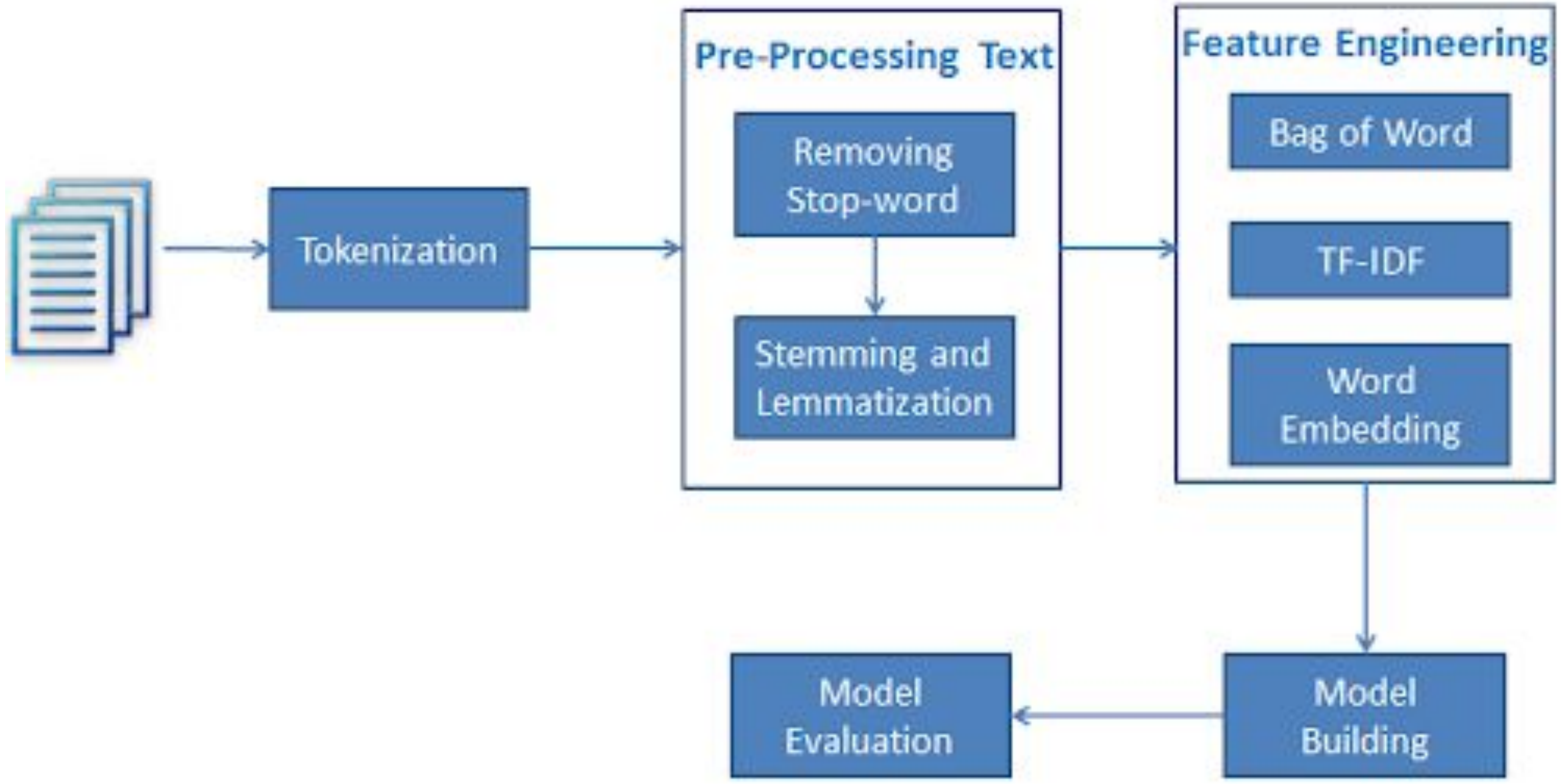
Day 11

Text Analytics

Text Analytics Process



nltk

- A platform for building Python programs to work with human language data 파이썬에 있는 언어처리를 위한 플랫폼 (라이브러리)
- Provides interfaces to over 50 corpora and lexical resources such as WordNet 워드넷같은 50개의 사전과 어휘자원에 대한 인터페이스 제공
- Provides a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries 분류, 토큰화, 스테밍, 테깅, 파싱, 의미추론, 래퍼등과 같은 문자처리 라이브러리를 제공

Installing nltk

- You can install the nltk package on the command line
  명령어 프롬프트에서 다음 명령어를 실행하여 패키지 설치

```
pip install nltk
python -m
nltk.downloader
popular
```

- After installing the nltk package, install the necessary datasets/models.
  파이썬에서 패키지를 불러옴

```
import nltk
help(nltk)
nltk.download()
nltk.download(‘all’)
nltk.download('popular')
```

Tokenization 토큰화

- The process of breaking down a text into words. 텍스트를 잘라주는 과정
  - Sentence Tokenization 문장토큰화
  - Word Tokenization 단어토큰화

Sentence Tokenization

```
from nltk.tokenize import sent_tokenize

text="""Hello Mr. Smith, how are you doing today? The
weather is great, and city is awesome.

The sky is pinkish-blue. You shouldn't eat cardboard"""

tokenized_sent=sent_tokenize(text)

print(tokenized_sent)
```

Word Tokenization

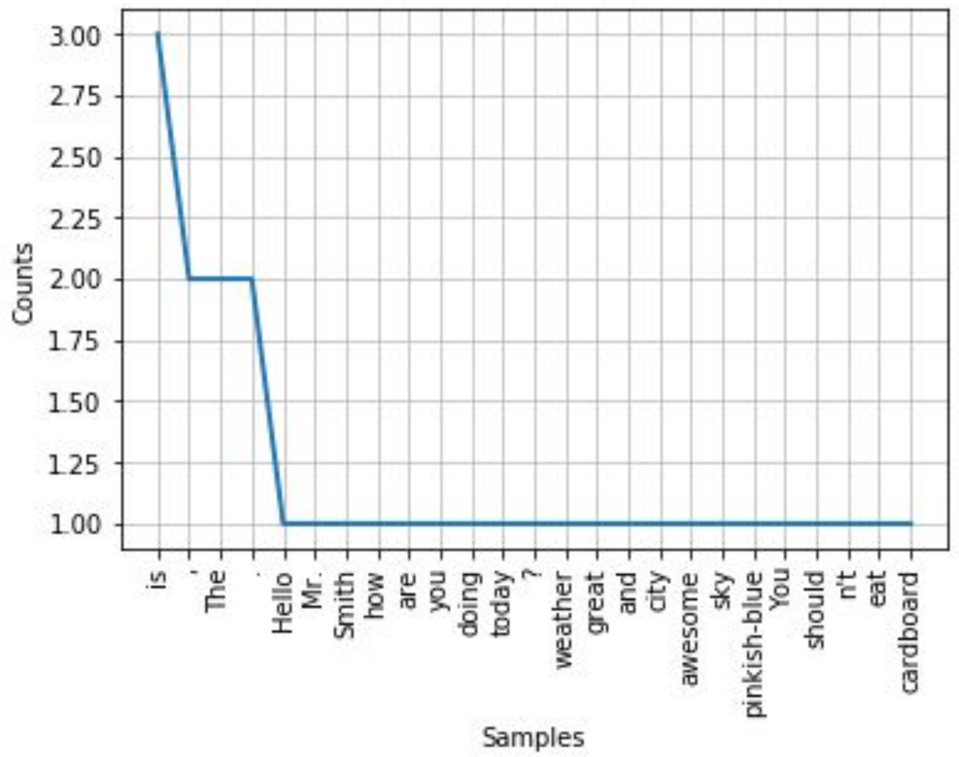
```
from nltk.tokenize import
word_tokenize

tokenized_word=word_tokenize(text)

print(tokenized_word)
```

Frequency Distribution

```
from nltk.probability import FreqDist
fdist = FreqDist(tokenized_word)
fdist
fdist.most_common(2) #상위2개
```

Frequency Distribution Plot

```
import
matplotlib.pyplot as
plt

fdist.plot(30,cumula
tive=False)
#or fdist.plot(30)

plt.show()
```



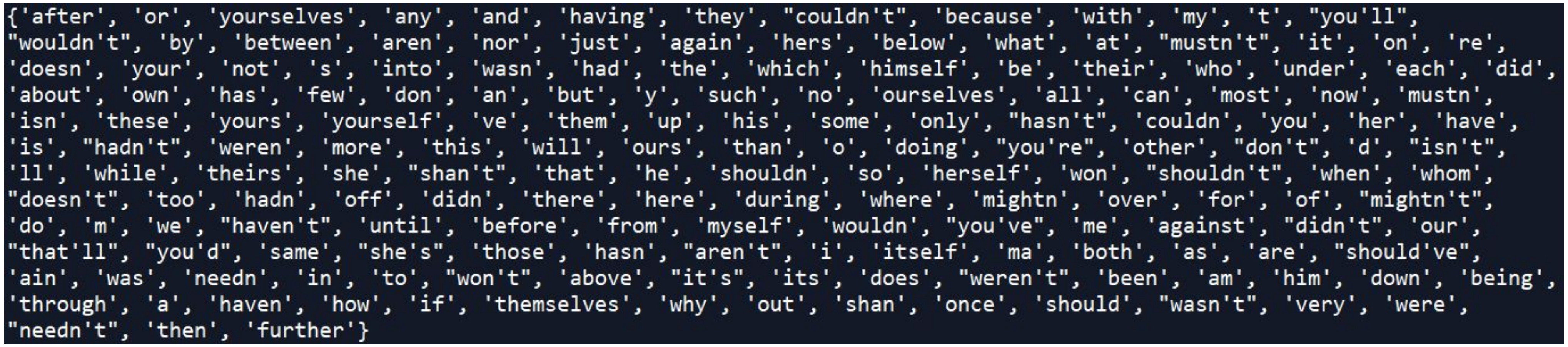
Exercise #11
Single Paragraph Example

- Split the text into individual words and create a frequency table and plot
  다음 글을 단어로 쪼개서 빈도테이블과 그래프 그리기
- Split the text into sentences and tokenize the sentences and count the number of words. Draw the bar plot. 먼저 문장으로 쪼개고 다시 문장을 단어로 쪼개서 문장별 단어의 갯수를 카운트하고 막대그래프 그리기

```
"Now, I understand that because it's an election season expectations for what we will
achieve this year are low But, Mister Speaker, I appreciate the constructive approach
that you and other leaders took at the end of last year to pass a budget and make tax
cuts permanent for working\
families. So I hope we can work together this year on some bipartisan priorities like
criminal justice reform and helping people who are battling prescription drug abuse
and heroin abuse. So, who knows, we might surprise the cynics again"
```

