

# 한국어 말뭉치 분석

**Prepared by Prof. Youn-Sik Hong**

# 목차

**01 konlpy 설치**

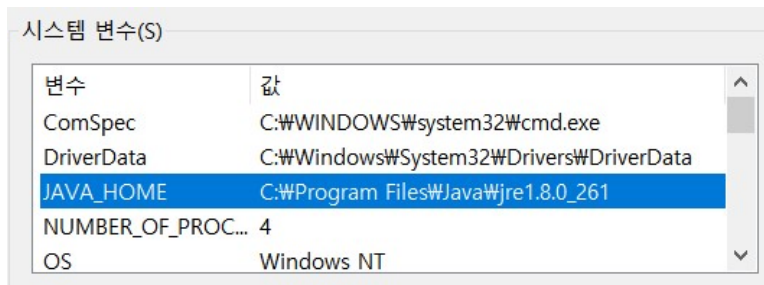
02 한국어 형태소 분석기

03 단어 출현 빈도 분석

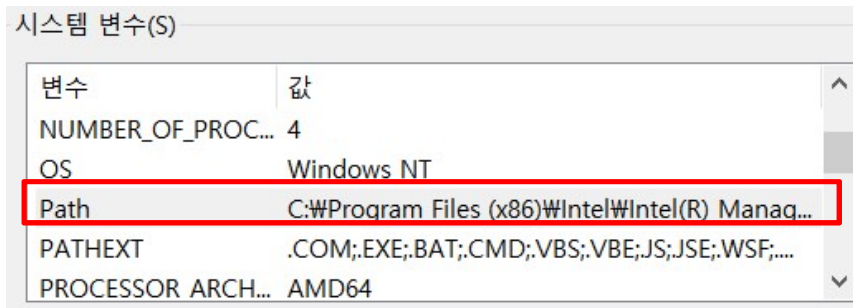
# konlpy 설치하기 전에 java가 설치되어 있어야 함

## ■ Java : 1.7 버전 이상이 설치되어 있어야 함

- 제어판 > 시스템 및 보안 > 시스템 > 고급 시스템 설정 (윈도우 10 기준)
- 고급 > 환경 변수 > 시스템 변수



**JAVA\_HOME 설정**



C:\Program Files\Java\jre1.8.0\_261\bin

**Path에 bin 경로 설정**

```
C:\Users\yshon>java -version
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)

C:\Users\yshon>echo %JAVA_HOME%
C:\Program Files\Java\jre1.8.0_261
```

**DOS 명령 창에서 확인**

**java -version**

**echo %JAVA\_HOME%**

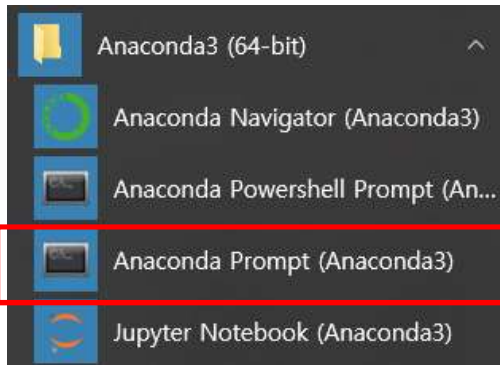
# konlpy 설치 (1/4)

## ■ KoNLPy (Korean NLP in Python)

- <http://konlpy.org/ko/latest/>
- 설치하기 > 윈도우 > JType1 (>=0.5.7) 링크 클릭
  - JType1-1.0.1-cp37-cp37-win\_amd64.whl (wheel 파일) 다운로드

64비트 버전  
파이썬 3.7

- Anaconda Prompt 실행



jupyter 노트북에서 코드 작성하고 실행하기 때문에  
아나콘다 창에서 설치 명령을 실행시켜야 함.

