

# 2023.02.06 미팅 자료

TF-IDF, KeyBERT, SentiBERT, Crawling

팀원 : 배한성, 김은서, 조현아, 조유진

## CONTENTS

1. TF-IDF
2. KeyBERT
3. SentiBERT
4. Crawling

# TF-IDF

## TF-IDF란?

- 정보 검색과 텍스트 마이닝에서 이용하는 가중치
- 여러 문서로 이루어진 문서군이 있을 때 어떤 단어가 특정 문서 내에서 얼마나 중요한 것인지를 나타내는 통계적 수치
- 문서의 핵심어 추출, 검색 엔진에서 검색 결과의 순위 결정, 문서들 사이의 비슷한 정도를 구하는 용도로 사용
- TF(Term Frequency)
  - 특정한 단어가 문서 내에 얼마나 자주 등장하는지를 나타내는 값
- IDF(Inverse Document Frequency)
  - '특정 단어가 등장하는 문서의 수(DF)'의 역수
- TF-IDF
  - $TF * IDF$

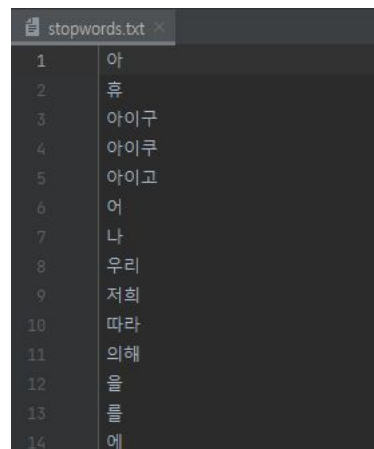
## 주요 라이브러리

- scikit-learn
  - 파이썬 프로그래밍 언어용 자유 소프트웨어 기계 학습 라이브러리
  - TfidfVectorizer 클래스 사용
    - 문서를 TF-IDF의 feature matrix (특성 행렬)로 변환하는 클래스
- konlpy
  - 한글 형태소 분석기
  - 문장에서 명사, 동사, 형용사만을 추출할 때 사용

## TF-IDF 테스트

### 과정

1. 데이터 전처리
  - a. 한글, 숫자, 영어 빼고 전부 제거
  - b. 불용어 제거 (불용어 파일 이용)
  - c. 명사, 동사, 형용사 추출
2. tf-idf 구하기



stopwords.txt	
1	아
2	휴
3	아이구
4	아이쿠
5	아이고
6	어
7	나
8	우리
9	저희
10	따라
11	의해
12	을
13	를
14	예

# TF-IDF 테스트

- 테스트 데이터

```
[ ] corpus = ["경소에 10년 된 노트북을 사용 중이었는데 고구마를 한 뭇 잎 먹은것처럼 너무 느리고 답답하여 노트북 구매를 위해 검색을 하던 중 삼성 노트북 플러스2 라는 제품을 발견하게  
" 대학 입학때 그냥저냥 쓴 제품 구매해서 사용하다가 이전 성능의 부족함을 느껴 가성비비 제품을 알아보면 중 삼성플러스2를 알게되었고 상품평과 후기들을 보고 이제점으로 구매하  
" 문서작업은 평소에 많이 하는 편이지만, 컴퓨터에 대한 지식은 많지 않은 사람입니다! 무언가 구매하기 전에도 엄청나게 이것저것 따져보고 사는 사람입니다!<br><br>자택용, 사  
" 기본가격으로사서 오기<em>전까진 용량을 너무작은걸났나 조금걱정하긴했는데</em> 일반 웹서핑하고 서류정리할정도로만 산거라 괜찮은거같아요 "" <em>가격대비 너무 만족하고  
" 7년 넘게 사용한 노트북이 하늘나리로 가서 새로운 노트북을 사게 됐어요. 가성비를 중요하게 생각하는 편이라 수리를 잘 <em>받을 수 있는 브랜드제품으로 저렴한 걸 찾고</em>  
" <em>백배도 안전하게와서 다행입니다</em> 전자제품이다보니 포장이 꼼꼼하게 잘 되어서 왔네요~ 문자마자 와~ <em>색상도 알리되었고 고급스러워서</em> <em>자 마음에 들  
기존 노트북을 15년정도 사용했습니다. 그래서 언제 갑자기 노트북이 안켜질지 모르겠다는 생각에 항상 불안 했었는데~<br><br>노트북을 구매하려고 앞지 삼성 주변에 대리점들  
" 내도 내산 후기<br>최근의 인터넷 광의를 돌으면서<br>휴대폰은 너무 작다는 생각이 들더라고요<br>그렇다고 태블릿으로 하기에는 <br><em>가격보다 가격이 많이 없고</em><br><em>  
" 인관을 돌거나 약간의 문서작업을 하기 위해서 9년프를 삼성노트북을 구매했다가 수명이 다해서 이리저리 둘러보면 중 이 제품을 발견하게 돼서 바로 구매했어요!!<br><br><em>  
" 50대 초반 사이버대학에 입학하여 동영상강의신청을 했습니다. 사무실에서 주로 컴퓨터를 사용해서 집에서 는 딱히 별로 사용하지 않았는데 학업을 시작하니 컴퓨터의 상태를 알  
" 2016년도 구매한 노트북이 좀 느려지기도하고 간단한 작업만 하는것이라서 고사양의 노트북이 필요한것은 아니라서 가성비가 좋은 노트북을 찾아보고 있었는데 마침 저렴한 가  
" 5년 전에 산 노트북을 켜려마다, 부팅시간이 최소 30분이었어오! π.π<br>인고의 시간을 버려야만, 속타지는 10분간의 컴퓨터 사용이 가능했조! <em>사</em><br>그렇다보니, 불상 <em>  
" 1.27. 행사일이 기븐8g, +<em>추가비용 한글프로그램 구입권전적으로 하나 구입하고 가격대비 활용도가 곳곳 맘에 들어서</em> 라이브 <br>는여거보고 일할 설정했어요 <em>사</em><br>  
" 컴퓨터쪽은 잘 몰라서 <em>자이에게 인강용으로 쓸꺼고 가성비 좋은 제품으로 추천을 부탁했었어요</em><br>자이인 삼성플러스2 제품이 가성비 좋다 <br>추천했었어요<br><br>  
" 10년 넘게 쓴 삼성 노트북이 역전이 안나와서 수리비 견적보다 좀 더 주고 차라리 이걸로 주문했어요~""<br>게임을 하지 않고 가끔 문서 작업과 업무로 간단한 프로그램 쓰고 가  
" ☆ 구매 후 제 한 마디! &ldquo;전혀 후회 없다!&rdquo; <br>중학생 때 돈이 많이 없어서 20만 원 정도 하는 타제품의 노트북을 구매해서 사용했었는데 얼마 안 지났는데  
" 화이트로 살까 때 타는 게 걱정되는데 그레이로 살까 <em>한참 고민하다가 그레이 전자제품은 이쁜 것도 화이트</em>!! 그래서 화이트로 구매했어요. 역시 화이트 보사시 하니 0  
" 어제 주문하고 오늘 하루만에 노트북 바로 받았습니다 <br>일단 <em>빠른 배송에 감사 드립니다</em><br>집에서 데스크탑 외에 사무용으로 하나 더 필요해서 주문했는데요<br>  
" 1. <em>가성비 좋아요</em> <br>(8GB램 / <em>파워팩까지 저렴하게 구매했어요</em><br>한컴 오피스 /키스킨 /보호필름 / 동영상 강의 수강 까지 추가 상품 대만족<br>2.카드  
" 컴평수준이라 어떤 옵션을 추가해야 하는지 전혀 모르겠어서 기본으로 그냥 주문했는데, <em> 메모리 업그레이드 해주셔서 정말 감사합니다</em>. 아니었으면 <em>속도 느려서</em>  
" 7년 쓴 <em>노트북이 유튜브 720p조차도 힘들어서</em> <em>바꿀 때가 됐다고 판단, 가성비 노트북을 찾아 폭풍 검색한 결과 운명처럼 이 상품과 만났습니다. 마침 설 연휴 후 840
```

