자연어 처리, NLP(Natural Language Processing)

KoNLPy 및 필요 모듈의 설치

> - KoNLPy : pip install konlpy
> - JPype1 : conda install -c conda-forge jpype1
> -
> - 이후 Jupyter Notebook 재실행 필요

In [5]:
```
!pip install konlpy
```

```
Collecting konlpy
  Downloading https://files.pythonhosted.org/packages/e5/3d/4e983cd98d87b50b2ab0387d73fa946f745aa8164e8888a714d5129f9765/konlpy-0.5.1-py2.py3-none-any.whl (https://files.pythonhosted.org/packages/e5/3d/4e983cd98d87b50b2ab0387d73fa946f745aa8164e8888a714d5129f9765/konlpy-0.5.1-py2.py3-none-any.whl) (19.4MB)
Collecting JPype1>=0.5.7 (from konlpy)
  Downloading https://files.pythonhosted.org/packages/d3/08/f4bb58c1c0dff93e9628cd0e1025f80fcb5a4551310455feb96b96e58ad1/JPype1-0.7.0-cp37-cp37m-win_amd64.whl (https://files.pythonhosted.org/packages/d3/08/f4bb58c1c0dff93e9628cd0e1025f80fcb5a4551310455feb96b96e58ad1/JPype1-0.7.0-cp37-cp37m-win_amd64.whl) (1.2MB)
Installing collected packages: JPype1, konlpy
Successfully installed JPype1-0.7.0 konlpy-0.5.1
```

In [1]:
```
from IPython.display import Image as Show
```

In [2]:
```
import platform

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# sns.set()

from matplotlib import font_manager, rc
if platform.system() == 'Darwin':
    rc('font', family='AppleGothic')
elif platform.system() == 'Windows':
    font_name = font_manager.FontProperties(fname="C:/Windows/Fonts/malgun.ttf").get_nam
    rc('font', family=font_name)
else:
    print("It's unknown system. Hangul fonts are not supported!")

# plt.rcParams['axes.unicode_minus'] = False
plt.rcParams["figure.figsize"] = [12,6]

%matplotlib inline
```

한글 자연어 처리 기초

In [3]:
```
from konlpy.tag import Kkma
kkma = Kkma()
```

```
C:₩Python₩Anaconda3₩lib₩site-packages₩jpype₩_core.py:210: UserWarning:
-------------------------------------------------------------------------------
Deprecated: convertStrings was not specified when starting the JVM. The default
behavior in JPype will be False starting in JPype 0.8. The recommended setting
for new code is convertStrings=False.  The legacy value of True was assumed for
this session. If you are a user of an application that reported this warning,
please file a ticket with the developer.
-------------------------------------------------------------------------------

  """)
```

In [4]:
```
kkma.sentences('한국어 분석을 시작합니다 재미있어요~')
```

Out[4]:
```
['한국어 분석을 시작합니다', '재미있어요~']
```

In [5]:
```
kkma.nouns('한국어 분석을 시작합니다 재미있어요~~')
```

Out[5]:
```
['한국어', '분석']
```

In [6]:
```
kkma.pos('한국어 분석을 시작합니다 재미있어요~~')
```

Out[6]:
```
[('한국어', 'NNG'),
 ('분석', 'NNG'),
 ('을', 'JKO'),
 ('시작하', 'VV'),
 ('ㅂ니다', 'EFN'),
 ('재미있', 'VA'),
 ('어요', 'EFN'),
 ('~~', 'SW')]
```

In [7]:
```
from konlpy.tag import Hannanum
```

In [8]:
```
hannanum = Hannanum()
```

In [9]:
```
hannanum.nouns('한국어 분석을 시작합니다 재미있어요~~')
```

Out[9]:
```
['한국어', '분석', '시작']
```

In [10]:
```
hannanum.morphs('한국어 분석을 시작합니다 재미있어요~~')
```

Out[10]:
```
['한국어', '분석', '을', '시작', '하', 'ㅂ니다', '재미있', '어요', '~~']
```

In [11]:
```
hannanum.pos('한국어 분석을 시작합니다 재미있어요~~')
```

Out[11]:
```
[('한국어', 'N'),
 ('분석', 'N'),
 ('을', 'J'),
 ('시작', 'N'),
 ('하', 'X'),
 ('ㅂ니다', 'E'),
 ('재미있', 'P'),
 ('어요', 'E'),
 ('~~', 'S')]
```

