인공지능

Presentation

개 인 화 를 위 한 도 서 추 천 시 스 템 분 석

2023. 11. 14

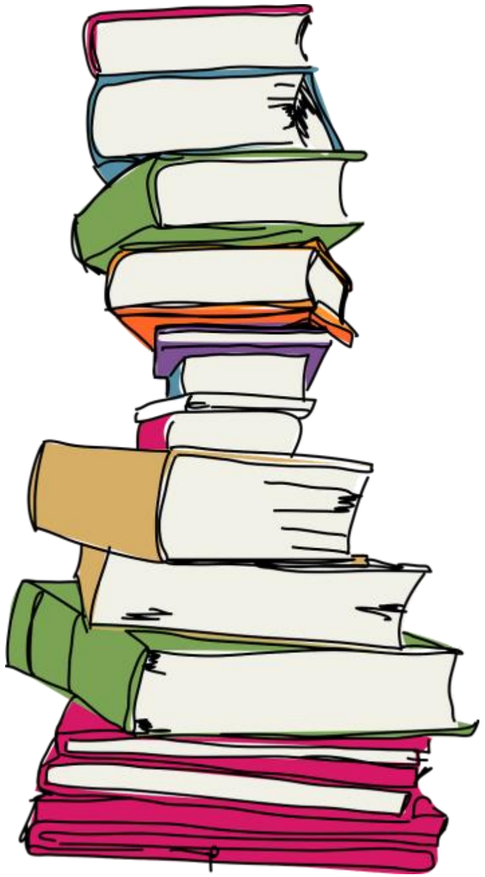
2022208021
컴퓨터공학전공 김재원

Contents

enter the contents

01 | 개요

- 책을 읽다가 취향에 맞지 않아서 다시 책을 골라서 읽는 경우 많음
- 사용자가 재미있게 읽은 책을 바탕으로 책을 추천받으면 더 취향에 맞는 책을 읽을 가능성이 높음
- 아이템 기반 추천 시스템과 컨텐츠 기반 추천 시스템을 이용



02 | 사용 라이브러리

pandas

데이터 조작 및 분석을 위한 파이썬 프로그래밍 언어용으로 작성된 소프트웨어 라이브러리

NumPy

행렬이나 일반적으로 대규모 다차원 배열을 쉽게 처리할 수 있도록 지원하는 파이썬의 라이브러리로 데이터 구조 외에도 수치 계산을 위해 효율적으로 구현된 기능을 제공

NLTK

영어의 기호 및 통계적 자연어 처리를 위한 라이브러리로 분류, 토큰화, 형태소 분석, 태그 지정, 구문 분석 및 의미론적 추론 기능을 지원

02 | 사용 라이브러리

Matplotlib

다양한 데이터를 많은 방법으로 도식화 해주는 파이썬 라이브러리

Scikit-learn

파이썬 프로그래밍 언어용 기계 학습 라이브러리

Seaborn

matplotlib를 기반으로 하는 Python 데이터 시각화 라이브러리

Requests

Requests는 Python 프로그래밍 언어용 HTTP 클라이언트 라이브러리

03 | 데이터 소개

```
books = pd.read_csv('C:/Users/jgimn/Downloads/Preprocessed_data.csv')
books.head()
```

278,858명의 사용자가 약 271,379권의 책에 대해 평가한 1,149,780개의 평점을 제공한다.

- user_id: 사용자 아이디
- location: 사용자 거주 지역
- age: 사용자 나이
- isbn: 국제표준도서번호(국제적으로 책에 붙이는 고유한 식별자)
- rating: 사용자가 도서에 매긴 평점
- book_title: 도서 제목
- book_author: 작가
- year_of_publication: 출판년도
- publisher: 출판사

	Unnamed: 0	user_id	location	age	isbn	rating	book_title	book_author	year_of_publication	publisher
0	0	2	stockton, california, usa	18.0000	0195153448	0	Classical Mythology	Mark P. O. Morford	2002.0	Oxford University Press
1	1	8	timmins, ontario, canada	34.7439	0002005018	5	Clara Callan	Richard Bruce Wright	2001.0	HarperFlamingo Canada
2	2	11400	ottawa, ontario, canada	49.0000	0002005018	0	Clara Callan	Richard Bruce Wright	2001.0	HarperFlamingo Canada
3	3	11676	n/a, n/a, n/a	34.7439	0002005018	8	Clara Callan	Richard Bruce Wright	2001.0	HarperFlamingo Canada

03 | 데이터 소개

- img_s, img_m, img_l: 도서 표지 이미지 링크
- Summary: 줄거리
- Language: 언어
- Category: 도서 카테고리
- city, state, country: 사용자 거주 지역

img_s	img_m	img_l
http://images.amazon.com/images/P/0195153448.0...	http://images.amazon.com/images/P/0195153448.0...	http://images.amazon.com/images/P/0195153448.0...
http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...
http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...
http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...

Summary	Language	Category	city	state	country
Provides an introduction to classical myths pl...	en	['Social Science']	stockton	california	usa
In a small town in Canada, Clara Callan reluct...	en	['Actresses']	timmins	ontario	canada
In a small town in Canada, Clara Callan reluct...	en	['Actresses']	ottawa	ontario	canada
In a small town in Canada, Clara Callan reluct...	en	['Actresses']	NaN	NaN	NaN

