

자연어처리를 이용한 청소기 리뷰

평가지표 개발 및 감성분석

NLP를 활용한 감성분석 프로젝트(1.8~1.14)

Work Team & Member

3팀(삼삼오오) 김찬희, 박유정, 정새하, 정한슬

Work Schedule

1.8~1.10 데이터 크롤링

1.11 데이터 전처리

1.12 감성사전 제작 및 감성분석

1.13 에러 해결, 점수화 및 시각화

Work Dataset

청소기 제품 40개의 리뷰 4000개

청소기 리뷰 분석을 통한 모델 비교 및 추천

리뷰에 포함된 평가요소와 이에 따른 감정분석

및 평가지표를 제공

시각적인 비교를 통해 모델 추천을 제공

Skills

PYTHON



40%

NLP



40%

Bs4



20%



00. 발표 순서

01

프로젝트 개요

02

코드 설명

03

결과 및 보완점

04

소감

프로젝트 목적 01

역할 분담 02

사용 기술 03

크롤링 04

전처리 05

감성사전 및 06

감성분석

점수화 및 시각화 07

결과 08

보완점 09

소감 10

01

프로젝트 개요

프로젝트 목적	01
역할 분담	02
사용 기술	03



01. 프로젝트 목적

★★★★★ 5 나우홈 · fkwp**** · 21.04.30. · 컬러: ALLNEW29000 폴라나이트그레이

1. 디자인 : 비슷한 가격대 비교했을때 가장 세련되고 이뻐서 샀어요. 2. 사용성 : 조립스위치의 재질은 저가 생 플라스틱이여서 품질이 좀 떨어지기는하나 기능상 빽빽함없이 잘 조립됩니다

1. 디자인 : 비슷한 가격대 비교했을때 가장 세련되고 이뻐서 샀어요.

2. 사용성 : 조립스위치의 재질은 저가 생 플라스틱이여서 품질이 좀 떨어지기는하나 기능상 빽빽함없이 잘 조립됩니다. 바퀴 부드럽게 잘 굴러가고, 부드럽게 잘 꺾입니다.

3. 무게 : 배터리와 모터 등 메인구성품들이 전부 손잡이 쪽에 있어서 무게감이 있습니다. 이건 구조상 어쩔수 없는 부분이라 판단되며 대신 거실장 아래 침대 밑 청소하기가 수월합니다. 무게감이 있다는 거지 결코 무겁지는 않습니다.

4. 배터리 : 이정도면 배터리양은 충분한 듯 합니다. 다만 전용 충전 도킹이 없는게 좀 아쉽습니다. 디자인은 이쁜데 선 꽂아놓으니영.. 모양새가 안나오더군요.

5. 흡입력 : 정밀하게 테스트 해본것은 아니지만 웬만한 집안 청소는 중간모드에 둬도 잘 됩니다. 여기서 오는 이점은 배터리도 오래 가고요.

6. 구성품 : 헤파필터 여분에, UV침구 키트 등 사용자가 필요한 구성을 잘 갖췄습니다.

7. 아쉬운 물걸레키트 : 률러 뒤에 부탁되어 진동기능이 있어서 쉽게 닦이는 구조인가 싶었는데, 그냥 물걸레입니다. 마찰이 어느정도 있기 때문에 힘이 필요합니다. 물이 마르지 않게 적당히 촉촉히 나오는 건 만족스러우나 힘줘서 빽빽 닦아야하는건 아쉽네요.

총평: 디자인, 흡입력, 풍족한구성, 넉넉한 배터리 만족스러우나 부분적으로 마감의 퀄리티가 떨어지는점, 물걸레 힘 줘서 닦아야 한다는 점이 아쉽습니다. 그래도 이정도면 가격 메리트 있다 생각합니다.

02. 역할 분담



김찬희

데이터 전처리
말뭉치 제작
감성사전 제작
PPT제작 및 발표
점수화



박유정

데이터 전처리
말뭉치 제작
감성사전 제작
불용어 제거
워드클라우드



정새하

셀레니움을 이용한 크롤러 제작
데이터 전처리
말뭉치 제작
감성사전 제작
단어 벡터화, 가중치 생성
에러 해결



정한슬

데이터 전처리
말뭉치 제작
감성사전 제작
시각화
점수화

03. 사용기술



<Jupyter Lab>



<KoNLPy>



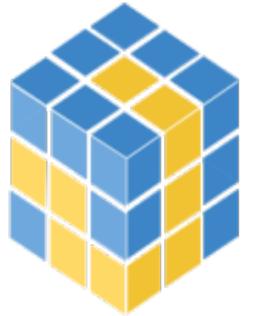
<Pandas>



<GitHub>



<BeautifulSoup>



NumPy

<NumPy>



<Scikits-learn>



<Selenium>

02

코드 설명

크롤링	04
전처리	05
감성사전 및	06
감성분석	
점수화 및 시각화	07



04. 크롤링

★★★★★ 5 나우홈 · fkwp**** · 21.04.30. · 컬러: ALLNEW29000 폴라나이트그레이

1.디자인 : 비슷한 가격대 비교했을때 가장 세련되고 이뻐서 샀어요. 2.사용성 : 조립스위치의 재질은 저가 ...

1.디자인 : 비슷한 가격대 비교했을때 가장 세련되고 이뻐서 샀어요.

2.사용성 : 조립스위치의 재질은 저가 생 플라스틱이어서 품질이 좀 떨어지기는 하나 기능상 빽빽함없이 잘 조립됩니다. 바퀴 부드럽게 잘 굴러가고, 부드럽게 잘 꺽입니다.

3. 무게 : 배터리와 모터 등 메인구성품들이 전부 손잡이 쪽에 있어서 무게감이 있습니다. 이건 구조상 어쩔수 없는 부분이라 판단되며 대신 거실장 아래 침대 밑 청소하기가 수월합니다. 무게감이 있다는 거지 결코 무겁지는 않습니다.

The screenshot shows the Google Chrome DevTools Elements tab. A tooltip at the top says "DevTools is now available in Korean!" with options to "Always match Chrome's language", "Switch DevTools to Korean", and "Don't show again". The Elements tab is selected. The DOM tree shows a list item with a class of "reviewItems_list_review_1sgc". Inside this list item, there is a link element with a class of "reviewItems_btn_area_2St26 reviewItems_btn_expand_3tN9_" and a href attribute of "#". The link has a color of #666 and a margin-top of 6px. The link is styled with a float: right; rule. The link is also styled with a text-decoration: none; rule. The link is styled with a color: -webkit-link; rule. The link is styled with a cursor: pointer; rule. The link is styled with a text-decoration: underline; rule. The link is styled with a user agent stylesheet rule.

```
.reviewItems_list_review_1sgc {928a117ec61..62d8b.css:1}
.reviewItems_btn_area_2St26 .reviewItems_btn_expand_3tN9_ {
  float: right;
  margin-top: 6px;
  color: #666;
}

a, a:active, a:focus, a:hover {96a564336ec..eb86c.css:1
  text-decoration: none;
}

a:-webkit-any-link {
  color: -webkit-link;
  cursor: pointer;
  text-decoration: underline;
}
```