Stopwords

```
from nltk.corpus import stopwords
stop_words=set(stopwords.words("english"))
print(stop_words)
len(stop_words)
```



```
{'after', 'or', 'yourselves', 'any', 'and', 'having', 'they', "couldn't", 'because', 'with', 'my', 't', "you'll",
"wouldn't", 'by', 'between', 'aren', 'nor', 'just', 'again', 'hers', 'below', 'what', 'at', "mustn't", 'it', 'on', 're',
'doesn', 'your', 'not', 's', 'into', 'wasn', 'had', 'the', 'which', 'himself', 'be', 'their', 'who', 'under', 'each', 'did',
'about', 'own', 'has', 'few', 'don', 'an', 'but', 'y', 'such', 'no', 'ourselves', 'all', 'can', 'most', 'now', 'mustn',
'isn', 'these', 'yours', 'yourself', 've', 'them', 'up', 'his', 'some', 'only', "hasn't", 'couldn', 'you', 'her', 'have',
'is', "hadn't", 'weren', 'more', 'this', 'will', 'ours', 'than', 'o', 'doing', "you're", 'other', "don't", 'd', "isn't",
'll', 'while', 'theirs', 'she', "shan't", 'that', 'he', 'shouldn', 'so', 'herself', 'won', "shouldn't", 'when', 'whom',
"doesn't", 'too', 'hadn', 'off', 'didn', 'there', 'here', 'during', 'where', 'mightn', 'over', 'for', 'of', "mightn't",
'do', 'm', 'we', "haven't", 'until', 'before', 'from', 'myself', 'wouldn', "you've", 'me', 'against', "didn't", 'our',
"that'll", "you'd", 'same', "she's", 'those', 'hasn', "aren't", 'i', 'itself', 'ma', 'both', 'as', 'are', "should've",
'ain', 'was', 'needn', 'in', 'to', "won't", 'above', "it's", 'its', 'does', "weren't", 'been', 'am', 'him', 'down', 'being',
'through', 'a', 'haven', 'how', 'if', 'themselves', 'why', 'out', 'shan', 'once', 'should', "wasn't", 'very', 'were',
"needn't", 'then', 'further'}
```

Removing Stopwords

```
filtered_word=[]
for w in tokenized_word:
    if w not in stop_words:
        filtered_word.append(w)
print("Tokenized Word:",tokenized_word)
print("Filterd Word:",filtered_word)
```

Stemming

- A crude way of chopping of an end to get base word and often includes removal of derivational affixes. **어미의 끝을 자르는 방법, 주로 접미사를 자름**
- The most common algorithm used for the purpose is Porter’s Algorithm. **포터의 알고리즘이 가장 많이 쓰임**

```
from nltk.stem import
PorterStemmer

ps = PorterStemmer()

ps.stem('flying')
```

Stemming Words

```
stemmed_word=[]
for word in filtered_word:
    stemmed_word.append(ps.stem(word))
print("Filtered Word:",filtered_word)
print("Stemmed Word:",stemmed_word)
```

Lemmatization

- Performs vocabulary and morphological analysis of the word and is normally aimed at removing inflectional endings only. **단어의 동질이형분석. 단어끝을 제거해서 원형으로 돌려줌**

```
from
nltk.stem.wordnet
import
WordNetLemmatizer

lem =
WordNetLemmatizer()

lem.lemmatize('flyin
g', 'v')
```

Part-of-Speech(POS) Tagging

- Identify what words act as nouns, pronouns, verbs, adverbs, etc. **무슨 단어가 어떤 품사의 역할을 하는지 식별**

```
sent = "Albert Einstein was born in Ulm, Germany in
1879."

tokens=nltk.word_tokenize(sent)

print(tokens)

nltk.pos_tag(tokens)
```

POS Tags

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun

Tag	Description
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Whdeterminer
WP	Whpronoun
WP$	Possessive whpronoun
WRB	Whadverb

Removing Punctuation

```
import string

string.punctuation


''.join([char for char in text if char not in
string.punctuation])
```

Removing Punctuation

```
import string
string.punctuation
nopunc_word = [word for word in
stemmed_word if word not in
string.punctuation]
```

Cleaning Text (Altogether)

- Remove punctuation, tokenize, remove stopwords, and stem 문장부호를 업애고, 토큰으로 나누고, 스탑워드를 없앤후, 어근으로 분리

```
text="Hello Mr. Smith, how are you doing today? \
The weather is great, and city is awesome.\
The sky is pinkish-blue. You shouldn't eat cardboard."
def clean_text(text):
    text = "".join([word.lower() for word in text if word not in string.punctuation])
    tokens = re.split('\W+', text)
    text = [ps.stem(word) for word in tokens if word not in stop_words]
    return text
clean_text(text)
```

Word Cloud Example

```
from PIL import Image

from wordcloud import WordCloud, STOPWORDS,
ImageColorGenerator

import matplotlib.pyplot as plt


text = "whatever text…."

wordcloud = WordCloud().generate(text)

plt.imshow(wordcloud,
interpolation='bilinear') #Make the displa
image appear more smoothly.

plt.axis("off")

plt.show()
```

understand
reform
make
achieve
last
surprise
approachthat
budget
work
year
hope
end
leaders
prescription
heroin
permanent
will
Speaker
criminal
thecynics
priorities
knows
together
abuse
workingfamilies
pass
lowBut
cuts
appreciate
justice
drug
took
andhelping
tax
yearto
people
battling
seasonexpectations
Now
abuseand
somebipartisan
election
might
constructive
Mister

Word Cloud Options

```
wordCloud(max_font_size=
50, max_words=100,
background_color="white“
,stopwords=stopwords).ge
nerate(text)
```

```
wordcloud.to_file("wordc
loud.png")
```

approachthat
battling
budget
together
reform
tax
seasonexpectations
appreciate
heroin
people
abuse
constructive
last
yearto
knows
prescription
election
year
permanent
thecynics
abuseand
took
Mister
Now
drug
achieve
work
andhelping
somebipartisan
priorities
surprise
leaders
end
understand
will
pass
make
lowBut
hope
might
Speaker
justice
criminal
workingfamilies
cuts

Exercise #11

- Using the text on http://programminghistorian.github.io/ph-submissions/assets/basic-text-processing-in-r/sotu_text/236.txt, draw a wordcloud
  - Urllib.request.urlopen.read or requests.get
  - BeautifulSoup.get_text

Web Scraping

- A computer software technique of extracting information from websites 웹사이트에서 정보를 가져오는 기술
- Mostly focuses on the transformation of unstructured data (HTML format) on the web into structured data (database or spreadsheet) 비구조화된 데이터를 가져와서 구조화된 데이터의 형식으로 전환

Python Libraries for Web Scraping

- Requests
  - A python module which can be used for requesting data to a webserver 웹서버에서 데이터를 가져올때 쓰는 모듈
- BeautifulSoup
  - An incredible tool for pulling out information from a webpage 웹페이지에서 정보를 가져오는 툴
  - Extract tables, lists, paragraph and also put filters to extract information from web pages 테이블, 리스트, 문장을 가져와서 필터를 적용
- Code

```
Import requests
from bs4 import BeautifulSoup
```

Get Request

- Query the website and return the html to the variable 'page‘ **웹주소를 리퀘스트해서 html 페이지를 받음**

```
page = requests.get("https://en.wikipe
dia.org/wiki/List_of_state_and_union_territory_capita
ls_in_India")
```

- Print the status code of the page. **상태확인**

```
page.status_code
```

Beautiful Soup

- Parse the html in the 'page' variable, and store it in Beautiful Soup format. 페이지내용을 받아서 뷰티플숲 형식으로 바꿈

```
    soup = BeautifulSoup(page.content, 'html.parser')
```

- Use function “prettify” to look at nested structure of HTML page 보기좋은 형태로 바꿈

```
    print(soup.prettify())
```

Working with HTML Tags

- Return content between opening and closing tag including tag. 택사이의 내용만 리턴

```
soup.title
```

- Return string within given tag 택안에 글자만 리턴

```
soup.title.string
```

Extracting Links within <a>

- Find the first link within page’s <a> tags 처음 링크만 리턴

```
soup.a
```

- Find all the links within page’s <a> tags 모든 링크를 리턴

```
soup.find_all("a")
```

Extracting Values of Specific Attribute

- Iterate over each a tag and then return the link using attribute "href" with **get**. **모든 링크중에서 href속성만 리턴**

```
all_links = soup.find_all("a")
for link in all_links:
    print(link.get("href"))
```

Extracting Information within <table>

- Extract information within all **table** tags. 모든 테이블텍 리턴

```
all_tables = soup.find_all("table")
```

- Use attribute “class” of table and use it to filter the right table. 특정 클래스이름을 가진 테이블만 리턴

```
right_table = soup.find('table', class_='wikitable
sortable plainrowheaders')
right_table
```

Extract Information to List

```
A = []
B = []
C = []
D = []
E = []
F = []
G = []
```

```
for row in right_table.findAll("tr"):
    cells = row.findAll('td')
    states=row.findAll('th') #To store second column data
    if len(cells)==6: #Only extract table body not heading
        A.append(cells[0].find(text=True))
        B.append(states[0].find(text=True))
        C.append(cells[1].find(text=True))
        D.append(cells[2].find(text=True))
        E.append(cells[3].find(text=True))
        F.append(cells[4].find(text=True))
        G.append(cells[5].find(text=True))
```

List to DataFrame

```
import pandas as pd
df = pd.DataFrame(A, columns=['Number'])
df['State/UT']=B
df['Admin_Capital']=C
df['Legistlative_Capital']=D
df['Judiciary_Capital']=E
df['Year_Capital']=F
df['Former_Capital']=G
df
```

Saving Dataframe as File

```
df.to_excel('table.xlsx')
df.to_csv('table.csv')
```

Lab #11

- Complete the web scraping tutorial on https://www.analyticsvidhya.com/blog/2015/10/beginner-guide-web-scraping-beautiful-soup-python/ and save the data frame as an excel file
- Complete the weather scraping tutorial on https://www.dataquest.io/blog/web-scraping-tutorial-python/ and save the data frame as an excel file

