



웹툰 인사이트

Text analysis to confirm trends of webtoon



01

INTRO

02

PROCESS

03

PART

04

RESULT

05

ASSESSMENT

개요

수행 절차

역할 분담

수행 결과

자체 평가 및 의견

프로젝트 개요

TEAM | ALPHA911

분석할 데이터가 충분한지

분석 실현이 가능한지

분석 후 인사이트가 있을지

심화 학습이 가능한지

BOMTOON • 웹툰 만화 소설 플레이

오늘봄툰 연재 랭킹 무료

월 화 수 목 금 토 일 열흘 완결

장르 작품유형

네리 Ⓜ 2.7만 명이나물
동아리방에 귀신 나온대 Ⓜ 13.6만 향지
타치바나 백작 Ⓜ 17.9만 활기
빙의물 여주의 언니입니다 Ⓜ 8.3만 한설미&백일몽

이웃집 길드원 Ⓜ 58.7만 ANGELA&비...
키메라 하트 Ⓜ 3.1만 NANO
안녕, 나의 폐하 Ⓜ 2.9만 W.Y
집에 가는 길에 USB를 ... Ⓜ 5.7만 백조아시&숙연

BOMTOON • 웹툰 만화 소설 플레이

이웃집 길드원

ANGELA&비치&하니트랩

전체보기 #BL #수요연재 #현대문 #코믹/개그 #귀여움 #삼설/오해

▶ 미친 것 같아(positive) (~4/10)

◀ <이웃집 길드원> 소설 보러가기

58.7만 1.2만 2,922 >

질금을봄툰에누자승 BEST
우리 치구면, 연상 쓰래기. 아파쓰나 했는데 나쁜 길드원 빠우는데나 쓰고있대. 잘한다 퍼사 가시 이세!!
이웃집 길드원: 11화 461

2023.08.08

첫화 무료 보기

대여하기 소장하기

전체 첫화부터

1화 2023.06.21 무료
2화 2023.06.21 오픈인 무료

작품설명

스토리에게 시달린 뒤 사람을 지독하게 경계하는 문지구. 때마침 이웃집에 성적 취향이 이상한 남자가 이사를 왔다...? “밀집에 변태 세끼가 사는 것 같아여...” 현실에서는 오해 가득, 수상한 이웃집 이