In [12]:
```
from konlpy.tag import Twitter
t = Twitter()
```

```
C:₩Python₩Anaconda3₩lib₩site-packages₩konlpy₩tag₩_okt.py:16: UserWarning: "Twitter" has changed to "Okt" since KoNLPy v0.4.5.
  warn('"Twitter" has changed to "Okt" since KoNLPy v0.4.5.')
```

In [14]:
```
t.nouns('한국어 분석을 시작합니다 재미있어요~~')
```

Out[14]:
```
['한국어', '분석', '시작']
```

In [15]:
```
t.morphs('한국어 분석을 시작합니다 재미있어요~~')
```

Out[15]:
```
['한국어', '분석', '을', '시작', '합니다', '재미있어요', '~~']
```

In [16]:
```
t.pos('한국어 분석을 시작합니다 재미있어요~~')
```

Out[16]:
```
[('한국어', 'Noun'),
 ('분석', 'Noun'),
 ('을', 'Josa'),
 ('시작', 'Noun'),
 ('합니다', 'Verb'),
 ('재미있어요', 'Adjective'),
 ('~~', 'Punctuation')]
```

KoNLPy 한국어 처리

In [17]:
```
from konlpy.corpus import kolaw
from konlpy.utils import concordance
```

In [19]:
```
constitution = kolaw.open('constitution.txt').read()
```

In [20]:
```
print(constitution )
```

```
대한민국헌법

유구한 역사와 전통에 빛나는 우리 대한국민은 3·1운동으로 건립된 대한민국임시정부의 법통과 불의에 항거한 4·19민주이념을 계승하고, 조국의 민주개혁과 평화적 통일의 사명에 입각하여 정의·인도와 동포애로써 민족의 단결을 공고히 하고, 모든 사회적 폐습과 불의를 타파하며, 자율과 조화를 바탕으로 자유민주적 기본질서를 더욱 확고히 하여 정치·경제·사회·문화의 모든 영역에 있어서 각인의 기회를 균등히 하고, 능력을 최고도로 발휘하게 하며, 자유와 권리에 따르는 책임과 의무를 완수하게 하여, 안으로는 국민생활의 균등한 향상을 기하고 밖으로는 항구적인 세계평화와 인류공영에 이바지함으로써 우리들과 우리들의 자손의 안전과 자유와 행복을 영원히 확보할 것을 다짐하면서 1948년 7월 12일에 제정되고 8차에 걸쳐 개정된 헌법을 이제 국회의 의결을 거쳐 국민투표에 의하여 개정한다.

        제1장 총강
  제1조 ① 대한민국은 민주공화국이다.
②대한민국의 주권은 국민에게 있고, 모든 권력은 국민으로부터 나온다.
  제2조 ① 대한민국의 국민이 되는 요건은 법률로 정한다.
②국가는 법률이 정하는 바에 의하여 재외국민을 보호할 의무를 진다.
  제3조 대한민국의 영토는 한반도와 그 부속도서로 한다.
```

In [21]:
```
concordance(u'민주', constitution, show=False) #u = 유니코드
```

Out[21]:
```
[13, 16, 35, 99, 136, 219, 251, 1026, 2850, 2854, 3649]
```

In [22]:
```
concordance(u'대한민국',constitution, show=True)
```

```
0       대한민국헌법 유구한 역사와
9       대한국민은 3·1운동으로 건립된 대한민국임시정부의 법통과 불의에
98      총강 제1조 ① 대한민국은 민주공화국이다. ②대한민국의
100     ① 대한민국은 민주공화국이다. ②대한민국의 주권은 국민에게
110     나온다. 제2조 ① 대한민국의 국민이 되는
126     의무를 진다. 제3조 대한민국의 영토는 한반도와
133     부속도서로 한다. 제4조 대한민국은 통일을 지향하며,
147     추진한다. 제5조 ① 대한민국은 국제평화의 유지에
787     군무원이 아닌 국민은 대한민국의 영역안에서는 중대한
1836    파견 또는 외국군대의 대한민국 영역안에서의 주류에
3620    경제 제119조 ① 대한민국의 경제질서는 개인과
```

