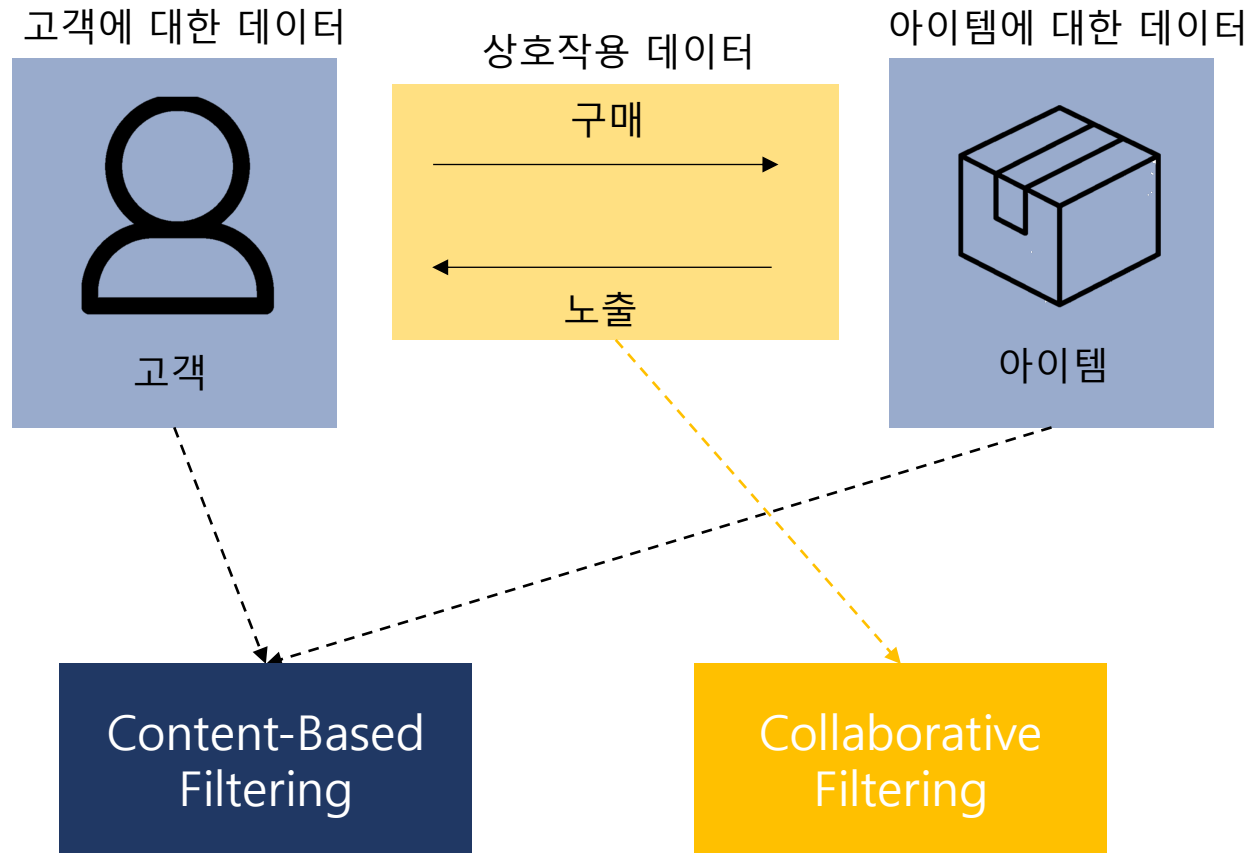


협업 필터링

<Collaborative Filtering>

추천 시스템의 알고리즘

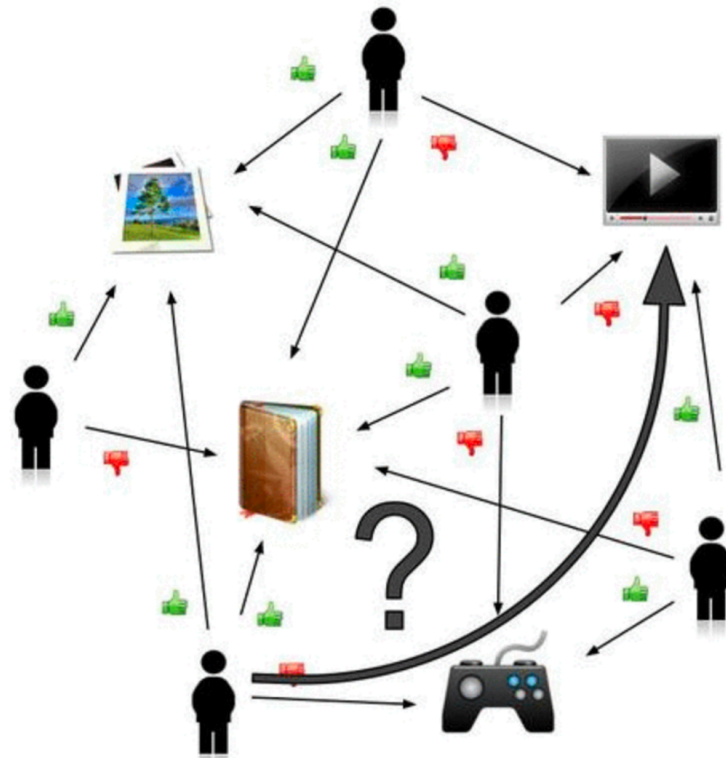
추천 시스템은 크게 콘텐츠 기반 추천과 협업 필터링 기반 추천으로 나뉘어짐



협업 필터링이란?

핵심 아이디어

만약 두 명의 사용자가 **유사한** 관심사를 가지고 있다면,
그들은 미래에도 **유사한** 취향을 가질 것이다.