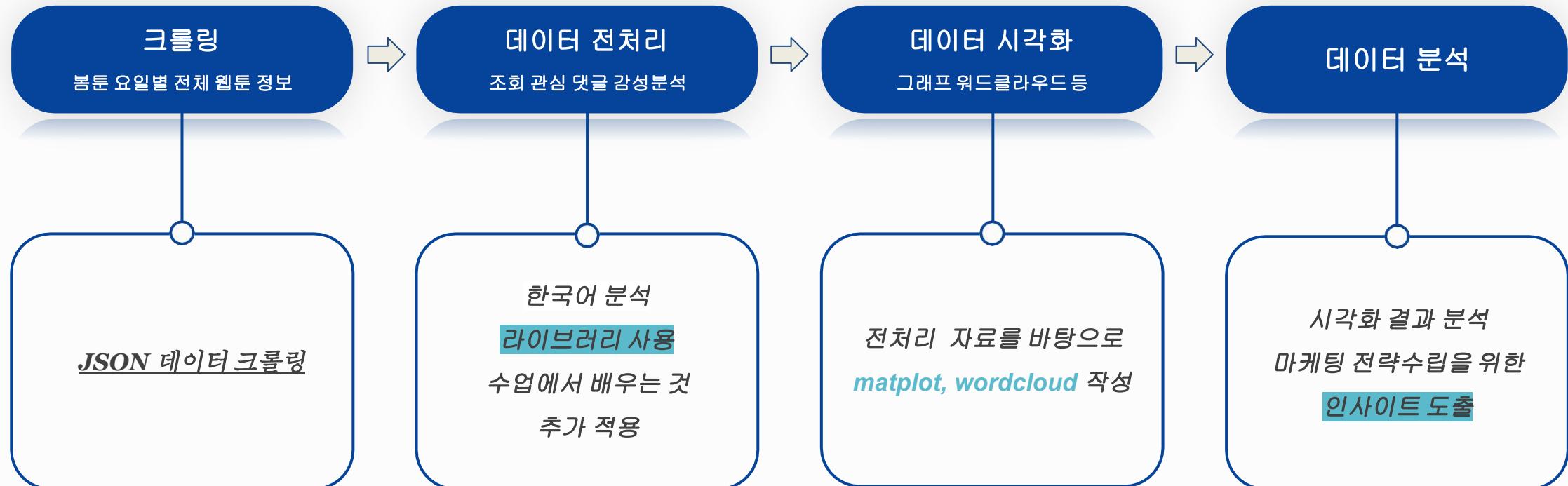
더 펼쳐보기 ▾

하니트랩 작가의 다른 작품

전체 대여하기

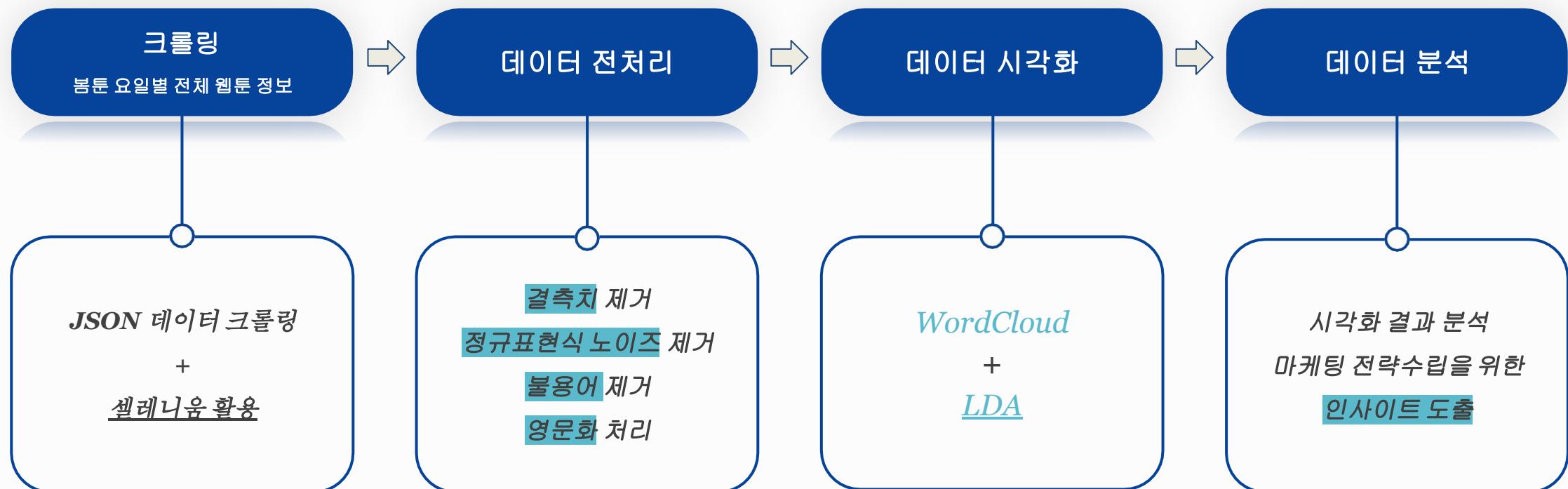
프로젝트 수행절차

프로젝트 예상 시나리오



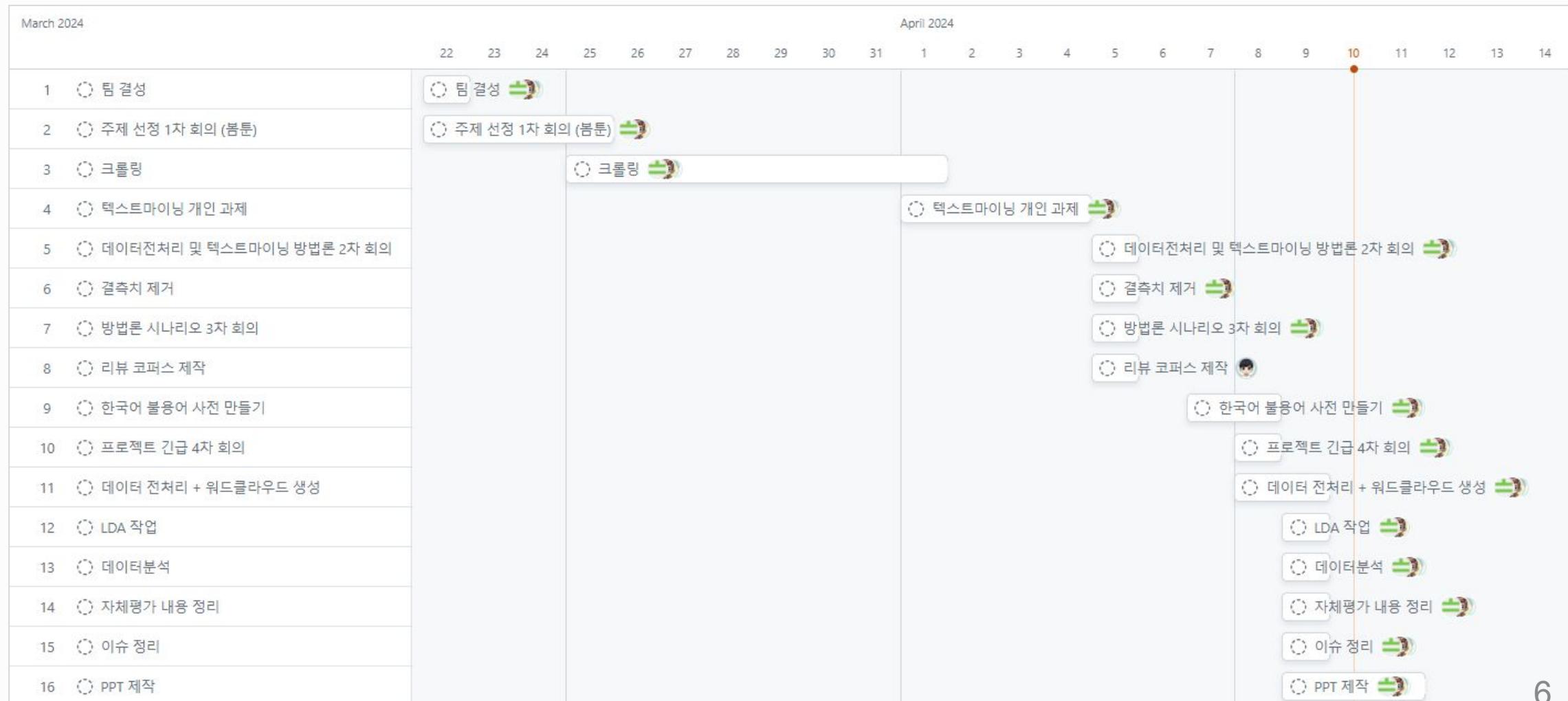
프로젝트 수행절차

프로젝트 실제 시나리오



프로젝트 수행절차

프로젝트 수행 WPT(Work Plan Timeline)



프로젝트 수행절차

프로젝트 수행 WPT(Work Plan Timeline)

구분	기간	활동	비고
팀 결성	3/22(금)		
주제선정 1차 회의 (봄툰)	3/22(금) ~ 3/25(월)	프로젝트 주제선정, 기획안 정리	다수결 주제선정
크롤링	3/25(월) ~ 4/1(월)	‘봄툰’ 댓글 크롤링 및 코드 작성	Json Request, Selenium
텍스트마이닝 개인과제	4/1(월) ~ 4/4(목)	개인과제 수행	프로젝트 적용 방안 고찰
데이터전처리 및 텍스트마이닝 방법론 2차회의	4/5(금)	댓글 결측치 제거 후 데이터전처리 방안 협의	
결측치 제거	4/5(금)	댓글 결측치 제거	
방법론 시나리오 3차회의	4/5(금)	워드클라우드 작성방안 논의	
리뷰 코퍼스 제작	4/5(금)	댓글 코퍼스 생성	
한국어 불용어 사전 만들기	4/7(일) ~ 4/8(월)	불용어 데이터 개별수집 및 코퍼스 제작	
프로젝트 긴급 4차 회의	4/8(월)	프로젝트 결과물 가공 및 역할분담	(한/영)워드클라우드, PPT초안 작성
데이터 전처리 + 워드클라우드 생성	4/8(월) ~ 4/9(화)	데이터 정규화, 형태소 분석기별 비교	OKt, Hannanum,Mecab, Pororo, Khaii
LDA 작업	4/9(화)	LDA 작성 및 코드 개선	
데이터분석	4/9(화)	워드클라우드, LDA 인사이트 분석	
자체평가 내용 정리	4/9(화)	프로젝트 자체평가 내용 정리	
이슈정리	4/9(화)	개발과정 발생 이슈 문서화	주제선정, 크롤링, 전처리 등
PPT 제작	4/9(화) ~ 4/11(목)	PPT 템플릿 작성, 자료취합, 레이아웃 조정	
총 개발기간 (총 3주)	3/22(금) ~ 4/11(목)		

프로젝트 팀 구성 및 역할

TEAM | ALPHA911



웹크롤링

데이터전처리

워드 클라우드 작성

PPT작성

대본 작성



Git repo/project

관리

웹크롤링

데이터 전처리

워드 클라우드 작성

데이터 LDA 처리

산출물 관리

PPT 작성

대본 작성



웹크롤링

데이터 전처리

데이터 영문 번역처리

영문 전처리

영문 워드클라우드 작성

영문 데이터 LDA 처리

PPT 작성



웹크롤링

데이터전처리

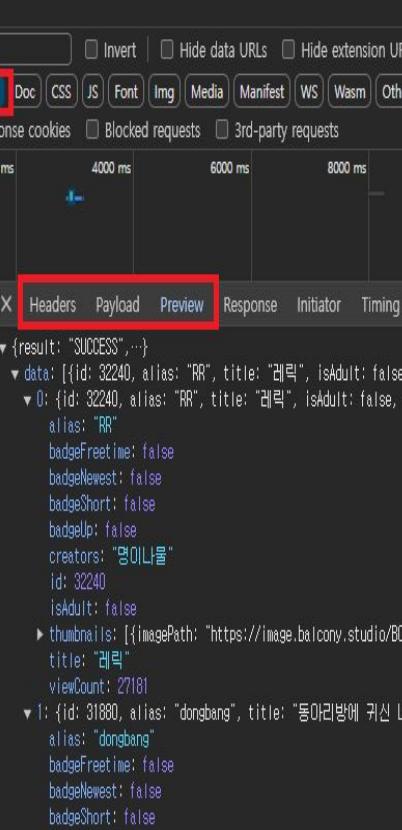
워드 클라우드 작성

데이터 LDA 처리

PPT작성

프로젝트 수행 결과

봄툰 웹 분석



The screenshot shows the Network tab of a browser developer tools interface. The 'Fetch/XHR' tab is selected. A specific request for 'details' is expanded, showing its Headers, Payload, and Preview. The Payload section displays a JSON object with fields like 'result', 'data', and 'thumbnails'. The 'data' array contains two items, each representing a comic entry with fields such as 'id', 'alias', 'title', 'isAdult', 'viewCount', and 'creators'. The 'thumbnails' field for the first item points to an image URL.

