

파이썬 라이브러리를 활용한 데이터 분석

14장 데이터 분석 예제

2020.07.10금 1h

14장 데이터 분석 예제

MovieLens의 영화 평점 데이터 처리

1h

GroupLens 연구소의 영화 평점 데이터

• 1990년대 말부터 2000년 초

- 약 6천 여명으로부터 4천 여 편의 영화에 대한 백만 개의 영화 평점
- 사용자, 영화, 평점 정보의 3개의 파일 제공
 - `datasets/movielens/users.dat`
 - `datasets/movielens/ratings.dat`
 - `datasets/movielens/movies.dat`



사용자 정보

열

- 사용자: user_id
- 성별: gender
- 나이: age
- 직업: occupation
- 우편번호: zip

```
In [158]: unames = ['user_id', 'gender', 'age', 'occupation', 'zip']
users = pd.read_table('datasets/movielens/users.dat', sep='::',
                    header=None, names=unames, skiprows=1, engine='python')
users
```

Out[158]:

	user_id	gender	age	occupation	zip
0	1	F	1	10	48067
1	2	M	56	16	70072
2	3	M	25	15	55117
3	4	M	45	7	02460
4	5	M	25	20	55455
...
6035	6036	F	25	15	32603
6036	6037	F	45	1	76006
6037	6038	F	56	1	14706
6038	6039	F	45	0	01060
6039	6040	M	25	6	11106

6040 rows × 5 columns

```
In [159]: users.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6040 entries, 0 to 6039
Data columns (total 5 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   user_id     6040 non-null   int64
1   gender      6040 non-null   object
2   age         6040 non-null   int64
3   occupation  6040 non-null   int64
4   zip         6040 non-null   object
dtypes: int64(3), object(2)
memory usage: 236.1+ KB
```

평점 정보

• 열 정보

- 사용자: user_id
- 영화ID: movie_id
- 평점: rating
- 시간: timestamp

```
In [160]: rnames = ['user_id', 'movie_id', 'rating', 'timestamp']
ratings = pd.read_table('datasets/movielens/ratings.dat', sep='::',
                        header=None, names=rnames, skiprows=1, engine='python')
ratings
```

Out[160]:

	user_id	movie_id	rating	timestamp
0	1	1193	5	978300760
1	1	661	3	978302109
2	1	914	3	978301968
3	1	3408	4	978300275
4	1	2355	5	978824291
...
1000204	6040	1091	1	956716541
1000205	6040	1094	5	956704887
1000206	6040	562	5	956704746
1000207	6040	1096	4	956715648
1000208	6040	1097	4	956715569

1000209 rows × 4 columns

```
In [161]: ratings.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000209 entries, 0 to 1000208
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   user_id     1000209 non-null  int64
1   movie_id    1000209 non-null  int64
2   rating      1000209 non-null  int64
3   timestamp   1000209 non-null  int64
dtypes: int64(4)
memory usage: 30.5 MB
```

영화 정보

• 열 정보

- 영화ID: movie_id
- 제목: title
- 장르: genres

```
In [162]: mnames = ['movie_id', 'title', 'genres']
movies = pd.read_table('datasets/movielens/movies.dat', sep=':::',
                      header=None, names=mnames, skiprows=1, engine='python')
movies
```

Out[162]:

	movie_id	title	genres
0	1	Toy Story (1995)	Animation Children's Comedy
1	2	Jumanji (1995)	Adventure Children's Fantasy
2	3	Grumpier Old Men (1995)	Comedy Romance
3	4	Waiting to Exhale (1995)	Comedy Drama
4	5	Father of the Bride Part II (1995)	Comedy
...
3878	3948	Meet the Parents (2000)	Comedy
3879	3949	Requiem for a Dream (2000)	Drama
3880	3950	Tigerland (2000)	Drama
3881	3951	Two Family House (2000)	Drama
3882	3952	Contender, The (2000)	Drama Thriller

3883 rows × 3 columns

```
In [163]: movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3883 entries, 0 to 3882
Data columns (total 3 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   movie_id    3883 non-null   int64
1   title       3883 non-null   object
2   genres      3883 non-null   object
dtypes: int64(1), object(2)
memory usage: 91.1+ KB
```

