

자연어 데이터 전처리

DEEP LEARNING AND NATURAL LANGUAGE PROCESSING

02

APPLICATION

ARTIFICIAL INTELLIGENCE

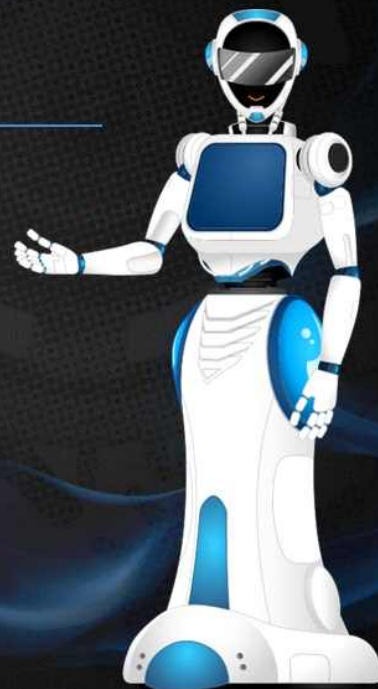
Notice

이 교육과정은 교육부 '성인학습자 역량 강화 교육콘텐츠 개발' 사업의 일환으로써
교육부로부터 예산을 지원 받아 고려사이버대학교가 개발하여 운영하고 있습니다.
제공하는 강좌 및 학습에 따르는 모든 산출물의 저작권은 교육부, 한국교육학술정보원,
한국원격대학협의회와 고려사이버대학교가 공동 소유하고 있습니다.

THINKING

생각해보기

✓ 자연어 데이터 전처리는 어떻게 할까요?



학습목표

Artificial Intelligence (AI) refers
to the simulation of human

GOALS

Artificial Intelligence has
risen to the attention of
human intelligence in various
fields and industries. It has
not only helped and solved their
problems.

The technology can be applied
to various fields such as
education, medicine, and
business, and it is expected
to be widely used in the future.

- 1 자연어 데이터 묶음 **코퍼스**에 대해 설명할 수 있다.
- 2 **NLTK**를 활용해 코퍼스에 대한 분석을 할 수 있다.
- 3 **텍스트 수준 텍스트 전처리**를 하는 방법을 설명할 수 있다.
- 4 **단어 수준 텍스트 전처리**를 하는 방법을 설명할 수 있다.
- 5 **정규 표현식**에 대해 이해하고 설명할 수 있다.



- 1 자연어 데이터
- 2 텍스트 데이터 전처리
- 3 실습

"The data used here for training
is only machine-generated text.
While machine-generated text
could be used for training and
evaluation purposes,
it is not recommended for
generalization to human
text. The data used here for
training is only machine-generated
text and should not be used
for generalization to human
text."

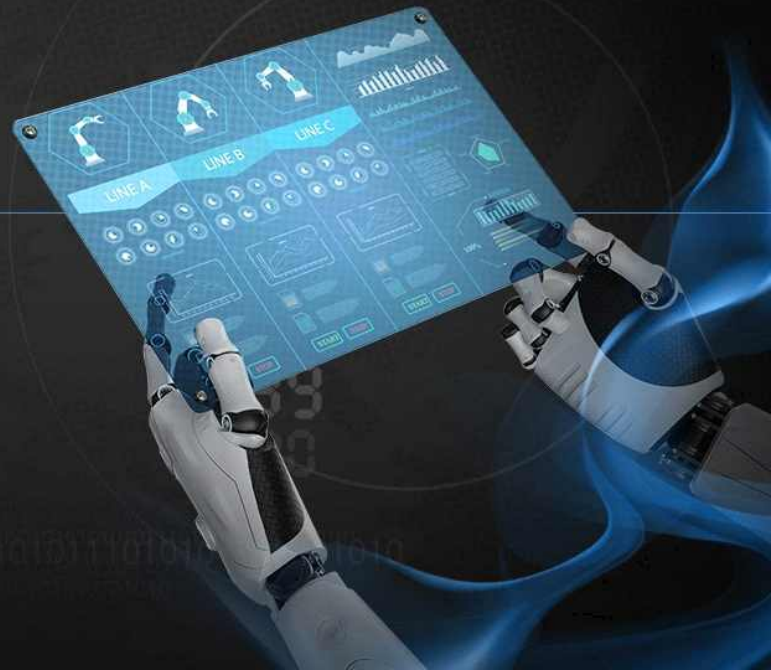
CONTENTS

학습내용

Artificial intelligence (AI) refers
to the simulation of human



자연어 데이터



01 코퍼스란?