Python IDE에서 실행할 경우 DOS 창(cmd)에서 실행

Python 버전 확인 : `python --version`

```
Anaconda Prompt (Anaconda3)

(base) C:\Users\Wyshon>python --version
Python 3.7.6
```

아나콘다 설치할 때  
파이썬 버전 3.7이 설치됨

## konlpy 설치 (2/4)

### ■ KoNLPy (Korean NLP in Python)

- pip 는 최신 버전으로 업그레이드하는 것이 필요!

**pip 업그레이드 : python.exe -m pip install --upgrade pip**

```
C:\Users\tyshon\Downloads>python.exe -m pip install --upgrade pip
Collecting pip
  Using cached pip-20.2-py2.py3-none-any.whl (1.5 MB)
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.1.1
    Uninstalling pip-20.1.1:
      Successfully uninstalled pip-20.1.1
Successfully installed pip-20.2
```



# konlpy 설치 (3/4)

## ■ KoNLPy (Korean NLP in Python)

다운로드 받은 wheel 파일 설치

```
(base) C:\Users\Wysyon\Downloads>pip install JPype1-1.0.1-cp37-cp37m-win_amd64.whl
```

konlpy 설치 : pip install konlpy

설치 명령은 아나콘다 창에서 실행시켜야 함.

```
C:\Users\Wysyon\Downloads>pip install konlpy
Collecting konlpy
  Using cached konlpy-0.5.2-py2.py3-none-any.whl (19.4 MB)
```



```
Collecting oauthlib>=3.0.0
  Using cached oauthlib-3.1.0-py2.py3-none-any.whl (147 kB)
Installing collected packages: colorama, chardet, urllib3, certifi, idna, PySocks, requests, oauthlib, requests-oauthlib, six, tweepy, BeautifulSoup4, lxml, numpy, konlpy
Successfully installed PySocks-1.7.1 BeautifulSoup4-4.6.0 certifi-2020.6.20 chardet-3.0.4 colorama-0.4.3 idna-2.10 konlpy-0.5.2 lxml-4.5.2 numpy-1.19.1 oauthlib-3.1.0 requests-2.24.0 requests-oauthlib-1.3.0 six-1.15.0 tweepy-3.9.0 urllib3-1.25.10
```


# konlpy 설치 (4/4) – 설치 경로 확인

## ■ KoNLPy (Korean NLP in Python)

- Konlpy는 어느 경로에 설치되었을까?

### 아나콘다 창에서 인스톨 명령 실행한 경우

내 PC > 로컬 디스크 (C:) > 사용자 > yshon > Anaconda3 > Lib > site-packages

 konlpy	2020-07-29 오후 7:16	파일 폴더
 konlpy-0.5.2.dist-info	2020-07-29 오후 7:45	파일 폴더

### DOS cmd 창에서 인스톨 명령 실행한 경우

<< yshon > AppData > Local > Programs > Python > Python37 > Lib > site-packages

 konlpy	2020-07-29 오후 4:45	파일 폴더
 konlpy-0.5.2.dist-info	2020-07-29 오후 4:45	파일 폴더

# konlpy 동작 확인 : konlpy-intro. ipynb

## ■ 아래 코드를 입력하고 실행시켜 보자!

```
>>> from konlpy.tag import Kkma
>>> from konlpy.utils import pprint
>>> kkma = Kkma()
>>> pprint(kkma.sentences(u'네, 안녕하세요, 반갑습니다.'))
['네, 안녕 하세요, 반갑습니다.']
>>> pprint(kkma.nouns(u'질문이나 건의사항은 깃헙 이슈 트래커에 남겨주세요.'))
['질문', '건의', '건의사항', '사항', '깃헙', '이슈', '트래커']
>>> pprint(kkma.pos(u'오류보고는 실행환경, 에러메세지와함께 설명을 최대한상세히!'))
[('오류', 'NNG'), ('보고', 'NNG'), ('는', 'JX'), ('실행', 'NNG'), ('환경', 'NNG'), ('에러', 'SP'), ('메세지', 'NNG'), ('와', 'JKM'), ('함께', 'MAG'), ('설명', 'NNG'), ('을', 'NNG'), ('최대', 'JKO'), ('한상', 'NNG'), ('세히', 'MAG'), ('!', 'SF'), ('', 'EMO')]
```

