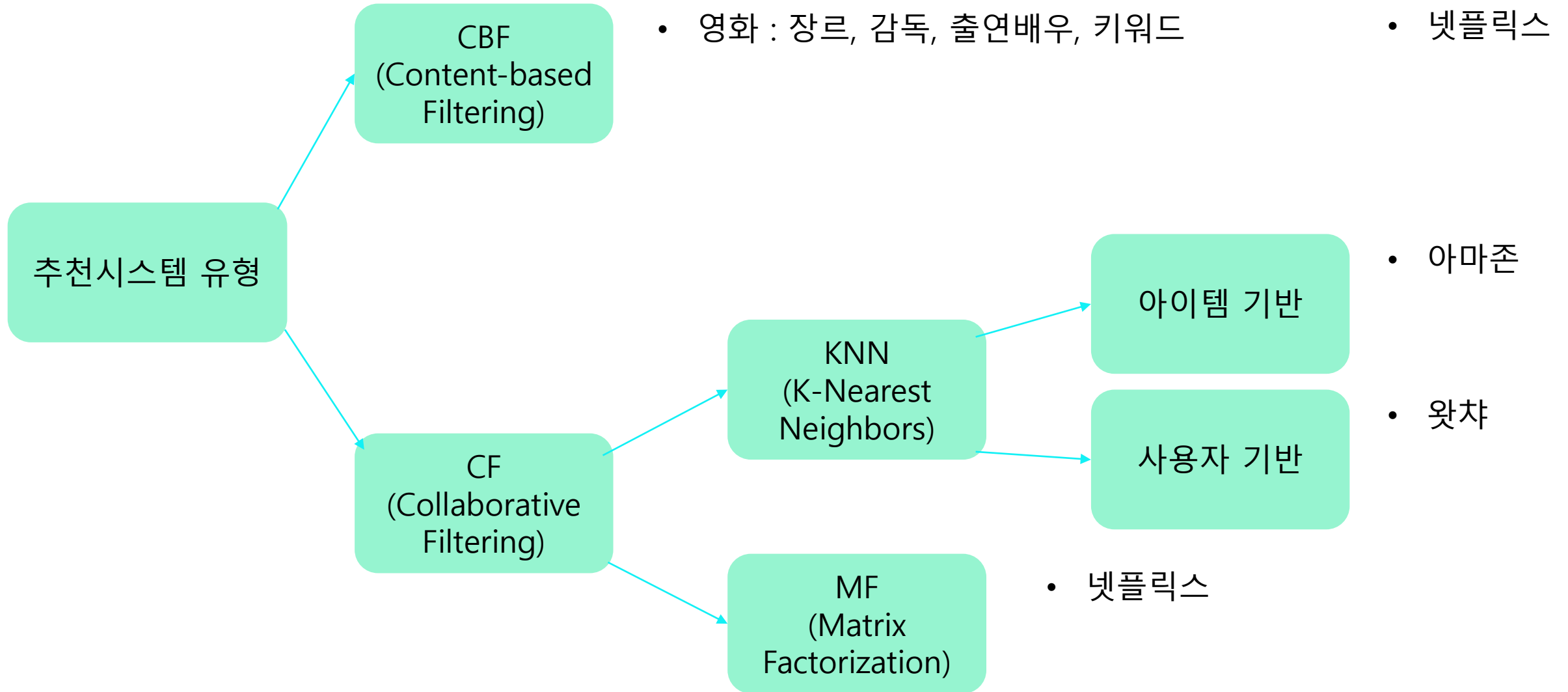


협업 필터링(CF)의 이해

추천시스템 종류



아이템 기반 인접 이웃
협업 필터링(CF)

협업 필터링 - Collaborative Filtering



이번에 개봉하는 그 영화 볼까 말까 궁금하다면 ?

“친구에게 물어봐라 “

단, 취향이 비슷한 친구에게 물어봐야 한다.