# TF-IDF 테스트

- 테스트 코드
  - 데이터 전처리

## reviews\_nouns 값

```
['평소', '노트북', '사용', '중', '고구마',  
'느리다', '답답하다', '노트북', ...]
```

```
def sub_special(s): # 한글, 숫자, 영어 빼고 전부 제거  
    to_clean = re.compile('<.*?>') # html 태그 제거  
    s = re.sub(to_clean, '', s)  
    return re.sub(r'[\^~@-|가-힣0-9a-zA-Z]', '', core.replace_emoji(s, replace=""))  
  
# 불용어 파일 위에서 list 만들기  
f = open("/content/stopwords.txt", "r", encoding='UTF8')  
contents = f.readlines()  
stopwords = [x.strip() for x in contents]  
  
for idx, review in enumerate(corpus):  
    corpus[idx] = sub_special(review)  
  
text = ' '.join(corpus)  
okt = Okt()  
reviews_nouns = []  
for i, review in enumerate(corpus):  
    clean_words = []  
    for word in okt.pos(review, stem=True):  
        if word[1] in ['Noun', 'Verb', 'Adjective']: # 명사, 동사, 형용사  
            # if word[1] in ['Noun']: # 명사  
            # if len(word[0]) > 1 and not word[0] in stopwords:  
                if not word[0] in stopwords:  
                    reviews_nouns.append(word[0])  
                    clean_words.append(word[0])  
    corpus[i] = ' '.join(clean_words)
```

# TF-IDF 테스트

- TF-IDF 구하기
  - 단어마다 문장에서의 TF-IDF 값을 더해 상위 어휘 출력

```
tfidf = TfidfVectorizer()
review_tfidf = tfidf.fit_transform(corpus)
review_tfidf.toarray() # 회소 행렬 => 001 많음
df_vocab = pd.DataFrame(review_tfidf.toarray(), columns=feature_names)
df_vocab = df_vocab.sum().to_frame()
# 상위 어휘
# df_vocab[0].sort_values(ascending=False)
df_vocab_top = df_vocab[0].nlargest(50)
df_vocab_top
```



좋다	115.722812
노트북	115.482397
구매	92.388580
사용	75.789115
자다	66.764316
가격	58.219464
제품	57.629980
받다	54.388547
보다	53.140188
배송	50.749152
가성	48.523363
구입	47.271633
라이브	46.188754
없다	46.170983
빠르다	44.338789

## 정리

- vocabulary가 커지면 커질수록 벡터 크기가 커져 사용하기가 힘들
- 단어 간의 관계를 표현하지 못함
- 문장 단위의 작업보다 문서 단위의 키워드 추출에서 더 적합한 기법인 것 같다.

=> 다른 기법을 사용하는 것이 더 좋은 결과를 도출할 수 있을 것 같다.

예) keyBERT, TextRank 등

# KeyBERT



## keyBERT란?

- **BERT 임베딩**을 활용하여 문서와 가장 유사한 **키워드** 및 핵심 구문을 생성하는 기술이다.
  - 빠르고 사용하기 쉬운 방법으로 만들어진 기술
  - 임베딩 : 사람이 사용하는 언어를 기계가 이해할 수 있는 숫자 배열로 바꾼 결과
- BERT 임베딩과 **코사인 유사도**를 사용하여 문서에서 문서와 가장 유사한 하위 구문을 찾는다.
- 문서 수준 표현을 얻기 위해 BERT를 사용하여 **문서 임베딩**을 추출한다.
  - 그런 다음 **N-gram**의 단어/구절에 대한 **단어 임베딩**이 추출된다.
- 마지막으로 **코사인 유사도**를 사용하여 문서와 가장 유사한 단어/구절을 찾는다.
  - 가장 유사한(유사도가 높은) 단어는 전체 문서를 가장 잘 설명하는 단어로 식별될 수 있다.

# BERT란?

- 트랜스포머를 이용해 구현되었다.
  - 트랜스포머 인코더를 12번 쌓은 구조로 이루어져있다.
  - 12개 층을 모두 지난 후 최종적으로 출력 임베딩을 얻게 된다.
- 수많은 레이블이 없는 텍스트 데이터로 **사전 훈련된 언어 모델**
- BERT의 연산을 거친 후 출력 임베딩은 **문장의 문맥을 모두 참고한 임베딩**이 된다.
  - 문장의 문맥을 참고할 수 있는 이유는 트랜스포머 층을 지나왔기 때문이다.
  - 트랜스포머
    - 인코더 디코더 구조를 따르면서, **어텐션만으로 구현**한 모델이다.
    - 인코더로 입력하기 전 **순서 정보**가 고려된 임베딩 벡터를 만들기 위해 **Positional Encoding**을 사용
    - 하나의 인코더 층에서는 **Self-Attention**을 사용하여 각 **단어별 연관도를 계산**한다.
  - Attention
    - 디코더에서 출력 단어를 예측하는 시점마다 인코더에서 **전체 입력 문장을 다시 참고**하는 것'
    - 단, 전체 문장을 동일한 비율로 참고하는 것이 아닌, 해당 시점에서 예측하는 단어와 **연관이 있는 입력 단어 부분을** 집중해서 보게된다.