코퍼스(corpus), 말뭉치

- 자연언어 연구를 위해 특정한 목적을 가지고 언어의 표본을 추출한 집합
- 컴퓨터에 저장된, 쓰이거나 말해진 자연어 자료 모음
- 둘 이상의 코퍼스가 존재하면 **코포라(corpora)**라고 부름

코퍼스 예시

'Austin/np-hl ,/, -hl Texas/np-hl \n--/-- Committee/nn approval/nn of/in Gov./nn-tl P
rice/np Daniel's/np\$ ``/`` abandoned/vbn property/nn ''/' act/nn seemed/vbd certai
n/jj Thursday/nr despite/in the/at adamant/jj protests/nns of/in Texas/np bankers/nn
s ./.\n\n\n\tDaniel/np personally/rb led/vbd the/at fight/nn for/in the/at measure/n
n ,/, which/wdt he/pps had/hvd watered/vbn down/rp considerably/rb since/in its/pp\$
rejection/nn by/in two/cd previous/jj Legislatures/nns-tl ,/, in/in a/at public/jj h
earing/nn before/in the/at House/nn-tl Committee/nn-tl on/in-tl Revenue/nn-tl and/cc
-tl Taxation/nn-tl ./.\n\n\n\tUnder/in committee/nn rules/nns ,/, it/pps went/vbd au
tomatically/rb to/in a/at subcommittee/nn for/in one/cd week/nn ./.\n\nBut/cc question
s/nns with/in which/wdt committee/nn members/nns taunted/vbd bankers/nns appearing/v
bg as/cs witnesses/nns left/vbd little/ap doubt/nn that/cs they/ppss will/md recomme
nd/vb passage/nn of/in it/ppo ./.\n\n\n\tDaniel/np termed/vbd ``/`` extremely/...'

코퍼스의 도움으로 자연어에 대한 빈도 분포,
단어의 동시 발생 등과 같은 통계적인 분석이 가능하다.



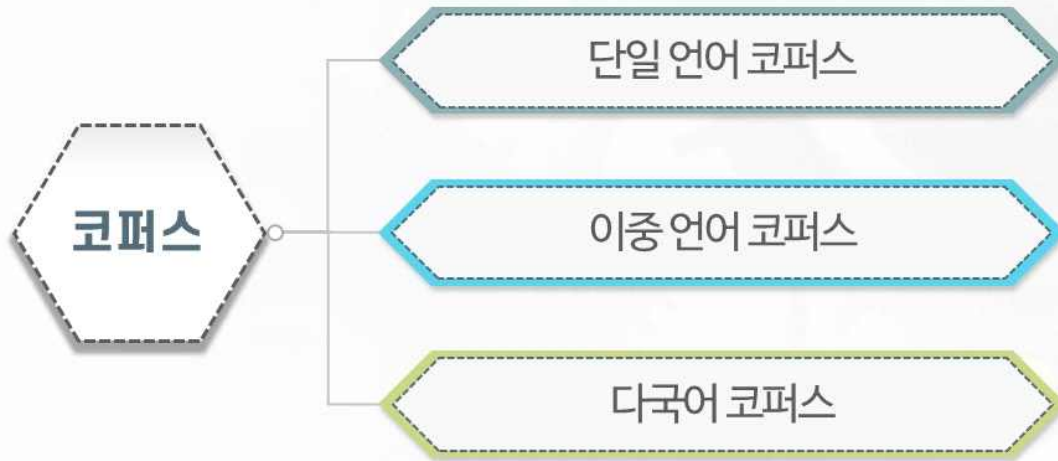
다양한 자연어처리 과정에서 자연어 데이터에 대한
언어 규칙을 정의하고 이를 검증할 수 있다.



