Title:

E-Book Recommendation System based on reviews with Sentiment Analysis.

Author:

202103672 Heo Kang

201902241 Lee GyuMin

201903345 Cho MinJung

202100640 Kim Minji

Abstract

This project aims to develop a personalized e-book recommendation system using data from Millie's Library. The primary focus is on conducting sentiment analysis of user reviews to understand each user's preferences and emotional responses. This analysis quantifies the users' positive or negative feelings, forming the foundation for customized e-book recommendations. The system has been implemented using both collaborative filtering and content-based approaches independently, and also by combining these two methods. Additionally, it considers various factors, including the popularity of e-books among users. Furthermore, the project has incorporated Collaborative Filtering using the Fastai Library. This research underscores the importance of sentiment analysis for recommendation systems on unlabelled datasets.

1. Introduction

1.1 Background

This project focuses on changing reading habits due to the increased use of personal electronic devices. Recently, there's been a trend of people preferring e-books over traditional paper books, primarily for their convenience and portability. However, the e-book market is lagging behind offline bookstores in offering a diverse range of book types. As e-books are not as varied as paper books, there is a greater need to recommend e-books that align more closely with users' preferences. Therefore, our goal is to develop a user-customized e-book recommendation system that better matches users' preferences and interests.

1.2 Project Objective

- Develop a personalized e-book recommendation system that utilizes data from Millie's Library to understand each user's preferences and emotional responses.

- Explore various algorithms including sentiment analysis, collaborative filtering, and content-based methodologies.

- Provide strategies for recommending a wide range of relevant and interesting e-books tailored to users' reading preferences, based on the developed system.

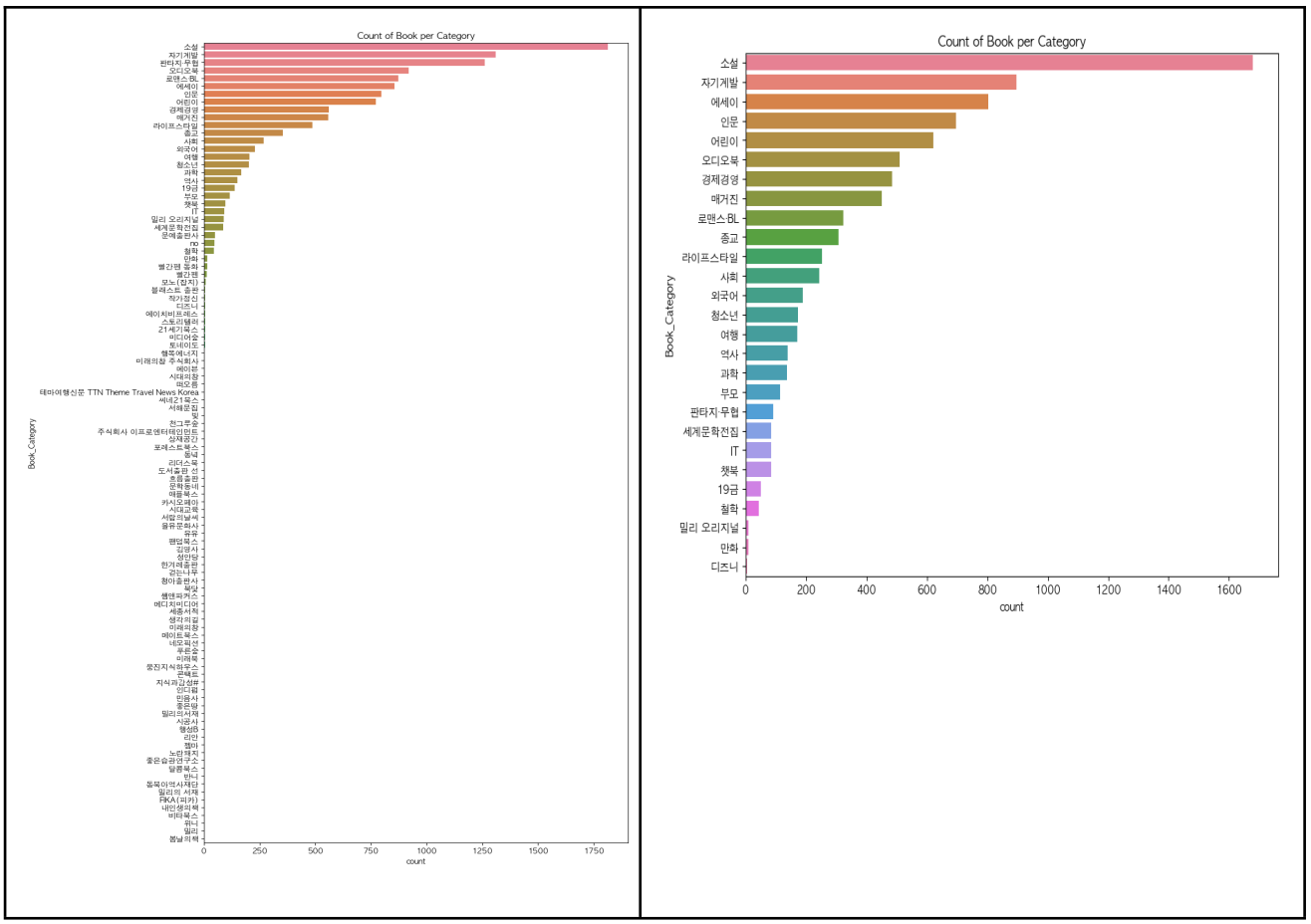
2 Dataset Preliminary

2.1 Exploratory Data Analysis

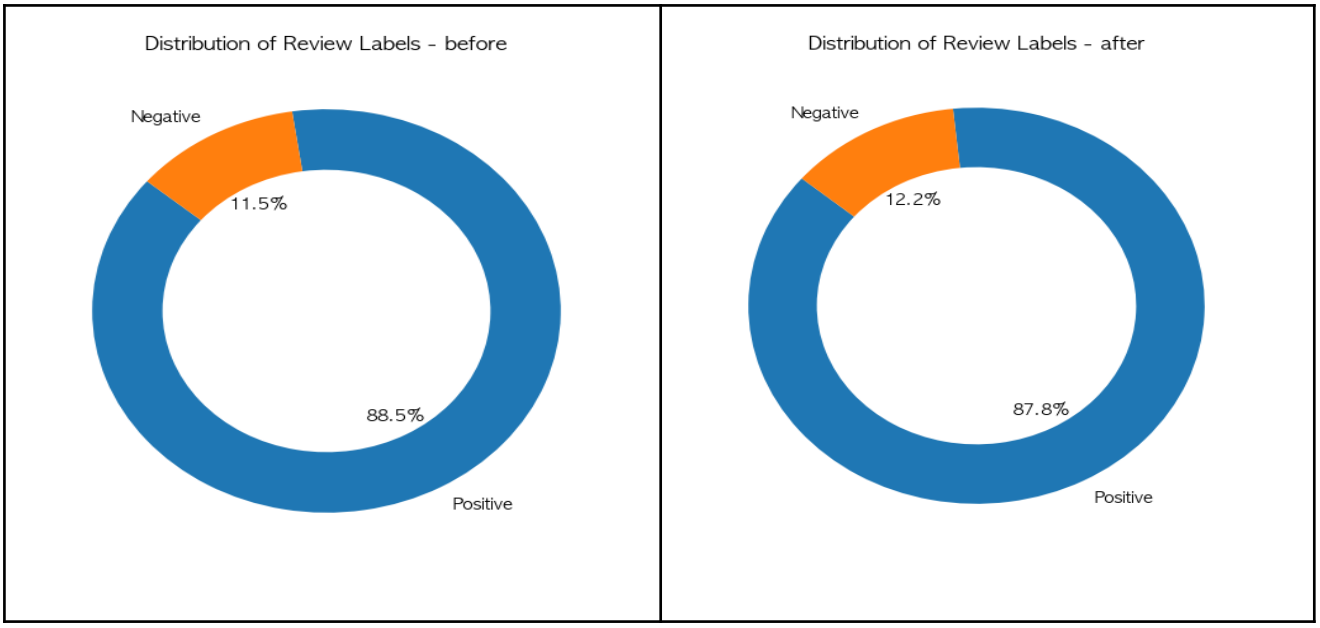
- *Statistical Information*

- *Data Visualization*

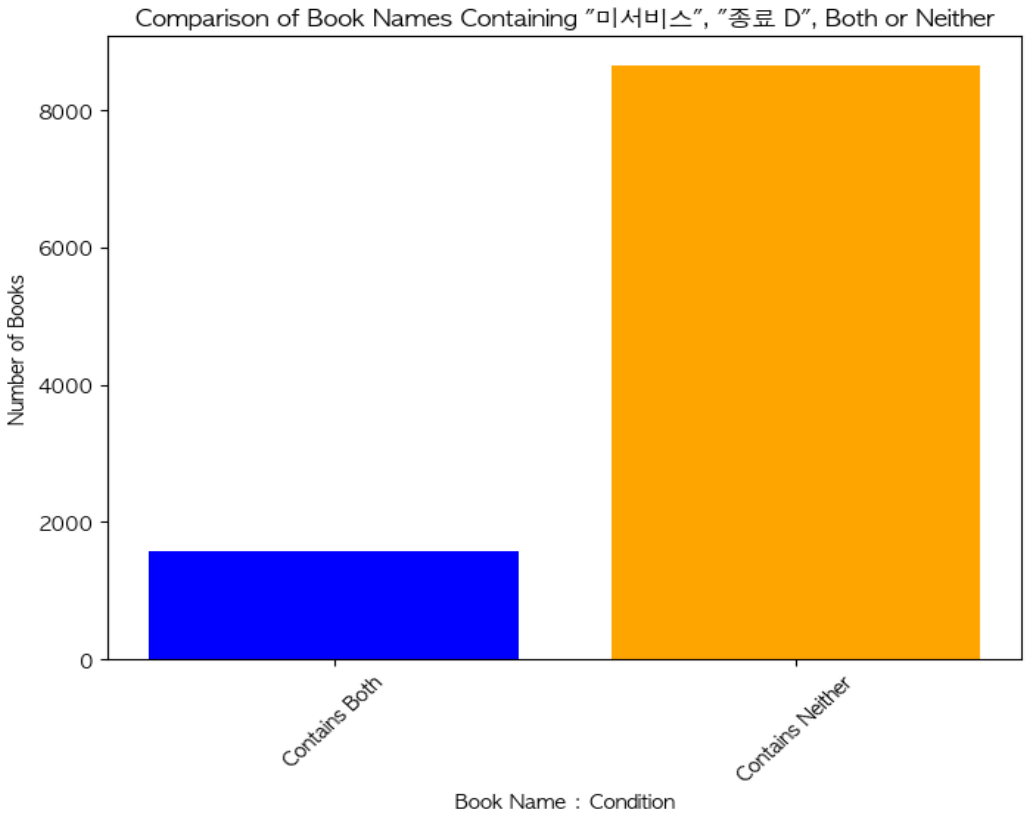
- Category changing per book (before vs after)



