

LG전자 H&A본부 - 한양대
DX Intensive Course

자연어처리 – 자연어처리 개요와 실습

김 종 우 교수



목차

- 자연어 처리 개요
- 자연어 처리 기본 패키지
- 키워드 빈도 분석
- 한글 자연어 처리 패키지 설치
- 토큰화와 형태소 분석

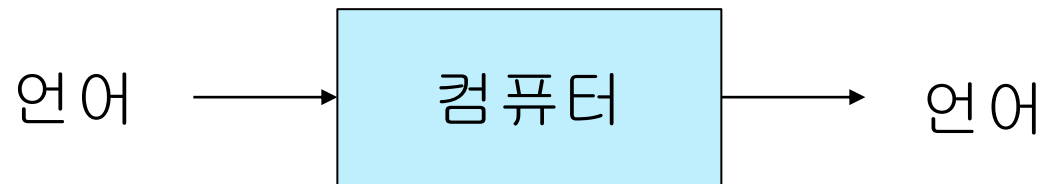
자연어 처리 개요

목 차

- 자연어 처리?
- 자연어 처리의 어려움
- 자연어 처리 관련 용어들

자연어 처리

- Natural Language Processing (NLP)
- 컴퓨터가 자연어를 이해하거나 생성할 수 있도록 하는 학문 분야
- 자연어 이해와 자연어 생성



Natural Language Understanding

Natural Language Generation

자연어 처리의 응용

- 기계번역
- 음성 인식(Speech-To-Text)
- 음성 생성(Text-To-Speech)
- 개인비서 서비스
 - 애플 Siri, 구글 Assistant, 아마존 Alexa, 마이크로소프트 Cortana, 삼성 빅스비, 네이버 클로버, KT 기가지니
- 문서 요약
- Q&A
- 감성분석
- ...



자연어 처리의 어려움

- 언어의 중의성
 - Ambiguity
 - 단어의 중의성
 - 배
 - 과일, 운송 수단, 신체의 일부, 크기 비교
 - 문장의 중의성
 - 너 참 잘 한다
- 규칙의 예외
- 언어의 유연성과 확장성
 - 신조어
- 언어 효과

자연어처리 연구의 패러다임

- 규칙 기반

- 언어의 문법적인 규칙을 사전에 정의해두고 그것에 기반하여 자연어를 처리하는 방식
 - 한계점
 - 언어의 규칙이 많고 예외가 많음

- 통계 기반

- 조건부 확률 사용
- 통계적으로 가장 자연스러운 단어 찾기

- 기계학습 기반

- 학습 데이터를 통해서 스스로 패턴 찾아냄
- 딥러닝을 통한 심층 분석
- 전체적인 맥락 파악 가능

자연어 처리 관련 용어들

- 코퍼스(corpus)

- 말뭉치
- 해당 문제와 관련된 모든 텍스트들
- 라틴어의 “body”라는 의미
- 문서(document)들의 집합
- 한 문서는 한 묶음의 텍스트 집합으로, 데이터 집합의 한 행(row)에 속하는 것과 같음
- 현실에서는 코퍼스는 정적이지 않음. 따라서 시간에 따라 관리되어야 함
- 목적에 따라 문장을 분석한 내역을 같이 넣기도 함
 - 입력 문장 + 정답

자연어 처리 관련 용어들

- 음절(Syllable)

- 언어를 말하고 들을 때, 하나의 덩어리로 여겨지는 가장 작은 발화의 단위
- 한글
 - 초성(Onset), 중성(Nucleus), 종성(Coda)으로 이루어짐
 - 초성, 종성은 자음(Consonant, C), 중성은 모음(Vowel, V)
 - V, V+C, C+V, C+V+C
 - 아, 앓, 가, 갓
 - 한글의 경우 한 글자

자연어 처리 관련 용어들

- 형태소(Morpheme)

- 의미를 가지는 가장 작은 단위
 - 쪼개면 더 이상 기능이나 의미를 가지지 않음
- 실질 형태소(어휘 형태소)와 형식 형태소(문법 형태소)
 - 실질 형태소: 명사, 동사, 형용사, 부사
 - 형식 형태소: 조사, 어미
- 나는 컴퓨터 공부가 좋아
 - 실질 형태소: 나, 컴퓨터, 공부, 좋-
 - 형식 형태소: -는, -가, -아

자연어 처리 관련 용어들

- 어절

- 한 개 이상의 형태소가 모여 구성된 단어
- 띄어쓰기 단위와 거의 일치

바닷가에 왔더니

바다와 같이 당신이 생각만 나는구려

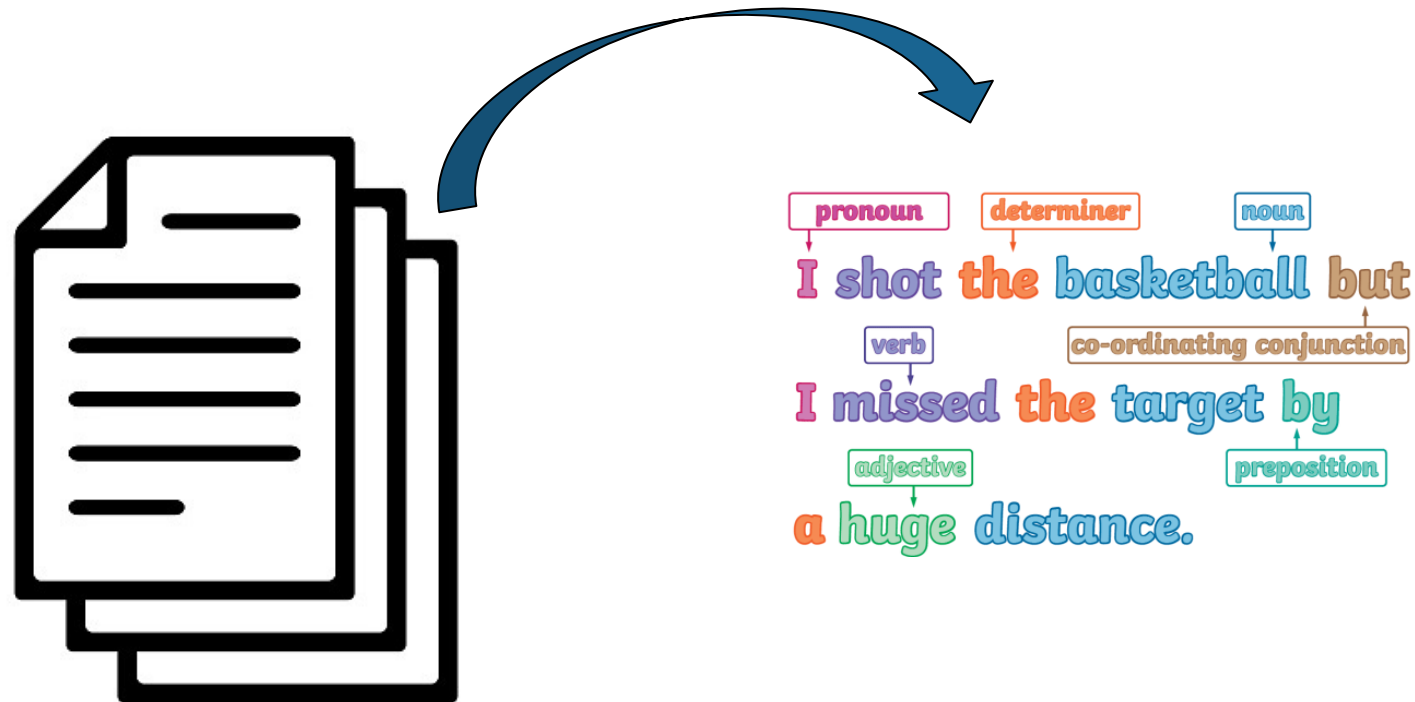
바다와 같이 당신을 사랑하고만 싶구려

- 백석, 바다 중에서

- 영어의 경우 단어(word, term)

자연어 처리 관련 용어들

- 코퍼스-문서-문장-어절-글자
- Corpus-Document-Sentence-Word-Character



자연어 처리 관련 용어들

- 불용어(Stop Words)

- 별다른 의미를 가지지 않는 단어들
- 여기서 “의미”는 문서들을 구분할 수 있는 능력을 의미
 - 예) 영어에서 the, and, of

자연어 처리 관련 용어들

- 어간 추출(Stemming)

- 단어의 어간(stem)을 추출하는 과정
- 어간은 추가적인 문법적 정보 없이 의미를 제공하는 기본 단어(base word) 또는 almost-word

자연어 처리 관련 용어들

- 단어 쌍(Word Pairs)

- 예) United Nations
- 예) 락 그룹 The Who
 - 대부분의 불용어 리스트들이 두 단어 모두 포함하고 있음
- 두 가지 해결책
 - 대문자 불용어는 유지, 또는 적어도 문장의 시작이 아닌데 대문자로 된 불용어는 유지
 - 많이 사용되는 단어 쌍 또는 구 찾기
 - 아마존의 서적 웹 사이트에서는 “statistically improbable phrases” 세션을 제공하고 있음. 이것을 통해서 예상보다 함께 나타나는 빈도가 높은 단어들을 찾을 수 있음
 - 예) “undirected data mining” and “automatic cluster detection”

자연어 처리 관련 용어들

- 단어 사전(어휘 목록, **Lexicon**)
 - 대부분의 비즈니스 텍스트 마이닝 응용에서 한 가지 또는 그 이상의 사전 사용
 - 사전은 중요한 단어들의 목록
 - 동의어들을 포함할 수 있음
 - 동일한 개념에 대한 다른 단어들
 - 틀린 철자
 - 예) 항공사의 고객 의견에서 “flight,” “fl,” “flt ” 는 모두 “flight”를 의미함

자연어 처리 관련 용어들

- 품사(Part of Speech, POS)

- 한글의 경우 문장 내에서 해당 단어가 수행하는 역할을 기준으로 5언으로 구분
 - 체언, 수식언, 관계언, 독립언, 용언
- 의미에 따라 9품사
 - 명사, 대명사, 수사, 관형사, 부사, 조사, 감탄사, 동사, 형용사

자연어 처리 관련 용어들

- 품사(Part of Speech, POS)