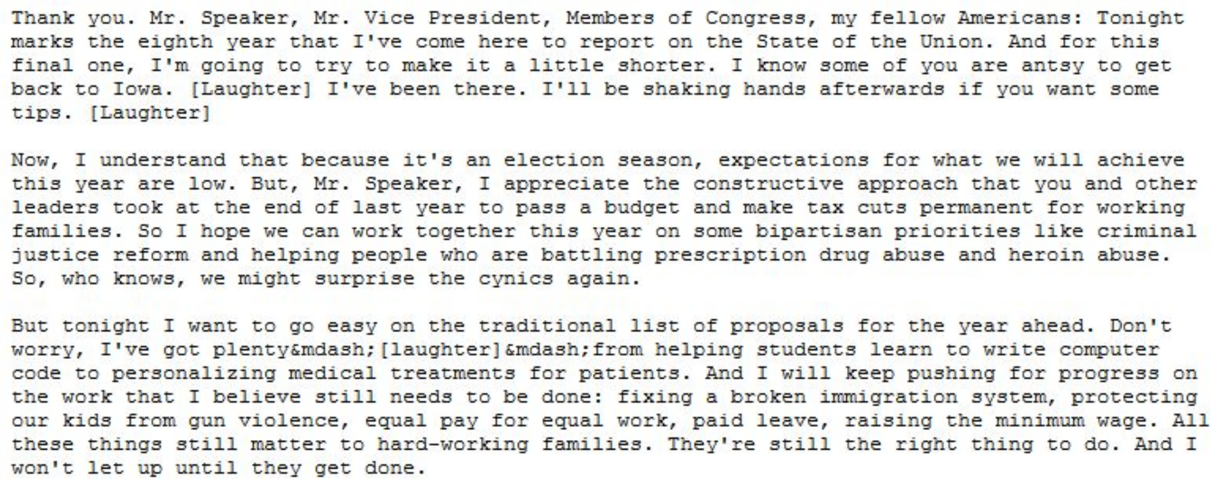
United States Presidential State of the Union Addresses 미국대통령 국정연설 분석

- Annual message presented by the President of the US to a joint session of the United States Congress. The message includes a budget message and economic report of the nation, and also allows the President to outline his legislative agenda and national priorities. During most of the country's first century, the President primarily only submitted a written report to Congress. With advent of radio and television, the address is now broadcast live across the country on most networks. 미국의회에서 대통령이 한 연초연설. 예산관련, 국가의 경제리포트를 포함. 입법아젠다, 국가의 우선순위사항을 설명. 처음에는 의회에 써서 제출을 했는데, 라디오와 티비가 나오면서 라이브방송을 함

Exercise #11
Barack Obama’s 2016 State of the Union Address Example

- Split the text into individual words and create a frequency table **문서를 단어로 쪼개서 빈도테이블을 만들기**
- Filter words with a frequency less than 0.1% using word_frequency.csv **빈도가 0.1 이하인 단어를 추출**
- Filter words with a frequency less than 0.002% **빈도가 0.002이하인 단어를 추출**
- Summarize the top five most used words that have a frequency less than 0.002% in the Google Web Corpus and extract the responding name and year from metadata.csv **빈도가 0.002이하인 단어중 가장 빈도수가 높은 5개의 단어요약**

Barack Obama's 2016 State of the Union Address
버락오바마 2016 국정연설



Thank you. Mr. Speaker, Mr. Vice President, Members of Congress, my fellow Americans: Tonight marks the eighth year that I've come here to report on the State of the Union. And for this final one, I'm going to try to make it a little shorter. I know some of you are antsy to get back to Iowa. [Laughter] I've been there. I'll be shaking hands afterwards if you want some tips. [Laughter]

Now, I understand that because it's an election season, expectations for what we will achieve this year are low. But, Mr. Speaker, I appreciate the constructive approach that you and other leaders took at the end of last year to pass a budget and make tax cuts permanent for working families. So I hope we can work together this year on some bipartisan priorities like criminal justice reform and helping people who are battling prescription drug abuse and heroin abuse. So, who knows, we might surprise the cynics again.

But tonight I want to go easy on the traditional list of proposals for the year ahead. Don't worry, I've got plenty&mdash;[laughter]&mdash;from helping students learn to write computer code to personalizing medical treatments for patients. And I will keep pushing for progress on the work that I believe still needs to be done: fixing a broken immigration system, protecting our kids from gun violence, equal pay for equal work, paid leave, raising the minimum wage. All these things still matter to hard-working families. They're still the right thing to do. And I won't let up until they get done.

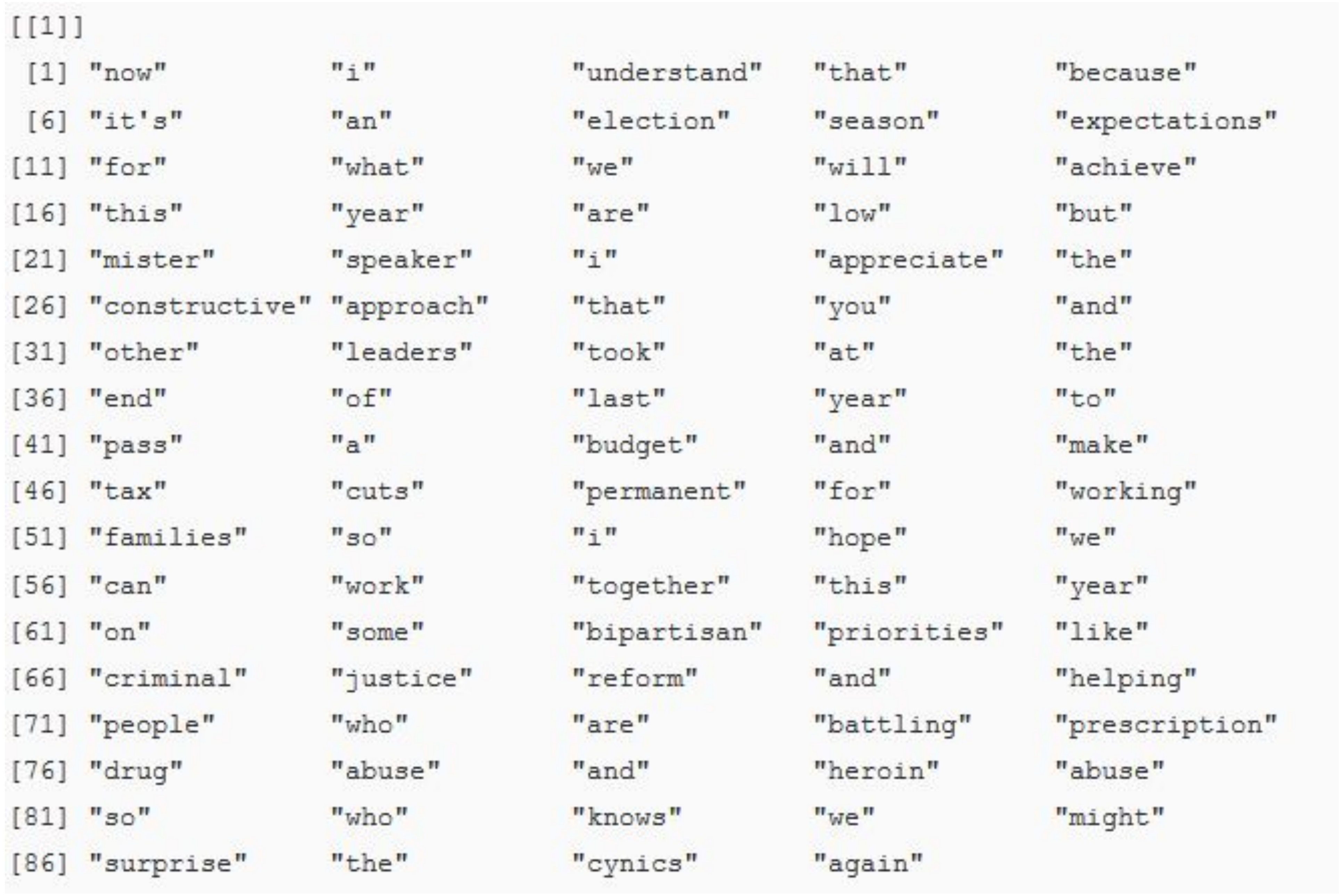
http://programminghistorian.github.io/ph-submissions/assets/basic-text-processing-in-r/sotu_text/236.txt

Word Tokenization 단어토큰화

```
text = "Now, I understand that because it's an election season\
expectations for what we will achieve this year are low\
But, Mister Speaker, I appreciate the constructive approach\
that you and other leaders took at the end of last year\
to pass a budget and make tax cuts permanent for working\
families. So I hope we can work together this year on some\
bipartisan priorities like criminal justice reform and\
helping people who are battling prescription drug abuse\
and heroin abuse. So, who knows, we might surprise the\
cynics again"
```

```
from nltk.tokenize import word_tokenize
tokenized_word=word_tokenize(text)
print(tokenized_word)
```

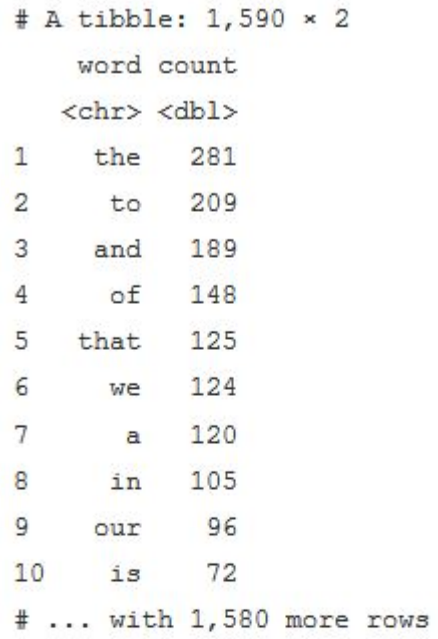
words



```
[[1]]
 [1] "now"          "i"            "understand"   "that"         "because"
 [6] "it's"         "an"           "election"     "season"       "expectations"
[11] "for"          "what"         "we"           "will"         "achieve"
[16] "this"         "year"         "are"          "low"          "but"
[21] "mister"       "speaker"      "i"            "appreciate"   "the"
[26] "constructive" "approach"     "that"         "you"          "and"
[31] "other"        "leaders"      "took"         "at"           "the"
[36] "end"          "of"           "last"         "year"         "to"
[41] "pass"         "a"            "budget"       "and"          "make"
[46] "tax"          "cuts"         "permanent"    "for"          "working"
[51] "families"     "so"           "i"            "hope"         "we"
[56] "can"          "work"         "together"     "this"         "year"
[61] "on"           "some"         "bipartisan"   "priorities"   "like"
[66] "criminal"     "justice"      "reform"       "and"          "helping"
[71] "people"       "who"          "are"          "battling"     "prescription"
[76] "drug"         "abuse"        "and"          "heroin"       "abuse"
[81] "so"           "who"          "knows"        "we"           "might"
[86] "surprise"     "the"          "cynics"       "again"
```