- After Pseudo label, Statistics Label about Review Text preprocessing *(before vs after)*



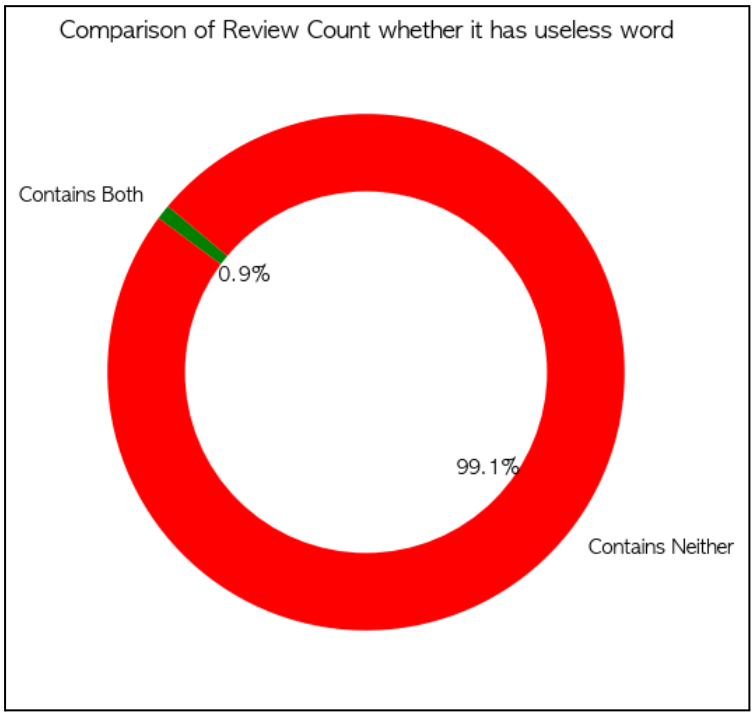
- Delete the book that has ***'no service'*** or ***'Finish D'***
- Describe the picture below :
  - *"Contains Both" the number of books that have* ***'no service' or 'Finish D'***
  - *"Contains Neither" the number of books that are not in the condition.*



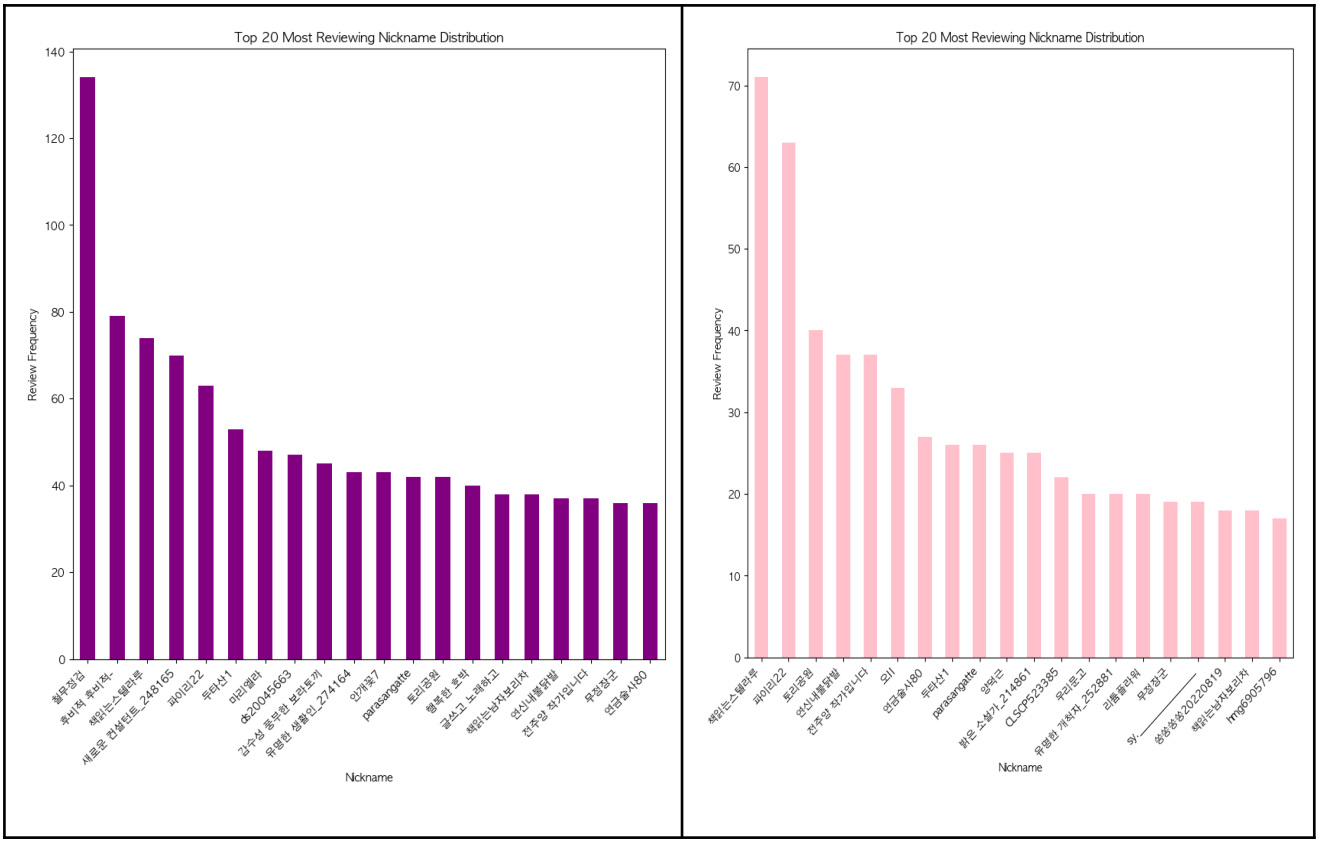
- Delete Review that have Useless words like “재구매 방지용” “재구독 방지” “재대여 방지” ...

	Nickname	Review_Text	Book_Name
436	철무정검	재구매 방지용 입니다	만능돌로 살아가는 법 1권
2203	철무정검	재구매 방지용 입니다	미서비스 \n \n 전능의 뮤...
2204	철무정검	재구매 방지용 입니다	미서비스 \n \n 전능의 뮤...
2285	철무정검	재구매 방지용 입니다	미서비스 \n \n 전능의 뮤...
2906	철무정검	재구매 방지용 입니다	지배자의 반지 2권
...	...	...	...

The picture about this data below :



- Variation about Most Reviewing Person (**before**-purple vs **after**-pink)
- Describe the picture below :
  - *It denotes the person whose nickname is ‘철무정검’ wrote useless review so he is bad user who can provide poor quality recommendation.*



- Integration Series book
  - ***The picture below :*** *example of Series book*

Before Preprocessing :

```
df_book[df_book['Book_Name'].str.contains("방탄 리더")].head(50)
```

✓ 0.0s

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook
1160	방탄 리더 의무교육 1 (리더 7대 의무교육 사용 설명서)	최보규	자기계발	NaN	1개	0	NaN	X
1161	방탄 리더 의무교육 2 (리더 7대 의무교육 사용 설명서)	최보규	자기계발	NaN	1개	0	NaN	X
1162	방탄 리더 의무교육 3 (리더 7대 의무교육 사용 설명서)	최보규	자기계발	NaN	1개	0	NaN	X
1163	방탄 리더 의무교육 4 (리더 7대 의무교육 사용 설명서)	최보규	자기계발	NaN	1개	0	NaN	X
1164	방탄 리더 의무교육 5 (리더 7대 의무교육 사용 설명서)	최보규	자기계발	NaN	1개	0	NaN	X
1165	방탄 리더 의무교육 6 (리더 7대 의무교육 사용 설명서)	최보규	자기계발	NaN	1개	0	NaN	X
1168	방탄 리더 태도 1 (태도 동기부여! 태도는 스펙이다!)	최보규	자기계발	NaN	9개	0	NaN	X
1169	방탄 리더 태도 2 (태도 동기부여! 태도는 스펙이다!)	최보규	자기계발	NaN	4개	0	NaN	X
1170	방탄 리더 태도 3 (태도 동기부여! 태도는 스펙이다!)	최보규	자기계발	NaN	3개	0	NaN	X
1171	방탄 리더 태도 4 (태도 동기부여! 태도는 스펙이다!)	최보규	자기계발	NaN	3개	0	NaN	X
1172	방탄 리더 태도 6 (태도 동기부여! 태도는 스펙이다!)	최보규	자기계발	NaN	8개	0	NaN	X
1323	방탄 리더 감정컨트롤 1 (리더 스트레스 관리)	최보규	자기계발	NaN	1개	0	NaN	X
1324	방탄 리더 감정컨트롤 2 (리더 스트레스 관리)	최보규	자기계발	NaN	1개	0	NaN	X
1325	방탄 리더 감정컨트롤 3 (리더 스트레스 관리)	최보규	자기계발	NaN	1개	0	NaN	X
1326	방탄 리더 감정컨트롤 4 (리더 스트레스 관리)	최보규	자기계발	NaN	1개	0	NaN	X

