8장 텍스트 분석

NLP: 머신이 인간의 언어를 이해하고 해석하는 데 더 중점을 두고 기술이 발전
텍스트 마이닝:텍스트 분석은 비정형 텍스트에서 의미 있는 정보를 추출하는 것에 더 중점을 두고 기술이 발전

NLP에는 언어 해석을 위한 기계 번역, 자동으로 질문을 해석하고 답하는 질의응답 시스템
NLP는 텍스트 분석을 향상하게 하는 기반 기술

텍스트 분석은 머신러닝, 언어 이해, 통계 등 활용해 모델 수립하고 정보를 추출해서 비즈니스 인텔리전스나 예측 분석 등의 분석 작업을 주로 수행

- 텍스트 분류: 문서가 특정 분류 또는 카테고리에 속하는 것을 예측하는 기법
  예) 특정 신문 기사 내용이 어떤 카테고리에 속하는지 자동 분류, 스팸 메일 검출 프로그램
- 감성 분석: 텍스트에서 나타나는 감정/판단/믿음/의견/기분 등 주관적 요소를 분석하는 기법 총칭
  예)소셜 미디어 감정 분석 영화나 제품에 대한 긍정 또는 리뷰, 여론조사 의견 분석
- 텍스트 요약: 텍스트 내 중요 주제나 중심 사상을 추출하는 기법 예)토픽 모델링
- 텍스트 군집화와 유사도 측정: 비슷한 유형의 문서에 대해 군집화를 수행하는 기법, 문서들 간 유사도 측정해 비슷한 문서끼리 모을 수 있는 방법

01 텍스트 분석 이해

비정형 텍스트 데이터를 어떻게 피처 형태로 추출하고 추출된 피처에 의미 있는 값을 부여하는가 하는 것이 매우 중요한 요소
텍스트를 word 기반의 다수 피처로 추출하고 이 피처에 단어 빈도수와 같은 숫자 값을 부여하면 텍스트는 단어의 조합인 벡터값으로 표현-> 이렇게 텍스트를 변환하는 것을 피처 벡터화(피처 추출)이라고 함
텍스트를 피처 벡터화해서 변환하는 방법: BOW, Word2Vec

텍스트 분석 수행 프로세스

1. 텍스트 사전 준비작업(텍스트 전처리): 텍스트를 피처로 만들기 전 대/소문자 변경, 특수문자 삭제 등 클렌징 작업, 단어 등의 토큰화 작업, 의미 없는 단어 제거 작업, 어근 추출 등 텍스트 정규화 작업 수행

2. 피처 벡터화/추출: 사전 준비 작업으로 가공된 텍스트에서 피처 추출 후 벡터 값 할당
   대표적 방법으로 BOW와 Word2Vec이 있으며 BOW는 대표적으로 Count 기반과 TF-IDF 기반 벡터화가 있음
3. ML 모델 수립 및 학습/예측/평가: 피처 벡터화된 데이터 세트에 ML모델을 적용해 학습/예측 및 평가를 수행

파이썬 기반의 NLP, 텍스트 분석 패키지

- NLTK: 가장 대표적인 NLP 패키지, 거의 모든 NLP 영역 커버, 수행 속도 측면에서 아쉬운 부분
- Gensim: 토픽 모델링 분야에서 두각 나타냄, Word2Vec 구현
- spaCy: 뛰어난 수행 성능으로 가장 주목 받는 패키지

02 텍스트 사전 준비 작업(텍스트 전처리)- 텍스트 정규화

텍스트 정규화는 클렌징, 정제, 토큰화, 어근화 등의 다양한 텍스트 데이터의 사전 작업을 수행하는 것 의미

클렌징

불필요한 문자, 기호 등을 사전에 제거
예)HTML, XML 태그나 특정 기호 등 제거

텍스트 토큰화

- 문장 토큰화: 문서에서 문장을 분리하는 토큰화
  문장의 마침표, 개행문자 등 문장 마지막 뜻하는 기호에 따라 분리/정규 표현식
- 단어 토큰화: 문장을 단어로 토큰화함
  공백, 콤마, 마침표, 개행문자 등으로 단어를 분리

