아이템 기반 협업 필터링 영화 추천

데이터 셋

- MovieLens 100K Dataset
- url : https://grouplens.org/datasets/movielens/100k/ (https://grouplens.org/datasets/movielens/100k/)
- u.data
  - user_id : 유저 정보
  - item_id : 영화 정보
  - ratings : 평점 정보
  - timestamp : 시간 정보

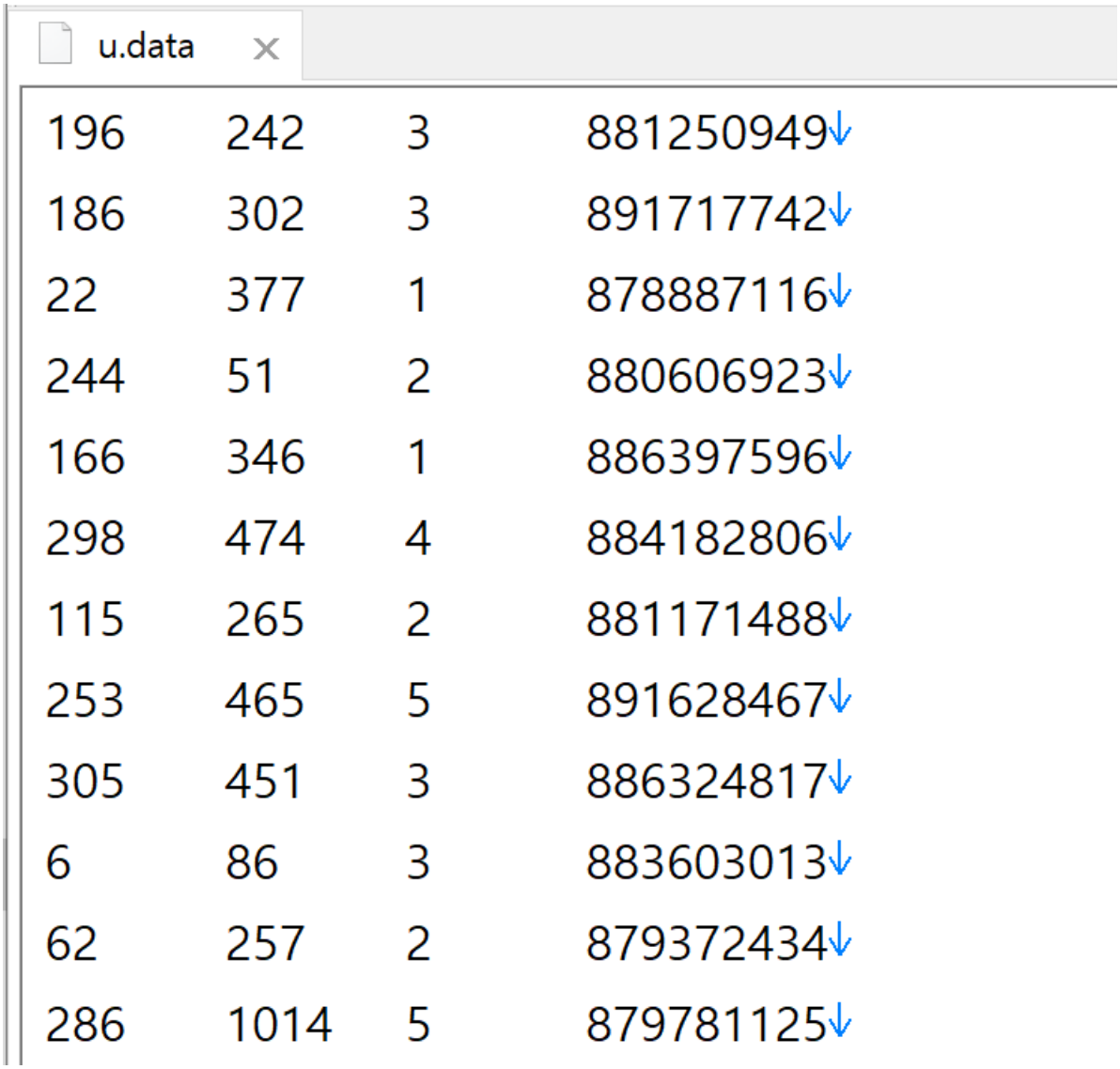
In [6]:

```
import pandas as pd
import numpy as np
import sklearn
from sklearn.decomposition import TruncatedSVD
from IPython.display import display, Image
```