After Preprocessing :

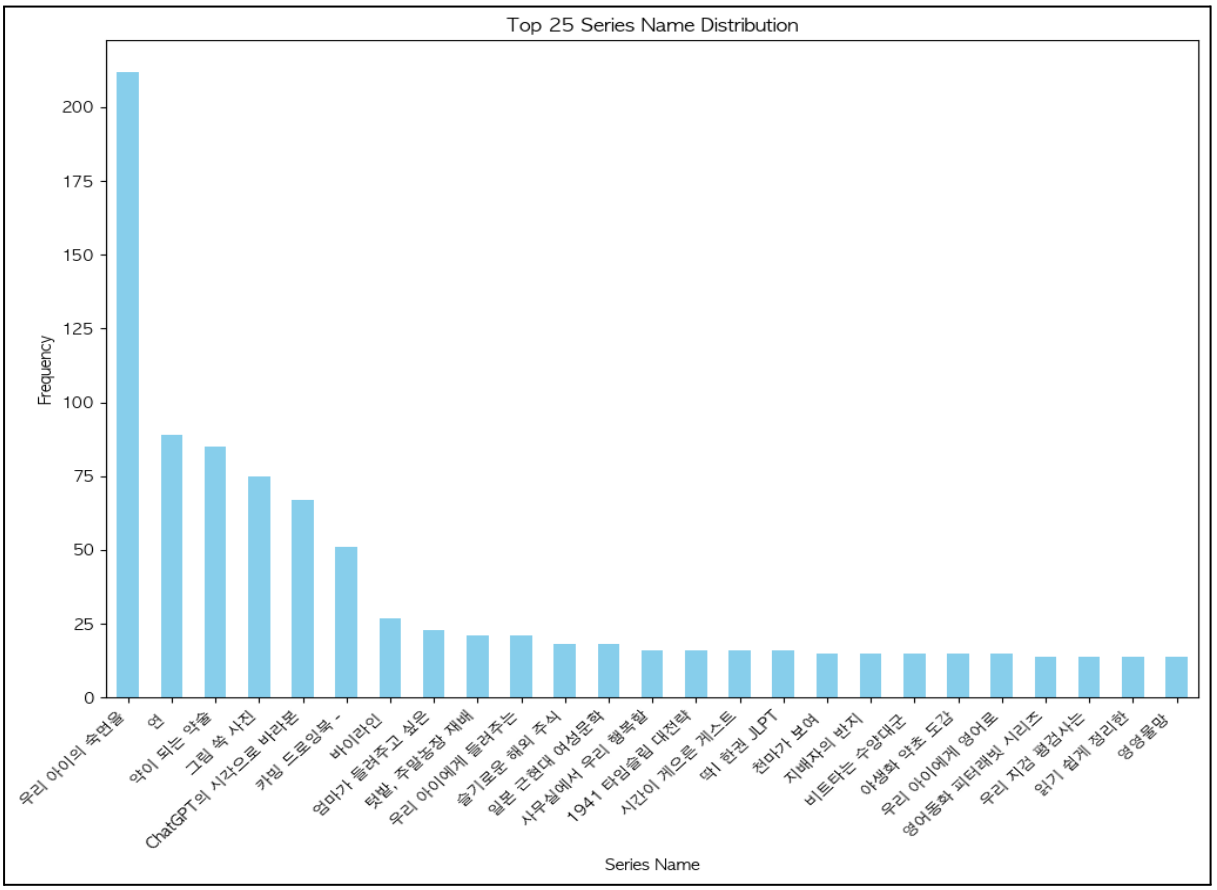
```
df_book_cleaned[df_book_cleaned['Book_Name'].str.contains('방탄 리더')]
```

✓ 0.0s Python

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	AudioBook	Series_Name
725	방탄 리더 의무교육 1 (리더 7대 의무교육 사용 설명서)	최보규	자기계발	0.0	1	0	X	방탄 리더 의무교육
728	방탄 리더 태도 1 (태도 동기부여! 태도는 스펙이다!)	최보규	자기계발	0.0	9	0	X	방탄 리더 태도
861	방탄 리더 감정컨트롤 1 (리더 스트레스 관리)	최보규	자기계발	0.0	1	0	X	방탄 리더 감정컨트롤

Additionally, We set the criteria for the series book. The criteria is the book name has 2 word continuously same with each other or is the book name ends with '권' '(완결)' '부' '화'...

- ***The picture below :*** *histogram about frequency of series name*



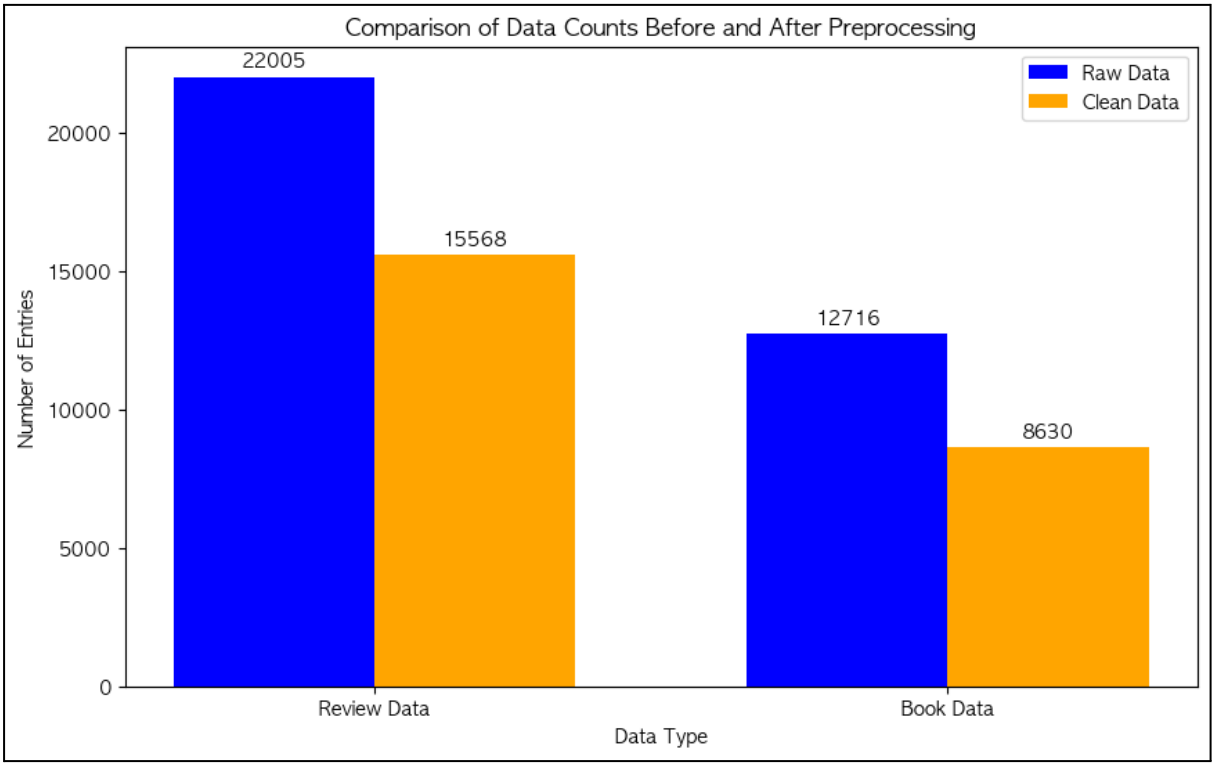
- ***The picture below :*** *Statistical Information about Series book*

```
# statistics about series book name
series_name_counts = pd.DataFrame(df_series['Series_Name'].value_counts()).reset_index()
series_name_counts.columns = ['Series_Name', 'Frequency']
print("-----------------------------------")
print("Max of frequency :", series_name_counts['Frequency'].max())
print("Min of frequency :", series_name_counts['Frequency'].min())
print("-----------------------------------")
series_name_counts
```

