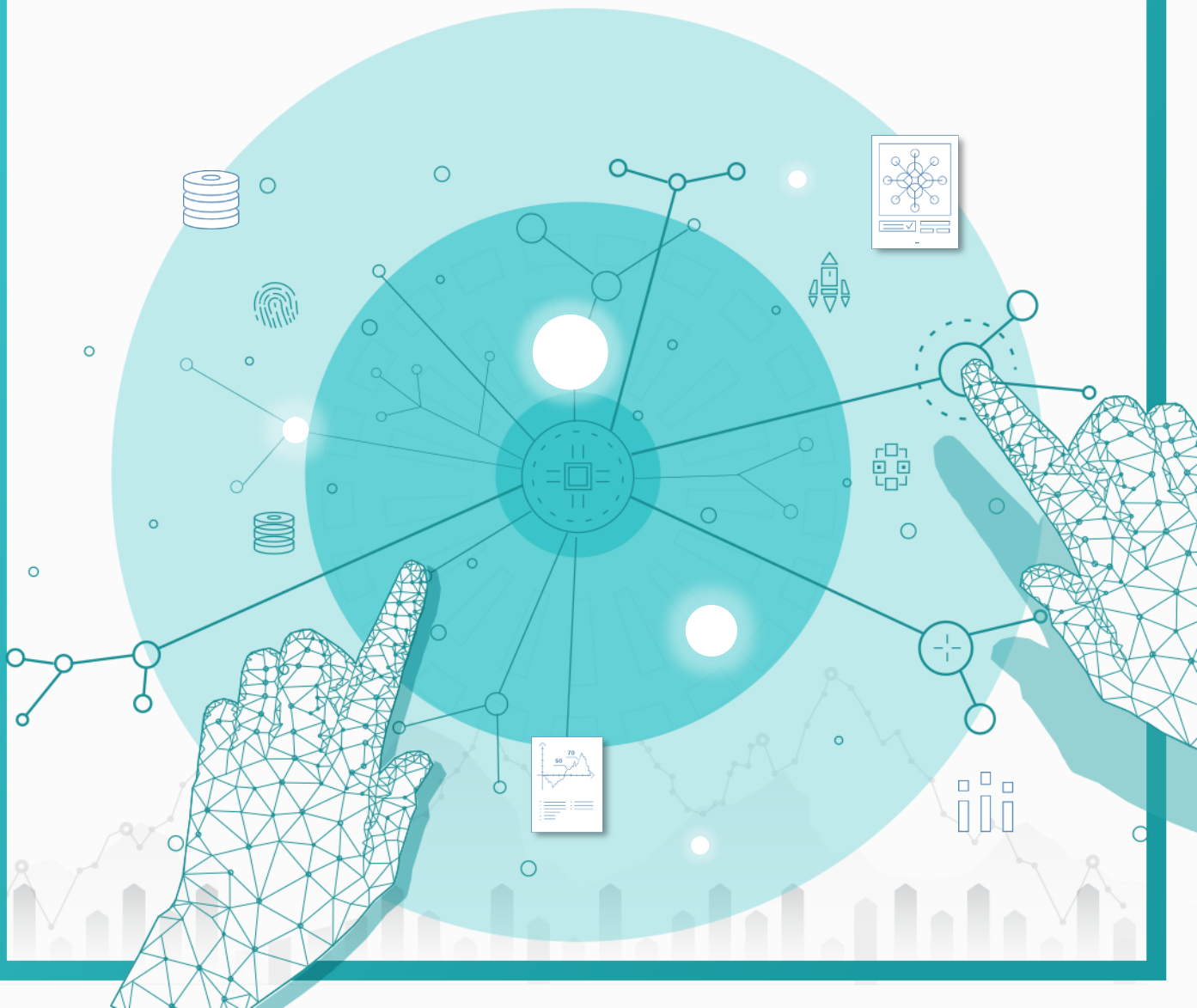
한국기술교육대학교
온라인평생교육원

파이썬을 활용한 인공지능 자연어 처리(실습)

인공지능 자연어 전처리



인공지능 자연어 전처리

학습 목표

1. 인공지능 자연어 처리 절차에 대해 설명할 수 있다.
2. 인공지능 자연어 전처리를 수행할 수 있다.