3개의 DataFrame을 병합

- 공통된 열로 병합: 중복되는 열 이름을 키로 조인
 - 먼저 ratings, users를 합병 후, 다시 결과와 movies를 합병
 - 총 10개의 열이 생김

```
In [170]: data = pd.merge(pd.merge(ratings, users), movies)
          data.head()
```

Out[170]:

	user_id	movie_id	rating	timestamp	gender	age	occupation	zip	title	genres
0	1	1193	5	978300760	F	1	10	48067	One Flew Over the Cuckoo's Nest (1975)	Drama
1	2	1193	5	978298413	M	56	16	70072	One Flew Over the Cuckoo's Nest (1975)	Drama
2	12	1193	4	978220179	M	25	12	32793	One Flew Over the Cuckoo's Nest (1975)	Drama
3	15	1193	4	978199279	M	25	7	22903	One Flew Over the Cuckoo's Nest (1975)	Drama
4	17	1193	5	978158471	M	50	1	95350	One Flew Over the Cuckoo's Nest (1975)	Drama

```
In [171]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1000209 entries, 0 to 1000208
Data columns (total 10 columns):
#   Column      Non-Null Count  Dtype
---  -
0   user_id     1000209 non-null  int64
1   movie_id    1000209 non-null  int64
2   rating      1000209 non-null  int64
3   timestamp   1000209 non-null  int64
4   gender      1000209 non-null  object
5   age         1000209 non-null  int64
6   occupation  1000209 non-null  int64
7   zip         1000209 non-null  object
8   title       1000209 non-null  object
9   genres      1000209 non-null  object
dtypes: int64(6), object(4)
memory usage: 83.9+ MB
```

여러 정보 분석

- 성별에 따른 평균 평점

```
In [173]: mean_ratings = data.pivot_table('rating', index='title',
                                             columns='gender', aggfunc='mean')
mean_ratings[:5]
```

Out[173]:

gender	F	M
title		
\$1,000,000 Duck (1971)	3.375000	2.761905
'Night Mother (1986)	3.388889	3.352941
'Til There Was You (1997)	2.675676	2.733333
'burbs, The (1989)	2.793478	2.962085
...And Justice for All (1979)	3.828571	3.689024

- 영화 제목에 따른 평점 건수

```
In [176]: ratings_by_title = data.groupby('title').size()
ratings_by_title[:5]
```

Out[176]:

title	
\$1,000,000 Duck (1971)	37
'Night Mother (1986)	70
'Til There Was You (1997)	52
'burbs, The (1989)	303
...And Justice for All (1979)	199

dtype: int64

- 평점 건수가 250 개 이상인 영화

- 인덱스만 저장

```
In [177]: active_titles = ratings_by_title.index[ratings_by_title >= 250]
active_titles[:5]
```

Out[177]:

```
Index([''burbs, The (1989)', '10 Things I Hate About You (1999)',
      '101 Dalmatians (1961)', '101 Dalmatians (1996)',
      '12 Angry Men (1957)'],
      dtype='object', name='title')
```


주요 영화 중, 여성에게 높은 평점을 받은 영화 목록

- 평점 건수가 250 개 이상인 영화 제목에 따른 성별 평점 평균

– 목록 active_titles를 인덱스로 사용

- 여성에게 높은 평점을 받은 영화 목록

– 열 F를 내림차순으로 정렬

```
In [178]: # Select rows on the index
mean_ratings = mean_ratings.loc[active_titles]
mean_ratings[:5]
```

Out[178]:

gender	F	M
title		
	'burbs, The (1989)	2.793478 2.962085
	10 Things I Hate About You (1999)	3.646552 3.311966
	101 Dalmatians (1961)	3.791444 3.500000
	101 Dalmatians (1996)	3.240000 2.911215
	12 Angry Men (1957)	4.184397 4.328421

```
In [179]: top_female_ratings = mean_ratings.sort_values(by='F', ascending=False)
top_female_ratings[:10]
```