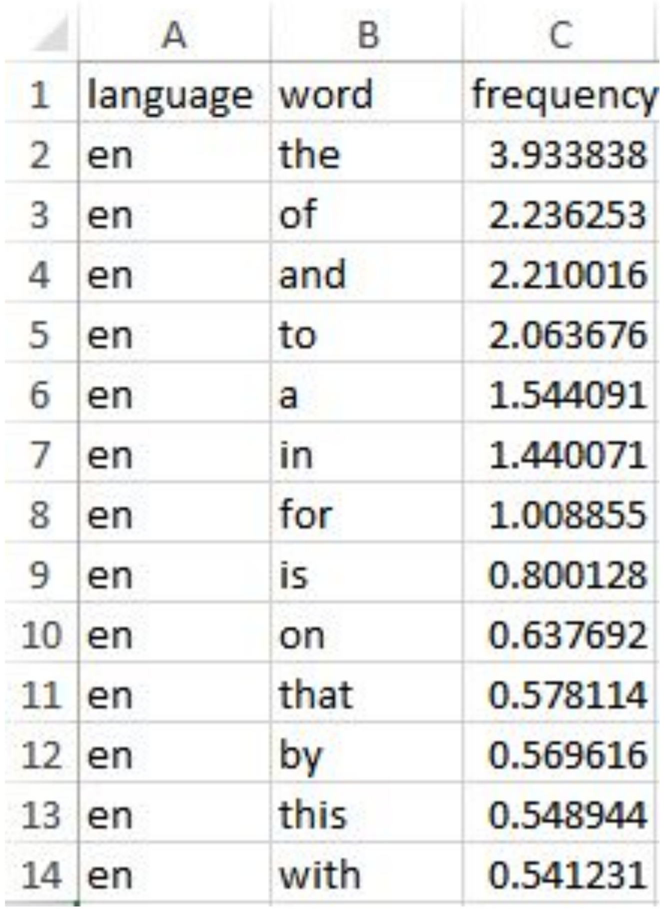
Frequency Distribution 빈도분포

```
from nltk.probability
import FreqDist

fdist =
FreqDist(tokenized_word)

fdist

fdist.most_common(10)
```



```
# A tibble: 1,590 × 2
    word count
   <chr> <dbl>
1    the   281
2     to   209
3    and   189
4     of   148
5   that   125
6     we   124
7      a   120
8     in   105
9    our    96
10    is    72
# ... with 1,580 more rows
```

Google Web Trillion Word Corpus
구글웹단어빈도표



	A	B	C
1	language	word	frequency
2	en	the	3.933838
3	en	of	2.236253
4	en	and	2.210016
5	en	to	2.063676
6	en	a	1.544091
7	en	in	1.440071
8	en	for	1.008855
9	en	is	0.800128
10	en	on	0.637692
11	en	that	0.578114
12	en	by	0.569616
13	en	this	0.548944
14	en	with	0.541231

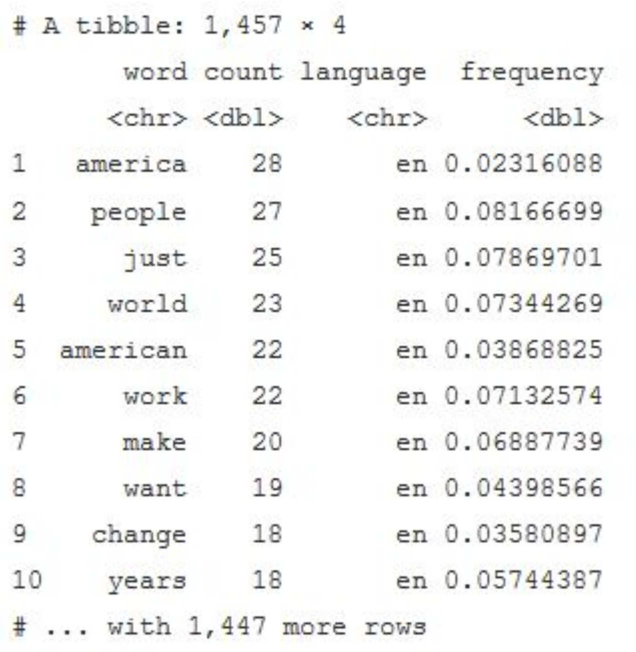
http://programminghistorian.github.io/ph-submissions/assets/basic-text-processing-in-r/word_frequency.csv

- Merge two data sets on word
  단어를 기초로 두 데이터셋을 결합

```
tab = pd.merge(obama_freq,
freq, on='word')
```

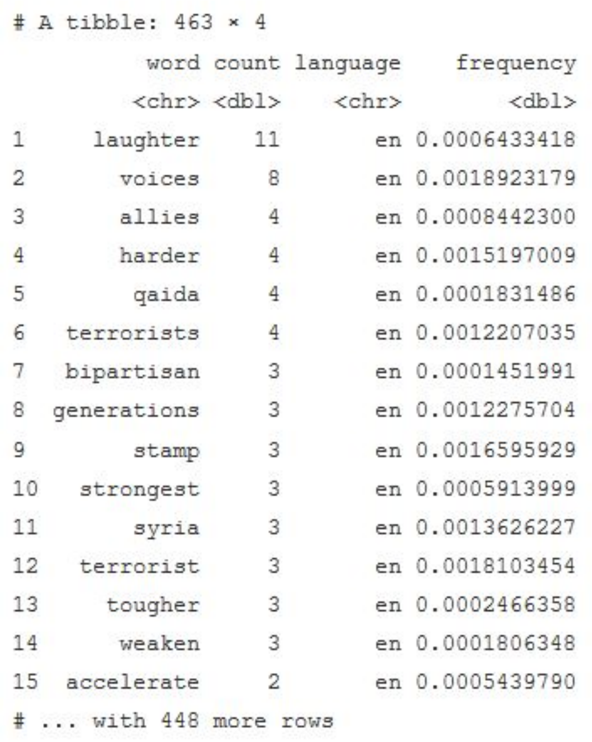
Exploratory Analysis 탐색분석

```
tab[tab.frequency < 0.1]
```



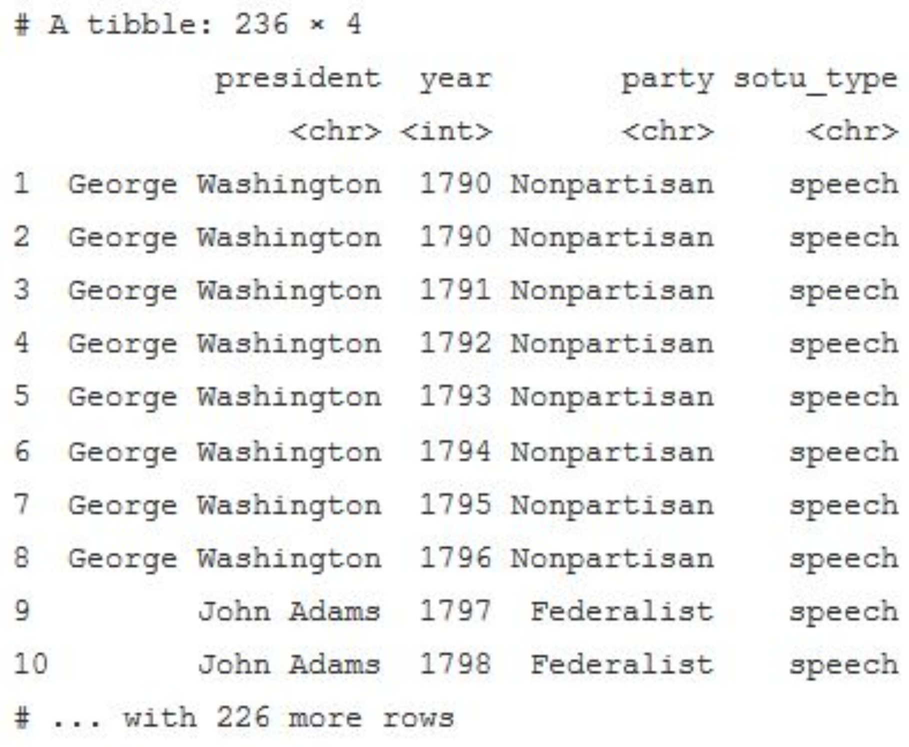
```
# A tibble: 1,457 × 4
       word count language  frequency
      <chr> <dbl>    <chr>      <dbl>
1   america    28       en 0.02316088
2    people    27       en 0.08166699
3      just    25       en 0.07869701
4     world    23       en 0.07344269
5  american    22       en 0.03868825
6      work    22       en 0.07132574
7      make    20       en 0.06887739
8      want    19       en 0.04398566
9    change    18       en 0.03580897
10    years    18       en 0.05744387
# ... with 1,447 more rows
```

```
tab[tab.frequency <
0.002].head(15)
```



```
# A tibble: 463 × 4
           word count language    frequency
          <chr> <dbl>    <chr>        <dbl>
1      laughter    11       en 0.0006433418
2        voices     8       en 0.0018923179
3        allies     4       en 0.0008442300
4        harder     4       en 0.0015197009
5         qaida     4       en 0.0001831486
6    terrorists     4       en 0.0012207035
7    bipartisan     3       en 0.0001451991
8   generations     3       en 0.0012275704
9         stamp     3       en 0.0016595929
10    strongest     3       en 0.0005913999
11        syria     3       en 0.0013626227
12    terrorist     3       en 0.0018103454
13      tougher     3       en 0.0002466358
14       weaken     3       en 0.0001806348
15   accelerate     2       en 0.0005439790
# ... with 448 more rows
```

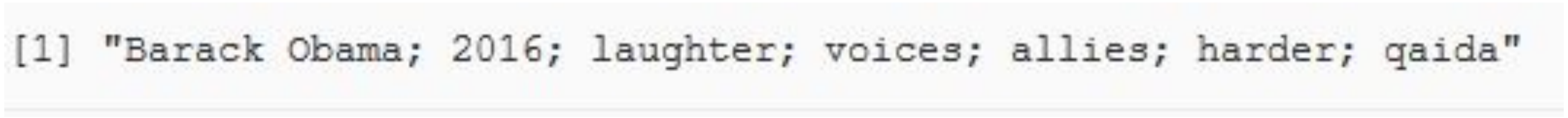
Metadata 메타데이터



```
# A tibble: 236 × 4
           president  year       party sotu_type
               <chr> <int>       <chr>     <chr>
1  George Washington  1790 Nonpartisan    speech
2  George Washington  1790 Nonpartisan    speech
3  George Washington  1791 Nonpartisan    speech
4  George Washington  1792 Nonpartisan    speech
5  George Washington  1793 Nonpartisan    speech
6  George Washington  1794 Nonpartisan    speech
7  George Washington  1795 Nonpartisan    speech
8  George Washington  1796 Nonpartisan    speech
9         John Adams  1797  Federalist    speech
10        John Adams  1798  Federalist    speech
# ... with 226 more rows
```

http://programminghistorian.github.io/ph-submissions/assets/basic-text-processing-in-r/metadata.csv