협업 필터링의 유형

- 최근접 이웃 기반(Nearest Neighbor)
 - 사용자 기반 (User-user CF)
 - 아이템 기반 (Item-item CF)
- 잠재 요인 기반(Latent Factor)
 - 행렬 분해 기반(Matrix Factorization)



협업 필터링의 특징

- User behavior (item 구매 이력 , 영화 평점 이력) 에만 기반하여 추천 알고리즘 들을 전반적으로 지칭함.
- 상품, 영화 등 사용자가 아직 평가하지 않은 item에 대한 평가 (rating)를 예측 하는 것이 주요 역할.

사용자가 평가하지 않은 Item 을 평가한 item에
기반하여 예측 평가하는 알고리즘.

	Item 1	Item 2	Item 3	Item M
User 1	3		3		✓
User 2	4	2			3
User 3		1	2		2
User 4	1				
.....		3	1		
User N	4	2			5

User1 은 item1 , 3 에 대한 평가 자료만 있다. Item M에
대한 평가를 예측할 수 있는가 ?

협업 필터링을 위한 데이터 세트-사용자 로우-아이템 컬럼

로우 레벨 형태의 사용자-아이템 평점 데이터

User ID	Item ID	Rating
User 1	Item1	3
User 1	Item3	3
User 2	Item1	4
User 2	Item 2	1
User 3	Item 4	5

변환

사용자 로우, 아이템 컬럼으로 구성된
사용자-아이템 평점 데이터

	Item 1	Item 2	Item 3	Item 4
User 1	3		3	
User 2	4	1		
User 3				5

판다스의 `pivot_table()`을 이용하여 쉽게 변환 가능

사용자 기반과 아이템 기반 협업 필터링 이해



사용자 기반 (User-User)

- 특정 사용자와 비슷한 고객들을 기반으로 이 비슷한 고객들이 선호하는 다른 상품을 추천
- 특정 사용자와 비슷한 상품을 구매해온 고객들은 비슷한 고객으로 간주
- 당신과 비슷한 고객들이 다음 상품도 구매했습니다(Customers like you also bought these items)



아이템 기반 (Item-Item)

- 특정 상품과 유사한 좋은 평가를 받은 다른 비슷한 상품을 추천.
- 사용자들로부터 특정 상품과 비슷한 평가를 받은 상품들은 비슷한 상품으로 간주
- 이 상품을 선택한 다른 고객들은 다음 상품도 구매했습니다(customers who bought this item also bought these items).

사용자 기반 협업 필터링

	다크 나이트	인터스텔라	엣지 오브 투모로우	프로메테우스	스타워즈 라스트제다이
사용자 A	5	4	4		
사용자 B	5	3	4	5	3
사용자 C	4	3	3	2	5

상호간
유사도 높음

사용자 A는 사용자 C 보다 사용자 B와 영화 평점 측면에서 유사도가 높음. 따라서 사용자 A에게는 사용자 B가 재미있게 본 '프로메테우스'를 추천

사용자 A의 벡터값 [5, 4, 4, ,]가 사용자 C 벡터값 [4, 3, 3, 2, 5] 보다 사용자 B 벡터값 [5, 3, 4, 5, 3]에 더욱 유사.

사용자 기반 vs 아이템 기반



일반적으로 사용자 기반 보다는 아이템 기반 방식이 더 선호됩니다.
사람간의 특성은 상대적으로 다양한 요소들에 기반합니다.
단순히 동일한 상품을 구입하였다고 유사한 사람이라고 판단하기
어려운 경우가 많기 때문입니다.

협업 필터링을 위한 코사인 유사도

0

1

2

3

1	7	2
1	2	4
0	8	3
2	0	3

기준행

0

1

2

3

1 (0 행 자신의 유사도)	0.68 (0행과 1행의 유사도)	0.99 (0행과 2행의 유사도)	0.3 (0행과 3행의 유사도)
0.68 (1행과 0행의 유사도)	1 (1행 자신의 유사도)	0.72 (1행과 2행의 유사도)	0.85 (1행과 3행의 유사도)
0.99 (2행과 0행의 유사도)	0.72 (2행과 1행의 유사도)	1 (2행 자신의 유사도)	0.29 (2행과 3행의 유사도)
0.3 (3행과 0행의 유사도)	0.85 (3행과 1행의 유사도)	0.29 (3행과 2행의 유사도)	1 (3행 자신의 유사도)

