intervent  
h regard to both consumer welfare and the healthiness of the
```



# Text Pre-Processing with tm Package

- Using `tm_map()`, we can simply do text-preprocessing tasks with multiple documents
  - Pre-processing with specialized packages makes our lives easy, but need to consider the following things

```
> removeSpecialChars <-  
+   function(x) gsub("'", "", x)  
> res.set.led <- res.set %>%  
+   tm_map(removePunctuation) %>%  
+   tm_map(removeNumbers) %>%  
+   tm_map(stripWhitespace) %>%  
+   tm_map(content_transformer(lemmatize_strings)) %>%  
+   tm_map(content_transformer(tolower)) %>%  
+   tm_map(content_transformer(removeSpecialChars))  
> res.set.led[[1]]$content  
[1] "this study examine effect of a policy intervention that provide a upper limit for handset subsidy on user intention to change handset and household expense on mobile telecommunication the korean government have prohibit mobile network provider from provide excessive subsidy for mobile handset to attract subscriber since nov accord to the mobile act use the exogenous variation we estimate the impact of the policy on the intention to change handset and expense on handset installment total mobile communication and online content the longitudinal datum be from the to wave of the korea medium panel survey the mobile act lower the predict probability of switch handset by point moreover the mobile act increase the predict probability of any expense on handset installment by point and have a significant impact on the amount of expense on handset installment with a increase of the mobile act lower user willingness to switch handset and increase spend on handset installment this increase burden in handset installment may shrink the online content market which have a large need for government support as good as decrease consumer welfare we assert that the policy intervention on handset subsidy be questionable with regard to both consumer welfare and the healthiness of the ict ecosystem"
```



# Text Pre-Processing with tm Package

- Three tokenization options
  - Boost\_tokenizer(), MC\_tokenizer(), scan\_tokenizer()

```
> res.set[[2]]$content %>%  
+   Boost_tokenizer %>% head(30)
```

[1]	"As"	"more"	"technologies"	"and"	"industries"
[6]	"converge,"	"technology"	"standards"	"are"	"more"
[11]	"likely"	"to"	"be"	"a"	"strategic"
[16]	"factor"	"for"	"firms"	"and"	"governments"
[21]	"that"	"are"	"interested"	"in"	"the"
[26]	"market"	"with"	"standards-based"	"competition."	"From"

```
> res.set[[2]]$content %>%  
+   MC_tokenizer %>% head(30)
```

[1]	"As"	"more"	"technologies"	"and"	"industries"	"converge"
[7]	"technology"	"standards"	"are"	"more"	"likely"	"to"
[13]	"be"	"a"	"strategic"	"factor"	"for"	"firms"
[19]	"and"	"governments"	"that"	"are"	"interested"	"in"
[25]	"the"	"market"	"with"	"standards"	"based"	"competition"

```
> res.set[[2]]$content %>%  
+   scan_tokenizer %>% head(30)
```

[1]	"As"	"more"	"technologies"	"and"	"industries"
[6]	"converge,"	"technology"	"standards"	"are"	"more"
[11]	"likely"	"to"	"be"	"a"	"strategic"
[16]	"factor"	"for"	"firms"	"and"	"governments"
[21]	"that"	"are"	"interested"	"in"	"the"
[26]	"market"	"with"	"standards-based"	"competition."	"From"

# Text Pre-Processing with tm Package

- User defined-function can be applied to text pre-processing

```
> space_tokenizer <- function(x){  
+   unlist(strsplit(as.character(x), "[[:space:]]+"))  
+ }  
  
> res.set.ted <- res.set %>%  
+   tm_map(removePunctuation) %>%  
+   tm_map(removeNumbers) %>%  
+   tm_map(stripWhitespace) %>%  
+   tm_map(content_transformer(lemmatize_strings)) %>%  
+   tm_map(content_transformer(tolower)) %>%  
+   tm_map(content_transformer(removeSpecialChars)) %>%  
+   tm_map(content_transformer(removeWords),  
+     stopwords::stopwords('en', source='stopwords-iso')) %>%  
+   tm_map(content_transformer(str_squish)) %>%  
+   tm_map(content_transformer(space_tokenizer))  
  
