Thực hành Hệ thống gợi ý

Giải thuật lọc Cộng tác

Mục tiêu:

- Củng cố lý thuyết về phương pháp lọc cộng tác
- Lọc cộng tác dựa trên người dùng (user)

1. Tập dữ liệu ví dụ

Dữ liệu đánh giá
Cho tập dữ liệu gồm 6 người dùng và 6 mục dữ liệu, giá trị đánh giá của người dùng cho mục dữ liệu có giá trị từ 1-10, **giá trị 0 nghĩa là chưa đánh giá**.

	0	1	2	3	4	5
0	3	7	4	9	9	7
1	7	0	5	3	8	8
2	7	5	5	0	8	4
3	5	6	8	5	9	8
4	5	8	8	8	10	9
5	7	7	0	4	7	8

Giả sử chúng ta cần xác định **người dùng 3** sẽ đánh giá **item 4** như thế nào bằng cách sử dụng các phương pháp lọc cộng tác dựa trên người dùng (ô tô màu cam).

Lọc cộng tác dựa trên người dùng

Để xác định người dùng 3 sẽ đánh giá item 4 như thế nào, người dùng 3 trở thành người dùng hiện hành và mục dữ liệu 4 là mục dữ liệu mục tiêu cần tìm giá trị đánh giá.

Trong tập dữ liệu này, ta có các item 1,2,3 là các item cần đánh giá, và các item 0,4,5 là các item chưa thông tin dùng để tính toán độ tương tự

Bước 1: Tìm danh sách các người dùng tương tự, sử dụng chỉ số
sử dụng độ đo Pearson để tính độ tương đồng giữa người dùng 3 và các người dùng còn lại

- P: tập các "item" đánh gía chung giữa các người dùng

$$sim(a,b) = \frac{\sum_{p \in P}(r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P}(r_{a,p} - \bar{r}_a)^2}\sqrt{\sum_{p \in P}(r_{b,p} - \bar{r}_b)^2}}$$

- a,b: users– người dùng
- $r_{b,p}$: đánh giá của người dùng b đối với item p
- r_a, r_b, : đánh giá trung bình của a và b

Bước 2: Dự báo

$$pred(a,p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a,b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a,b)}$$

a. Đọc dữ liệu

```
import pandas as pd
import numpy as np
dt = pd.read_csv("CF_6users.csv", header=None)
dt
```

b. Tính độ tương tự

Xác định tập dữ liệu để tính độ tương quan

```
dt1 = dt.iloc[:,[0,4,5]]
```

```
>>> dt1 = dt.iloc[:,[0,4,5]]
>>> dt1
   0   4   5
0  3   9   7
1  7   8   8
2  7   8   4
3  5   9   8
4  5  10   9
5  7   7   8
```

Tính độ tương quan giữa người dùng 3 và những người dùng còn lại

```
from scipy.stats.stats import pearsonr
for i in range(0,6):
    cor = pearsonr(dt1.ix[i,], dt1.ix[2,])
    print(cor)
```

Parameters:	**x** *: (N,) array_like* Input **y** *: (N,) array_like* Input
Returns:	*(Pearson's correlation coefficient,* 2-tailed p-value)

```
(0.05241424183609589, 0.9666167600124109)
(-0.2773500981126146, 0.8210876249779328)
(1.0, 0.0)
(-0.038461538461538484, 0.9755085832892001)
(-0.09078412990032037, 0.94212534330161)
(-0.9707253433941508, 0.15442095831126684)
```

Biến cor chứa giá trị tương quan Pearson và P-value tương ứng, đoạn code sau sẽ tạo ra 1 biến list_cor để lưu trữ duy nhất độ tương quan Pearson.

```
list_cor = []
for i in range(0, 6):
    cor = pearsonr(dt1.iloc[i, :], dt1.iloc[2, :])
    list_cor.append(cor[0])

print(list_cor)
```

```
>>> print(list_cor)
[0.05241424183609593, -0.2773500981126146, 1.0, -0.03846153846153849, -0.0907841
2990032037, -0.970725343394151]
```

c. Dự báo

$$pred(a,p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a,b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a,b)}$$