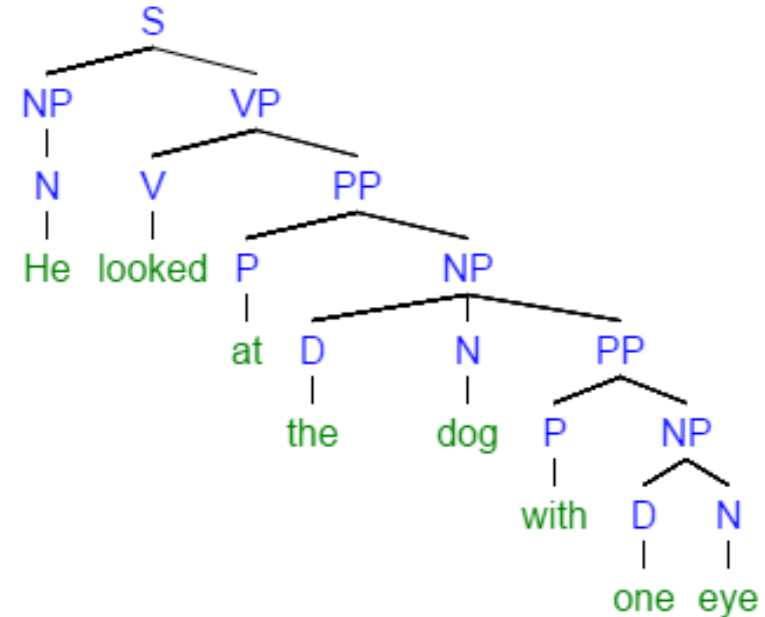
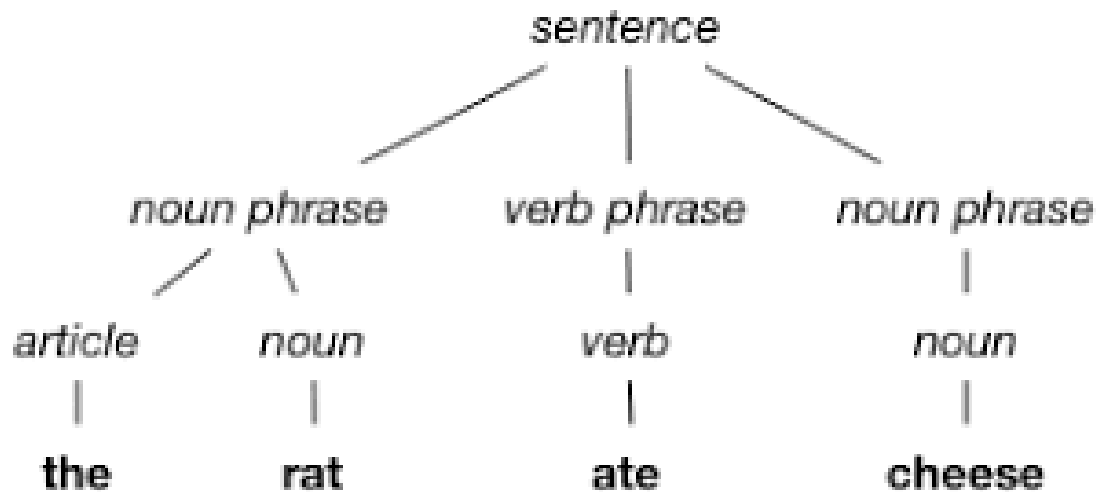
- 체언: 문장에서 몸통, 중심이 되는 역할
 - 명사, 대명사, 수사(수량, 순서를 나타내는 단어)
- 수식언: 다른 말을 꾸며주는 역할
 - 관형사와 부사
 - 관형사는 체언 앞에서 체언을 꾸며 줌
 - 새, 윗, 옛, 그, 다른, 무슨, 한, 두, 여러
 - 부사는 용언(동사와 형용사) 앞에서 내용을 꾸며주거나 문장 전체를 꾸밈
 - 매우, 잘, 반드시, 아마, 또는, 그리고, 왜냐하면, 그러나

자연어 처리 관련 용어들

- 품사(Part of Speech, POS)
 - 관계언
 - 조사
 - 독립언
 - 감탄사
 - 용언
 - 서술어 기능
 - 동사와 형용사

자연어 처리 관련 용어들

- 구(Phrase)
 - 영어 분석에 많이 사용
 - 문장(S)=명사구(NP, Noun Phrase)+동사구(VP, Verb Phrase)



자연어 처리의 접근법

- 단어 주머니(**bag of words**) 접근법 vs. 이해(Understanding) 접근법
 - 문서들은 단순히 문서들이 포함하고 있는 단어들의 집합으로 생각
 - “Man eats fish”
 - “Fish eats man”
 - 각 단어의 의미가 무엇인지, 문서 내용이 무엇인지 실제로 이해하려고 시도함

정리

- 자연어처리?
- 자연어처리의 어려움
- 자연어처리 관련 용어들
 - 음절
 - 형태소
 - 어절
 - 문장
 - 문서
 - 코퍼스
 - 품사

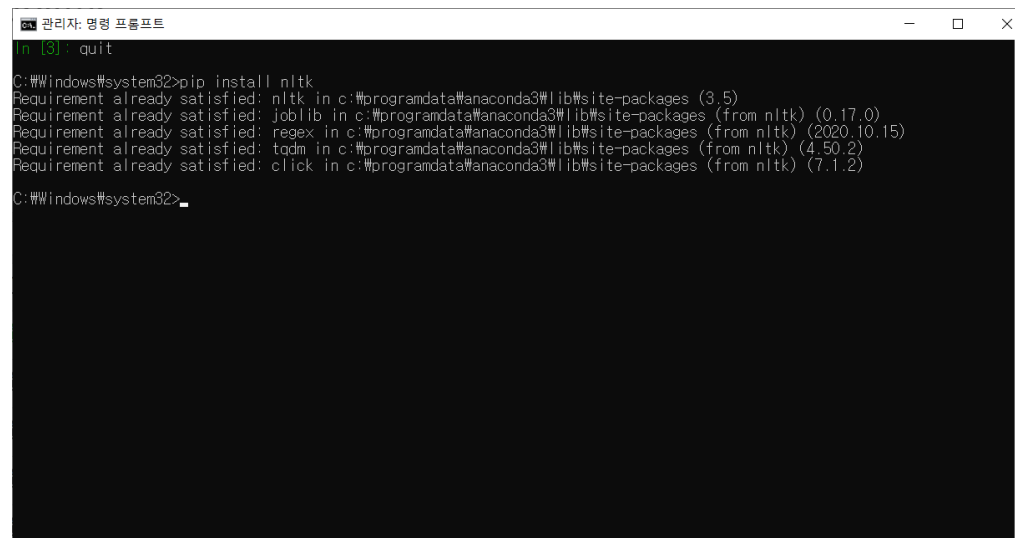
자연어 처리 기본 패키지

목차

- nltk
- konlpy

nltk 설치

- NLTK(Natural Language Toolkit)
 - 영어 텍스트 전처리에 가장 많이 사용
- 명령 Prompt 관리자 권한으로 실행
- `pip install nltk`
- 말뭉치 다운로드 필요



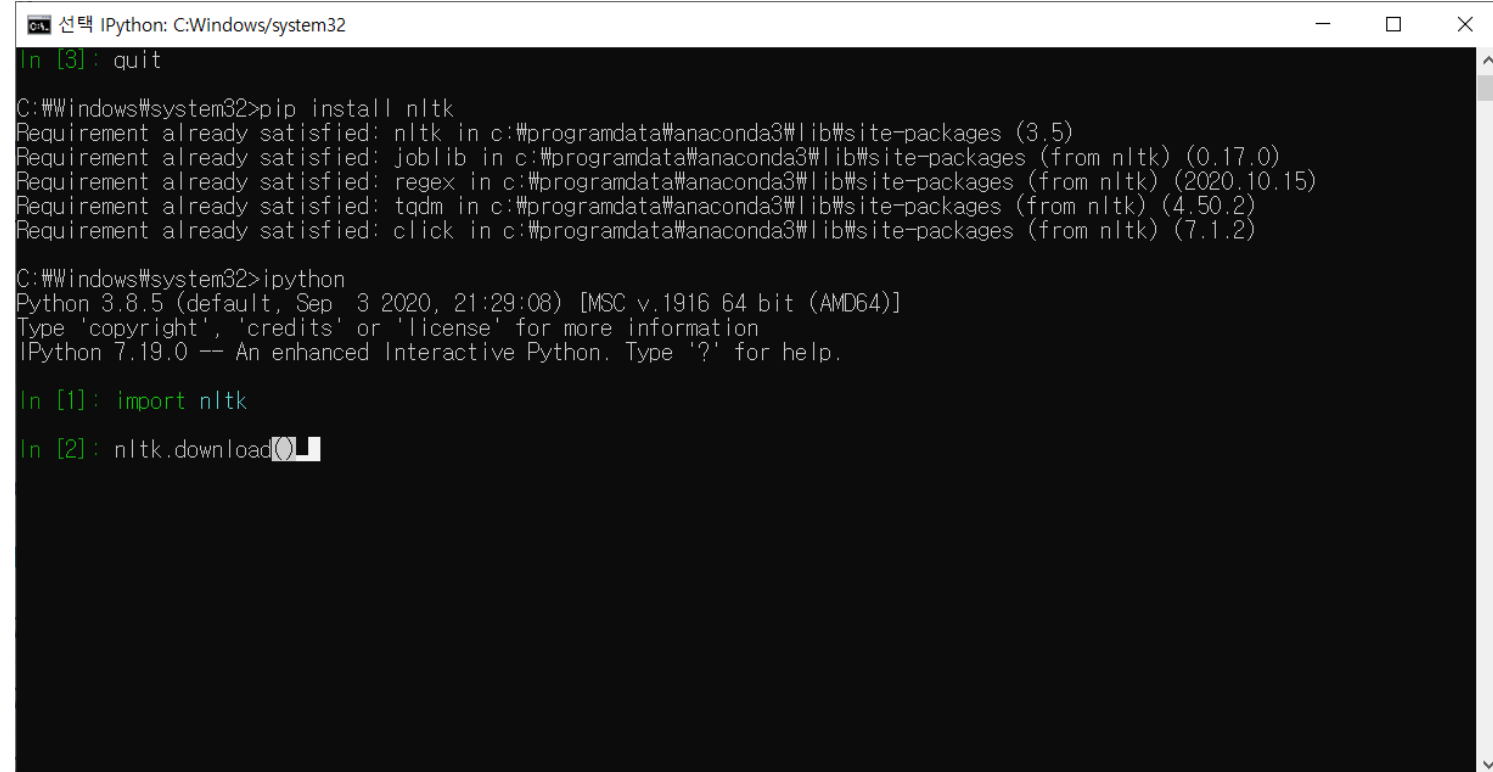
```
관리자: 명령 프롬프트
In [3]: quit

C:\Windows\system32>pip install nltk
Requirement already satisfied: nltk in c:\programdata\anaconda3\lib\site-packages (3.5)
Requirement already satisfied: joblib in c:\programdata\anaconda3\lib\site-packages (from nltk) (0.17.0)
Requirement already satisfied: regex in c:\programdata\anaconda3\lib\site-packages (from nltk) (2020.10.15)
Requirement already satisfied: tqdm in c:\programdata\anaconda3\lib\site-packages (from nltk) (4.50.2)
Requirement already satisfied: click in c:\programdata\anaconda3\lib\site-packages (from nltk) (7.1.2)

C:\Windows\system32>
```