Out[22]:
```
[0, 9, 98, 100, 110, 126, 133, 147, 787, 1836, 3620]
```

꼬꼬마, Kkma

In [23]:
```
from konlpy.tag import Kkma
from konlpy.utils import pprint
```

In [24]:
```
kkma = Kkma()
```

In [25]:
```
text = u'네, 안녕하세요. 반갑습니다.'
```

In [27]:
```
text_s = pprint(kkma.sentences(text))
text_s
```

```
['네, 안녕하세요.', '반갑습니다.']
```

In [28]:
```
text_s = kkma.sentences(text)
text_s
```

Out[28]:
```
['네, 안녕하세요.', '반갑습니다.']
```

In [29]:
```
type(text_s), text_s[0], text_s[-1]
```

Out[29]:
```
(list, '네, 안녕하세요.', '반갑습니다.')
```

In [30]:
```
kkma.tagset
```

Out[30]:
```
{'EC': '연결 어미',
 'ECD': '의존적 연결 어미',
 'ECE': '대등 연결 어미',
 'ECS': '보조적 연결 어미',
 'EF': '종결 어미',
 'EFA': '청유형 종결 어미',
 'EFI': '감탄형 종결 어미',
 'EFN': '평서형 종결 어미',
 'EFO': '명령형 종결 어미',
 'EFQ': '의문형 종결 어미',
 'EFR': '존칭형 종결 어미',
 'EP': '선어말 어미',
 'EPH': '존칭 선어말 어미',
 'EPP': '공손 선어말 어미',
 'EPT': '시제 선어말 어미',
 'ET': '전성 어미',
 'ETD': '관형형 전성 어미',
 'ETN': '명사형 전성 어미',
 'IC': '감탄사',
 'JC': '접속 조사',
 'JK': '조사',
 'JKC': '보격 조사',
 'JKG': '관형격 조사',
 'JKI': '호격 조사',
 'JKM': '부사격 조사',
 'JKO': '목적격 조사',
 'JKQ': '인용격 조사',
 'JKS': '주격 조사',
 'JX': '보조사',
 'MA': '부사',
 'MAC': '접속 부사',
 'MAG': '일반 부사',
 'MD': '관형사',
 'MDN': '수 관형사',
 'MDT': '일반 관형사',
 'NN': '명사',
 'NNB': '일반 의존 명사',
 'NNG': '보통명사',
 'NNM': '단위 의존 명사',
 'NNP': '고유명사',
 'NP': '대명사',
 'NR': '수사',
 'OH': '한자',
 'OL': '외국어',
 'ON': '숫자',
 'SE': '줄임표',
 'SF': '마침표, 물음표, 느낌표',
 'SO': '붙임표(물결,숨김,빠짐)',
 'SP': '쉼표,가운뎃점,콜론,빗금',
 'SS': '따옴표,괄호표,줄표',
 'SW': '기타기호 (논리수학기호,화폐기호)',
 'UN': '명사추정범주',
 'VA': '형용사',
 'VC': '지정사',
 'VCN': "부정 지정사, 형용사 '아니다'",
 'VCP': "긍정 지정사, 서술격 조사 '이다'",
 'VV': '동사',
 'VX': '보조 용언',
 'VXA': '보조 형용사',
 'VXV': '보조 동사',
 'XP': '접두사',
```

```
 'XPN': '체언 접두사',
 'XPV': '용언 접두사',
 'XR': '어근',
 'XSA': '형용사 파생 접미사',
 'XSN': '명사파생 접미사',
 'XSV': '동사 파생 접미사'}
```

