8. 텍스트의 비지도학습

감정 분석이나 문서 분류 등은 감정이나 분류와 같은 정답이 있고, 텍스트와 정답의 관계를 기계학습을 통해 예측한다. 이렇게 정답이 있는 문제를 다룰 때는 지도학습(supervised learning)을 사용한다.

그러나 텍스트를 분석할 때는 정답이 없더라도 방대한 텍스트에 어떠한 정보가 있는지 요약해서 보고 싶을 때가 있다. 이럴 때 사용할 수 있는 것이 비지도학습(unsupervised learning)이다.

비지도학습은 데이터의 구조를 가정하고, 이 가정된 구조에 맞춰 데이터를 분석하는 방법이다. 텍스트에 적용할 수 있는 대표적인 비지도학습으로는 토픽 모형(topic model) 또는 잠재 디리클레 할당(Latent Dirichlet Allocation)이 있다.

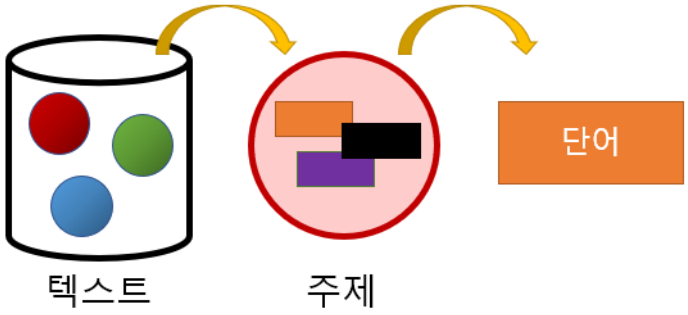
8.1. 토픽 모형

토픽 모형 또는 잠재 디리클레 할당은 2003년 David Blei, Andrew Ng, 그리고 Michael I. Jordan 세 사람이 개발한 텍스트 분석 방법이다. 이후로 대단한 관심을 사서 2018년 현재 해당 논문은 11,188번 인용되었다.(semanticscholar 기준) 참고로 1997년에 개발되어 자연어 처리에 가장 많이 사용되는 딥러닝 기법인 LSTM이 5,949번 인용된 것과 비교하면 토픽 모형이 얼마나 많은 관심을 받은 방법인지 짐작할 수 있을 것이다.

8.1.1. 토픽 모형의 가정

토픽 모형은 다음과 같은 구조를 가정한다.

- 하나의 텍스트에는 여러 가지 주제가 일정 비율로 포함되어 있다
- 주제마다 고유한 단어의 분포가 있다
- 텍스트의 단어 분포는 각 주제의 비율과 각 주제의 단어 분포에 따라 정해진다



현재는 기본적인 토픽 모형에 더해 다양한 확장 버전이 개발되어 있다. 예를 들면 주제의 시간에 따른 변화라든지, 저자의 독특한 개성 등을 포함시켜 분석할 수 있다. 그러나 이러한 분석을 위해서는 더 많은 텍스트들이 필요하고 학습시키기가 까다롭기 때문에 실제로 많이 사용되지는 않는다.

8.2. 토픽 모형 실습

8.2.1. 다음 뉴스 기사 수집

```
import requests
import lxml.html
```

2018년 4월 2일부터 8일까지 인공지능으로 검색했을 때 url

```
url = 'http://search.daum.net/search?p={}&w=news&cluster=n&q=인공지능&sort=recency&DA=PGD&sd=20180402000000&ed=20180408235959&period=u'
```

1페이지부터 250페이지까지 뉴스 링크를 수집한다.

```
urls = []
for page in range(1, 251):
    res = requests.get(url.format(page))
    root = lxml.html.fromstring(res.text)
    for link in root.cssselect('a.f_nb'):
        urls.append(link.attrib['href'])
```

1297개의 링크가 수집되었다.

```
len(urls)
```

```
1297
```

기사 본문을 수집한다.

```
articles = []
for u in tqdm.tqdm_notebook(urls):
    if not u.startswith('http'):
        continue
    res = requests.get(u)
    root = lxml.html.fromstring(res.text)
    body = root.cssselect('.article_view').pop()
    content = body.text_content().strip()  # 본문을 가져와 앞뒤 공백을 제거
    articles.append(content)
```

엑셀 파일로 저장한다.

```
import pandas
df = pandas.DataFrame({'article': articles})
df.to_excel('daum_news_ai.xlsx')
```