> res.set.ted[[1]]$content  
[1] "study" "examine" "policy" "intervention"  
[5] "provide" "upper" "limit" "handset"  
[9] "subsidy" "user" "intention" "change"  
[13] "handset" "household" "expense" "mobile"  
[17] "telecommunication" "korean" "government" "prohibit"
```

*Remove stopwords*

# *Text Pre-processing*

## *Check Points*

# Text Pre-processing Check Points

- Check the connected words with symbols

- Ex) Woo-to-the-Young-to-the-Woo → WoototheYoungtotheWoo / a.k.a → aka

- Ex) default-setting → defaultsetting / and/or → andor

- Ex) 12.1 → 121

*Based on the domain knowledge, determine whether to remove '-' to keep it as a single word, or separate them as two words*

```
> res.set[[17]]$content
```

```
[1] "This paper aims to uncover the mechanism of how the network properties of regional knowledge spaces contribute to technological change from the perspective of regional knowledge entry-relatedness and regional knowledge entry-potential. Entry-relatedness, which has been previously employed to investigate the technology evolution of regional economies, is advanced by introducing a knowledge gravity model. The entry-potential of a newly acquired regional specialisation has been largely ignored in the relevant literature; surprisingly given the high relevance that is attributed to the recombination potential of new capabilities. In other words, just adding new knowledge domains to a system is not sufficient alone, it really depends on how these fit into the existing system and thus can generate wider economic benefits. Based on an empirical analysis of EU-15 Metro and non-Metro regions from 1981 to 2015, we find that entry-relatedness has a significant negative association with novel inventive activities, while entry-potential has a significant positive association with the development of novel products and processes of economic value. This highlights that regions' capacity to venture into high-potential areas of technological specialization in the knowledge space outperforms purely relatedness driven diversification that is frequently promoted in the relevant literature."
```

```
> res.set.led[[17]]$content
```

```
[1] "this paper aim to uncover the mechanism of how the network property of regional knowledge space contribute to technological change from the perspective of regional knowledge entryrelatedness and regional knowledge entrypotential. entryrelatedness which have be previously employ to investigate the technology evolution of regional economy be advance by introduce a knowledge gravity model the entrypotential of a newly acquire regional specialisation have be largely ignore in the relevant literature surprisingly give the high relevance that be attribute to the recombination potential of new capability in other word just add new knowledge domain to a system be not sufficient alone it really depend on how this fit into the exist system and thus can generate wide economic benefit base on a empirical analysis of leu metro and nonmetro region from to we find that entryrelatedness have a significant negative association with novel inventive activity while entrypotential al have a significant positive association with the development of novel product and process of economic value this highlight that region ' capacity to venture into highpotential area of technological specialization in the knowledge space outperform purely relatedness drive diversification that be frequently promote in the relevant literature"
```

# Text Pre-processing Check Points

- Simple way to find cases with symbols

```
> res.set[[17]]$content %>%  
+   str_extract_all("[[:alnum:]]{1,}[[:punct:]]{1}?[[:alnum:]]{1,}")
```

*Alphabets or numbers  
appeared at least once*

*Punctuations  
appeared once*

*Alphabets or numbers  
appeared at least once*

```
[[1]]  
[1] "entry-relatedness" "entry-potential" "Entry-relatedness" "entry-potential" "EU-15"  
[6] "non-Metro" "entry-relatedness" "entry-potential" "high-potential"
```

# Text Pre-processing Check Points

- Simple way to find cases with symbols
  - Create a user-defined function + use it with lapply()
  - Check how punctuations are used in each document → Make a decision

```
> ExtractPuncWords <- function(x){  
+   str_extract_all(x$content,  
+                   "[[:alnum:]]{1,}[[:punct:]]{1}?[[:alnum:]]{1,}")  
+ }
```

```
> lapply(res.set, ExtractPuncWords)
```

```
$`2017_1.txt`
```

```
$`2017_1.txt`[[1]]
```

```
[1] "0.4" "7.5" "7.8"
```

```
$`2018_1.txt`
```

```
$`2018_1.txt`[[1]]
```

```
[1] "standards-based" "HD-DVD" "Blu-ray" "firm's" "firm's" "Toshiba's"
```

```
$`2018_2.txt`
```

```
$`2018_2.txt`[[1]]
```

```
[1] "support's"
```

# Text Pre-processing Check Points

- Check the words with numbers
  - Some numbers may have a significant meaning

*Based on the domain knowledge, determine whether to remove numbers or not*