On the right side of the screen, there is a social media post from a user named '길드원'. The post has 58.7k likes, 1.2k comments, and 2,922 shares. It includes a caption in Korean and several replies from other users. Below the post, there are sections for '작품설명' (Work Description), '여행기' (Travel Diary), '소장하기' (Collection), and '1회 첫회부터' (1st Chapter). There are also thumbnail images for chapters 1 and 2.

프로젝트 실행 결과

봄툰 크롤링 코드

```
CommentCrawling_BUM  
CommentCrawling_Jo.ipynb  
CommentCrawling_Kim  
CommentCrawling_Moon
```

```
00_CommentCrawling_Main.ipynb
```

```
AllReview_Jo.csv  
FridayReview.csv  
MondayReview_Jo.csv  
SaturdayReview_Jo.csv  
SundayReview_Jo.csv  
TenDaysReview_Jo.csv  
ThirsdayReview_Jo.csv  
TuesdayReview_Jo.csv  
WendnesdayReview_Jo.csv
```

```
# 1단계 : 해당 도메인의 툰 정보를 얻기위해선 제일 먼저 요일별 제공하는 툰 고유 id를 알아야한다는 특성이 있어서 id부터 스크랩한다.  
weeks = ('MONDAY','TUESDAY','WEDNESDAY','THURSDAY','FRIDAY','SATURDAY','SUNDAY')  
  
db = {"toons_ids":[],  
      "reviews_list" : {}  
     }  
  
for week in tqdm(weeks,desc='각曜일 툐면서 전체 만화 id 스크랩'):  
    url = f'https://www.bomtoon.com/api/balcony-api-v2/contents/tab/schedule/comic?groupMenu=[{week}]&isIncludeTen=false&sort=POPULAR&adultToggle=false'  
    headers = {  
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36',  
        'X-Balcony-Id': 'BOMTOON_COM',  
        'Content-Type': 'application/json'  
    }  
    resp = requests.request("GET", url, headers=headers)  
    jsn = json.loads(resp.text)  
    lens = len(jsn['data'])  
    for l in range(lens):  
        db['toons_ids'].append(jsn['data'][l])  
  
# 2 단계 : 툐ids를 기반으로 툐 상세정보를 스크랩한다.  
url = f'https://www.bomtoon.com/api/balcony-api-v2/contents/tab/details'  
contentsIdsid = db['toons_ids']  
contentsIdsid = ','.join(map(str, db['toons_ids']))  
# print(contentsIdsid)  
payloads = json.dumps({  
    "contentsIds": contentsIdsid, #알고싶은 툐 id들이 미곳에 들어오면 됨.  
    "contentsThumbnailType": "VERTICAL,MAIN,SQUARE"  
})  
  
resp = requests.request("POST", url, headers=headers, data=payloads)  
jsn = json.loads(resp.text)  
lens = len(jsn['data'])  
for l in tqdm(range(lens),desc="리뷰 수집 중"): # 실제로 불러와진 툐 정보 갯수만큼 반복 수행.  
# for l in tqdm(range(10),desc="리뷰 수집 중"): # 테스트  
    title = jsn['data'][l]['title'] # 툐제목 : 댓글 할 예정  
    alias = jsn['data'][l]['alias'] # 각 툐 댓글 페이지 들어가려면 툐 alias가 필요  
    toon_url= f'https://www.bomtoon.com/api/balcony-api-v2/contents/{alias}?isNotLoginAdult=false&isPorch=false'  
    toon_resp = requests.request("GET", toon_url, headers=headers)  
    toon_jsn = json.loads(toon_resp.text)  
    commentCount = toon_jsn['data']['commentCount'] # 댓글 수집을 하려면 갯수 정보가 있어야 함.  
    if commentCount != 0: #댓글이 있는지 확인하고 있으면 수집 없으면 "noreviewstoon."
```

프로젝트 수행 결과

크롤링 결과 데이터

작가명	작가ID	댓글총갯수	댓글총페이지	시물인덱스	댓글쓴날짜	댓글의idx	작품명	작품회차	댓글자명	댓글	좋아요횟수
반들	321	20	11	12.0	2023-11-23 09:51	1508184.0	3과 2분의	1화	봄밤달빛	다들 비주얼 합격!	2.0
반들	321	20	11	18.0	2024-03-18 07:48	1679736.0	3과 2분의	1화	윤희거미니	첫화부터 흥미진진. 기대된다.	1.0
반들	321	20	11	19.0	2023-11-24 09:41	1509237.0	3과 2분의	1화	밤에 몰래!	아따마. 흥미진진하다	1.0
반들	321	20	11	3.0	2023-11-23 14:14	1508294.0	3과 2분의	1화	그럴쑤도	9 삼각구도 좋구나	1.0
반들	321	20	11	6.0	2023-11-22 21:27	1507632.0	3과 2분의	1화	봄툰러	3cf 엄청난 삼각관계	1.0
반들	321	20	11	3.0	2023-11-22 21:37	1507637.0	3과 2분의	2화	봄툰러	3cf ㅋㅋㅋㅋㅋㅋㅋ 아미친... 덕질메이트 찾았네 너무 귀엽다	13.0
반들	321	20	11	14.0	2023-11-24 23:11	1509743.0	3과 2분의	2화	메인공은	이야 이런 삼각구도 첨봐 ㅋㅋㅋㅋ 얘들 넘고 암다	6.0
반들	321	20	11	7.0	2023-11-25 10:41	1510077.0	3과 2분의	2화	ㅎㅎㅎ	덕질메이트 ㅎㅎㅎ	4.0
반들	321	20	11	2.0	2023-11-28 22:41	1512493.0	3과 2분의	2화	가즈아	미친 ㅋㅋㅋㅋㅋㅋㅋㅋㅋ 덕메찾았네 ㅋㅋㅋㅋㅋ 나도 초대해줘 같이 빨게 ㅋㅋㅋ	3.0
반들	321	20	11	11.0	2023-11-23 22:51	1508789.0	3과 2분의	2화	점	나두 껴줘 애두라 나도 태경이 혼자 좋아하고 있는 거 같아	2.0
반들	321	20	11	3.0	2024-01-16 14:31	1618674.0	3과 2분의	2화	봄툰러	024 나두 한태경 좋아해 부려고 계속 돌아와 계속	1.0

댓글	
0	아니.... 상상도 못한 방법으로 회피하셔 가지고 저도 놀랐어요...
1	아무리 생각해도 어릴적에 너무나도 매력적인 남성과의 접촉이,,, 그의 성적 취형을...
...	...
87780	봄툰은 연재빠른데 카카오페이지는 진짜 안올라오네ㅠㅠㅠ... 이용권 다 바꿔놨는데
87781	오~~~♡♡근데 무료가 더있었으면....