nlTK 라이브러리 설치

- ipython
- import nltk
- nltk.download()



```
선택 IPython: C:\Windows\system32
In [3]: quit

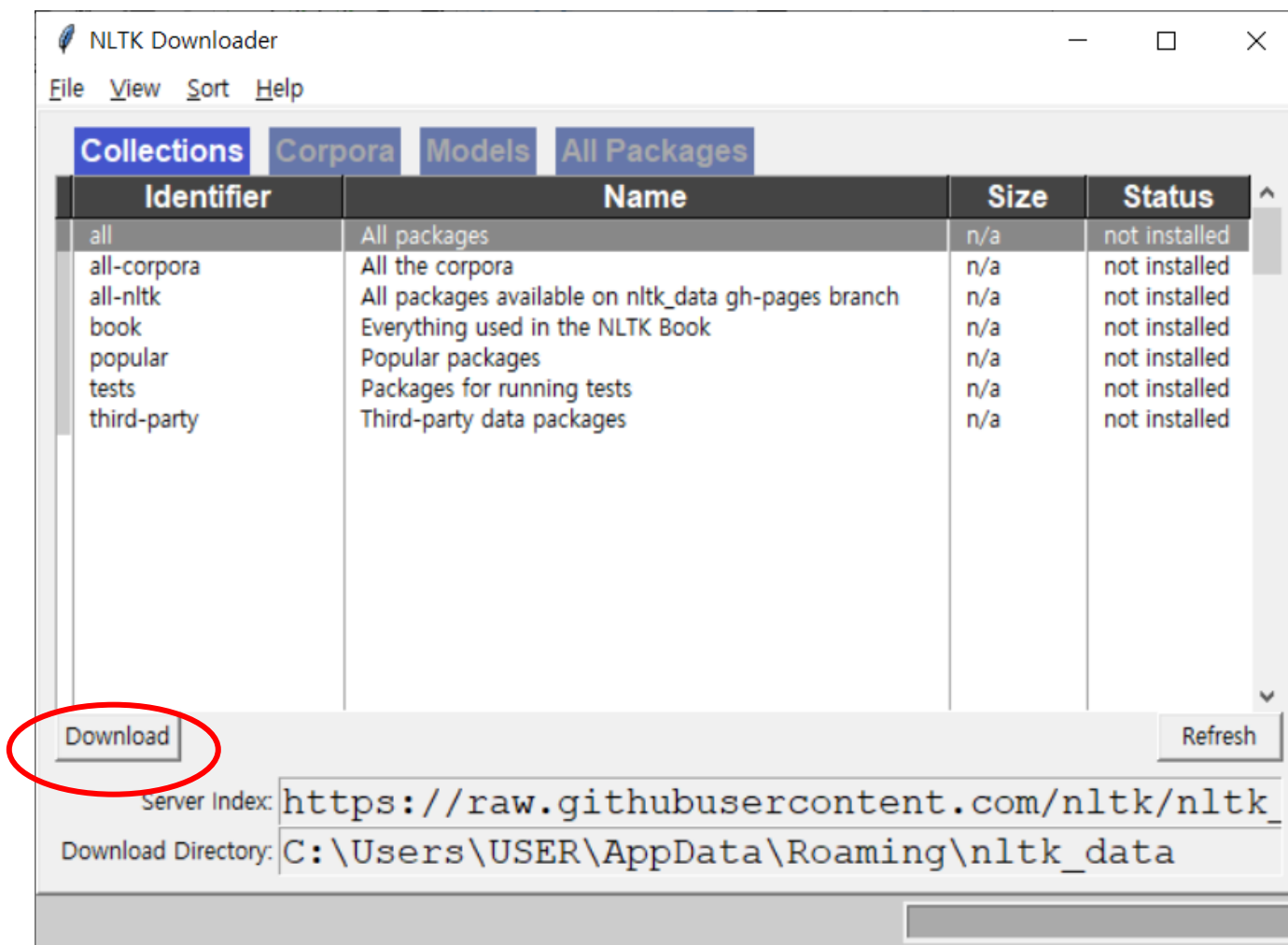
C:\Windows\system32>pip install nltk
Requirement already satisfied: nltk in c:\programdata\anaconda3\lib\site-packages (3.5)
Requirement already satisfied: joblib in c:\programdata\anaconda3\lib\site-packages (from nltk) (0.17.0)
Requirement already satisfied: regex in c:\programdata\anaconda3\lib\site-packages (from nltk) (2020.10.15)
Requirement already satisfied: tqdm in c:\programdata\anaconda3\lib\site-packages (from nltk) (4.50.2)
Requirement already satisfied: click in c:\programdata\anaconda3\lib\site-packages (from nltk) (7.1.2)

C:\Windows\system32>ipython
Python 3.8.5 (default, Sep 3 2020, 21:29:08) [MSC v.1916 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.19.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import nltk

In [2]: nltk.download()
```

nlTK 설치



nlTK 토큰나이징 라이브러리 실행

- Word 토큰나이징

```
from nltk.tokenize import word_tokenize
```

```
sentence = "Natural language processing (NLP) is a subfield of computer science,  
information engineering, and artificial intelligence concerned with the interactions  
between computers and human (natural) languages, in particular how  
to program computers to process and analyze large amounts of natural language data."
```

```
print(word_tokenize(sentence))
```

```
['Natural', 'language', 'processing', '(', 'NLP', ')', 'is', 'a', 'subfield', 'of', 'computer',  
'science', ',', 'information', 'engineering', ',', 'and', 'artificial', 'intelligence', 'concerned',  
'with', 'the', 'interactions', 'between', 'computers', 'and', 'human', '(', 'natural', ')',  
'languages', ',', 'in', 'particular', 'how', 'to', 'program', 'computers', 'to', 'process', 'and',  
'analyze', 'large', 'amounts', 'of', 'natural', 'language', 'data', '.']
```

nlTK 토크나이징 라이브러리 실행

- Sentence 토크나이징

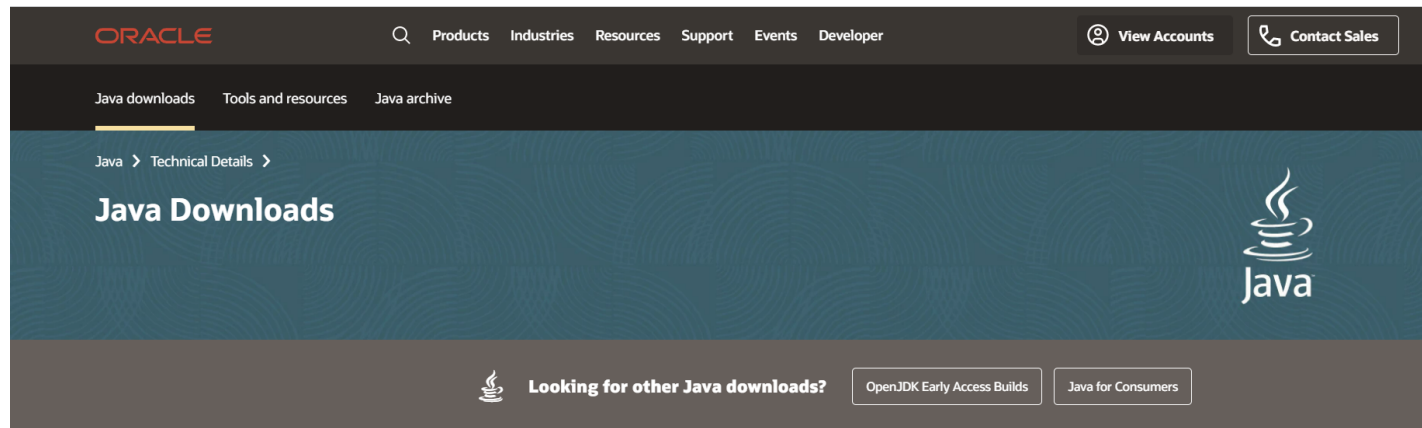
```
from nltk.tokenize import sent_tokenize
```

```
paragraph = "Natural language processing (NLP) is a subfield of computer science,  
information engineering, and artificial intelligence concerned with the interactions  
between computers and human (natural) languages, in particular how to program  
computers to process and analyze large amounts of natural language data.  
Challenges in natural language processing frequently involve speech recognition,  
natural language understanding, and natural language generation."  
print(sent_tokenize(paragraph))
```

```
['Natural language processing (NLP) is a subfield of computer science, information  
engineering, and artificial intelligence concerned with the interactions between  
computers and human (natural) languages, in particular how to program computers  
to process and analyze large amounts of natural language data.', 'Challenges in  
natural language processing frequently involve speech recognition, natural  
language understanding, and natural language generation.']
```

konlpy 설치

- JDK 1.7 이상 설치 필요
 - 설치 주소
 - <https://www.oracle.com/technetwork/java/javase/downloads/index.html>



Java 17 available now

Java 17 LTS is the latest long-term support release for the Java SE platform. JDK 17 binaries are free to use in production and free to redistribute, at no cost, under the [Oracle No-Fee Terms and Conditions License](#).

[Learn about Java SE Subscription](#)

JDK 17 will receive updates under these terms, until at least September 2024.

Java SE Development Kit 17 downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications and components using the Java

konlpy 설치

- 다운로드 후 실행

Java SE Development Kit 17 downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications and components using the Java programming language.

The JDK includes tools for developing and testing programs written in the Java programming language and running on the Java platform.