n-gram은 연속된 n개의 단어를 하나의 토큰화 단위로 분리해 내는 것

스톱 워드 제거

분석에 큰 의미가 없는 단어를 지칭
예)is, the, a, will 등 문장 구성하는 필수 문법 요소지만 문맥적으로 큰 의미가 없는 단어 해당

Stemming과 Lemmatization

문법적으로 또는 의미적으로 변화하는 단어의 원형을 찾는 것
Lemmatization이 Stemming보다 정교하며 의미론적인 기반에서 단어의 원형을 찾음
Stemming은 원형 단어로 변환시 일반적인 방법을 적용하거나 더 단순화된 방법을 적용해 원래 단어에서 일부 철자가 훼손된 어근 단어를 추출하는 경향
Lemmatization은 품사와 같은 문법적 요소와 더 의미적인 부분 감안해 정확한 철자로 된 어근 단어 찾아줌(더 오랜 시간 소요)
Lemmatization은 보다 정확한 원형 단어 추출 위해 단어의 품사 입력해줘야함

03 Bag of Words - BOW

BOW: 문서가 가지는 모든 단어를 문맥이나 순서를 무시하고 일괄적으로 단어에 대해 빈도 값 부여해 피처 값을 추출하는 모델

1. 각 문장에 대해 모든 단어에서 중복을 제거하고 각 단어를 칼럼 형태로 나열
   각 단어에 고유 인덱스를 부여
2. 개별 문장에서 해당 단어가 나타나는 횟수를 각 단어에 기재

장점: 쉽고 빠른 구축
단점:문맥의미 반영 부족, 희소 행렬 문제

- 문맥 의미 반영 부족: 단어 순서를 고려하지 않기 때문에 문장 내에서 단어의 문맥적인 의미가 무시됨
- 희소 행렬 문제: 매우 많은 단어가 칼럼으로 만들어져 희소 행렬 형태의 데이터 세트가 만들어지기 쉬움
  희소행렬: 대규모의 칼럼으로 구성된 행렬에서 대부분의 값이 0으로 채워지는 행렬
  밀집 행렬: 대부분의 값이 0이 아닌 의미 있는 값으로 채워진 행렬

BOW 피처 벡터화

모든 문서에서 모든 단어를 칼럼 형태로 나열하고 각 문서에서 해당 단어의 횟수나 정규화된 빈도를 값으로 부여하는 데이터 세트 모델로 변경하는 것

두 가지 방식 존재

- 카운트 기반 벡터화: 단어 피처에 값을 부여할 때 각 문서에서 해당 단어가 나타나는 횟수, 즉 count를 부여하는 경우

카운트 값이 높을수록 중요 단어로 인식
언어 특성상 문장에서 자주 사용될 수 밖에 없는 단어까지 높은 값 부여

- TF-IDF 기반 벡터화: 개별 문서에서 자주 나타나는 단어에 높은 가중치를 주되, 모든 문서에서 전반적으로 자주 나타나는 단어에 대해서는 페널티 값을 부여하는 방식으로 단어에 대한 가중치의 균형을 맞춤

사이킷런의 count 및 TF-IDF 벡터화 구현: CountVectorizer, TfidfVectorizer

CountVectorizer 클래스는 카운트 기반의 벡터화 구현한 클래스
텍스트 전처리도 함께 수행

<파라미터>

- max_df: 전체 문서에 걸쳐 너무 높은 빈도수 가지는 단어 피처를 제외하기 위한 파라미터
- min_df: 전체 문서에 걸쳐 너무 낮은 빈도수를 가지는 단어 피처를 제외하기 위한 파라미터
- max_features: 추출하는 피처의 개수를 제한
- stop_words: 스톱 워드로 지정된 단어는 추출에서 제외
- n_gram_range: n_gram 범위 설정, 튜플 형태로 (범위 최솟값, 범위 최댓값) 지정
- analyzer: 피처 추출을 수행한 단위를 지정, 디폴트는 'word'
- token_pattern: 정규 표현식 패턴 지정
- tokenizer: 토큰화를 별도의 커스텀 함수로 이용시 적용