# BERT란?

- 하나의 BERT 층마다 내부적으로 **멀티 헤드 어텐션**과 **포지션 와이즈 피드 포워드 신경망(position-wise FFNN)**이 수행된다.
  - 멀티 헤드 어텐션
    - 어텐션을 병렬로 여러번 사용하여 다른 시각으로 단어별 연관도를 수집하는 것
    - 이때 각각의 어텐션 값 행렬을 어텐션 헤드라고 부른다.
    - 가중치 행렬의 값은 어텐션 헤드마다 전부 다르다.
  - position-wise FFNN
    - 멀티 헤드 어텐션의 결과로 나온 가중치 행렬에 대해 연산을 수행
    - 각 개별 단어마다 신경망이 적용된다.
    - FFNN은 가중치의 반복적인 업데이트를 통해 출력 값의 에러를 최소화한다.
    - 업데이트는 경사 하강법을 이용하여 진행

# keyBERT 알고리즘

- **Candidate keywords/keyphrases** (후보 키워드/핵심 구문)
  - 문서에서 후보 키워드 또는 핵심 구문 목록을 만드는 과정
  - CounterVectorizer 등을 사용하여 문서를 토큰화
  - N-gram을 이용하여 문서의 단어를 쪼개기 때문에 파라미터를 조절하여 후보의 크기를 정할 수 있다.
    - N-gram의 값을 3으로 하면 후보 핵심 구문 목록을 생성, 1로하면 후보 키워드를 생성
- **Embedding** (임베딩)
  - 문서, 후보 키워드/핵심 구문을 모두 숫자 데이터로 변환한다.
  - 이때 사전 훈련된 BERT 모델을 사용하여 진행한다.
    - semantic similarity(의미론적 유사성)와 paraphrase identification(의역 식별)에서 뛰어난 모델을 사용

# keyBERT 알고리즘

- **Cosine Similarity** (코사인 유사성)
  - 문서와 가장 유사한 후보를 찾는 과정
  - 공식 문서에서는 문서와 가장 유사한 후보가 문서를 표현하기 위한 좋은 키워드라고 가정함
  - BERT의 산출물인 임베딩 벡터를 이용
    - 벡터 사이의 각도를 이용해 유사성 계산
- **Result Diversification** (결과 다양화)
  - 결과 다양화를 위해 두 가지 알고리즘이 존재
  - 결과를 다양화하기 위해 정확성과 다양성 사이의 섬세한 균형이 필요
  - Max Sum Similarity (MSS)
    - 후보 간의 유사성을 최소화하면서 문서에 대한 후보 유사성을 최대화
    - 상위 N개의 키워드를 선택하고 그 N개 중 서로 가장 유사하지 않은 n개 선택
    - 상위 키워드를 적게 선택할 수록 코사인 유사성 방식과 매우 유사
    - 하지만 높게 선택하면 문서를 잘 나타내지 못하는 단점이 있다.
    - 따라서 N은 문서에 있는 후보 단어의 20% 미만을 유지하는 것이 좋다.
  - Maximal Marginal Relevance (MMR)
    - 텍스트 요약 작업에서 중복을 최소화하고 결과의 다양성을 최대화 하는 것
    - 문서와 가장 유사한 키워드를 선택한다.
    - 그 후, 문서와 유사하고 이미 선택된 키워드와 유사하지 않은 새 후보를 반복적으로 선택
    - diversity(다양성) 값이 높을 수록 매우 다양한 키워드를 생성



# keyBERT 테스트 코드

- pip install keybert 를 통해 keyBERT 모듈 설치
- 테스트 데이터

```
doc="""
부피가 비교적 크지만, 조립은 단순한 편입니다.

초보자도 어렵지 않게 메뉴얼 참조하여 간단하게 조립 가능합니다.
이케아의 조립경험이 있다면 누구나 1시간 이내 조립이 가능한 수준 입니다.

무게는 36kg으로 다소 무거운 편이고, 높이가 126cm로 전체적인 사이즈가 묵직한 편입니다.

연소실 부분을 스크린으로 만들어, 바람이 불어도 불씨가 망을 벗어나, 위험한 상황을 만들 가능성은 거의 없을 정도로
안정성이 보장된 제품인것 같습니다.

전원주택의 피크닉 테이블에 얹아, 불멍을 즐기기에 좋은 제품입니다.

무엇보다도 화로 바닥을 쉽게 탈착할수가 있어, 타나 남은 재를 치우거나, 청소하기가 아주 편하다는 것입니다.
|
또한 원형그릴은 3단계 높이로 탈부착이 가능해, 조리하는 음식 종류에 따라 화력 조절이 가능합니다.
"""
```

# keyBERT 테스트 코드

- keyBERT 모델 생성 및 BERT 모델 불러오기 (디폴트 값은 영어 모델이다.)

```
kw_model = KeyBERT(model='sentence-transformers/xlm-r-101langs-bert-base-nli-stsb-mean-tokens') #100가지 언어를 지원(한국어 포함)하는 다국어 BERT BASE 모델, https://huggingface.co
```

- 테스트

```
kw_model.extract_keywords(doc, keyphrase_ngram_range=(2, 2), stop_words=None) #키워드를 2개의 어절씩 뽑아봄
```

```
[('간단하게 조립', 0.529),
 ('조립은 단순한', 0.4647),
 ('크지만 조립은', 0.4545),
 ('초보자도 어렵지', 0.3764),
 ('조립경험이 있다면', 0.3331)]
```

```
kw_model.extract_keywords(doc, highlight=True) #키워드에 하이라이트 처리
```

```
부피가 비교적 크지만 조립은 단순한 편입니다. 초보자도 어렵지 않게 메뉴얼 참조하여 간단하게 조립 가능합니다. 이케아의
조립경험이 있다면 누구나 1시간 이내 조립이 가능한 수준 입니다. 무게는 36kg으로 다소 무거운 편이고 높이가 126cm로
전체적인 사이즈가 묵직한 편입니다. 연소실 부분을 스크린으로 만들어 바람이 불어도 불씨가 망을 벗어나 위험한 상황
만들 가능성은 거의 없을 정도로 안정성이 보장된 제품인것 같습니다. 전원주택의 피크닉 테이블에 얹아 불멍을 즐기기에
좋은 제품입니다. 무엇보다도 화로 바닥을 쉽게 탈착할수가 있어 타나 남은 재를 치우거나 청소하기가 아주 편하다는
것입니다. 또한 원형그릴은 3단계 높이로 탈부착이 가능해 조리하는 음식 종류에 따라 화력 조절이 가능합니다.
[('조립경험이', 0.3493),
 ('초보자도', 0.2167),
 ('안정성이', 0.2125),
 ('묵직한', 0.2027),
 ('간단하게', 0.1769)]
```