Result of Barack Obama’s 2016 State of the Union Address 국정연설 결과요약

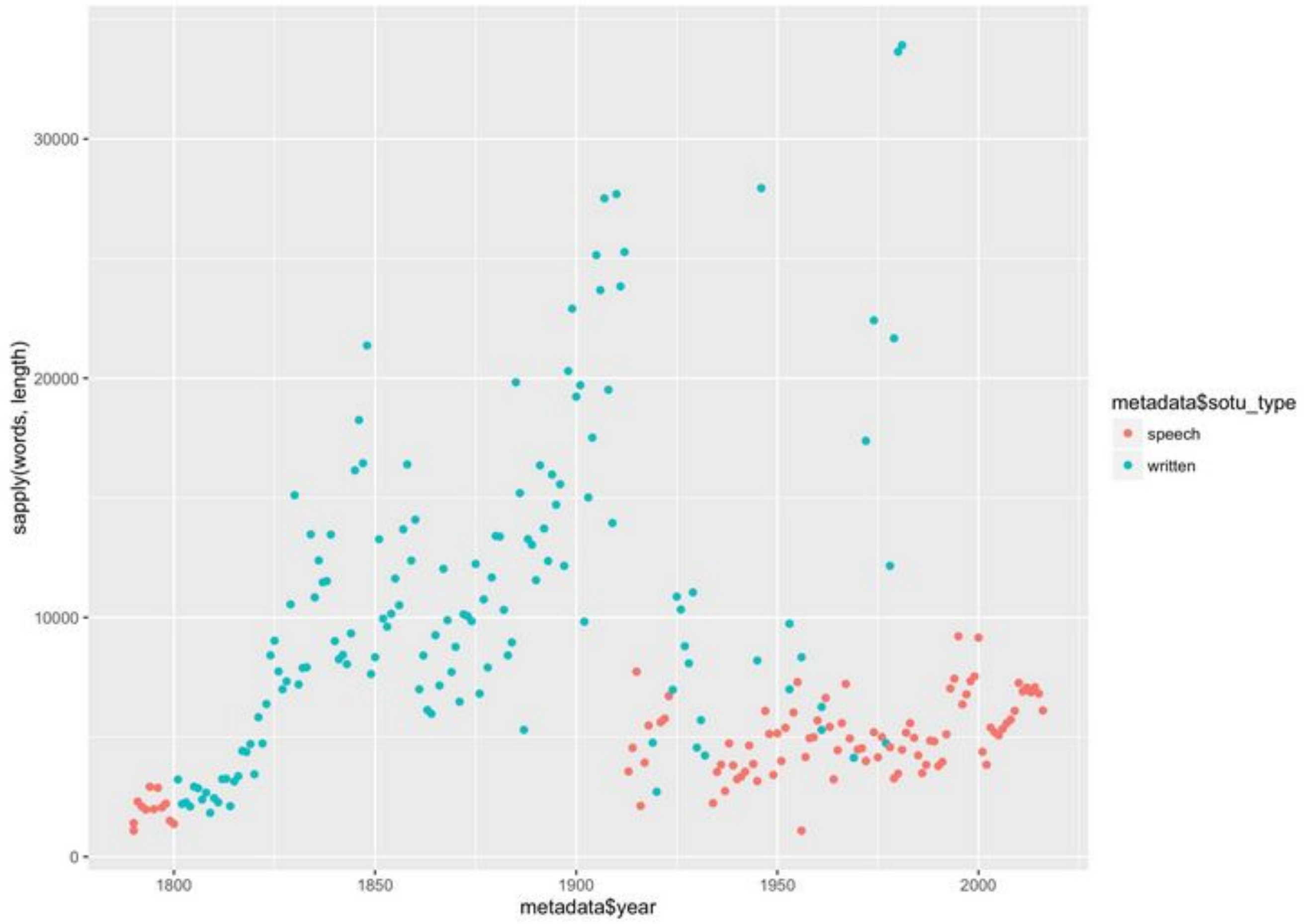


```
[1] "Barack Obama; 2016; laughter; voices; allies; harder; qaida"
```

Exercise #11

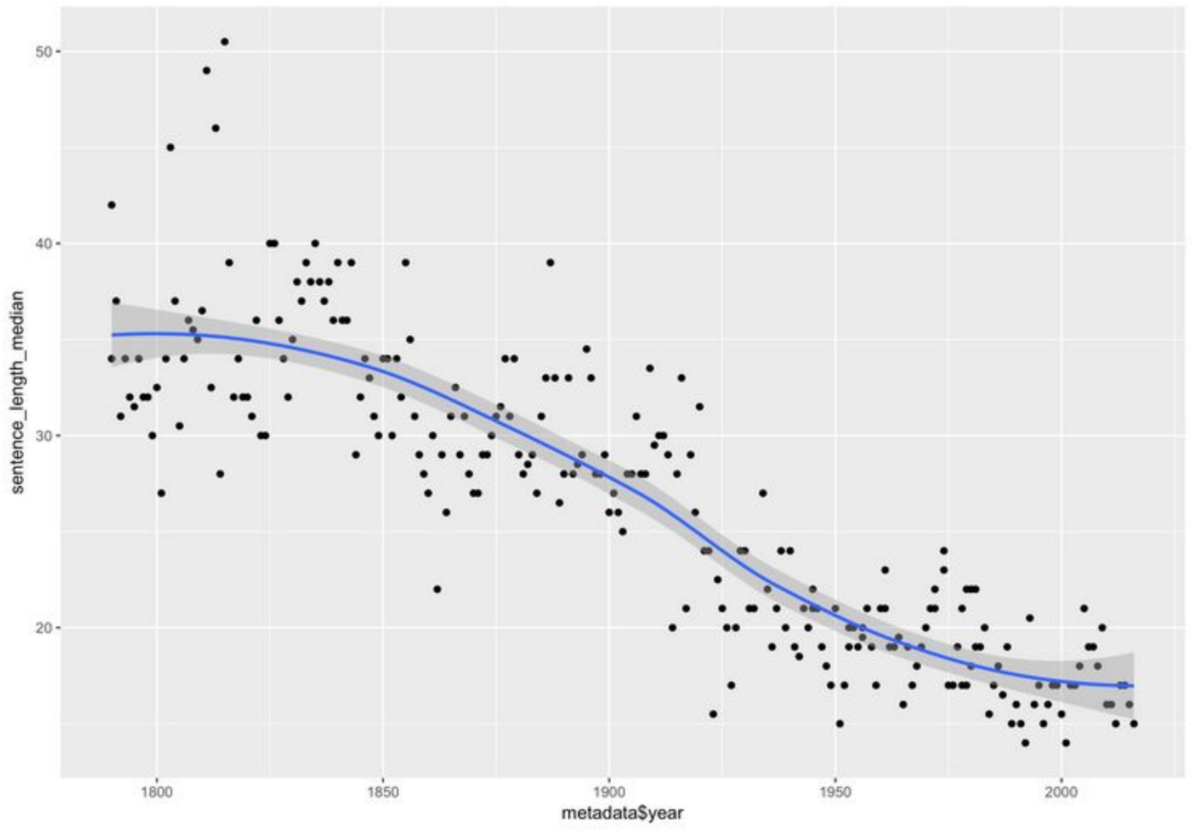
- State of the Union Address from 1790 to 2016
  - Split the text into individual words and count the number of words in each document 문서를 개별 단어로 분리, 문서당 단어의 수를 카운트
  - Plot the number of words in each State of the Union Address by year, with color denoting whether it was a written or oral message 연도별 연설에 쓰인 단어수를 그래프로 그리기, 쓰여진 연설인지 구두연설인지 다른 컬러로!
  - Plot the median sentence length for each State of the Union Address, with a smoothing line 각 연설단 문장의 길이의 중위값 그래프
  - Summarize the top five most used words in each document that have a frequency less than 0.002% in the Google Web Corpus with metadata.csv 각 문서당 빈도수 0.002이하인 단어중 상위5개의 단어요약

Exploratory Analysis 탐색분석



Number of words in each State of the Union Address plotted by year, with color denoting whether it was a written or oral message.

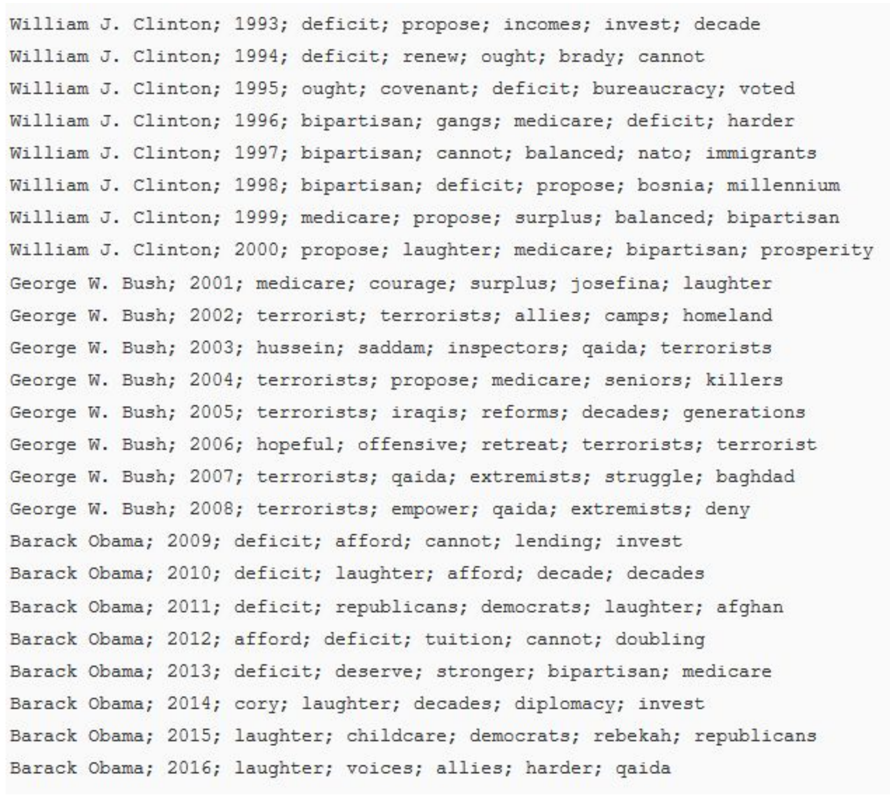
Stylometric Analysis 문체측정분석



Median sentence length for each State of the Union Address, with a smoothing line.

```
plt.scatter(metadata.year, sentence_length_median)
```

Document Summarization 문서요약



```
William J. Clinton; 1993; deficit; propose; incomes; invest; decade
William J. Clinton; 1994; deficit; renew; ought; brady; cannot
William J. Clinton; 1995; ought; covenant; deficit; bureaucracy; voted
William J. Clinton; 1996; bipartisan; gangs; medicare; deficit; harder
William J. Clinton; 1997; bipartisan; cannot; balanced; nato; immigrants
William J. Clinton; 1998; bipartisan; deficit; propose; bosnia; millennium
William J. Clinton; 1999; medicare; propose; surplus; balanced; bipartisan
William J. Clinton; 2000; propose; laughter; medicare; bipartisan; prosperity
George W. Bush; 2001; medicare; courage; surplus; josefina; laughter
George W. Bush; 2002; terrorist; terrorists; allies; camps; homeland
George W. Bush; 2003; hussein; saddam; inspectors; qaida; terrorists
George W. Bush; 2004; terrorists; propose; medicare; seniors; killers
George W. Bush; 2005; terrorists; iraqis; reforms; decades; generations
George W. Bush; 2006; hopeful; offensive; retreat; terrorists; terrorist
George W. Bush; 2007; terrorists; qaida; extremists; struggle; baghdad
George W. Bush; 2008; terrorists; empower; qaida; extremists; deny
Barack Obama; 2009; deficit; afford; cannot; lending; invest
Barack Obama; 2010; deficit; laughter; afford; decade; decades
Barack Obama; 2011; deficit; republicans; democrats; laughter; afghan
Barack Obama; 2012; afford; deficit; tuition; cannot; doubling
Barack Obama; 2013; deficit; deserve; stronger; bipartisan; medicare
Barack Obama; 2014; cory; laughter; decades; diplomacy; invest
Barack Obama; 2015; laughter; childcare; democrats; rebekah; republicans
Barack Obama; 2016; laughter; voices; allies; harder; qaida
```