Documentation Download

Linux macOS **Windows**

Product/file description	File size	Download
x64 Compressed Archive	170.64 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip (sha256 ↗)
x64 Installer	151.99 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe (sha256 ↗)
x64 MSI Installer	150.88 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi (sha256 ↗)

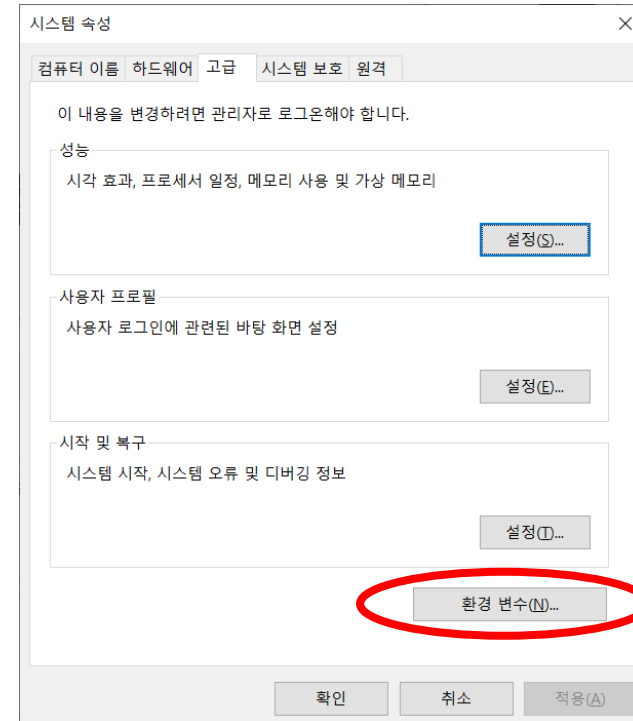
JDK 17 Script-friendly URLs

The URLs listed above will remain the same for all JDK 17 updates to allow their use in scripts.

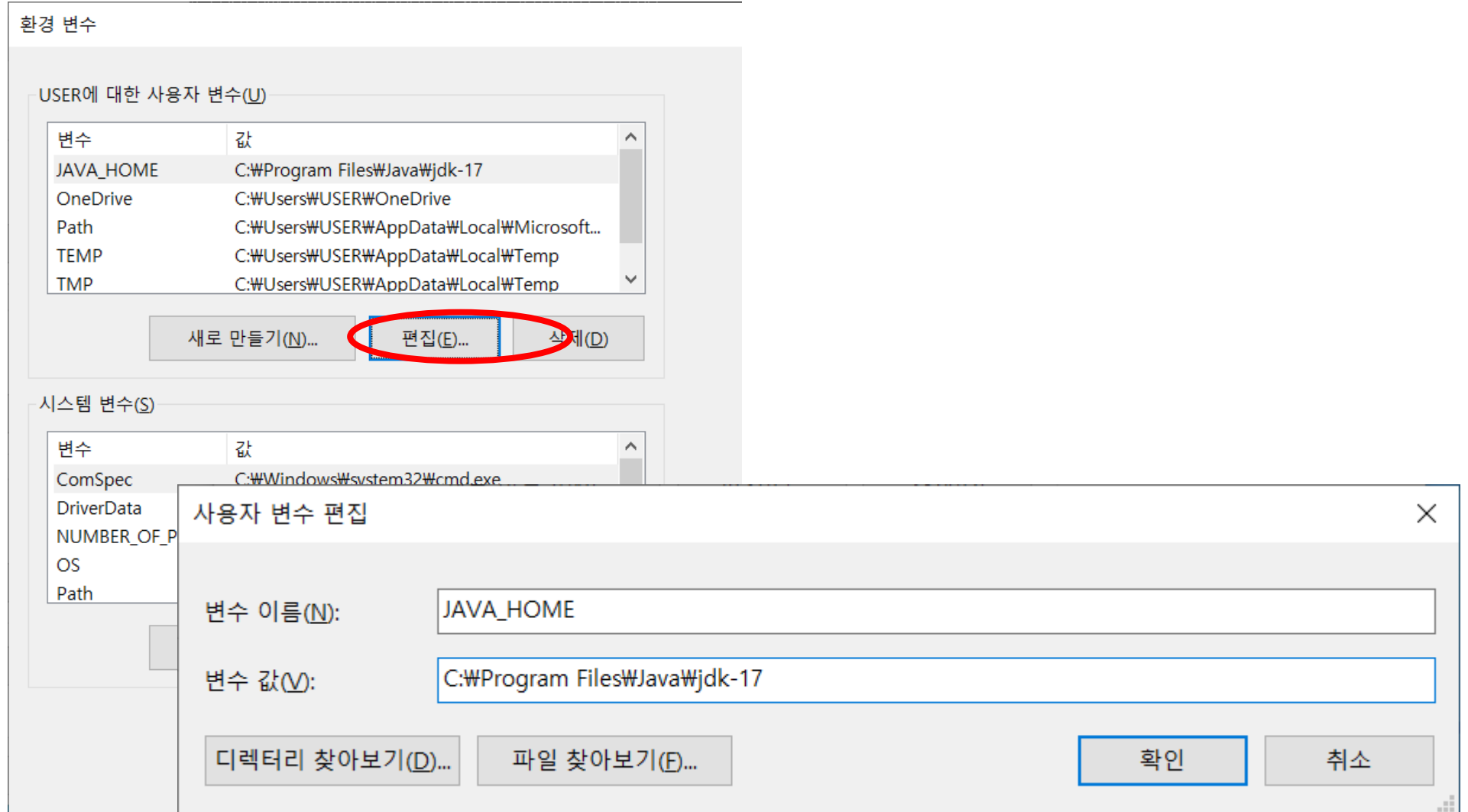
[Learn more about automating the downloads of JDK 17](#)

konlpy 설치

- 환경 변수 설정
 - Java 설치 위치 확인
 - C:\Program Files\Java\jdk-16
 - 윈도우 검색 창에
 - 고급 시스템 설정 보기 입력



konlpy 설치



konlpy 설치

- Jpype 설치

- JAVA와 Python을 연결해주는 역할
- 설치 주소

- <https://www.lfd.uci.edu/~gohlke/pythonlibs/#jpype>

- 파이썬 버전 확인

- 명령 Prompt
 - Python --version
 - Python 3.8.5

JPype: allows full access to Java class libraries.

[JPype1-1.3.0-cp310-cp310-win_amd64.whl](#)

[JPype1-1.3.0-cp310-cp310-win32.whl](#)

[JPype1-1.3.0-cp39-cp39-win_amd64.whl](#)

[JPype1-1.3.0-cp39-cp39-win32.whl](#)

[JPype1-1.3.0-cp38-cp38-win_amd64.whl](#)

[JPype1-1.3.0-cp38-cp38-win32.whl](#)

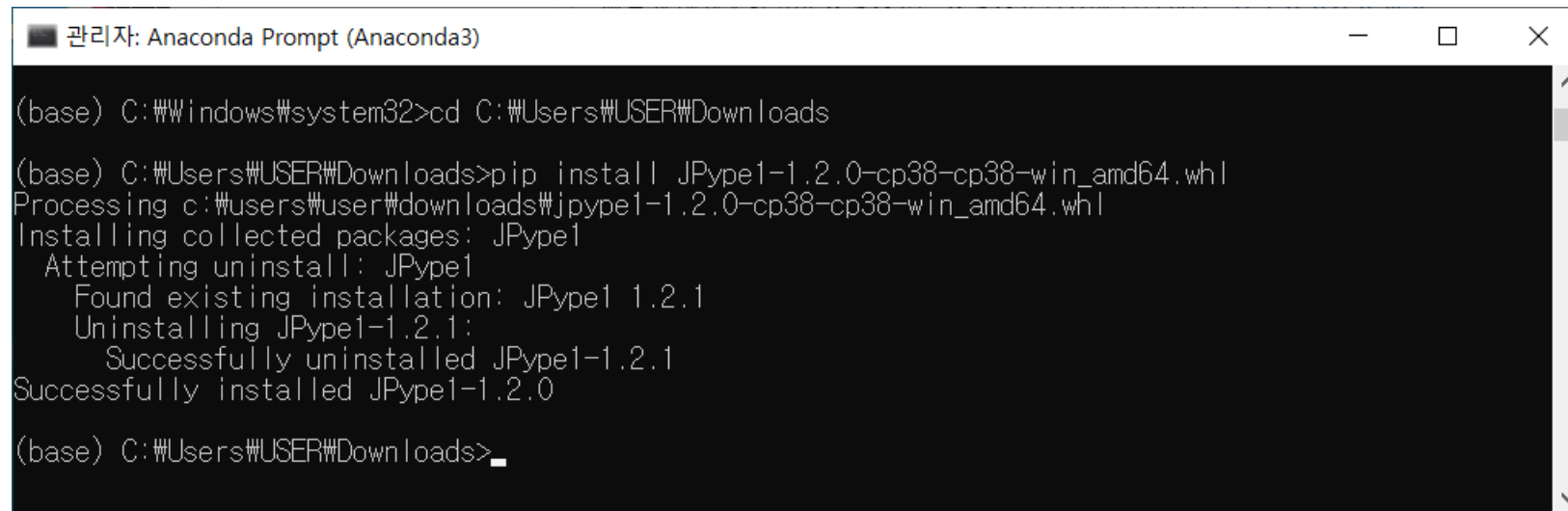
[JPype1-1.3.0-cp37-cp37m-win_amd64.whl](#)

[JPype1-1.3.0-cp37-cp37m-win32.whl](#)

[JPype1-1.2.0-cp36-cp36m-win_amd64.whl](#)

konlpy 설치

- 해당 경로로 이동하여 다음 명령 실행
 - `cd C:\Users\USER\Downloads`
 - `pip install JPype1-1.3.0-cp38-cp38-win_amd64.whl`



```
관리자: Anaconda Prompt (Anaconda3)

(base) C:\Windows\system32>cd C:\Users\USER\Downloads

(base) C:\Users\USER\Downloads>pip install JPype1-1.2.0-cp38-cp38-win_amd64.whl
Processing c:\users\user\downloads\jpype1-1.2.0-cp38-cp38-win_amd64.whl
Installing collected packages: JPype1
  Attempting uninstall: JPype1
    Found existing installation: JPype1 1.2.1
    Uninstalling JPype1-1.2.1:
      Successfully uninstalled JPype1-1.2.1
Successfully installed JPype1-1.2.0

(base) C:\Users\USER\Downloads>
```

konlpy 설치

- KoNLPy
 - 한글 자연어 처리 라이브러리
 - 기존 다른 형태소 분석기 사용 가능
- 명령 Prompt 관리자 권한으로 실행
- `pip install konlpy`