엑셀 파일을 읽어오자. 엑셀 파일은 다음 링크에서 다운 받을 수 있다. http://doc.mindscale.kr/km/unstructured/daum_news_ai.xlsx

```
df = pandas.read_excel('daum_news_ai.xlsx')
```

```
df.head()
```

	article
0	Elon Musk, the visionary entrepreneur, fired a...
1	배달앱 '배달의민족', 프리미엄 외식 배달 서비스 '배민라이더스' 등을 운영하는 '...
2	[서울경제] 서울경제신문이 4차 산업혁명 시대에 필요한 인재 육성을 위한 교육혁신 ...
3	[경향신문] 인공지능 스피커를 몇 개 구해서 집과 사무실에 연결해 두었다. 친구 삼...
4	의료분야에도 4차 산업혁명 바람이 거세다. 최근 등장한 인공지능(AI)·빅데이터와 ...

8.2.2. TDM 만들기

형태소 분석기를 불러온다.

```
from konlpy.tag import Twitter

tagger = Twitter()
```

konlpy 설치에 문제가 있는 경우 월인을 사용한다.

```
from worin.tag import FeedForward

tagger = FeedForward()
```

2글자 이상인 명사만 추출하는 함수를 만든다.

```
def kor_noun(text):
    words = []
    for w in tagger.nouns(text):
        if len(w) > 1:
            words.append(w)
    return words
```

```
kor_noun('한글은 한국어를 표현하는 문자인 것이다.')
```

```
['한글', '한국어', '문자']
```

```
from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(tokenizer=ko_noun, max_features=1000)
```

기사를 파일에서 읽어들일 때 자동으로 숫자로 변환되는 경우가 있다. 강제로 문자열로 변환해준다.

```
articles = df['article'].astype(str)
```

TDM을 만든다.

```
tdm = cv.fit_transform(articles)

words = cv.get_feature_names()
```

많이 나온 단어들을 확인해본다.

```
sorted(zip(tdm.sum(axis=0).flat, words), reverse=True)[:20]
```

```
[(4664, '점수'),
 (3801, '투자'),
 (3748, '기록'),
 (2948, '종목'),
 (2700, '지능'),
 (2545, '인공'),
 (2396, '기자'),
 (2368, '기업'),
 (2083, '시장'),
 (1981, '진단'),
 (1940, '기술'),
 (1890, '업종'),
 (1768, '상위'),
 (1746, '한국'),
 (1513, '최근'),
 (1512, '상장'),
 (1494, '수익률'),
 (1466, '전체'),
 (1452, '라이온'),
 (1444, '그림')]
```

8.2.3. gensim 설치

토픽 모형을 돌리기 위해서는 gensim이 필요하다. gensim의 설치는 아나콘다일 경우 주피터 노트북에서 다음과 같이 입력한다.

```
!conda install -y gensim
```

colab을 쓰는 경우는 pip로 설치한다.

```
!pip install gensim
```

8.2.4. 데이터 형식 바꾸기

gensim에서는 고유한 데이터 형식을 쓰기 때문에 CountVectorizer로 만든 tdm을 다음과 같이 변환해주어야 한다.

```
from gensim.matutils import Sparse2Corpus

corpus = Sparse2Corpus(tdm.T)
```

```
C:\Users\user\Anaconda3\lib\site-packages\gensim\utils.py:855: UserWarning: detected Windows; aliasing chunkize to chunkize_serial
  warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

8.2.5. 분석

실제 분석은 다음과 같이 한다. num_topics로 주제의 개수를 지정해주어야 한다. 주제의 수는 문서의 양에 따라 적절히 넣어준다. 만약 분석 결과 비슷비슷한 주제가 너무 많다면 주제의 수를 크게 잡은 것이므로 줄여준다.

passes와 iterations는 계산량을 조절한다. 숫자가 적으면 gensim에서 경고를 보여준다. 그 경우 값을 키워준다. passes는 데이터가 적을 때 높여주면 효과적이다.

토픽 모형은 실행할 때마다 결과가 조금씩 다르게 나온다. random_state를 고정해주면 항상 같은 결과를 얻을 수 있다.