03 | 데이터 소개

```
books.info()
```

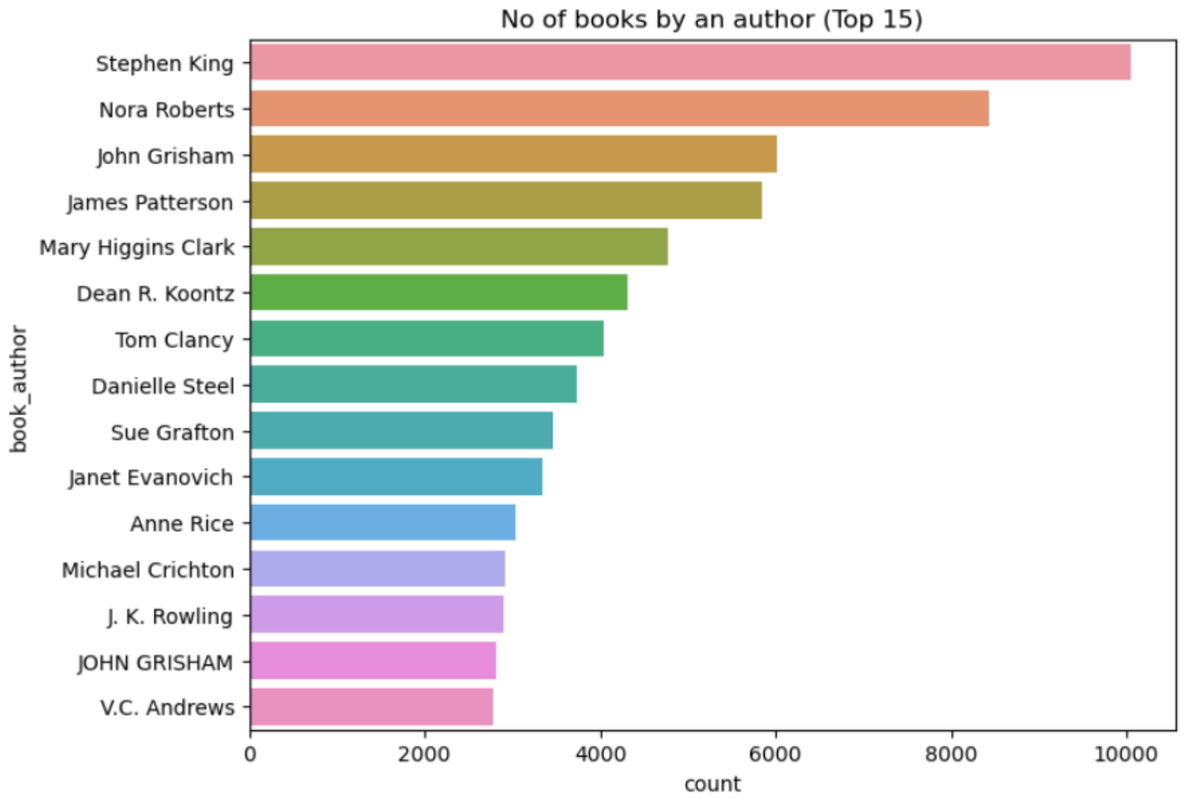
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1031175 entries, 0 to 1031174
Data columns (total 19 columns):
 #   Column               Non-Null Count    Dtype
---  ------               --------------    -----
 0   Unnamed: 0           1031175 non-null  int64
 1   user_id              1031175 non-null  int64
 2   location             1031175 non-null  object
 3   age                  1031175 non-null  float64
 4   isbn                 1031175 non-null  object
 5   rating               1031175 non-null  int64
 6   book_title           1031175 non-null  object
 7   book_author          1031175 non-null  object
 8   year_of_publication  1031175 non-null  float64
 9   publisher            1031175 non-null  object
 10  img_s                1031175 non-null  object
 11  img_m                1031175 non-null  object
 12  img_l                1031175 non-null  object
 13  Summary              1031175 non-null  object
 14  Language             1031175 non-null  object
 15  Category             1031175 non-null  object
 16  city                 1017072 non-null  object
 17  state                1008377 non-null  object
 18  country              995801 non-null   object
dtypes: float64(2), int64(3), object(14)
memory usage: 149.5+ MB
```

1,031,175개의 데이터가 있다.

city, state, country에 누락된 값이 존재하지만 사용하지 않으므로 데이터 전처리 단계에서 삭제한다.

04 | 데이터 분석 및 시각화