꼬꼬마

u : unicode

pos : 품사

Python으로 만든 한국어 형태소 분석기  
참고 사이트

<http://konlpy-ko.readthedocs.io/ko/v0.4.3/references>

KoNLPy (2014) : 박 은정(서울대)  
Umorpheme (2014): 김 경훈(UNIST)



# 목차

01 konlpy 설치

**02 한국어 형태소 분석기**

03 단어 출현 빈도 분석

# 한국어 형태소 분석기

## ■ 현재 5개의 한국어 형태소 분석기가 있음

- **Hannanum** (한나눔, KAIST, 1999)
- **Kkma** (꼬꼬마, 서울대)
- **Komoran** (Shineware, 2013, 벤처기업)
- **Mecab** (일본 Kyoto 대)
  - 일본어 형태소 분석기로 개발된 것을 한국어 형태소 분석기로 변환
  - Window 버전은 지원하지 않음
- **Okt** (Open Korean Text, Will Hohyon Ryu – 개발 언어: Scala)
  - 이전 버전 이름은 Twitter

```
morphs(phrase, norm=False, stem=False)  
    Parse phrase to morphemes.  
  
nouns(phrase)  
    Noun extractor.  
  
phrases(phrase)  
    Phrase extractor.  
  
pos(phrase, norm=False, stem=False, join=False)
```

참고 : <http://konlpy.org/ko/latest/morph>

# 한국어 형태소 분석 및 품사 태깅

## ■ 형태소 분석 (morphological analysis)

- 자연어 문장을 형태소 열로 바꾸는 작업
  - 형태소(morpheme)
    - 의미의 최소 단위: 어근(원형), 어미, 접두사, 접미사, 명사, 대명사, 조사, 형용사, 부사, 동사, ...

## ■ 품사 태깅(POS tagging)

- 형태소의 뜻과 문맥을 고려하여, 품사를 붙임
- 품사(POS, part-of-speech)

```
>>> mlist = kkma.pos(u'아버지 가방에 들어가신다.')
>>> print(mlist)
[('아버지', 'NNG'), ('가방', 'NNG'), ('에', 'JKM'), ('들어가', 'VV'), ('시', 'EPH'), ('니다', 'EFN'), ('.', 'SF')]
```

- **NNG**: 보통명사, **JKM** : 부사격 조사
- **VV**: 동사, **EPH**: 존칭 선어말 어미
- **EFN**: 평서형 종결 어미, **SF**: 마침표. 물음표. 느낌표

참고 : <http://konlpy.org/ko/latest/morph>

# 한국어 형태소 분석 : konlpy-intro. ipynb

## ■ 형태소 분석

```
>>> from konlpy.tag import Okt
>>> twitter = Okt()
>>> malist = twitter.pos("아버지 가방에 들어가신다.", norm=True, stem=True)
>>> print (malist)
[('아버지', 'Noun'), ('가방', 'Noun'), ('에', 'Josa'), ('들어가다', 'Verb'), ('.', 'Punctuation')]
```

- norm : "그래용ㅋㅋ?" → "그렇다 "
- stem : 원형을 찾음. "들어가신다" → "들어가다"

## Practice #1: 형태소 분석, 품사 태깅

- 아래 명령을 실행시키고, 결과를 확인 하시오.

```
from konlpy.tag import Okt
twitter = Okt()

text = "한글 자연어 처리는 재밌다 이제부터 열심히 해야지 ㅎㅎㅎ"

pprint(twitter.morphs(text)) #형태소 분석
pprint(twitter.morphs(text, stem=True)) #형태소 분석 -> 어간 추출
pprint(twitter.nouns(text)) #명사 추출
pprint(twitter.phrases(text)) #어절 추출
pprint(twitter.pos(text)) #품사 추출
pprint(twitter.pos(text, join=True)) #형태소와 품사를 함께 출력

text2 = "이것도 되나욧ㅋㅋㅋ"

pprint(twitter.pos(text2))
pprint(twitter.pos(text2, norm=True))
pprint(twitter.pos(text2, norm=True, stem=True))
```