1. 사전데이터 가공: 모든 문자를 소문자로 변환 하는 등
2. 토큰화: n_gram_range 반영
3. 텍스트 정규화: stop_words 필터링, tokenizer 파라미터에 커스텀 어근 변환 함수 적용해 어근 변환 수행
4. 피처 벡터화: max_df, min_df, max_features 등 파라미터 이용해 토큰화된 단어를 피처로 추출하고 단어 빈도수 벡터 값 적용

BOW 벡터화를 위한 희소 행렬

BOW 형태를 가진 언어 모델의 피처 벡터화는 대부분 희소 행렬
메모리 공간이 많이 필요, 연산 시에도 데이터 액세스를 위한 시간이 많이 소모
물리적으로 적은 메모리 공간을 차지할 수 있도록 변환-> 대표적으로 COO, CSR 형식(많이 사용)

희소행렬- COO 형식

COO 형식: 0이 아닌 데이터만 별도의 데이터 배열에 저장하고 그 데이터가 가리키는 행과 열의 취리를 별도의 배열로 저장하는 방식
주로 사이파이(Scipy)이용, 사이파이의 sparse 패키지는 희소 행렬 변환을 위한 다양한 모듈 제공

희소행렬- CSR 형식

CSR 형식:행 위치 배열 내에 있는 고유한 값의 시작 위치만 다시 별도의 위치 배열로 가지는 변환 방식
COO 형식이 행과 열의 위치를 나타내기 위해서 반복적인 위치 데이터를 사용해야 하는 문제점 해결한 방식
COO 방식보다 메모리가 적게 들고 빠른 연산이 가능
csr_matrix 클래스 이용

04 텍스트 분류 실습 - 20 뉴스그룹 분류

희소 행렬에 분류를 효과적으로 잘 처리할 수 있는 알고리즘은 로지스틱 회귀, 선형 서포트 벡터 머신, 나이브 베이즈 등
로지스틱 회귀 이용해 분류 수행

텍스트 정규화

fetch_20newgroups()
파이썬 딕셔너리와 유사한 Bunch 객체 반환
내용 제외하고 제목 등 다른 정보 제거->순수한 텍스트만으로 구성된 기사 내용으로 어떤 뉴스그룹에 속하는 지 분류할 것
remove 파라미터 이용하면 뉴스그룹 기사의 헤더, 푸터 등 제거
subset 파라미터를 이용해 학습 데이터 세트와 테스트 데이터 세트를 분리해 내려받을 수 있음

피처 벡터화 변환과 머신러닝 모델 학습/예측/평가

테스트 데이터에서 CountVecotrizer를 적용할 때는 반드시 학습 데이터를 이용해 fit()이 수행된 CountVectorizer 객체를 이용해 테스트 데이터를 변환해야함
그래야 학습 시 설정된 CountVectorizer의 피처 개수와 테스트 데이터를 CountVectorizer로 변환할 피처 개수가 같아짐
테스트 데이터의 피처 벡터화는 학습 데이터에 사용된 CountVectorizer 객체 변수인 cnt_vect.transform() 이용해 변환

테스트 데이터의 피처 벡터화 시 fit_transform()을 사용하면 안됨
CountVectorizer.fit_transform(테스트 데이터)을 테스트 데이터 세트에 적용하면 테스트 데이터 기반으로 다시 CountVectorizer가 fit()을 수행하고 transform()하기 때문에 학습 시 사용된 피처 개수와 예측 시 사용할 피처 개수가 달라짐

TR-IDF가 단순 카운트 기반보다 훨씬 높은 예측 정확도 제공
TfidfVectorizer 클래스의 스톱 워드를 기존 'None'에서 'english'로 변경하고, ngram_range는 기존 (1,1)에서 (1,2)로, max_df=300으로 변경한 뒤 다시 예측 성능 측정
GridSearchCV로 로지스틱 회귀의 하이퍼 파라미터 최적화 수행

사이킷런 파이프라인 사용 및 GridSearchCV와의 결합

Pipeline 클래스를 이용하면 피처 벡터화와 ML 알고리즘 학습/예측을 위한 코드 작성을 한번에 진행 가능
모든 데이터 전처리 작업과 estimator 결합 가능
pipeline에 기술된 각각 객체 변수에 언더바 2개 연달아 붙여 gridsearchCV에 사용될 파라미터/하이퍼 파라미터 이름과 값 설정