04. 크롤링

```
from selenium import webdriver
from bs4 import BeautifulSoup
import pandas as pd

from selenium.webdriver.chrome.service import Service
from selenium.webdriver.common.by import By

driver_path = r"C:\Users\hp\chromedriver\chromedriver.exe"
driver = webdriver.Chrome(driver_path)
url = ('https://search.shopping.naver.com/catalog/21241320389?query=%EC%B2%AD%EC%86%8C%EA%B8%B0&NaPm=ct%3Dky81dr34%7Cci%3Dc1b5eb3167c2c1d670798a9c8961218ec79d8f01%7ctr%3Dsls1%7Csn%3D95694%7chk%3De0ea793f522165629b7f27f73416d3132cb47456')

driver.get(url)

html = driver.page_source
soup = BeautifulSoup(html, 'html.parser')

item_name = soup.select('#_next > div > div.style_container_3iYev > div.style_inner_1Eo2z > div.top_summary_title_15yAr > h2')[0].text
print(item_name)

price = soup.select('#_next > div > div.style_container_3iYev > div.style_inner_1Eo2z > div.style_content_wrap_2VTvX > div.style_content_36DCX > div > div.summary_info_area_3XT5U > div.lowestPrice_price_area_0kxBK > div.lowestPrice_low_price_fByaG > em')[0].text
print(price)
```

```
import time
review_data_list = []

for i in range(1,11):
    review_page_btn = '#section_review > div.pagination_pagination_2M9a4 > a:nth-child({num})'.format(num=i)
    print('클릭 확인', review_page_btn)

    driver.find_element_by_css_selector(review_page_btn).click()
    #driver.find_element(By.CSS_SELECTOR,review_page_btn).click()

    html = driver.page_source
    soup = BeautifulSoup(html, 'html.parser')

    #리뷰 목록
    review_list = soup.select('#section_review > ul')
    time.sleep(5)

    for review in review_list[0]:
        info = review.select('div.reviewItems_etc_area_2P8i3 > span')
        score = info[0].text
        channel = info[1].text
        date = info[3].text
        print('평점', info[0].text)
        print("채널", info[1].text)
        print("아이디", info[2].text)
        print('날짜', info[3].text)
        content = review.select('div.reviewItems_review_1eF8A > div > p')[0].text
        print('리뷰내용', content)
        review_data_list.append([price, score, channel, date, item_name, content])
```

04. 크롤링

A	B	C	D	E	F	G	H
1	가격	평점	채널	날짜	제품명	리뷰	
2	399,000	평점5	제이케이 구매대행	21.02.04.	해외다이슨 V7 플러피 차이슨 쓰다 고장나서 국내브랜드 구매후		
3	399,000	평점4	제이케이 구매대행	20.05.27.	해외다이슨 V7 플러피 첫날은 건부분을 누르면서 청소한다는게		
4	399,000	평점5	Neopeak	21.04.24.	해외다이슨 V7 플러피 일단 문의내용 답변이 해외직구인데도 잘		
5	399,000	평점1	비사이드커넥트	21.09.06.	해외다이슨 V7 플러피 플러피 새상품으로 주문하였으나, 구매 등		
6	399,000	평점5	제이케이 구매대행	19.12.07.	해외다이슨 V7 플러피 주문은 11월 25일 월요일에 했고 수령은		
7	399,000	평점5	제이케이 구매대행	20.06.01.	해외다이슨 V7 플러피 돼지코가 같이 도착안하고 다음날 따로 올		
8	399,000	평점4	제이케이 구매대행	20.05.31.	해외다이슨 V7 플러피 DC 모델 쓰다가 v7으로 바꿨습니다. DC		
9	399,000	평점5	제이케이 구매대행	20.04.17.	해외다이슨 V7 플러피 역시 명성다운 다이슨이네요. 청소는 아주		
10	399,000	평점5	제이케이 구매대행	20.09.02.	해외다이슨 V7 플러피 전에도 다른 모델의 다이슨을 사용했는데		
11	399,000	평점5	제이케이 구매대행	20.04.30.	해외다이슨 V7 플러피 4번째 청소기 구매라 굉장히 신중하게 선		
12	399,000	평점5	제이케이 구매대행	21.07.23.	해외다이슨 V7 플러피 일주일만에 배송됐습니다. *장점 1. 거친		
13	399,000	평점5	제이케이 구매대행	20.01.13.	해외다이슨 V7 플러피 돼지코를 못사서 완벽히 써보진 못했는데		
14	399,000	평점1	제이케이 구매대행	20.05.12.	해외다이슨 V7 플러피 3시간30분 충전 완료 하고 롤러헤드 장착		
15	399,000	평점5	제이케이 구매대행	20.11.12.	해외다이슨 V7 플러피 배송은 주말포함 15일정도 소요되었네요		
16	399,000	평점4	제이케이 구매대행	20.05.18.	해외다이슨 V7 플러피 배송이 느린게 문제가 아니고 (열흘걸렸을		
17	399,000	평점5	제이케이 구매대행	19.11.25.	해외다이슨 V7 플러피 배송은 12일? 거의 2주 걸렸어요. 박스가		
18	399,000	평점4	다루 스마트직구	20.08.27.	해외다이슨 V7 플러피 배송은 택배사 사정인지 생각보다는 오래		
19	399,000	평점5	제이케이 구매대행	20.04.23.	해외다이슨 V7 플러피 배송 기다리느라 힘들었지만 날짜 세보니		
20	399,000	평점4	제이케이 구매대행	20.09.09.	해외다이슨 V7 플러피 제가 다이슨 청소기 사용하는 거 보시고		
21	399,000	평점5	다루 스마트직구	21.02.02.	해외다이슨 V7 플러피 다른곳 주문했다가 열흘뒤 연락와서 물건		
22	399,000	평점5	제이케이 구매대행	21.02.04.	해외다이슨 V7 플러피 차이슨 쓰다 고장나서 국내브랜드 구매후		
23	399,000	평점4	제이케이 구매대행	20.05.27.	해외다이슨 V7 플러피 첫날은 건부분을 누르면서 청소한다는게		
24	399,000	평점5	Neopeak	21.04.24.	해외다이슨 V7 플러피 일단 문의내용 답변이 해외직구인데도 잘		
25	399,000	평점4	비사이드커넥트	21.09.06.	해외다이슨 V7 플러피 차이슨 쓰다 고장나서 국내브랜드 구매후		

-초기-

총 40개의 모델

청소기 모델 1개당 200개씩
8000개의 리뷰 수집

-변경-

01후 문장 전처리 단계에서

문장 처리에 끌어로

처리시간이 길어져

50%인 모델당 100개씩으로
수정

05. 데이터 전처리

```
dyson_data= pd.read_csv('./data/다이슨.csv',encoding = 'cp949')

dyson_list = []
dyson_dict = {}
dyson_star = dyson_data['평점'].values
dyson_model_name = dyson_data.drop_duplicates(['제품명'])['제품명']
dyson_model_cost = dyson_data.drop_duplicates(['가격'])['가격']

# 해당 제조사의 제품숫자로 변경
for i in range(5):

    dyson_dict = {}
    dyson_name = dyson_model_name.iloc[i]
    dyson_cost = dyson_model_cost.iloc[i]

    dyson_test_1 = dyson_data['제품명'] == dyson_name
    dyson_test = dyson_data[dyson_test_1]
    dyson_star = dyson_test['평점'].tolist()[:100]
    dyson_channel = dyson_test['채널'].tolist()[:100]
    dyson_date = dyson_test['날짜'].tolist()[:100]
    dyson_review = dyson_test['리뷰'].tolist()[:100]

    dyson_dict['제품명'] = dyson_name
    dyson_dict['가격'] = dyson_cost
    dyson_dict['평점'] = dyson_star
    dyson_dict['채널'] = dyson_channel
    dyson_dict['날짜'] = dyson_date
    dyson_dict['리뷰'] = dyson_review

    dyson_list.append(dyson_dict)

dyson_list.append(dyson_dict)
```

제조사 별로

[{'모델':'a','가격':'b',
'평점':['2','4'],
'채널':['ㄱ','ㄴ','ㄹ'],
'날짜':['2','1','6'],
'리뷰':[!,#,%,...]}],{...},...]