73335 rows x 1 columns

#	Column	Non-Null Count	Dtype
0	Unnamed: 0	89489	non-null float64
1	작가명	89496	non-null object
2	작가ID	89496	non-null object
3	댓글총갯수	89496	non-null object
4	댓글총페이지	89496	non-null object
5	게시물인덱스	85146	non-null object
6	댓글쓴날짜	85146	non-null object
7	댓글의idx	85146	non-null object
8	작품명	85146	non-null object
9	작품회차	82156	non-null object
10	댓글자명	85138	non-null object
11	댓글	73335	non-null object
12	좋아요횟수	85146	non-null object

```
import pandas as pd

df1 = pd.read_csv('/content/리뷰데이터네임.csv')
df1.info()
df1.isna().sum()
df1.isnull().sum() # 결측치
df1 = df1.dropna(subset = ['댓글']) # 결측치 있는 댓글 행 제거
df1['댓글']
df1.isnull().sum() # 결측치
df1.info()
df1
df1.to_csv("AllReviewMVDrop_BUM.csv", index = False)
df1.to_excel("AllReviewMVDrop_BUM.xlsx", index = False)

review_data = pd.read_csv('/content/boom_final.csv') # 결측치 제거된 데이터 업로드 후 읽음 "봄"
review_data.head() # 불러온 데이터 빠르게 확인
print('데이터 개수: {}'.format(len(review_data)))
# 문자열 아닌 데이터 모두 제거
review_data = [review for review in review_data['댓글'] if type(review) is str]
# 리스트를 DataFrame으로 변환
review_df = pd.DataFrame({'댓글': review_data})
# CSV 파일로 내보내기
review_df.to_csv('AllReviewCorpus.csv', index=False)
# XLSX 파일로 내보내기 (openpyxl 라이브러리가 필요할 수 있습니다)
review_df.to_excel('AllReviewCorpus.xlsx', index=False)
```

no_review_data.csv

파일 편집 보기

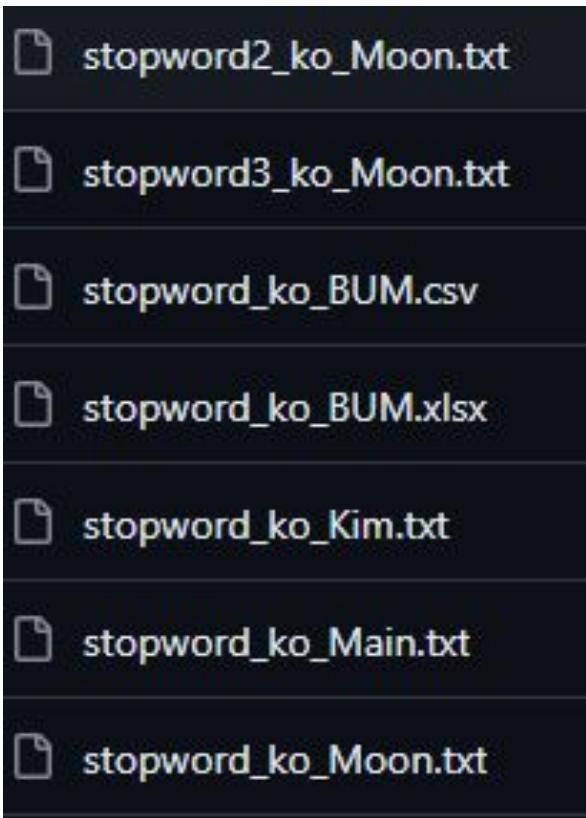
cleaned_review

상상도 못한 방법으로 회피하셔가지고 저도 놀랐어요
아무리 생각해도 어릴적에 너무나도 매력적인 남성과의 접촉이 그의 성적 취형을 결정짓게 만든것이 아닐까 합니다
하계란야무지게얹은미니토스트아이팟케이스쓰는공어때
아저씨가 평생 책임져요
죄송한데 한번만 실수로 벽에 끼이셨으면
뽀갈하고 이마까지기큐
문성이 가슴에 있는 햄버거 먹으면 되겠네
좋은 득자가 될게요
과거의 옆집 아저씨가 지금은 자신을 선생님이라고 부르고 있다니
아기공 등정따먹고 감옥가도 할말없다는 연상수
가슴이 시끄럽네요
너는 싸우겠다는 말을 빤스만 입고 하니
진심 미칠거같아 총 안맞을 수 있는데 공 차에 안타서 맞은거임퓨
수 허벅지랑 공 허리가 눈물이 입가에서 흐른다
거대 날다람쥐야
저런 엄한 몸매로 밥을 해주다니 결혼이죠
술 약한 아기공 집에 데려다 줘 떨어지는 라면 봉지 잡아 줘 연하의 키스는 피하는문성씨 죄가 많네
멀끔하고 가슴을 내미는데 고백을어떻게거 절합니까 여우수가 분명합니다
와시 허리들어간거봐 옷 바람직하다
문성씨 시계 찬거 꿀림 쌍 댄디 쌍떡대 쌍 간지남 깔리는 같은 느낌을 바로 주는 같아서 미치겠음
먼곳에 가있었다는 과거였던걸까요 미인공 떡대미남수
수도 은근 자각없이 플러팅하는거같은데 아무리 의구심때문에 신체적 접촉을 한거긴 호감도 없이 글케 스킨쉽을
호연이 빽친 얼굴 치명적이라 명치가 아프다
가슴이 시끄러워요
참아야 하는거 알지만 벽고 번만
마빡 우짜노
아자씨 귀여워요
문성씨 코트 안에 입고 활동하시는 거예요
육체미 난리났다
침대 구깃한거보고 얼굴 붉히냐고 새파랗게 어린것한테 동했다고 말을 못해

