텍스트 분석 (2)

8.6 토픽 모델링

개념

- 문서 집합에 숨어 있는 주제 찾는 것

종류

- LDA (Latent Dirichlet Allocation)
- LSA (Latent Semantic Analysis)

LDA

1. categories 파라미터를 통해 필요한 주제만 필터링해 추출하고 추출한 텍스트를 count 기반으로 벡터화 변환
2. n_components 파라미터를 이용해 토픽 개수 조정
3. 개별 토픽별로 각 word 피처가 얼마나 많은 그 토픽에 할당됐는지에 대한 수치를 가지고 있음

→ 높은 값일 수록 해당 word 피처는 그 토픽의 중심 word 가 된다.

8.7 문서 군집화 소개와 실습

개념

- 비슷한 텍스트 구성의 문서를 군집화 하는 것
- 텍스트 분류 기반의 문서 분류와 비교
    - 공통점:

문서 군집화는 동일한 군집에 속하는 문서를 같은 카테고리 소속으로 분류할 수 있음

- 차이점:

  문서 군집화- 학습 데이터가 필요 없는 비지도 학습

  텍스트 분류 기반의 문서 분류- 사전에 결정 카테고리 값을 가진 학습 데이터 세트 필요

Opinion Review 데이터 세트를 이용한 문서 군집화 수행

👉 여러 개의 파일을 한 개의 Data Frame으로 로딩해 데이터 처리

- 해당 디렉터리 내에 있는 파일을 하나씩 읽어서 파일명과 파일 리뷰를 하나의 DataFrame 으로 로드
- 데이터를 로드할 절대 경로 디렉터리를 먼저 지정한 뒤, 이 위치에서 데이터를 로딩

- 해당 디렉터리 내의 모든 파일에 대해 각각 for 반복문으로 반복하면서 개별 파일명을 파일명 리스트에 추가
- 개별 파일은 DataFrame 으로 읽은 후 다시 문자열로 반환한 뒤 파일 내용 리스트에 추가
- 파일명 리스트와 파일 내용 리스트를 이용해 새롭게 파일명과 파일 내용을 칼럼으로 가지는 DataFrame 생성

👉 TF-IDF 형태로 피처 벡터화

TfidfVectorizer 의 fit_transform() 인자로 document_df DataFrame 의 opinion_text 칼럼 입력하면 개별 문서 텍스트에 대해 TF-IDF 변환된 피처 벡터화된 행렬 구할 수 있음

군집별 핵심 단어 추출하기

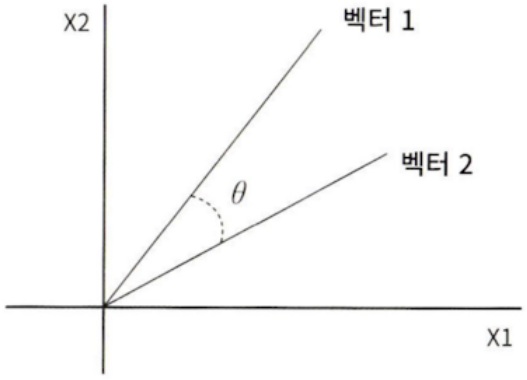
- KMeans 객체는 각 군집을 구성하는 단어 피처가 군집의 중심을 기준으로 얼마나 가깝게 위치해 있는지 clusters_centers_ 라는 속성으로 제공
- cluster_centers_ 배열
  - 행: 개별 군집

    - 열: 개별 피처
    - 각 배열 내의 값은 개별 군집 내의 상대 위치를 숫자 값으로 표현한 좌표 값

- get_cluster_details()
    - cluster_centers 배열 내 값이 큰 순으로 정렬된 위치 인덱스 값을 반환
    - 가장 값이 큰 데이터의 위치 인덱스를 추출한 뒤, 해당 인덱스를 이용해 핵심 단어 이름과 그때의 상대 위치 값을 추출
    - cluster_details 라는 Dict 객체 변수에 기록하고 반환
    - 개별 군집번호, 핵심 단어, 핵심 단어 중심 위치 상댓값, 파일명 속성 값 정보가 있음