```
plt.figure(figsize=(8,6))
sns.countplot(y='book_author', data=books,order=books['book_author'].value_counts().index[0:15])
plt.title("No of books by an author (Top 15)")
```



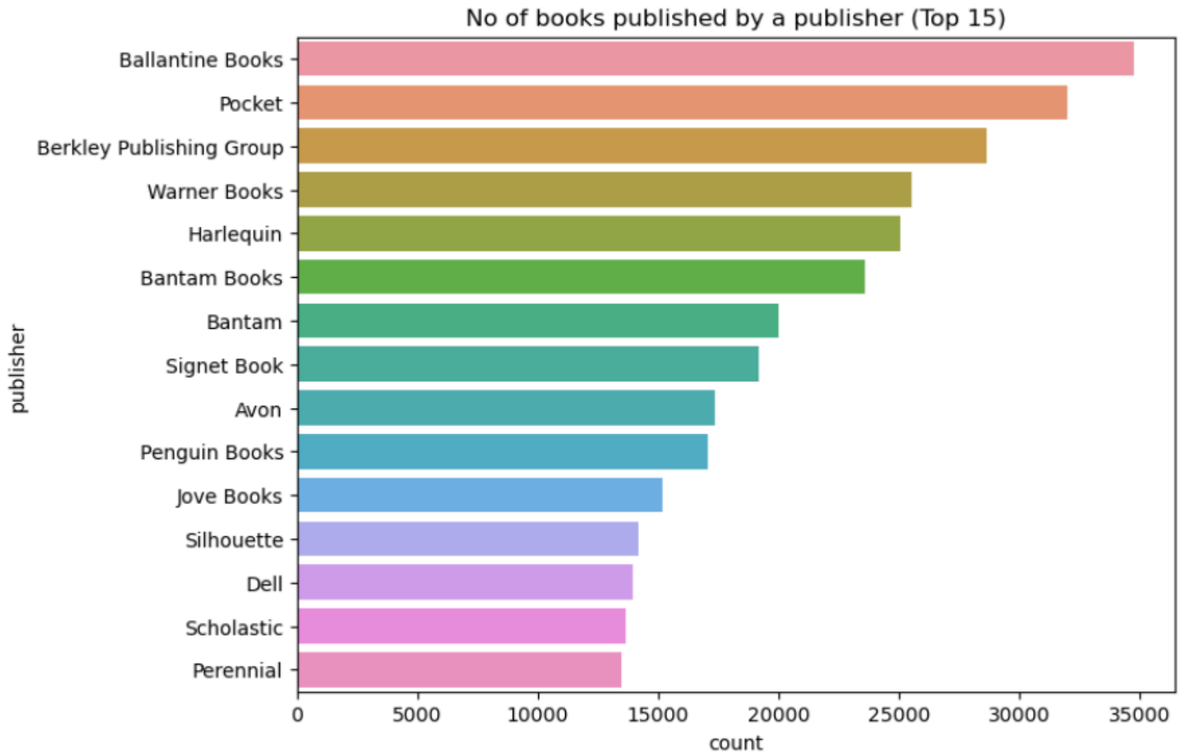
도서 데이터가 많은 상위 15명의 작가를 보여준다.

Stephen King, Nora Roberts, John Grisham 작가 순으로 도서 데이터가 많은 것을 알 수 있다.

04 | 데이터 분석 및 시각화

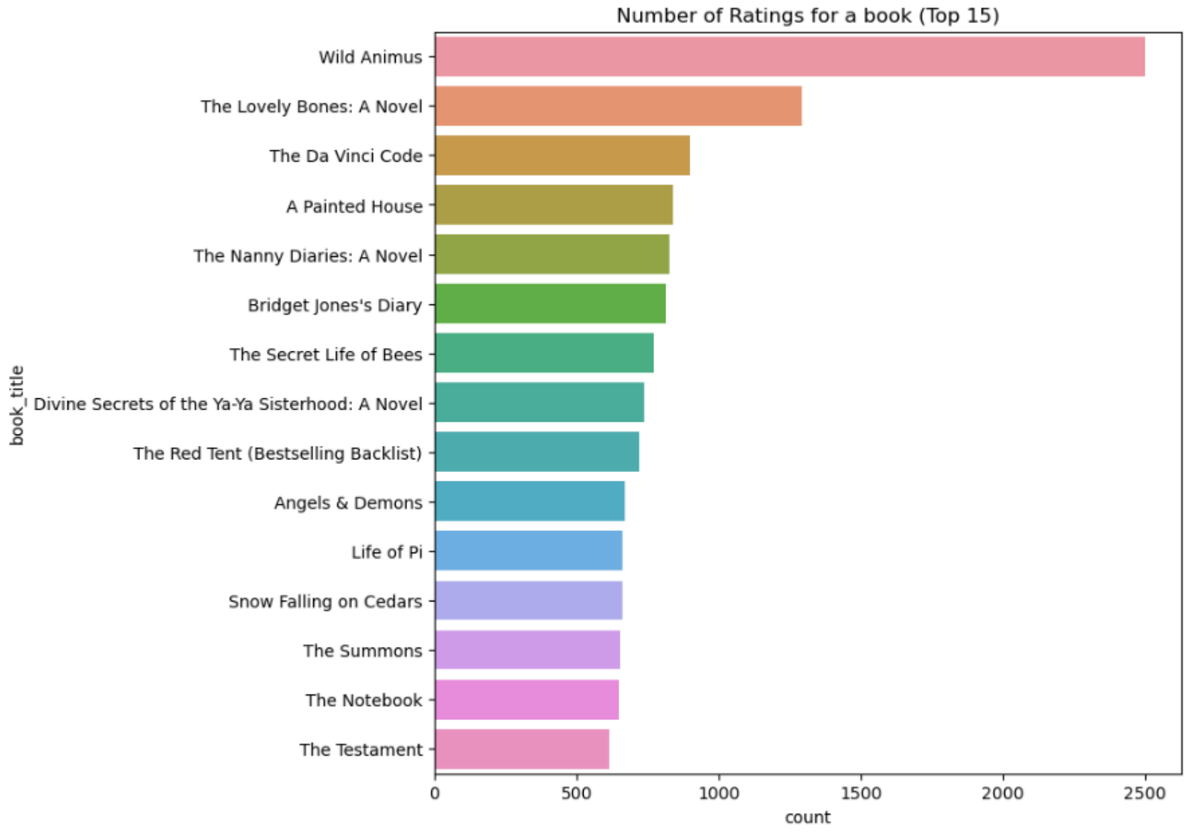
```
plt.figure(figsize=(8,6))
sns.countplot(y='publisher', data=books,order=books['publisher'].value_counts().index[0:15])
plt.title("No of books published by a publisher (Top 15)")
```

도서 데이터가 많은 상위 15개의 출판사를 보여준다.



04 | 데이터 분석 및 시각화