```
> res.set[[4]]$content
```

```
[1] "The implacable list of diversification indices allows a wide range of selection opportunities for the researchers. The absence of consensus on the selection of suitable technology diversification index, however, may lead to a lack of objectivity with ample grounds. In this study, we focus on the case of technology diversification using patent to derive empirical implication for selecting suitable diversification index. To obtain the content validity of diversification index, three cases were tested: cross section, single and multiple time periods. As a result, diversification indices are separated into two groups: HHI, Gini-Simpson, 1/HHI, and Entropy for PC1 and Variety and Rao-Stirling for PC2. In this context, technology diversification can be explained by two perspectives of diversification: balance-centered and hetero-centered diversification. The balance-centered diversification implies the proportion of elements are the target of interest while hetero-centered diversification refers to variety and disparity of diversification, which focuses on the degree of differentiation among elements. In applicant-level technology diversification studies, these two diversification perspectives are recommended to be used."
```

```
> res.set.led[[4]]$content
```

```
[1] "the implacable list of diversification index allow a wide range of selection opportunity for the researcher the absence of consensus on the selection of suitable technology diversification index however may lead to a lack of objectivity with ample ground in this study we focus on the case of technology diversification use patent to derive empirical implication for select suitable diversification index to obtain the content validity of diversification index three case be test cross section single and multiple time period as a result diversification index be separate into two group hhi ginisimpson hhi and entropy for pc and variety and raostirling for pc in this context technology diversification can be explain by two perspective of diversification balancecentered and heterocentered diversification the balancecentered diversification imply the proportion of element be the target of interest while heterocentered diversification refer to variety and disparity of diversification which focus on the degree of differentiation among element in applicantlevel technology diversification study this two diversification perspective be recommend to be use"
```

# Text Pre-processing Check Points

- Simple way to find cases with numbers

```
> res.set[[4]]$content %>%  
+   str_extract_all("[[:graph:]]{0,}[[:digit:]]{1,}[[:graph:]]{0,}")
```

*Alphabets or numbers or  
punctuations appeared at  
least 0 times*

*Why 0??*

*Numbers  
appeared at least  
once*

*Alphabets or numbers or  
punctuations appeared at  
least 0 times*

*Setting {0,}*

*→ we can consider any  
cases that start or finish  
with the number*

```
> res.set[[4]]$content %>%  
+   str_extract_all("[[:graph:]]{0,}[[:digit:]]{1,}[[:graph:]]{0,}")  
[[1]]  
[1] "1/HHI," "PC1"      "PC2."
```

```
> res.set[[4]]$content %>%  
+   str_extract_all("[[:graph:]]{1,}[[:digit:]]{1,}[[:graph:]]{1,}")  
[[1]]  
[1] "PC2."
```



# Text Pre-processing Check Points

- Simple way to find cases with numbers
  - Create a user-defined function + use it with lapply()
  - Check how numbers are used in each document → Make a decision

```
> ExtractNumText <- function(x){  
+   str_extract_all(x$content,  
+                   "[[:graph:]]{0,}[[:digit:]]{1,}[[:graph:]]{0,}")  
+ }
```

```
> lapply(res.set, ExtractNumText)
```

```
$`2017_1.txt`
```

```
$`2017_1.txt`[[1]]
```

```
[1] "2014" "2014" "2015" "0.4%" "7.5%" "7.8%."
```

```
$`2018_1.txt`
```

```
$`2018_1.txt`[[1]]
```

```
character(0)
```

```
$`2018_2.txt`
```

```
$`2018_2.txt`[[1]]
```

```
[1] "296"
```

```
$`2019_1.txt`
```

```
$`2019_1.txt`[[1]]
```

```
[1] "1/HHI," "PC1" "PC2."
```

# Text Pre-processing Check Points

- Check the words with capital letters
  - Words with capital letters may indicate the name, abbreviation, or first word of a sentence, etc.

```
> res.set[[16]]$content
```

```
[1] "Disruptive advancements in science and technology often rely on new ideas and findings, which in turn brings us to focus on the value of novelty in scholarly activities. Using Web of Science publication data from European regions for the period between 2008 and 2017, this study examines, first, the impact of scientific collaboration on novelty of research. Here, five levels of collaboration are considered for each article"
```

```
> res.set.ed[[16]]$content
```

```
[1] "disruptive advancement in science and technology often rely on new idea and finding which in turn bring us to focus on the value of novelty in scholarly activity use web of science publication datum from European region for the period between and this study examine first the impact of scientific collaboration on novelty of research here five level of collaboration be consider for each article"
```

# Text Pre-processing Check Points

- Simple way to find cases with capital letters

```
> res.set[[16]]$content %>%  
+   str_extract_all("[[:upper:]]{1}[[:alnum:]]{1,}")
```

*Capital letters appeared  
once*

*Alphabets or numbers  
appeared at least once*