규칙 기반 시스템의 도움으로 언어 사용에 따라
각 언어에 대한 언어 규칙을 정의할 수 있다.



구성된 언어를 기준으로



코퍼스 수집 방법

- 공개된 오픈소스 코퍼스 사용
- 다양한 웹사이트에서 크롤링을 통한 자연어 데이터 수집

💡 주의 : 저작권 문제, 트래픽 문제 등을 확인하며 크롤링 해야함

1

음성데이터에서의 코퍼스 분석

- 각 데이터의 음성 이해에 대한 분석 및 대화 분석이 필요

2

텍스트데이터에서의 코퍼스 분석

- 일반적으로 코퍼스에 단어가 몇 개 나오는지, 코퍼스 내에 있는 특정 단어의 빈도수가 얼마인지를 분석
- 코퍼스에 노이즈가 있으면 노이즈를 제거

NLTK(Natural Language Toolkit)

- Python 프로그래밍 언어로 작성된 영어의 자연어 처리를 위한 라이브러리 및 프로그램 모음
- 다양한 기능을 가지고 있으며 교육용 뿐만 아니라 실무 및 연구에서도 많이 사용
- 50종류 이상의 코퍼스와 어휘 자료를 가지고 있는 오픈소스 라이브러리

코포라 타입	설명	예시 코퍼스
아이솔레이트 코퍼스 (isolate corpus)	텍스트 또는 자연어의 모음	gutenberg 코퍼스 webtext 코퍼스
카테고리화 코퍼스 (Categorized corpus)	뉴스, 취미, 유머 등 다양한 타입의 부류로 그룹화된 텍스트 모음	brown 코퍼스
오버래핑 코퍼스 (Overlapping corpus)	분류된 텍스트 모음이지만 카테고리가 서로 겹침	reuters 코퍼스
템포럴 코퍼스 (Temporal corpus)	일정 기간동안 자연어를 사용한 데이터 모음	Inaugural address 코퍼스

NLTK의 “brown” 코퍼스와 “gutenberg” 코퍼스 가져오기

```
In [1]: 1 import nltk
        2 from nltk.corpus import brown as cb
        3 from nltk.corpus import gutenberg as cg
```

```
In [2]: 1 nltk.download('brown')
        2 nltk.download('gutenberg')
```

```
[nltk_data]
[nltk_data]
[nltk_data]
[nltk_data]
[nltk_data]
[nltk_data]
```

```
Out[2]: True
```


.fileids()를 통해 brown 코퍼스 파일 확인하기

```
In [3]: 1 cb.fileids()
```

```
Out[3]: ['ca01',  
'ca02',  
'ca03',  
'ca04',  
'ca05',  
'ca06',  
'ca07',  
'ca08',  
'ca09',  
'ca10',  
'ca11',  
'ca12',  
'ca13',  
'ca14',  
'ca15',  
'ca16',  
'ca17',  
'ca18',  
'ca19',  
'ca20']
```

.fileids()를 통해 brown 코퍼스 파일 확인하기

```
In [4]: 1 cg.fileids()
```

```
Out[4]: ['austen-emma.txt',  
'austen-persuasion.txt',  
'austen-sense.txt',  
'bible-kjv.txt',  
'blake-poems.txt',  
'bryant-stories.txt',  
'burgess-busterbrown.txt',  
'carroll-alice.txt',  
'chesterton-ball.txt',  
'chesterton-brown.txt',  
'chesterton-thursday.txt',  
'edgeworth-parents.txt',  
'melville-moby_dick.txt',  
'milton-paradise.txt',  
'shakespeare-caesar.txt',  
'shakespeare-hamlet.txt',  
'shakespeare-macbeth.txt',  
'whitman-leaves.txt']
```