형태의 리스트 생성

05. 데이터 전처리

#말뭉치 생성

```
import chardet
spacing = Spacing()
def do_stemming(reviews):
    total_review = ''

    for idx in range(len(reviews)):
        review_text = reviews[idx]
        # if chardet.detect(review_text.encode())['encoding'] == 'cp949':
        #     review_text = unicode(review_text, 'cp949').encode('utf-8')

        #한국의 리뷰에서 문장 단위로 자르기
        for sentence in kkma.sentences(review_text):
            # if chardet.detect(sentence.encode())['encoding'] == 'cp949':
            #     sentence = unicode(sentence, 'cp949').encode('utf-8')

            sentence = re.sub('([a-zA-Z])', '', sentence)
            sentence = re.sub('[ㄱ-ㅎㅏ-ㅣ]+', '', sentence)
            sentence = re.sub('[-=+,#/\\?:^$.@*\"※~&%·!„\\\\‘|(\\)\\[\\]\\\\<\\>`\\...]', '', sentence)

            if len(sentence) == 0:
                continue
            if len(sentence) < 198:
                sentence = spacing(sentence)
                sentence += '.'
            total_review += sentence

    total_review = okt.normalize(total_review)

return total_review
```

1. 리뷰를 문장 단위로 분리
2. 영어와 'ㅋㅋ', 'ㅎㅎ'같은 한글 자모음,
그리고 !,# 와 같은 특수문자를 제거
3. kospacing을 통해서 띄어쓰기 교정
4. okt. normalize를 통해 맞춤법 교정
5. 이후 제조사 별 전처리 과정을 통해
얻어진 데이터를 json파일로 저장 후
추합

06. 감성사전 및 감성분석

```
weight_good_feature = {'무게': ['가볍', '가벼', '적당', '괜찮', '좋', '무난', '없']}
weight_bad_feature = {'무게': ['무거', '무겁', '나가', '있']}
power_good_feature = {'흡입': ['좋', '강', '충분', '만족', '괜찮', '잘', '짱', '적당']}
power_bad_feature = {'흡입': ['나쁘', '별로', '약', '부족', '아직', '장난']}
charge_good_feature = {'충전': ['짧', '빨리', '급속', '대박', '빠른', '빠르게', '번거', '잘', '편하', '편한', '쉬운', '쉽']}
charge_bad_feature = {'충전': ['오래', '길', '짜증', '느', '불편', '어렵', '아쉽']}

design_good_feature = { '디자인': ['예뻐', '이쁜', '이뻐', '예쁜', '깔끔', '세련', '고급', '반해', '군더더기', '잘', '스며드는', '굿', '무난', '색상': ['예뻐', '이쁜', '이뻐', '예쁜', '깔끔', '세련', '고급', '반해', '군더더기', '잘', '스며드는', '좋', '괜찮', '무난']},
design_bad_feature = { '디자인': ['못', '실망', '별로', '구리', '나빠', '나쁜', '이상', '투박', '별로'],
    '색상': ['못', '실망', '별로', '구리', '나빠', '나쁜', '이상', '투박', '별로'] },
cost_effectiveness_good_feature = { '가성': ['괜찮', '괜춘', '좋은', '최고', '굿', '추천', '만족', '짱', '훌륭', '있는'],
    '가성 비': ['괜찮', '괜춘', '좋은', '최고', '굿', '추천', '만족', '짱', '훌륭', '있는'],
    '가격 대비': ['괜찮', '괜춘', '좋아', '최고', '굿', '추천', '만족', '짱', '훌륭'] }
cost_effectiveness_bad_feature = { '가성': ['별로', '실망', '그닥', '없는', '비추', '나빠', '나쁜', '글쎄', '떨어'],
    '가성 비': ['별로', '실망', '그닥', '없는', '비추', '나빠', '나쁜', '글쎄', '떨어'],
    '가격 대비': ['별로', '실망', '그닥', '비추', '나쁜', '나빠', '글쎄', '떨어'] }

negative_word_emotion = ['안', '않', '못', '없', '아닌', '아니']
```

06. 감성사전 및 감성분석

```
def get_feature_keywords(feature_keywords, review):
    feature_temp = []
    for keyword in feature_keywords: #감성 사전키 리스트에서 키를 뽑아서
        if re.findall(keyword, review): #패턴을 찾음
            sub_list = ['개','고','음','며','데','만','도','면']

            for sub in sub_list:
                if sub+' ' in review: #리뷰안에 sub +' ' 가 있으면
                    review = re.sub(sub+' ', sub+',', review) # review 안에 sub+' '를 sub+','로 치환

                a = re.findall(keyword + '[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+\\s+[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+\\s+[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+',review)
                b = re.findall(keyword + '\\s+[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+\\s+[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+',review)
                c = re.findall('[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+\\s+' + keyword + '[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+',review) |
                d = re.findall('[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+\\s+' + keyword + '\\s+[ㄱ-ㅎ|ㅏ-ㅣ|가-핳]+',review)

                for ngram in a: #패턴찾기로 찾아낸 문구들을 (문구, 키워드) 형태로 만드는 ['소음이 커요', '소음이 작아요','소음만 어찌했으면']
                    t = ()
                    feature_temp.append(t + (ngram,keyword)) # ('소음이 커요', '소음')
                for ngram in b:
                    t = ()
                    feature_temp.append(t + (ngram,keyword))
                for ngram in c:
                    t = ()
                    feature_temp.append(t + (ngram,keyword))
                for ngram in d:
                    t = ()
                    feature_temp.append(t + (ngram,keyword))

    return feature_temp
```

06. 감성사전 및 감성분석

```
def get_feature_emotions(feature_good_dict, feature_bad_dict, feature_temp):
    good_feature_emotion_list = []
    bad_feature_emotion_list = []

    for ngrams in feature_temp: #(문구, 키워드) 형태
        keyword = ngrams[1]
        ngram = ngrams[0]
        is_bad_feature = None

        good_emotion_list = feature_good_dict[keyword] #키워드(key)를 이용해서 value 가져옴
        bad_emotion_list = feature_bad_dict[keyword]

        for emotion in good_emotion_list:
            if re.findall(emotion, ngram): # 문구에서 가지고 온 value 리스트에서 단어 빼와서 패턴 찾기
                is_bad_feature = False
        for emotion in bad_emotion_list:
            if re.findall(emotion, ngram):
                is_bad_feature = True
        for negative in negative_word_emotion: # 문구안에 부정적인 워드 있는지 확인
            if re.findall(negative, ngram):
                if is_bad_feature == True:
                    is_bad_feature = False
                    break
                elif is_bad_feature == False:
                    is_bad_feature = True
                    break
            else:
                is_bad_feature = True
                break
        if is_bad_feature:
            bad_feature_emotion_list.append(ngram)
        elif is_bad_feature == False:
            good_feature_emotion_list.append(ngram)
        else:
            pass
    return good_feature_emotion_list, bad_feature_emotion_list
```

