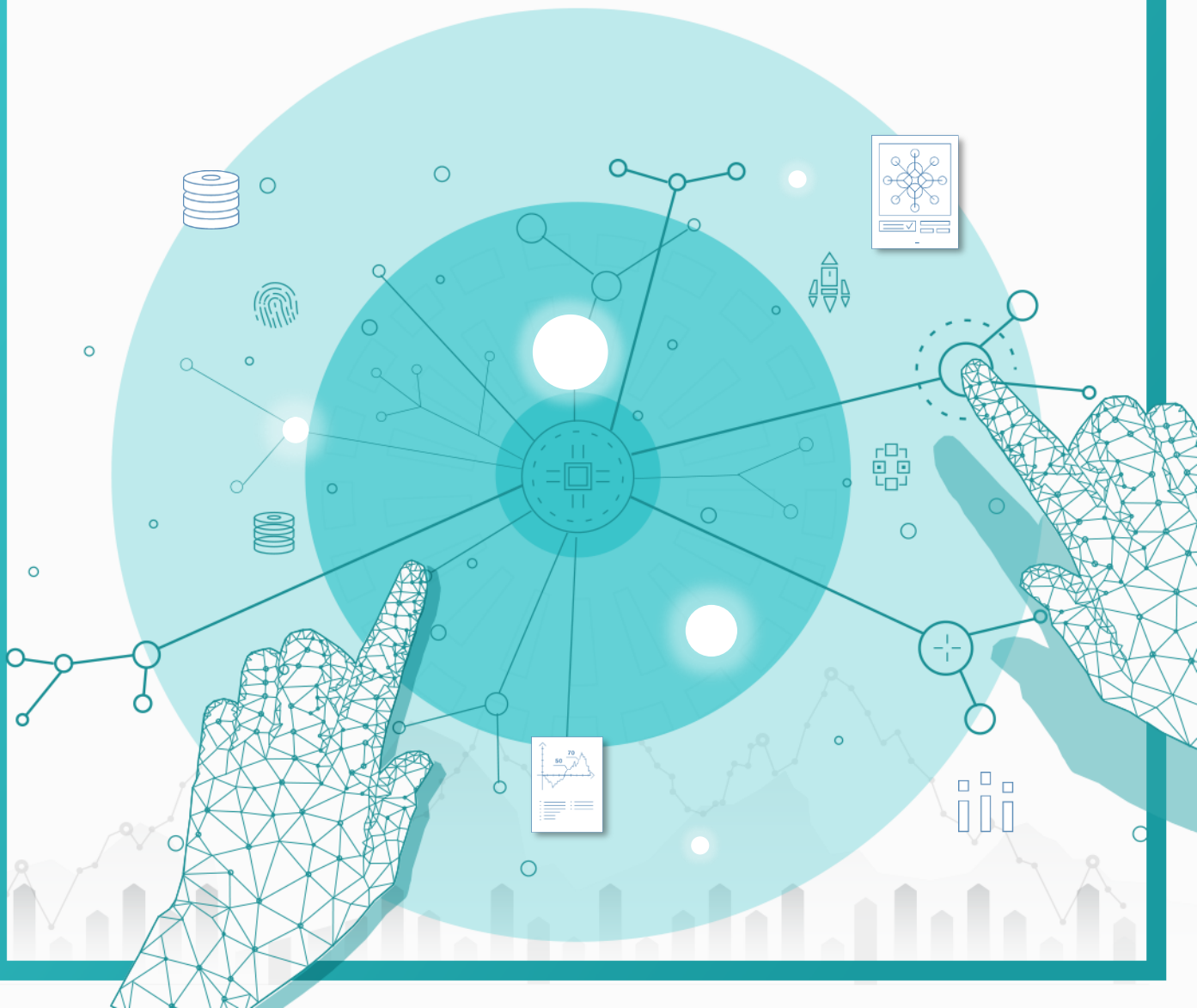
한국기술교육대학교
온라인평생교육원

파이썬을 활용한 인공지능 자연어 처리(실습)

자연어 처리를 위한 토픽 모델링



자연어 처리를 위한 토픽 모델링

학습 목표

1. 토픽 모델링을 위한 형태소 분석을 수행할 수 있다.
2. LDA를 이용하여 토픽 모델링을 수행할 수 있다.
3. 토픽 모델을 시각화 할 수 있다.