.categories()를 통해 brown 코퍼스의 카테고리 나열하기

```
In [5]: 1 cb.categories()
Out[5]: ['adventure',
        'belles_lettres',
        'editorial',
        'fiction',
        'government',
        'hobbies',
        'humor',
        'learned',
        'lore',
        'mystery',
        'news',
        'religion',
        'reviews',
        'romance',
        'science_fiction']
```

.words()를 통해 brown 코퍼스의 전체 단어 나열하기

```
In [6]: 1 cb.words()[:20]
Out[6]: ['The',
        'Fulton',
        'County',
        'Grand',
        'Jury',
        'said',
        'Friday',
        'an',
        'investigation',
        'of',
        'Atlanta's',
        'recent',
        'primary',
        'election',
        'produced',
        '...',
        'no',
        'evidence',
        '...',
        'that']
```

.words()의 categories 인자로 코퍼스 내 지정한 category의 단어 나열하기

```
In [7]: 1 cb.words(categories='news')[10:30]
```

```
Out[7]: ["Atlanta's",  
        'recent',  
        'primary',  
        'election',  
        'produced',  
        '\n',  
        'no',  
        'evidence',  
        '\n',  
        'that',  
        'any',  
        'irregularities',  
        'took',  
        'place',  
        '\n',  
        'The',  
        'jury',  
        'further',  
        'said',  
        'in']
```

.Text()를 통해 특정 문서만 불러오기

```
In [8]: 1 raw_text = nltk.Text(cb.words('ca01'))  
        2 raw_text
```

```
Out[8]: <Text: The Fulton County Grand Jury said Friday an...>
```

.concordance()를 통해 특정 단어가 들어간 문장 가져오기

```
In [9]: 1 raw_text.concordance("The")
```

Displaying 25 of 155 matches:

The Fulton County Grand Jury said Friday that any irregularities took place . The jury further said in term-end present er said in term-end presentments that the City Executive Committee , which had mittee , which had over-all charge of the election , `` deserves the praise and charge of the election , `` deserves the praise and thanks of the City of Atlanta `` deserves the praise and thanks of the City of Atlanta `` for the manner in thanks of the City of Atlanta `` for the manner in which the election was cond of Atlanta `` for the manner in which the election was conducted . The Septembe in which the election was conducted . The September-October term jury had been s of possible `` irregularities `` in the hard-fought primary which was won by ful of such reports was received `` , the jury said , `` considering the widesp d `` , the jury said , `` considering the widespread interest in the election , onsidering the widespread interest in the election , the number of voters and t widespread interest in the election , the number of voters and the size of this e election , the number of voters and the size of this city `` . The jury said voters and the size of this city `` . The jury said it did find that many of Ge ave these laws studied and revised to the end of modernizing and improving them f modernizing and improving them `` . The grand jury commented on a number of o a number of other topics among them the Atlanta and Fulton County purchasing