05 감성분석

감성 분석 소개

감성 분석: 문서의 주관적인 감성/의견/감정/기분 등을 파악하기 위한 방법으로 소셜 미디어, 여론 조사, 온라인 리뷰, 피드백 등 다양한 분야에서 활용
문서 내 텍스트가 나타내는 여러 가지 주관적 단어와 문맥 기반으로 감성 수치를 계산하는 방법 이용
긍정 감성 지수와 부정 감성 지수로 구성

지도학습: 학습 데이터와 타깃 레이블 값을 기반으로 감성 분석 학습을 수행한 뒤 이를 기반으로 다른 데이터의 감성 분석을 예측하는 방법으로 일반적인 텍스트 기반의 분류와 거의 동일
비지도학습: 'lexicon'이라는 일종의 감성 어휘 사전 이용

지도학습 기반 감성 분석 실습- IMDB 영화평

지도 학습 기반 감성 분석은 텍스트 기반의 이진 분류

labeledTrainData.csv 데이터 이용
-id: 각 데이터 id
-sentiment: 1은 긍정적 평가, 0은 부정적 평가
-review: 영화평의 텍스트

HTML 태그 삭제
영어가 아닌 숫자, 특수문자 공란으로 변경
결정 값 클래스인 sentiment 칼럼을 별도로 추출해 결정 값 데이터 세트를 만들고 원본 데이터 세트에서 id와 sentiment 칼럼 삭제해 피처 데이터 세트 생성
train_test_split() 이용해 학습용과 테스트용 데이터 세트로 분리
감상평 텍스트를 피처 벡터화한 후 ML 분류 알고리즘 적용해 예측 성능을 측정
count 벡터화 적용해 예측 성능을 측정하고 TF-IDF 벡터화 적용

비지도 학습 기반 감성 분석 소개

비지도 감성 분석은 Lexicon을 기반
많은 감성 분석용 데이터는 결정된 레이블 값 가지고 있지 않음
lexicon: 감성만을 분석하기 위해 지원하는 감성 어휘 사전 의미
감성 사전은 긍정 감성 또는 부정 감성의 정도를 의미하는 수치를 가지고 있으며 이를 감성 지수라고 함
감성 지수는 단어 위치나 주변 단어, 문맥, POS 등 참고해 결정
감성 사전을 구현한 대표격은 NLTK 패키지
NLP에서 제공하는 WordNet 모듈은 방대한 영어 어휘 사전, 시맨틱 분석을 제공하는 어휘 사전(시맨틱은 '문맥상 의미'를 뜻함)
WordNet은 다양한 상황에서 같은 어휘라도 다르게 사용되는 어휘의 시맨틱 정보를 제공하며 각각의 품사로 구성된 개별 단어를 Synset이라는 개념 이용해 표현
Synset은 단순한 하나의 단어가 아니라 그 단어가 가지는 문맥, 시맨틱 정보를 제공하는 WordNet의 핵심개념

<NLTK를 포함한 대표적인 감성 사전>

- SentiWordNet: 감성 단어 전용의 WordNet 구현, WordNet의 Synset 개념을 감성 분석에 적용한 것, WordNet의 Synset 별로 3가지 감성 점수를 할당(긍정 감성 지수, 부정 감성 지수, 객관성 지수- 긍정/부정 감성 지수와 완전히 반대되는 개념으로 단어가 감성과 관계없이 얼마나 객관적인지를 수치로 나타낸 것)
- VADER: 주로 소셜 미디어의 텍스트에 대한 감성 분석을 제공하기 위한 패키지
- Pattern: 예측 성능 측면에서 가장 주목 받는 패키지

SentiWordNet을 이용한 감성 분석

WordNet의 synset()는 파라미터로 지정된 단어에 대해 WordNet에 등재된 모든 Synset 객체 반환
Synset은 POS, 정의, 부명제 등으로 시맨틱적인 요소를 표현
parth_similarity(): 어떤 어휘와 다른 어휘 간의 관계를 유사도로 나타낼 수 있음