데이터 불러오기

In [7]:

```
display(Image(filename='data_u_data.png'))
```



In [5]:

```
columns = ['user_id', 'item_id', 'rating', 'timestamp']
df = pd.read_csv('../data/ml-100k/u.data', sep='₩t', names=columns)
print(df.shape)
df.head()
```

```
(100000, 4)
```

Out[5]:

	user_id	item_id	rating	timestamp
0	196	242	3	881250949
1	186	302	3	891717742
2	22	377	1	878887116
3	244	51	2	880606923
4	166	346	1	886397596

데이터 불러오기

데이터 정보

- 파일명 : u.u_item
  - item_id : 영화 정보
  - movie title : 영화 제목
  - release date : 출시일
  - video release date : 비디오 출시일
  - IMDb URL : IMDb URL 정보
  - unkonwn, ... : 기타 장르 정보

In [8]:

```
display(Image(filename='data_u_item.png'))
```

In [11]:

```
# 장르 분야
columns = ['item_id', 'movie title', 'release date', 'video release date', 'IMDb URL',
           'unknown', 'Action', 'Adventure', 'Animation', 'Childrens',
           'Comedy', 'Crime', 'Documentary', 'Drama', 'Fantasy',
           'Film-Noir', 'Horror', 'Musical', 'Mystery', 'Romance',
           'Sci-Fi', 'Thriller', 'War', 'Western']

movies = pd.read_csv('../data/ml-100k/u.item', sep='|', names=columns, encoding='latin-1')

movie_names = movies[['item_id', 'movie title']]
c_movies_data = pd.merge(df, movie_names, on='item_id')
print(c_movies_data.shape)
c_movies_data.head()
```

```
(100000, 5)
```

Out[11]:

	user_id	item_id	rating	timestamp	movie title
0	196	242	3	881250949	Kolya (1996)
1	63	242	3	875747190	Kolya (1996)
2	226	242	5	883888671	Kolya (1996)
3	154	242	3	879138235	Kolya (1996)
4	306	242	5	876503793	Kolya (1996)

사용자-아이템 표 만들기

- 우리의 데이터에 대해 pivot를 사용해본다.
- 빈 값은 0으로 채우기

In [12]:

```
rating_crosstab = c_movies_data.pivot_table(values='rating',
                                            index='user_id',
                                            columns='movie title', fill_value=0)
rating_crosstab.head()
```

Out[12]:

movie title	'Til There Was You (1997)	1-900 (1994)	101 Dalmatians (1996)	12 Angry Men (1957)	187 (1997)	2 Days in the Valley (1996)	20,000 Leagues Under the Sea (1954)	2001: A Space Odyssey (1968)	3 Ninjas: High Noon At Mega Mountain (1998)	39 Steps, The (1935)
user_id										
1	0	0	2	5	0	0	3	4	0	0
2	0	0	0	0	0	0	0	0	1	0
3	0	0	0	0	2	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	2	0	0	0	0	4	0	0

5 rows × 1664 columns

아이템 기반 협업 필터링을 위해 행열 바꾸기

In [14]:

```
X = rating_crosstab.T
print(X.shape)
```

(1664, 943)

SVD

- 사이킷런을 활용하여 SVD를 할수 있다.
- truncated SVD를 사용하여 차원 축소

In [15]:

```
SVD = TruncatedSVD(n_components=12, random_state=5)
resultant_matrix = SVD.fit_transform(X)
resultant_matrix.shape
```

Out[15]:

(1664, 12)

- 1664개 행과 잠재변수 12개의 열를 갖는 행렬 생성

Correlation Pearson

- 피어슨 상관계수 , 코사인 유사성과 같은 다양한 유사성 측정 지표를 사용할 수 있다.
- 피어슨 상관계수를 이용하여 상관 행렬을 만들어봄.