기준행

비교 대상 행

0

1

2

3

기준행

0

1

2

3

cosine_similarities() 적용

사용자 기반과 아이템 기반 간의 데이터 세트 변환

	다크 나이트	인터스텔라	엣지 오브 투모로우	프로메테우스	스타워즈 라스트제다이		사용자 A	사용자 B	사용자 C	사용자 D	사용자 E	
사용자 A	5	4	4			➔	다크 나이트	5	4	5	5	5
사용자 B	5	3	4	5	3		프로메테우스	5	4	4		5
사용자 C	4	3	3	2	5		스타워즈 라스트제다이	4	3	3		4

판다스의 `transpose()`를 이용하여
행과 열 위치의 값을 서로 교환

아이템 기반 협업 필터링의 개인화된 영화 추천

아이템 기반의 협업 필터링에서 개인화된 평점 예측

Weighted Rating Sum

사용자 u 의 아이템 i 에 대한 평점 예측을 사용자 u 가 아이템 i 와 유사한 다른 아이템들(N 개의 다른 아이템)의 합으로 계산하되 아이템 i 와 다른 아이템들간의 유사도를 반영한 합으로 계산.

$$\hat{R}_{u,i} = \sum^N (S_{i,N} * R_{u,N}) / \sum^N (|S_{i,N}|)$$

사용자 u 의 유사한 item 들의 평점과 유사도

	Item j	k	l	m	n
Rating	5	4	1	3	2
	(i, j)	(i, k)	(i, l)	(i, m)	(i, n)
유사도	0.2	0.1	0.4	0.1	0.2



Item j	k	l	m	n
5	4	1	3.	2

*

0.2	(i, j)
0.1	(i, k)
0.4	(i, l)
0.1	(i, m)
0.2	(i, n)

$$5*0.2 + 4*0.1 + 1*0.4 + 3*0.1 + 2*0.2 = \mathbf{2.5}$$

아이템 기반 협업 필터링의 개인화된 영화 추천

아이템 기반의 협업 필터링에서 개인화된 예측 평점

$$\hat{R}_{u,i} = \sum^N (S_{i,N} * R_{u,N}) / \sum^N (|S_{i,N}|)$$

- $\hat{R}_{u,i}$: 사용자 u , 아이템 i 의 개인화된 예측 평점 값
- $S_{i,N}$: 아이템 i 와 가장 유사도가 높은 Top-N개 아이템의 유사도 벡터
- $R_{u,N}$: 사용자 u 의 아이템 i 와 가장 유사도가 높은 Top-N개 아이템에 대한 실제 평점 벡터

아이템 기반 협업 필터링 구현 순서

1. 사용자-아이템 행렬 데이터를 아이템-사용자 행렬 데이터로 변환
2. 아이템간의 코사인 유사도로 아이템 유사도 산출
3. 사용자가 관람(구매) 하지 않은 아이템들 중에서 아이템간 유사도를 반영한 예측 점수 계산
4. 예측 점수가 가장 높은 순으로 아이템 추천

아이템 기반 협업 필터링 실습 - MovieLens 데이터 세트

movielens

Non-commercial, personalized movie recommendations.

[sign up now](#)or [sign in](#)

recommendations

MovieLens helps you find movies you will like. Rate movies to build a custom taste profile, then MovieLens recommends other movies for you to watch.

top picks [see more](#)

based on your ratings. MovieLens recommends these movies:



무비 렌즈 데이터 다운로드

The screenshot shows the Kaggle interface for the 'Movie Lens Small Latest Dataset'. The left sidebar contains navigation links: Home, Compete, Data (selected), Notebooks, Communities, Courses, and More. Below the sidebar, 'Recently Viewed' items are listed. The main content area features a search bar, the dataset title 'Movie Lens Small Latest Dataset' with the subtitle 'Benchmark dataset for recommendation systems', and a 'Download (971 KB)' button highlighted with a red box. Below the download button, there is a section for 'Usability' (5.9) and 'Tags' (business, eyes and vision, recommender systems). The 'Description' section includes a 'Summary' paragraph.