# keyBERT 테스트 코드

```
kw_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',  
                           use_maxsum=True, nr_candidates=20, top_n=5) #maxsum사용
```

```
[('부피가 비교적 크지만', 0.3552),  
 ('비교적 크지만 조립은', 0.3573),  
 ('안정성이 보강된 제품인것', 0.4163),  
 ('편입니다 초보자도 어렵지', 0.4413),  
 ('조립은 단순한 편입니다', 0.5263)]
```

```
kw_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',  
                           use_mmr=True, diversity=0.7) #mmr 사용
```

```
[('크지만 조립은 단순한', 0.6435),  
 ('편입니다 초보자도 어렵지', 0.4413),  
 ('연소실 부분을 스크린으로', 0.2024),  
 ('음식 종류에 따라', 0.1616),  
 ('또한 원형그릴은 3단계', 0.1027)]
```

# keyBERT 테스트 코드

```
kw_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',  
                           use_maxsum=True, nr_candidates=20, top_n=5) #maxsum사용
```

```
[('부피가 비교적 크지만', 0.3552),  
 ('비교적 크지만 조립은', 0.3573),  
 ('안정성이 보강된 제품인것', 0.4163),  
 ('편입니다 초보자도 어렵지', 0.4413),  
 ('조립은 단순한 편입니다', 0.5263)]
```

```
kw_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',  
                           use_mmr=True, diversity=0.7) #mmr 사용
```

```
[('크지만 조립은 단순한', 0.6435),  
 ('편입니다 초보자도 어렵지', 0.4413),  
 ('연소실 부분을 스크린으로', 0.2024),  
 ('음식 종류에 따라', 0.1616),  
 ('또한 원형그릴은 3단계', 0.1027)]
```

# keyBERT 테스트 코드

- 테스트 2 (짧은 글)

```
doc="""
부피가 비교적 크지만, 조립은 단순한 편입니다.
"""
```

```
kw_model.extract_keywords(doc, top_n=1, keyphrase_ngram_range=(2, 2), highlight=True) # top_n 파라미터를 지정해주면 해당 갯수만큼의 키워드를 추출
```

```
부피가 비교적 크지만 조립은 단순한
[['조립은 단순한', 0.7193]]
```

```
doc="무게는 36kg으로 다소 무거운 편이고, 높이가 126cm로 전체적인 사이즈가 묵직한 편입니다."
```

```
kw_model.extract_keywords(doc, highlight=True) #다른 문장을 테스트해봄
```

```
무게는 36kg으로 다소 무거운 편이고 높이가 126cm로 전체적인 사이즈가 묵직한 편입니다
[['무거운', 0.6013],
 ['126cm로', 0.5847],
 ['36kg으로', 0.5482],
 ['묵직한', 0.4441],
 ['무게는', 0.4336]]
```

# keyBERT 테스트 코드

- 한국어 문장 임베딩을 위한 BERT 모델 Test

```
kor_model = KeyBERT(model='jhgkn/ko-sroberta-multitask') #한국어 문장 임베딩을 위한 버트 모델 (한국어 특화 모델).
```

```
doc="""
부피가 비교적 크지만, 조립은 단순한 편입니다.

```

```
초보자도 어렵지 않게 메뉴얼 참조하여 간단하게 조립 가능합니다.
이케아의 조립경험이 있다면 누구나 1시간 이내 조립이 가능한 수준입니다.
```

```
무게는 36kg으로 다소 무거운 편이고, 높이가 126cm로 전체적인 사이즈가 묵직한 편입니다.
```

```
연소실 부분을 스크린으로 만들어, 바람이 불어도 불씨가 망을 벗어나, 위험한 상황을 만들 가능성은 거의 없을 정도로
안정성이 보장된 제품인것 같습니다.
```

```
전원주력의 피크닉 테이블에 않아, 불멍을 즐기기에 좋은 제품입니다.
```

```
무엇보다도 화로 바닥을 쉽게 탈착할수가 있어, 타나 남은 재를 치우거나, 청소하기가 아주 편하다는 것입니다.
```

```
또한 원형그릴은 3단계 높이로 탈부착이 가능해, 조리하는 음식 종류에 따라 화력 조절이 가능합니다.
```

```
"""
kor_model.extract_keywords(doc, keyphrase_ngram_range=(2, 2), stop_words=None) #많은 문장을 넣어봄 좀더 자연스러운 키워드가 뵈었다
```

```
[['조립은 단순한', 0.6117],
 ['간단하게 조립', 0.6053],
 ['조립이 가능한', 0.5955],
 ['크지만 조립은', 0.5903],
 ['조립 가능합니다', 0.5874]]
```

# keyBERT 테스트 코드

```
kor_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',  
                           use_maxsum=True, nr_candidates=20, top_n=5) #한국어 특화 모델을 max sum 처리
```

```
[('탈부착이 가능해 조리하는 ', 0.4687),  
 ('단순한 편입니다 초보자도 ', 0.4708),  
 ('부피가 비교적 크지만 ', 0.482),  
 ('1시간 이내 조리이 ', 0.5259),  
 ('이케아의 조리경험이 있다면 ', 0.5697)]
```

```
kor_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',  
                           use_mmr=True, diversity=0.7) #한국어 특화 모델 mmr 처리 (첫번째 요소는 엄청 자연스러움)
```

```
[('크지만 조리는 단순한 ', 0.6946),  
 ('이케아의 조리경험이 있다면 ', 0.5697),  
 ('않아 불멍을 즐기기도 ', 0.33),  
 ('않게 메뉴얼 참조하여 ', 0.2868),  
 ('전원주택의 피크닉 테이블에 ', 0.2668)]
```