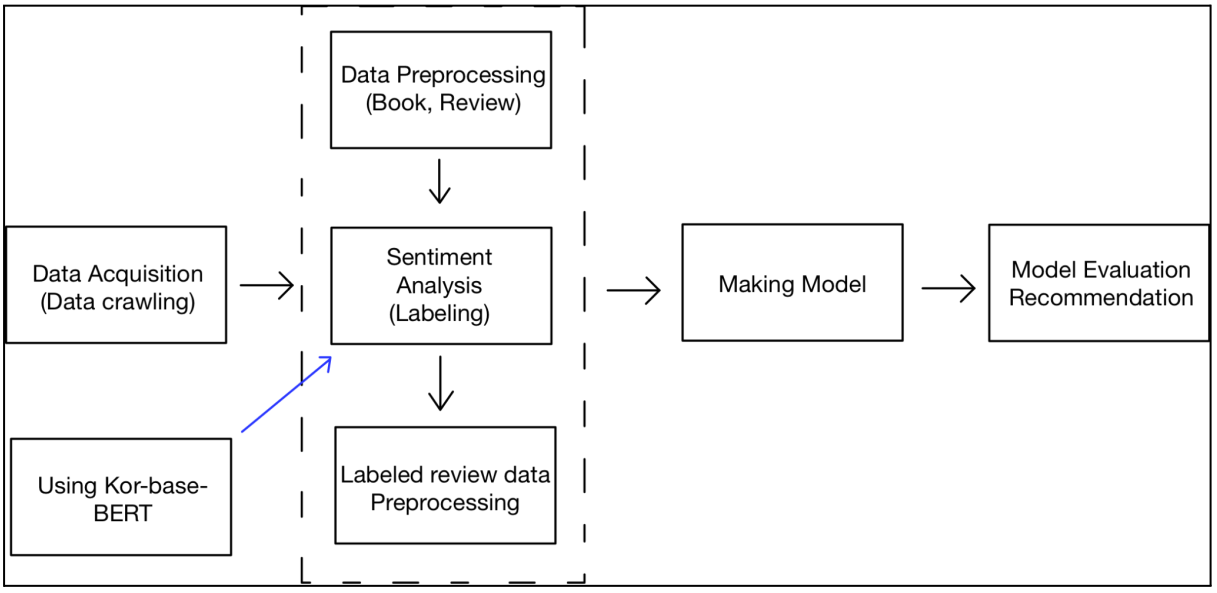
```
-----------------------------------
Max of frequency : 212
Min of frequency : 1
-----------------------------------
```

	Series_Name	Frequency
0	우리 아이의 숙면을	212
1	연	89
2	약이 되는 약술	85
3	그림 쑥 사진	75
4	ChatGPT의 시각으로 바라본	67
...	...	...
8641	모두모두 소중해요	1
8642	문어빵 파티에 초대해요	1
8643	나의 마흔에	1

- Overall Preprocessing Result
- Describe the picture below :
    - *The picture show difference between before and after preprocessing in Review and Book Data*



2.2 Flow of Analysis



Data Acquisition

- Collect data by web crawling '밀리의 서재' web site

```
base_url = "https://www.millie.co.kr/v3/bookDetail/{}"
```

- Gather information about the book

```
target = soup.find('div', class_='meta-data')
b_name_element = target.find('p', class_='book-name')
b_name = b_name_element.text.strip() if b_name_element else None

b_author_element = soup.find('p', class_='author')
b_author = b_author_element.find('span').text.strip() if b_author_element else None

b_readtogether_element = soup.find('div', class_='read-together')
b_readtogether = b_readtogether_element.find('strong',class_='number').text.strip()

category_data = soup.find('li', class_='slide-item')
b_category = category_data.find('a').text.strip() if category_data else None


b_percent_element = soup.find('strong', class_='line-desc')
b_percent = b_percent_element.text.strip() if b_percent_element else None
```

- We also collect reviews using review links with the same book number

```
review_url = f"https://www.millie.co.kr/v3/bookDetail/more/review/{book_number}"
```

```
target = soup.find('ul', class_='review-list')
p_names = target.find_all('p', class_='nickname')
p_texts = target.find_all('pre', class_='cont')
```

- Since the book number is included in the link, set the book number to open the link repeatedly

```
start_number = 179493566
end_number = 179493570
```

- This automatically scrolls and collects all reviews on the page

```
# Scroll to the end of the page
last_height = driver.execute_script("return document.body.scrollHeight")
while True:
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
    time.sleep(2)  # Add a delay after scrolling

    new_height = driver.execute_script("return document.body.scrollHeight")
    if new_height == last_height:
        break
    last_height = new_height
```

- Because it is based on preferences known from reviews, it is excluded from data collection if there is no review.

```
if not reviews_df.empty:
    # 책 정보를 리뷰 데이터프레임에 추가
    reviews_df['Book_Name'] = book_info['Book_Name']

    # CSV 파일로 저장
    reviews_df.to_csv(f'All_book_reviews_{book_number}.csv', index=False, encoding='utf-8')

    print(f"Data for book number {book_number} saved successfully.")
else:
    print(f"No reviews for book number {book_number}. Skipping...")
book_df.to_csv('Bookdata_.csv', index=False, encoding='utf-8')
```

- This is raw data through crawling.

Book Data

Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook
어린 왕자, 후쿠시마 이후	변홍철	에세이		2개	0		X
경제지표를 읽는 시간	빈센트(김두언)	경제경영	완독할 확률 49%	1,300개	4개	경제입문	X
거대한 충격 이후의 세계	서영민	경제경영	완독할 확률 65%	378개	3개		X
노화의 재설계	모건 레빈 지음 / 이한음 옮김	인문	완독할 확률 43%	473개	3개		X
근대의 아틀리에	김영동	인문		9개	0		X
호밀밭	박월복	에세이		1개	0		X
보리밥 묵고 방구 뀜께 배가 폭 꺼져불등만	김옥태	에세이		5개	0		X
공부라는 여정	유청훈/최미선	청소년		10개	0		X
햇살 가득한 소풍길	김희옥	에세이		7개	0		X
꿈이 있는 여행영어	김계희	외국어		2개	0		X
콘크리트 폴리싱 DF 공법	㈜에이치케이닥터프로아	자기계발		3개	0		X
크리티크M Critique M 2023 Vol.5	안치용 외	매거진		28개	0		X
여행하며 크는 아이들 — 경북편	구경래	여행		2개	0		X
내일 뭐 하지?	신원철	밀리 오리지널	완독할 확률 74%	4,049개	60개		X
사무실에서 우리 행복할 수 있을까 1화	안나	밀리 오리지널	완독할 확률 90%	1,230개	3개		X
사무실에서 우리 행복할 수 있을까 2화	안나	밀리 오리지널	완독할 확률 78%	273개	1개		X

Review Data

Nickname	Review_Text	Book_Name	Unnamed: 0	Column1	Column2	Column3
그냥 대충 살기로 했다	내 고추가 필요할 때 잘 달리기 위해서는 평소에 내가 잘 달려야 한다.	남자는 어떻게 일어서는가				
드로니	성기능과 건강은 연결되어있다. 건강에 왕도는 없다. 꾸준한 자기관리가 답이다.	남자는 어떻게 일어서는가				
연신내불닭발	역시 달리기?	남자는 어떻게 일어서는가				
유명한 개척자_252881	상대의 말만 듣지말고 상대를 들어보자 말의 한계 어휘부족 부적합한 제스처집중치말기	왜 욱하세요?				
yyerin	욱"하는 사람이 무례한 사람	왜 욱하세요?				
유명한 개척자_252881	더하지도 빼지도 말고 내 맘 내가 알아주기	왜 욱하세요?				
kitty-100	잘 읽었습니다. 정독해보려고요	왜 욱하세요?				
지혜로운 희망전도사_332209	시종일관 읽으면서 남는 단어는 욱하기전 그냥듣자	왜 욱하세요?				
zazazaza	욱하지말자	왜 욱하세요?				
전주양 작가입니다	감사합니다	왜 욱하세요?				
덕질왜안해	필력이 많이 부족하고 읽다보면 잉? 스러운 부분이 꽤 많습니다.	왜 욱하세요?				
소소한 독서가 호두샘	침착하게 응대하고 관계를 지속하는 방법, 쉽지만 실천하기 어려운 것들을 숙지하게 하는 책	왜 욱하세요?				
짱구맛나	가볍게 빨리 읽고!! 다시 정독하기를 권한다!	왜 욱하세요?				
디노젤리	내 마음에 좀 더 집중하자	왜 욱하세요?				

Data Preprocessing

- Book Data Preprocessing

```
df_book = pd.read_csv("all_book.csv")
df_book
```

Python

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook
0	어린 왕자, 후쿠시마 이후	변홍철	에세이	NaN	2개	0	NaN	X
1	경제지표를 읽는 시간	빈센트(김두언)	경제경영	완독할 확률 49%	1,300개	4개	경제입문	X
2	거대한 충격 이후의 세계	서영민	경제경영	완독할 확률 65%	378개	3개	NaN	X
3	노화의 재설계	모건 레빈 지음 / 이한음 옮김	인문	완독할 확률 43%	473개	3개	NaN	X
4	근대의 아틀리에	김영동	인문	NaN	9개	0	NaN	X
...	...	...	...	...	...	...	...	...
12711	다음 생엔 돌이 되고 싶다.	강다희	에세이	NaN	3개	0	NaN	X
12712	배터리 다이제스트 TOP9	선우 준	과학	NaN	5개	0	NaN	X
12713	배터리 다이제스트 TOP10	선우 준	과학	NaN	5개	0	NaN	X
12714	평신도 및 목회자를 위한 다니엘서 Q&A	우슬초	종교	NaN	3개	0	NaN	X
12715	나로 돌아가는 여행	곽나원	에세이	NaN	4개	0	NaN	X

12716 rows × 8 columns

Unnecessary Book_Category Removing

Removing 'no', '미래북', '웅진지식하우스' etc...

```
df_book["Book_Category"].unique()
```