CF 알고리즘의 데이터, 상호작용 정보

	user_id	movie_id	rating	rated_at
0	1	2	3.5	1112486027
175	2	3	4.0	974820889
236	3	1	4.0	944919407
423	4	6	3.0	840879227
424	4	10	4.0	840878922
451	5	2	3.0	851527569
517	6	1	5.0	858275452
518	6	3	3.0	858275558
519	6	7	5.0	858275558
541	7	3	3.0	1011208463
542	7	7	3.0	1011208220
817	8	1	4.0	833981871
818	8	3	5.0	833981733
819	8	6	3.0	833982631
820	8	10	4.0	833981834
922	10	1	4.0	943497887

고객이 아이템에게
어떠한 액션을 취했는지에 대한 정보

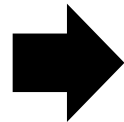
예시)

6번 고객이 7번 영화에 평점 5점을 부여하였음

CF 알고리즘의 데이터, 상호작용 정보

	user_id	movie_id	rating	rated_at
0	1	2	3.5	1112486027
175	2	3	4.0	974820889
236	3	1	4.0	944919407
423	4	6	3.0	840879227
424	4	10	4.0	840878922
451	5	2	3.0	851527569
517	6	1	5.0	858275452
518	6	3	3.0	858275558
519	6	7	5.0	858275558
541	7	3	3.0	1011208463
542	7	7	3.0	1011208220
817	8	1	4.0	833981871
818	8	3	5.0	833981733
819	8	6	3.0	833982631
820	8	10	4.0	833981834
922	10	1	4.0	943497887

고객이 아이템에게
어떠한 액션을 취했는지에 대한 정보

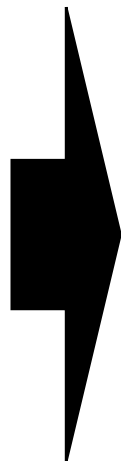


이러한 데이터의 포맷에서는
고객과 아이템 간 특징들을 파악하기 어려움

USER-ITEM Matrix

	user_id	movie_id	rating	rated_at
0	1	2	3.5	1112486027
175	2	3	4.0	974820889
236	3	1	4.0	944919407
423	4	6	3.0	840879227
424	4	10	4.0	840878922
451	5	2	3.0	851527569
517	6	1	5.0	858275452
518	6	3	3.0	858275558
519	6	7	5.0	858275558
541	7	3	3.0	1011208463
542	7	7	3.0	1011208220
817	8	1	4.0	833981871
818	8	3	5.0	833981733
819	8	6	3.0	833982631
820	8	10	4.0	833981834
922	10	1	4.0	943497887

PIVOT

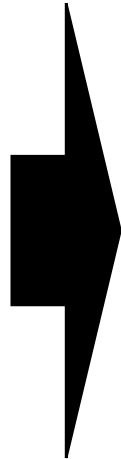


movie_id	1	2	3	6	7	10
user_id						
1	NaN	3.5	NaN	NaN	NaN	NaN
2	NaN	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	3.0	NaN	4.0
5	NaN	3.0	NaN	NaN	NaN	NaN
6	5.0	NaN	3.0	NaN	5.0	NaN
7	NaN	NaN	3.0	NaN	3.0	NaN
8	4.0	NaN	5.0	3.0	NaN	4.0
10	4.0	NaN	NaN	NaN	NaN	NaN

행(user)과 열(item)으로 정렬한 행렬

USER-ITEM Matrix

	user_id	movie_id	rating	rated_at
0	1	2	3.5	1112486027
175	2	3	4.0	974820889
236	3	1	4.0	944919407
423	4	6	3.0	840879227
424	4	10	4.0	840878922
451	5	2	3.0	851527569
517	6	1	5.0	858275452
518	6	3	3.0	858275558
519	6	7	5.0	858275558
541	7	3	3.0	1011208463
542	7	7	3.0	1011208220
817	8	1	4.0	833981871
818	8	3	5.0	833981733
819	8	6	3.0	833982631
820	8	10	4.0	833981834
922	10	1	4.0	943497887



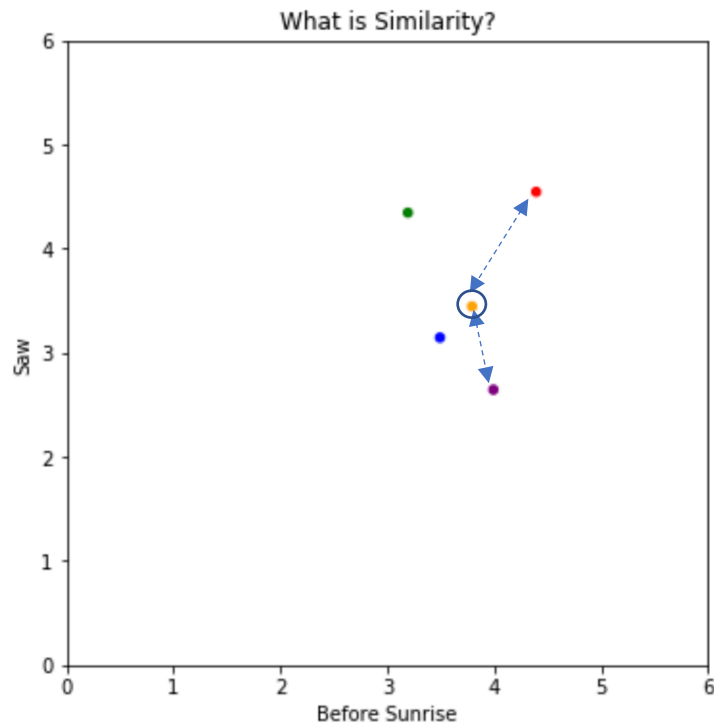
유사도						
movie_id	1	2	3	6	7	10
user_id						
1	NaN	3.5	NaN	NaN	NaN	NaN
2	NaN	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	3.0	NaN	4.0
5	NaN	3.0	NaN	NaN	NaN	NaN
6	5.0	NaN	3.0	NaN	5.0	NaN
7	NaN	NaN	3.0	NaN	3.0	NaN
8	4.0	NaN	5.0	3.0	NaN	4.0
10	4.0	NaN	NaN	NaN	NaN	NaN

유사도를 통해 우리는 어떤 영화끼리 비슷한지를
알 수 있음

협업 필터링 연산의 핵심 : 유사도

핵심 아이디어

만약 두 명의 사용자가 **유사한** 관심사를 가지고 있다면,
그들은 미래에도 **유사한** 취향을 가질 것이다.