Dataset

Movie Lens Small Latest Dataset

Benchmark dataset for recommendation systems

Shubham Mehta • updated 2 years ago (Version 1)

Data Tasks Notebooks (13) Discussion Activity Metadata **Download (971 KB)** New Notebook

Your Dataset download has started. Show your appreciation with an upvote 15

Usability 5.9 Tags business, eyes and vision, recommender systems

Description

Summary

This dataset (ml-latest-small) describes 5-star rating and free-text tagging activity from [MovieLens](#), a movie recommendation service. It contains 100836 ratings and 3683 tag applications across 9742 movies. These data were created by 610 users between March 29, 1996 and September 24, 2018. This dataset was generated on September 26, 2018.

Users were selected at random for inclusion. All selected users had rated at least 20 movies. No demographic information is included. Each user is represented by an id, and no other information is provided.

- 이름
- links.csv
 - movies.csv
 - ratings.csv
 - README.txt
 - tags.csv


9,000개 영화에 대해 600명의 사용자가 100,000개 평점을 매긴 데이터

Small: 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. Last updated 9/2018.

- [README.html](#)
- [ml-latest-small.zip](#) (size: 1 MB)

In [2]:  pwd


Out[2]: 'C:\\\\Users\\\\runia\\\\Desktop\\\\9장'

In [41]:  `import pandas as pd
import numpy as np`

```
movies = pd.read_csv('./data_movie_lens/movies.csv')
ratings = pd.read_csv('./data_movie_lens/ratings.csv')

print(movies.shape)
print(ratings.shape)
```

(9742, 3)
(100836, 4)

In [13]:  `# 영화 정보 데이터
print(movies.shape)
movies.head()`

(9742, 3)

Out[13]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
1	2	Jumanji (1995)	Adventure Children Fantasy

행렬 분해(MF) 기반의 잠재 요인 협업필터링

잠재 요인 협업 필터링의 개요

잠재 요인 협업 필터링은 사용자-아이템 평점 행렬 속에 숨어 있는 잠재 요인을 추출해 추천 예측을 할 수 있게 하는 기법입니다. 대규모 다차원 행렬을 SVD와 같은 행렬 분해(Matrix Factorization) 기법으로 분해하는 과정에서 잠재 요인을 추출하는데, 이 잠재 요인을 기반으로 사용자-아이템 평점 행렬을 재 구성하면서 추천을 구현합니다.