Python

```
array(['에세이', '경제경영', '인문', '청소년', '외국어', '자기계발', '매거진', '여행', '밀리 오리지널',
       '오디오북', 'no', '소설', '미래북', '로맨스·BL', '어린이', '판타지·무협', '웅진지식하우스',
       '과학', '19금', 'IT', '사회', '철학', '종교', '라이프스타일', '부모', '콘택트',
       '세계문학전집', '지식과감성#', '챗북', '역사', '인디펍', '만화', '문학동네', '민음사', '좋은땅',
       '빨간펜 동화', '밀리의서재', '시공사', '디즈니', '빨간펜', '도서출판 선', '빛', '상재공간',
       '주식회사 이프로엔터테인먼트', '행성B', '리안', '젬마', '블래스트 출판', '노란돼지', '시대교육',
       '미디어숲', '메이트북스', '떠오름', '좋은습관연구소', '에이치비프레스', '달콤북스', '포레스트북스',
       '스토리텔러', '미래의창 주식회사', '반니', '동북아역사재단', '밀리의 서재', 'FIKA(피카)',
       '행복에너지', '씨네21북스', '문예출판사', '작가정신', '메이븐', '내인생의책', '모노(잡지)',
       '비타북스', '테마여행신문 TTN Theme Travel News Korea', '위니', '밀리', '푸른숲',
       '네오픽션', '천그루숲', '서해문집', '팬덤북스', '시대의창', '21세기북스', '성안당', '동녘',
       '걷는나무', '리더스북', '메디치미디어', '애플북스', '세종서적', '쌤앤파커스', '한겨레출판', '생각의길',
       '카시오페아', '미래의창', '북닷', '토네이도', '청아출판사', '흐름출판', '김영사', '유유',
       '을유문화사', '서랍의날씨', '봄날의책', nan], dtype=object)
```

```
categories_to_remove = ["no", "미래북", "웅진지식하우스", "콘택트", "지식과감성#", "인디펍", "문학동네", "민음사", "좋은땅", "빨간펜 동화",
                        "시공사", "밀리의 서재", "빨간펜", "도서출판 선", "빛", "상재공간", "주식회사 이프로엔터테인먼트", "행성B", "리안",
                        "젬마", "블래스트 출판", "노란돼지", "시대교육", "미디어숲", "메이트북스", "떠오름", "좋은습관연구소", "에이치비프레스",
                        "달콤북스", "포레스트북스", "스토리텔러", "미래의창 주식회사", "반니", "동북아역사재단", "밀리의 서재", "FIKA(피카)",
                        "행복에너지", "씨네21북스", "문예출판사", "작가정신", "메이븐", "내인생의책", "모노(잡지)", "비타북스", "테마여행신문 TTN Theme Tr
                        "위니", "밀리", "푸른숲", "네오픽션", '네오픽션', '천그루숲', '서해문집', '팬덤북스', '시대의창', '21세기북스', '성안당', '동녘',
                        '리더스북', '메디치미디어', '애플북스', '세종서적', '쌤앤파커스', '한겨레출판', '생각의길', '카시오페아', '미래의창', '북닷', '토네이
                        '흐름출판', '김영사', '유유', '을유문화사', '서랍의날씨', '봄날의책', '밀리의서재']

df_book = df_book[~df_book['Book_Category'].isin(categories_to_remove)]

df_book = df_book.dropna(subset=['Book_Category'])

df_book
```

Python

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook
0	어린 왕자, 후쿠시마 이후	변홍철	에세이	NaN	2개	0	NaN	X
1	경제지표를 읽는 시간	빈센트(김두언)	경제경영	완독할 확률 49%	1,300개	4개	경제입문	X
2	거대한 충격 이후의 세계	서영민	경제경영	완독할 확률 65%	378개	3개	NaN	X
3	노화의 재설계	모건 레빈 지음 / 이한음 옮김	인문	완독할 확률 43%	473개	3개	NaN	X
4	근대의 아틀리에	김영동	인문	NaN	9개	0	NaN	X
...	...	...	...	...	...	...	...	...
12711	다음 생엔 돌이 되고 싶다.	강다희	에세이	NaN	3개	0	NaN	X
12712	배터리 다이제스트 TOP9	선우 준	과학	NaN	5개	0	NaN	X
12713	배터리 다이제스트 TOP10	선우 준	과학	NaN	5개	0	NaN	X
12714	평신도 및 목회자를 위한 다니엘서 Q&A	우슬초	종교	NaN	3개	0	NaN	X
12715	나로 돌아가는 여행	곽나원	에세이	NaN	4개	0	NaN	X

12456 rows × 8 columns

```
df_book["Book_Category"].unique()
```

```
array(['에세이', '경제경영', '인문', '청소년', '외국어', '자기계발', '매거진', '여행', '밀리 오리지널',
       '오디오북', '소설', '로맨스·BL', '어린이', '판타지·무협', '과학', '19금', 'IT', '사회',
       '철학', '종교', '라이프스타일', '부모', '세계문학전집', '챗북', '역사', '만화', '디즈니'],
      dtype=object)
```

Delete Missing Value in "Book_Author"

```
#Check Missing Value in Book_Author
missing_authors = df_book[df_book['Book_Author'].isna()]

missing_authors
```

Python

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook
3235	김언호의 서재 탐험	NaN	오디오북	NaN	36개	0	NaN	X
3563	우리 아이에게 들려주는 엄마의 인형동화 - 늑대와 일곱마리 아기 염소	NaN	오디오북	NaN	2개	0	NaN	X
3723	엄마가 들려주고 싶은 동화 - 호랑이가 준 요술 보자기	NaN	오디오북	NaN	1개	0	NaN	X
4517	앵커 씨의 행복 이야기	NaN	오디오북	NaN	3개	0	NaN	X
4606	임금님 집에 예쁜옷을 입혀요	NaN	오디오북	NaN	1개	0	NaN	X
7060	미서비스 \n \n 스프린터	NaN	오디오북	NaN	576개	4개	NaN	X
7061	미서비스 \n \n 튤립 모양	NaN	오디오북	NaN	423개	1개	NaN	X
8544	서비스 종료 도서	NaN	19금	NaN	0개	0	NaN	X

```
# Delete Missing Value
df_book = df_book.dropna(subset=['Book_Author'])

df_book
```

Python

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook
0	어린 왕자, 후쿠시마 이후	변홍철	에세이	NaN	2개	0	NaN	X
1	경제지표를 읽는 시간	빈센트(김두언)	경제경영	완독할 확률 49%	1,300개	4개	경제입문	X
2	거대한 충격 이후의 세계	서영민	경제경영	완독할 확률 65%	378개	3개	NaN	X
3	노화의 재설계	모건 레빈 지음 / 이한음 옮김	인문	완독할 확률 43%	473개	3개	NaN	X
4	근대의 아틀리에	김영동	인문	NaN	9개	0	NaN	X
...	...	...	...	...	...	...	...	...
12711	다음 생엔 돌이 되고 싶다.	강다희	에세이	NaN	3개	0	NaN	X
12712	배터리 다이제스트 TOP9	선우 준	과학	NaN	5개	0	NaN	X
12713	배터리 다이제스트 TOP10	선우 준	과학	NaN	5개	0	NaN	X
12714	평신도 및 목회자를 위한 다니엘서 Q&A	우슬초	종교	NaN	3개	0	NaN	X
12715	나로 돌아가는 여행	곽나원	에세이	NaN	4개	0	NaN	X

12448 rows × 8 columns

Check Missing Value in "ReadTogether"

ReadTogther means "like"

```
#Check Missing Value in Book_Author
missing_readtogether = df_book[df_book['ReadTogether'].isna()]

missing_readtogether
```

Python

Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook

Check Data in "ReviewCount"

```
missing_reviewcount = df_book[df_book['ReviewCount'].isna()]

missing_reviewcount
```

Python

Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	Keyword	AudioBook

Drop "Keyword" Column

```python
# Drop the 'Keyword' column
df_book = df_book.drop(columns=['Keyword'])

df_book
```

Python

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	AudioBook
0	어린 왕자, 후쿠시마 이후	변홍철	에세이	NaN	2개	0	X
1	경제지표를 읽는 시간	빈센트(김두언)	경제경영	완독할 확률 49%	1,300개	4개	X
2	거대한 충격 이후의 세계	서영민	경제경영	완독할 확률 65%	378개	3개	X
3	노화의 재설계	모건 레빈 지음 / 이한음 옮김	인문	완독할 확률 43%	473개	3개	X
4	근대의 아틀리에	김영동	인문	NaN	9개	0	X
...	...	...	...	...	...	...	...
12711	다음 생엔 돌이 되고 싶다.	강다희	에세이	NaN	3개	0	X
12712	배터리 다이제스트 TOP9	선우 준	과학	NaN	5개	0	X
12713	배터리 다이제스트 TOP10	선우 준	과학	NaN	5개	0	X
12714	평신도 및 목회자를 위한 다니엘서 Q&A	우슬초	종교	NaN	3개	0	X
12715	나로 돌아가는 여행	곽나원	에세이	NaN	4개	0	X

12448 rows × 7 columns

Integration Series books and deleting useless words about Book_Name

```python
import pandas as pd
import re

def delete_rows_containing_text(df, column_name, texts_to_remove):
    for text in texts_to_remove:
        df = df[~df[column_name].str.contains(text, na=False, case=False)]

    return df

texts_to_remove = ['종료 D', '미서비스']
df_book = delete_rows_containing_text(df_book, 'Book_Name', texts_to_remove)


def delete_rows_by_conditions(df, conditions1, condition2):
    for condition1 in conditions1:
        df = df[~((df['Book_Name'].str.contains(condition1, na=False, case=False)) &
                  (df['Book_Category'] == condition2))]

    return df

condition1 = ['기출', '모의고사','EBS']
condition2 = '자기계발'

df_book = delete_rows_by_conditions(df_book, condition1, condition2)

def identify_series_name(book_title, num_words=3):
    words = book_title.split()
    return ' '.join(words[:num_words]) if len(words) >= num_words else book_title
```

```python
def integrate_series_data(data):
    data['Book_Name'] = data['Book_Name'].str.replace(r'\(\w+\)', '').str.replace(r'\d+권|\d+부|\d+화|\(완결\)|\(완료\)', '',regex=True)

    data['Series_Name'] = data['Book_Name'].apply(lambda x: identify_series_name(x))

    def extract_number(text):
        numbers = re.findall(r'\d+', str(text))
        return int(numbers[0]) if numbers else 0

    data['Completion_Percent'] = data['Completion_Percent'].apply(extract_number)
    data['ReadTogether'] = data['ReadTogether'].apply(extract_number)
    data['ReviewCount'] = data['ReviewCount'].apply(extract_number)

    grouped_data = data.groupby(['Series_Name', 'Book_Author', 'AudioBook'])

    data['Completion_Percent'] = grouped_data['Completion_Percent'].transform('mean').round()
    data['ReadTogether'] = grouped_data['ReadTogether'].transform('max')
    data['ReviewCount'] = grouped_data['ReviewCount'].transform('max')

    return data.drop_duplicates(subset=['Series_Name', 'Book_Author', 'Book_Category', 'Completion_Percent', 'ReadTogether', 'Review


df_book = integrate_series_data(df_book)
```

Series were identified as duplicates in the data, and a process was undertaken to consolidate them . When recognized as a series, 'Book_Name', 'Book_Author', 'Book_Category', and 'AudioBook' fields were deduplicated. The 'Completion_percent' was replaced with the mean of values from the series, while 'ReadTogether' and 'ReviewCount' were substituted with the max value across the series.