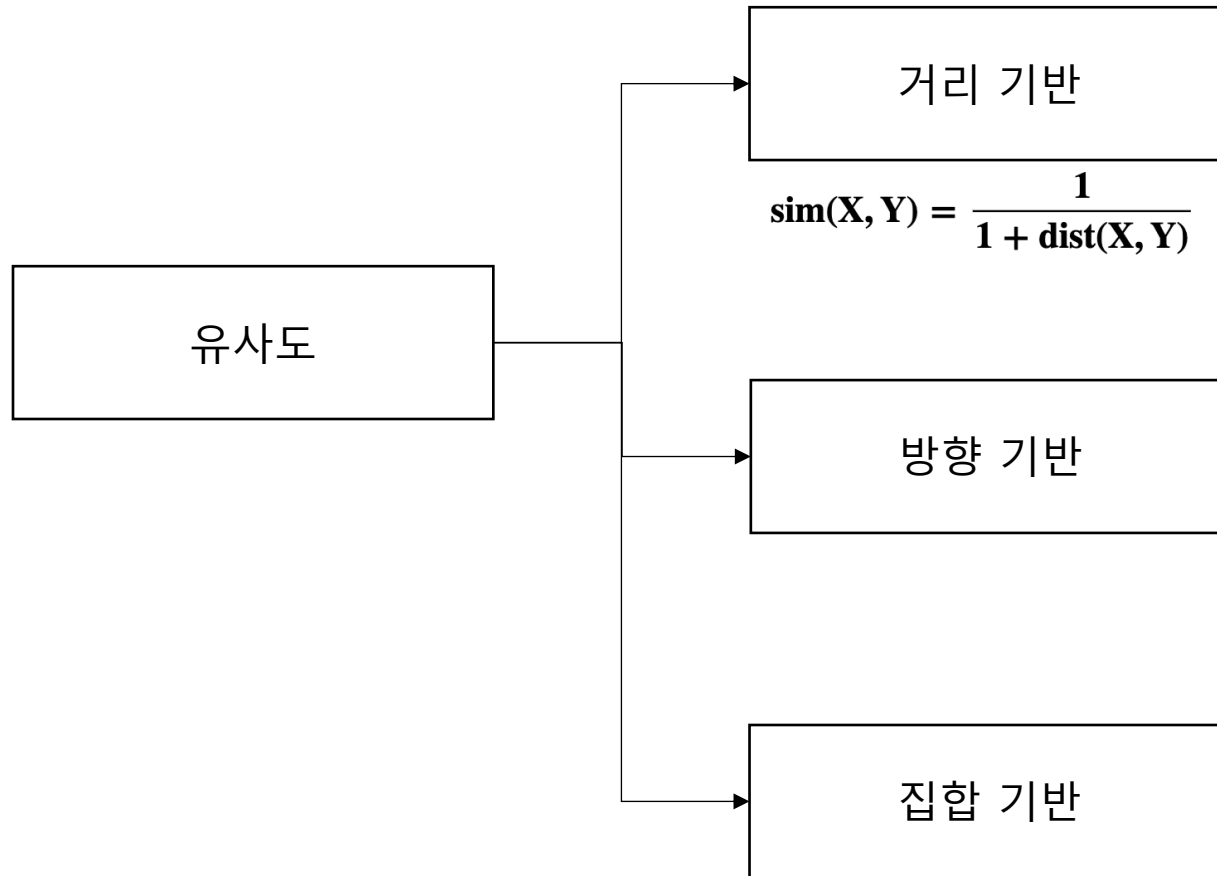
유사도 계산

노란색 유저는

* 빨간색 유저와 유사할까?

* 보라색 유저와 유사할까?

유사도의 다양한 기준들



맨해튼 유사도

$$\text{dist}(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^n (|x_i - y_i|)$$

유클리디안 유사도

$$\text{dist}(\mathbf{X}, \mathbf{Y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