원본 사용자-아이템 평점 행렬



\approx



Σ



$=$

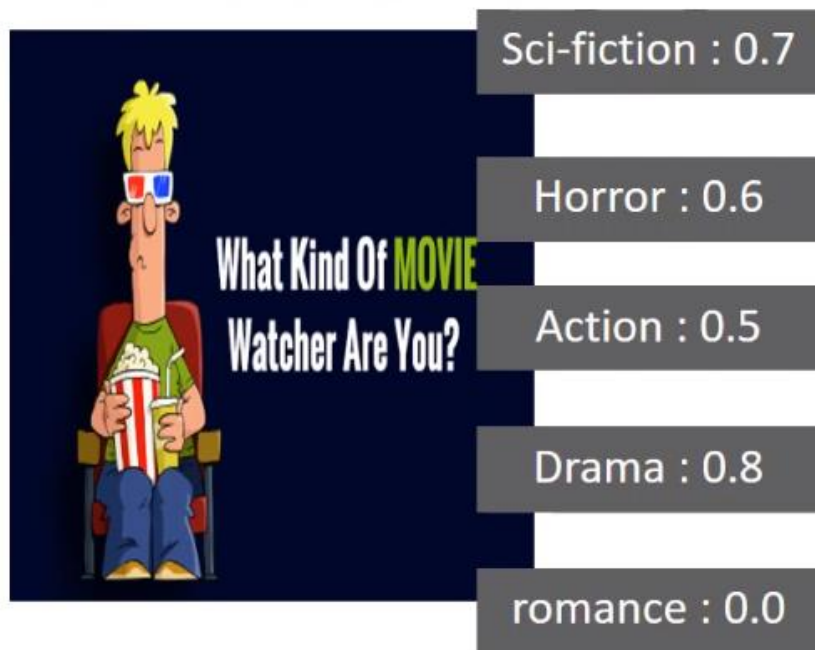
잡음이 제거된 형태로
재구성



잠재 요인 협업 필터링의 이해

잠재 요인 협업 필터링은 사용자-아이템 평점 행렬 속에 숨어 있는 잠재 요인을 추출해 추천 예측을 할 수 있게 하는 기법입니다

사용자 레벨의 잠재요인



아이템 레벨의 잠재요인



영화의 잠재
요인 추출

Item 1 Item 2 Item 3 Item M

User 1	3		3	
User 2	4	2		3
User 3		1	2	2

참재 요인 협업 필터링 - 넷플릭스 추천 엔진 경연



백만 달러의 상금이 걸린 넷플릭스 추천 엔진 경연 대회 우승팀 사진

행렬 분해를 통한 잠재 요인 협업 필터링

잠재 요인 협업 필터링의 행렬 분해

$$\begin{array}{ccccc} \mathbf{R} & \approx & \mathbf{P} & * & \mathbf{Q}^T \\ \text{사용자-아이템} & \text{Decomposed} & \text{사용자-잠재 요인} & & \text{잠재 요인-아이템} \\ \text{평점 행렬} & \text{by} & \text{행렬} & & \text{행렬} \end{array}$$

- 잠재 요인 협업 필터링의 행렬 분해 목표는 희소 행렬 형태의 사용자-아이템 평점 행렬을 밀집(Dense) 행렬 형태의 사용자-잠재 요인 행렬과 잠재 요인-아이템 행렬로 분해 한 뒤 이를 재 결합하여 밀집 행렬 형태의 사용자-아이템 평점 행렬을 생성하여 사용자에게 새로운 아이템을 추천하는 것입니다.

행렬 분해를 통한 잠재 요인 협업 필터링

원본 행렬

사용자 - 아이템 평점 행렬

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	4			2	
User 2		5		3	
User 3			3	4	4
User 4	5	2	1	2	

~

행렬 분해

사용자 - 잠재 요인 행렬

	factor 1	factor 2
User 1	0.94	0.96
User 2	2.14	0.08
User 3	1.93	1.79
User 4	0.58	1.59

잠재 요인 - 아이템 행렬

	Item 1	Item 2	Item 3	Item 4	Item 5
factor 1	1.7	2.3	1.41	1.36	0.41
factor 2	2.49	0.41	0.14	0.75	1.77

*

내적 곱
결과

예측 평점

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	3.98	2.56	1.46	2	2.08
User 2	3.82	4.96	3.02	2.97	1.02
User 3	5	5	2.96	3.97	4.95
User 4	4.95	1.99	1.04	1.99	3.05

행렬 분해를 통한 잠재 요인 협업 필터링

사용자 아이템 평점 행렬 R

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	4	?		2	
User 2		5		3	
User 3			3	4	4
User 4	5	2	1	2	

$R[1, 2]$

?