- Analysis of every State of the Union Address from 1790 to 2016 a great high-level summary of each presidency 년도별 대통령 연설 분석으로 각 대통령에 대한 높은 수준에 대한 요약이 가능함

Sentiment Analysis 정서분석

- The computational task of automatically determining what feelings a writer is expressing in text 텍스트안에 나타나는 글쓴 사람의 감정을 계산을 통해 분석
- Sentiment
  - Often framed as a binary distinction (positive vs. negative) 주로 두가지 정서 (긍정 또는 부정)
  - Can also be a more fine-grained, like identifying the specific emotion an author is expressing (like fear, joy or anger) 저자가 표현하는 특정한 감정을 더 표현할수 있음 (두려움, 기쁨, 분노)

Applications for Sentiment Analysis 정서분석의 영역

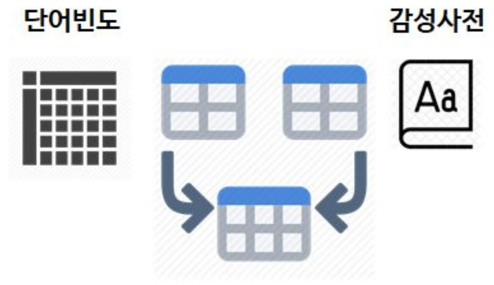
- Analyzing the social media discussion around a certain topic 특정 토픽에 대한 소셜미디어 토론을 분석
- Evaluating survey responses 설문조사 반응을 평가
- Determining whether product reviews are positive or negative 상품리뷰가 긍정인지 부정인지 결정

Type of Sentiment Analysis

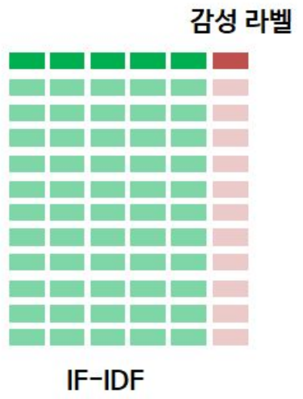
규칙/어휘 기반 감성분석

M/L 기반 감성분석

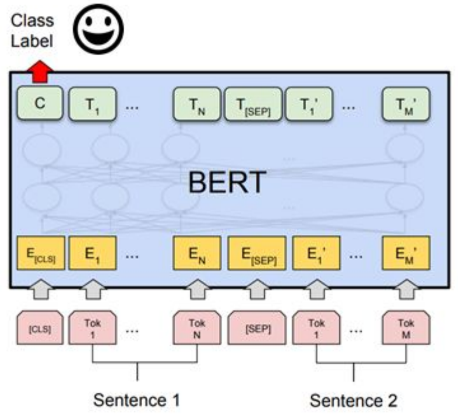
딥러닝 기반 감성분석



단어	감성점수	빈도수	합계
Happy	1	3	+3
Sad	-1	2	-2
Great	1	2	2
Ugly	-1	1	-1
합계			+2 ☺



$$y = f(x)$$



Steps for Sentiment Analysis 정서분석단계

Create or find a list of words associated with strongly positive or negative sentiment. 긍정, 부정과 관련된 단어리스트 생성

Count the number of positive and negative words in the text. 긍정, 부정 단어의 갯수 카운트

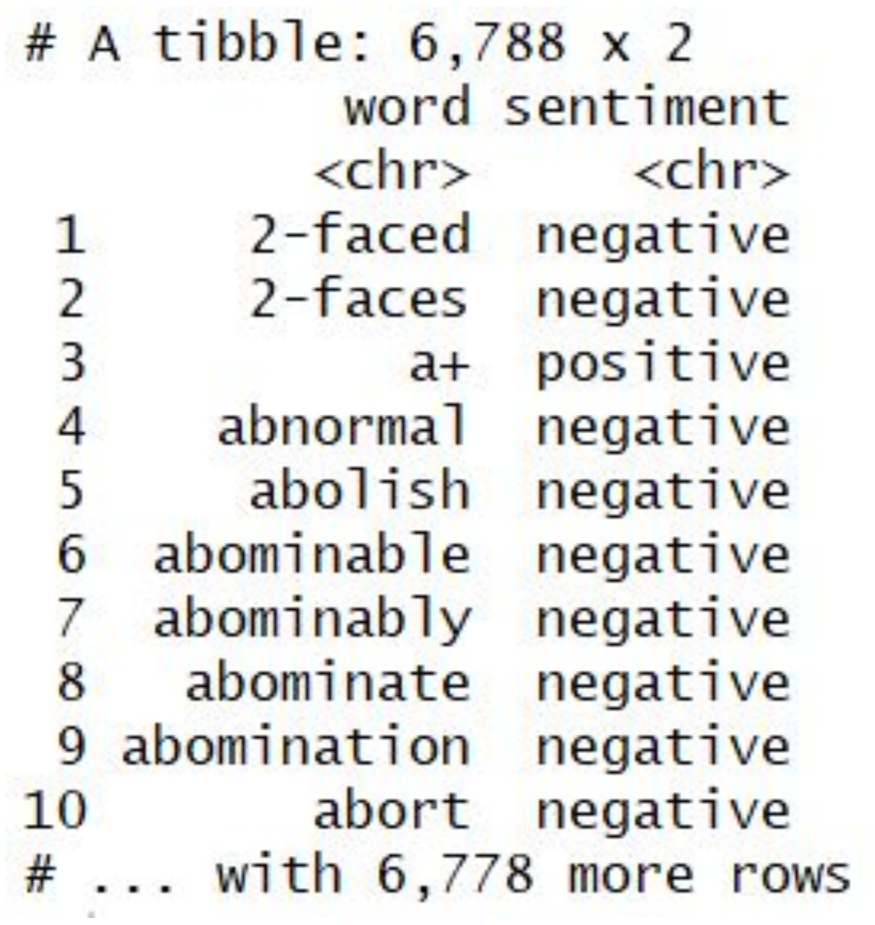
Analyze the mix of positive to negative words. 긍정, 부정단어를 합쳐서 분석

Many positive words and few negative words indicates positive sentiment, while many negative words and few positive words indicates negative sentiment. 긍정단어가 많으면 긍정정서, 부정잔어가 더 많으면 부정정서

Sentiment Lexicon 정서어휘

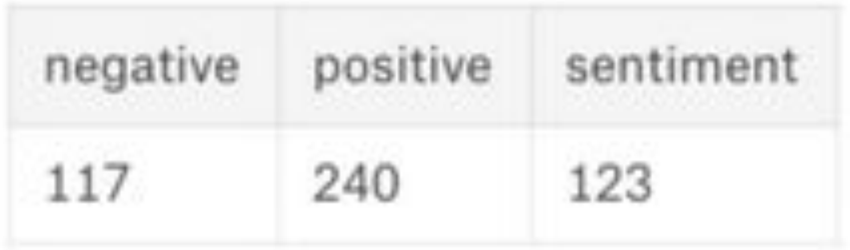
- A list of words associated with a specific sentiment 특정정서에 따른 단어리스트
- Can get it from bing 빙에서 정서어휘를 가져옴

```
library(tidytext)

get_sentiments("bing
")
```



```
# A tibble: 6,788 x 2
           word sentiment
          <chr>     <chr>
 1      2-faced  negative
 2      2-faces  negative
 3           a+  positive
 4     abnormal  negative
 5      abolish  negative
 6   abominable  negative
 7   abominably  negative
 8    abominate  negative
 9  abomination  negative
10        abort  negative
# ... with 6,778 more rows
```

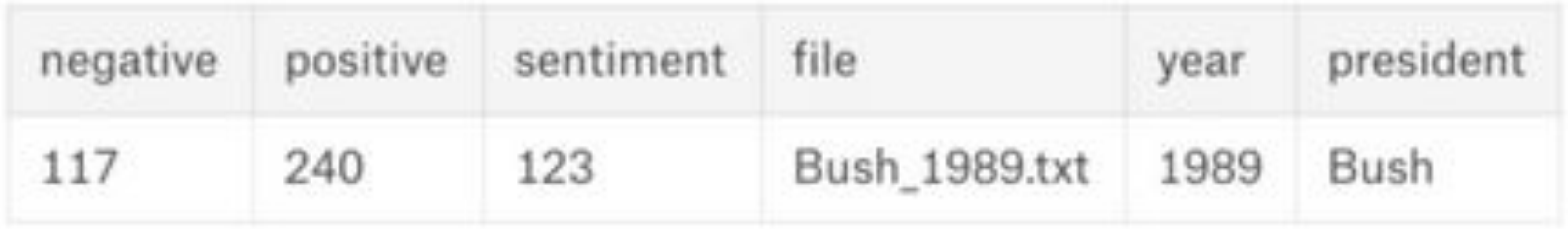
Sentiment Analysis 정서분석



negative	positive	sentiment
117	240	123

Sentiment = # of positive words - # of negative words

정서 = 긍정단어수 - 부정단어수



negative	positive	sentiment	file	year	president
117	240	123	Bush_1989.txt	1989	Bush