SentiWordNet은 WordNet의 Synset과 유사한 Senti_Synset 클래스를 가짐
senti_synsets()는 WordNet 모듈이라서 synset()와 비슷하게 Senti_Synset 클래스를 리스트 형태로 반환
SentiSynset 객체는 단어의 감성을 나타내는 감성 지수와 객관성을 나타내는 객관성 지수 가짐(어떤 단어가 전혀 감성적이지 않으면 객관성 지수는 1, 감성 지수는 0이됨)

<IMDB 영화 감상평 감성 분석을 SentiWordNet Lexicon 기반으로 수행>

1. 문서를 문장 단위로 분해
2. 다시 문장을 단어 단위로 토큰화하고 품사 태깅
3. 품사 태깅된 단어 기반으로 synset 객체와 senti_synset 객체 생성
4. Senti_synset에서 긍정 감성/부정 감성 지수 구하고 이를 모두 합산해 특정 임계치 값 이상일 때 긍정 감성으로, 그렇지 않을 때 부정 감성으로 결정

WordNet 이용해 문서를 다시 단어로 토큰화한 뒤 어근 추출과 품사 태킹을 적용
SentiSynset 클래스 생성 후 Polarity Score을 합산하는 함수 생성(단어의 긍정 감성 지수와 부정 감성 지수 모두 합한 총 감성 지수가 0 이상일 경우 긍정 감성, 그렇지 않을 경우 부정 감성으로 예측)
swn_polarity(text) 함수를 IMDB 감상평의 개별 문서에 적용해 긍정 및 부정 감성을 예측
실제 감성 평가인 'sentiment' 칼럼과 swn_polarity(text)로 반환된 결과의 정확도, 정밀도, 재현율 값을 모두 측정

SentiWordNet은 WordNet의 하위 모듈로서 감성 분석을 위한 다양한 프레임워크 제공

VADER을 이용한 감성 분석

VADER는 소셜 미디어의 감성 분석 용도로 만들어진 룰 기반의 Lexicon
SentimentIntensityAnalyzer 클래스를 이용해 쉽게 감성 분석 제공
VADER는 NLTK 패키지의 서브 모듈로 제공될 수도 있고 단독 패키지로 제공 될 수도 있음

SentimentIntensityAnalyzer 객체 생성 뒤 문서별로 polarity_scores() 메서드 호출해 감성 점수 구하고 특정 임계값 이상이면 긍정, 그렇지 않으면 부정으로 판단
'neg'는 부정 감성 지수, 'neu'는 중립적인 감성 지수,'pos'는 긍정 감성 지수, compound는 neg, neu, pos score을 적절히 조합해 -1에서 1사이의 감성 지수를 표현한 값

compound score 기반으로 부정 감성 또는 긍정 감성 여부 결정
보통 0.1이상이면 긍정, 그 이하는 부정 감성으로 판단

vader_polarity() 함수는 입력 파라미터로 영화 감상평 텍스트와 긍정/부정을 결정하는 임곗값을 가지고, SentimentIntensityAnalyzer 객체의 polarity_scores() 메서드 호출해 감성 결과를 반환
저장된 감성 분석 결과 기반으로 VADER 예측 성능 측정
정확도는 SentiWordNet 보다 향상

06 토픽 모델링(Topic Modeling)- 20 뉴스그룹

토픽 모델링: 문서 집합에 숨어 있는 주제를 찾아내는 것
자주 사용되는 기법: LSA, LDA(차원 축소의 LDA와 다름)

LDA 기반의 토픽 모델링 LatentDirichletAllocation 클래스로 제공
fetch_20newsgroups() API는 categories 파라미터를 통해 필요한 주제만 필터링해 추출하고 추출한 텍스트를 count 기반 벡터화 변환(LDA는 count 기반의 벡터화만 사용)
피처 벡터화된 데이터 세트를 기반으로L LDA 토픽 모델링 수행
n_components 파라미터 이용해 토픽 개수 조정
fit을 수행하면 개별 토픽별로 각 word 피처가 얼마나 많이 그 토픽에 할당됐는지에 대한 수치인 components_ 속성값을 가짐