07. 점수화 및 시각화

목표: 카테고리에서 중요하게 생각하는 키워드의 가중치를 구해서 점수화에 반영이 목표
HOW? 단어 빈도수를 이용(BOW 모델)

BOW(Bag of Words) : 단어들의 문맥이나 순서를 무시하고, 단어들에 대해 빈도 값(frequency)을 부여해
II쳐 값을 만드는 모델

BOW II처 벡터화 방법 - CounterVectorizer

- TF-IDF
- CounterVectorizer : 각 텍스트에서 단어 출현 횟수를 카운팅한 벡터
- TF-IDF : 단어의 빈도를 의미하는 Term Frequency와 문서 빈도(Document Frequency)의 역수인 I
DF(Inverse Document Frequency) 합성어,

어떤 단어가 특정 문서 내에서 얼마나 중요한 것인지를 나타내는 통계적 수치

⇒ 빈도수만으로는 중요한 단어라 보기 어려워 CounterVectorizer를 이용해 단어 카운트로 벡터화 한 후
TF-IDF 변환해서 각 단어들의 중요도를 구함

07. 점수화 및 시각화

```
#CountVectorizer
def text_cleaning(text):
    hangul = re.compile('[^ㄱ-ㅣ 가-힣]') # 정규 표현식 처리
    result = hangul.sub('', text)
    okt = Okt() # 형태소 추출
    nouns = okt.nouns(text)
    nouns = [x for x in nouns if len(x) > 1] # 한글자 키워드 제거
    nouns = [x for x in nouns if x not in stopwords] # 놀음어 제거
    return nouns

vect = CountVectorizer(tokenizer = lambda x: text_cleaning(x))
bow_vect = vect.fit_transform(all_cleaner_review_list) #각 단어가 리뷰별 등장 횟수(2차원 배열)
word_list = vect.get_feature_names() # 단어 리스트
count_list = bow_vect.toarray().sum(axis=0) # 단어 빈도수
```

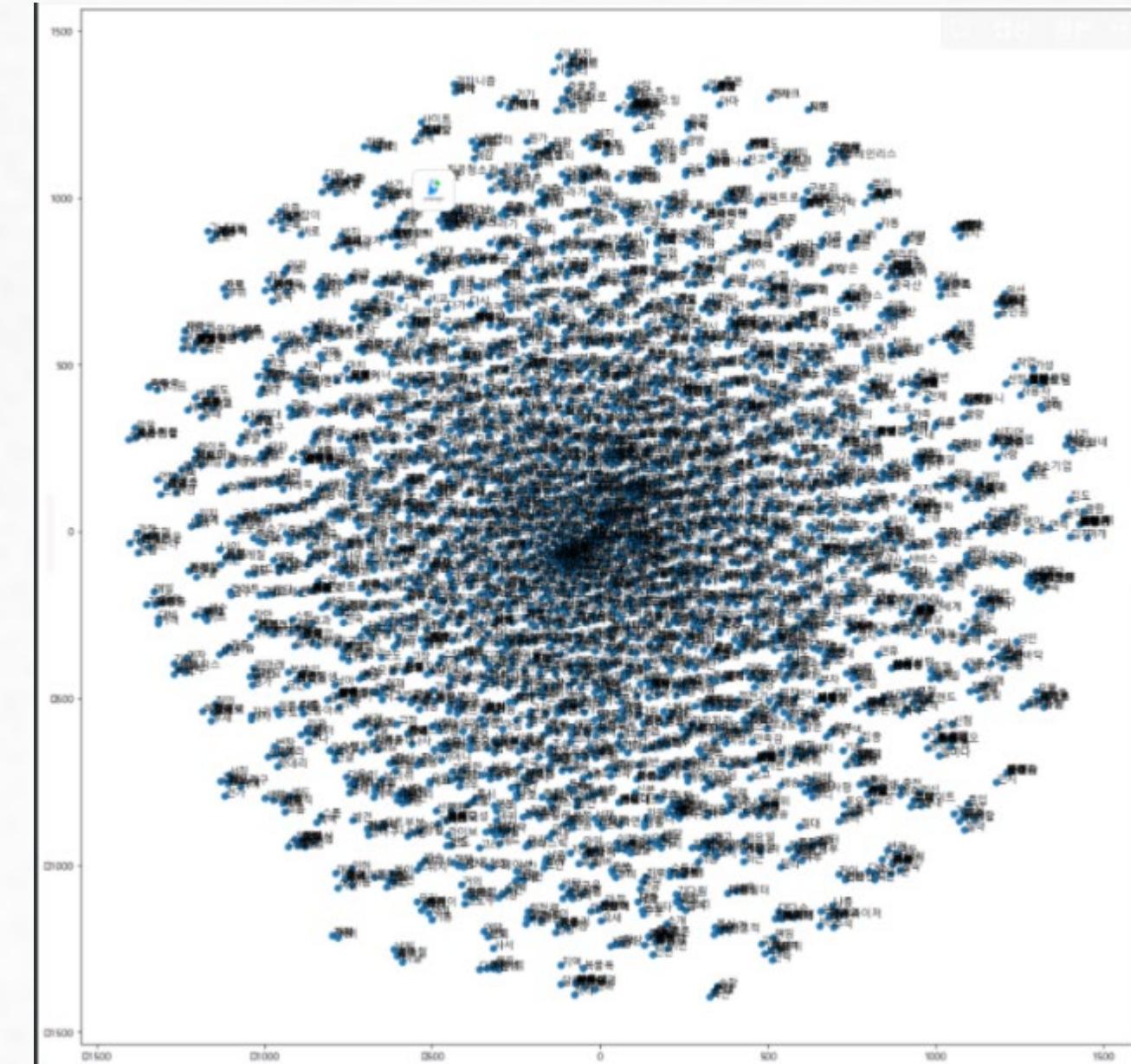
```
#TF-IDF로 변환
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.metrics.pairwise import linear_kernel

tfidf_vectorizer = TfidfTransformer()
tf_idf_vect = tfidf_vectorizer.fit_transform(bow_vect)

print(tf_idf_vect.toarray())
print(tf_idf_vect.toarray().shape)
```

07. 점수화 및 시각화

단어	중요도
1588 사용	62.364272
3765 흡입	49.891892
1056 먼지	42.002885
1 가격	37.563972
314 구매	37.331979
1308 배송	32.281591
1670 생각	30.970608
1147 무선	30.010101
1174 물걸레	28.815689
3238 충전	26.520421
1784 소음	24.955866
2870 정도	24.773706
1135 무게	22.411627



TSNE를 통해 시각화

07. 점수화 및 시각화

많은 키워드 안에서 관계성을 찾기에는
시간과, 상위의 것들만 취급했을 때
반영 여부가 불확실

=> 앞에서 정해둔 각 카테고리에
해당하는 키워드만을 추출하기로 함



1 삼성전자 비스포크제트 VS20A956A3
Good noise: 4/15, 긍정률: 27.0
Good battery: 3/8, 긍정률: 38.0
Good price: 16/20, 긍정률: 80.0
Good power: 17/29, 긍정률: 59.0
Good weight: 2/11, 긍정률: 18.0
Good charge: 1/4, 긍정률: 25.0
Good durability : 0/0, 긍정률: 0
Good service : 1/1, 긍정률: 100.0
Good design : 16/19, 긍정률: 84.0
Good cost eff : 1/1, 긍정률: 100.0
총점: 11560.92990169276