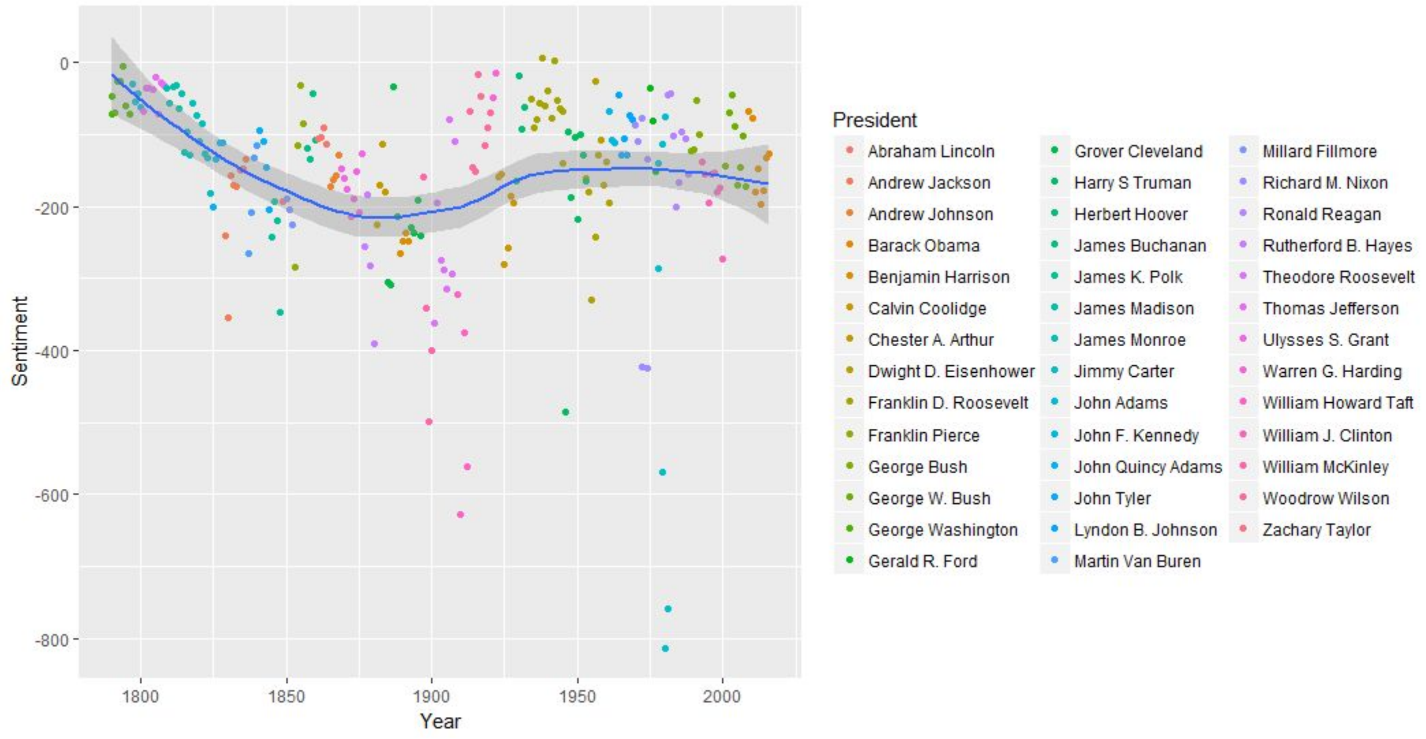
TextBlob

```
from textblob import TextBlob

wiki = TextBlob("Python is a high-level, general-purpose
programming language.")
testimonial = TextBlob("Textblob is amazingly simple to use.
What great fun!")

# 감성 출력
print('Sentiment of wiki: ', wiki.sentiment)
print('Sentiment of wiki: ', testimonial.sentiment)
```

Exercise #11

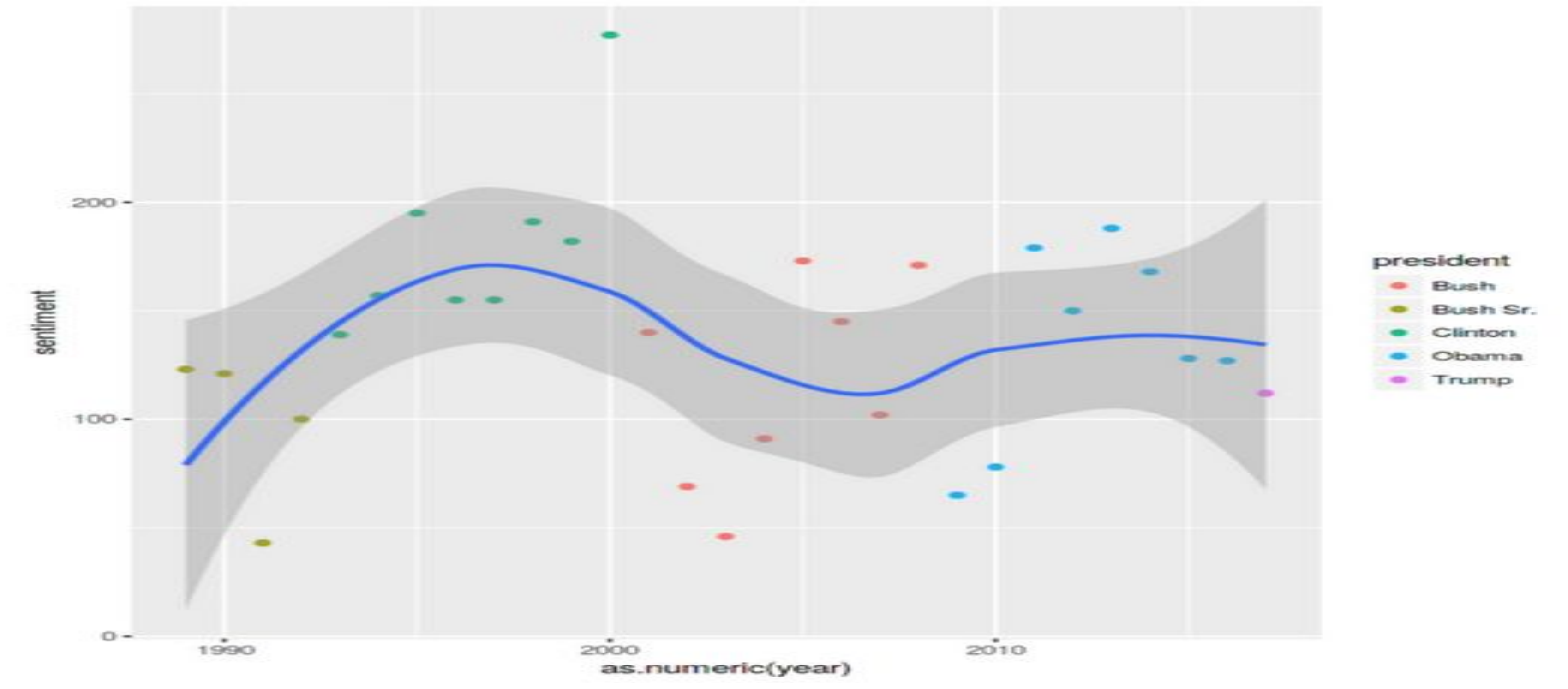
- Count the number of negative words, positive words, sentiments, year, and president name 부정어, 긍정어, 정서, 년도, 대통령이름을 카운트
- Analyze the change of the sentiment over the time 시간에 따른 정서의 변화분석
- Analyze the sentiment by presidents 대통령에 대한 정서 분석
- Analyze the sentiment by parties 당에 따른 정서분석

Sentiment over Time



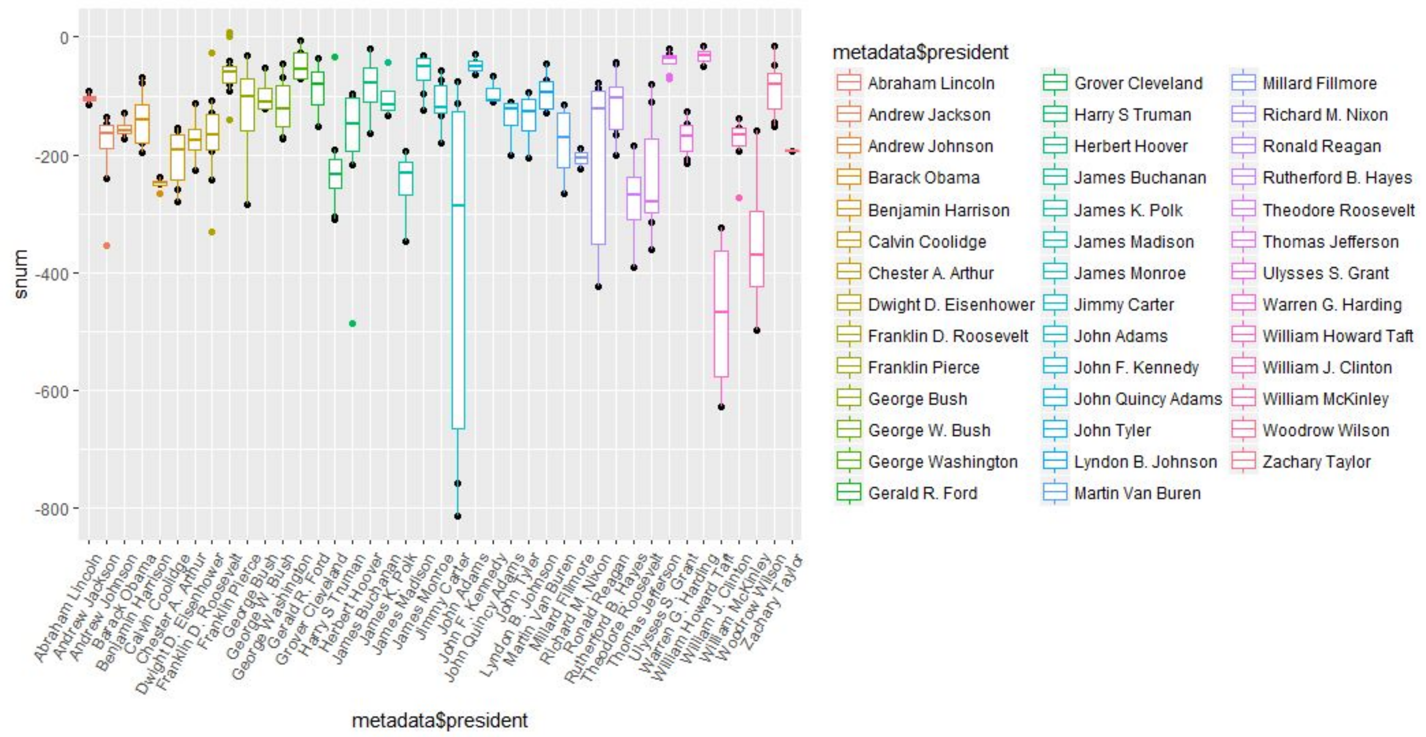
Color-coded by president 대통령에 따라 다른 색상

Sentiment over Time (Selected Presidents)