```
[[1]]  
[1] "Disruptive" "Using"      "Web"        "Science"    "European"   "Here"
```

# Text Pre-processing Check Points

- Simple way to find cases with capital letters
  - Create a user-defined function + use it with lapply()
  - Check how capital letters are used in each document → Make a decision

```
> ExtractCapText <- function(x){
+   str_extract_all(x$content,
+                   "[[:upper:]]{1}[[:alnum:]]{1,}")
+ }
> lapply(res.set, ExtractCapText)
$`2017_1.txt`
$`2017_1.txt`[[1]]
[1] "This"      "The"      "Korean"   "Nov"      "Using"    "The"      "Korea"    "Media"    "Panel"    "Survey"
[11] "The"      "Moreover" "The"      "This"     "We"       "ICT"
$`2018_1.txt`
$`2018_1.txt`[[1]]
[1] "As"      "From"    "In"      "HD"      "DVD"     "Blu"     "With"    "Toshiba"
$`2018_2.txt`
$`2018_2.txt`[[1]]
[1] "This"      "Binh"     "Dinh"     "Vietnam"  "AT"       "AT"       "This"    "Binh"
[9] "Dinh"      "The"      "UTAUT"    "The"      "AT"       "Thus"     "AT"      "Furthermore"
[17] "AT"       "Finally"
```

# *Value of Text Pre-processing*

- Comparison of non-preprocessed text and preprocessed text
  - Less number of words → text data dimension reduction

```
> res.set.ted <- res.set %>%
+   tm_map(removePunctuation) %>%
+   tm_map(removeNumbers) %>%
+   tm_map(stripWhitespace) %>%
+   tm_map(content_transformer(lemmatize_strings)) %>%
+   tm_map(content_transformer(tolower)) %>%
+   tm_map(content_transformer(removeSpecialChars)) %>%
+   tm_map(content_transformer(removeWords),
+     stopwords::stopwords('en', source='stopwords-iso')) %>%
+   tm_map(content_transformer(space_tokenizer))
> res.set.ted.wo <- res.set %>%
+   tm_map(content_transformer(space_tokenizer))
```

```
> res.set.ted.wo[[1]]$content %>% length
[1] 211
> res.set.ted[[1]]$content %>% length
[1] 108
> res.set.ted.wo[[1]]$content %>% unique %>% length
[1] 116
> res.set.ted[[1]]$content %>% unique %>% length
[1] 62
```

```
> res.set.ted.wo.corp <- NA
> res.set.ted.corp <- NA
> for(i in 1:length(res.set.ted)){
+   res.set.ted.wo.corp <-
+     c(res.set.ted.wo.corp, res.set.ted.wo[[i]]$content)
+   res.set.ted.corp <-
+     c(res.set.ted.corp, res.set.ted[[i]]$content)
+ }
> res.set.ted.wo.corp <-
+   res.set.ted.wo.corp[res.set.ted.wo.corp != 0]
> res.set.ted.corp <-
+   res.set.ted.corp[res.set.ted.corp != 0]
```

```
> res.set.ted.wo.corp %>% length
[1] 3801
> res.set.ted.corp %>% length
[1] 1931
> res.set.ted.wo.corp %>%
+   unique %>% length
[1] 1363
> res.set.ted.corp %>%
+   unique %>% length
[1] 710
```

Total word frequency

Total “unique” word frequency

- Comparison of non-preprocessed text and preprocessed text
  - More meaningful text can be captured

```
> res.set.ted.wo.corp %>%  
+   table %>% data.frame %>%  
+   dplyr::arrange(desc(Freq)) %>%  
+   head(10)
```

		Freq
1	the	208
2	of	147
3	and	145
4	to	72
5	in	58
6	a	56
7	on	46
8	knowledge	39
9	is	38
10	by	35

```
> res.set.ted.corp %>%  
+   table %>% data.frame %>%  
+   dplyr::arrange(desc(Freq)) %>%  
+   head(10)
```

		Freq
1	knowledge	52
2	diversification	39
3	technology	38
4	study	30
5	product	24
6	firm	23
7	patent	22
8	regional	22
9	analysis	19
10	innovation	19

# Rearrange Text Pre-processing Task

- Based on this finding, determine the “order” and function of the text pre-processing task
  - Text pre-processing for specific-cases may be needed
  - Would “fixing all issues” be the best solution? We have to think about the efficiency of work

**Fix all issues**

3 hours

**Run typical text pre-processing**

10 minutes

**Fix some critical issues**

xx hours

*Proportion of cases  
→ experience +  
domain knowledge*