2 삼성전자 삼성 VC-H22
Good noise: 4/12, 긍정률: 33.0
Good battery: 2/2, 긍정률: 100.0
Good price: 9/9, 긍정률: 100.0
Good power: 14/24, 긍정률: 57.99999999999999
Good weight: 0/1, 긍정률: 0.0
Good charge: 1/3, 긍정률: 33.0
Good durability : 0/0, 긍정률: 0
Good service : 0/0, 긍정률: 0
Good design : 2/2, 긍정률: 100.0
Good cost eff : 3/3, 긍정률: 100.0
총점: 12735.594135691637

3 삼성전자 삼성 VC-H71
Good noise: 2/3, 긍정률: 67.0
Good battery: 0/0, 긍정률: 0
Good price: 5/5, 긍정률: 100.0
Good power: 7/10, 긍정률: 70.0
Good weight: 0/0, 긍정률: 0
Good charge: 0/5, 긍정률: 0.0
Good durability : 0/0, 긍정률: 0
Good service : 0/0, 긍정률: 0
Good design : 1/1, 긍정률: 100.0
Good cost eff : 2/2, 긍정률: 100.0
총점: 13887.711316494766

07. 점수화 및 시각화

```
#가중치 리스트 업데이트
for i , point in enumerate(w_point_list):
    w_point_list[i] = first_point[i]

#categorysDF.reset_index()
#정규화로 값 좀더 다음은
from sklearn.preprocessing import MinMaxScaler
robustScaler = MinMaxScaler()
print(robustScaler.fit(categorysDF))
train_data_robustScaled = robustScaler.transform(categorysDF)

MinMaxScaler()

df_scaled_df = pd.DataFrame(train_data_robustScaled)
df_scale_df_list = df_scaled_df[0].tolist()
df_scaled_df
```

총점 수치가 너무 커서 정규화
(**MinMaxScaler**)를 통해 값을
0~1 사이 값으로 조정

0	0.437938
1	1.000000
2	0.521977
3	0.259217
4	0.334583
5	0.459584
6	0.179977
7	0.747853
8	0.000000
9	0.103346

07. 점수화 및 시각화

```
check_division = lambda x, y: y == 0 else round((x / y), 2)
num = 0
score_list = []
company_score_list = []
temp = []

for k, company in enumerate(json_data):
    company_score_list.append(temp)
    temp = []

    for i, item in enumerate(company):
        cleaner_good_noise_count = 0
        cleaner_bad_noise_count = 0
        num += 1

        print(num, item['제품명'])

        nois_temp = get_feature_keywords(noise_good_feature.keys(), item['리뷰'])
        good_noise, bad_noise = get_feature_emotions(noise_good_feature, noise_bad_feature, nois_temp)
        cleaner_good_noise_count = len(good_noise)
        cleaner_bad_noise_count = len(bad_noise)

        noise_rate = 100 * check_division(cleaner_good_noise_count, cleaner_good_noise_count + cleaner_bad_noise_count)
        print('Good noise: {} / {}, 긍정률: {}'.format(cleaner_good_noise_count, cleaner_good_noise_count + cleaner_bad_noise_count, noise_rate))

        if cleaner_good_noise_count + cleaner_bad_noise_count == 0:
            noise_rate = 50

score = w_point_list[5] * noise_rate + w_point_list[6] * battery_rate + w_point_list[7] * price_rate + w_point_list[1] * power_rate + w_point_list[0] * weight_rate + \
w_point_list[2] * charge_rate + w_point_list[8] * dura_rate + w_point_list[9] * servi_rate + w_point_list[3] * design_rate + w_point_list[4] * cost_eff_rate
```

07. 점수화 및 시각화

1 삼성전자 비스포크제트 VS20A956A3

Good noise: 4/15, 금점률: 27.0
Good battery: 3/8, 금점률: 38.0
Good price: 16/20, 금점률: 80.0
Good power: 17/29, 금점률: 59.0
Good weight: 2/11, 금점률: 18.0
Good charge: 1/4, 금점률: 25.0
Good durability : 0/0, 금점률: 0
Good service : 1/1, 금점률: 100.0
Good design : 16/19, 금점률: 84.0
Good cost eff : 1/1, 금점률: 100.0
총점: 224.57568058218402

2 삼성전자 삼성 VC-H22

Good noise: 4/12, 금점률: 33.0
Good battery: 2/2, 금점률: 100.0
Good price: 9/9, 금점률: 100.0
Good power: 14/24, 금점률: 57.99999999999999
Good weight: 0/1, 금점률: 0.0
Good charge: 1/3, 금점률: 33.0
Good durability : 0/0, 금점률: 0
Good service : 0/0, 금점률: 0
Good design : 2/2, 금점률: 100.0
Good cost eff : 3/3, 금점률: 100.0
총점: 247.72193632482157

3 삼성전자 삼성 VC-H71

Good noise: 2/3, 금점률: 67.0
Good battery: 0/0, 금점률: 0
Good price: 5/5, 금점률: 100.0
Good power: 7/10, 금점률: 70.0
Good weight: 0/0, 금점률: 0
Good charge: 0/5, 금점률: 0.0
Good durability : 0/0, 금점률: 0
Good service : 0/0, 금점률: 0
Good design : 1/1, 금점률: 100.0
Good cost eff : 2/2, 금점률: 100.0
총점: 271.02062934309686

07. 점수화 및 시각화

json파일의 리뷰를 합산해
하나의 말뭉치 생성

korean_stopwords_.txt

불용어 사전 가져오기

('청소기','제품','청소','사용')

불용어 사전에 추가

불용어 제거 후 명사만 추출

한글자 단어 제거(것, 못, 건 등등)

```
total_review_corpus = ''  
for k, company in enumerate(json_data):  
    for i, item in enumerate(company):  
        total_review_corpus += item['리뷰']
```

```
file_path = './data/korean_stopwords_.txt'  
  
with open(file_path) as f:  
    stopwords = f.readlines()  
  
stopwords = [line.rstrip('\n') for line in stopwords]  
  
def text_preprocessing(text,okt):  
  
    total_stopwords = stopwords  
    txt = re.sub('[^가-힣a-z]', ' ', text)  
    token = okt.nouns(txt)  
    token = [t for t in token if t not in stopwords]  
  
    return token  
  
total= text_preprocessing(total_review_corpus,okt)
```

```
available_counter = Counter({x: counter[x] for x in counter if len(x) > 1})
```

07. 점수화 및 시각화

```
rank = sorted(score_list, key = lambda x : -x[1])
```

```
df=pd.DataFrame((zip(x, y)), columns = [ 'Name', 'Score'])  
df
```

	Name	Score
0	LG전자 A9S 오브제컬렉션 AO9471WKT	295.611321
1	다이슨 옴니 글라이드 컴플리트 플러스	290.023357
2	쿠쿠홈시스 쿠쿠 CVC-A1420UG	283.161925
3	다이슨 V10 카본파이버	279.758412
4	데탈 TY6545KL	278.744521
35	신일전자 SVC-8858PNX	195.097987
36	다이슨 V15 디텍트 컴플리트	195.061202
37	다이슨 V11 220 에어와트 CF+	194.365402
38	비브르 스톰파워 VE30	193.693700
39	куку홈시스 쿠쿠 CVC-A1410UG	171.927228