```
plt.figure(figsize=(8,8))
sns.countplot(y='book_title', data=books, order=books['book_title'].value_counts().index[0:15])
plt.title("Number of Ratings for a book (Top 15)")
```



평점을 가장 많이 받은 15개의 도서를 보여준다.

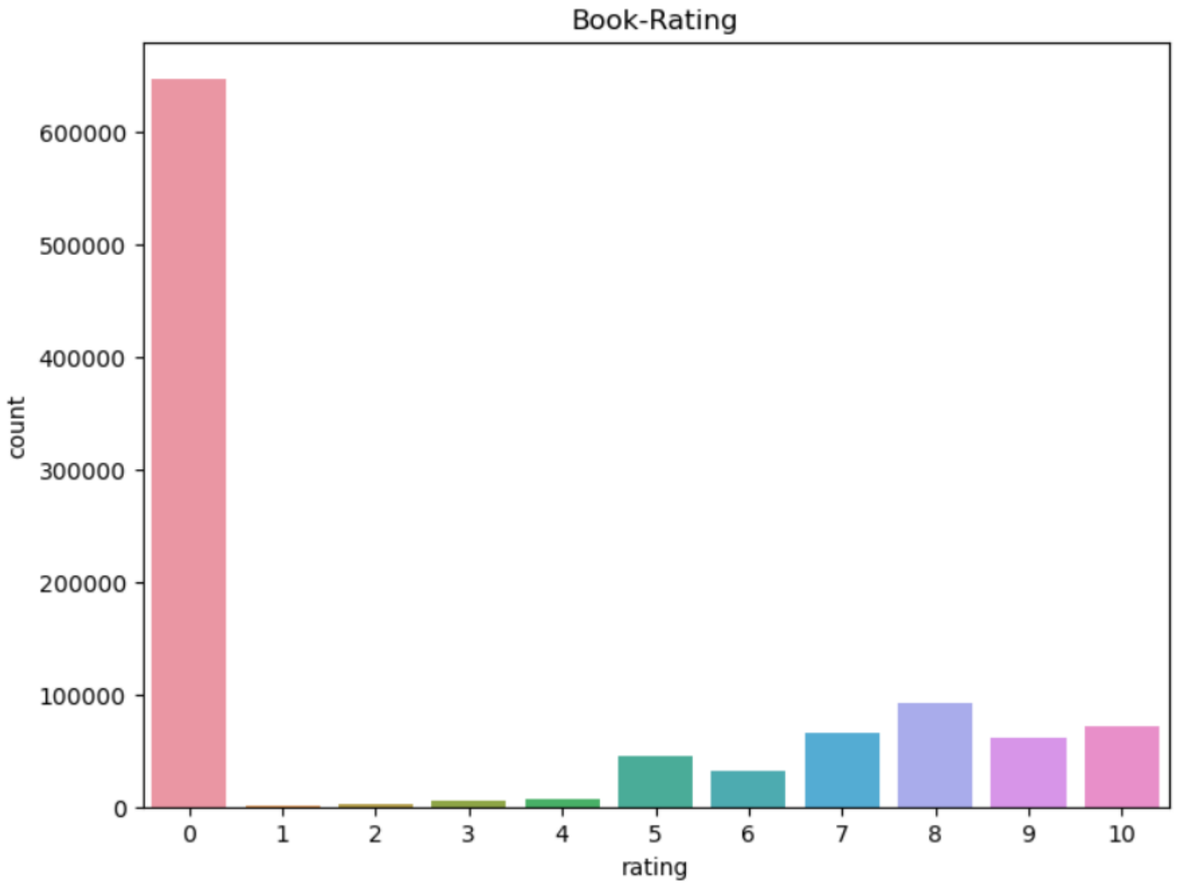
Wild Animus, The Lovely Bones: A Novel, The Da Vinci Code 순으로 사용자의 평점이 많다.

04 | 데이터 분석 및 시각화

```
plt.figure(figsize=(8,6))
sns.countplot(x="rating", data=books['rating'])
plt.title("Book-Rating")
```

0점부터 10점까지 평점의 분포를 보여준다.

0점으로 rating된 데이터는 사용할 수 없기 때문에 전처리 단계에서 삭제해야 한다.



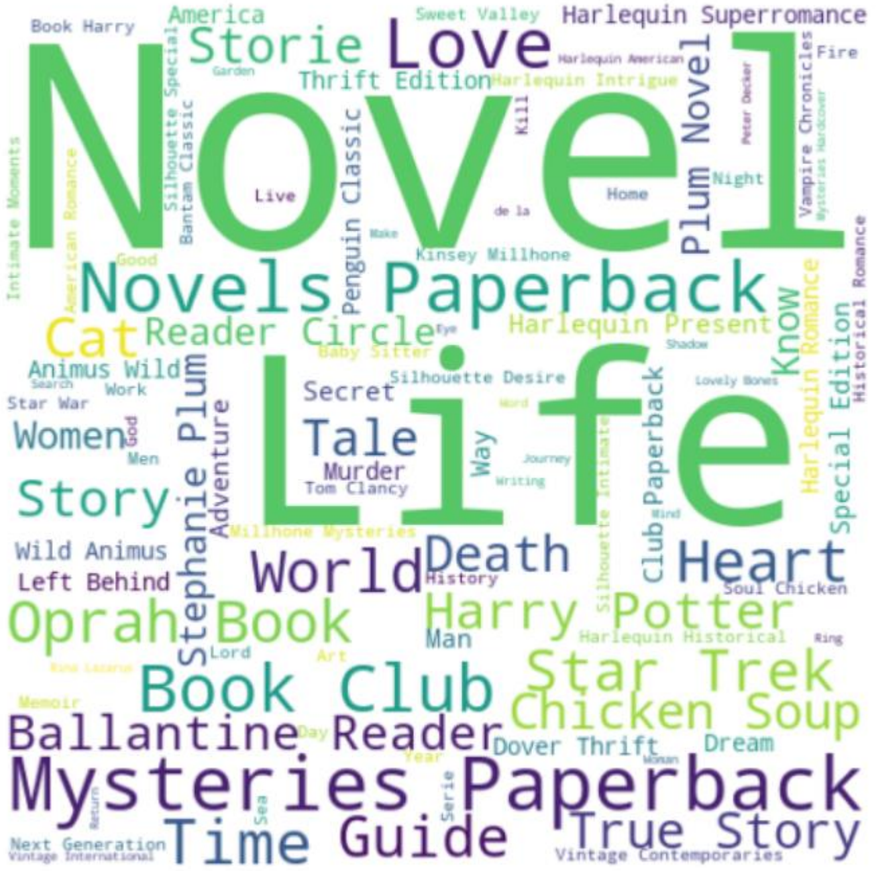
04 | 데이터 분석 및 시각화