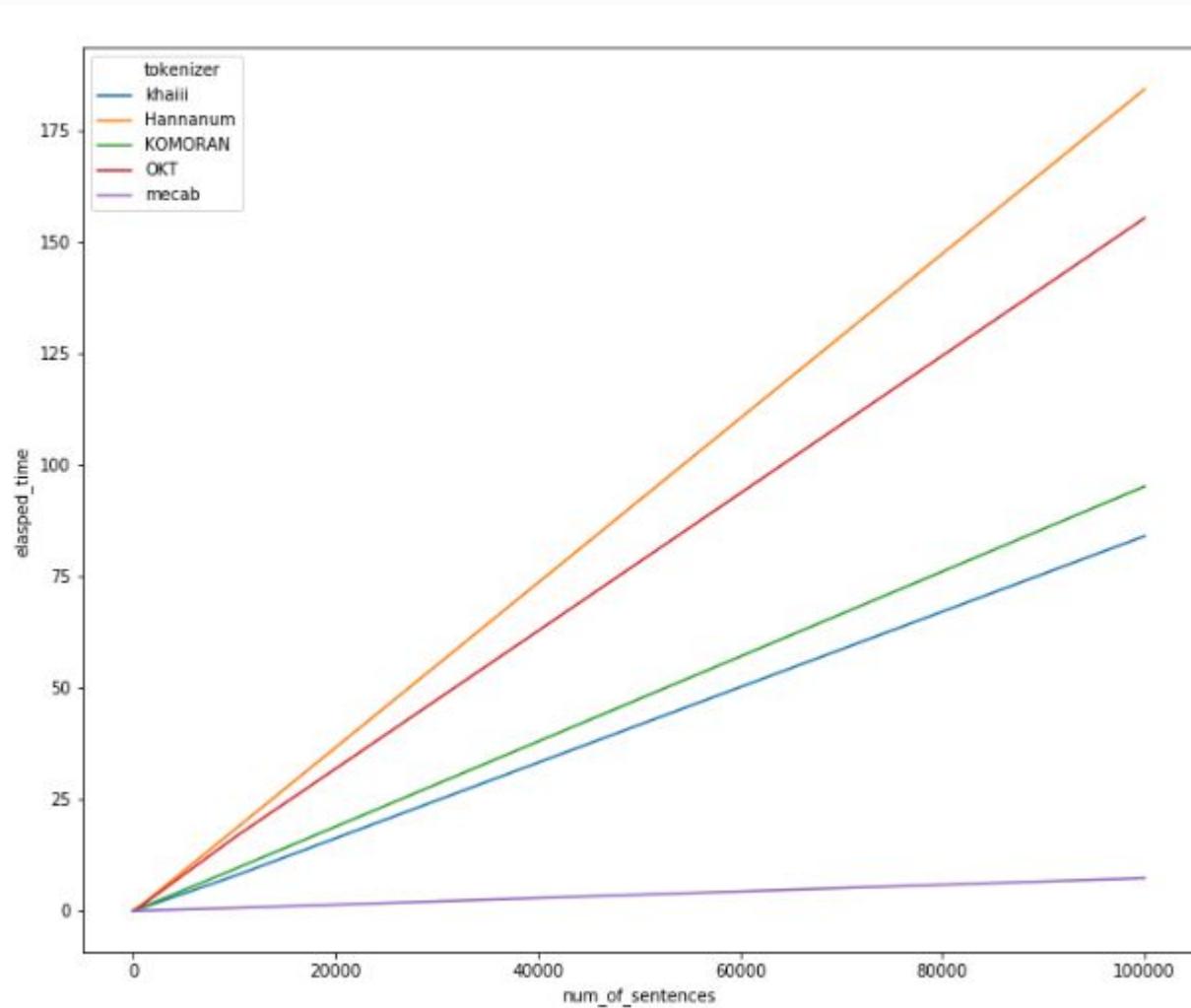
줄 1, 열 1 | 1,560,294자

프로젝트 실행 결과

불용어 사전 구축



28899	씨땡
28900	씨뎅
28901	씨발
28902	씨벨
28903	씨팔
28904	씹창
28905	씨8
28906	씨비니
28907	씨가령님
28908	씨가령년
28909	씨가령놈
28910	씨걸
28911	씨댕
28912	씨댕이
28913	씨뎅
28914	씨바라
28915	씨바알
28916	씨박색희
28917	씨박색히
28918	씨박색히
28919	씨발년
28920	씨발률
28921	씨발병신
28922	씨방새
28923	씨방세
28924	씨별가리
28925	씨버럼
28926	씨별년
28927	씨별쉐이
28928	씨별탱
28929	씨별
28930	씨불탱
28931	씨부랄
28932	씨부렬
28933	씨부렬
28934	씨불알
28935	씨뷰렬
28936	씨뷰렬
28937	씨브렬
28938	씨불년
28939	씨불



Python의 `timeit` 으로 측정

출처 및 참고 자료 : 속도 분석 + 성능 비교

<https://iostream.tistory.com/144>

한국어 분석 도구들

- . KoNLPy : 다양한 형태소 분석기를 제공하는 파이썬 패키지. Okt(Open Korea Text), Hannanum, Kkma, Komoran, Mecab 등 포함됨.
- . Okt : 트위터분석기라는 이름으로 진행되었던 프로젝트가 fork되어 진행되는 것으로 기본 베이스는 트위터 분석기
 - . Hannanum : KAIST에서 개발한 형태소 분석기
 - . Mecab : 일본어 형태소 분석기를 한국에 맞게 수정한 버전.
 - . Komoran : Java로 작성된 형태소 분석기,
 - . Soynlp : 비지도 학습을 기반으로 하는 한국어 전용 토큰화 도구. 데이터 내에서 자주 등장하는 단어를 기반으로 하는 토큰화를 수행.
 - . Kkma : 서울대학교에서 만든 분석기, 성능은 좋으나 속도면에서 단점이 있다.
 - . Khaiii : 카카오에서 개발한 형태소 분석기. 딥러닝 기반으로 정확도가 높음.
 - . KSS(Korean Sentence Splitter) : 한국어 문장 분리를 위한 도구. 문장의 경계를 잘 인식하여 텍스트를 문장 단위로 나누는데 유용.
 - . ETRI : 한국전자통신연구원에서 개발한 한국어 처리 API. 형태소 분석, 개체명 인식, 의존 구문분석 등 다양한 기능을 제공. 인터넷 연결이 필요.
 - . Pororo : 카카오브레인에서 만든 한글 자연어처리 라이브러리.
 - . KoBERT : 한국어 정보 처리를 위해 특별히 훈련된 BERT 모델. 토큰화 기능 외에도 한국어 텍스트의 이해와 생성 작업에 탁월한 성능을 보임.
 - . Kiwi : 다양한 자연어 처리 기능을 제공. 형태소 분석, 품사 태깅 등 지원.
 - . KoGPT-2 : 한국어 생성 모델, 자연어 이해 및 생성 작업에 사용됨. 대화 생성, 문장 완성 등의 작업에 적합.

```
import pandas as pd
import numpy as np
import matplotlib.font_manager as fm
import matplotlib.pyplot as plt
import re
from tqdm import tqdm

!apt -qq -y install fonts-nanum > /dev/null

# 데이터 불러오기
train_data = pd.read_csv('/content/drive/MyDrive/boom_final.csv')
print('데이터 개수:', len(train_data))

# 문자열이 아님 데이터 제거
train_review = train_data['댓글'].dropna().map(str)

# 문자열이 아님 데이터를 모두 제거한 후 numpy 배열로 변환합니다.
train_review = np.array([review for review in train_data['댓글'] if type(review) is str])

# 불용어 목록 파일을 불러옵니다.
with open('/content/drive/MyDrive/stopword_main.txt', 'r', encoding='utf-8') as f:
    stopwords = np.array([line.strip().replace('\r', '') for line in f.readlines()])

# 불용어를 제거하고, 대소문자 및 특수문자 처리합니다. (리스트 컴프리헨션을 사용하여 처리)
filtered_review = []
for review in tqdm(train_review, desc="리뷰 처리 중"):
    review = review.lower()
    review = re.sub(r'[^가-힣\w]', '', review)
    words = review.split()

    # numpy 배열을 사용하여 불용어를 필터링합니다. 이 경우, 리스트 컴프리헨션을 사용하는 것이 더 적합합니다.
    filtered_words = ' '.join([word for word in words if word not in stopwords])
    filtered_review.append(filtered_words)
```

리스트 대신
numpy 씀

프로젝트 수행 결과

okt 명사 추출 + 불용어 처리

```
!pip install konlpy

import pandas as pd
import numpy as np
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import re
from tqdm import tqdm
from konlpy.tag import Hannanum # 한나눔으로 변경

!apt -qq -y install fonts-nanum > /dev/null

# 형태소 분석기 초기화 (한나눔으로 변경)
hannanum = Hannanum()

# 데이터 불러오기
train_data = pd.read_csv('/content/코퍼스데이터.csv')
print('데이터 개수:', len(train_data))

# 문자열이 아닐 데이터 제거
train_review = train_data['댓글'].dropna().map(str)

# 불용어 목록 파일을 불러옵니다.
with open('/content/drive/MyDrive/stopword_main.txt', 'r', encoding='utf-8') as f:
    stopwords = np.array([line.strip().replace('\r', '') for line in f.readlines()])

# 명사 추출 및 불용어 처리 (한나눔 사용)
filtered_review = []
for review in tqdm(train_review, desc="리뷰 처리 중"):
    review = review.lower()
    review = re.sub(r'[ㄱ-ㅎㅏ-ㅣㅏ-ㅣ가-힣]+', '', review)
    nouns = hannanum.nouns(review) # 한나눔 형태소 분석을 통해 명사만 추출

    # numpy 배열을 사용하여 불용어를 필터링합니다. 이 경우, 리스트 컴프리헨션을 사용하는 것이 더 적합합니다.
    filtered_nouns = ''.join([noun for noun in nouns if noun not in stopwords])
    filtered_review.append(filtered_nouns)
```