설치 후 오류 발생 시

jvm.py

- 명령어 창에서
 - `pip install -U "jpype1<1.3"`
- jvm.py 파일의 위치 찾아서 이동
 - C:\ProgramData\Anaconda3\Lib\site-packages\konlpy
- 주피터 노트북을 종료 후 다시 시작

```
folder_suffix = [  
    # JAR  
    '{0}',  
    # Java sources  
    '{0}{1}bin',  
    '{0}{1}*',  
    # Hannanum  
    '{0}{1}jhannanum-0.8.4.jar',  
    ...  
    # Twitter (Okt)  
    '{0}{1}snakeyaml-1.12.jar',  
    '{0}{1}scala-library-2.12.3.jar',  
    '{0}{1}open-korean-text-2.1.0.jar',  
    '{0}{1}twitter-text-1.14.7.jar',  
    '{0}{1}*',  
]
```

* 2개 삭제

konlpy 사용하기

- Konlpy 내에 포함된 형태소
 - Hannaum
 - Kkma
 - Komoran
 - Mecab
 - Okt
 - 전 이름 Twitter

konlpy 사용하기

- okt 객체의 함수
 - okt.morphs()
 - 텍스트를 형태소 단위로 나눔
 - okt.nouns()
 - 명사 추출
 - okt.phrases()
 - 어절 추출
 - okt.pos()
 - 품사 태깅

konlpy 사용하기

```
import konlpy
```

```
from konlpy.tag import Okt
```

```
okt = Okt()
```

```
Text = “한글 자연어 처리는 재밌다 이제부터 열심히 해야지 ㅎㅎㅎ”
```

```
print(okt.morphs(text))
```

```
print(okt.morphs(text, stem=True)) # 형태소 단위로 나눈 후 어간을 추출
```

```
['한글', '자연어', '처리', '는', '재밌다', '이제', '부터', '열심히', '해야지', 'ㅎㅎㅎ']
```

```
['한글', '자연어', '처리', '는', '재밌다', '이제', '부터', '열심히', '하다', 'ㅎㅎㅎ']
```

konlpy 사용하기

```
print(oka.nouns(text))
```

['한글', '자연어', '처리', '이제']

```
print(oka.phrases(text))
```

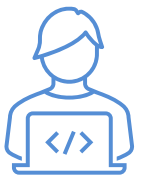
['한글', '한글 자연어', '한글 자연어 처리', '이제', '자연어', '처리']

konlpy 사용하기

```
print(oka.pos(text))  
print(oka.pos(text, join=True)) # 형태소와 품사를 붙여서 리스트화
```

```
[('한글', 'Noun'), ('자연어', 'Noun'), ('처리', 'Noun'), ('는', 'Josa'), ('재밋다',  
'Adjective'), ('이제', 'Noun'), ('부터', 'Josa'), ('열심히', 'Adverb'), ('해야지',  
'Verb'), (' ㅎㅎㅎ ', 'KoreanParticle')]
```

```
['한글/Noun', '자연어/Noun', '처리/Noun', '는/Josa', '재밋다/Adjective',  
'이제/Noun', '부터/Josa', '열심히/Adverb', '해야지/Verb', ' ㅎㅎㅎ  
/KoreanParticle']
```



Lab 7-10

정리

- nltk의 설치와 실행
- konlpy의 설치와 실행

키워드 빈도 분석

목차

- 단어 분포 살펴보기
 - 영어
 - 한글

단어 분포 살펴보기 - 영어

- 파일 열기와 단어 분리

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
```

```
os.chdir('D:\kjh\lecture\LG전자_21\week7_자연어처리')
review_df=pd.read_csv("thinq_e_review.csv")
```

```
#word count
word_counts = review_df['comment'].apply(lambda x:len(x.split(' ')))
```

Series



단어 분포 살펴보기 - 영어

- 단어 수 분포 보기

```
word_counts.head()  
word_counts.describe()
```

```
0    57  
1    59  
2    56  
3    58  
4    57
```

Name: comment, dtype: int64

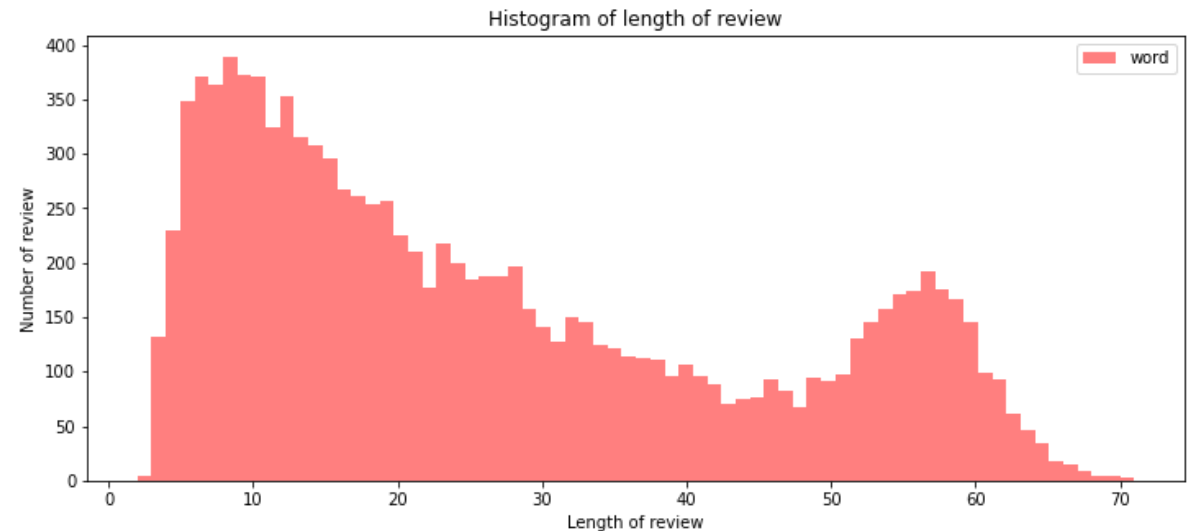
```
count    11280.000000  
mean      26.573670  
std       18.052904  
min        2.000000  
25%       11.000000  
50%       21.000000  
75%       40.000000  
max       71.000000
```

Name: comment, dtype: float64

단어 분포 살펴보기 - 영어

- 히스토그램 그리기

```
plt.figure(figsize=(12, 5))  
plt.hist(word_counts, bins=70, alpha=0.5, color= 'r', label='word')  
plt.legend()  
# 그래프 제목  
plt.title('Histogram of length of review')  
# 그래프 x 축 라벨  
plt.xlabel('Length of review')  
# 그래프 y 축 라벨  
plt.ylabel('Number of review')  
plt.show()
```



단어 분포 살펴보기 - 영어

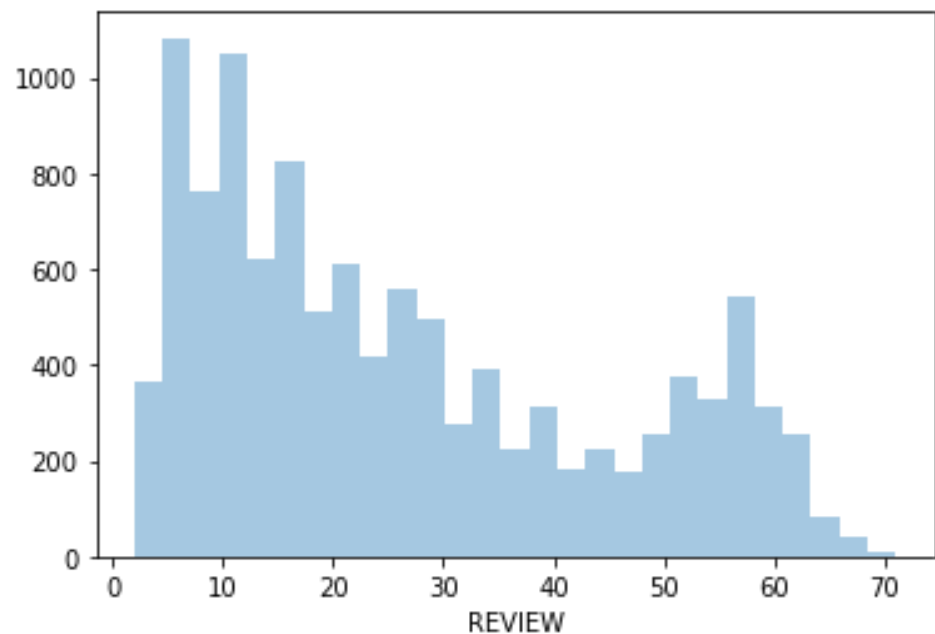
- 히스토그램 그리기