```
plt.subplots(figsize=(8,6))
wc = WordCloud(background_color="white", max_words=100, stopwords=STOPWORDS, max_font_size=256,
               random_state=42, width=500, height=500)
wc.generate(' '.join(books['book_title']))
plt.imshow(wc, interpolation="bilinear")
plt.axis('off')
plt.show()
```

워드 클라우드(word cloud)

워드 클라우드는 메타 데이터에서 얻어진 태그들을 분석하여 중요도나 인기도 등을 고려하여 시각적으로 늘어놓아 표시하는 것이다. 보통은 2차원의 표와 같은 형태로 태그들이 배치되며 이때 순서는 알파벳/가나다 순으로 배치된다. 시각적인 중요도를 강조를 위해 각 태그들은 그 중요도(혹은 인기도)에 따라 글자의 색상이나 굵기 등 형태가 변한다.

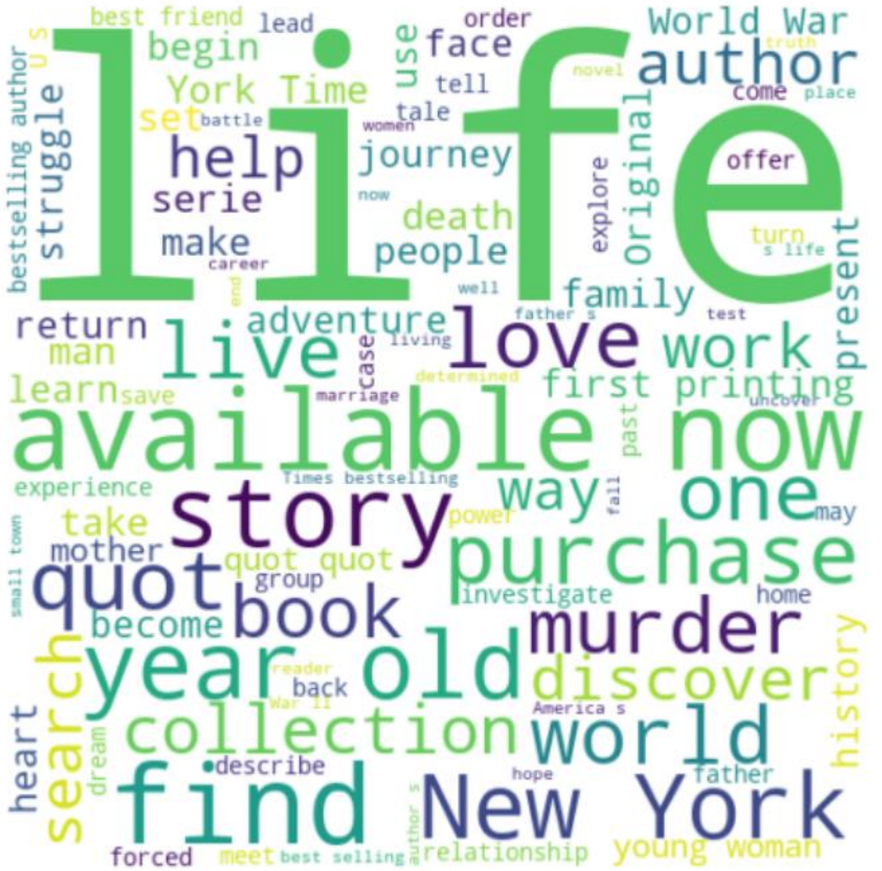
도서 제목에 대한 워드 클라우드를 보여준다.



04 | 데이터 분석 및 시각화

```
plt.subplots(figsize=(8,6))
wc = WordCloud(background_color="white", max_words=100, stopwords=STOPWORDS, max_font_size=256,
               random_state=42, width=500, height=500)
wc.generate(' '.join(books['Summary']))
plt.imshow(wc, interpolation="bilinear")
plt.axis('off')
plt.show()
```

줄거리에 대한 워드 클라우드를 보여준다.



05 | 데이터 전처리

```
df = books.copy()
df.dropna(inplace=True)
df.reset_index(drop=True, inplace=True)

df.drop(columns = ['Unnamed: 0','location','isbn','img_s','img_m','city','age',
                   'state','Language','country','year_of_publication'],axis=1,inplace = True)
#필요없는 컬럼 삭제

df.drop(index=df[df['Category'] == '9'].index, inplace=True)

df.drop(index=df[df['rating'] == 0].index, inplace=True)
#평점이 0인 경우 삭제

df['Category'] = df['Category'].apply(lambda x: re.sub('[₩_]+',' ',x).strip())
#카테고리에서 특수문자 제거

df.head()
```

필요 없는 데이터인 Unnamed: 0, location, isbn, img_s, img_m, city, age, state, Language, country, year_of_publication을 제거한다.

평점이 0인 데이터를 제거한다.

카테고리의 특수문자를 제거한다.

	user_id	rating	book_title	book_author	publisher	img_l	Summary	Category
1	8	5	Clara Callan	Richard Bruce Wright	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	In a small town in Canada, Clara Callan reluct...	Actresses
4	67544	8	Clara Callan	Richard Bruce Wright	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	In a small town in Canada, Clara Callan reluct...	Actresses
7	123629	9	Clara Callan	Richard Bruce Wright	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	In a small town in Canada, Clara Callan reluct...	Actresses
9	200273	8	Clara Callan	Richard Bruce Wright	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	In a small town in Canada, Clara Callan reluct...	Actresses
10	210926	9	Clara Callan	Richard Bruce Wright	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	In a small town in Canada, Clara Callan reluct...	Actresses

06 | 아이템 기반 추천 시스템

아이템 기반 추천 시스템

- columns는 item, rows는 user 행렬을 가진다.
- 아이템A와 아이템B는 비슷한 평점분포를 가지고 있기 때문에 아이템A와 아이템B를 비슷하다고 판단한다.
- user3에게 아이템B를 추천해준다.

	아이템 A	아이템 B	아이템 C
	3	3	5
	5	4	-
	4	-	1
	-	2	3

출처: https://ai-with-sudal-ee.tistory.com/6

06 | 아이템 기반 추천 시스템

```
book_title = 'The Street Lawyer'
#책 제목 입력

rating_counts = pd.DataFrame(df['book_title'].value_counts())
rare_books = rating_counts[rating_counts['book_title'] <= 180].index
common_books = df[~df['book_title'].isin(rare_books)]

#유저 아이디와 책 제목 별로 평점을 구하기(피벗 테이블 만들기)
user_book_df = common_books.pivot_table(index=['user_id'], columns=['book_title'], values='rating')
user_book_df.head()
```

도서에 대한 평점의 개수를 카운트한다.

평점의 개수가 180개 이하인 도서의 book_title을 rare_books에 저장

book_title이 rare_books에 속하는 않는 데이터를 common_books에 저장한다.

유저 아이디와 책 제목 별로 평점을 구한다. (피벗 테이블 만들기)

book_title	A Painted House	A Time to Kill	Bridget Jones's Diary	Divine Secrets of the Ya-Ya Sisterhood: A Novel	Fahrenheit 451	Girl with a Pearl Earring	Good in Bed	Harry Potter and the Goblet of Fire (Book 4)	Harry Potter and the Order of the Phoenix (Book 5)	Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))	...	The Nanny Diaries: A Novel	The Red Tent (Bestselling Backlist)	The Secret Life of Bees	The Street Lawyer
user_id															
26	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
51	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	9.0
183	NaN	NaN	NaN	NaN	9.0	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
242	NaN	NaN	NaN	NaN	8.0	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN
243	7.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN

06 | 아이템 기반 추천 시스템