.raw_content()를 통해 가공 전 데이터 가져오기

```
In [10]: 1 raw_content = cb.raw('ca02')
2 raw_content
```

```
Out[10]: "Austin/np-hl ,/-hl Texas/np-hl \n-- Committee/nn approval/nn of/in Gov./nn-tl Price/np Daniel's/np$ ``/`` abando
ned/vbn property/nn ''/'' act/nn seemed/vbd certain/jj Thursday/nr despite/in the/at adamant/jj protests/nns of/in Te
xas/np bankers/nns ./.\n\n\nDaniel/np personally/rb led/vbd the/at fight/nn for/in the/at measure/nn ,/, which/wdt
he/pps had/hvd watered/vbn down/rp considerably/rb since/in its/pp$ rejection/nn by/in two/cd previous/jj Legislatu
s/nns-tl ,/, in/in a/at public/jj hearing/nn before/in the/at House/nn-tl Committee/nn-tl on/in-tl Revenue/nn-tl and/
cc-tl Taxation/nn-tl ./.\n\n\nUnder/in committee/nn rules/nns ,/, it/pps went/vbd automatically/rb to/in a/at subco
mmittee/nn for/in one/cd week/nn ./.\n\nBut/cc questions/nns with/in which/wdt committee/nn members/nns taunted/vbd ban
kers/nns appearing/vbg as/cs witnesses/nns left/vbd little/ap doubt/nn that/cs they/ppss will/md recommend/vb passag
e/nn of/in it/ppo ./.\n\n\nDaniel/np termed/vbd ``/`` extremely/rb conservative/jj ''/'' his/pp$ estimate/nn that/c
s it/pps would/md produce/vb 17/cd million/cd dollars/nns to/to help/vb erase/vb an/at anticipated/vbn deficit/nn of/
in 63/cd million/cd dollars/nns at/in the/at end/nn of/in the/at current/jj fiscal/jj year/nn next/ap Aug./np 31/cd
./.\n\n\nHe/pps told/vbd the/at committee/nn the/at measure/nn would/md merely/rb provide/vb means/nns of/in enfor
cing/vbg the/at escheat/nn law/nn which/wdt has/hvz been/ben on/in the/at books/nns ``/`` since/in Texas/np was/bedz
a/at republic/nn ''/'' ./.\n\nIt/pps permits/vbz the/at state/nn to/to take/vb over/rp bank/nn accounts/nns ,/, stocks/
nns and/cc other/ap personal/jj property/nn of/in persons/nns missing/vbg for/in seven/cd years/nns or/cc more/ap
./.\n\n\nThe/at bill/nn ,/, which/wdt Daniel/np said/vbd he/pps drafted/vbd personally/rb ,/, would/md force/vb ban
ks/nns ,/, insurance/nn firms/nns ,/, pipeline/nn companies/nns and/cc other/ap corporations/nns to/to report/vb suc
h/jj property/nn to/in the/at state/nn treasurer/nn ./.\n\nThe/at escheat/nn law/nn cannot/md* be/be enforced/vbn now/r
b because/cs it/pps is/bez almost/rb impossible/jj to/to locate/vb such/jj property/nn ,/, Daniel/np declared/vbd
./.\n\n\nDeputy/np Lawrence/np / a/at Tyler/np lawyer/np representing/vbg the/at Texas/np-tl Bankers/np-tl Associ
```


02

텍스트 데이터 전처리



01 데이터 전처리란?

데이터 전처리

- 어떤 작업을 진행하기 이전에 주어진 데이터를 목적에 맞추어 변형 또는 가공하는 과정
- 자연어처리분야 뿐만 아니라 산업 전반적으로 매우 중요한 과정

이미 전처리된 데이터가 아닌
직접 크롤링하여 수집한 데이터 등의 경우
결과의 정확성을 위해 전처리는 필수적임

자연어처리에서 데이터 전처리 과정

토큰화

정제

정규화

등의 과정이 있음

데이터 토큰화(data tokenization)

주어진 코퍼스를 토큰이라는 단위로 나누는 과정

- ⑤ 문장, 단어, 구두점 등 다양한 단위로 토큰을 설정할 수 있음
- ⑤ nltk, OpenNLP, kss 등 토큰화를 수행하는 다양한 오픈소스 툴킷들이 존재함

03 NLTK를 활용한 문장 토큰화

텍스트 데이터에서 데이터는 단락 형태이므로 단락에서 문장을 뽑아내려면 문장 레벨로 토큰화하는 과정이 필요

```
In [1]: 1 text = """NLTK is a leading platform for building Python programs to work with human language data. It provides eas
        2 text
```

```
Out[1]: 'NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use in
terfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries fo
r classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength
NLP libraries, and an active discussion forum. Thanks to a hands-on guide introducing programming fundamentals alongs
ide topics in computational linguistics, plus comprehensive API documentation, NLTK is suitable for linguists, engine
ers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux.
Best of all, NLTK is a free, open source, community-driven project.'
```