```
sns.distplot(word_counts,kde=False)
```

```
plt.figure(figsize=(12, 5))  
ax = sns.distplot(word_counts,kde=False,bins=70,color='r')  
ax.set_xlabel('Length of review')  
ax.set_ylabel('Number of review')  
ax.set_title('Histogram of length of review')  
ax.legend(labels=['word'])  
plt.show()
```

단어 분포 살펴보기 - 영어

- 히스토그램 그리기

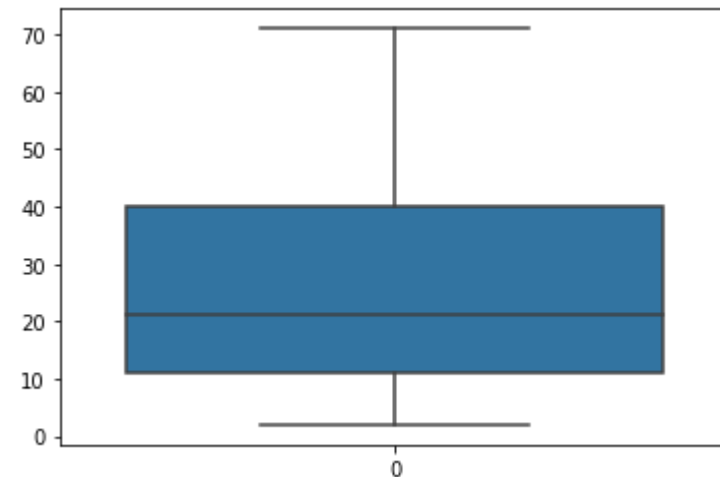
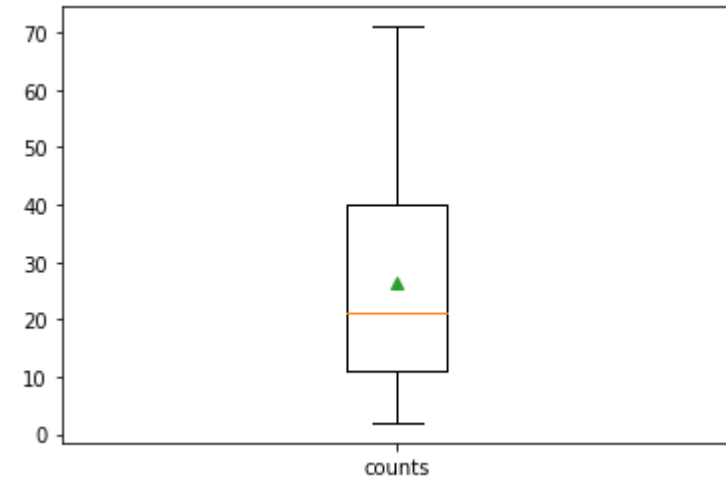


단어 분포 살펴보기 - 영어

- 박스 플랏

```
plt.boxplot(word_counts,  
            labels=['counts'],  
            showmeans=True)
```

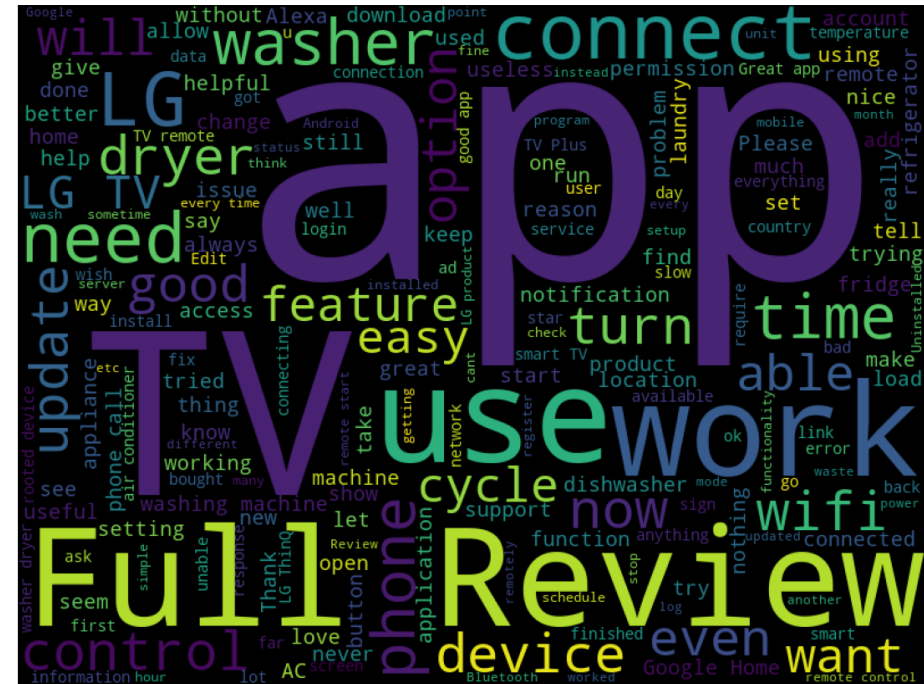
```
sns.boxplot(data=word_counts)
```



단어 분포 살펴보기 - 영어

- 워드 클라우드 그리기

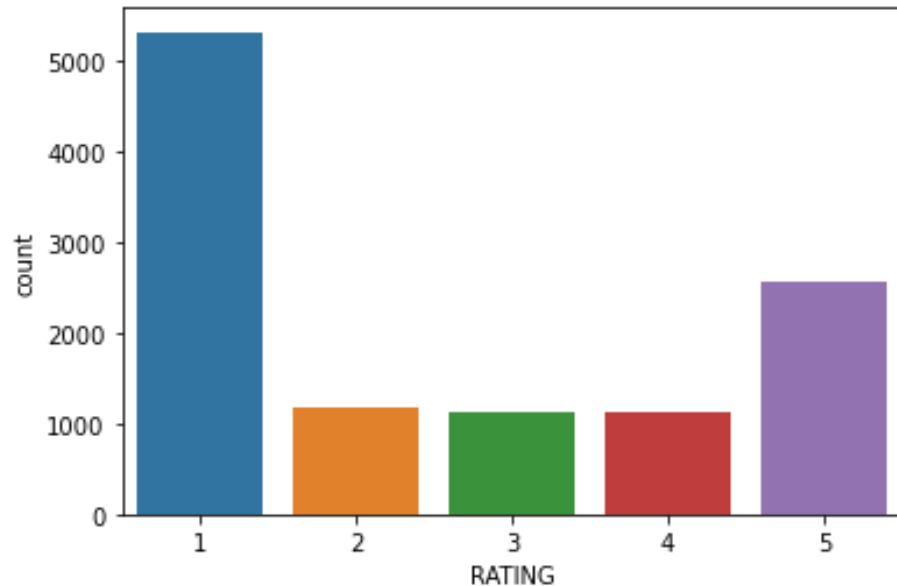
```
from wordcloud import WordCloud
cloud = WordCloud(width=800, height=600).generate(" ".join(review_df['comment']))
plt.figure(figsize=(20, 15))
plt.imshow(cloud)
plt.axis('off')
```



단어 분포 살펴보기 - 영어

- rating 분포

```
print(review_df['star'].value_counts().sort_index())  
sns.countplot(review_df[' star '],order=review_df[' star '].value_counts().sort_index().index)
```



```
1    5318  
2    1164  
3    1127  
4     1112  
5    2559  
Name: star, dtype: int64
```



Lab 7-11

단어 분포 살펴보기 - 한글

- 파일 열기와 단어 분리

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
```

```
os.chdir('D:\kjl\lecture\LG전자_21\week7_자연어처리')
review_df=pd.read_csv("thing_review.csv")
```

```
#word count
word_counts = review_df['comment'].apply(lambda x:len(x.split(' ')))
```

단어 분포 살펴보기 - 한글

- 단어 수 분포 보기

```
word_counts.head()  
word_counts.describe()
```

```
0    44  
1    56  
2    28  
3    16  
4     2
```

Name: REVIEW, dtype: int64

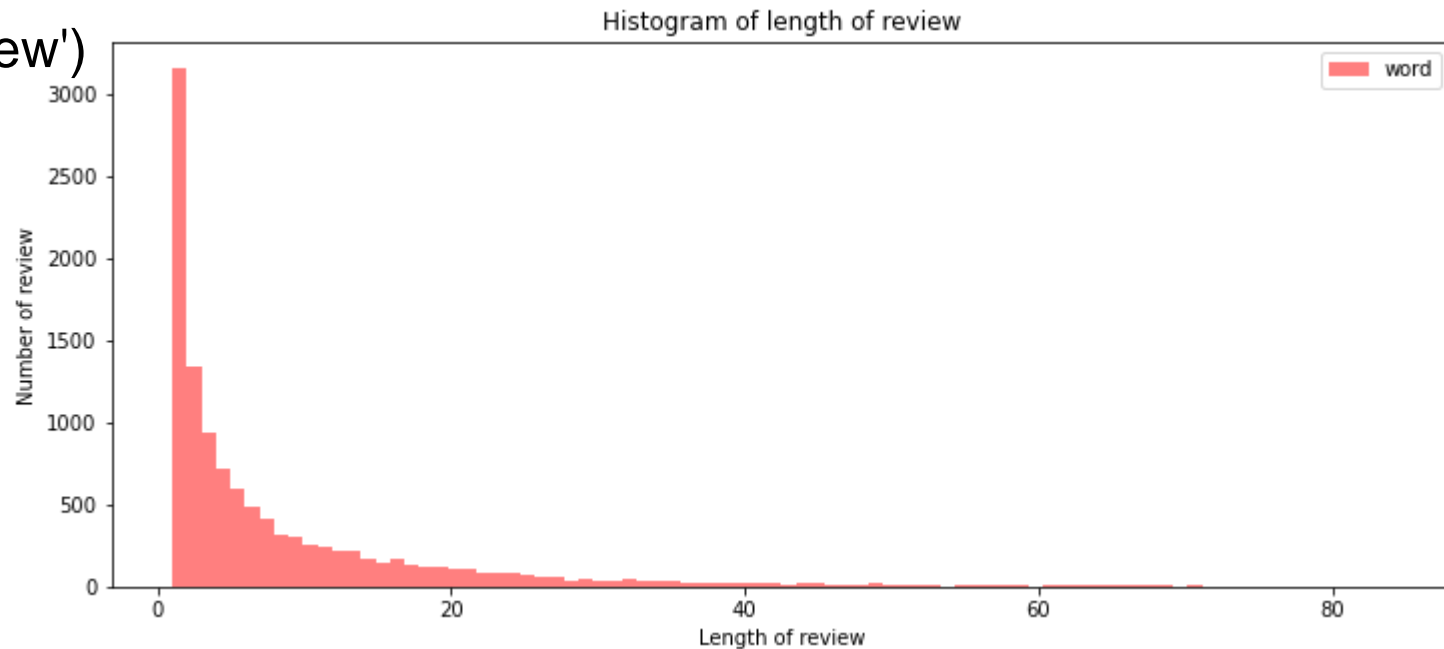
```
count    11379.000000  
mean         8.276826  
std        10.924154  
min         1.000000  
25%         1.000000  
50%         4.000000  
75%        11.000000  
max        84.000000
```

Name: REVIEW, dtype: float64

단어 분포 살펴보기 - 한글

- 히스토그램 그리기

```
plt.figure(figsize=(12, 5))  
plt.hist(word_counts, bins=84, alpha=0.5, color= 'r', label='word')  
plt.legend()  
# 그래프 제목  
plt.title('Histogram of length of review')  
# 그래프 x 축 라벨  
plt.xlabel('Length of review')  
# 그래프 y 축 라벨  
plt.ylabel('Number of review')  
plt.show()
```