In [31]:
```
text = '자연어처리는 재미있습니다. 그러나 한국어 분석은 쉽지않습니다.'
```

In [32]:
```
# 명사 추출기 nouns
text_nouns = kkma.nouns(text)
text_nouns
```

Out[32]:
```
['자연어', '자연어처리', '처리', '한국어', '분석']
```

In [34]:
```
# 형태소 해석
text_morphs = kkma.morphs(text)
text_morphs
```

Out[34]:
```
['자연어',
 '처리',
 '는',
 '재미있',
 '습니다',
 '.',
 '그러',
 '나',
 '한국어',
 '분석',
 '은',
 '쉽',
 '지',
 '않',
 '습니다',
 '.']
```

In [35]:
```
# 문장 검색, Sentence detection.
text_sentences = kkma.sentences(text)
text_sentences
```

Out[35]:
```
['자연어처리는 재미있습니다.', '그러나 한국어 분석은 쉽지 않습니다.']
```

In [36]:
```
len(text_sentences)
```

Out[36]:
```
2
```

In [37]:
```
# POS 태그
pos_tagger = kkma.pos(text)
pos_tagger
```

Out[37]:
```
[('자연어', 'NNG'),
 ('처리', 'NNG'),
 ('는', 'JX'),
 ('재미있', 'VA'),
 ('습니다', 'EFN'),
 ('.', 'SF'),
 ('그러', 'VV'),
 ('나', 'ECE'),
 ('한국어', 'NNG'),
 ('분석', 'NNG'),
 ('은', 'JX'),
 ('쉽', 'VA'),
 ('지', 'ECD'),
 ('않', 'VXV'),
 ('습니다', 'EFN'),
 ('.', 'SF')]
```

In [38]:
```
pos_tagger_f = kkma.pos(text, flatten=False)
pos_tagger_f
```

Out[38]:
```
[[('자연어', 'NNG'), ('처리', 'NNG'), ('는', 'JX')],
 [('재미있', 'VA'), ('습니다', 'EFN'), ('.', 'SF')],
 [('그러', 'VV'), ('나', 'ECE')],
 [('한국어', 'NNG')],
 [('분석', 'NNG'), ('은', 'JX')],
 [('쉽', 'VA'), ('지', 'ECD')],
 [('않', 'VXV'), ('습니다', 'EFN'), ('.', 'SF')]]
```

In [39]:
```
len(pos_tagger), type(pos_tagger), type(pos_tagger[0])
```

Out[39]:
```
(16, list, tuple)
```

In [40]:
```
len(pos_tagger_f), type(pos_tagger_f), type(pos_tagger_f)
```

Out[40]:
```
(7, list, list)
```

In [41]:
```
text = '''
하늘아래 땅이있고 그위에 내가있으니
어디인들 이내몸 둘곳이야 없으리
하루해가 저문다고 울터이냐 그리도 내가 작더냐
별이 지는 저 산넘어 내 그리 쉬어 가리라
바람아 불어라 이내몸을 날려 주려마
하늘아 구름아 내몸 실어 떠나 가련다

해가지고 달이뜨고 그안에 내가숨쉬니
어디인들 이내몸 갈곳이야 없으리
작은것을 사랑하며 살터이다 친구를 사랑하리라
말이없는 저들녘에 내님을 그려보련다
바람아 불어라 이내몸을 날려주려마
하늘아 구름아 내몸실어 떠나가련다
바람아 불어라 이내몸을 날려주려마
하늘아 구름아 내몸실어 떠나가련다
'''
```