Sinh viên tự cài đặt công thức dự báo giá trị đánh giá của người dùng 3 đối với item 4 (item có chỉ số 3 và user có chỉ số 2 trong dataset)

Gợi ý:

Đánh giá trung bình của một người dùng

```
import math

def average(x):
    assert len(x) > 0
    return float(sum(x)) / len(x)
```

Duyệt qua các người dùng có chỉ số 0,1,3,4,5 để tính độ tương tự giữa người dùng 3 và các người dùng này, sau đó tìm ra giá trị dự đoán cho item 4 dựa vào các giá trị tương tự vừa tính được + đánh giá trung bình của từng ngừoi dùng theo công thức

Kết quả dự đoán cho item 4 (chỉ số 3) của user 3 (chỉ số 2 trong dataset)

```
>>> print(pred)
2.740955168005794
>>> print(round(pred))
3.0
```

2. Tập dữ liệu movieLens

- Download dữ liệu từ trang: https://grouplens.org/datasets/movielens/latest/
- Tập dữ liệu nhỏ (ml-latest-small.zip (size: 1 MB)) gồm 100.000 đánh giá bởi 600 người dùng, với 3600 tag của 9000 bộ phim.
- Tập dữ liệu gồm các tập tin: "links.csv", "movies.csv", "ratings.csv" and "tags.csv"

a. Đọc dữ liệu

- Dữ liệu đánh giá: vào biến **"rating_dt"**

```
import numpy as np
import pandas as pd

ratings_dt = pd.read_csv("ratings.csv")
ratings_dt.head()
```

Dữ liệu có dạng:

	userId	movieId	rating	timestamp
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815

- Thông tin bộ phim: vào biến **"movie_names"**

```
movie_names = pd.read_csv("movies.csv")
movie_names.head()
```

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

```
>>>
```

b. Tiền xử lý dữ liệu

```
movie_dt = pd.merge(ratings_dt, movie_names, on='movieId')
movie_dt.head()
```

Kết quả:

	userId	movieId	rating	timestamp	title	genres
0	1	1	4.0	964982703	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	5	1	4.0	847434962	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	7	1	4.5	1106635946	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
3	15	1	2.5	1510577970	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
4	17	1	4.5	1305696483	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

c. Xây dựng ma trận đánh giá

```
ratings_mean_count = pd.DataFrame(movie_dt.groupby('title')['rating'].mean())
```

title	
'Hellboy': The Seeds of Creation (2004)	4.000000
'Round Midnight (1986)	3.500000
'Salem's Lot (2004)	5.000000
'Til There Was You (1997)	4.000000
'Tis the Season for Love (2015)	1.500000
'burbs, The (1989)	3.176471

```
ratings_mean_count['rating_counts'] = pd.DataFrame(movie_dt.groupby('title')['rating'].count())
```

title	rating	rating_counts
'Hellboy': The Seeds of Creation (2004)	4.000000	1
'Round Midnight (1986)	3.500000	2
'Salem's Lot (2004)	5.000000	1
'Til There Was You (1997)	4.000000	2
'Tis the Season for Love (2015)	1.500000	1
'burbs, The (1989)	3.176471	17

```
user_movie_rating = movie_dt.pivot_table(index='userId', columns='title', values='rating')
user_movie_rating.head()
user_movie_rating.iloc[1:7,0:5]
```

```
>>> user_movie_rating.iloc[1:7,0:5]
title   '71 (2014)  'Hellboy': The Seeds of Creation (2004)  'Round Midnight (1986)  'Salem's Lot (2004)  'Til There Was You (1997)
userId
2              NaN                                      NaN                     NaN                  NaN                       NaN
3              NaN                                      NaN                     NaN                  NaN                       NaN
4              NaN                                      NaN                     NaN                  NaN                       NaN
5              NaN                                      NaN                     NaN                  NaN                       NaN
6              NaN                                      NaN                     NaN                  NaN                       NaN
7              NaN                                      NaN                     NaN                  NaN                       NaN
```

https://pandas.pydata.org/pandas-docs/version/0.23.0/generated/pandas.DataFrame.pivot_table.html

d. Tính độ tương tự

Mỗi cột chứa tất cả các xếp hạng của người dùng cho một bộ phim cụ thể. Ví dụ bên dưới giúp tìm tất