코사인 유사도

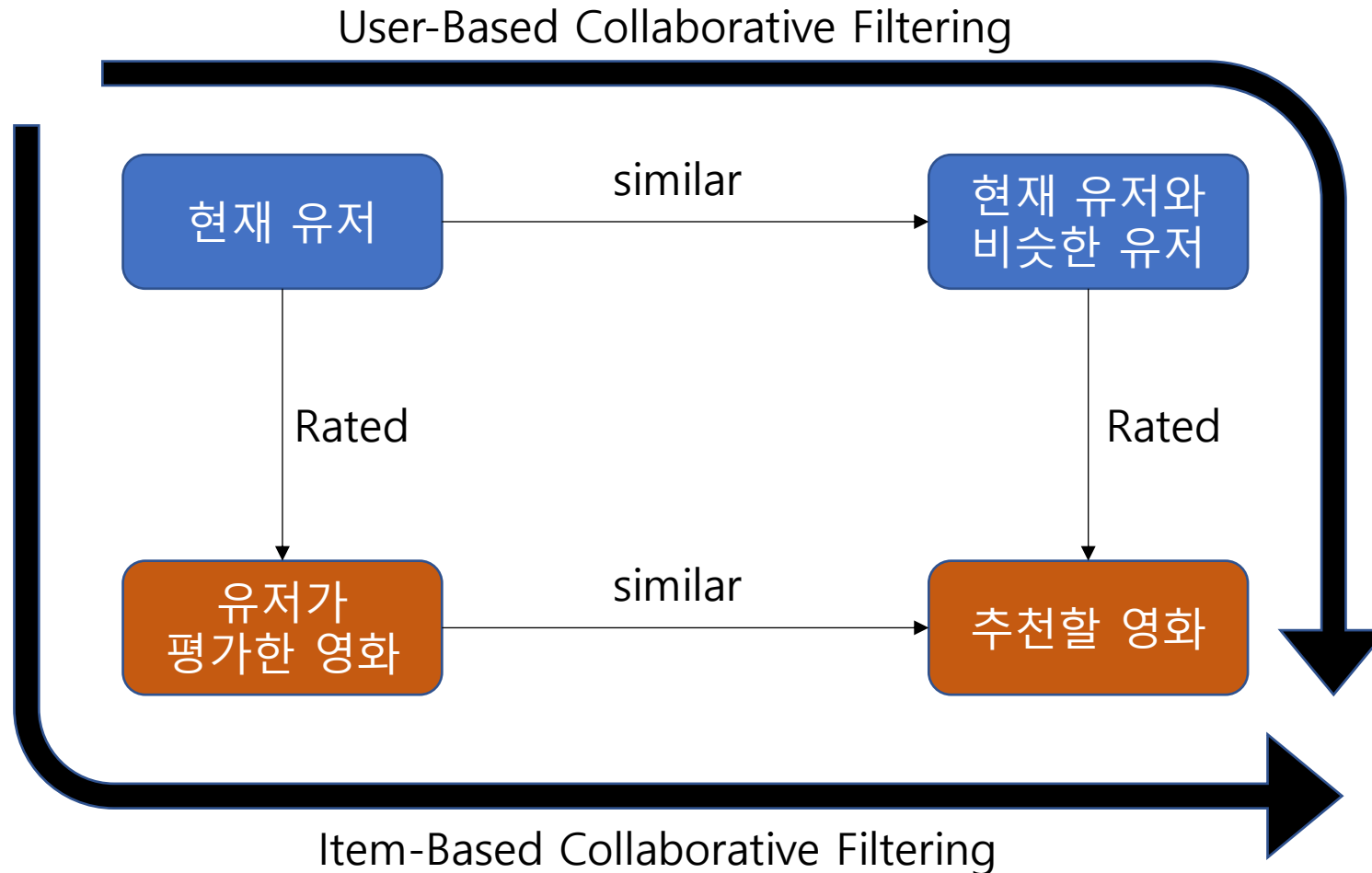
$$\text{sim}(\mathbf{X}, \mathbf{Y}) = \frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}}$$

자카드 유사도

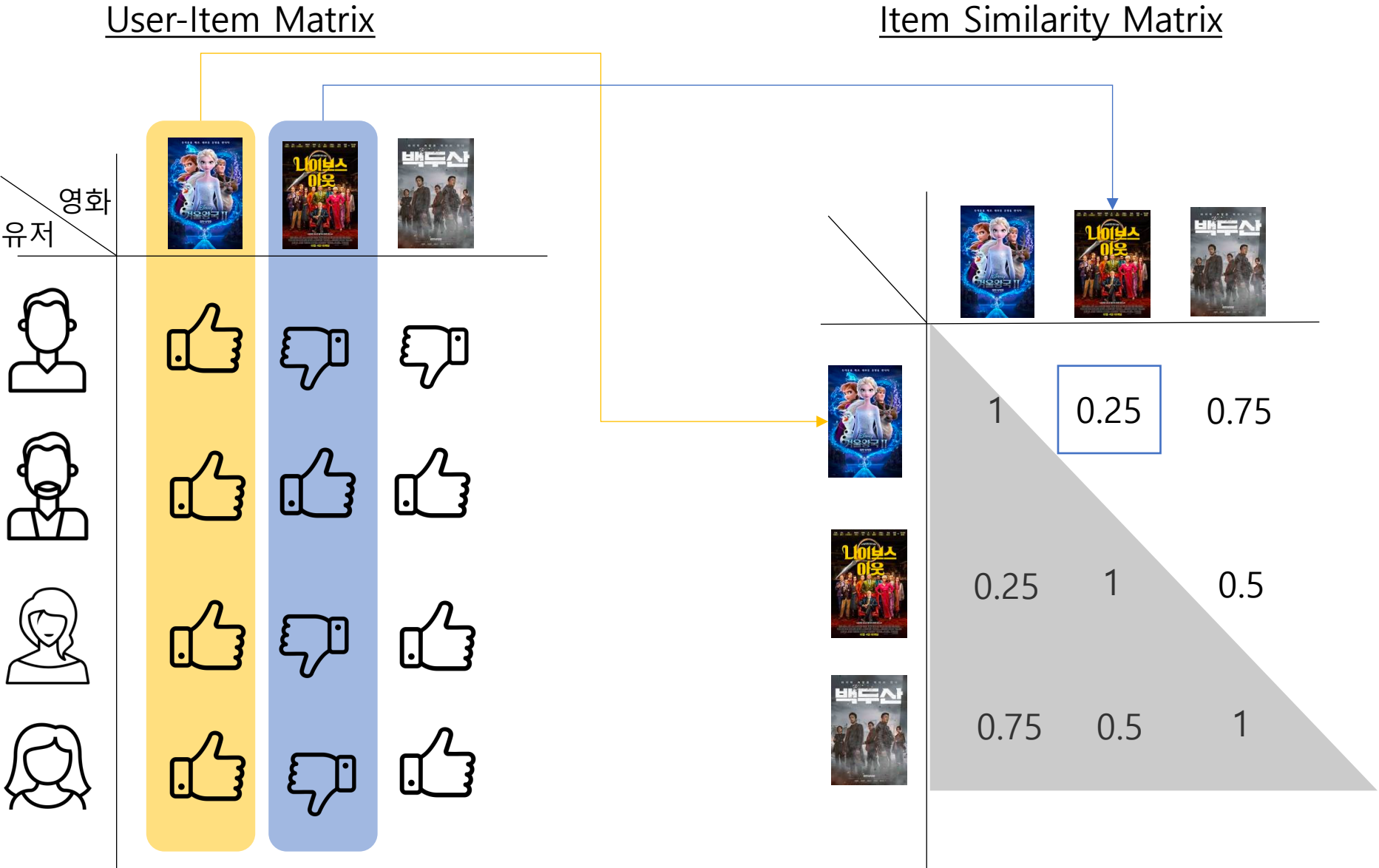
$$\text{sim}(\mathbf{X}, \mathbf{Y}) = \frac{(\mathbf{X} \cap \mathbf{Y})}{(\mathbf{X} \cup \mathbf{Y})}$$