학습 내용

1. 영문 자연어 전처리
2. 한글 자연어 전처리

1. 영문 자연어 전처리

1) 영문 자연어 전처리 개요

(1) 정제(Cleaning)

- 특수문자 제거
  - 특수문자 : '!"#$%&₩'()*+,-./:;<=>?@[₩₩]^_`{|}~'
- 대소문자 통일
  - KOREA, Korea, korea → korea

(2) 토큰화(Tokenization)

- 코퍼스(Corpus)에서 분리자(Separator)를 포함하지 않는 연속적인 문자열 단위로 분리
- 토큰화 단위에 따른 분류
  - 문장(Sentence)단위 토큰화
  - 단어(Word)단위 토큰화
- 토큰화 단위에 따른 분류
  - 어절 단위 : 띄어쓰기 단위, 영문 적합
  - 형태소 단위 : 한글 적합
  - Subword : 형태소와 유사, 의미 대신 통계적 방법 적용

(3) 불용어 제거(Stopword Elimination)

- 전치사, 관사 등 문장이나 문서의 특징을 표현하는데 불필요한 단어를 제거하는 단계

  (예) you, my, the, a, of, at 등

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(1) 정제(Cleaning)
| 특수문자 제거

- 제거 대상 특수문자의 종류

```
import string

print(string.punctuation)
```

```
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

- 영문 자연어 전처리를 위한 코퍼스(Corpus)

Beneath it were the words: “Stay Hungry. Stay Foolish.” It was their farewell message as they signed off. And I have always wished that for myself. And now, as you graduate to begin anew, I wish that for you.

출처 : 스티브 잡스, 2005, 스탠포드대학교 졸업식

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(1) 정제(Cleaning)
| 특수문자 제거

- 정규식을 이용한 특수문자 제거
  - re.sub('[^\w\.\s]','',text)를 이용하여 '.'을 제외한 특수문자 제거

```
import re

cleaned_text = re.sub('[^\w\.\s]','',text)
print(cleaned_text)
```

```
Beneath it were the words Stay Hungry.
Stay Foolish. It was their farewell message as they
signed off. And I have always wished that for myself.
And now as you graduate to begin anew I wish that
for you..
```

- 정규식을 이용한 '\n' 문자 제거
  - re.sub('\n','',cleaned_text)을 이용하여 '\n' 문자 제거

```
cleaned_text = re.sub('\n','',cleaned_text)
print(cleaned_text)
```

```
Beneath it were the words Stay Hungry. Stay Foolish. It was their farew
```

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(2) 토큰화(Tokenization)
| nltk를 이용한 영문 토큰화

- nltk를 이용한 영문 토큰화를 위해 punkt 모듈 다운로드 및 활용
  - nltk punkt 모듈 다운로드

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(2) 토큰화(Tokenization)
| 문장, 단어 단위 토큰화

- nltk.sent_tokenize를 이용한 문장 단위 토큰화 수행

```
sent_tokens = nltk.sent_tokenize(cleaned_text)
print(sent_tokens)
```

```
['Beneath it were the words Stay Hungry.', 'Stay Foolish.', 'It was th
```

- nltk.word_tokenize를 이용한 단어 단위 토큰화 수행

```
tokens = nltk.word_tokenize(cleaned_text)
tokens
```

```
['Beneath',
 'it',
 'were',
 'the',
 'words',
 'Stay',
 'Hungry',
 '.',
 'Stay',
```

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(3) 불용어 제거(Stopword Elimination)
| nltk패키지의 stopword를 이용한 불용어 제거

- nltk패키지의 stopword 다운로드

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(3) 불용어 제거(Stopword Elimination)
| 영문 불용어 로드 및 출력

- nltk패키지의 stopword 중 영문 불용어를 로드하고 출력

```
from nltk.corpus import stopwords
stop = stopwords.words('english')
print(stop)
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "
```

2) 영문 자연어 전처리 적용

(3) 불용어 제거(Stopword Elimination)
| nltk패키지의 stopword를 이용한 불용어 제거

- nltk패키지의 stopword와 python List Comprehension을 이용한 불용어 제거

```
tokens = [token for token in tokens if token not in stop]
print(tokens)
```

```
['Beneath', 'words', 'Stay', 'Hungry', '.', 'Stay', 'Foolish', '.',
```

(4) 소문자화(Lower Capitalization)
| 정제(Cleaning) 과정 중 하나인 소문자화

- lower( ) 함수와 List Comprehension을 이용한 소문자화

```
print('소문자화 수행전 : ', tokens)
tokens_lower = [token.lower() for token in tokens]
print('소문자화 수행후 : ', tokens_lower)
```