In [19]:

```
### correlation matrix
corr_mat = np.corrcoef(resultant_matrix)
print( corr_mat.shape )
corr_mat
```

(1664, 1664)

Out[19]:

```
array([[ 1.        , -0.11573577,  0.51362284, ...,  0.38310045,
         0.20193733,  0.5065142 ],
       [-0.11573577,  1.        ,  0.05820808, ...,  0.15805829,
         0.51795357,  0.27104818],
       [ 0.51362284,  0.05820808,  1.        , ...,  0.76575655,
         0.43824619,  0.19507139],
       ...,
       [ 0.38310045,  0.15805829,  0.76575655, ...,  1.        ,
         0.18043708,  0.12115972],
       [ 0.20193733,  0.51795357,  0.43824619, ...,  0.18043708,
         1.        ,  0.20126072],
       [ 0.5065142 ,  0.27104818,  0.19507139, ...,  0.12115972,
         0.20126072,  1.        ]])
```

유사 영화를 찾기

Similar Movies to Star Wars (1977)

In [21]:

```
rating_crosstab.columns.get_loc("Star Wars (1977)")
```

Out[21]:

1398

In [22]:

```
col_idx = rating_crosstab.columns.get_loc("Star Wars (1977)")
corr_specific = corr_mat[col_idx]    # Star Wars (1977)의 위치 행 획득
print(corr_specific.shape)
```

(1664,)

In [23]:

```
result = pd.DataFrame({'corr_specific':corr_specific, 'Movies': rating_crosstab.columns})
print(result.shape)
result.head()
```

(1664, 2)

Out[23]:

	corr_specific	Movies
0	0.357238	'Til There Was You (1997)
1	0.421507	1-900 (1994)
2	0.593815	101 Dalmatians (1996)
3	0.722361	12 Angry Men (1957)
4	0.325221	187 (1997)

10개의 영화 추천

In [24]:

```
result.sort_values('corr_specific', ascending=False).head(10)
```

Out[24]:

	corr_specific	Movies
1398	1.000000	Star Wars (1977)
1234	0.988052	Return of the Jedi (1983)
1460	0.942655	Terminator 2: Judgment Day (1991)
1523	0.933978	Toy Story (1995)
1461	0.931701	Terminator, The (1984)
1205	0.925185	Raiders of the Lost Ark (1981)
456	0.923562	Empire Strikes Back, The (1980)
570	0.915965	Fugitive, The (1993)
414	0.914299	Die Hard (1988)
44	0.892894	Aliens (1986)

(실습) Godfather, The (1972)에 대한 10개의 영화 추천해 보기

In [27]:

```
col_idx = rating_crosstab.columns.get_loc("Godfather, The (1972)")
corr_specific = corr_mat[col_idx]    # Godfather, The (1972)의 위치 행 획득
print(corr_specific.shape)
```

(1664,)

In [28]:

```
result = pd.DataFrame({'corr_specific':corr_specific, 'Movies': rating_crosstab.columns})
result.sort_values('corr_specific', ascending=False).head(10)
```

Out[28]:

	corr_specific	Movies
612	1.000000	Godfather, The (1972)
613	0.921444	Godfather: Part II, The (1974)
498	0.921420	Fargo (1996)
623	0.900758	GoodFellas (1990)
237	0.865385	Bronx Tale, A (1993)
1398	0.865148	Star Wars (1977)
209	0.864269	Boot, Das (1981)
389	0.857308	Dead Man Walking (1995)
622	0.845558	Good, The Bad and The Ugly, The (1966)
1190	0.842705	Pulp Fiction (1994)

- 우리는 Godfather의 영화를 좋아하는 사람이 있다면 Godfather: Part II, Star Wars (1977)를 볼 것을 제안할 수 있다.
- 역으로 Godfather의 영화를 피하는 사람이라면 Godfather: Part II, Star Wars (1977) 를 피할 것을 제안할 수 있다.

실습해 보기

- Pulp Fiction (1994) 에 대한 유사 영화 10개를 추천해 보자.

History

- 2021-10 ver 01