■ nltk의 sent_tokenize를 활용해 텍스트를 문장으로 토큰화하기

```
In [2]: 1 from nltk.tokenize import sent_tokenize
```

```
In [3]: 1 sent_tokenize(text)
```

```
Out[3]: ['NLTK is a leading platform for building Python programs to work with human language data.',
'It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of
text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrap-
pers for industrial-strength NLP libraries, and an active discussion forum.',
'Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus
comprehensive API documentation, NLTK is suitable for linguists, engineers, students, educators, researchers, and ind-
ustry users alike.',
'NLTK is available for Windows, Mac OS X, and Linux.',
'Best of all, NLTK is a free, open source, community-driven project.']
```

■ 단어 토큰화는 텍스트를 단어, 구, 의미 있는 문자열로 자르는 과정이다.

```
In [1]: 1 text = "If you really want to do something, you'll find a way. If you don't, you'll find an excuse."
2 text
```

```
Out[1]: "If you really want to do something, you'll find a way. If you don't, you'll find an excuse."
```

< 가공 전 텍스트 >

■ nltk의 word_tokenize를 활용해 텍스트를 단어로 토큰화하기

```
In [2]: 1 from nltk.tokenize import word_tokenize
        2 word_tokenize(text)
```

```
Out[2]: ['If',
        'you',
        'really',
        'want',
        'to',
        'do',
        'something',
        '.',
        'you',
        "'ll",
        'find',
        'a',
        'way',
        '.',
        'If',
        'you',
        'do',
        "n't",
        'you',
        '']
```

If you really want to do something, you'll find a way.
If you don't, you'll find an excuse.

■ nltk의 WordPunctTokenizer를 활용한 토큰화

```
In [3]: 1 from nltk.tokenize import WordPunctTokenizer
        2 WordPunctTokenizer().tokenize(text)
```

```
Out[3]: ['If',
        'you',
        'really',
        'want',
        'to',
        'do',
        'something',
        '.',
        'you',
        "'",
        'll',
        'find',
        'a',
        'way',
        '.',
        'If',
        'you',
        'don',
        "'",
        't',
        'you',
        '']
```

If you really want to do something, you'll find a way.
If you don't, you'll find an excuse.

1

상황과 목적에 맞는 토큰화 툴을 사용해야 한다.

- 앞의 예시와 같이, nltk의 word_tokenizer 의 경우 의미에 따라 단어를 분리하지만 WordPunctTokenizer의 경우 구두점을 기준으로 단어를 분리하므로 알맞은 토큰화 툴을 선택해야 함

예시) don't → do / n't // don't → don / ' / t

2

구두점 또는 특수문자가 이미 포함된 단어의 경우

- She finally completed her Ph.D. degree. At that point, her boyfriend was eating hamburger which only costs \$7.59 while reading J.K. Rowling's new book
- 위 문장의 경우 "Ph.D.", "\$7.59", J.K. 의 경우 이미 온점이 포함되어 있기 때문에 Ph / D / 또는 \$7 / 59 / 와 같이 분리하지 않도록 주의해야함

데이터 정제(data cleansing/cleaning)

- 데이터에서 손상되거나 부정확한 부분을 감지하고 수정하는 과정
- 데이터의 불완전하거나 부정확하거나 관련이 없는 부분을 식별한 뒤에 해당 부분을 대체, 수정 또는 삭제하는 과정

데이터 정규화(data normalization)

- 텍스트 데이터에서 정규화는 표현 방법이 다른 단어들을 통합시켜 같은 단어로 통합시키는 과정

“

영어권에서 대소문자 통합과정은 일반적으로 소문자로 변환

”

- 대소문자가 존재하는 영어권 언어의 경우 대소문자 통합과정은 서로 다른 단어의 개수를 줄일 수 있음
- 텍스트 내에서 ‘language’란 단어를 찾는 과정에서 문장 맨 앞의 ‘Language’의 L은 대문자이므로 찾지 못하는 상황을 방지함
- 사람 / 회사의 이름 등과 같이 반드시 대문자로 존재하는 단어를 고려하여 통합해야 함