```
소문자화 수행전 :  ['Beneath', 'words', 'Stay', 'Hungry', 'Stay', 'Fool
소문자화 수행후 :  ['beneath', 'words', 'stay', 'hungry', 'stay', 'fool
```

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(3) 불용어 제거(Stopword Elimination)
| nltk패키지의 stopword를 이용한 불용어 제거

- nltk패키지의 stopword와 python List Comprehension을 이용한 불용어 제거

```
tokens = [token for token in tokens if token not in stop]
print(tokens)
```

```
['Beneath', 'words', 'Stay', 'Hungry', '.', 'Stay', 'Foolish', '.',
```

- 불용어 제거 후 단어의 길이가 2글자 이하인 단어 제거

```
tokens = [token for token in tokens if len(token) >= 3]
print(tokens)
```

```
['Beneath', 'words', 'Stay', 'Hungry', 'Stay', 'Foolish', 'farewell',
```

1. 영문 자연어 전처리

2) 영문 자연어 전처리 적용

(4) 소문자화(Lower Capitalization)
| 정제(Cleaning) 과정 중 하나인 소문자화

- lower( ) 함수와 List Comprehension을 이용한 소문자화

```
print('소문자화 수행전 : ', tokens)
tokens = [token.lower() for token in tokens]
print('소문자화 수행후 : ', tokens)
```

```
소문자화 수행전 :  ['Beneath', 'words', 'Stay', 'Hungry', 'Stay',
소문자화 수행후 :  ['beneath', 'words', 'stay', 'hungry', 'stay',
```

2. 한글 자연어 전처리

1) 한글 자연어 전처리 개요

(1) 토큰화(Tokenization)

- 코퍼스(Corpus)에서 분리자(Separator)를 포함하지 않는 연속적인 문자열 단위로 분리

- 토큰화 단위에 따른 분류
  - 문장(Sentence)단위 토큰화
  - 단어(Word)단위 토큰화

- 단어 단위 토큰화 방법에 따른 분류
  - 어절 단위 : 띄어쓰기 단위, 영문에 적합
  - 형태소 단위 : 한글에 적합
  - Subword : 형태소와 유사, 의미 대신 통계적 방법 적용

(2) 불용어 제거(Stopword Elimination)

- 문장이나 문서의 특징을 표현하는데 불필요한 단어를 제거하는 단계
  불용어 사전을 활용하여 제거, 불용어 사전 관리

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(1) 정제(Cleaning)
| 특수문자 제거

- 제거 대상 특수문자의 종류 - 영문과 동일

```
import string

print(string.punctuation)
```

```
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

- 한글 자연어 전처리를 위한 코퍼스(Corpus)

그 밑에는 "계속 배고픔을 느끼세요. 계속 바보로 남으세요" 라는 문구가 새겨져 있었습니다. 그들이 전한 마지막 인사말이었습니다. 계속 배고픔을 느끼세요, 계속 바보로 남으세요. 그리고 저는 항상 제 자신이 그렇길 바랬습니다. 이제는 졸업을 하고 새로운 출발을 하는 여러분에게 바라는 바 입니다.

출처 : 스티브 잡스, 2005, 스탠포드대학교 졸업식

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(1) 정제(Cleaning)
| 특수문자 제거

- 정규식을 이용한 특수문자 제거
  - re.sub('[^\w\.\s]','',text)를 이용하여 '.'을 제외한 특수문자 제거

```
import string
import re

cleaned_text = re.sub('[^\w\s]','',text)
print(cleaned_text)
```

```
그 밑에는 계속 배고픔을 느끼세요
계속 바보로 남으세요 라는 문구가 새겨져 있었습니다
그들이 전한 마지막 인사말이었습니다  계속 배고픔을
느끼세요 계속 바보로 남으세요
그리고 저는 항상 제 자신이 그렇길 바랬습니다
이제는 졸업을 하고 새로운 출발을 하는
여러분에게 바라는 바 입니다
```

- 정규식을 이용한 '\n' 문자 제거
  - re.sub('\n','',cleaned_text)을 이용하여 '\n' 문자 제거

```
cleaned_text = re.sub('\n',' ',cleaned_text)
print(cleaned_text)
```

```
그 밑에는 계속 배고픔을 느끼세요 계속 바보로 남으세요 라는 문구가
```

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(1) 정제(Cleaning)
| 띄어쓰기가 안되어 있는 경우 띄어쓰기 적용