학습 내용

1. 토픽 모델링을 위한 형태소 분석
2. LDA를 이용한 토픽 모델링
3. 토픽 모델 시각화

1. 토픽 모델링을 위한 형태소 분석

1) KoNLPy를 이용한 형태소 분석

(1) 형태소 분석 수행을 위한 데이터 확인

- Pandas 패키지의 DataFrame으로부터 일부 데이터 출력

```
text = df_news[(df_news.category==7) & (df_news.num==7001)]['news'].
print(text)
```

```
23일 도쿄올림픽이 개막식을 열고 본격적인 스포츠 축제의 시작을 알리는 가운데,
```

※출처: 공공데이터포털, 한국언론진흥재단_뉴스빅데이터_메타데이터_올림픽, 2021, https://www.data.go.kr

(2) Okt 클래스를 이용한 형태소 분석

- Okt 클래스의 pos( ) 함수를 이용한 형태소 분석

```
okt = konlpy.tag.Okt()
morphs = okt.pos(text, stem = True)
print(morphs)
```

```
[('정현', 'Noun'), ('한국', 'Noun'), ('선수', 'Noun'), ('최초', 'Noun'), ('로', 'Josa'),
```

1. 토픽 모델링을 위한 형태소 분석

1) KoNLPy를 이용한 형태소 분석

(3) 형태소 분석 수행을 위한 함수 정의

- 전체 데이터의 형태소 분석 수행을 위한 함수 정의

① 명사, 형용사, 동사만 추출

② 형태소 길이가 2 이상인 토큰만 추출

```
def get_words(text):
    morphs = okt.pos(text, stem = True)

    word_list = []
    for word, pos in morphs:
        if pos == 'Noun' or pos == 'Adjective' or pos == 'Verb':
          if len(word) > 1:
            word_list.append(word)

    words = ' '.join(word_list)
    return words
```

1. 토픽 모델링을 위한 형태소 분석

2) KoNLPy를 이용한 형태소 분석 결과 저장

(1) 전체 데이터의 형태소 분석 및 결과 저장

- Pandas 패키지의 apply( )를 이용하여 전체 데이터에 적용

```
df_news['words'] = df_news.news.apply(get_words)
```

- 별도의 칼럼(Series)을 추가하여 저장

	news
	"종주국 자존심 지킨다" 25일 도쿄올림픽 태권도 남자 68kg급에 출전하는 이대훈(...
	호텔 전체 빌려 급식지원센터로 ..생선 제외, 육류 뉴질랜드·호주산 ..채소·과일도...
	23일 도쿄올림픽이 개막식을 열고 본격적인 스포츠 축제의 시작을 알리는 가운데, 재...

	words
	종주국 자존심 지키다 도쿄올림픽 태권도 남자 출전 하다 이대훈 왼쪽 진천선수촌 동료...
	호텔 전체 빌리다 급식 센터 생선 제외 육류 뉴질랜드 호주 채소 과일 후쿠시마 써다...
	도쿄올림픽 개막식 열다 본격 스포츠 축제 시작 알리다 가운데 재계 대회 출전 선수 ...

※출처: 공공데이터포털, 한국언론진흥재단_뉴스빅데이터_메타데이터_올림픽, 2021, https://www.data.go.kr

2. LDA를 이용한 토픽 모델링

1) LDA 적용을 위한 CountVectorizer 생성

(1) CountVectorizer 개요

- CountVectorizer

Scikit-learn 패키지의 코퍼스 단어 토큰을 생성하고,
각 단어의 수를 계산하여 BOW(Bag Of Words) 인코딩 벡터 생성을 위한 클래스

- TfidfVectorizer

TF-IDF 값을 사용하여 단어의 가중치를 조정하고,
BOW(Bag Of Words) 인코딩 벡터 생성을 위한 클래스

→ 모델 다운로드 및 Fine-tunning을 이용하여 활용 가능

2. LDA를 이용한 토픽 모델링

1) LDA 적용을 위한 CountVectorizer 생성

(2) CountVectorizer 적용

- Scikit-learn 패키지의 CountVectorizer 클래스 객체 생성

```
from sklearn.feature_extraction.text import CountVectorizer
count_vectorizer = CountVectorizer(max_df=0.1, max_features=1000,
                                   min_df=2, ngram_range=(1,2))
```

① 전체의 10% 이상 자주 등장하는 단어 제외
② Features의 개수: 1,000
③ 특정 문서에 의존하는 단어 제외(2개 문서 미만)
④ unigram과 bigram 적용

- 생성된 Feature Vector 확인

```
feature_names = count_vectorizer.get_feature_names()
print(feature_names[:5])
```

```
['가격', '가구', '가능성 있다', '가능하다', '가량']
```

2. LDA를 이용한 토픽 모델링

2) LDA 클래스를 이용한 LDA 적용

(1) LDA 클래스 객체 생성

- 토픽 수를 설정하여 LatentDirichletAllocation 클래스 객체 생성 및 fit( ) 호출

```
from sklearn.decomposition import LatentDirichletAllocation
topic_cnt = 8
lda = LatentDirichletAllocation(n_components=topic_cnt)
lda.fit(feat_vect)
```

```
LatentDirichletAllocation(batch_size=128, doc_topic_prior=None,
                          evaluate_every=-1, learning_decay=0.7,
                          learning_method='batch', learning_offset=10.0,
                          max_doc_update_iter=100, max_iter=10,
                          mean_change_tol=0.001, n_components=8, n_jobs=None,
                          perp_tol=0.1, random_state=None,
                          topic_word_prior=None, total_samples=1000000.0,
                          verbose=0)
```