\approx User 1

사용자별
장르 선호도 행렬 P
 $P[1, :]$

	Action	Romance
User 1	0.94	0.96

*

영화별 장르 요소
행렬 Q의 전치행렬
 $Q.T[:, 2]$

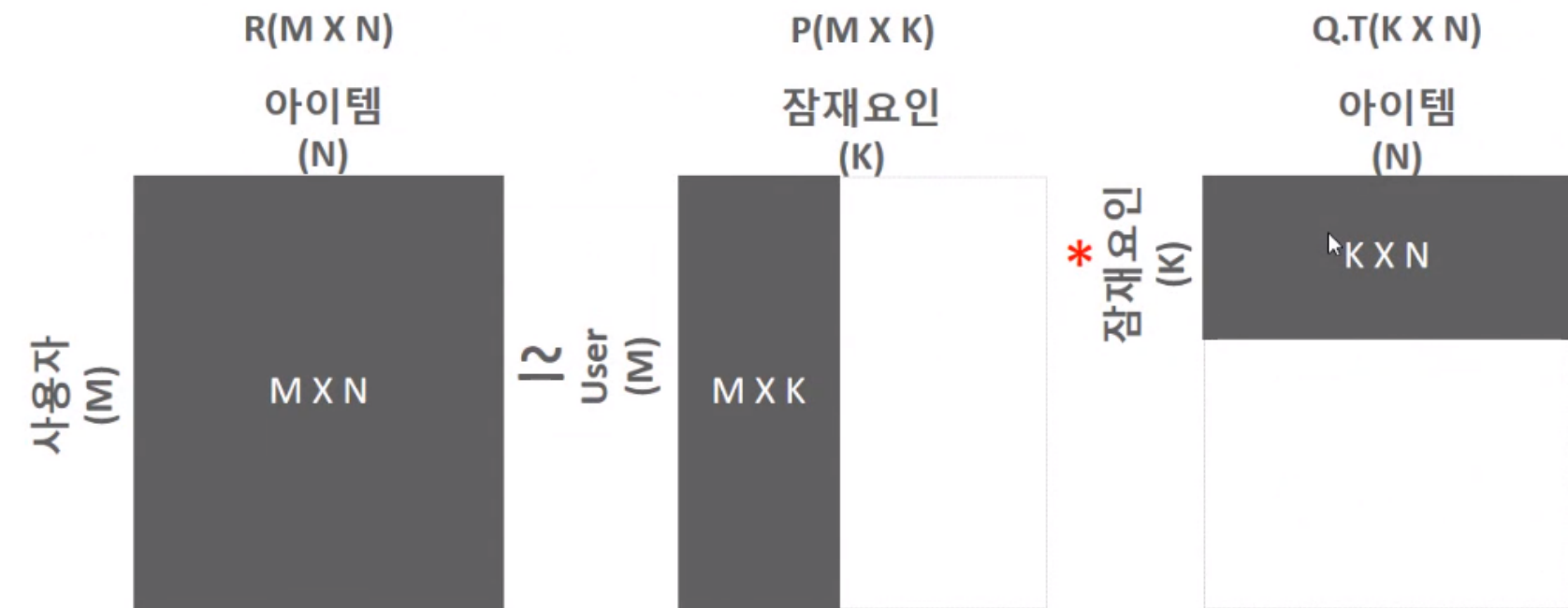
	Item 2
Action	2.3
Romance	0.41

$$0.94 * 2.3 + 0.96 * 0.41$$



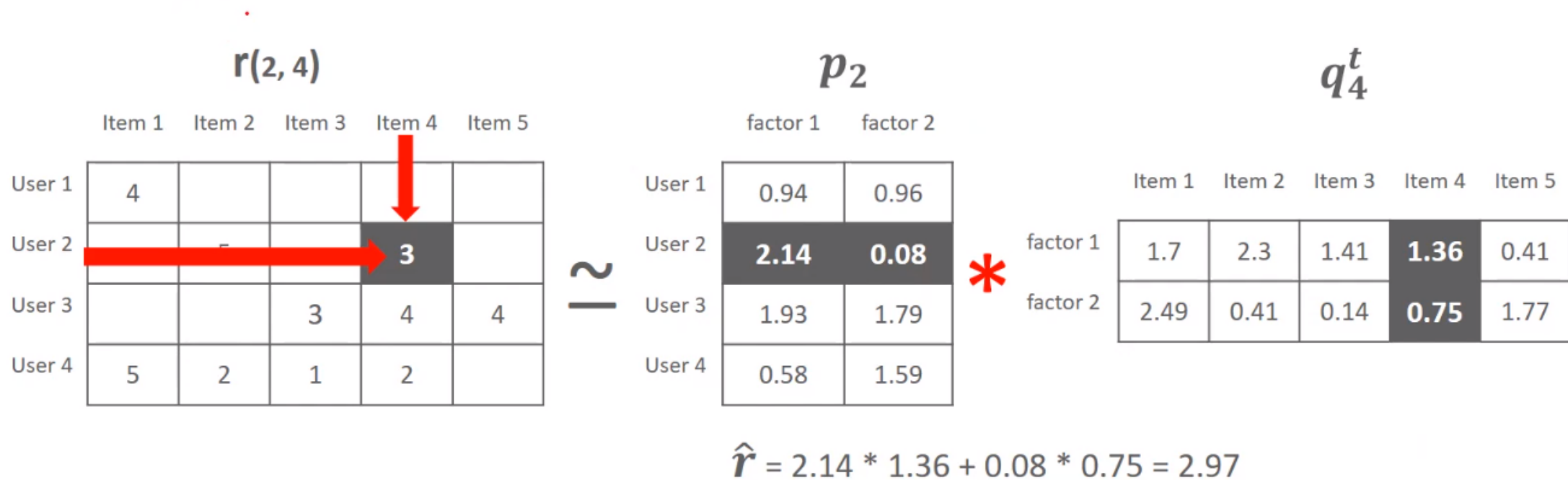
2.56

잠재 요인 기반의 행렬 분해 이해



- M은 총 사용자 수
- N은 총 아이템 수
- K는 잠재 요인의 차원 수
- R은 $M \times N$ 차원의 사용자-아이템 평점 행렬
- P는 사용자와 잠재 요인과의 관계 값을 가지는 $M \times K$ 차원의 사용자-잠재 요인 행렬
- Q는 아이템과 잠재 요인과의 관계 값을 가지는 $N \times K$ 차원의 아이템-잠재 요인 행렬
- Q.T는 Q 매트릭스의 행과 열 값을 교환한 전치 행렬

행렬 분해를 통한 평점 예측



행렬 분해를 통한 평점 예측

$r(2, 3)$

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	4			2	
User 2			?	3	
User 3			3	4	4
User 4	5	2	1	2	

2
1

p_2

	factor 1	factor 2
User 1	0.94	0.96
User 2	2.14	0.08
User 3	1.93	1.79
User 4	0.58	1.59

*

q_3^t

	Item 1	Item 2	Item 3	Item 4	Item 5