During the process of consolidating the series, missing values in 'Completion_percent' and 'Readtogether' were replaced with 0 after converting text to numbers

Bookdata to CSV

```
df_book.to_csv('all_book_preprocessing.csv', index=False)
```

Review Data

```
df_review = pd.read_csv("all_review.csv")
df_review = df_review.drop(["Unnamed: 0", "Column1", "Column2", "Column3"], axis=1)
df_review[100:120]
```

	Nickname	Review_Text	Book_Name
100	꿈과함께성장한다	너무 재밌습니다 진솔한 이야기이고 부모님을 떠오르게 하는 책이에요	나의 이상하고 평범한 부동산 가족
101	그레이트토리	우연히 발견하고 단숨에 읽었네요. 우리 부모님은 어떤 역경을 딛고 살아오셨을까.....	나의 이상하고 평범한 부동산 가족
102	조심스러운 창의력 대장_959952	대한민국 주택 개발의 미시사. 부동산으로 흥하고 망한 한 가족의 이야기가 쩡하게 와...	나의 이상하고 평범한 부동산 가족
103	예리한 명사_279044	차가운 현실에 얼얼하다가도\n마음이 먹먹해집니다	나의 이상하고 평범한 부동산 가족
104	어야둥기둥기	가볍게 읽기 시작했다가 후반부에 쩡쩡 울면서 봤네요 ㅎㅎㅎ잘 봤어요	나의 이상하고 평범한 부동산 가족
105	선선!	영화 <버블 패밀리> 제작노트 같은 느낌. 좋은 영화이니 함께 보시길 더욱 추천드려요.	나의 이상하고 평범한 부동산 가족
106	cooljun	은퇴준비에 도움이 될만한 내용이네요.	이것은 빠른 경제적 자유를 위한 책
107	제주바다	하느범더 읽어 볼게요	이것은 빠른 경제적 자유를 위한 책
108	vision215	재미있네요	이것은 빠른 경제적 자유를 위한 책

Review Data Preprocessing

```
df_review = df_review.dropna(subset=['Review_Text'])

# Extract only the strings from the 'Review_Text' column
df_review['Review_Text'] = df_review['Review_Text'].apply(lambda x: ' '.join([word for word in str(x).split() if word.isalpha()]))

df_review
```

✓ 0.0s

	Nickname	Review_Text	Book_Name
0	그냥 대충 살기로 했다	내 고추가 필요할 때 잘 달리기 위해서는 평소에 내가 잘 달려야	남자는 어떻게 일어서는가
1	드로니	성기능과 건강은 건강에 왕도는 꾸준한 자기관리가	남자는 어떻게 일어서는가
2	연신내볼닭발	역시	남자는 어떻게 일어서는가
3	유명한 개척자_252881	상대의 말만 듣지말고 상대를 들어보자 말의 한계 어휘부족 부적합한 제스처집중치말기	왜 욱하세요?
4	yyerin	사람이 무례한 사람	왜 욱하세요?
...	...	...	...
22000	비범한 아티스트_792651	재미있네요	종료 D-5\n \n 지옥연가
22001	꼼꼼한 개척자_609322	근데 왜 언니일을두고계속남주탓하지	종료 D-5\n \n 지옥연가

Delete Row with Useless Words

```
####################### 재구매 방지용 없애는 작업
# 필터링할 키워드 리스트
keywords = ["재독서방지", "재구독 방지", '재구매방지', '재열람방지','재구매 방지', '재구독방지용', '재구매 방지용', '재대여방지', '재구독방지']

# 각 키워드에 대해 str.contains를 사용하고 결과들을 OR 연산으로 결합
filter_condition = df_review['Review_Text'].str.contains(keywords[0])
for keyword in keywords[1:]:
    filter_condition |= df_review['Review_Text'].str.contains(keyword)

# 조건에 해당하지 않는 행들을 선택
df_review = df_review[~filter_condition]
```

✓ 0.0s

Identify Preprocessing about Book Name of Review Data with Book Data

Integration about Series Book in "Book Name" Column of Review Data

Delete "종료 D" , "미서비스" text in "Book Name"

```
#book 데이터 전처리한 것 기반으로 실행
texts_to_remove = ['종료 D', '미서비스']
df_review = delete_rows_containing_text(df_review, 'Book_Name', texts_to_remove)

#시리즈 통합
def identify_series_name(book_title, num_words=3):
    # 연속된 num_words 단어를 기준으로 시리즈 이름 생성
    words = book_title.split()
    return ' '.join(words[:num_words]) if len(words) >= num_words else book_title

def integrate_series_data(data):
    # 정규 표현식을 사용하여 괄호와 숫자 등을 제거
    data['Clean_Book_Name'] = data['Book_Name'].str.replace(r'\(\w+\)', '').str.replace(r'\d+권|\d+부|\d+화|\(완결\)|\(완료\)', '', regex=True)

    # 시리즈 이름을 식별하여 새로운 열에 저장
    data['Series_Name'] = data['Clean_Book_Name'].apply(lambda x: identify_series_name(x))

    # 시리즈 이름별로 첫 번째 Book_Name 찾기
    first_book_name_by_series = data.groupby('Series_Name')['Book_Name'].first().to_dict()

    # Book_Name을 시리즈별 첫 번째 Book_Name으로 통일
    data['Book_Name'] = data['Series_Name'].map(first_book_name_by_series)

    return data.drop('Clean_Book_Name', axis=1)
# 함수 호출
df_review = integrate_series_data(df_review)
```

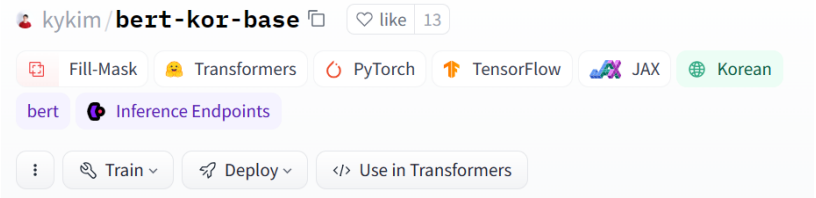
Review Data to CSV

```
df_review.to_csv('all_review_preprocessing.csv', index=False)
```

✓ 0.0s

- **Sentiment analysis**

Since there is no data on whether we prefer it or not in the review, we decide through sentiment analysis. We will use the Korean-based emotional analysis model provided by Hugging.



```
from transformers import BertTokenizerFast, BertModel

tokenizer_bert = BertTokenizerFast.from_pretrained("kykim/bert-kor-base")
model_bert = BertModel.from_pretrained("kykim/bert-kor-base")
```

We will train book reviews to facilitate analysis in book reviews. Reviews related to books such as '간직하고싶어요','읽기 편해요','~권에서 하차합니다', '내용이 장황해요', and '서사가 부족해요' were not well analyzed. Like this, it often appears in book reviews, and various expressions are extracted and labeled manually.

	Review_Text	label
0	좋은책이다 이책을 계기로 많은생각을\n하게되었다...	1
1	인생에도 공략집이 있다. 자의식 해체, 유전자오작동 극복, 독서와 글쓰기	1
2	책이 정말 쉽게 읽히는 편이에여	1
3	너무 따듯해요❣️	1
4	재밌어요 난테크 해봐야겠네여	1
...	...	...
105	필력이 많이 부족하고 읽다보면 잉? 스러운 부분이 꽤 많습니다.	0
106	필력이 부족해서 인물들 감정들이 급발진으로 느껴짐..	0
107	ㅋㅋㅋㅋ필력이 좀 떨어져서 몰입도 또한 떨어지는 작품이지만 진짜 킬링타임용으로는 굿	0
108	애들 장난이네. 재미없네요.	0
109	흡입력이 좀 떨어지는 느낌. 그래도 연자씨가 삶을 대하는 태도는 뭉클하고 기억에 남네요.	0

Next, learn this data into the model and save the new model. Use this model to perform emotional analysis. The tokenizer used the existing model's one

```
review_label = pd.read_csv("finetune_lable.csv")

review_texts = review_label['Review_Text'].tolist()
review_labels = review_label['label'].tolist()

# Review 데이터셋 초기화
review_dataset = BookReviewDataset(review_texts, review_labels, tokenizer)

# BERT 모델 및 감정 분석을 위한 파인튜닝
model = BertForSequenceClassification.from_pretrained('kykim/bert-kor-base', num_labels=2)
optimizer = AdamW(model.parameters(), lr=1e-5)
```

```
# 파인튜닝이 완료된 모델을 저장
model.save_pretrained('book_finetuning_model2')
```

```
# 사전 훈련된 BERT 모델과 토크나이저 불러오기
tokenizer = BertTokenizer.from_pretrained('kykim/bert-kor-base')
model = BertForSequenceClassification.from_pretrained('book_finetuning_model2')
```