단어 분포 살펴보기 - 한글

- 히스토그램 그리기

```
sns.distplot(word_counts,kde=False)
```

```
plt.figure(figsize=(12, 5))
```

```
ax = sns.distplot(word_counts,kde=False,bins=84,color='r')
```

```
ax.set_xlabel('Length of review')
```

```
ax.set_ylabel('Number of review')
```

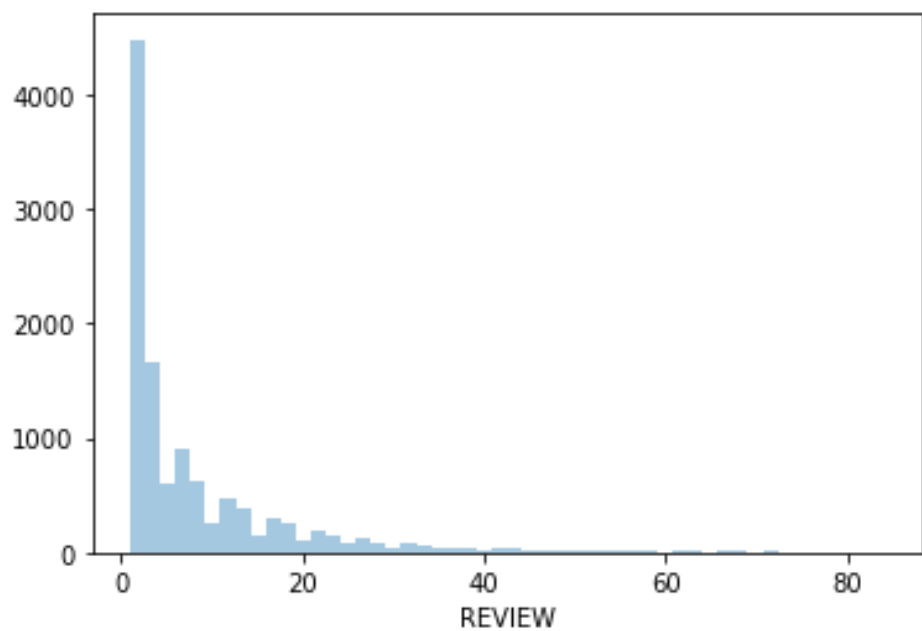
```
ax.set_title('Histogram of length of review')
```

```
ax.legend(labels=['word'])
```

```
plt.show()
```

단어 분포 살펴보기 - 한글

- 히스토그램 그리기

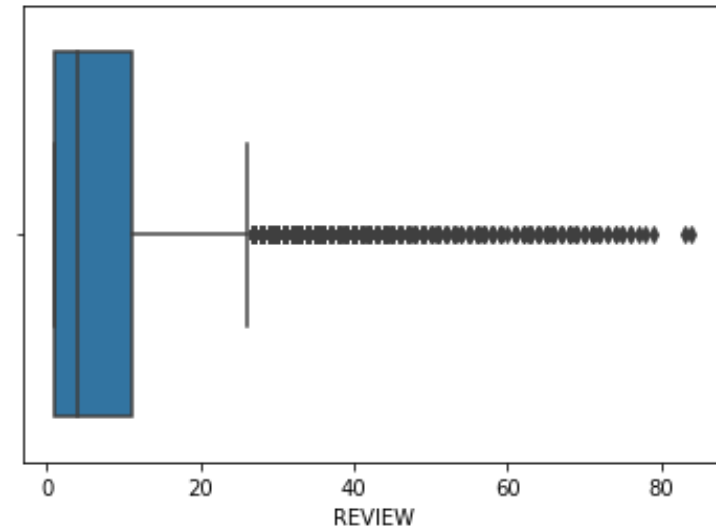
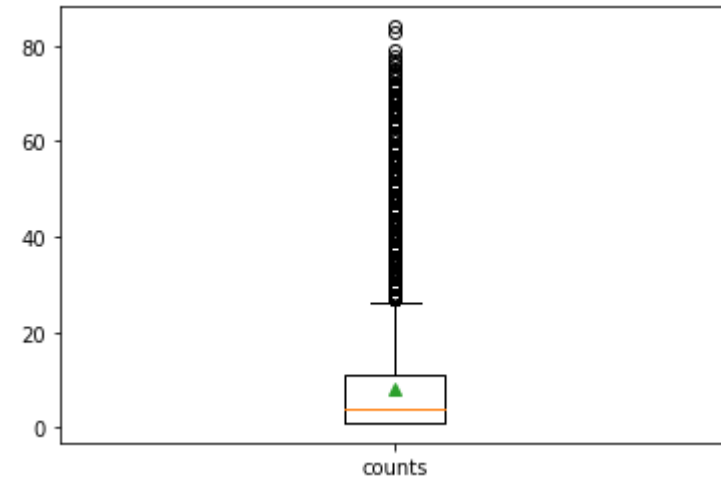


단어 분포 살펴보기 - 한글

- 박스 플랏

```
plt.boxplot(word_counts,  
            labels=['counts'],  
            showmeans=True)
```

```
sns.boxplot(word_counts)
```



단어 분포 살펴보기 - 한글

- 워드 클라우드 그리기

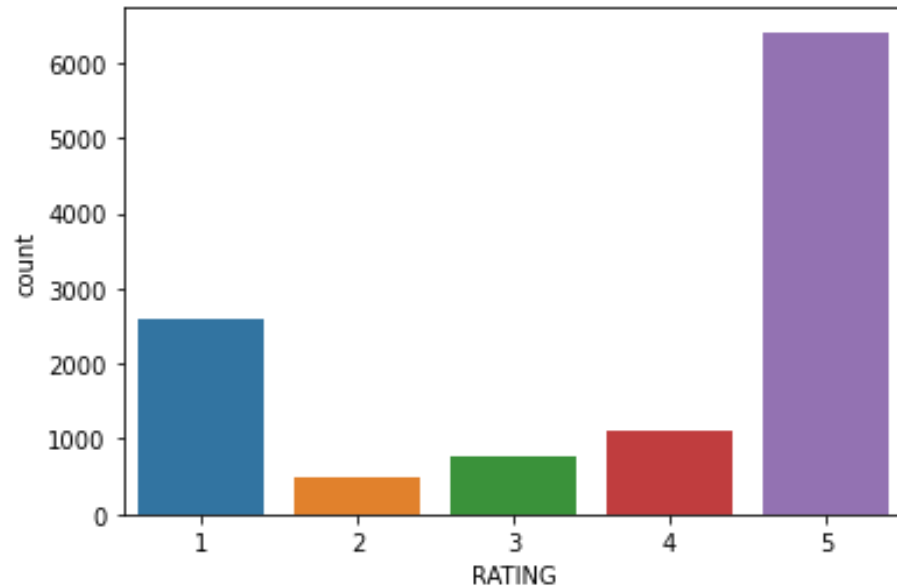
```
from wordcloud import WordCloud
font_path="c:/Windows/Fonts/'
cloud = WordCloud(font_path=font_path+'NanumGothic.ttf',\
                  width=800, height=600).generate(" ".join(review_df['comment']))
plt.figure(figsize=(20, 15))
plt.imshow(cloud)
plt.axis('off')
```



단어 분포 살펴보기 - 한글

- rating 분포

```
review_df['RATING']=review_df['STAR'].apply(lambda x: x[10])  
print(review_df['RATING'].value_counts().sort_index())  
sns.countplot(review_df['RATING'],order=review_df['RATING'].value_counts().sort_index().index)
```



```
1    2583  
2     494  
3     780  
4    1113  
5    6409  
Name: star, dtype: int64
```



Lab 7-12

- 한글 자연어 처리 패키지 설치

설치할 패키지들

- kss
 - 문장 토큰화
- hanspell
 - 맞춤법

kss - 한글 문장 토큰화

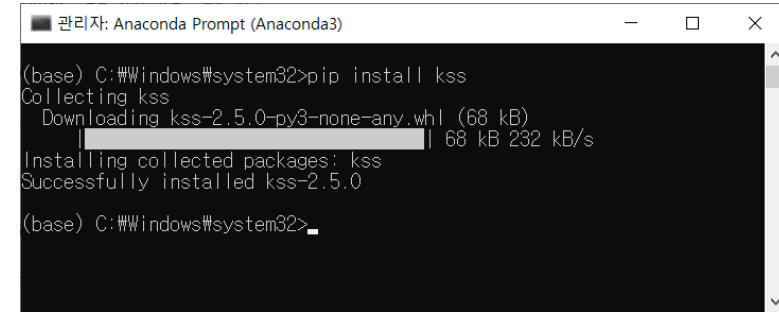
- KSS(Korean Sentence Splitter)
 - pip install kss

```
import kss
```

```
text='딥 러닝 자연어 처리가 재미있기는 합니다. 그런데  
문제는 영어보다 한국어로 할 때 너무 어려워요.  
농담아니에요. 이제 해보면 알걸요?'
```

```
print(kss.split_sentences(text))
```

```
['딥 러닝 자연어 처리가 재미있기는 합니다.', '그런데 문제는  
영어보다 한국어로 할 때 너무 어려워요.', '농담아니에요.',  
'이제 해보면 알걸요?']
```



```
관리자: Anaconda Prompt (Anaconda3)
(base) C:\Windows\system32>pip install kss
Collecting kss
  Downloading kss-2.5.0-py3-none-any.whl (68 kB)
    | 68 kB 232 kB/s
Installing collected packages: kss
Successfully installed kss-2.5.0
(base) C:\Windows\system32>
```

hanspell - 맞춤법 검사

- 맞춤법 검사기 설치
 - `pip install hanspell` 또는
 - `git clone https://github.com/ssut/py-hanspell.git`
 - `cd py-hanspell`
 - `python setup.py install`
 - 이후에 Jupyter 노트북 종료 후 다시 열면 import 가능

hanspell - 맞춤법 검사

```
from hanspell import spell_checker
```

```
text = '딥 러닝 자연어 처리가 재미있기는합니다. 그런데문제는영어보다 한국어로 할때  
너무어려워요. 농담아니예요. 이제 해보면알겠요?'
```

```
print(type(spell_checker.check(text)))  
print(spell_checker.check(text))  
text_spell=spell_checker.check(text).checked  
print(text+"\n")  
print(text_spell)
```

hanspell - 맞춤법 검사

```
<class 'hanspell.response.Checked'>
```

```
Checked(result=True, original='딥 러닝 자연어 처리가 재미있기는합니다. 그런데문제는영어보다  
한국어로 할때 너무어려워요. 농담아니예요. 이제 해보면알걸요?', checked='딥 러닝 자연어 처리가  
재미있기는 합니다. 그런데 문제는 영어보다 한국어로 할 때 너무 어려워요. 농담 아니예요. 이제 해보면  
알걸요?', errors=6, words=OrderedDict([('딥', 0), ('러닝', 0), ('자연어', 0), ('처리가', 0), ('재미있기는', 1),  
('합니다.', 1), ('그런데', 4), ('문제는', 4), ('영어보다', 4), ('한국어로', 0), ('할', 2), ('때', 2), ('너무', 2),  
('어려워요.', 2), ('농담', 4), ('아니예요.', 4), ('이제', 0), ('해보면', 2), ('알걸요?', 2)]),  
time=0.26964306831359863)
```

딥 러닝 자연어 처리가 재미있기는합니다. 그런데문제는영어보다 한국어로 할때 너무어려워요.
농담아니예요. 이제 해보면알걸요?