(2) LDA 적용 결과 확인

- 토픽별 단어 빈도수 높은 단어 목록 10개 추출

```
topic_num = 2
topic = lda.components_[topic_num]

topic_word_indexes = topic.argsort()[::-1]
top_indexes=topic_word_indexes[:10]

feature_name_list = ' '.join([feature_names[i] for i in top_indexes])
print(feature_name_list)
```

```
주택 아파트 규제 인상 대출 통화 부동산 투자 가상 시행
```

3. 토픽 모델 시각화

1) pyLDAvis 설치 및 적용

(1) pyLDAvis 개요

- pyLDAvis

Scikit-learn 패키지의 LDA 모델에 최적화된 토픽 모델 시각화 라이브러리

(2) pyLDAvis 설치

- pip install을 이용하여 pyLDAvis 설치

```
!pip install pyLDAvis
```

```
Requirement already satisfied: pyLDAvis in /usr/local/lib/python3
Requirement already satisfied: scikit-learn in /usr/local/lib/pyt
Requirement already satisfied: pandas>=1.2.0 in /usr/local/lib/py
```

3. 토픽 모델 시각화

1) pyLDAvis 설치 및 적용

(3) pyLDAvis 적용

- pyLDAvis import 및 jupyter notebook에서 사용 가능하도록 설정

```
import pyLDAvis.sklearn

pyLDAvis.enable_notebook()
```

- pandas 버전 1.2 업그레이드 필요

```
!pip install --upgrade pandas==1.2
```

```
Collecting pandas==1.2
  Downloading pandas-1.2.0-cp37-cp37m-manylinux1_x86_64.whl (9.9 MB)
     |████████████████████████████████████████████████████████████████| 9.
```

3. 토픽 모델 시각화

2) pyLDAvis를 이용한 토픽 모델 시각화

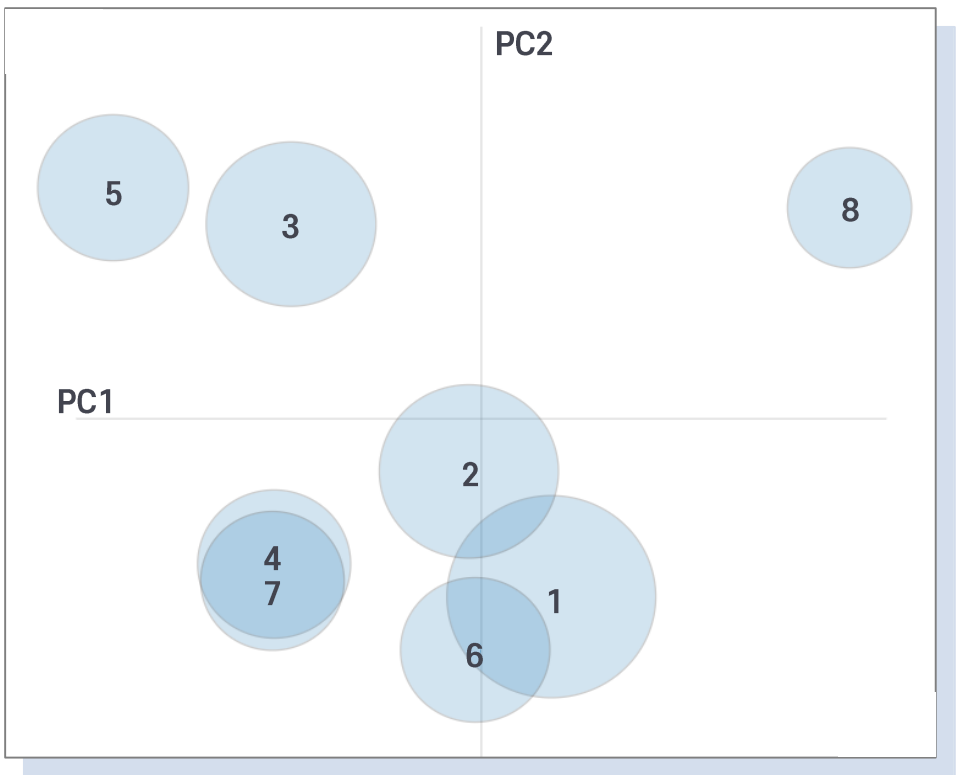
(1) 토픽 모델 시각화 활용

- pyLDAvis의 prepare( ) 함수 및 display( ) 함수를 이용한 토픽 모델 시각화
  - LDA 모델, Feature Vector, CountVectorizer 사용

```
vis = pyLDAvis.sklearn.prepare(lda, feat_vect, count_vectorizer)
pyLDAvis.display(vis)
```

- 토픽 간 Distance Map

Intertopic Distance Map(via multidimensional scaling)



3. 토픽 모델 시각화

2) pyLDAvis를 이용한 토픽 모델 시각화

(1) 토픽 모델 시각화 활용

- 토픽별 Max 30 단어

Top-30 Most Salient Terms