The following attachment is the result. We are going to show you some of the parts that show the changes well.

```
model = BertForSequenceClassification.from_pretrained('kykim/bert-kor-base')
```

```
감정: 부정
Review 21: 재미없네요. 장황해요.
감정: 긍정
Review 22: 학교물 감성의 유치한 면 많음. 성인들은 읽기 쉽지 않을 듯. 2권에서 하차.
감정: 긍정
Review 23: 회귀물은 진리지
감정: 긍정
Review 24: 뭔생각으로 이렇게 진행하는지 이해진짜안가네 ㅋㅋㅋ.
감정: 부정
Review 25: 초반인데 참.. 숨겨야되는데 여자옷을 입고가지않나,  본명도 밝히네ㅋㅋ. 어이없어 하차함.
감정: 긍정
Review 26: 기대됩니다!
감정: 긍정
Review 27: 가볍게 읽기좋아요
감정: 긍정
Review 28: 재미있게 읽었어요
에필로그가 없어 아쉽네요
감정: 긍정
Review 29: 여주의 주접이 풍년.. 의술 장면 땜에 주인공들의 서사가 부족해요
감정: 긍정
```

→ This is using ‘kykim/bert-kor-base model’

```
model = BertForSequenceClassification.from_pretrained('book_finetuning_model2')
```

```
감정: 부정
Review 21: 재미없네요. 장황해요.
감정: 부정
Review 22: 학교물 감성의 유치한 면 많음. 성인들은 읽기 쉽지 않을 듯. 2권에서 하차.
감정: 부정
Review 23: 회귀물은 진리지
감정: 긍정
Review 24: 뭔생각으로 이렇게 진행하는지 이해진짜안가네 ㅋㅋㅋ.
감정: 부정
Review 25: 초반인데 참.. 숨겨야되는데 여자옷을 입고가지않나,  본명도 밝히네ㅋㅋ. 어이없어 하차함.
감정: 부정
Review 26: 기대됩니다!
감정: 긍정
Review 27: 가볍게 읽기좋아요
감정: 긍정
Review 28: 재미있게 읽었어요
에필로그가 없어 아쉽네요
감정: 긍정
Review 29: 여주의 주접이 풍년.. 의술 장면 땜에 주인공들의 서사가 부족해요
```

→ This is using our model

We were able to confirm the results we wanted.

- **Handling Missing Data (if it is available)**

This process is in “Data Processing”.

- **Data Integration**

This process is in “Data Processing”

- **Experiments**

- Train & Test Split

I marked it in the algorithm below. (red box)

- k-fold cross validation (if applicable)

I marked it in the algorithm below. (red box)

2.3 Algorithms

Using Data set “all_review_preprocessing”, “all_book_preprocessing”.

1. SVD Collaborative Filtering with the Surprise Library

```
review_data = pd.read_csv("all_review_preprocessing.csv")
```
✓ 0.0s Python

```
review_data.tail()
```
✓ 0.0s Python

	Nickname	Review_Text	Book_Name	Label	Series_Name
15563	꿈이현실	주린이들이 꼭 읽어봐야 할 책 이네요. 오랜만에 주식관련 책을 재미있게 잘 읽었습니다.	세력주 투자 기술	1	세력주 투자 기술
15564	지미오양	주식 잘 모르겠으면 이것부터 읽어보시라	세력주 투자 기술	1	세력주 투자 기술
15565	플러스비	장난이 너무 심한거 아니요?	세력주 투자 기술	0	세력주 투자 기술
15566	ADHD 직장인의 일생다반사	투자에 공부에 대한 마음에 다시끔 불을 지펴준 책	세력주 투자 기술	1	세력주 투자 기술
15567	kosmes2112029	도움이 많이되는 실전 노하우를 알려주는 좋은책	세력주 투자 기술	1	세력주 투자 기술

```
book_data = pd.read_csv("all_book_preprocessing.csv")
```
✓ 0.0s Python

```
book_data.tail()
```
✓ 0.0s Python

	Book_Name	Book_Author	Book_Category	Completion_Percent	ReadTogether	ReviewCount	AudioBook	Series_Name	Content
8641	150세에도 뭘 거야! 2편	유왕규	과학	0.0	1	0	X	150세에도 뭘 거야!	150세에도 뭘 거야! 2편 유왕규 과학
8642	다음 생엔 돌이 되고 싶다.	강다희	에세이	0.0	3	0	X	다음 생엔 돌이	다음 생엔 돌이 되고 싶다. 강다희 에세이
8643	배터리 다이제스트 TOP9	선우 준	과학	0.0	5	0	X	배터리 다이제스트 TOP9	배터리 다이제스트 TOP9 선우 준 과학
8644	배터리 다이제스트 TOP10	선우 준	과학	0.0	5	0	X	배터리 다이제스트 TOP10	배터리 다이제스트 TOP10 선우 준 과학
8645	나로 돌아가는 여행	곽나원	에세이	0.0	4	0	X	나로 돌아가는 여행	나로 돌아가는 여행 곽나원 에세이

```
from surprise import Reader, Dataset, SVD
from surprise.model_selection import train_test_split
import numpy as np
from surprise.model_selection import cross_validate

reader = Reader(rating_scale=(0, 1))
data = Dataset.load_from_df(review_data[['Nickname', 'Book_Name', 'Label']], reader)

trainset, testset = train_test_split(data, test_size=0.2, random_state=42)

model = SVD()
model.fit(trainset)

predictions = model.test(testset)
```

```
cross_validate(model, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
```

✓ 1.3s

```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.3198  0.3148  0.3172  0.3168  0.3129  0.3163  0.0023
MAE (testset)     0.1960  0.1914  0.1930  0.1934  0.1900  0.1928  0.0020
Fit time          0.24    0.23    0.23    0.24    0.23    0.23    0.01
Test time         0.03    0.03    0.03    0.03    0.03    0.03    0.00
```

```
user = "무모한"
bname = "꿀벌과 천둥" # Changing the book name which user read.

if user not in review_data['Nickname'].unique():
    print(f"'{user}' 사용자는 리뷰 데이터에 없습니다.")
elif bname not in review_data['Book_Name'].unique():
    print(f"'{bname}' 책은 리뷰 데이터에 없습니다.")
else:
    pred = model.predict(user, bname, verbose=True)
```

✓ 0.0s

```
user: 무모한        item: 꿀벌과 천둥     r_ui = None   est = 0.95   {'was_impossible': False}
```

First, we use SVD Collaborative Filtering with the Surprise Library. We set that 0 is a negative score, 1 is a positive score. SVD is a latent factor based algorithm. Using the equation "$M = U * \sum * V^T$" This algorithm has about 0.3 RMSE, 0.19 MAE. There are many books with a predicted value 1. So we recommend books in the order of the books with the highest number of "Readtogethers" in Book_data

2. KNN Collaborative Filtering with Surprise Library.

```python
from surprise import Dataset, Reader
from surprise.model_selection import cross_validate
from surprise import KNNBasic
from surprise import accuracy

reader = Reader(rating_scale=(0, 1))
data = Dataset.load_from_df(review_data[['Nickname', 'Book_Name', 'Label']], reader)

# Using KNNBasic
algo = KNNBasic()

algo.fit(trainset)

predictions = algo.test(testset)

rmse = accuracy.rmse(predictions)

mae = accuracy.mae(predictions)
```

✓ 0.7s Python

```
Computing the msd similarity matrix...
Done computing similarity matrix.
RMSE: 0.3356
MAE:  0.2072
```

```python
cross_validate(algo, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
```

✓ 4.0s

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	0.3283	0.3445	0.3362	0.3351	0.3395	0.3367	0.0053
MAE (testset)	0.2011	0.2096	0.2067	0.2068	0.2078	0.2064	0.0029
Fit time	0.55	0.66	0.55	0.50	0.49	0.55	0.06
Test time	0.18	0.16	0.17	0.17	0.16	0.17	0.01

Additionally, we use the KNN algorithm with the Surprise Library. KNN is the K-nearest neighborhood which is a basic Collaborative Filtering algorithm. KNN finds users similar to the selected user and recommends books read by those users. In our data, SVD has better performance than KNN algorithm.

```python
from sklearn.neighbors import NearestNeighbors
import numpy as np

user_book_matrix = review_data.pivot_table(index='Nickname', columns='Book_Name', values='Label')

user_book_matrix = user_book_matrix.fillna(0)

model_knn = NearestNeighbors(metric='cosine', algorithm='brute', n_neighbors=5, n_jobs=-1)

model_knn.fit(user_book_matrix)

def get_user_recommendations(user_id, user_book_matrix, model_knn, n_recommendations=5):
    user_idx = list(user_book_matrix.index).index(user_id)

    user_data = user_book_matrix.iloc[user_idx].values.reshape(1, -1)

    distances, indices = model_knn.kneighbors(user_data, n_neighbors=n_recommendations+1)

    similar_users_book_ratings = user_book_matrix.iloc[indices.flatten()[1:]]
    avg_ratings = similar_users_book_ratings.mean(axis=0)

    already_read = user_book_matrix.loc[user_id]
    avg_ratings = avg_ratings[already_read.isna() | (already_read == 0)]

    recommended_books = avg_ratings.sort_values(ascending=False).head(n_recommendations).index.tolist()

    return recommended_books

user_id = '무모한'
recommended_books = get_user_recommendations(user_id, user_book_matrix, model_knn)

print("추천된 책 목록:", recommended_books)
```

```
추천된 책 목록: ['사라진 여자들', '꿀벌의 예언 1', '나는 미니멀 유목민 입니다', '상식이 결여된 카페', '내 몸의 설계자, 호르몬 이야기', '본심', "'시'가 머물러 있는 그곳에 글자락이 메아리 치다", '용의 만화경', '우리 MBTI가 같네요!', '우등생논술 2023년 7월호']
```

These are 10 books recommended with KNN algorithms.