```
from gensim.models.ldamodel import LdaModel
lda = LdaModel(corpus=corpus, id2word=dict(enumerate(words)),
               num_topics=10, passes=10, iterations=50,
               random_state=1234)
```

주제별 단어 분포

10개의 주제로 학습을 시키면 0번부터 9번까지 주제가 생성된다. 주제의 순서나 번호에는 특별한 의미가 없다. show_topic 메소드로 주제를 확인하면 해당 주제의 단어 분포를 확인할 수 있다.

0번 주제는 점수, 기록, 투자, 종목 등의 단어가 포함되는 것으로 보아 주식 관련된 주제으로 짐작할 수 있다.

```
lda.show_topic(0)
```

```
[('점수', 0.06050807611089212),
 ('기록', 0.047183343674121414),
 ('투자', 0.04345185200020927),
 ('종목', 0.03773752731509858),
 ('진단', 0.0254030654819613 76),
 ('업종', 0.023931418436736032),
 ('상위', 0.02280177958526912),
 ('기업', 0.020875245260415617),
 ('상장', 0.019187434391780188),
 ('기자', 0.018910169415263748)]
```

1번 주제는 반도체 시장 관련된 주제로 보인다.

```
lda.show_topic(1)
```

```
[('반도체', 0.06718254613484592),
 ('시장', 0.03178420663611034),
 ('투자', 0.026654382193444983),
 ('지난해', 0.024130539634868393),
 ('호황', 0.021008793485102876),
 ('수출', 0.02082634234431819),
 ('수요', 0.01889418763090043),
 ('메모리', 0.014797125811305281),
 ('성장', 0.013627953691853807),
 ('비중', 0.013470502641745699)]
```

50번 문서를 보자.

```
articles[50]
```

```
'[서울경제] \n          삼성전자(005930) 갤럭시 스마트폰의 음악서비스인 '삼성 뮤직'을 삼성 스마트TV를 통해서도 이용할 수 있게 된다. \n          \n          삼성
```

tdm에서 i번째 행을 gensim 형식으로 바꿔주는 함수를 만들자.

```
def tdm2doc(tdm, i):
    doc = []
    for i, n in enumerate(tdm[i].toarray().flat):
        if n > 0:
            doc.append((i, n))
    return doc

doc = tdm2doc(tdm, 50)
```

단어 번호와 단어의 사용횟수의 짝으로 표현된다.

```
doc
```

```
[(29, 1),
 (48, 1),
 (51, 1),
 (97, 1),
 (130, 1),
 (133, 1),
 (139, 1),
 (270, 1),
 (389, 9),
 (397, 1),
 (408, 2),
 (409, 1),
 (463, 4),
 (464, 1),
 (523, 1),
 (544, 2),
 (618, 1),
 (628, 1),
 (631, 3),
 (659, 1),
 (661, 1),
 (727, 3),
 (746, 1),
 (794, 1),
 (852, 1),
 (857, 1),
 (878, 1),
 (925, 1)]
```

50번 문서의 주제 비율을 확인한다. 2번 주제가 22.8%, 4번 주제가 3.44%, 5번 주제가 32.58%, 6번 주제가 39.81% 포함되어 있다.

```
lda.get_document_topics(doc)
```

```
[(2, 0.2284575106133336),
 (4, 0.03449179334832947),
 (5, 0.32588335753238046),
 (6, 0.39812181552500403)]
```

주요 토픽을 한 번에 보려면 다음과 같이 한다.

```
for t in range(10):
    topic = [word for word, p in lda.show_topic(t)]
    print(t, ' '.join(topic))
```

```
0 점수 기록 투자 종목 진단 업종 상위 기업 상장 기자
1 반도체 시장 투자 지난해 호황 수출 수요 메모리 성장 비중
2 지능 인공 스마트 기술 시티 중국 통해 기자 시장 사업
3 서울 시민 서울시 창업 혁신 도시 교육 안철수 미래 기술
4 택시 서비스 호출 카카오 고객 국토부 배차 모빌리티 암웨이 유료
5 서비스 기술 체인 정보 블록 냉장고 기반 시스템 기능 추천
6 전자 삼성 분기 사업 시장 실적 투자 억원 영업 스마트폰
7 연구 무기 지능 로봇 카이스트 인공 시스템 개발 인간 기술
8 기업 부회장 삼성 정부 교육 지원 지역 사업 억원 창업
9 기술 센터 지능 인공 텔레콤 게임 데이터 개발 구글 미국
```