프로젝트 수행 결과

mecab 명사 추출 + 불용어 처리

```
!pip install mecab-python #메캅 : 일본어를 형태소 분석기를 한국 버전으로 수정한 것
!pip install install curl git
!bash <(curl -s https://raw.githubusercontent.com/konlpy/konlpy/master/scripts/mecab.sh)
!pip install konlpy #코NFL파이 : 한국어 자연어 처리 페키지

from konlpy.tag import Mecab
mecab = Mecab()
print(mecab.nouns('메캅이 잘 설치되었는지 확인하는 테스트 문장입니다.'))

import pandas as pd
import numpy as np
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import re
from tqdm import tqdm
from konlpy.tag import Mecab

!apt -qq -y install fonts-nanum > /dev/null
# 형태소 분석기 초기화
mecab = Mecab()

# 데이터 불러오기
train_data = pd.read_csv('/content/코퍼스데이터.csv')
print('데이터 개수:', len(train_data))

# 문자열이 아님 데이터 제거
train_review = train_data['댓글'].dropna().map(str)

# 불용어 목록 파일을 불러옵니다.
with open('/content/drive/MyDrive/stopword_main.txt', 'r', encoding='utf-8') as f:
    stopwords = np.array([line.strip().replace('\#r', '') for line in f.readlines()])

# 명사 추출 및 불용어 처리
filtered_review = []
for review in tqdm(train_review, desc="리뷰 처리 중"):
    review = review.lower()
    review = re.sub(r'[ㄱ-ㅎㅏ-ㅣㅏ-ㅣㄱ-ㅎ]+|[가-힝][a-zA-Z0-9]+', '', review)
    nouns = mecab.nouns(review) # 형태소 분석을 통해 명사만 추출

    # numpy 배열을 사용하여 불용어를 필터링합니다. 이 경우, 리스트 컴프리헨션을 사용하는 것이 더 적합합니다.
    filtered_nouns = ''.join([noun for noun in nouns if noun not in stopwords])
    filtered_review.append(filtered_nouns)
```

프로젝트 수행 결과

khaiii 명사 추출 + 불용어 처리

```
git clone https://github.com/kakao/khaiii.git
!pip install cmake
!mkdir build
!cd build && cmake /content/khaiii
!cd /content/khaiii/build/ && make all
!cd /content/khaiii/build/ && make resource
!cd /content/khaiii/build/ && make install
!cd /content/khaiii/build/ && make package_python
!pip install /content/khaiii/build/package_python

from khaiii import KhaiiiApi
import pandas as pd
import numpy as np
import re
from tqdm import tqdm

# 형태소 분석기 초기화
api = KhaiiiApi()

# 데이터 불러오기
train_data = pd.read_csv('/content/코퍼스데이터.csv')
print('데이터 개수:', len(train_data))

# 문자열이 아닌 데이터 제거
train_review = train_data['댓글'].dropna().map(str)

# 불용어 목록 파일을 불러옵니다.
with open('/content/drive/MyDrive/stopword_main.txt', 'r', encoding='utf-8') as f:
    stopwords = np.array([line.strip().replace('\#r', '') for line in f.readlines()])

# 명사 추출 및 불용어 처리
filtered_review = []
for review in tqdm(train_review, desc="리뷰 처리 중"):
    review = review.lower()
    review = re.sub(r'[^\uac00-\ud7ff|\uad50-\ud7ff|\uac00-\ud7ff|\uac00-\ud7ff]+', '', review)
    nouns = []
    for word in api.analyze(review):
        for morph in word.morphs:
            if morph.tag in ['NNG', 'NNP']: # 일반 명사와 고유 명사만 추출
                nouns.append(morph.lex)

    filtered_nouns = ''.join([noun for noun in nouns if noun not in stopwords])
    filtered_review.append(filtered_nouns)
```

프로젝트 수행 결과

pororo 명사 추출 + 불용어 처리

```
# !apt -qq -y install fonts-nanum > /dev/null #한글 글꼴 설치
# !pip install wordcloud

import pandas as pd
import numpy as np
from wordcloud import WordCloud
import matplotlib.pyplot as plt

data = pd.read_csv('/content/filtered_review.csv') #워드클라우드 파일 불러오기

# 다시 numpy 배열로 변환합니다.
filtered_review = np.array(filtered_review)

# 워드 클라우드 생성 및 출력
fontpath = '/usr/share/fonts/truetype/nanum/NanumBarunGothic.ttf'
wordcloud = WordCloud(font_path=fontpath, background_color='white', width=800, height=800, stopwords=set(stopwords)).generate(' '.join(filtered_review))
plt.figure(figsize=(8, 8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

프로젝트 수행 결과



불용어 처리



mecab + 불용어 처리

```
1 pip install pororo
Collecting pororo
  Using cached pororo-0.4.2-py3-none-any.whl (256 kB)
INFO: pip is looking at multiple versions of pororo to determine which version is compatible with other requirements. This could take a while.
Using cached pororo-0.4.1-py3-none-any.whl (253 kB)
Using cached pororo-0.4.0-py3-none-any.whl (253 kB)
Using cached pororo-0.3.5-py3-none-any.whl (253 kB)
Using cached pororo-0.3.4-py3-none-any.whl (216 kB)
Using cached pororo-0.3.3-py3-none-any.whl (215 kB)
Using cached pororo-0.3.2-py3-none-any.whl (212 kB)
Using cached pororo-0.3.1-py3-none-any.whl (211 kB)
INFO: pip is looking at multiple versions of pororo to determine which version is compatible with other requirements. This could take a while.
ERROR: Cannot install pororo==0.3.1, pororo==0.3.2, pororo==0.3.3, pororo==0.3.4, pororo==0.4.0, pororo==0.4.1 and pororo==0.4.2 because these package versions have conflicting dependencies.

The conflict is caused by:
pororo 0.4.2 depends on torch==1.6.0
pororo 0.4.1 depends on torch==1.6.0
pororo 0.4.0 depends on torch==1.6.0
pororo 0.3.5 depends on torch==1.6.0
pororo 0.3.4 depends on torch==1.6.0
pororo 0.3.3 depends on torch==1.6.0
pororo 0.3.2 depends on torch==1.6.0
pororo 0.3.1 depends on torch==1.6.0

To fix this you could try:
1. loosen the range of package versions you've specified
2. remove package versions to allow pip attempt to solve the dependency conflict

-- 
ResolutionImpossible: for help visit https://pypi.org/en/latest/topics/dependency-resolution/#dealing-with-dependency-conflicts
```

pororo 에러

각 WordCloud 결과



okt + 불용어 처리



hannanum + 불용어 처리

```
CMake Error at /root/.hunter/_Base/Download/Hunter/0.23.34/70287b1/Unpacked/cmake/modules/hunter_wiki.cmake:12 (message):
Call Stack (most recent call first):
/root/.hunter/_Base/Download/Hunter/0.23.34/70287b1/Unpacked/cmake/modules/hunter_fatal_error.cmake:20 (hunter_wiki)
/root/.hunter/_Base/Download/Hunter/0.23.34/70287b1/Unpacked/cmake/modules/hunter_download.cmake:613 (hunter_fatal_error)
/root/.hunter/_Base/Download/Hunter/0.23.34/70287b1/Unpacked/cmake/projects/GTest/hunter.cmake:244 (hunter_download)
/root/.hunter/_Base/Download/Hunter/0.23.34/70287b1/Unpacked/cmake/modules/hunter_add_package.cmake:62 (include)
CMakeLists.txt:63 (hunter_add_package)
```