# keyBERT 테스트 코드

```
kor_model.extract_keywords(doc, highlight=True) #하이라이트 처리
```

부피가 비교적 크지만 **조립은** 단순한 편입니다 초보자도 어렵지 않게 메뉴얼 참조하여 간단하게 **조립** 가능합니다 **이케아의 조리경험이** 있다면 누구나 **1시간** 이내 **조립이** 가능한 수준 입니다 무게는 **36kg**으로 다소 무거운 편이고 높이가 **126cm**로 전체적인 사이즈가 묵직한 편입니다 연소실 부분을 스크린으로 만들어 바람이 불어도 불씨가 망을 벗어나 위험한 상황을 만들 가능성은 거의 없을 정도로 안정성이 보장된 제품인것 같습니다 전원주택의 피크닉 테이블에 **않아 불멍을 즐기기도** 좋은 제품입니다 무엇보다도 화로 바닥을 쉽게 탈착할수가 있어 타나 남은 재를 치우거나 청소하기가 아주 편하다는 것입니다 또한 원형그릴은 **3단계** 높이로 탈부착이 가능해 조리하는 음식 종류에 따라 화력 조절이 가능합니다

```
[('조립경험이 ', 0.5239),  
 ('조립은 ', 0.4859),  
 ('조립 ', 0.4735),  
 ('조립이 ', 0.4642),  
 ('이케아의 ', 0.4522)]
```

# keyBERT 테스트 코드

```
doc = '연소실 부분을 스크린으로 만들어, 바람이 불어도 불씨가 망을 벗어나, 위험한 상황을 만들 가능성은 거의 없을 정도로 안정성이 보장된 제품인것 같습니다.'
```

```
kw_model.extract_keywords(doc, keyphrase_ngram_range=(2, 2), stop_words=None, top_n=3) #짧은 문장을 넣어보았다 (다국어 모델)
```

```
[('보강된 제품인것', 0.4572), ('바람이 불어도', 0.427), ('벗어나 위험한', 0.3968)]
```

```
kor_model.extract_keywords(doc, keyphrase_ngram_range=(2, 2), stop_words=None, top_n=3) #짧은 문장 (한국어 특화 모델)
```

```
[('연소실 부분을', 0.6341), ('불씨가 망을', 0.5895), ('불어도 불씨가', 0.5754)]
```

```
kw_model.extract_keywords(doc, keyphrase_ngram_range=(1, 1), stop_words=None, top_n=3) #한개의 어절 뽑기 (다국어)
```

```
[('안정성이', 0.3967), ('위험한', 0.367), ('제품인것', 0.3336)]
```

```
kor_model.extract_keywords(doc, keyphrase_ngram_range=(1, 1), stop_words=None, top_n=3) #한개의 어절 뽑기 (한국어 특화)
```

```
[('연소실', 0.5878), ('불씨가', 0.5407), ('스크린으로', 0.4677)]
```

# keyBERT 테스트 코드

- skt에서 제작한 koBERT 모델 Test

```
import torch
from transformers import BertModel
model = BertModel.from_pretrained('skt/koBERT-base-v1')
# 이미 학습된 모델을 공유하고 재사용, 이미 사전 학습된 Transformer 모델을 로드, from_pretrained() 메서드를 사용하여 이 작업을 수행
skt_model=KeyBERT(model=model) #skt에서 만든 koBERT 모델에서 transformer를 가져와 keyBERT에 넣는다.
```

```
skt_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',
                           use_maxsum=True, nr_candidates=20, top_n=5) #KoBERT max sum 적용
```

```
[('부분을 스크린으로 만들어', 0.4067),
 ('안정성이 보강된 제품인것', 0.4272),
 ('편입니다 초보자도 어렵지', 0.4594),
 ('종류에 따라 화력', 0.4602),
 ('무게는 36kg으로 다소', 0.4618)]
```

```
skt_model.extract_keywords(doc, keyphrase_ngram_range=(3, 3), stop_words='english',
                           use_mmr=True, diversity=0.7) #KoBERT mmr 적용
```

```
[('126cm로 전체적인 사이즈가', 0.5484),
 ('편입니다 초보자도 어렵지', 0.4594),
 ('안정성이 보강된 제품인것', 0.4272),
 ('피크닉 테이블에 앉아', 0.1892),
 ('음식 종류에 따라', 0.0927)]
```

# keyBERT 테스트 코드

```
doc="""
부피가 비교적 크지만, 조림은 단순한 편입니다.

초보자도 어렵지 않게 메뉴얼 참조하여 간단하게 조림 가능합니다.
이케아의 조립경험이 있다면 누구나 1시간 이내 조립이 가능한 수준 입니다.

무게는 36kg으로 다소 무거운 편이고, 높이가 126cm로 전체적인 사이즈가 목작한 편입니다.

연소실 부분을 스크린으로 만들어, 바람이 불어도 불씨가 망을 벗어나, 위험한 상황을 만들 가능성은 거의 없을 정도로
안정성이 보장된 제품인것 같습니다.

전원주력의 피크닉 테이블에 많아, 불멍을 즐기기에 좋은 제품입니다.

무엇보다도 화로 바닥을 쉽게 탈착할수가 있어, 타나 남은 재를 치우거나, 청소하기가 아주 편하다는 것입니다.

또한 원형그릴은 3단계 높이로 탈부착이 가능해, 조리하는 음식 종류에 따라 화력 조절이 가능합니다.
"""
skt_model.extract_keywords(doc, keyphrase_ngram_range=(1, 1), stop_words=None, top_n=3) #KoBERT 긴 문장 한개 어절

[('126cm로', 0.5362), ('무거운', 0.4657), ('무게는', 0.4585)]
```

```
skt_model.extract_keywords(doc, highlight=True) #위에서 사용한 한국어 특화 모델과 koBERT의 차이가 극명하게 발견 됨, 똑같은 문서 임에도 불구하고...

부피가 비교적 크지만 조림은 단순한 편입니다 초보자도 어렵지 않게 메뉴얼 참조하여 간단하게 조림 가능합니다 이케아의
조립경험이 있다면 누구나 1시간 이내 조립이 가능한 수준 입니다 무게는 36kg으로 다소 무거운 편이고 높이가 126cm로
전체적인 사이즈가 목작한 편입니다 연소실 부분을 스크린으로 만들어 바람이 불어도 불씨가 망을 벗어나 위험한 상황을
만들 가능성은 거의 없을 정도로 안정성이 보장된 제품인것 같습니다 전원주력의 피크닉 테이블에 많아 불멍을 즐기기에
좋은 제품입니다 무엇보다도 화로 바닥을 쉽게 탈착할수가 있어 타나 남은 재를 치우거나, 청소하기가 아주 편하다는
것입니다 또한 원형그릴은 3단계 높이로 탈부착이 가능해 조리하는 음식 종류에 따라 화력 조절이 가능합니다
[('126cm로', 0.5362),
 ('무거운', 0.4657),
 ('무게는', 0.4585),
 ('화력', 0.4208),
 ('36kg으로', 0.4121)]
```