```
book = user_book_df[book_title]
recom_data = pd.DataFrame(user_book_df.corrwith(book).sort_values(ascending=False)).reset_index(drop=False)
recom_data.head()
```

	book_title	0
0	The Street Lawyer	1.000000
1	Girl with a Pearl Earring	0.960769
2	Tuesdays with Morrie: An Old Man, a Young Man,...	0.845154
3	The Testament	0.844485
4	The Red Tent (Bestselling Backlist)	0.817424

corrwith(data) : 특정 컬럼에 대한 상관관계 계수 표현

book_title 변수의 도서와 다른 도서들의 상관관계를 저장한다.

book_title의 도서와 비슷한 도서순으로 정렬한다.

06 | 아이템 기반 추천 시스템

```
if book_title in [book for book in recom_data['book_title']]:
    recom_data = recom_data.drop(recom_data[recom_data['book_title'] == book_title].index[0])
    #추천받으려고 하는 book_title이 포함되기 때문에 해당 행 삭제 필요

recom_book1 = []
recom_book1 = recom_data['book_title'].values.tolist()
del recom_book1[5:]
recom_book1
```

```
['Girl with a Pearl Earring',
 "Tuesdays with Morrie: An Old Man, a Young Man, and Life's Greatest Lesson",
 'The Testament',
 'The Red Tent (Bestselling Backlist)',
 'The Secret Life of Bees']
```

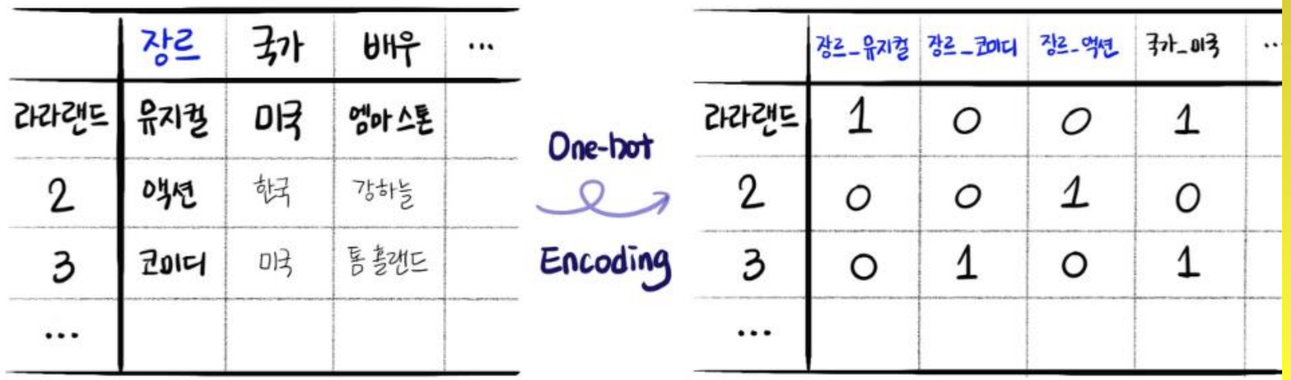
추천받으려고 할때 기준이 되는 도서도 포함되기 때문에 해당 데이터를 제거해야 한다.

추천 도서 5권을 보여준다.

07 | 컨텐츠 기반 추천 시스템

컨텐츠 기반 추천 시스템

- 컨텐츠와 관련된 항목들을 feature로 분류한다.
- 학습을 위한 input 데이터를 만들기 위해 테이블 형태의 정형화된 데이터 형식을 가정하고 테이블의 행에는 컨텐츠의 제목을, 열에는 feature를 나열한다.
- feature와 컨텐츠의 값을 합쳐 새로운 feature를 생성하여 해당 feature에 속하면 1, 아니면 0의 값을 부여한다. 이를 one-hot encoding이라고 한다.
- 라라랜드와 비슷한 컨텐츠를 찾기 위해 [1,0,0,1,...]과 비슷한 컨텐츠를 찾는다.



출처: https://ai-with-sudal-ee.tistory.com/4

07 | 컨텐츠 기반 추천 시스템

Title, Author, Publisher, Category

```
book_title = 'The Testament'

rating_counts = pd.DataFrame(df['book_title'].value_counts())
rare_books = rating_counts[rating_counts['book_title'] <= 100].index
common_books = df[~df['book_title'].isin(rare_books)]

common_books = common_books.drop_duplicates(subset=['book_title'])
#책 제목이 중복되는 행을 삭제

common_books.reset_index(inplace= True)
common_books['index'] = [i for i in range(common_books.shape[0])]
#인덱스 재정렬
common_books.head()
```

도서 제목이 중복되는 행을 삭제한다.

데이터의 인덱스를 재정렬한다.

	index	user_id	rating	book_title	book_author	publisher	img_l	Summary	Category
0	0	3329	8	The Testament	John Grisham	Dell	http://images.amazon.com/images/P/0440234743.0...	A suicidal billionaire, a burnt-out Washington...	Fiction
1	1	899	2	Wild Animus	Rich Shapero	Too Far	http://images.amazon.com/images/P/0971880107.0...	Wild animus is a search for the primordial, a ...	Fiction
2	2	16	9	Airframe	Michael Crichton	Ballantine Books	http://images.amazon.com/images/P/0345402871.0...	A fatal mid-air collision involving a commerci...	Fiction
3	3	1376	8	Timeline	MICHAEL CRICHTON	Ballantine Books	http://images.amazon.com/images/P/0345417623.0...	Using a quantum time machine, a group of young...	Fiction
4	4	26	10	To Kill a Mockingbird	Harper Lee	Little Brown & Company	http://images.amazon.com/images/P/0446310786.0...	The unforgettable novel of a childhood in a sl...	Fiction

07 | 컨텐츠 기반 추천 시스템

Title, Author, Publisher, Category