Memory-Based 협업 필터링의 종류

Memory-Based 협업 필터링은 크게
User-Based Collaborative Filtering과 Item-Based Collaborative Filtering으로 나뉘어짐




















Item-Based Collaborative Filtering










User-Based Collaborative Filtering

User-Item Matrix

영화 유저			
			
			
			
			

User Similarity Matrix

영화 유저				
	1	0.33	0.66	0.66
	0.33	1	0.66	0.66
	0.66	0.66	1	1
	0.66	0.66	1	1

Item-User Matrix

영화 \ 유저				
영화				

●
행렬 곱

User-Item Matrix

영화 \ 유저			
유저			

=

Similarity Matrix

	1	0.25	0.75
	0.25	1	0.5
	0.75	0.5	1

Item Based Collaborative Filtering으로 영화 추천하기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

예시) 6번 고객은 어떤 영화를 추천할까?

Item Based Collaborative Filtering으로 영화 추천하기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

예시) 6번 고객은 어떤 영화를 추천할까?

6번 고객이 보지 않은 영화 : **Heat, GoldenEye**

보지 않은 영화 중에서 고객이 선호할만한
영화를 찾자

Item Based Collaborative Filtering으로 영화 추천하기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

예시) 6번 고객은 어떤 영화를 추천할까?

6번 고객이 보지 않은 영화 : **Heat, GoldenEye**

보지 않은 영화 중에서 고객이 선호할만한 영화를 찾자

6번 고객이 보지 않은 영화에 대한 **Rating**을 예측하자

Item Based Collaborative Filtering으로 영화 추천하기

예시) 6번 고객은 어떤 영화를 좋아할까?

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

(1) Item Similiarity Matrix 구성하기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
Toy Story	1.00	0.53	0.33	0.50	0.21
Grumpier Old Men	0.53	1.00	0.46	0.54	0.29
Heat	0.33	0.46	1.00	0.00	0.95
Sabrina	0.50	0.54	0.00	1.00	0.00
GoldenEye	0.21	0.29	0.95	0.00	1.00

Item Based Collaborative Filtering으로 영화 추천하기

예시) 6번 고객은 어떤 영화를 좋아할까?

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

(2) 보지 않은 영화와의 유사도 가져오기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
Toy Story	1.00	0.53	0.33	0.50	0.21
Grumpier Old Men	0.53	1.00	0.46	0.54	0.29
Heat	0.33	0.46	1.00	0.00	0.95
Sabrina	0.50	0.54	0.00	1.00	0.00
GoldenEye	0.21	0.29	0.95	0.00	1.00

가정

유사도가 높은 영화에게 유저가 내린 평점과 비슷할 것

Item Based Collaborative Filtering으로 영화 추천하기

예시) 6번 고객은 어떤 영화를 좋아할까?

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

(2) 보지 않은 영화와의 유사도 가져오기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
Toy Story	1.00	0.53	0.33	0.50	0.21
Grumpier Old Men	0.53	1.00	0.46	0.54	0.29
Heat	0.33	0.46	1.00	0.00	0.95
Sabrina	0.50	0.54	0.00	1.00	0.00
GoldenEye	0.21	0.29	0.95	0.00	1.00

해당 영화와 유사도가 높은 K개를 뽑은 후,
평점의 **평균**을 계산하자
(유사도 기반 가중 평균으로)

Item Based Collaborative Filtering으로 영화 추천하기

예시) 6번 고객은 어떤 영화를 좋아할까?

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

6번 고객의 영화 HEAT에 대한 평점 값

$$\frac{\text{평점}_{\text{toystory}} * \text{유사도}_{\text{toystory}} + \text{평점}_{\text{grumpier}} * \text{유사도}_{\text{grumpier}} + \text{평점}_{\text{sabrina}} * \text{유사도}_{\text{sabrina}}}{\text{유사도}_{\text{toystory}} + \text{유사도}_{\text{grumpier}} + \text{유사도}_{\text{sabrina}}} = \frac{5.0 * 0.33 + 3.0 * 0.46 + 5.0 * 0.0}{0.33 + 0.46 + 0.0} = 3.85$$

(2) 보지 않은 영화와의 유사도 가져오기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
Toy Story	1.00	0.53	0.33	0.50	0.21
Grumpier Old Men	0.53	1.00	0.46	0.54	0.29
Heat	0.33	0.46	1.00	0.00	0.95
Sabrina	0.50	0.54	0.00	1.00	0.00
GoldenEye	0.21	0.29	0.95	0.00	1.00

해당 영화와 유사도가 높은 K개를 뽑은 후,
평점의 **평균**을 계산하자
(유사도 기반 가중 평균으로)

Item Based Collaborative Filtering으로 영화 추천하기

예시) 6번 고객은 어떤 영화를 좋아할까?

