Chapter08. 텍스트 분석(8-6 ~ 8-9)

06. 토픽 모델링(Topic Modeling) - 20 뉴스그룹

- 토픽 모델링 : 문서 집합에 숨어 있는 주제를 찾아내는 것
- 머신러닝 기반의 토픽 모델링에 자주 사용되는 기법
  - LSA(Latent Semantic Analysis)
  - LDA(Latent Dirichlet Allocation) → 이 절에서 LDA만 수행
    - 토픽 모델링의 LDA는 앞선 차원 축소의 LDA와 약어만 같고 서로 다른 알고리즘이므로 유의 !

- 앞의 텍스트 분류에서 소개한 20 뉴스그룹 데이터 셋을 이용해 적용

```
['alt.atheism', 'comp.graphics', 'comp.os.ms-windows.misc', 'comp.sys.ibm.pc.hardware',
'comp.sys.mac.hardware', 'comp.windows.x', 'misc.forsale', 'rec.autos', 'rec.motorcycles',
'rec.sport.baseball', 'rec.sport.hockey', 'sci.crypt', 'sci.electronics', 'sci.med', 'sci.space',
'soc.religion.christian', 'talk.politics.guns', 'talk.politics.mideast', 'talk.politics.misc',
'talk.religion.misc']
```

- 사이킷런은 LDA 기반의 토픽 모델링을 LatentDirichletAllocation 클래스로 제공
  - n_components 파라미터로 토픽 개수 조정
- LatentDirichletAllocation(데이터 세트) 수행 → 객체는 components_속성값 가짐
  - components_ : 개별 토픽별로 각 word피처가 얼마나 많이 그 토픽에 할당됐는지에 대한 수치를 가짐. 높은 값일수록 해당 word 피처는 그 토픽의 중심 word

07. 문서 군집화 소개와 실습(Opinion Review 데이터 세트)

1) 문서 군집화 개념

> 💡 **문서 군집화(Document Clustering)**
>
> 비슷한 텍스트 구성의 문서를 군집화 하는 것

- 문서 군집화는 학습 데이터 세트가 필요 없는 비지도 학습 기반으로 동작
- 이전 장의 군집화 기법을 활용해 텍스트 기반의 문서 군집화를 적용할 예정

2) Opinion Review 데이터 세트를 이용한 문서 군집화 수행하기

- 각 파일 이름 자체만으로 의견(opinion)의 텍스트(text)가 어떠한 제품/서비스에 대한 리뷰인지 알 수 있음
- 문서를 TF-IDF 형태로 피처 벡터화
- 군집화 기법은 K-평균 적용
- 문서는 크게 전자제품, 자동차, 호텔로 돼있음
    - 전자제품은 다시 네비, 아이팟, 킨들, 랩탑 컴퓨터 등과 같은 세부 요소로 나뉨
- 5개의 중심 기반으로 군집화 하였을 때, 군집 개수가 약간 많게 설정돼 있어서 세분화 되어 군집화된 경향이 있음

  ⇒ 5 → 3개 그룹으로 군집화 다시 시도
- 3개로 군집화 하였을 때 지난번보다 더 군집화가 잘 되었음

3) 군집별 핵심 단어 추출하기

- 각 군집에 속한 문서는 핵심 단어를 주축으로 군집화 됨
  - 이번에는 각 군집을 구성하는 핵심 단어가 어떤 건지 확인

- KMeans 객체
  - clusters_centers_ : 각 군집을 구성하는 단어 피처가 군집의 중심을 기준으로 얼마나 가깝게 위치해있는지 나타내는 속성
    - 배열 값으로 제공, 행은 개별 군집, 열은 개별 피처를 의미
    - 각 배열 내의 값은 개별 군집 내의 상대 위치를 숫자 값으로 표현한 일종의 좌표 값
    - 넘파이의 ndarray
    - argsort()[:,::-1] : 배열 내 값이 큰 순으로 정렬된 위치 인덱스 값 반환
      - 핵심 단어 피처의 이름을 출력하기 위해 위치 인덱스 필요

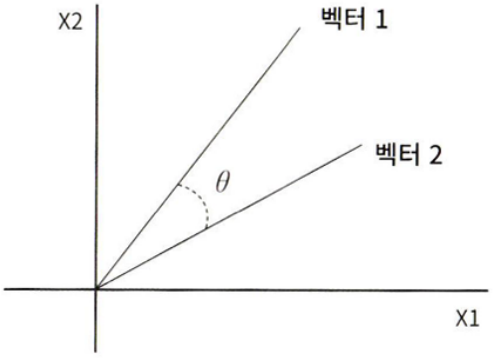
- 군집화 결과
  - Cluster #0 : 화면과 배터리 수명 등이 핵심 단어로 군집화
  - Cluster #1 : 방과 서비스 등이 핵심 단어로 군집화
  - Cluster #2 : 실내 인테리어, 좌석, 연료 효율 등이 핵심 단어로 군집화