07. 점수화 및 시각화

```
import matplotlib.pyplot as plt  
  
x=df_sorted_by_values['단어'][:10]  
y=df_sorted_by_values['중요도'][:10]  
  
plt.bar(x,y)  
plt.show()
```

```
x= df2['Name']  
y= df2['Score']  
  
#종 브랜드, 제품들 중에 상위권 5개의 제품과 그 점수 시각화  
plt.figure(figsize=(5,5))  
plt.bar(x,y)  
plt.xlabel('Name',fontsize=15)  
plt.ylabel('Score', fontsize=15)  
plt.xticks(fontsize= 10, rotation =30)  
plt.show()
```

데이터 프레임 슬라이싱을 통해
각 모델별 점수 시각화

03

결과 및 보완점

결과	08
보완점	09

08. 결과

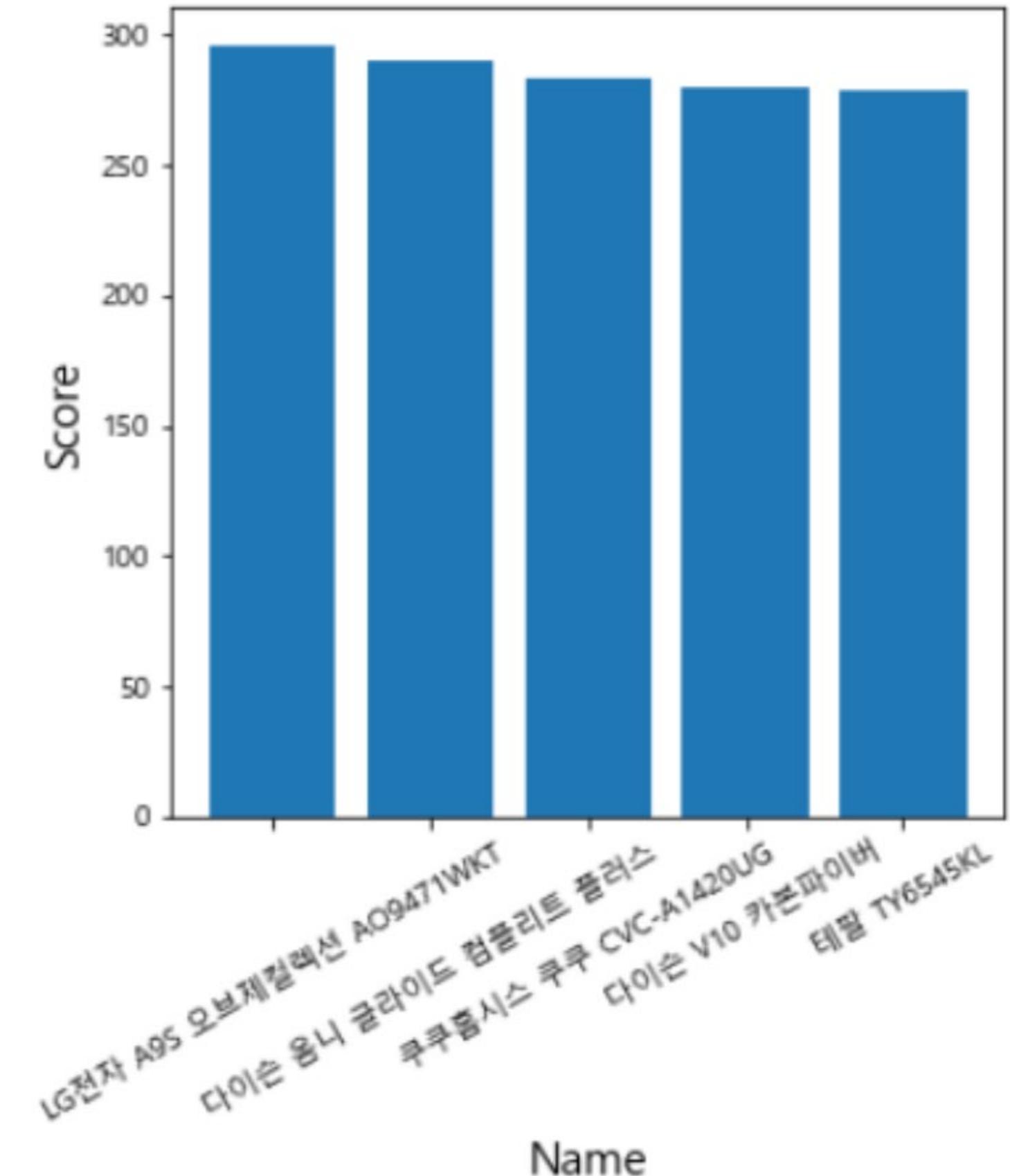
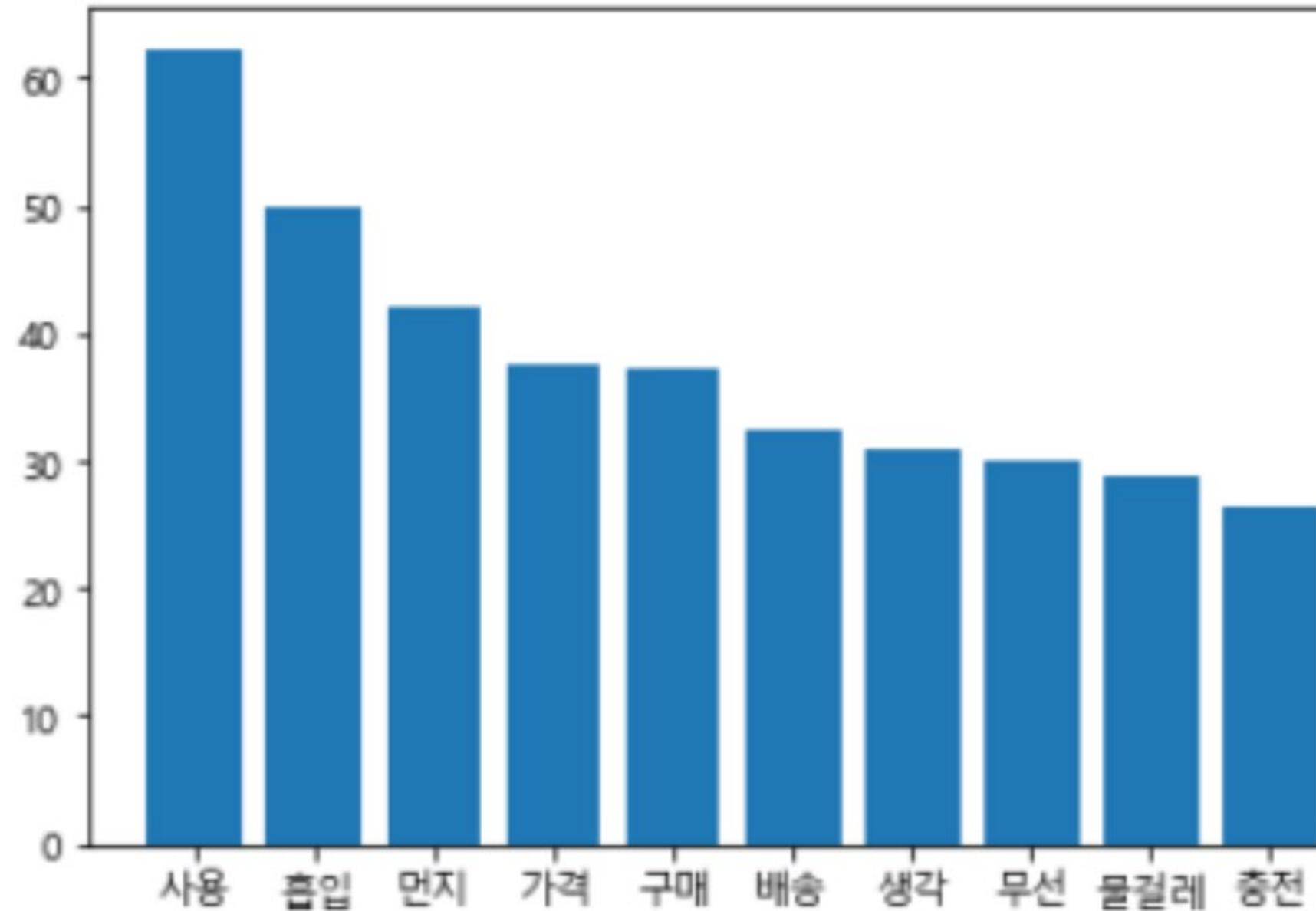
<불용어 추가 전>