## ■ 대한민국 헌법

```
ko law.open('constitution.txt').read() #대한민국 헌법 말뭉치
```

[illegible]

# Konlpy 말뭉치 (2/2)

## ■ 대한민국 국회 의안 말뭉치 : 10개

```
from konlpy.corpus import kobill
```

```
kobill.open('1809890.txt').read() # '1809890.txt'~'1809899.txt' 파일로 구성
```

```
'지방공무원법 일부개정법률안\n\n(정의화의원 대표발의)\n\n의안\n번호\n1809890\n발의연월일 : 2010. 11. 12.\n발의자 : 정의화.이명수.김을동\n이사항\n초상권.여상규.안규백\n박영아.김정훈\n김학송 의원(10인)\n제한이유\n초등교육\n초등학교 저학년의 경우에도\n모든 아동의 따뜻한 사랑과 보살핌이 필요\n한 나이이나, 현재 공무원이 자녀를 양육하기 위하여 육아휴직을 할 수 있는 자녀의 나이는 만 6세 이하로 되어 있어 초등학교 저학년인\n자녀를 돌보기 위해서 해당 부모님은 일 자리를 그만 두어야 하고\n이는 공금출산의 욕을 저하시키는 문제로 이어질 수 있을 것임.\n\n따라서 육아휴직이 가능한 자녀의 연령을 만 8세 이하로 개정하려\n는 것임
```

# 목차

01 konlpy 설치

02 한국어 형태소 분석기

**03 단어 출현 빈도 분석**



# Counter 모듈: word-count. ipynb

## ■ Counter는 컬렉션(Collection)에 속한 모듈.

- Counter의 각 원소는 dictionary 구조.
  - key:원소, value: (발생)빈도.
- 문장에서 단어의 발생 빈도를 구할 때 편리하게 사용

```
from collections import Counter

s = 'aabbbcdddeeeefffff'

container = Counter(s)
#container
for key, val in container.items():
    print(key, ': ', val)
print()
print('원소=', set(container.elements()))
```

```
container.most_common(3) #빈도 순으로 3개 원소를 출력
```

```
[('f', 5), ('e', 4), ('b', 3)]
```



## '토지2'에서 단어 출현 빈도 (1/2) - Without Counter

### ■ 빈도 수 기준으로 상위 10개 단어 출력

```
nouns_tagger = Okt()
word_dic = {}

lines = text.split("\r\n")
for line in lines:
    noun_list = nouns_tagger.nouns(line)
    for word in noun_list:
        if not (word in word_dic):
            word_dic[word] = 1
        word_dic[word] += 1

keys = sorted(word_dic.items(), key=lambda x:x[1], reverse=True)
#빈도 수를 기준으로 정렬
for word, count in keys[:10]:
    print("{0}({1}) ".format(word, count), end="")
print()
```

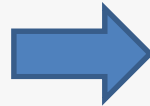
## '토지2'에서 단어 출현 빈도 (2/2) – With Counter

### ■ 빈도 수 기준으로 상위 10개 단어 출력

```
nouns_tagger = Okt()
word_dic = {}

lines = text.split("\r\n")
for line in lines:
    noun_list = nouns_tagger.nouns(line)
    for word in noun_list:
        if not (word in word_dic):
            word_dic[word] = 1
        word_dic[word] += 1

keys = sorted(word_dic.items(), key=lambda x:x[1], reverse=True)
#빈도 수를 기준으로 정렬
for word, count in keys[:10]:
    print("{0}({1})".format(word, count), end="")
print()
```



```
from collections import Counter

nouns_tagger = Okt()
noun_list = nouns_tagger.nouns(text)
container2 = Counter(noun_list)
container2.most_common(10)
```

```
[('식', 322),
 ('말', 217),
 ('시', 207),
 ('포수', 192),
 ('포구', 176),
 ('하', 174),
 ('아', 109),
 ('추물', 102),
 ('사', 99),
 ('생각', 99)]
```