08. 문서 유사도

1) 문서 유사도 측정 방법 - 코사인 유사도

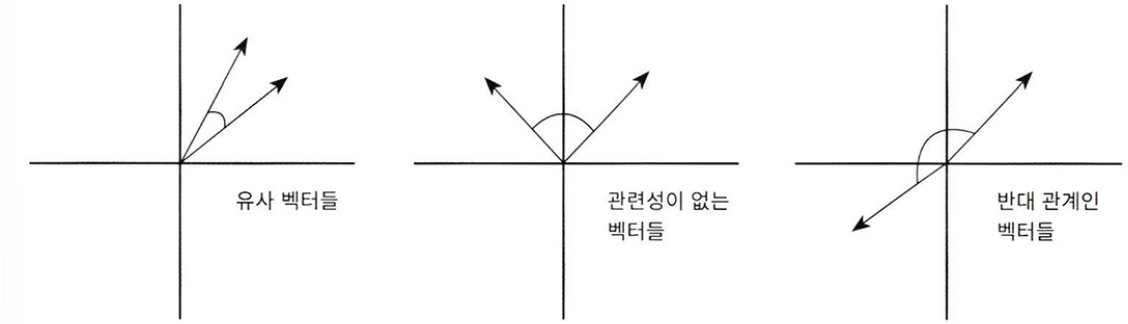
- 문서와 문서 간의 유사도 비교는 일반적으로 코사인 유사도(Cosine Similarity) 사용
  - 코사인 유사도는 벡터와 벡터 간의 유사도를 비교할 때, 벡터의 크기보다는 벡터의 상호 방향성이 얼마나 유사한지에 기반으로

⇒ 즉 코사인 유사도는 두 벡터 사이의 사잇각을 구해 얼마나 유사한지 수치로 적용



2) 두 벡터 사잇각

- 두 벡터의 사잇각에 따라 상호 관계는 다음과 같이 유사하거나 관련이 없거나 아예 반대 관계일 수 있음



- 두 벡터 A와 B의 코사인 값

$$A * B = \| A \| \| B \| cos\theta$$

두 벡터 A와 B의 내적 값은 두 벡터의 크기를 곱한 값의 코사인 각도 값을 곱한 것

⇒ 유사도 cos세타 = 두 벡터의 내적을 총 벡터 크기의 합으로 나눈 것

(즉, 내적 결과를 총 벡터 크기로 정규화)

$$\text{similarity} = cos\theta = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

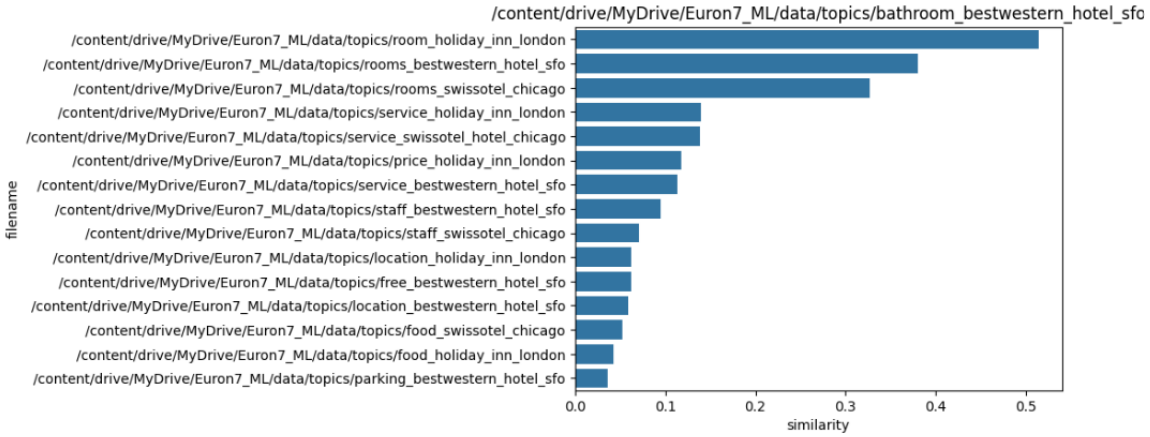
- 코사인 유사도가 문서의 유사도 비교에 많이 사용되는 이유
  - 문서를 피처 벡터화 변환하면 차원이 매우 많은 희소 행렬이 되기 쉬움
    - 이러한 희소 행렬 기반에서 문서와 문서 벡터 간의 크기에 기반한 유사도 지표는 정확도가 떨어지기 쉬움
  - 문서가 매우 긴 경우 단어의 빈도수가 더 많을 것이기 때문에 이러한 빈도수에만 기반하는 것은 공정한 비교 불가