In [42]:
```
text_sentences = kkma.sentences(text)
text_sentences
```

Out[42]:
```
['하늘 아래 땅이 있고 그 위에 내가 있으니 어디인들 이내 몸 둘 곳이야 없으리',
 '하루해가 저문다고 울 터이냐',
 '그리도 내가 작더냐',
 '별이 지는 저 산 넘어 내 그리 쉬어 가리라 바람 아 불어라',
 '이내 몸을 날려 주려 마 하늘 아 구름 아 내 몸 실어 떠나 가련 다 해 가지고 달이 뜨고 그
안에 내가 숨쉬니 어디인들 이내 몸 갈 곳이야 없으리',
 '작은 것을 사랑하며 살 터이다',
 '친구를 사랑하리라',
 '말이 없는 저 들녘에 내 님을 그려 보련 다 바람 아 불어라',
 '이내 몸을 날려 주려 마 하늘 아 구름 아 내 몸 실어 떠나가련',
 '다 바람 아 불어라',
 '이내 몸을 날려 주려 마 하늘 아 구름 아 내 몸 실어 떠나가련',
 '다']
```

In [43]:
```
len(text_sentences)
```

Out[43]: 12

In [44]:
```
constitution = kolaw.open('constitution.txt').read()
```

In [45]:
```
pos_const = kkma.pos(constitution)
```

In [46]:
```
len(pos_const)
```

Out[46]: 10053

In [47]:
```
# 보통명사만 추출
pos_const_NNG = [ word[0] for word in pos_const if word[1]=='NNG']
len(pos_const_NNG)
```

Out[47]: 3433

In [48]:
```
pos_const_NNG.sort()
```

In [49]:
```
pos_const_NNG[:10], pos_const_NNG[-10:]
```

Out[49]:
```
(['가격', '가부', '가입', '가입', '가족', '가족', '가치', '각급', '각급', '각급'],
 ['후보자', '후보자', '후임자', '후임자', '후임자', '후임자', '훈련', '훈장', '훈장',
'히'])
```

In [53]:
```
# 모든 명사 추출
NN_list = [ 'NN', 'NNB', 'NNG', 'NNM', 'NNP', 'NP' ]
pos_const_NN = [ word[0] for word in pos_const if word[1] in NN_list]
len(pos_const_NN)
```

Out[53]: 3883

In [54]:
```
pos_const_NN.sort()
pos_const_NN[:10], pos_const_NN[-10:]
```

Out[54]:
```
(['가격', '가부', '가입', '가입', '가족', '가족', '가치', '각급', '각급', '각급'],
 ['후보자', '후보자', '후임자', '후임자', '후임자', '후임자', '훈련', '훈장', '훈장',
'히'])
```

In [56]:
```
len(set(pos_const_NN))
```

Out[56]:
```
938
```

In [57]:
```
def getNounCnt(pos_list) :
    noun_cnt = {}

    for noun in pos_list :
        if noun_cnt.get(noun) :
            noun_cnt[noun] +=1
        else :
            noun_cnt[noun] = 1
    return noun_cnt
```

In [58]:
```
noun_dict = getNounCnt(pos_const_NN)
len(noun_dict), noun_dict
```

Out[58]:
```
(938,
 {'가격': 1,
  '가부': 1,
  '가입': 2,
  '가족': 2,
  '가치': 1,
  '각급': 3,
  '각부': 9,
  '각인': 1,
  '간': 4,
  '간의': 2,
  '간첩죄': 1,
  '감봉': 1,
  '감사': 7,
  '감사원': 5,
  '감사원장': 2,
  '감찰': 1,
  '감형': 3,
  '강': 1,
```

In [59]:
```
from collections import Counter
```

In [60]:
```
counter = Counter(pos_const)
counter.most_common(10)
```

Out[60]:
```
[(('의', 'JKG'), 532),
 (('.', 'SF'), 359),
 (('하', 'XSV'), 350),
 (('에', 'JKM'), 328),
 (('ㄴ다', 'EFN'), 243),
 (('ㄴ', 'ETD'), 234),
 (('을', 'JKO'), 211),
 (('은', 'JX'), 182),
 (('는', 'JX'), 180),
 (('저', 'NP'), 155)]
```

In [61]:
```
counter = Counter(noun_dict)
counter.most_common(10)
```

Out[61]:
```
[('저', 155),
 ('법률', 121),
 ('수', 88),
 ('대통령', 84),
 ('국가', 73),
 ('국민', 69),
 ('헌법', 69),
 ('조', 58),
 ('국회', 55),
 ('때', 55)]
```