딥 러닝 자연어 처리가 재미있기는 합니다. 그런데 문제는 영어보다 한국어로 할 때 너무 어려워요. 농담
아니예요. 이제 해보면 알걸요?



Lab 7-13

- 토큰화와 형태소 분석

토큰화

- 코퍼스에서 토큰(token) 단위로 나누는 작업
- 문장 토큰화와 단어 토큰화
- 형태소 토큰화

단어 토큰화

```
from nltk.tokenize import word_tokenize
print(word_tokenize("Don't be fooled by the dark sounding name, Mr. Jones Orphanage is as cheery as cheery goes for a pastry shop."))
```

```
['Do', "n't", 'be', 'fooled', 'by', 'the', 'dark', 'sounding', 'name', ',', 'Mr.', 'Jones', "s", 'Orphanage', 'is', 'as', 'cheery', 'as', 'cheery', 'goes', 'for', 'a', 'pastry', 'shop', '.']
```

```
from nltk.tokenize import WordPunctTokenizer
print(WordPunctTokenizer().tokenize("Don't be fooled by the dark sounding name, Mr. Jones Orphanage is as cheery as cheery goes for a pastry shop."))
```

```
['Don', "'", 't', 'be', 'fooled', 'by', 'the', 'dark', 'sounding', 'name', ',', 'Mr', '.', 'Jones', "'", 's', 'Orphanage', 'is', 'as', 'cheery', 'as', 'cheery', 'goes', 'for', 'a', 'pastry', 'shop', '.']
```

문장 토큰화

- Sentence Tokenization
- Sentence Segmentation
- 마침표(.), ?, !
- 마침표가 문장 구분자로 적절하지 않은 경우
 - EX1) IP 192.168.56.31 서버에 들어가서 로그 파일 저장해서 ukairia777@gmail.com로 결과 좀 보내줘. 그러고나서 점심 먹으러 가자.
 - EX2) Since I'm actively looking for Ph.D. students, I get the same question a dozen times every year.

문장 토큰화

```
from nltk.tokenize import sent_tokenize  
text="His barber kept his word. But keeping such a huge secret to  
himself was driving him crazy. Finally, the barber went up a mountain  
and almost to the edge of a cliff. He dug a hole in the midst of some  
reeds. He looked about, to make sure no one was near."  
print(sent_tokenize(text))
```

```
['His barber kept his word.', 'But keeping such a huge secret to himself  
was driving him crazy.', 'Finally, the barber went up a mountain and  
almost to the edge of a cliff.', 'He dug a hole in the midst of some reeds.',  
'He looked about, to make sure no one was near.']
```

문장 토큰화

```
from nltk.tokenize import sent_tokenize  
text="I am actively looking for Ph.D. students. and you are a Ph.D student."  
print(sent_tokenize(text))
```

['I am actively looking for Ph.D. students.', 'and you are a Ph.D student.']

형태소 분석

- Morphological Analysis
- 형태소
 - 더 이상 분해될 수 없는 최소한의 의미를 갖는 단위
 - 컴퓨터를 -> 컴퓨터 + 를
- 품사 태깅(Part of Speech Tagging)
 - 문서에 품사 정보를 부착하는 것

나/NP + 는/JX + 학교/NNG + 예/JKB + 가/VX + 았/EP + 다/EF + ./SF

영어 품사 태깅

Tag	Description	설명	Example
CC	coordinating conjunction		
CD	cardinal digit		
DT	determiner		
EX	existential there		'there' is.., 'there' exists..
FW	foreign word		
IN	preposition/subordinateing conjunction		
JJ	adjective	형용사	'big'
JJR	adjective, comparative	형용사, 비교급	'bigger'
JJS	adjective, superlative	형용사, 최상급	'biggest'
LS	list marker		'1)'
MD	modal		'could', 'will'
NN	noun, singular	명사, 단수형	'desk'
NNS	noun, plural	명사, 복수형	'desks'
NNP	proper noun, singular	고유명사, 단수형	'Harrison'
NNPS	proper noun, plural	고유명사, 복수형	'Americans'
PDT	predeterminer		'all the kids'
POS	possessive ending		'parent's '

영어 품사 태깅

Tag	Description	설명	Example
PRP	personal pronoun	인칭 대명사	'I', 'he', 'she'
PRP\$	possessive pronoun	소유 대명사	'my', 'his', 'hers'
RB	adverb	형용사	'very', 'silently'
RBR	adverb, comparative	형용사, 비교급	'better'
RBS	adverb, superlative	형용사, 최상급	'best'
RP	particle		'give up'
TO	to		go 'to' the store
UH	interjection		'errrrrm'
VB	verb, base form	동사, 원형	'take'
VBD	verb, past tense	동사, 과거형	'took'
VBG	verb, gerund/present participle	동사, 현재분사	'taking'
VBN	verb, past participle	동사, 과거분사	'taken'
VBP	verb, sing. Present, non-3d		'take'
VBZ	verb, 3rd person sing. Present		'takes'
WDT	wh-determiner		'which'
WP	wh-pronoun		'who', 'what'
WP\$	possessive		'wh-pronoun', 'whose'
WRB	wh-adverb		'where', 'when'

영어 품사 태깅

```
from nltk.tokenize import word_tokenize
text="I am actively looking for Ph.D. students. and you are a Ph.D. student."
print(word_tokenize(text))
```

```
['I', 'am', 'actively', 'looking', 'for', 'Ph.D.', 'students', '.', 'and', 'you', 'are', 'a', 'Ph.D.', 'student', '.']
```

```
from nltk.tag import pos_tag
x=word_tokenize(text)
pos_tag(x)
```

Personal pronoun(인칭대명사)
verb, sing. Present, non-3d
Adverb(형용사)
...

```
[('I', 'PRP'),  
 ('am', 'VBP'),  
 ('actively', 'RB'),  
 ('looking', 'VBG'),  
 ('for', 'IN'),  
 ('Ph.D.', 'NNP'),  
 ('students', 'NNS'),  
 ('.', '.'),  
 ('and', 'CC'),  
 ('you', 'PRP'),  
 ('are', 'VBP'),  
 ('a', 'DT'),  
 ('Ph.D.', 'NNP'),  
 ('student', 'NN'),  
 ('.', '.')]
```

한국어 토큰화

- 한국어 토큰화의 어려움
 - 띄어쓰기만으로 토큰화가 어려움
 - 이유
 - 교착어
 - 조사, 어미 등이 붙어있음
 - ‘그가’, ‘그에게’, ‘그와’ -> 조사 분리해줘야 함
 - 어절 토큰화보다 형태소 토큰화가 더 적절
 - 띄어쓰기가 잘 지켜지지 않음
 - 띄어쓰기를 하지 않아도 글을 이해할 수 있음

KoNLPy의 형태소 분석기

- 사용가능 형태소 분석기
 - Okt(Open Korea Text)
 - Mecab
 - Komoran
 - Hannanum
 - Kkma

KoNLPy의 형태소 분석기

- Okt 형태소 분석기 사용하기

```
from konlpy.tag import Okt
okt=Okt()
print(okt.morphs("열심히 코딩한 당신, 연휴에는 여행을 가봐요")) # 형태소 추출
```

```
['열심히', '코딩', '한', '당신', ',', '연휴', '에는', '여행', '을', '가봐요']
```

```
print(okt.pos("열심히 코딩한 당신, 연휴에는 여행을 가봐요")) # 품사 태깅
```

```
[('열심히', 'Adverb'), ('코딩', 'Noun'), ('한', 'Josa'), ('당신', 'Noun'), (',', 'Punctuation'), ('연휴', 'Noun'), ('에는', 'Josa'), ('여행', 'Noun'), ('을', 'Josa'), ('가봐요', 'Verb')]
```

```
print(okt.nouns("열심히 코딩한 당신, 연휴에는 여행을 가봐요")) # 명사 추출
```

```
['코딩', '당신', '연휴', '여행']
```

KoNLPy의 형태소 분석기

- Kkma

```
from konlpy.tag import Kkma  
kkma=Kkma()
```

```
print(kkma.morphs("열심히 코딩한 당신, 연휴에는 여행을 가봐요"))
```

```
['열심히', '코딩', '하', 'ㄴ', '당신', ',', '연휴', '에', '는', '여행', '을', '가보', '아요']
```

```
print(kkma.pos("열심히 코딩한 당신, 연휴에는 여행을 가봐요"))
```

```
[('열심히', 'MAG'), ('코딩', 'NNG'), ('하', 'XSV'), ('ㄴ', 'ETD'), ('당신', 'NP'), (',',  
'SP'), ('연휴', 'NNG'), ('에', 'JKM'), ('는', 'JX'), ('여행', 'NNG'), ('을', 'JKO'),  
(가보', 'VV'), ('아요', 'EFN')]
```

```
print(kkma.nouns("열심히 코딩한 당신, 연휴에는 여행을 가봐요"))
```

```
['코딩', '당신', '연휴', '여행']
```

KoNLPy의 형태소 분석기

- Hannanum

```
from konlpy.tag import Hannanum
Hannanum=Hannanum()
print(Hannanum.morphs("열심히 코딩한 당신, 연휴에는 여행을 가봐요"))
print(Hannanum.pos("열심히 코딩한 당신, 연휴에는 여행을 가봐요"))
print(Hannanum.nouns("열심히 코딩한 당신, 연휴에는 여행을 가봐요"))
```



Lab 7-14