```
target_cols = ['book_title','book_author','publisher','Category']#타겟팅할 feature 정하기
common_books['combined'] = [' '.join(common_books[target_cols].iloc[i,].values) for i in range(common_books[target_cols].shape[0])]
#feature들의 값을 저장
common_books[target_cols].shape[0]
common_books.head()
```

```
cv = CountVectorizer()
count_matrix = cv.fit_transform(common_books['combined'])

column_names = cv.get_feature_names_out()
cv_df = pd.DataFrame(count_matrix.toarray(), columns = column_names)
cv_df.head()
```

	1st	45	451	against	agency	airframe	albom	alex	alexander	along	...
0	0	0	0	0	0	0	0	0	0	0	...
1	0	0	0	0	0	0	0	0	0	0	...
2	0	0	0	0	0	1	0	0	0	0	...
3	0	0	0	0	0	0	0	0	0	0	...
4	0	0	0	0	0	0	0	0	0	0	...

타겟팅할 feature를 정한다.

feature들의 값을 join하여 combined 열에 저장한다.

combined 열을 벡터화한다. (단어별 개수를 카운팅)

one-hot encoding의 형식으로 변환한다.

07 | 컨텐츠 기반 추천 시스템

Title, Author, Publisher, Category

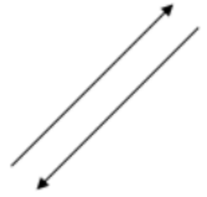
```
cosine_sim = cosine_similarity(count_matrix)
#벡터화된 값의 코사인 유사도 구하기
cosine_sim
```

```
array([[1.        , 0.15430335, 0.16666667, ..., 0.72168784, 0.28867513,
        0.36514837],
       [0.15430335, 1.        , 0.15430335, ..., 0.13363062, 0.13363062,
        0.16903085],
       [0.16666667, 0.15430335, 1.        , ..., 0.14433757, 0.14433757,
        0.18257419],
       ...,
       [0.72168784, 0.13363062, 0.14433757, ..., 1.        , 0.25      ,
        0.31622777],
       [0.28867513, 0.13363062, 0.14433757, ..., 0.25      , 1.        ,
        0.15811388],
       [0.36514837, 0.16903085, 0.18257419, ..., 0.31622777, 0.15811388,
        1.        ]])
```

벡터화된 값의 코사인 유사도를 구한다.

코사인 유사도(Cosine Similarity)

코사인 유사도는 두 벡터 간의 코사인 각도를 이용하여 구할 수 있는 두 벡터의 유사도를 의미한다. 두 벡터의 방향이 완전히 동일한 경우에는 1의 값을 가지며, 90° 의 각을 이루면 0, 180° 로 반대의 방향을 가지면 -1의 값을 갖게 된다. 즉, 결국 코사인 유사도는 -1 이상 1 이하의 값을 가지며 값이 1에 가까울수록 유사도가 높다고 판단할 수 있다.



코사인 유사도 : -1

코사인 유사도 : 0

코사인 유사도 : 1

출처: https://wikidocs.net/24603

07 | 컨텐츠 기반 추천 시스템

Title, Author, Publisher, Category

```
index = common_books[common_books['book_title'] == book_title]['index'].values[0]

sim_books = list(enumerate(cosine_sim[index]))
#리스트의 원소에 순서값을 부여해주는 함수
sorted_sim_books = sorted(sim_books,key=lambda x:x[1], reverse=True)[1:6]
#내림차순으로 정렬하고 5개의 값을 저장
sorted_sim_books
```

```
[(59, 0.8333333333333336),
 (7, 0.7715167498104596),
 (87, 0.7216878364870323),
 (97, 0.7216878364870323),
 (78, 0.6666666666666669)]
```

enumerate: 리스트의 원소에 순서값을 부여해주는 함수

내림차순으로 정렬하고 5개의 값을 저장한다.

```
recom_book2 = []
for i in range(len(sorted_sim_books)):
    recom_book2.append(common_books[common_books['index'] == sorted_sim_books[i][0]]['book_title'].item())
    #리스트에 index 값에 해당하는 book_title 값을 저장
recom_book2
```

```
['The Rainmaker',
 'The Street Lawyer'
 'The Chamber',
 'The Summons',
 'The Brethren']
```

index 값에 해당하는 book_title 값을 저장하여 추천 도서를 출력한다.

07 | 컨텐츠 기반 추천 시스템

Summary

```
book_title = 'To Kill a Mockingbird'

rating_counts = pd.DataFrame(df['book_title'].value_counts())
rare_books = rating_counts[rating_counts['book_title'] <= 100].index
common_books = df[~df['book_title'].isin(rare_books)]

common_books = common_books.drop_duplicates(subset=['book_title'])
common_books.reset_index(inplace= True)
common_books['index'] = [i for i in range(common_books.shape[0])]

summary_filtered = []
for i in common_books['Summary']:#줄거리 필터링하기
    i = re.sub("[^a-zA-Z]"," ",i).lower()#문자가 아닌 데이터를 찾아서 공백으로 대체
    i = nltk.word_tokenize(i)# 단어 단위로 나누기
    i = [word for word in i if not word in set(stopwords.words("english"))]
    #stopwords.words("english")는 NLTK가 정의한 영어 불용어 리스트를 리턴, 불용어 제거
    i = " ".join(i)
    summary_filtered.append(i)
print(summary_filtered[0:5])
```

도서 제목이 중복되는 행을 삭제한다.

데이터의 인덱스를 재정렬한다.

줄거리 필터링한다.

문자가 아닌 데이터를 찾아서 공백으로 대체

단어 단위로 나누기

stopwords.words("english"): NLTK가 정의한 영어 불용어 리스트를 리턴, 불용어 제거

```
['suicidal billionaire burnt washington litigator woman forsaken technology work wilds brazil brought together astounding mystery testamen
t', 'wild animus search primordial test human foundations journey breaking point', 'fatal mid air collision involving commercial airliner p
rompts frantic desperate investigation causes accident thriller exploring issue safety security aircraft industry', 'using quantum time mac
hine group young historians sent back year rescue trapped project leader', 'unforgettable novel childhood sleepy southern town crisis consc
ience rocked kill mockingbird became instant bestseller critical success first published']
```

07 | 컨텐츠 기반 추천 시스템

Summary