한나눔, Hannanum

In [62]:
```
from konlpy.tag import Hannanum
```

In [63]:
```
hannanum = Hannanum()
```

In [64]:
```
hannanum.tagset
```

Out[64]:
```
{'E': '어미',
 'EC': '연결 어미',
 'EF': '종결 어미',
 'EP': '선어말어미',
 'ET': '전성 어미',
 'F': '외국어',
 'I': '독립언',
 'II': '감탄사',
 'J': '관계언',
 'JC': '격조사',
 'JP': '서술격 조사',
 'JX': '보조사',
 'M': '수식언',
 'MA': '부사',
 'MM': '관형사',
 'N': '체언',
 'NB': '의존명사',
 'NC': '보통명사',
 'NN': '수사',
 'NP': '대명사',
 'NQ': '고유명사',
 'P': '용언',
 'PA': '형용사',
 'PV': '동사',
 'PX': '보조 용언',
 'S': '기호',
 'X': '접사',
 'XP': '접두사',
 'XS': '접미사'}
```

In [65]:
```
text = '자연어처리는 재미있습니다. 그러나 한국어 분석은 쉽지않습니다.'
```

In [67]:
```
# 명사 추출기, Noun extractor
text_nouns = hannanum.morphs(text)
text_nouns
```

Out[67]:
```
['자연어처리',
 '는',
 '재미있',
 '습니다',
 '.',
 '그러나',
 '한국어',
 '분석',
 '은',
 '쉽',
 '지',
 '않',
 '습니다',
 '.']
```

In [68]:
```
# 형태소 해석, Parse phrase to morphemes
text_morphs = hannanum.morphs(text)
text_morphs
```

Out[68]:
```
['자연어처리',
 '는',
 '재미있',
 '습니다',
 '.',
 '그러나',
 '한국어',
 '분석',
 '은',
 '쉽',
 '지',
 '않',
 '습니다',
 '.']
```

In [69]:
```
# 구문 분석, Phrase analyzer
text_analyze = hannanum.analyze(text)
text_analyze
```

Out[69]:
```
[[[('자연어처리', 'ncpa'), ('는', 'jxc')],
  [('자연어', 'ncn'), ('처리', 'ncpa'), ('는', 'jxc')]],
 [[('재미있', 'pvg'), ('습니다', 'ef')],
  [('재미', 'ncn'), ('있', 'xsmn'), ('습니다', 'ef')]],
 [[('.', 'sf')], [('.', 'sy')]],
 [],
 [[('그러나', 'maj')]],
 [[('한국어', 'ncn')]],
 [[('분석', 'ncpa'), ('은', 'jxc')], [('분석', 'ncpa'), ('은', 'ncn')]],
 [[('쉽', 'paa'), ('지', 'ecx'), ('않', 'px'), ('습니다', 'ef')]],
 [[('.', 'sf')], [('.', 'sy')]]]
```

In [70]:
```
# POS tagger
text_pos = hannanum.pos(text)
text_pos
```

Out[70]:
```
[('자연어처리', 'N'),
 ('는', 'J'),
 ('재미있', 'P'),
 ('습니다', 'E'),
 ('.', 'S'),
 ('그러나', 'M'),
 ('한국어', 'N'),
 ('분석', 'N'),
 ('은', 'J'),
 ('쉽', 'P'),
 ('지', 'E'),
 ('않', 'P'),
 ('습니다', 'E'),
 ('.', 'S')]
```

In [71]:
```
text_pos_f = hannanum.pos(text, flatten=False)
text_pos_f
```

Out[71]:
```
[[[('자연어처리', 'N'), ('는', 'J')]],
 [[('재미있', 'P'), ('습니다', 'E')]],
 [[('.', 'S')]],
 [],
 [[('그러나', 'M')]],
 [[('한국어', 'N')]],
 [[('분석', 'N'), ('은', 'J')]],
 [[('쉽', 'P'), ('지', 'E'), ('않', 'P'), ('습니다', 'E')]],
 [[('.', 'S')]]]
```