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

6번 고객의 영화 GoldenEye에 대한 평점 값

$$\frac{\text{평점}_{\text{toystory}} * \text{유사도}_{\text{toystory}} + \text{평점}_{\text{grumpier}} * \text{유사도}_{\text{grumpier}} + \text{평점}_{\text{sabrina}} * \text{유사도}_{\text{sabrina}}}{\text{유사도}_{\text{toystory}} + \text{유사도}_{\text{grumpier}} + \text{유사도}_{\text{sabrina}}} = \frac{5.0 * 0.21 + 3.0 * 0.29 + 5.0 * 0.0}{0.21 + 0.29 + 0.0} = 3.84$$

(2) 보지 않은 영화와의 유사도 가져오기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
Toy Story	1.00	0.53	0.33	0.50	0.21
Grumpier Old Men	0.53	1.00	0.46	0.54	0.29
Heat	0.33	0.46	1.00	0.00	0.95
Sabrina	0.50	0.54	0.00	1.00	0.00
GoldenEye	0.21	0.29	0.95	0.00	1.00

해당 영화와 유사도가 높은 K개를 뽑은 후,
평점의 **평균**을 계산하자
(유사도 기반 가중 평균으로)

Item Based Collaborative Filtering으로 영화 추천하기

예시) 6번 고객은 어떤 영화를 좋아할까?

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
user_id					
2	NaN	4.0	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN
4	NaN	NaN	3.0	NaN	4.0
6	5.0	3.0	NaN	5.0	NaN
7	NaN	3.0	NaN	3.0	NaN
8	4.0	5.0	3.0	NaN	2.0
10	4.0	NaN	NaN	NaN	NaN

(2) 보지 않은 영화와의 유사도 가져오기

	Toy Story	Grumpier Old Men	Heat	Sabrina	GoldenEye
Toy Story	1.00	0.53	0.33	0.50	0.21
Grumpier Old Men	0.53	1.00	0.46	0.54	0.29
Heat	0.33	0.46	1.00	0.00	0.95
Sabrina	0.50	0.54	0.00	1.00	0.00
GoldenEye	0.21	0.29	0.95	0.00	1.00

해당 영화와 유사도가 높은 K개를 뽑은 후,
평점의 **평균**을 계산하자
(유사도 기반 가중 평균으로)

6번 고객의 영화 Heat에 대한 평점 값 = 3.85

6번 고객의 영화 GoldenEye에 대한 평점 값 = 3.84

6번 고객에게는 영화 Heat를 추천

Memory Based의 한계

User의 수와 Item의 수가 늘어난다면?

movie_id	1	2	3	4	5	6	7	8	9	10	...	111921	112138	112290	112556	112852	116797	117511	117590	118696	125916
user_id																					
1	NaN	3.5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN	4.0	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	5.0	NaN	3.0	NaN	NaN	NaN	5.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7	NaN	NaN	3.0	NaN	NaN	NaN	3.0	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
8	4.0	NaN	5.0	NaN	NaN	3.0	NaN	NaN	NaN	4.0	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
10	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
11	4.5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2.5	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

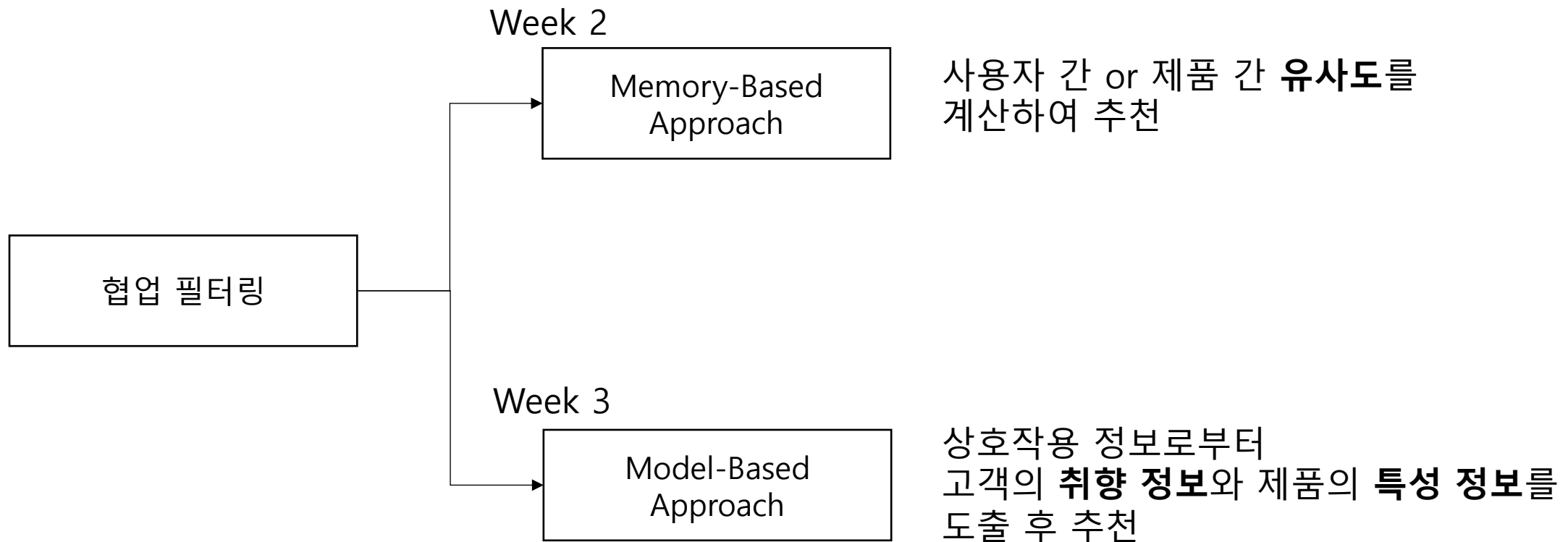
보지 않은 영화에 대해 모두 가중 평균을 계산해야 함



너무 많은 연산이 필요...