Out[179]:

gender	F	M
title		
	Close Shave, A (1995)	4.644444 4.473795
	Wrong Trousers, The (1993)	4.588235 4.478261
	Sunset Blvd. (a.k.a. Sunset Boulevard) (1950)	4.572650 4.464589
	Wallace & Gromit: The Best of Aardman Animation (1996)	4.563107 4.385075
	Schindler's List (1993)	4.562602 4.491415
	Shawshank Redemption, The (1994)	4.539075 4.560625
	Grand Day Out, A (1992)	4.537879 4.293255
	To Kill a Mockingbird (1962)	4.536667 4.372611
	Creature Comforts (1990)	4.513889 4.272277
	Usual Suspects, The (1995)	4.513317 4.518248

```
In [177]: active_titles = ratings_by_title.index[ratings_by_title >= 250]
active_titles[:5]
```

Out[177]: Index(['burbs, The (1989)', '10 Things I Hate About You (1999)', '101 Dalmatians (1961)', '101 Dalmatians (1996)', '12 Angry Men (1957)'], dtype='object', name='title')

남녀 간의 호불호가 갈리는 영화

• 열 'diff'

- 평균 평점 차를 저장하는 칼럼 추가

• 성별 선호, 상위 5개

- 여자가 선호
 - 열 diff로 정렬
- 남자가 선호
 - 열 diff로 역정렬

```
In [180]: mean_ratings['diff'] = mean_ratings['M'] - mean_ratings['F']
          mean_ratings.head()
```

Out[180]:

gender		F	M	diff
	title			
	'burbs, The (1989)	2.793478	2.962085	0.168607
	10 Things I Hate About You (1999)	3.646552	3.311966	-0.334586
	101 Dalmatians (1961)	3.791444	3.500000	-0.291444
	101 Dalmatians (1996)	3.240000	2.911215	-0.328785
	12 Angry Men (1957)	4.184397	4.328421	0.144024

```
In [182]: # 여자가 선호하는 영화
          sorted_by_diff = mean_ratings.sort_values(by='diff')
          sorted_by_diff.head()
```

gender		F	M	diff
	title			
	Dirty Dancing (1987)	3.790378	2.959596	-0.830782
	Jumpin' Jack Flash (1986)	3.254717	2.578358	-0.676359
	Grease (1978)	3.975265	3.367041	-0.608224
	Little Women (1994)	3.870588	3.321739	-0.548849
	Steel Magnolias (1989)	3.901734	3.365957	-0.535777

```
In [184]: # Reverse order of rows, take first 10 rows
          sorted_by_diff[::-1][:5]
```

Out[184]:

gender		F	M	diff
	title			
	Good, The Bad and The Ugly, The (1966)	3.494949	4.221300	0.726351
	Kentucky Fried Movie, The (1977)	2.878788	3.555147	0.676359
	Dumb & Dumber (1994)	2.697987	3.336595	0.638608
	Longest Day, The (1962)	3.411765	4.031447	0.619682
	Cable Guy, The (1996)	2.250000	2.863787	0.613787

성별에 관계 없이 극명한 호불호가 있는 영화

• 호불호 측정

- 표준편차인 std() 함수로 계산

```
In [185]: # Standard deviation of rating grouped by title
rating_std_by_title = data.groupby('title')['rating'].std()
rating_std_by_title.head()
```

```
Out[185]: title
$1,000,000 Duck (1971)      1.092563
'Night Mother (1986)      1.118636
'Til There Was You (1997)  1.020159
'burbs, The (1989)         1.107760
...And Justice for All (1979) 0.878110
Name: rating, dtype: float64
```

```
In [187]: # Filter down to active titles
rating_std_by_title = rating_std_by_title.loc[active_titles]
rating_std_by_title.head()
```

```
Out[187]: title
'burbs, The (1989)      1.107760
10 Things I Hate About You (1999) 0.989815
101 Dalmatians (1961)    0.982103
101 Dalmatians (1996)    1.098717
12 Angry Men (1957)     0.812731
Name: rating, dtype: float64
```

```
In [188]: # Order Series by value in descending order
rating_std_by_title.sort_values(ascending=False)[:10]
```

```
Out[188]: title
Dumb & Dumber (1994)      1.321333
Blair Witch Project, The (1999) 1.316368
Natural Born Killers (1994) 1.307198
Tank Girl (1995)         1.277695
Rocky Horror Picture Show, The (1975) 1.260177
Eyes Wide Shut (1999)    1.259624
Evita (1996)             1.253631
Billy Madison (1995)     1.249970
Fear and Loathing in Las Vegas (1998) 1.246408
Bicentennial Man (1999)   1.245533
Name: rating, dtype: float64
```