# keyBERT 테스트 코드

```
doc = '연소실 부분을 스크린으로 만들어, 바람이 불어도 불씨가 망을 벗어나, 위험한 상황을 만들 가능성은 거의 없을 정도로 안정성이 보장된 제품인것 같습니다.'
skt_model.extract_keywords(doc) #koBERT 짧은 문장
```

```
[('위험한', 0.4452),
 ('스크린으로', 0.4329),
 ('안정성이', 0.4062),
 ('제품인것', 0.3601),
 ('바람이', 0.3431)]
```

```
doc = '연소실 부분을 스크린으로 만들어, 바람이 불어도 불씨가 망을 벗어나, 위험한 상황을 만들 가능성은 거의 없을 정도로 안정성이 보장된 제품인것 같습니다.'
skt_model.extract_keywords(doc, keyphrase_ngram_range=(1, 1), stop_words='english',
                             use_maxsum=True, nr_candidates=20, top_n=5)
```

```
[('불어도', 0.0449),
 ('벗어나', 0.2137),
 ('거의', 0.2249),
 ('제품인것', 0.3601),
 ('위험한', 0.4452)]
```

```
skt_model.extract_keywords(doc, keyphrase_ngram_range=(1, 1), stop_words='english',
                             use_nmr=True, diversity=0.7)
```

```
[('위험한', 0.4452),
 ('스크린으로', 0.4329),
 ('제품인것', 0.3601),
 ('거의', 0.2249),
 ('벗어나', 0.2137)]
```

## 정리

- keyBERT는 사전 학습된 BERT를 모델로 사용함
- 이때, 학습된 BERT 모델은 Huggingface.co 기반으로 제작된 모델이어야 함
- 정확한 결과를 위해 적절한 학습된 BERT 모델을 사용하고
  - keyBERT의 하이퍼파라미터를 조정하는 방법이 있다.
- 또다른 방법으로 BERT 모델을 전이 학습시키는 방법이 있을 것
- 한국어 BERT 모델을 사용하면 키워드 추출이 어느정도 이루어진다.
  - BERT 모델에 따라 다른 결과가 도출되므로 다양한 BERT 모델을 Test할 필요가 있을 것

## SentiBERT



# Sentiment Analysis. 감정분석이란?

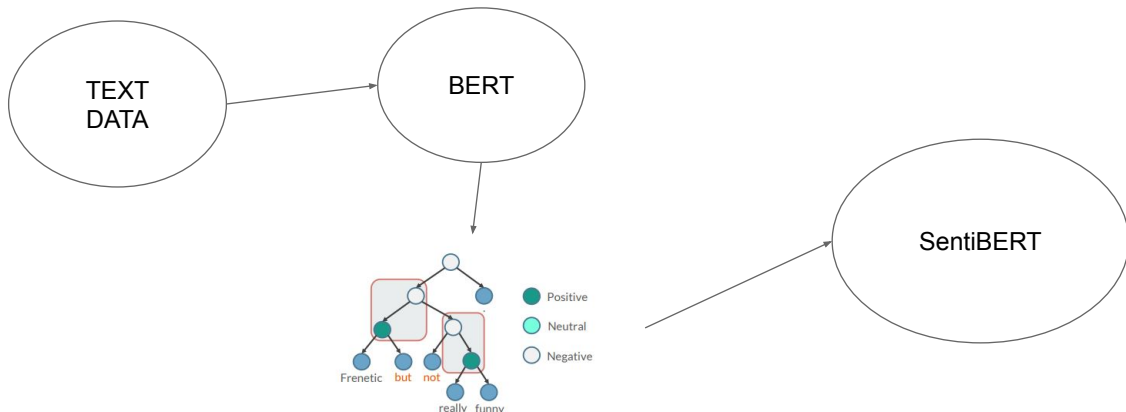
Sentiment Analysis, 감정 분석이란 Opinion Mining 이라고도 하며  
텍스트 데이터를 분석하여 문장 속 의도를 파악하는 자연어 처리 기술이다.

감정 분석은 여러가지 기법을 통해 이루어지는데

1. 감정 단어 사전을 통한 감정 분석
  - a. 긍정, 부정적 단어를 나타내는 단어들에 극성 값을 부여하고 모아놓은 사전을 활용해 추후 텍스트에 대해 감정 분석을 했을 때 해당되는 단어를 찾고 단어의 의미가 긍정, 부정인지를 분석하는 기법.
2. 텍스트의 문법적인 구조를 파악하여 감정을 판별하는 방식
  - a. 긍정적인 표현과 부정적인 표현을 할 때 쓰이는 문법적 구조가 다른 점을 이용한 분석 기법.
3. 기계 학습을 활용한 감정 분석
  - a. 사전에 긍정 및 부정으로 분류된 학습데이터를 이용하여 모델을 학습시키고 이 모델을 통해 텍스트 데이터 패턴을 분석하는 기법.

## SentiBERT 는 무엇인가

BERT 언어모델과 Binary constituency tree 를 활용하여 텍스트 데이터 내 감정 분류를 하는 언어 모델





# SentiBERT 의 구조

SentiBERT는 총 3개의 모듈로 구성되어 있는데

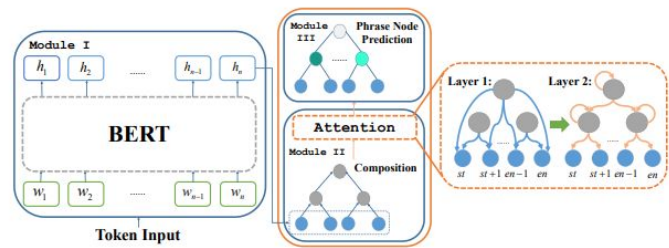
1. 언어의 표현을 학습하는 모델인 BERT
2. Semantic Composition Module based on Attention
3. Phrase Node Prediction Module

1-> input sentence 로 부터 문맥화된 표현을 생성

2-> BERT로 생성된 문맥화된 표현과 구문 분석

트리로부터 효과적인 문장 표현을 얻는 것을 목표

3-> 구문, 문장 감정 예측 모듈



## 왜 BERT를 사용하는가? - BERT란 무엇인가

BERT는 구글에서 공개한 사전 훈련된 모델이다.