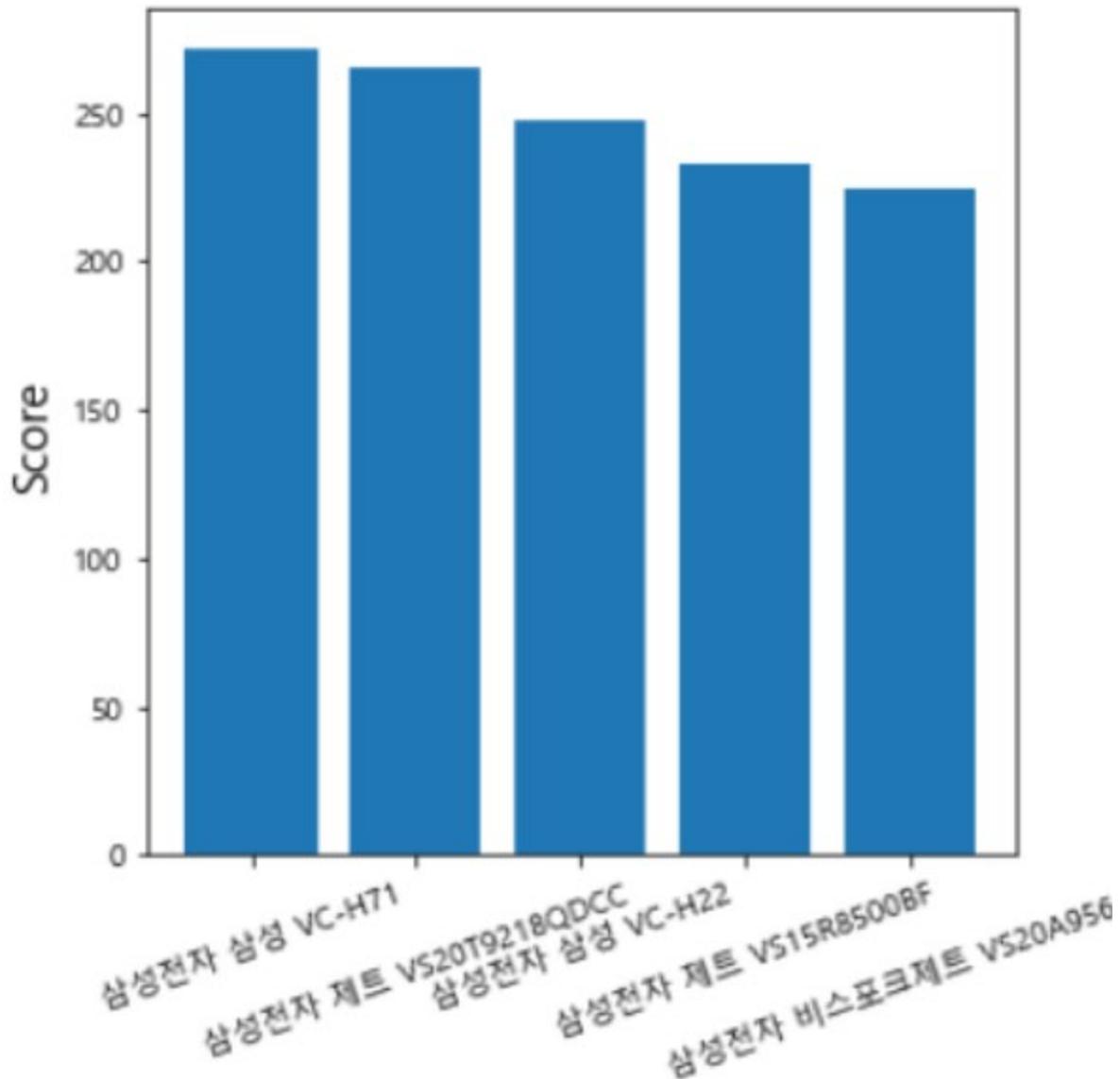
<불용어 추가 후>



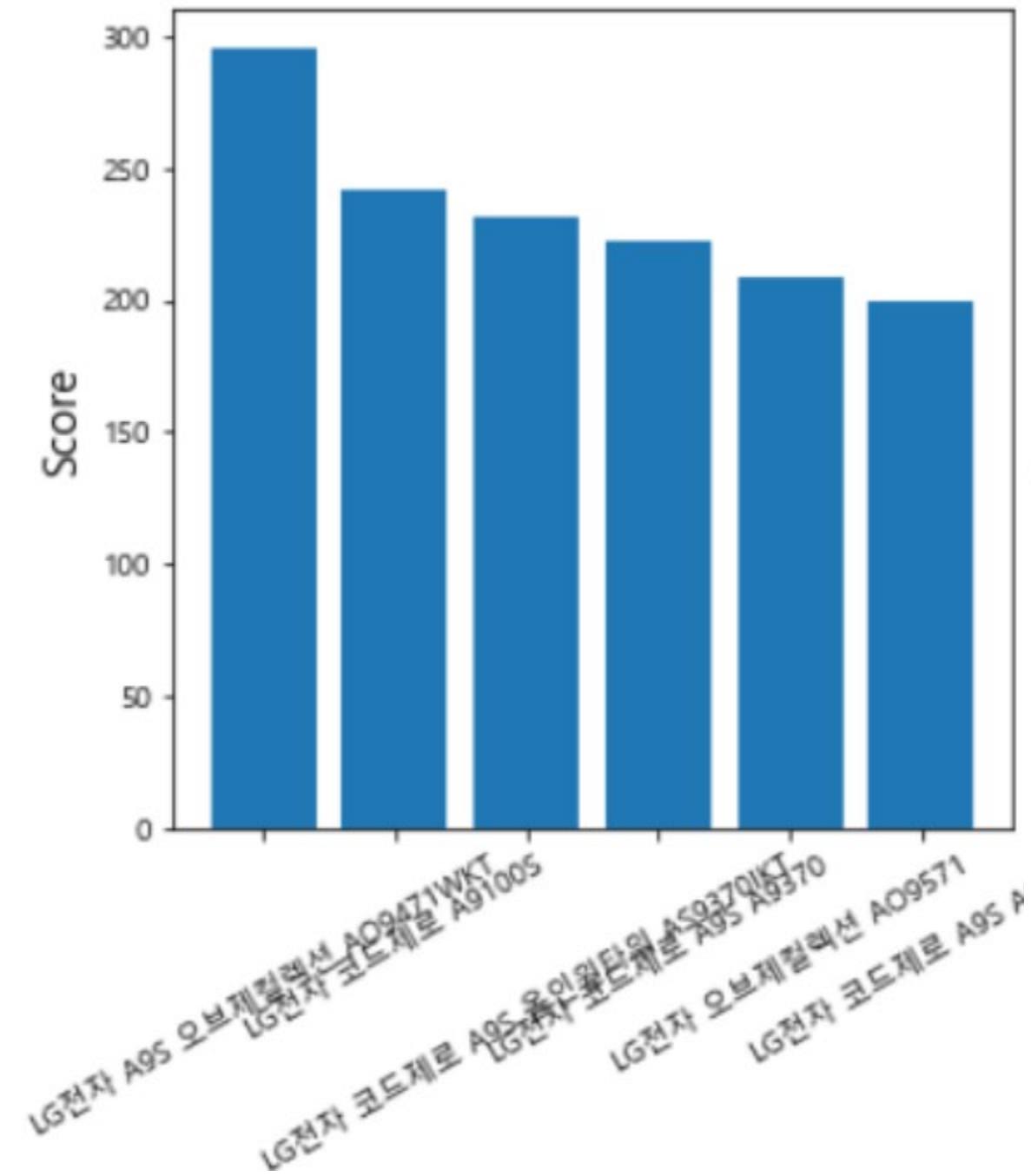
08. 결과



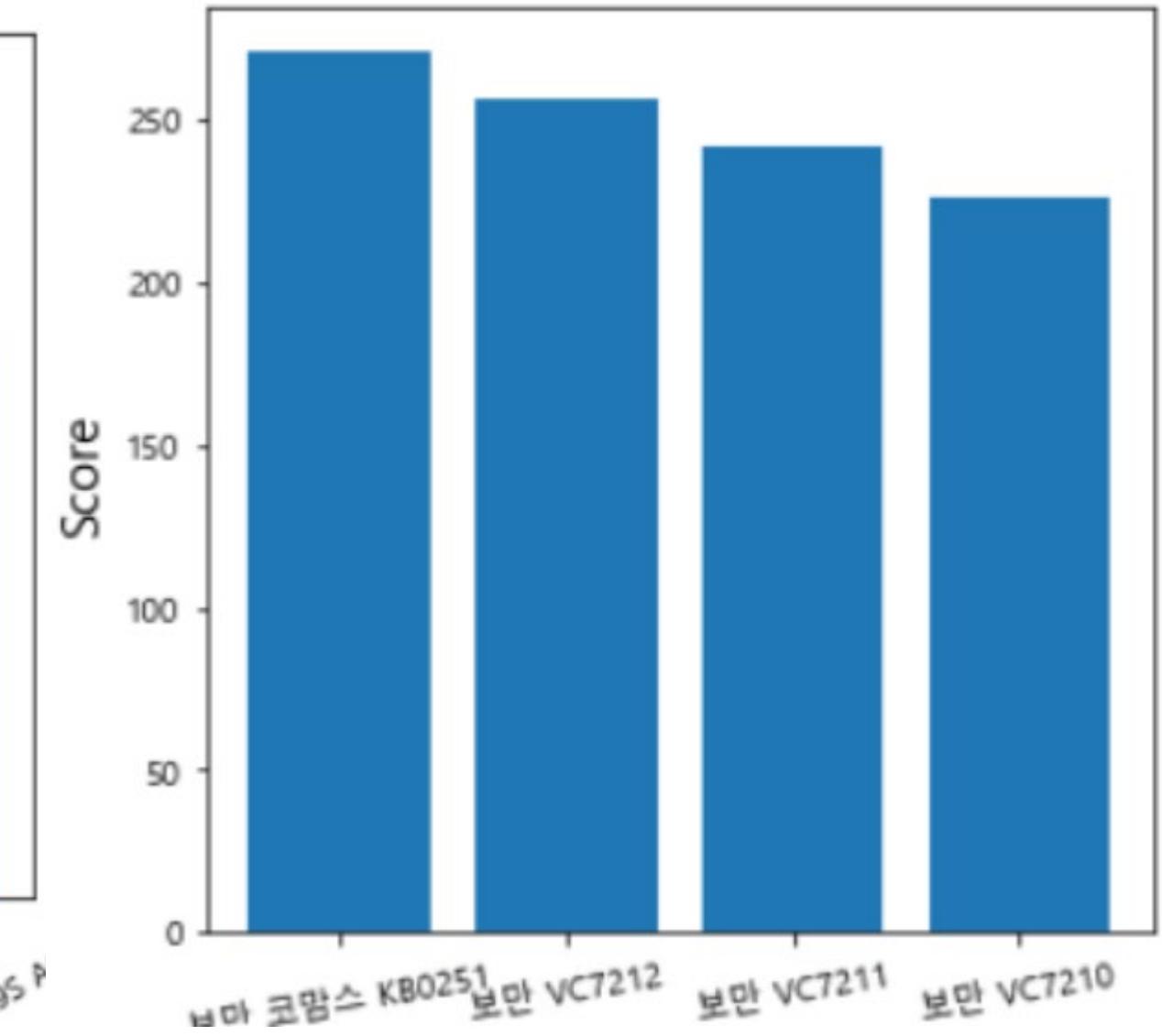
08. 결과



<삼성>



<엘지>



<보만>

09. 보완점

부족한 데이터 양

자연어 전처리와 예상해결에 시간이
많이 걸려 많은 데이터를 사용하지 못함

협업 도구 사용의 미숙함

git을 이용하는 주피터 랩, 파이참
그리고 코랩 사용법 숙지 필요성

04

소감

소감

..... 10

10. 소감

김찬희

자연어 처리를 하는데에는 고려사항도 많고 데이터가 많아졌을 때 처리시간이 길어져서 힘들기는 했지만 여러 문제들을 해결하고 결과물을 도출했을 때 들인 노력 이상의 성취감을 얻을 수 있었습니다.

이후에는 자연어처리에 딥러닝 모델을 추가해서 학습시키는 것도 해보고 싶다는 생각이 들었습니다.
같이 진행한 팀원들 수고하셨습니다!

박유정