- 띄어쓰기가 적용되지 않은 상태로 코퍼스 변환
  - 정규식을 이용하여 띄어쓰기가 적용되지 않은 상태로 코퍼스 변환

```
non_space = re.sub('\s', '', text)
print(non_space)
```

```
그밑에는"계속배고픔을느끼세요.계속바보로남으세요"라는문구가새겨져있었습니다.
```

- 띄어쓰기 적용을 위한 PyKoSpacing 설치
  - PyPi.org에 등록되지 않은 상태이므로 git을 이용한 설치

```
!pip install git+https://github.com/haven-jeon/PyKoSpacing.git
```

```
Collecting git+https://github.com/haven-jeon/PyKoSpacing.git
  Cloning https://github.com/haven-jeon/PyKoSpacing.git to /tmp/pip-req-b
  Running command git clone -q https://github.com/haven-jeon/PyKoSpacing.
```

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(1) 정제(Cleaning)
| 띄어쓰기가 안되어 있는 경우 띄어쓰기 적용

- PyKoSpacing을 이용한 띄어쓰기 적용
  - spacing( ) 함수를 이용하여 띄어쓰기 적용

```
from pykospacing import Spacing

spacing = Spacing()
new_sent = spacing(non_space)
print(new_sent)
```

```
그 밑에는 "계속 배고픔을 느끼세요. 계속 바보로 남으세요"라는 문구가 새
```

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(2) 토큰화(Tokenization)
| 한글 문장 단위 토큰화

- 한글 문장 단위 토큰화를 위해 kss 모듈 설치 및 활용
  - 한글 문장 단위 토큰화 적용을 위해 kss 모듈 설치

```
!pip install kss
```

```
Collecting kss
  Downloading kss-3.2.0.tar.gz (42.4 MB)
     |
```

- kss 모듈을 활용한 한글 문장단위 토큰화 적용
  - kss.split_sentences( )를 이용하여 한글 문장단위 토큰화 적용

```
import kss

sent_tokens = kss.split_sentences(cleaned_text)
print(sent_tokens)
```

```
['그밑에는계속배고픔을느끼세요', '계속바보로남으세요라는문구가새겨져
```

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(2) 토큰화(Tokenization)
| 한글 단어 단위 토큰화

- 한글 단어단위 토큰화 적용 - 어절 토큰화

```
def tokenizer(words):
  tokens = words.split()
  return tokens

tokens = tokenizer(cleaned_text)
print(tokens)
```

```
['그', '밑에는', '계속', '배고픔을', '느끼세요', '계속', '바보로',
```

2) 한글 자연어 전처리 적용

(3) 불용어 제거(Stopword Elimination)
| 한글 불용어 사전 작성 및 업로드

- 한글 불용어 사전 작성 및 stopword_dict.csv로 저장

	A
1	stopword
2	그
3	라는
4	바
5	입니다
6	하고

- 한글 불용어 사전 업로드

```
from google.colab import files
import os

data_dir = 'data'

if not os.path.exists(data_dir):
  os.mkdir(data_dir)
os.chdir(data_dir)
files.upload()
os.chdir('..')
```

'파일 선택' 클릭 후
stopword_dict.csv 파일 선택

```
파일 선택  stopword_dict.csv
• stopword_dict.csv(application/vnd.ms-excel) - 50 bytes, last modified: 2021.
Saving stopword_dict.csv to stopword_dict.csv
```

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(3) 불용어 제거(Stopword Elimination)
| 한글 불용어 사전 작성 및 업로드

- pandas 패키지를 이용하여 업로드된 불용어 사전 출력

```
import pandas as pd

stop = pd.read_csv('./data/stopword_dict.csv')
print(stop[:10])
```

```
  stopword
0        그
1       라는
2        바
3      입니다
4       하고
```

- pandas 패키지를 이용하여 업로드된 불용어 사전 리스트로 변환

```
print(list(stop.stopword))
```

```
['그', '라는', '바', '입니다', '하고']
```

2. 한글 자연어 전처리

2) 한글 자연어 전처리 적용

(3) 불용어 제거(Stopword Elimination)
| 불용어 사전을 활용한 불용어 제거

- 파이썬 List Comprehension을 이용하여 불용어 제거

```
tokens = [token for token in tokens if token not in list(stop.stopword)]
print(tokens)
```

```
['밑에는', '계속', '배고픔을', '느끼세요', '계속', '바보로', '남으세요', '문구가',
```