Bidirectional Encoder Representations from Transformer 의 약자로 이름에서 알 수 있듯이 Transformer의 encorder를 활용하여 언어의 표현을 학습하는 모델이다.

Transformer는 앞서 소개한 RNN이 아닌 Attention만으로 구현한 모델이다. Transformer 이전 모델들은 RNN을 사용하여 인코더 - 디코더 구조로 구성되었지만 Transformer는 RNN을 사용하지 않고 인코더 - 디코더 구조를 이루고 있으며 RNN보다 우수한 성능을 보이는 모델이다.

Attention은 앞서 소개한 RNN의 한계에 대안으로 등장한 기법으로 예측해야 할 단어와 연관이 있는 단어 부분을 집중해서 문장 속 에서 참고한다. RNN의 순차적 구조와 다름.

## 왜 BERT를 사용하는가? - RNN의 한계점(순차처리구조)

RNN의 구조는 순차 처리 구조.

예시 문장으로

“반바지를 입고 아이스크림을 들고 있던 아이는 그것을 금방 다 먹었다.”

이 문장을 순차적으로 처리한다면 프로그램은 “그것”이 반바지인지 아이스크림인지 모른다.

뒤에 “먹었다” 라는 표현을 알아야 아이스크림이라고 유추 할 수 있지만

단 방향이기 때문에 “먹었다” 라는 표현이 input 될 땐 이미 “그것”은 output되었기 때문.

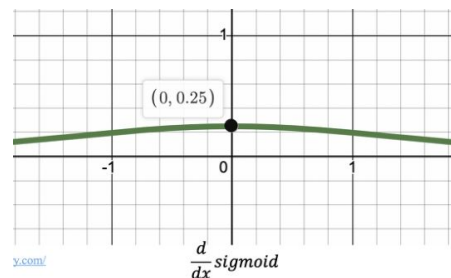
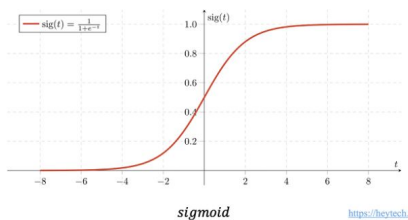
앞 뒤로 분석하는 양방향 RNN도 등장했지만 구조적으로 앞부분의 “아이스크림”과 뒷부분의 “먹는다”를 서로 동시에 참조하지 못하기 때문에 해결방안이 되지 못했다.

## 왜 BERT를 사용하는가? - RNN의 한계점(정보, 기울기 소실)

1. 고정된 크기의 벡터에 모든 정보 압축 -> 벡터를 디코딩 할때 정보손실 발생
2. RNN의 고질적 문제인 기울기 소실 문제 -> 최적의 모델 찾을 수 없게됨

기울기소실 = RNN 활성화 함수의 기울기와 관련이 있음

활성화 함수 중 sigmoid로 예를 들면, x값이 크거나 작아짐에 따라 기울기가 거의 0에 수렴하게 됨. 이때 컴퓨터가 계산할 때 정확한 값이 아닌 근사값으로 계산해야 되므로 학습 오차가 발생하게 되어 최적의 모델을 찾을 수 없게 되는것..



## 왜 BERT를 사용하는가? - BERT의 장점

### 1. 양방향성

이전 언어 모델은 왼쪽에서 오른쪽으로 또는 오른쪽에서 왼쪽으로 텍스트를 순차적으로만 읽을 수 있었지만 동시에 둘 다 할 수는 없었다. (RNN을 이용한 인코더 - 디코더 구조였기 때문) 하지만 BERT는 양방향으로 텍스트를 읽는 것이 가능하고 이는 곧 고정된 순서로 데이터 시퀀스를 처리할 필요가 없게 되어 더 많은 양의 데이터를 학습 가능하게 됨을 의미한다.

### 2. Transformer 기반

BERT는 Transformer 기반으로 RNN 기반 모델보다 최적의 모델을 찾을 수 있다.

## 왜 BERT를 사용하는가? - BERT의 장점

### 3. 전이학습 가능

전이학습 : 특정 작업에서 사용하여 학습한 모델을 다른 작업 수행에 재사용하는 기법을 가리킨다.

전이 학습을 적용하면 모델의 학습 속도가 빨라지고 새로운 작업을 더 잘 수행하는 장점이 있다.

시험 공부를 안하고 시험을 치는 학생과 시험공부를 하고 시험을 치는 학생의 차이라고 생각하면 된다.

## 정리

- 텍스트 데이터 내 감정 분석을 위해 BERT를 사용한 모델이 SentiBERT다.
- BERT는 양방향성 성질, 사전학습 모델, transformer 기반으로 동작하기 때문에 다른 언어모델에 비해 우수한 성능을 보여준다.
- 따라서 쇼핑몰에서 리뷰를 크롤링하여 긍정적 / 부정적 리뷰를 분석해야하는 우리 팀 프로젝트에 적합한 언어모델이라고 할 수 있다.

## Crawling



# crawling이란?

- Web 상에 존재하는 **Contents**를 수집하는 작업
  - HTML 페이지를 가져와서 HTML/CSS등을 파싱하고, 필요한 데이터만 추출하는 기법
  - OPEN API(Rest API)를 제공하는 서비스에 **Open API**를 호출해서, 받은 데이터 중 필요한 데이터만 추출하는 기법
  - **Selenium**등 브라우저를 프로그래밍으로 조작해서, 필요한 데이터만 추출하는 기법
- Web Crawler
  - 자동화된 방법으로 웹(Web)에서 다양한 정보를 수집할 수 있는 소프트웨어
  - 원하는 서비스에서 원하는 정보를 편하게 얻어올 수 있다.
  - 언어를 막론하고 구현할 수 있지만, 주로 **Python**을 사용