다른 분야보다 본인의 주관과 판단이 많이 필요해서 힘들었습니다. 또한 예러가 발생해도 한국어라 나오는 자료가 많이 없어서 어려움을 겪었습니다. 가장 어려우면서도 가장 재미있었습니다. NLP는 프로세스 하나하나가 다 고비였는데 팀원들이 없었으면 이겨내지 못했을 것입니다.
팀원들에게 누구보다 수고했다고 말해주고 싶습니다.

정새하

자연어 처리를 막연하게 어렵게만 생각했었는데 해보면서 조금은 가닥을 잡을 수 있었던 계기였습니다. 처음에는 할 수 있을까란 걱정도 있었지만 팀원들과 함께 오류를 해결하고 생각을 공유면서 잘 해나갈 수 있었습니다. 같이 성장할 수 있는 소중한 시간이었습니다. 이런 저런 오류속에서도 같이 힘내서 열심히 한 팀원들에게 고맙고 고생했다고 말해주고 싶습니다.

정한슬

한글 자연어 처리가 어려운 것은 알고는 있었지만 실제로 해보니 표현과 쓰는 방법이 다양해서 분석하는데 어려움이 있었습니다. 하지만 그것을 찾아내서 분리를 할 수 있을 때 재밌었습니다. 또한 조원들이 다같이 힘을 합쳐 각 제품에 대한 점수도 매길 수 있어서 뿌듯한 프로젝트였습니다.
앞으로 더 조원들에게 도움이 되는 팀원이 되고 싶습니다.

THANK YOU

감사합니다.