## 한글 전처리 (1/3) – 무의미한 단어 제외

### ■ 한 글자인 단어 제거 : '것', '그', '말', ...

```
new_container = Counter({ x : container2[x] for x in container2 if len(x) > 1})  
new_container.most_common(10)
```

Left box (red border):

('것', 322)
('그', 217)
('말', 207)
('것', 192)
('것', 176)
('가', 174)
('아', 109)
('아', 102)
('사', 99)
('생', 99)

Right box (blue border):

('것', 192)
('것', 176)
('것', 102)
('사', 99)
('사', 99)
('사', 99)
('사', 72)
('사', 72)
('사', 62)
('사', 60)

3695개

3219개

## 한글 전처리 (2/3) – 불용어 제외

- **한국어 불용어 리스트** [https://www.ranks.nl/stopwords/Korean'](https://www.ranks.nl/stopwords/Korean)
  - 절대적 기준은 아님. 문장 종류에 따라 불용어(stopwords) 추가/수정/삭제.

아	어찌됐든
휴	그위에
아이구	게다가
아이쿠	점에서 보아
아이고	비추어 보아
어	고려하면
나	하게될것이다

- **불용어 리스트 : 675개 – 텍스트 파일에서 읽어 옴**

```
kr_stopwords = "kr_stopwords.txt"

with open(kr_stopwords, encoding='utf8') as f:
    raw_list = f.readlines()
    stopword_list = [x.strip() for x in raw_list]
```

- **불용어 제거 : 3,219개 → 3,143개**

```
new_container = Counter({x:new_container[x] for x in new_container if x not in stopword_list})
```

## 한글 전처리 (3/3): '토지' 불용어 추가

### ■ '토지'는 사투리가 포함된 의태어 등 토속적 단어를 많이 사용

- 다양한 단어를 사용 → 전체 단어 수 증가
  - 각 단어의 빈도 수는 낮음.
- 문장 분석에 도움이 되지 않는 단어는 추가로 불용어 사전에 추가

```
toji_stopwords = ['상위', '영영', '간혹', '아무', '여느', '무지', '우짜', '뚝딱', '오늘이', '가라', '아얏',  
                 '오냐', '종종', '와드득', '하는', '아니겄소', '곧잘', '겄너', '가야', '으이잉']  
  
for w in toji_stopwords:  
    stopword_list.append(w)
```



# 시각화를 위한 word cloud 설치 : pytagcloud

**konlpy 설치 : pip install pytagcloud pygame simplejson**

**설치 명령은 아나콘다 창에서 실행시켜야 함.**

Anaconda Prompt (Anaconda3)

```
(base) C:\Users\Wyshon>pip install pytagcloud pygame simplejson
Collecting pytagcloud
  Downloading https://files.pythonhosted.org/packages/a6/36/8626ddead8006c0
/pytagcloud-0.3.5.tar.gz (754kB)
    |████████████████████████████████████████| 757kB 930kB/s
Collecting pygame
  Downloading https://files.pythonhosted.org/packages/ed/56/b63ab3724acff69
/pygame-1.9.6-cp37-cp37m-win_amd64.whl (4.3MB)
    |████████████████████████████████████████| 4.3MB 3.3MB/s
Collecting simplejson
  Downloading https://files.pythonhosted.org/packages/ae/fd/36160c9ba8623b3
/simplejson-3.17.2-cp37-cp37m-win_amd64.whl (73kB)
    |████████████████████████████████████████| 81kB 5.5MB/s
Building wheels for collected packages: pytagcloud
  Building wheel for pytagcloud (setup.py) ... done
  Created wheel for pytagcloud: filename=pytagcloud-0.3.5-cp37-none-any.whl
b0b2863eb82d3e766abe3b022ea847a2b0c13b2
  Stored in directory: C:\Users\Wyshon\AppData\Local\pip\Cache\wheels\08\bf\W
828c8c1f
Successfully built pytagcloud
Installing collected packages: pytagcloud, pygame, simplejson
Successfully installed pygame-1.9.6 pytagcloud-0.3.5 simplejson-3.17.2
```