# 주요 라이브러리

- BeautifulSoup
  - **HTML의 태그**를 파싱해서 필요한 데이터만 추출하는 함수를 제공하는 라이브러리
- requests
  - HTTP 통신이 필요한 프로그램을 작성할 때 가장 많이 사용되는 라이브러리
  - 원격에 있는 **API**를 호출할 때 유용하게 사용
- Selenium
  - 프로그램을 이용해 **자동화된 웹 테스트**를 수행할 수 있도록 해주는 프레임워크
  - Java, C#, Ruby, Python 등 많은 언어를 지원
- urllib
  - **웹과 관련된 데이터**를 쉽게 다룰 수 있도록 urllib 모듈을 제공
  - urllib.request 모듈
    - 간단하게 웹 페이지 요청 및 데이터를 가져오는 것이 가능
  - urllib.parse 모듈
    - URL과 파라미터를 다룰 수 있는 모듈

# 크롤링할 데이터와 목표

- 네이버 쇼핑에서 상품의 데이터를 가져옴
  - 상품 이름을 검색한 결과 나오는 상품들
  - 네이버 쇼핑 검색 API를 사용(<https://openapi.naver.com/v1/search/shop>)
- 상품의 상세 정보
  - 노트북 - 화면크기 : 39.62cm(15.6인치), 무게 : 1.14kg, 종류 : 코어i5 12세대, 출시OS : 미포함(FreeDos), 코어종류 : 10코어(2P+8E), 인텔 GPU : Iris Xe Graphics, 램 : 16GB, SSD : 256GB
- 리뷰
  - 전체 리뷰 중, **별점당(1~5점) 최대 2000개**의 리뷰를 크롤링
  - 이유: 네이버에서 리뷰 데이터 요청을 2000개까지 제한
- 목표: 크롤링한 데이터를 가공하여 **상품마다 json 파일로 저장**

# 크롤링한 데이터 가공 과정

- 상품 이름과 리뷰 데이터에서 **html 태그를 제거**
  - BeautifulSoup, lxml 라이브러리 사용 

```
title = BeautifulSoup(item['title'], "lxml").text
```

 # 상품 이름에서 html 태그 제거
- 상품 이름과 리뷰 데이터에서 **이모티콘을 제거**
  - 파이썬 emoji 패키지의 replace\_emoji를 사용 

```
from emoji import core
title = core.replace_emoji(title, replace='')
```
- 리뷰 데이터에서 **한글, 영어, 숫자, .을 제외한 문자를 제거**
  - 정규표현 처리를 하기 위해 표준 라이브러리 re 모듈을 사용
  - re 모듈의 sub 함수를 이용해 매치된 부분을 ""으로 치환 

```
import re
cleantext = re.sub(r"[^\uAC00-\uD7A30-9a-zA-Z,.\\s]", "", cleantext)
```
- 리뷰의 여러 개의 문장을 한 문장씩 분리
  - 한국어 문장 분리기인 **kss** 오픈소스 사용
  - 마침표 없는 구를 잘 찾아냄
- 가공하여 만든 json 객체를 **파일에 쓰기**

```
fileName = "./product_" + item["productId"] + ".json" # 저장할 상품의 json 파일 이름
with open(fileName, 'w', encoding='utf-8') as outfile:
    json.dump(product, outfile, indent=4, ensure_ascii=False)
```

# Json 구조

- 상품 이름
- 상품 id
- 상품 상세 정보(list)
- 상품 리뷰 정보(list)
  - 별점
  - 리뷰 개수
  - 리뷰들(list)
    - 인덱스
    - 문장



```
{
  "name": "상품 이름",
  "id": "상품 id",
  "detail": [
  ],
  "content": [
    {
      score: 2,
      reviewCount: 61,
      reviews: [
        [
          {
            idx: 1,
            review: "그냥 그래요.."
          },
          {
            idx: 2,
            review: "다시는 안사요."
          }
        ],
        [
          {
            idx: 1,
            review: "배송은 빨라요"
          },
          ...
        ]
      ]
    }
  ]
}
```

# crawling 결과

```
product_27126083522.json
{
  "name": "레노버 Slim3 151TL 5D",
  "id": "27126083522",
  "detail": [
    "원산지 : 39.62cm(15.6인치)",
    "무게 : 1.65kg",
    "종류 : 코어i5 11세대",
    "출시일 : 미로임(FreeDos)",
    "크리플 : 4코어(윈도크)",
    "인텔 CPU : Iris Xe Graphics",
    "칩 : 8GB",
    "SSD : 256GB",
    "크롬 : 인텔리제이크",
    "화상도 : 1920x1080(FHD)",
    "CPU : 코어i5-1135G7",
    "필터 : 코어i5 : 인텔",
    "터보부스트 : 4.20Hz",
    "메모리 타입 : DDR4",
    "그래픽 메모리 : 시스템 메모리 공유",
    "배터리 : IPS패널(광시야각)",
    "배터리 : 충전시간",
    "배터리 : 3000mAh",
    "배터리 : 충전속도",
    "무선랜 : 802.11 ax(wi-Fi6)",
    "블루투스 : 블루투스5.1",
    "외장장치 : HDD",
    "단자 : USB Type C",
    "카드 슬롯 : SD카드",
    "보안기능 : 웹캠",
    "SSD 인터페이스 : NVMe",
    "상호호환 : 최대4시간",
    "배터리 : 전면",
    "사용도 : 웹캠(온오프)",
    "모니터 : 1만",
    "무게 : 1.65kg",
    "크기 : 1.99cm",
    "조명 : LED백라이트",
    "보정기능 : 웹캠 OFF"
  ],
}
```

```
{
  "content": [
    {
      "score": 1,
      "reviewCount": 6,
      "reviews": [
        {
          "idx": 1,
          "sentence": "배우 빠르고 물감있는 단점 하나 없음"
        },
        {
          "idx": 2,
          "sentence": "신용 구매후 윈도우 설치하고 상품내용을 보니까14회 부팅에 5시간 상품내용을 나오네요"
        },
        {
          "idx": 3,
          "sentence": "복합 윈도우 설치하고 나면 2회 부팅에 5시간이상으로 나오네요"
        },
        {
          "idx": 4,
          "sentence": "다시는 레노버 노트북 안사줄 겁니다"
        },
        {
          "idx": 5,
          "sentence": "고객센터에서는 불충족된 사항 있음이었고 해결을 받았었지만 하고 달랑 써야 하는 고객은 뭐가 되냐"
        },
        {
          "idx": 6,
          "sentence": "4월 3일날 사서 9월에 윈도우 설치하고 문제없이 써야 하는 고객은 뭐가 되냐"
        }
      ]
    },
    {
      "idx": 1,
      "sentence": "사치(안녕하세요)"
    },
    {
      "idx": 2,
      "sentence": "문제가 없음"
    }
  ]
}
```