3. Content-based with Book Metadata(Book_Name, Author, Genre)

```python
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

vectorizer = CountVectorizer()

book_features = book_data[['Book_Name', 'Book_Author', 'Book_Category']].astype(str).agg(' '.join, axis=1)
count_matrix = vectorizer.fit_transform(book_features)

cosine_sim = cosine_similarity(count_matrix)

def get_recommendations_with_rounded_scores(book_name, cosine_sim=cosine_sim):
    idx = book_data.index[book_data['Book_Name'] == book_name].tolist()

    if not idx:
        return []

    idx = idx[0]

    sim_scores = list(enumerate(cosine_sim[idx]))

    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)

    sim_scores = sim_scores[1:11]
    book_indices_scores = [(book_data['Book_Name'].iloc[i[0]], round(i[1], 2)) for i in sim_scores]

    return book_indices_scores

selected_book_name = "꿀벌과 천둥"
recommendations = get_recommendations_with_rounded_scores(selected_book_name)

recommendations
```

```
[('플로라', 0.4),
 ('평균의 종말', 0.4),
 ('초기업', 0.4),
 ('본심', 0.4),
 ('장내세균의 역습', 0.4),
 ('남자아이 대백과', 0.37),
 ('성차별주의는 전쟁을 불러온다', 0.37),
 ('안아주는 말들', 0.37),
 ('세계사를 바꾼  의 책', 0.37),
 ('홀로서기 심리학', 0.37)]
```

We use Content-based using Book metadata. In order to obtain cosine similarity, a matrix was created using the Countvectorizer. We set the book name is “꿀벌과 천둥” which is different book name in presentation.

4. Combining Collaborative + Content-based

```
nickname = '무모한'
book_name = '꿀벌과 천둥' # Chaning book name which user read.
top_n = 10
recommendations = get_book_hybrid_recommendations(nickname, book_name, top_n)
recommendations
```

✓ 30.9s

```
['성차별주의는 전쟁을 불러온다기계가 언어를 이해하는 방법 (자연어 처리 기술 이해)',
 '처음 읽는 여성 철학사감정은 습관이다',
 '디즈니의 악당들 1 : 사악한 여왕제노사이드',
 '당신은 어떤 가면을 쓰고 있나요연남동 빙굴빙굴 빨래방',
 '플로라인생에서 8가지 일에만 집중하라',
 '틀을 깨는 사고력나는 왜 자꾸 내 탓을 할까',
 '홀로서기 심리학서평 쉽게 쓰는 법',
 '남자아이 대백과나는 왜 저 인간이 싫을까?',
 '잉글사이드의 릴라비스트로 쿠킹 앳 홈',
 '처음 읽는 클래식 음악의 역사나에겐 상처받을 이유가 없다']
```

Content based function is similar to our lecture. We get some text from the book author, category and name. Additionally, It is combined with the Collaborative Filtering function to predict how much the user would like for items that the user has not yet evaluated.

In the Collaborative Filtering, we generate a test set from a training set and generate by predicting pairs of users and items that are not in the training set. In other words, we predict ratings for items that users have not yet interacted with, which can be a feature of this model that we built to fit our data.

5. Collaborative Filtering Using **fastai** Library

```
# pip install fastai

from sklearn.model_selection import KFold
from fastai.collab import CollabDataLoaders, collab_learner
from sklearn.metrics import mean_squared_error, mean_absolute_error

data = review_data[['Nickname', 'Book_Name', 'Label']]

kf = KFold(n_splits=5)
maes, rmses = [], []

for train_idx, test_idx in kf.split(data):
    train_data, test_data = data.iloc[train_idx], data.iloc[test_idx]

    dls = CollabDataLoaders.from_df(train_data, user_name='Nickname', item_name='Book_Name', rating_name='Label', bs=64)

    learn = collab_learner(dls, n_factors=100, y_range=(0, 1), wd=0.1, use_nn=True, layers=[100])
    learn.fit_one_cycle(5, 1e-3)

    test_dl = dls.test_dl(test_data)
    predictions, _ = learn.get_preds(dl=test_dl)

    mae = mean_absolute_error(test_data['Label'], predictions)
    rmse = np.sqrt(mean_squared_error(test_data['Label'], predictions))

    maes.append(mae)
    rmses.append(rmse)

avg_mae = np.mean(maes)
avg_rmse = np.mean(rmses)

print(f"Average MAE: {avg_mae:.3f}, Average RMSE: {avg_rmse:.3f}")
```

```
Average MAE: 0.199, Average RMSE: 0.346
```

Additionally, we use Collaborative Filtering with deep learning using fastai library. This Train set, Test set is different from the Surprise Library(These libraries have different functions like “from sklearn.model_selection import train_test_spit” and “from surprise.model_selection import train_test_split”. So, we use K-fold validation. To prevent overfitting, we set epochs to be 5. The result is Average MAE is 0.199, Average RMSE is 0.346.

3 Result and Analysis

1. This is RMSE, MAE, which the SVD surprise library has

```
Evaluating RMSE, MAE of algorithm SVD on 5 split(s).

                  Fold 1  Fold 2  Fold 3  Fold 4  Fold 5  Mean    Std
RMSE (testset)    0.3198  0.3148  0.3172  0.3168  0.3129  0.3163  0.0023
MAE (testset)     0.1960  0.1914  0.1930  0.1934  0.1900  0.1928  0.0020
Fit time          0.24    0.23    0.23    0.24    0.23    0.23    0.01
Test time         0.03    0.03    0.03    0.03    0.03    0.03    0.00
```

2. This is RMSE, MAE, which the KNN surprise library has.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (testset)	0.3283	0.3445	0.3362	0.3351	0.3395	0.3367	0.0053
MAE (testset)	0.2011	0.2096	0.2067	0.2068	0.2078	0.2064	0.0029
Fit time	0.55	0.66	0.55	0.50	0.49	0.55	0.06
Test time	0.18	0.16	0.17	0.17	0.16	0.17	0.01

3. This is RMSE, MAE which the deep learning **fastai** library has.

```
Average MAE: 0.199, Average RMSE: 0.346
```

When comparing algorithms using collaborative filtering, the worst is the KNN algorithm and the best looking is the SVD. But we don't have details about deep learning. So, if we have more insight about deep learning, we think that the deep learning model will be better than SVD.

4 Conclusion (and recommendations)

4-1. Conclusion

If a user is new, ask for a book the user wants or a book user likes and recommend it as content-based. For example, the user wants to read “꿀벌과 천둥” the result is below picture.

```
[('플로라', 0.4),
 ('평균의 종말', 0.4),
 ('초기업', 0.4),
 ('본심', 0.4),
 ('장내세균의 역습', 0.4),
 ('남자아이 대백과', 0.37),
 ('성차별주의는 전쟁을 불러온다', 0.37),
 ('안아주는 말들', 0.37),
 ('세계사를 바꾼  의 책', 0.37),
 ('홀로서기 심리학', 0.37)]
```

When the data is collected, recommendations are made based on SVD, and additionally recommendations are made with an algorithm that combines collaborative filtering and content-based recommendations. And If we use SVD collaborative filtering, we will use “ReadTogether” data for making ranking books like the picture below.

```
Book 3669: 적막한 폭발, Favor Count: 968
Book 6086: 트러블 사전, Favor Count: 965
Book 2861: 아레나옴므플러스 Arena Homme+ 2023년 11월호, Favor Count: 934
Book 8163: 나는 정신장애 아들을 둔 아버지입니다, Favor Count: 929
Book 7492: 쫓기지 않는 50대를 사는 법, Favor Count: 908
Book 18: 66일 인문학 대화법, Favor Count: 904
Book 1761: 김씨네과일, Favor Count: 897
Book 2012: 이코노미 조선 509호 : 2023.09.20, Favor Count: 856
Book 8155: 문장의 시대, 시대의 문장, Favor Count: 845
Book 8137: 컬러는 나를 알고 있다, Favor Count: 827
```

4-2. Limitation

First, the data was too sparse. To solve this problem, it would be a good idea to first gather as much data as possible and use collaborative filtering to filter users who have written more than 3~5 reviews, or to filter books with more than 5~10 reviews. There was not much time in this project, and the number of data was too small when filtering was applied.

Second, we think it would have been better to set the emotional analysis to 1-5. At first, based on the thesis on building a customized emotional analysis for each domain(A Domain Adaptive Sentiment Dictionary Construction Method for Domain Sentiment Analysis[2]), we implemented our review based on the Gunsan University emotional analysis dictionary. However, the performance was not as good as that of BERT, and it was similar to writing a basic emotional analysis dictionary, so I used BERT.

Role of each team members

202103672 Heo Kang: Web Crawling, Sentiment Analysis, Algorithm using surprise library.

201902441 Lee GyuMin: Data preprocessing, Sentiment Analysis, Algorithms using surprise library, fastai library.

201903345 Cho MinJung: Data Preprocessing, EDA, Sentiment Analysis, Hybrid Modeling (CF+Content)

202100640 Kim Minji: Web Crawling, Data Preprocessing, Algorithm using surprise library.

References

1. kykim/bert-kor-base · Hugging Face
2. *Kim Dahae, Cho Taemin, Lee Jee-Hyeong (2018)* 도메인 별 감성분석을 위한 도메인 맞춤형 감성사전 구축 기법*(A Domain Adaptive Sentiment Dictionary Construction Method for Domain Sentiment Analysis) 15~18 pages.*
3. surprise library: https://surpriselib.com/
4. fastai library: https://docs.fast.ai/