# 한글 폰트(NamuGothic) 다운로드 및 설치 (1/2)

## ■ 한글 폰트 다운로드

① [hangeul.naver.com/webfont/NanumGothic/NanumGothic.ttf](https://hangeul.naver.com/webfont/NanumGothic/NanumGothic.ttf)

## ■ 폰트를 pytagcloud가 설치된 fonts 폴더에 복사

로컬 디스크 (C:) > 사용자 > yshon > Anaconda3 > Lib > site-packages

pyrsistent	2020-01-22 오후 7:20
pyrsistent-0.15.6.dist-info	2020-01-22 오후 7:17
PySocks-1.7.1.dist-info	2020-01-22 오후 7:03
pytagcloud	2020-07-30 오후 1:49
pytagcloud-0.3.5.dist-info	2020-07-30 오후 1:49

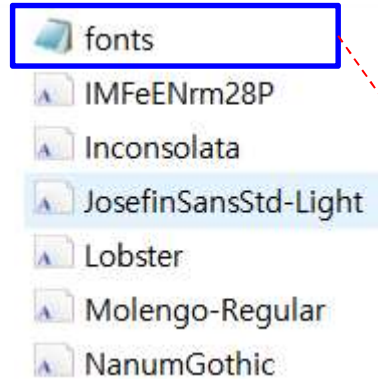
Anaconda3 > Lib > site-packages > pytagcloud > fonts

fonts  
IMFeENrm28P  
Inconsolata  
JosefinSansStd-Light  
Lobster  
Molengo-Regular  
NanumGothic

## 한글 폰트(NamuGothic) 다운로드 및 설치 (2/2)

### ■ pytagcloud가 설치된 fonts 폴더에 있는 fonts.json 수정

Anaconda3 > Lib > site-packages > pytagcloud > fonts



**fonts.json → 메모장 → 아래 내용으로 수정**

```
{
  "name": "NanumGothic",
  "ttf": "NanumGothic.ttf",
  "web": "http://fonts.googleapis.com/css?family=Nanum+Gothic"
}
```

### ■ Jupyter 노트북 파일 모두 닫고 Anaconda 실행 종료

### ■ 다시 Anaconda 실행

# 한국어 분석 - '토지'에서 단어 출현 빈도 (1/2)

## ■ 출현 빈도 순으로 40개 단어 선택

```
# 발생 빈도 순으로 40개 단어 선택
ranked_tags = new_container.most_common(40)

# 워드 클라우드(word cloud)로 출력할 단어 리스트 : ranked_tags
# 단어 최대 크기(maxsize): 80
taglist = pytagcloud.make_tags(ranked_tags, maxsize=80)
```

## ■ 워드 클라우드 이미지 생성

```
# 워드 클라우드 이미지 생성(wordcloud.jpg)
# 폰트: NanumGothic
pytagcloud.create_tag_image(taglist, 'wordcloud.jpg', size=(900, 600),
                             fontname='NanumGothic', rectangular=False)
```

## ■ Jupyter notebook에서 이미지 출력

```
from IPython.display import Image
Image(filename='wordcloud.jpg')
```

## 한국어 분석 - '토지'에서 단어 출현 빈도 (2/2)



## Practice #2: '대한민국 헌법'에서 단어 출현 빈도

- '토지'에서 했던 방식대로, '대한민국 헌법'에서 단어 출현 빈도를 조사하시오(결과는 word cloud로 출력).