오늘의 주제 : 모델 기반 협업 필터링









Memory-Based CF의 문제를 해결하기 위한 모델 기반 협업 필터링



오늘의 주제 : 모델 기반 협업 필터링

모델 기반 협업 필터링 핵심 아이디어

User – Item Matrix를 인수분해(Factorization)하자

음악 유저				
		4.5	2.0	
	4.0		3.5	
		5.0		2.0
		3.5	4.0	1.0

이 행렬을 크게 두가지 인수, **고객에 대한 행렬**과 **아이템에 대한 행렬**로 분해하자

행렬곱 (Inner Product)

행렬곱

→

2	3
3	2
1	4
3	4
4	3

●

3	4	2
1	2	4

=

(5, 2) (2, 3)

행렬곱 (Inner Product)

행렬곱

→

2	3
3	2
1	4
3	4
4	3

●

3	4	2
1	2	4

=

9	14	16
11	16	14
7	12	18
13	20	22
15	22	20

(5, 2) (2, 3) (5, 3)

행렬곱 (Inner Product)

행렬곱

		가	나	다
A	2	3		
B	3	2		
C	1	4		
D	3	4		
E	4	3		

(5, 2)

●

	가	나	다
3	4	2	
1	2	4	

(2, 3)

=

	가	나	다
A	9	14	16
B	11	16	14
C	7	12	18
D	13	20	22
E	15	22	20

(5, 3)

행렬곱 (Inner Product)

행렬곱

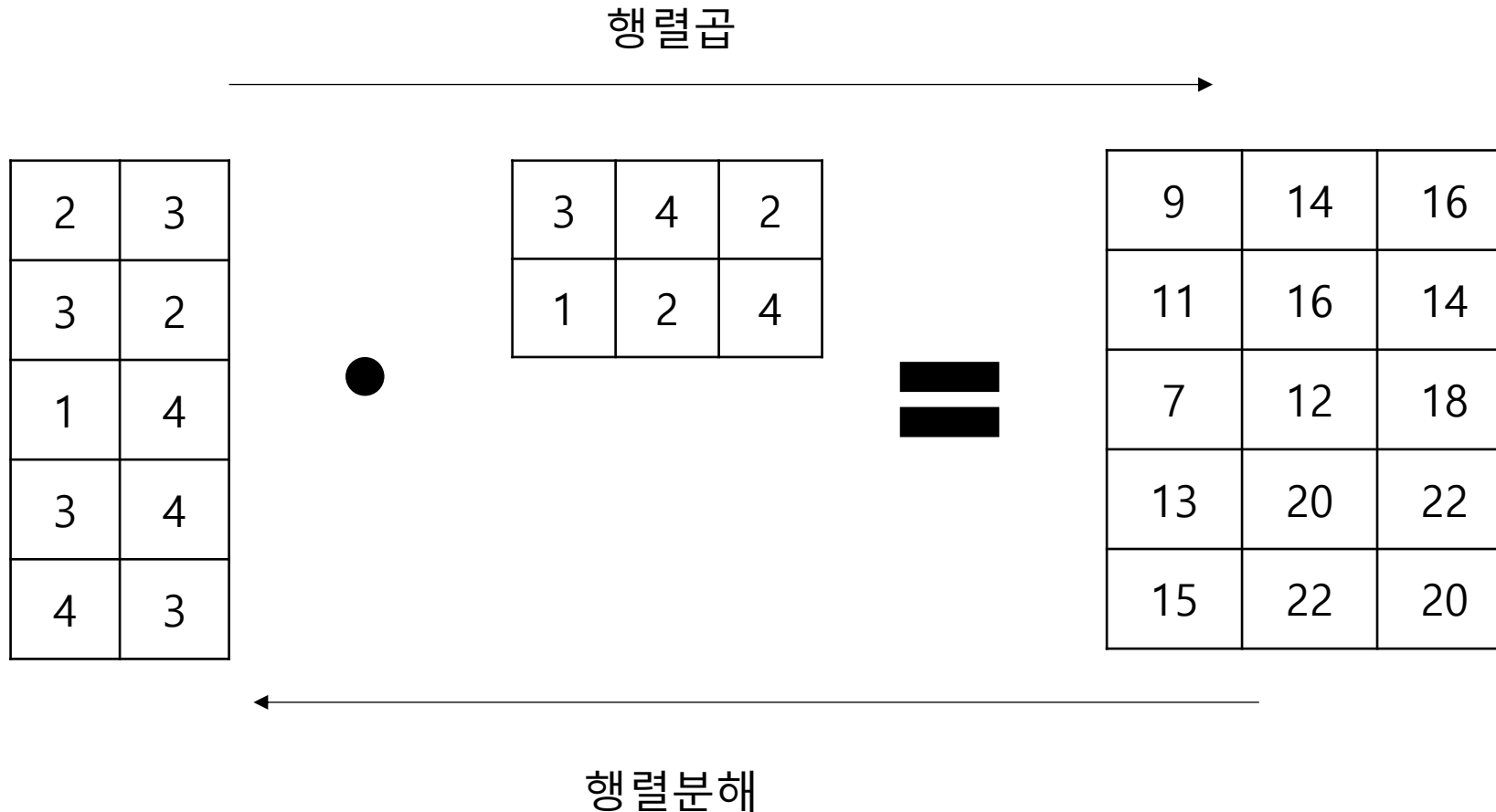
		가	나	다		가	나	다	
A	2	3		3	4	2	9	14	16
B	3	2		1	2	4	11	16	14
C	1	4					7	12	18
D	3	4					13	20	22
E	4	3					15	22	20

(5, 2) (2, 3) (5, 3)

행렬 원소 하나는 두 인수의 조합으로 구성

행렬분해 (Matrix Decomposition)

행렬분해는 행렬곱의 반대로, 하나의 행렬을 두 개의 인수행렬로 나누는 것



행렬분해 (Matrix Decomposition)

행렬분해는 행렬곱의 반대로, 하나의 행렬을 두 개의 인수행렬로 나누는 것

←

행렬분해

2	3
3	2
1	4
3	4
4	3

●

3	4	2
1	2	4

=

9	14	16
11	16	14
7	12	18
13	20	22
15	22	20

(5, 2) (2, 3) (5, 3)

└──────────────────────────┘

Factor의 크기

오늘의 주제 : 모델 기반 협업 필터링

우리의 목표 :

유저-아이템 행렬을 유저의 행렬과 아이템의 행렬로 **분해**하는 것

User-Artist 행렬

음악					
유저					
		4.5	2.0		
	4.0		3.5		
		5.0		2.0	
		3.5	4.0	1.0	

≈

$Embedding_{user}$

유저의 취향 행렬

	?	?
	?	?
	?	?
	?	?