파이썬 내장함수 .lower()를 활용해 소문자로 통합하기

```
In [1]: 1 def lowertext(text):  
2         return(text.lower())  
3  
4 text = "Happiness depends upon ourselves. Remember that happiness is a way of travel not a destination."  
5  
6 lowertext(text)
```

```
Out[1]: 'happiness depends upon ourselves. remember that happiness is a way of travel not a destination.'
```


어간 추출(Stemming)

- 접미사를 삭제하거나 대체하여 문장의 각 단어를 어근 형태로 변환하는 과정
- ⑤ 접미사를 삭제 또는 대체하는 과정이므로 결과물이 사전에 존재하지 않는 단어일 수 있음
- ⑤ 다양한 어간 추출 툴킷이 존재하지만 언어별로 잘 동작하지 않을 수 있음

가공 전 텍스트

```
In [1]: 1 text = "He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way
         2 text
Out[1]: 'He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way to make
         his dream come true'
```

nltk의 PorterStemmer를 활용해 어간 추출하기

```
In [2]: 1 from nltk.stem import PorterStemmer
2 [PorterStemmer().stem(word) for word in text.split()]
```

```
'be',
'a',
'comput',
'scientist.',
'So',
'he',
'studi',
'program',
'languages.',
'studi',
'hard',
'is',
'onli',
'way',
'to',
'make',
'hi',
'dream',
'come',
'true']
```

He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way to make his dream come true

nltk의 LancasterStemmer를 활용해 어간 추출하기

```
In [3]: 1 from nltk.stem import LancasterStemmer
2 [LancasterStemmer().stem(word) for word in text.split()]
```

```
'be',
'a',
'comput',
'scientist.',
'so',
'he',
'study',
'program',
'languages.',
'study',
'hard',
'is',
'on',
'way',
'to',
'mak',
'his',
'dream',
'com',
'tru']
```

He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way to make his dream come true

표제어 추출(Lemmatizing)

- 정확하게 의도된 품사(POS, part-of-speech)와 문장에 있는 단어의 의미를 식별하는 과정



- 활용 어미를 없애고 단어를 사전이나 어휘에 존재하는 기본형으로 변환
- Wordnet과 같은 태그가 있는 사전을 사용해 가공 전 텍스트를 표제어(lemma)로 변환

10 NLTK의 품사 태깅

품사 태그(POS tagging)

- 단어는 형태가 같아도 품사에 따라 의미가 크게 달라질 수 있음
- 정의와 문맥에 따라 텍스트의 특정 부분에 해당하는 단어를 마크 업하는 과정임

```
In [2]: 1 from nltk.tokenize import word_tokenize
        2 from nltk.tag import pos_tag
        3
        4 text = "She finally completed Ph.D. degree."
        5
        6 word_token = word_tokenize(text)
        7 print(pos_tag(word_token))

[('She', 'PRP'), ('finally', 'RB'), ('completed', 'VBD'), ('Ph.D.', 'NNP'), ('degree', 'NN'), ('.', '.'), ('.', '.')]

```

품사를 지정하여 표제어 추출하기

```
In [7]: 1 WordNetLemmatizer().lemmatize('better', pos='a')
```

```
Out[7]: 'good'
```

```
In [8]: 1 WordNetLemmatizer().lemmatize('funnier', pos='a')
```

```
Out[8]: 'funny'
```

```
In [9]: 1 WordNetLemmatizer().lemmatize('meeting', pos='n')
```

```
Out[9]: 'meeting'
```

```
In [10]: 1 WordNetLemmatizer().lemmatize('meeting', pos='v')
```

```
Out[10]: 'meet'
```

```
In [11]: 1 WordNetLemmatizer().lemmatize('expected', pos='v')
```

```
Out[11]: 'expect'
```

```
In [1]: 1 text = "He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way  
2 text
```

```
Out[1]: 'He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way to make  
his dream come true'
```