```
-- Configuring incomplete, errors occurred!
/bin/bash: line 1: cd: /content/khaiii/build/: No such file or directory
/bin/bash: line 1: cd: /content/khaiii/build/: No such file or directory
/bin/bash: line 1: cd: /content/khaiii/build/: No such file or directory
/bin/bash: line 1: cd: /content/khaiii/build/: No such file or directory
ERROR: Invalid requirement: '/content/khaiii/build/package_python'
Hint: It looks like a path. File '/content/khaiii/build/package_python' does not exist.
```

프로젝트 수행 결과

pyLDA 코드

```
!pip install konlpy

from konlpy.tag import Okt
okt = Okt()

import pandas as pd
DATA_PATH = '/content/'
filtered_review = pd.read_csv(DATA_PATH+'filtered_review.csv')

import re
f = open('/content/stopword_ko_Main.txt')
stop_words = f.read().split()
def preprocessing_vec(review, okt): # 불러온 데이터셋은 이미 잘 전처리가 되어 있지만 마지막 더블체킹 수행.
    # 1. 정규화 : 한글 및 공백을 제외한 문자 모두 제거.
    review_text = re.sub("[^가-힣\w]", "", review)

    # 2. 형태소 : okt 형태소 토큰화 + 품사 태깅
    word_review = okt.pos(review_text, stem=True)

    # 노이즈 & 불용어 제거
    word_review = [(token, pos) for token, pos in word_review if not token in stop_words and len(token) > 1]

    # 명사 추출
    word_review = [token for token, pos in word_review if pos in ['Noun']]

    return word_review

from tqdm import tqdm
clean_review = []
for review in filtered_review['filtered_review']:
    if type(review) == str:
        clean_review.append(preprocessing_vec(review,okt))
    else:
        clean_review.append([])

clean_review = sum(clean_review,[])
```

```
clean_review = pd.DataFrame(clean_review)

clean_review[0]

clean_review.to_csv("clean_review.csv", index = False)

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.datasets import fetch_20newsgroups

vectorizer = TfidfVectorizer(max_features= 1000, # 상위 1,000개의 단어를 보존
max_df = 0.5, smooth_idf=True)

X = vectorizer.fit_transform(clean_review[0])

# TF-IDF 행렬의 크기 확인
print('TF-IDF 행렬의 크기 : ',X.shape)

from sklearn.decomposition import LatentDirichletAllocation
# 모델 객체 생성
lda_model = LatentDirichletAllocation(n_components=20, random_state=777)

# fit transform
lda_model.fit(X)

# 단어 집합. 1,000개의 단어가 저장됨.
terms = vectorizer.get_feature_names_out()

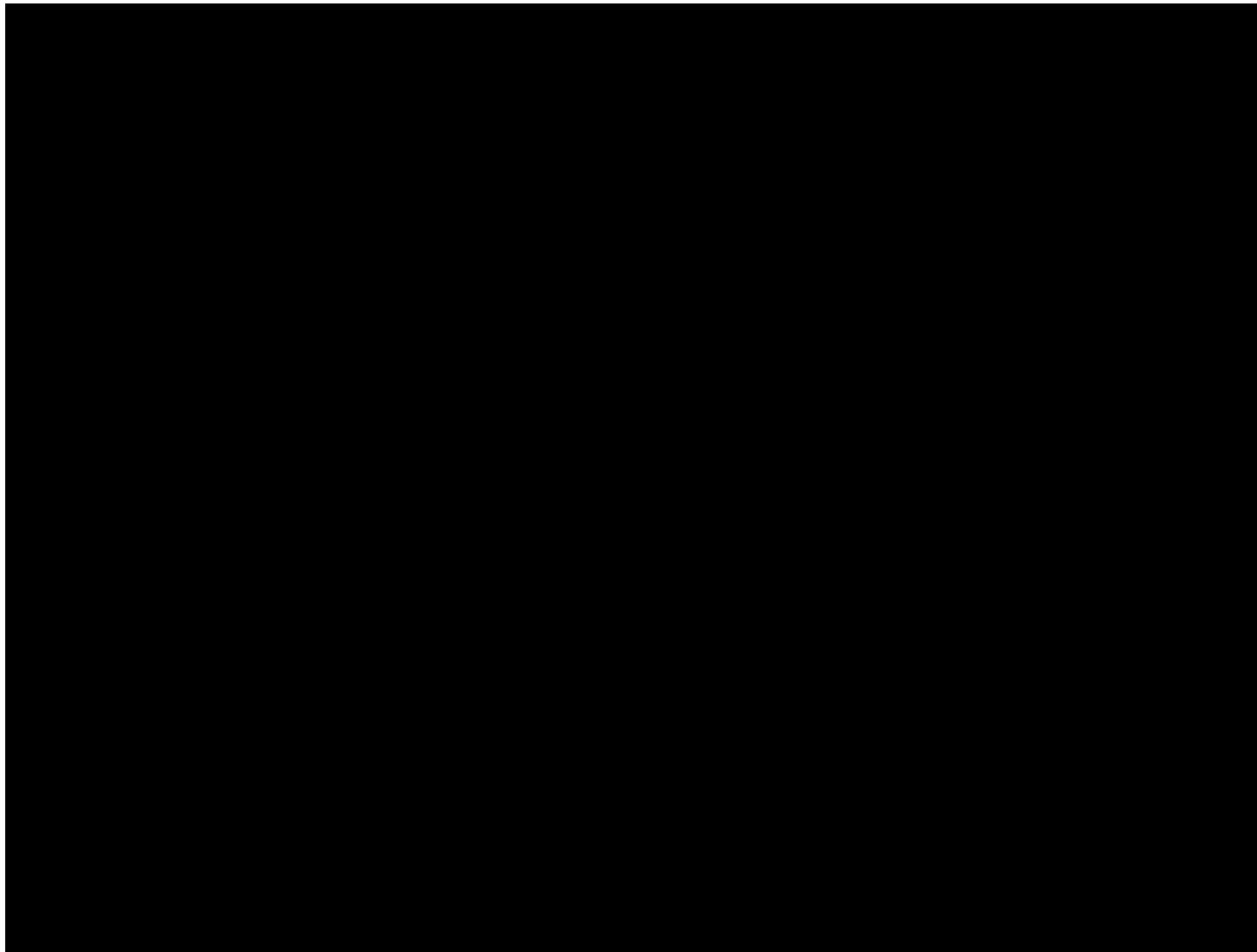
def get_topics(components, feature_names, n=5):
    for idx, topic in enumerate(components):
        print("Topic %d: " % (idx+1), [(feature_names[i], topic[i].round(2)) for i in topic.argsort()[:-n-1:-1]])

get_topics(lda_model.components_,terms)

# 시각화
# LDavis 는 토픽 모델링에 자주 이용되는 Latent Dirichlet Allocation (LDA) 모델의 학습 결과를 시각적으로 표현하는 라이브러리입니다
!pip install -U pyLDAvis
```

프로젝트 수행 결과

pyLDA 시연 영상



프로젝트 수행 결과

한국어 코퍼스 -> 영어 전환

Translator()

```
!pip install googletrans==4.0.0-rc.1

from googletrans import Translator
import pandas as pd
import os
from tqdm import tqdm

...
AllReviewCorpus_Moon
해당 파일은 깃허브 파일 기준 csv 파일입니다.
...
local=os.getcwd()

ppd=pd.read_csv(local+'//AllReviewCorpus_Moon.csv')
tran = Translator()
#tran.translate('안녕하세요.')
# <Translated src=ko dest=en text=Good evening. pronunciation=Good evening.>

def Tran_KRtoEN(txt,c,y):
    ...
        txt : pandas.core.series.Series 컬럼 인수값
        y   : 원하는 컬럼 갯수