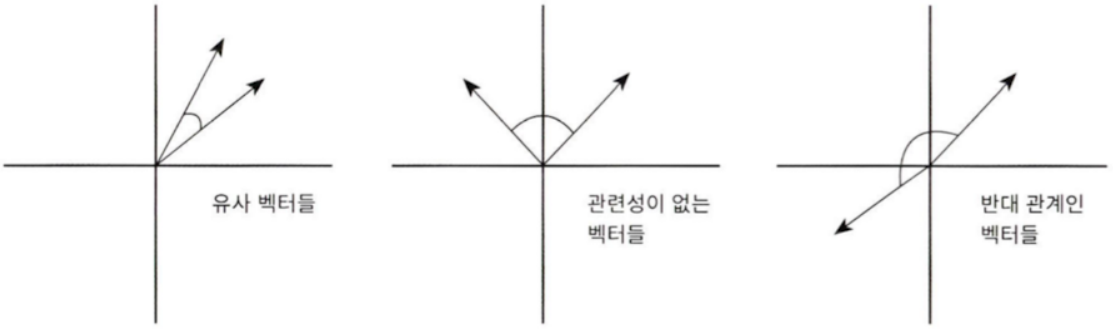
8.8 문서 유사도

- 문서와 문서 간의 유사도 비교는 일반적으로 코사인 유사도 사용
- 벡터와 벡터 간의 유사도를 비교할 때 벡터의 크기보다는 벡터의 상호방향성이 얼마나 유사한지에 기반
- 두 벡터 사이의 사잇각을 구해서 얼마나 유사한지 수치로 적용

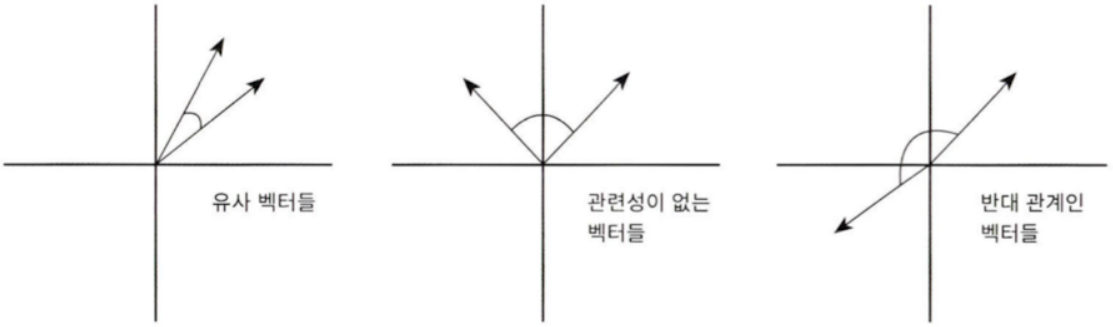


두 벡터 사잇각

- 두 벡터의 사잇각 상호관계



- 두 벡터의 내적 값



- 유사도

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

- 문서를 피처 벡터화 변환하면 차원이 매우 많은 희소 행렬이 되기 쉽다.
- 이러한 희소 행렬 기반에서 문서와 문서 벡터간의 크기에 기반한 유사도 자료는 정확도가 떨어짐
- 문서가 매우 긴 경우에는 단어의 빈도수도 더 많기 때문에 빈도수에만 기반해서는 공정한 비교를 할 수 없음

Opinion Review 데이터 세트를 이용한 문서 유사도 측정

- 문서를 피처 벡터화해 변환하면 문서 내 단어에 출현 빈도와 같은 값을 부여해 각 문서가 단어 피처의 값으로 벡터화된다.
- cosine_simularity() 이용

- 군집화된 데이터를 먼저 추출하고 데이터에 해당하는 TfidVectorizer 데이터를 추출
- cosine_similarity()는 쌍 형태의 ndarray 를 반환하므로 이를 판다스 인덱스로 이용하기 위해 reshape(-1) 로 차원 변경

8.9 한글 텍스트 처리

한글 NLP 처리의 어려움

- 띄어쓰기
- 조사
  - 어근 추출 등의 전처리 시 제거하기 까다롭다.

형태소 분석

- 말뭉치를 형태소 어근 단위로 쪼개고 각 형태소에 품사 태깅 (POS tagging)을 부착
- KoNLPy : 파이썬의 한글 형태소 패키지

네이버 영화 평점 감성 분석

- Null 값이 있으면 공백으로 변환
- 숫자의 경우에도 정규 표현식 모듈 re를 이용해 공백으로 변환

👉 TF-IDF 방식으로 단어를 벡터화

- 각 문장을 한글 형태소 분석을 통해 형태소 단어로 토큰화
- Twitter 클래스 : SNS 분석에 적합

    - morphs() 메서드를 이용하면 입력 인자로 들어온 문장을 형태소 단어 형태로 토큰화해 list 객체로 변환
- tokenizer 함수: 문장을 형태소 단어 형태로 반환

👉 로지스틱 회귀

- 분류 기반 감성 분석
- GridSearchCV