factor 1	1.7	2.3	1.41	1.36	0.41
factor 2	2.49	0.41	0.14	0.75	1.77

$$\hat{r}_{(2,3)} = 2.14 * 1.41 + 0.08 * 0.14 = 3.02$$

행렬 분해를 통한 평점 예측

P

	factor 1	factor 2
User 1	0.94	0.96
User 2	2.14	0.08
User 3	1.93	1.79
User 4	0.58	1.59

*

Q.T

	Item 1	Item 2	Item 3	Item 4	Item 5
factor 1	1.7	2.3	1.41	1.36	0.41
factor 2	2.49	0.41	0.14	0.75	1.77



예측 사용자-아이템 평점 행렬

\hat{R}

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	3.98	2.56	1.46	2	2.08
User 2	3.82	4.96	3.02	2.97	1.02
User 3	5	5	2.96	3.97	4.95
User 4	4.95	1.99	1.04	1.99	3.05

사용자-아이템 평점 행렬 분해 이슈

사용자-아이템 평점 행렬

	Item 1	Item 2	Item 3	Item M
User 1	3	?	3	?	?
User 2	4	2		.	3
User 3		1	2		2
User 4	1				
.....		3	1		
User N	4	2			5

그러나 SVD 는 Missing Value 가 없는 행렬에 적용 가능 합니다. 따라서 P 와 Q 행렬을 일반적인 SVD 방식으로는 분해 할 수 없습니다.

P 와 Q 를 모르는데 어떻게 R 을 예측할 수가 있는가 ?