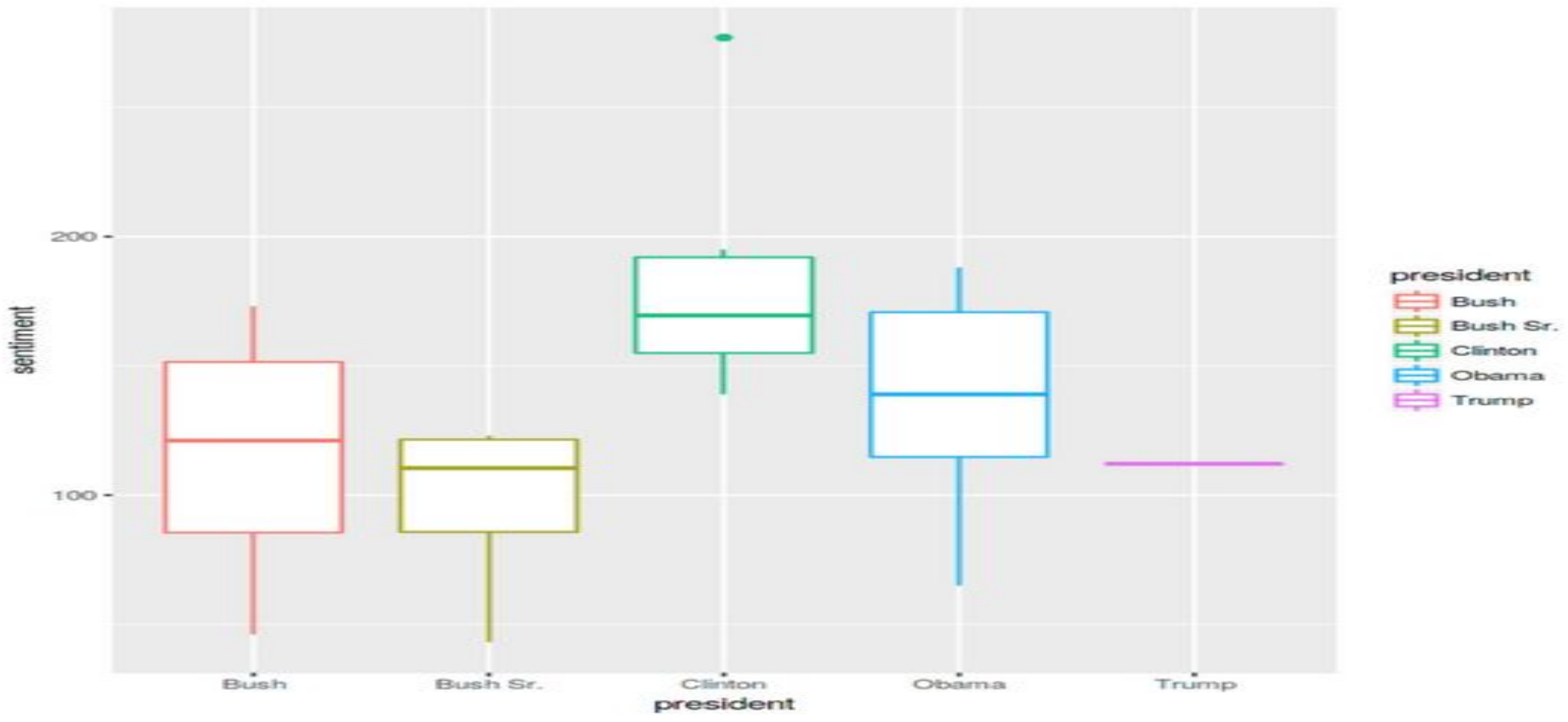
Color-coded by president **대통령에 따라 다른 색상**

Sentiment by President



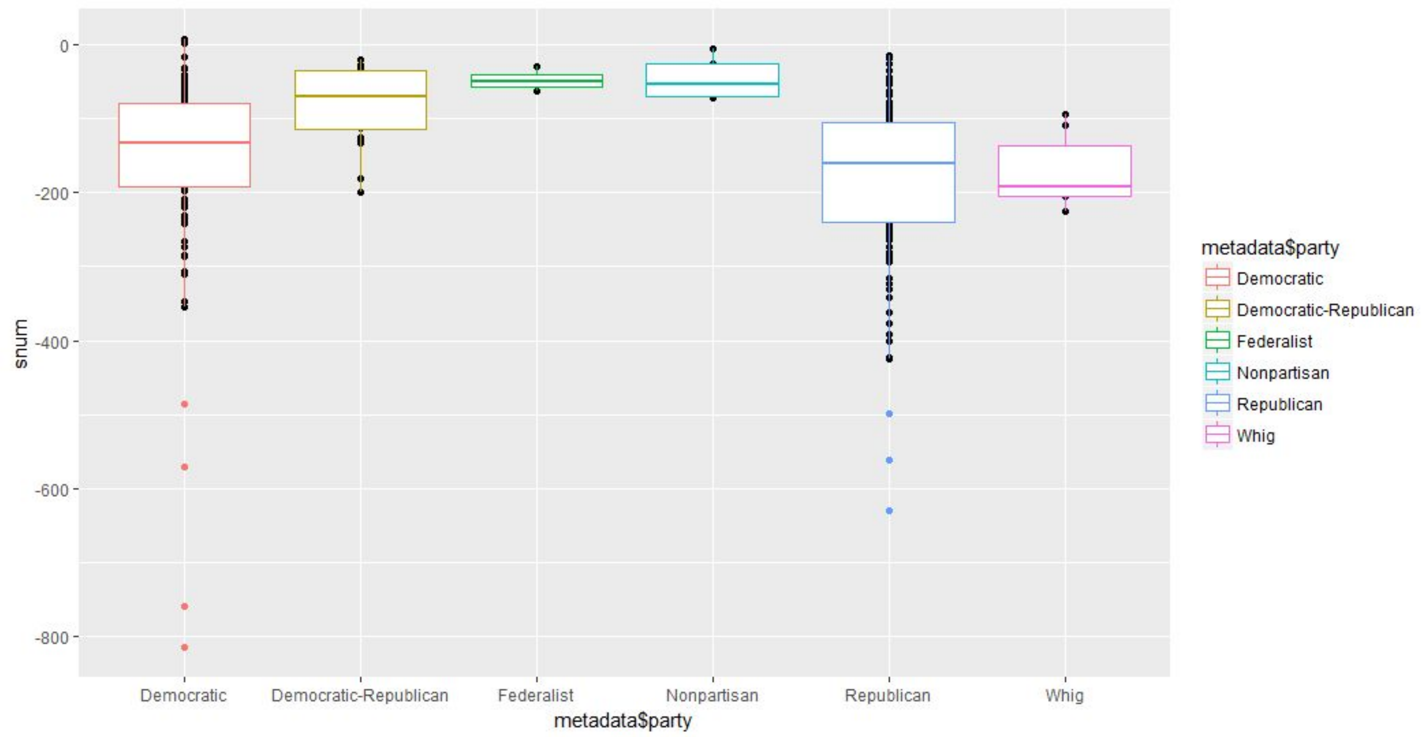
Color-coded by president 대통령에 따라 다른 색상

Sentiment by Five Presidents



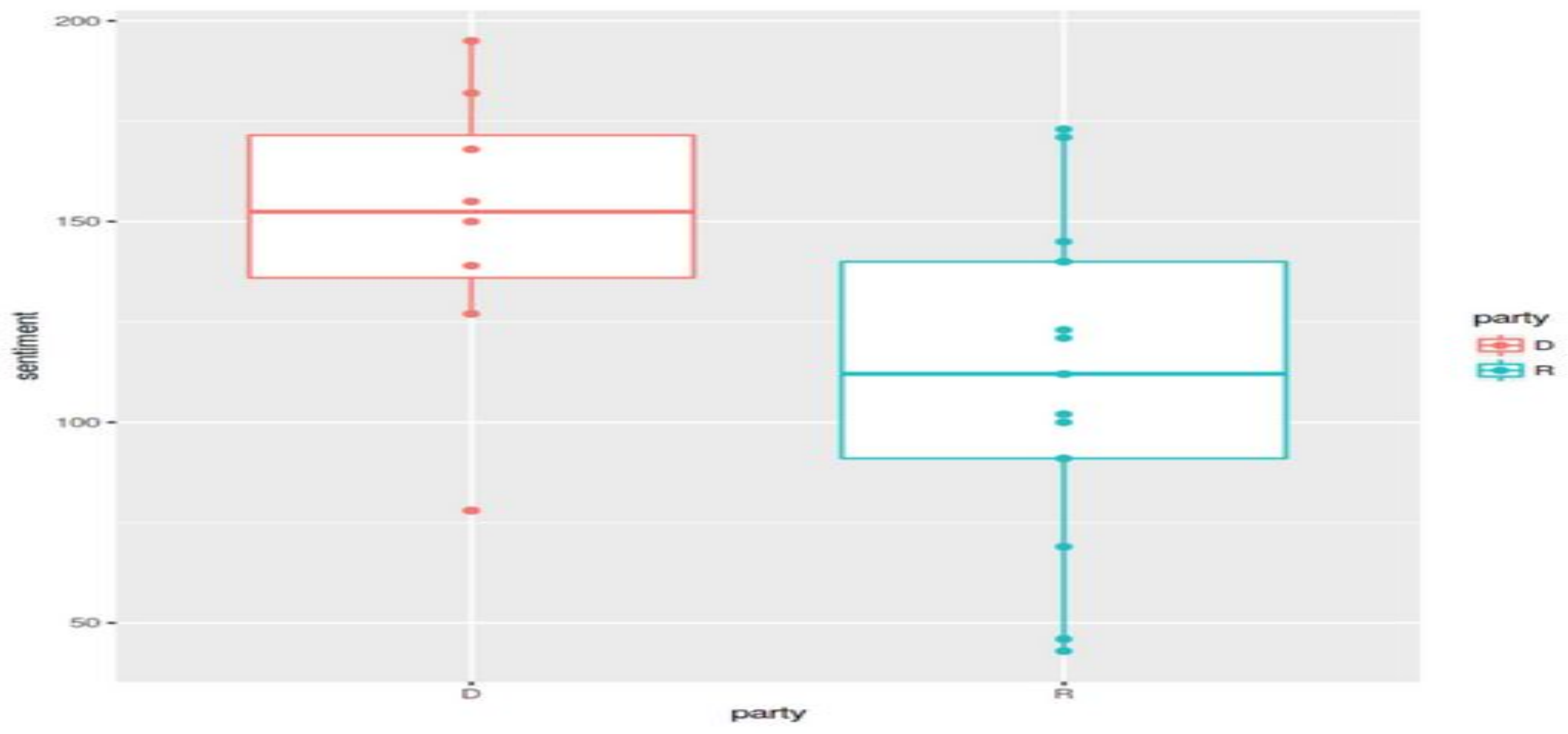
Color-coded by president **대통령에 따라 다른 색상**

Sentiment by All Parties



Color-coded by president 대통령에 따라 다른 색상

Sentiment by Two Parties



Color-coded by president 대통령에 따라 다른 색상

Exercise #11
Data Files

- http://d396qusza40orc.cloudfront.net/dsscapstone/dataset/Coursera-SwiftKey.zip
- final/en_US/
  - en_US.blogs.txt: text obtained from blogs (900,000 lines of text)
  - en_US.news.txt: text obtained from news feeds
  - en_US.twitter.txt: text obtained from Twitter (2 million)

Exercise #11

- Read train.tsv **훈련데이터 읽기**
- Count phrase by sentiments and draw bar chart **감정에 따른 표현을 카운트, 막대그래프**
- Remove stopwords and punctuations **스탑워드와 문장기호 삭제**
- Use CountVectorizer and TF-IDF to generate bag of words **단어를 벡터화**
- Train and test split (.3 test set) **30프로의 테스트셋**
- Build a naïve Bayesian model (MultinomialNB) **나이브제이지안 모델 생성**
- Compare the results **결과 비교**