cả các xếp hạng của người dùng cho bộ phim "Forrest Gump (1994)" và tìm những bộ phim tương tự như nó.

```
forrest_gump_ratings = user_movie_rating['Forrest Gump (1994)']
forrest_gump_ratings.head()
```

userId	
1	4.0
2	NaN
3	NaN
4	NaN
5	NaN

Bây giờ, hãy truy xuất tất cả các bộ phim tương tự như "Forrest Gump (1994)". Chúng ta có thể tìm thấy mối tương quan giữa xếp hạng của người dùng cho "Forest Gump (1994)" và tất cả các phim khác sử dụng hàm Corrwith () như hiển thị bên dưới:
https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corrwith.html

```
movies_like_forest_gump = user_movie_rating.corrwith(forrest_gump_ratings,method="pearson")

corr_forrest_gump = pd.DataFrame(movies_like_forest_gump, columns=['Correlation'])
corr_forrest_gump.dropna(inplace=True)
corr_forrest_gump.head()
```

```
>>> corr_forrest_gump.head()
                                    Correlation
title
'burbs, The (1989)                     0.197712
(500) Days of Summer (2009)            0.234095
*batteries not included (1987)         0.892710
...And Justice for All (1979)          0.928571
10 Cent Pistol (2015)                 -1.000000
```

```
corr_forrest_gump.sort_values('Correlation', ascending=False).head(10)
```

title	Correlation
Lost & Found (1999)	1.0
Century of the Self, The (2002)	1.0
The 5th Wave (2016)	1.0
Play Time (a.k.a. Playtime) (1967)	1.0
Memories (Memorîzu) (1995)	1.0
Playing God (1997)	1.0
Killers (2010)	1.0
Girl Walks Home Alone at Night, A (2014)	1.0
Tampopo (1985)	1.0
Cercle Rouge, Le (Red Circle, The) (1970)	1.0

Từ đầu ra, có thể thấy rằng những bộ phim có tương quan cao với "Forrest Gump (1994)" không được biết đến nhiều. Điều này cho thấy rằng tương quan một mình không phải là một thước đo tốt cho sự tương đồng bởi vì có thể có một người dùng đã xem '"Forest Gump (1994)" và chỉ có một bộ phim khác và đánh giá cả hai là 5.

Một giải pháp cho vấn đề này là chỉ truy xuất những bộ phim tương quan có ít nhất hơn 50 xếp hạng. Để làm như vậy, sẽ thêm cột rating_counts từ khung dữ liệu rating_mean_count vào khung dữ liệu Corr_forrest_gump.

title	Correlation	rating_counts
'burbs, The (1989)	0.197712	17
(500) Days of Summer (2009)	0.234095	42
*batteries not included (1987)	0.892710	7
...And Justice for All (1979)	0.928571	3
10 Cent Pistol (2015)	-1.000000	2

Có thể thấy rằng bộ phim "..And Justice for All (1979)", có tương quan cao nhất chỉ có ba xếp hạng. Điều này có nghĩa là chỉ có ba người dùng đưa ra xếp hạng tương tự cho "Forest Gump (1994)", " And Justice for All (1979)", Tuy nhiên, chúng ta có thể suy luận rằng một bộ phim không thể được tuyên bố tương tự như một bộ phim khác chỉ dựa trên 3 xếp hạng. Đây là lý do tại sao chúng ta đã thêm cột "rating_counts". Bây giờ chúng ta hãy lọc các bộ phim tương quan với "Forest Gump (1994)", có hơn 50 xếp hạng. Câu lệnh sau đây sẽ làm điều đó:

```
corr_forrest_gump[corr_forrest_gump ['rating_counts']>50].sort_values('Correlation', ascending=False).head()
```

e. Dự báo

Sinh viên tự xây dựng hàm dự báo: nhập tiêu đề của 1 bộ phim trong dataset, số lượt đánh giá, số lượng phim gợi ý (ví dụ Pocahontas (1995), số lượt đánh giá – 50, số lượng phim gợi ý - 10), hàm sẽ đưa ra danh sách 10 bộ phim tương tự với bộ phim nhập vào dựa vào chỉ số tương tự và số lượt đánh giá.