- 사이킷런: 코사인 유사도 측정을 위해 sklearn.metrics.pairwise.cosine_similarity API 제공

3) Opinion Review 데이터 세트를 이용한 문서 유사도 측정

- 앞 절의 문서 군집화에서 사용한 Opinion Review 데이터 셋을 이용해 이들 문서 간의 유사도를 측정할 것임
  - 호텔을 주제로 군집화된 문서를 이용해 특정 문서와 다른 문서 간의 유사도를 알아볼 것임
- 호텔을 주제로 군집화된 데이터 추출 → 이 데이터에 해당하는 TfidfVectorizer의 데이터 추출
  - DataFrame객체 변수인 document_df에서 먼저 호텔로 군집화된 문서의 인덱스 추출
  - 이를 이용해 TfidfVectorizer 객체 변수인 feature_vect에서 호텔로 군집화된 문서의 피처 벡터 추출

- 첫 번째 문서와 다른 문서 간에 유사도가 높은 순으로 정렬 후 시각화



09. 한글 텍스트 처리 - 네이버 영화 평점 감성 분석

이번 절에서는 네이버 영화 평점 데이터를 기반으로 감성 분석을 적용할 것임 !

1) 한글 NLP 처리의 어려움

- 일반적으로 한글 언어 처리는 영어 등의 라틴어 처리보다 어려움
  - 주된 원인 : 띄어쓰기, 다양한 조사
    - 띄어쓰기: 한글 같은 경우 띄어쓰기를 잘못하면 의미가 왜곡되어 전달될 수 있지만 영어는 잘못된 또는 없는 단어로 인식
    - 조사: 주어나 목적어를 위해 추가되며, 경우의 수가 많기 때문에 어근 추출(Stemming/Lemmatization)등의 전처리 시 제거가 까다로움

2) KoNLPy 소개

> 💡 **KoNLPy**
>
> 파이썬의 대표적인 한글 형태소 패키지

- 형태소의 사전적 의미 : 단어로서 의미를 가지는 최소 단위
- 형태소 분석 : 말뭉치를 이러한 형태소 어근 단위로 쪼개고 각 형태소에 품사 태깅을 부착하는 작업을 지칭

- KoNLPy는 C/C++, Java로 잘 만들어진 한글 형태소 엔진을 파이썬 래퍼 기반으로 재작성한 패키지
  - 기존의 엔진은 그대로 유지한 채 파이썬 기반에서 인터페이스 제공 → 검증된 패키지의 안정성 유지
- 꼬꼬마(Kkma), 한나눔(Hannanum), Komoran, 은전한닢 프로젝트(Mecab), Twitter와 같이 5개의 형태소 분석 모듈 사용 가능
  - Mecab은 윈도우 환경에서 구동X, 리눅스 환경에서 가능

3) 데이터 로딩

- 학습 데이터의 라벨 값 비율을 살펴보았을 때 균등한 분포를 지님
- train_df의 경우 리뷰 텍스트를 가지는 document칼럼에 Null이 일부 존재 → 공백 변환
- 문자가 아닌 숫자의 경우 → 단어적인 의미로 부족하므로 파이썬의 정규 표현식 모듈인 re이용하여 공백 변환
- 각 문장을 한글 형태소 분석을 통해 형태소 단어로 토큰화
  - SNS분석에 적합한 Twitter 클래스 이용
  - Twitter 객테의 morphs() 메서드 이용 → 입력 인자로 들어온 문장을 형태소 단어 형태로 토큰화해 list객체로 반환
- 사이킷런의 TfidfVectorizer를 이용해 TF-IDF 피처 모델 생성 → 로지스틱 회귀를 이용해 분류 기반의 감성 분석 수행 → 테스트 세트를 이용해 최종 감성 분석 예측 수행

- 테스트 세트를 이용해 예측할 때, 학습할 때 적용한 TfidfVectorizer를 그대로 적용

  (학습 시 설정된 TfidfVectorizer의 피처 개수와 테스트 데이터를 TfidfVectorizer로 변환할 피처 개수가 같아짐)