< 가공 전 텍스트 >

nlk의 WordNetLemmatizer를 활용해 표제어 추출하기

```
In [5]: 1 from nltk.stem import WordNetLemmatizer
        2 [WordNetLemmatizer().lemmatize(word) for word in text.split()]
```

```
Out[5]: ['He',
         'always',
         'wanted',
         'to',
         'be',
         'a',
         'computer',
         'scientist.',
         'So',
         'he',
         'study',
         'programming',
         'languages.',
         'Studying',
         'hard',
         'is',
         'only',
         'way',
         'to',
         'make']
```

He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way to make his dream come true

품사를 지정하여 표제어 추출하기

```
In [6]: 1 from nltk.stem import WordNetLemmatizer
        2 [WordNetLemmatizer().lemmatize(word, 'v') for word in text.split()]
```

```
Out[6]: ['He',
         'always',
         'want',
         'to',
         'be',
         'a',
         'computer',
         'scientist.',
         'So',
         'he',
         'study',
         'program',
         'languages.',
         'Studying',
         'hard',
         'be',
         'only',
         'way',
         'to',
         'make']
```

He always wanted to be a computer scientist. So he studies programming languages. Studying hard is only way to make his dream come true

불용어(stopwords)

- 문장 형성에 있어서 중요한 구문론적 가치를 지니고 있지만 무시할 수 있거나 최소한의 의미적 가치를 지닌 단어

nlTK의 불용어 리스트

'i', 'me', 'my', 'myself', 'you', 'you're', 'am', 'is', 'are', 'was', 'were', 'be' 등

```
In [ ]: 1 import nltk
        2 from nltk.corpus import stopwords
        3 nltk.download('stopwords')
        4
        5 stopwordslist = stopwords.words('english')
        6 stopwordslist
```

APPLICATION

03

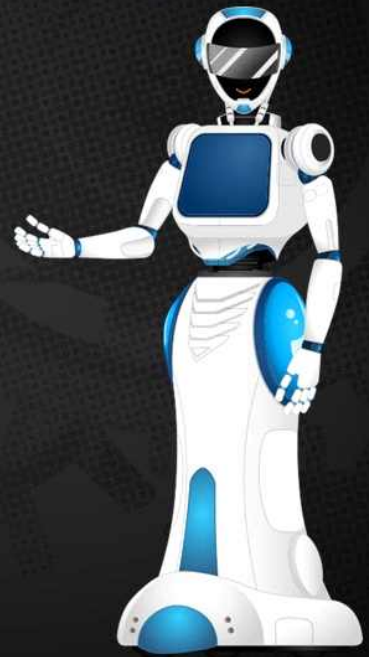
실습



SUMMARY

학습정리

- ◆ 자연어 데이터 묶음 코퍼스
- ◆ NLTK를 활용한 코퍼스에 대한 분석
- ◆ 텍스트 수준 텍스트 전처리
- ◆ 단어 수준 텍스트 전처리
- ◆ 정규 표현식



EXPANSION

확장하기

1. 코퍼스로 우리는 무엇을 할 수 있을까요?
2. NLTK란 무엇일까요?
3. 데이터의 토큰화는 무엇인가요?
4. NLTK에서 어간을 추출하는 방법에는 어떤 것들이 있을까요?
5. 불용어는 무엇인가요?



참고 문헌

REFERENCE

The copyright clearance center
is providing this document as
a public service. It is not
responsible for any damage or
loss caused by its use.

- ◆ 참고 논문
 - Young, T, Hazarika, D., Poria, S., & Cambria, E. (2017). Recent Trends in Deep Learning Based Natural Language Processing. arXov preprint arXiv:1708.02709
- ◆ 참고 사이트
 - 용어들에 대한 정의: <https://ko.wikipedia.org/wiki>.
 - 퍼블릭에이아이(www.public.co.kr)
- ◆ 참고 서적
 - 김기현, 『김기현의 자연어 처리 딥러닝 캠프 파이토치 편』, 한빛미디어, 2019
 - 잘라지 트하나키, 『파이썬 자연어 처리의 이론과 실제』, 에이콘, 2017

☎ 서체 출처 : 에스코어드림체-(주)에스코어, 나눔글꼴체-(주)네이버