        반환값 : list형식
    ...
krenStr=[]
for i in tqdm(range(c,y)):
    try:
        kkr=tran.translate(txt[i], dest='en')
        #print(kkr.text)
        krenStr.append(kkr.text)
```

프로젝트 수행 결과

한국어 코퍼스 영문 전환 결과

프로젝트 수행 결과

영문_1만개 워드클라우드



작성 코드

```
local=os.getcwd()

example=pd.read_csv(local+'//csv//bomtoon_final.csv')
colname=list(example.keys())
# 추가할 영문 불용어 정의
additional_stop_words = ['ha', 'im','tt','going','oh','wow','know','ah','dont','think','come','want','like','g
# scikit-Learn에서 제공하는 영어 불용어에 사용자 정의 불용어를 합침
stop_words = List(text.ENGLISH_STOP_WORDS.union(additional_stop_words))

example['0'].isna().sum() # 전체 컬럼에 NaN 확인

cv = CountVectorizer(max_features=1000, stop_words = stop_words ,ngram_range=(1,2)) # 문장별로 단어 행렬 변환을
tr_comment = cv.fit_transform(example[colname[1]].values.astype('U')) # 학습을 위해 fit_transform

words = cv.get_feature_names_out() # 결측치 무결성

doc = tr_comment[0].toarray()
```

추가된 불용어 미처리 및 테스트데이터 확인

프로젝트 수행 결과

영문_ALL댓글 워드클라우드



작성 코드

```
# 추가할 영문 불용어 정의
additional_stop_words = ['ha', 'im','tt','going','oh','wow','know','ah','dont','think','come','want','like','good']

# scikit-learn에서 제공하는 영어 불용어에 사용자 정의 불용어를 합침
stop_words = list(text.ENGLISH_STOP_WORDS.union(additional_stop_words))

example[1].__()__() # 전체 커먼에 노드 __
```

프로젝트 수행 결과

영문 버전 LDA 결과

TF-IDF 행렬의 크기 : [72635, 1000]

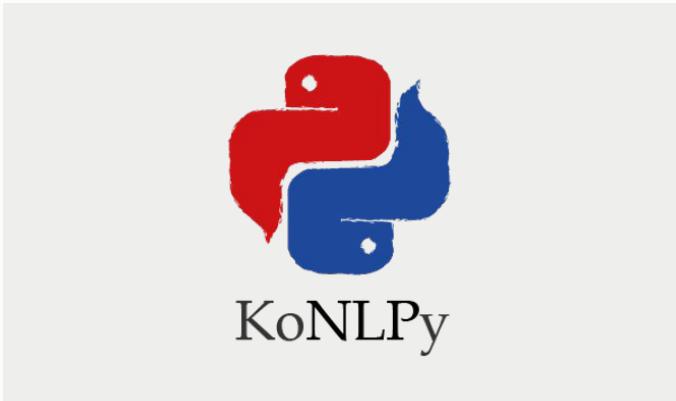
```
vectorizer = TfidfVectorizer(stop_words=stop_words, max_features=1000,  
# 상위 1,000개의 단어를 보존  
max_df = 0.5, smooth_idf=True)  
  
X = vectorizer.fit_transform(getRead_word['0'].values.astype('U'))  
# astype()을 사용한 데이터형 dtype 변환 (캐스트)  
  
# TF-IDF 행렬의 크기 확인  
print('TF-IDF 행렬의 크기 :',X.shape)  
.  
. .  
. .  
pyLDAvis.enable_notebook()  
vis = pyLDAvis lda_model.prepare(  
    lda_model = lda_model,  
    dtm = X,  
    vectorizer = vectorizer,  
    mds='tsne')  
pyLDAvis.display(vis)
```



What we used



PYTHON / Google Colab



KoNLPy



GitHub



LDA / word cloud

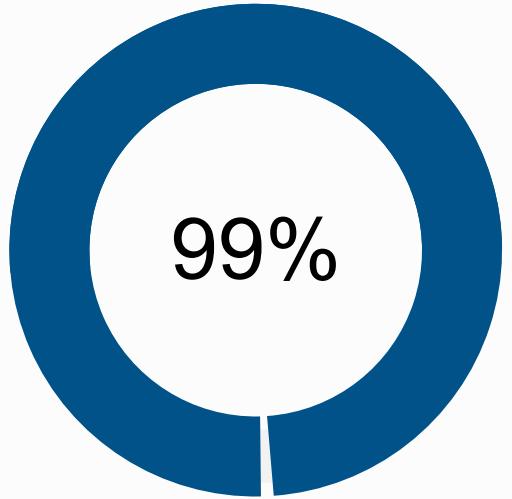


NumPy

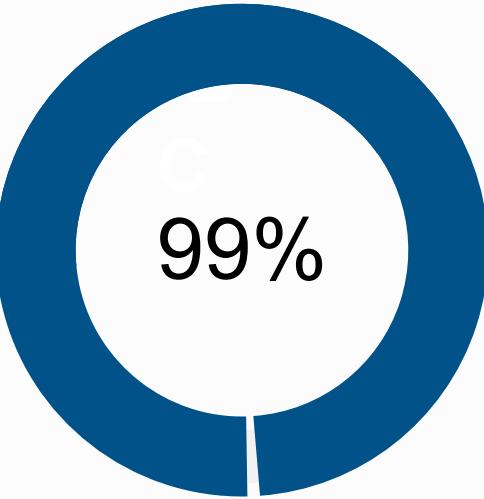


Pandas

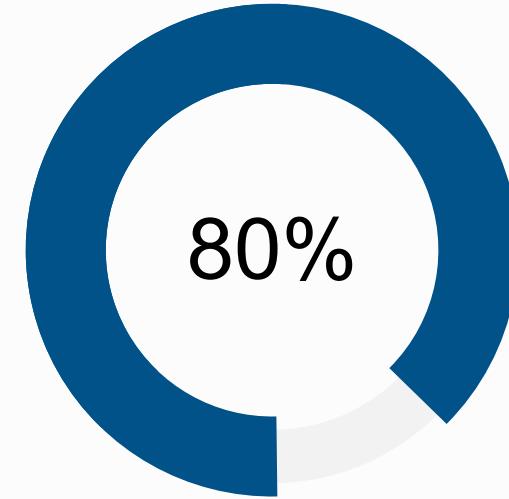
자체 평가 의견



기획 부합도



실무 활용도



달성도

- ◆ 데이터 크롤링 OK
- ◆ 텍스트 전처리 OK
- ◆ 워드 클라우드 OK
- ◆ LDA OK
- ◆ 페어플롯 Not Yet
- ◆ 감성분석 Not Yet

자체 평가

김예선 : 팀원분들과 협업해서 결과를 도출하는 경험이 큰 성장이 되었습니다. **LDA** 등 많이 배울 수 있어서 좋았습니다.

문영식 : 전반적인 진행과 협업에 있어서 만족도가 크지만 결과물은 아쉬움이 있다, 더 많은 기술을 실험 분석 이해 적용하고 싶었다.

조인준 : 각각의 개별 아이디어를 하나로 융합하여 수업 시간 배운 내용을 접목하는 것이 원활히 진행되어 뜻깊은 여정이었다.

최희범 : 서로 익숙하지 않은 분야에 대해 이해해가며, 지속적인 커뮤니케이션을 통해 프로젝트를 완성한 점에서 좋은 경험이었음.

감사합니다

Q & A