```
common_books['Summary'] = summary_filtered
cv = CountVectorizer()
count_matrix = cv.fit_transform(common_books['Summary'])
cosine_sim = cosine_similarity(count_matrix)
index = common_books[common_books['book_title'] == book_title]['index'].values[0]
sim_books = list(enumerate(cosine_sim[index]))
sorted_sim_books = sorted(sim_books,key=lambda x:x[1],reverse=True)[1:6]

recom_book3 = []
for i in range(len(sorted_sim_books)):
    recom_book3.append(common_books[common_books['index'] == sorted_sim_books[i][0]]['book_title'].item())
recom_book3
```

```
["Bridget Jones's Diary",
 'The Green Mile',
 'Empire Falls',
 'The King of Torts',
 'Divine Secrets of the Ya-Ya Sisterhood: A Novel']
```

Summary 열을 벡터화한다. (단어별 개수를 카운팅)

벡터화된 값의 코사인 유사도를 구한다.

내림차순으로 정렬하고 5개의 값을 저장한다.

index 값에 해당하는 book_title 값을 저장하여 추천 도서를 출력

08 | 결론 및 소감

```
fig, axs = plt.subplots(1, 5,figsize=(18,5))
fig.suptitle('You may also like these books', size = 22)
for i in range(len(books)):
    #이미지 주소를 이용하여 이미지 저장 없이 바로 사용
    url = common_books.loc[common_books['book_title'] == books[i],'img_l'][:1].values[0]
    im = Image.open(requests.get(url, stream=True).raw)
    #request.get 요청
    axs[i].imshow(im)
    axs[i].axis("off")
    fig.show()
```

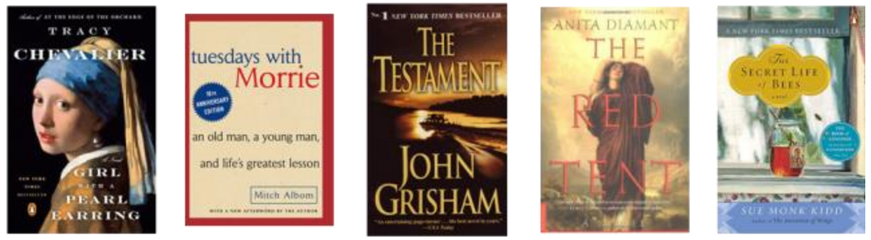
도서 제목 리스트가 아닌 도서 표지 이미지를 사용하여 추천 도서를 보여주기 위해 img_l 데이터를 사용한다.

request.get 요청을 이용하여 이미지 주소를 통해 이미지를 저장 없이 바로 보여준다.

각 추천 시스템을 함수로 만들어 앞에서 추천받은 도서를 표지 이미지로 보여준다.

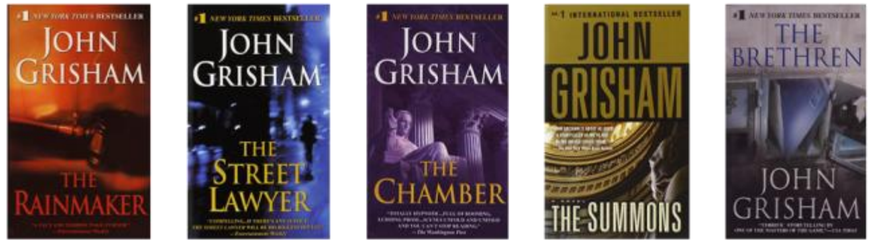
```
item_based_recommender('The Street Lawyer')
```

You may also like these books



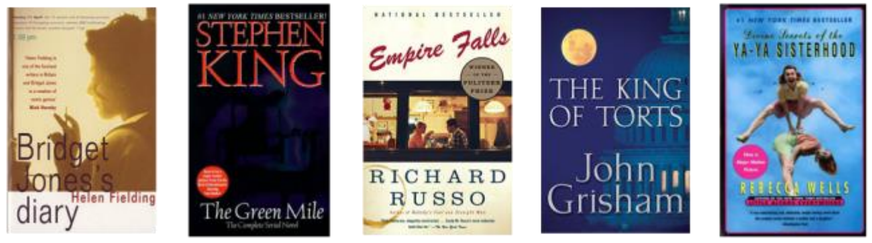

```
content_based_recommender('The Testament')
```

You may also like these books




```
content_based_recommender2('To Kill a Mockingbird')
```

You may also like these books



08 | 결론 및 소감

- 같은 도서를 바탕으로 도서를 추천받아도 추천 시스템에 따라 다른 결과가 나오는 것을 확인할 수 있습니다.

- 추천 시스템에 따라 필요한 데이터가 다르기 때문에 가지고 있는 데이터에 맞는 추천 시스템을 선택할 필요가 있습니다.

- 이번 주제에 대해 공부하면서 추천 시스템의 원리에 대해 자세히 알게 되었고 다양한 추천 시스템을 사용해보는 계기가 되었습니다.

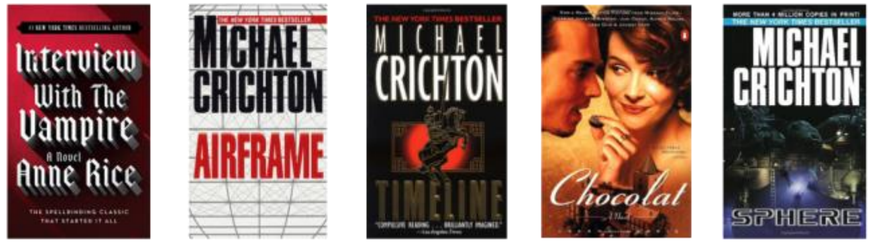
- 사용한 추천 시스템을 바탕으로 다음에는 여러 추천 시스템을 결합한 추천 시스템을 작성해 보고 싶습니다.

```
item_based_recommender('Girl with a Pearl Earring')
```

You may also like these books


```
content_based_recommender('Girl with a Pearl Earring')
```

You may also like these books




```
content_based_recommender2('Girl with a Pearl Earring')
```

You may also like these books

Thank you

enter the contents