경사 하강법(Gradient Descent)를 이용하여 P 와 Q에 기반한 예측 R 값이 실제 R 값과 가장 최소의 오류를 가질 수 있도록 비용함수 최적화를 통해 P 와 Q를 최적화 유추.

경사 하강법 기반의 행렬 분해

경사 하강법을 이용한 행렬 분해 방법은 P 와 Q 행렬로 계산된 예측 R 행렬 값이 실제 R 행렬 값과 가장 최소의 오류를 가질 수 있도록 반복적인 비용 함수 최적화를 통해 P 와 Q 를 유추해내는 것입니다

경사 하강법 기반의 행렬 분해 순서

1. P 와 Q 를 임의의 값을 가진 행렬로 설정합니다.
2. P 와 Q 의 값을 곱해 예측 R 행렬을 계산하고 예측 R 행렬과 실제 R 행렬에 해당하는 오류 값을 계산합니다.
3. 이 오류 값을 최소화할 수 있도록 P 와 Q 행렬을 적절한 값으로 각각 업데이트합니다.
4. 만족할 만한 오류 값을 가질 때까지 2, 3번 작업을 반복하면서 P 와 Q 값을 업데이트해 근사화합니다

경사 하강법 기반의 행렬 분해 비용 함수

실제 값과 예측값의 오류 최소화와 L2 규제(Regularization)를 고려한 비용 함수식

$$\min \sum (r_{u,i} - p_u q_i^t)^2 + \lambda (\|q_i\|^2 + \|p_u\|^2)$$

실제값과 예측값의 오류 최소화

과적합 개선을 위한 L2 규제

경사 하강법 기반의 행렬 분해 업데이트 식

실제 R 행렬 값과 예측 R 행렬 값의 차이를 최소화하는 방향성을 가지고 P행렬과 Q행렬에 업데이트 값을 반복적으로 수행하면서 최적화된 예측 R 행렬을 구하는 방식이 경사 하강법 기반의 행렬 분해입니다

비용 함수를 최소화하기 위해서
새롭게 업데이트되는 p'_u 와 q'_i

$$p'_u = p_u + \eta (e_{(u,i)} * q_i - \lambda * p_u)$$

$$q'_i = q_i + \eta (e_{(u,i)} * p_u - \lambda * q_i)$$

- p_u : P 행렬의 사용자 u행 벡터
- q_i^t : Q 행렬의 아이템 i행의 전치 벡터(transpose vector)
- $r_{(u,i)}$: R 행렬의 u행, i열에 위치한 값.
- $\hat{r}_{(u,i)}$: $p_u * q_i^t$ 로 계산하며, u행, i열에 위치한 행렬의 예측값
- $e_{(u,i)}$: $r_{(u,i)} - \hat{r}_{(u,i)}$ 의 값으로, u행, i열에 위치한 실제 행렬 값과 예측 행렬 값의 차이 오류
- α : SGD 학습률
- λ : L2 Regularization 계수



경사하강법을 이용한 행렬 분해 이해

```
In [1]: ▶ import numpy as np

# 원본 행렬 R 생성,
# 분해 행렬 P와 Q 초기화, 잠재요인 차원 K는 3 설정.
R = np.array([[4, np.NaN, np.NaN, 2, np.NaN ],
              [np.NaN, 5, np.NaN, 3, 1 ],
              [np.NaN, np.NaN, 3, 4, 4 ],
              [5, 2, 1, 2, np.NaN ]])

R # 4X5 행렬
```

```
Out[1]: array([[ 4., nan, nan,  2., nan],
               [nan,  5., nan,  3.,  1.],
               [nan, nan,  3.,  4.,  4.],
               [ 5.,  2.,  1.,  2., nan]])
```

행렬 R을 행렬 P, Q로 분해할 예정

```
In [2]: ▶ num_users, num_items = R.shape
K=3 # 잠재 요인은 3개

# P와 Q 매트릭스의 크기를 지정하고 정규분포를 가진 random한 값으로 입력합니다.
np.random.seed(1)
P = np.random.normal(scale=1./K, size=(num_users, K))
Q = np.random.normal(scale=1./K, size=(num_items, K))
```