$Embedding_{artist}$

아티스트의 특성 행렬









	?	?
	?	?
	?	?
	?	?

오늘의 주제 : 모델 기반 협업 필터링

오늘 배우는 BPR

유저-아이템 행렬을 어떻게 유저의 행렬과 아이템의 행렬로 분해하는 가에 대한 방법





User-Artist 행렬

음악				
유저				
		4.5	2.0	
	4.0		3.5	
		5.0		2.0
		3.5	4.0	1.0

≈

$Embedding_{user}$

유저의 취향 행렬

	1.2	1.5
	1.4	0.9
	1.5	1.0
	1.2	0.8









$Embedding_{artist}$

아티스트의 특성 행렬

	1.5	1.9
	2.4	1.0
	1.0	2.4
	0.8	0.4

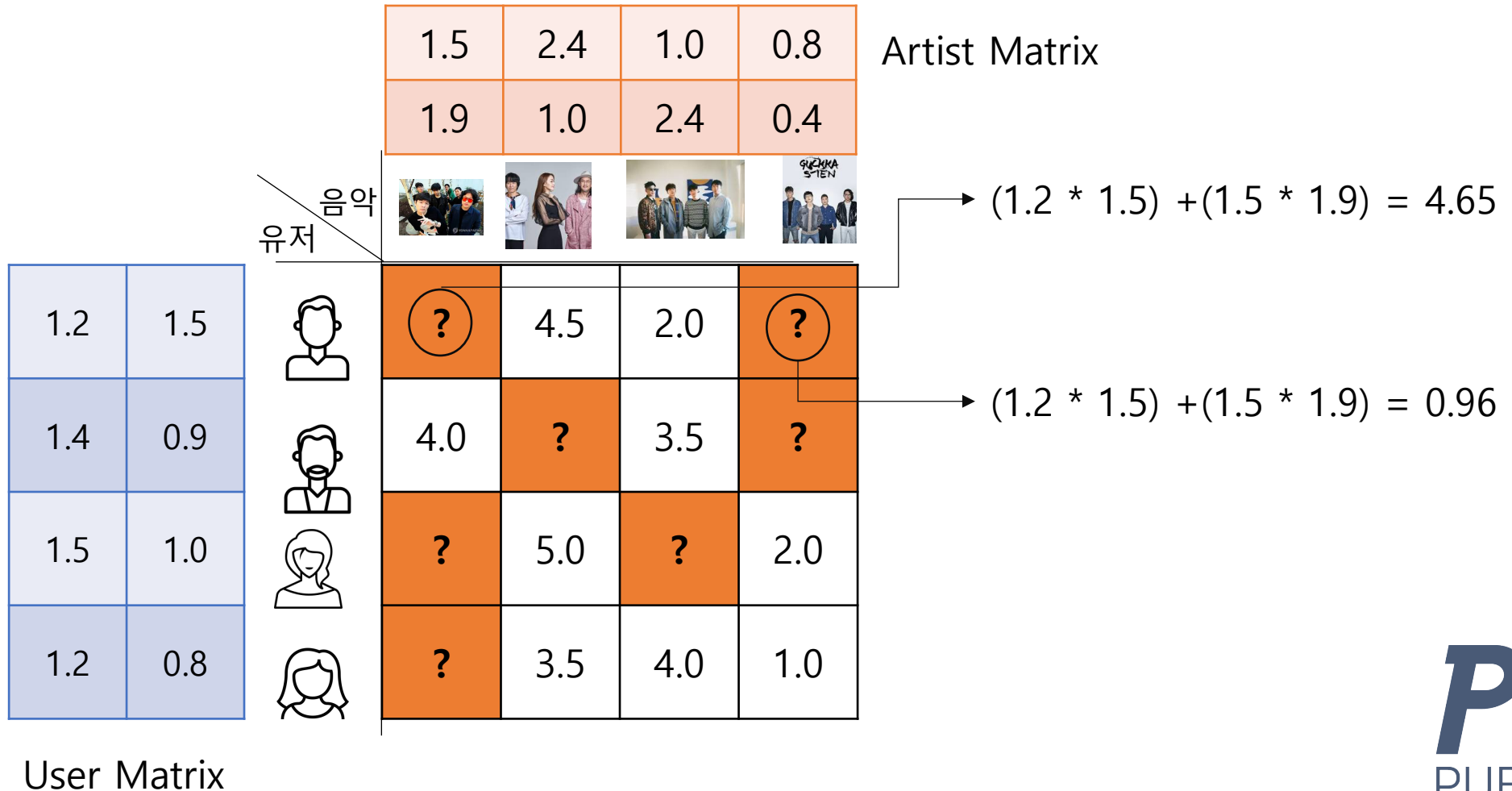
임베딩을 통해 우리가 할 수 있는 것들

1. 고객이 듣지 않은 음악에 대한 선호도 예측

		Artist Matrix							
		1.5	2.4	1.0	0.8				
		1.9	1.0	2.4	0.4				
유저	음악								
		?	4.5	2.0	?				
		4.0	?	3.5	?				
		?	5.0	?	2.0				
		?	3.5	4.0	1.0				
User Matrix		1.2	1.5	1.4	0.9	1.5	1.0	1.2	0.8

임베딩을 통해 우리가 할 수 있는 것들

1. 고객이 듣지 않은 음악에 대한 선호도 예측







임베딩을 통해 우리가 할 수 있는 것들

2. 유사 유저 및 유사 아티스트 찾기

$Embedding_{user}$

유저의 취향 행렬

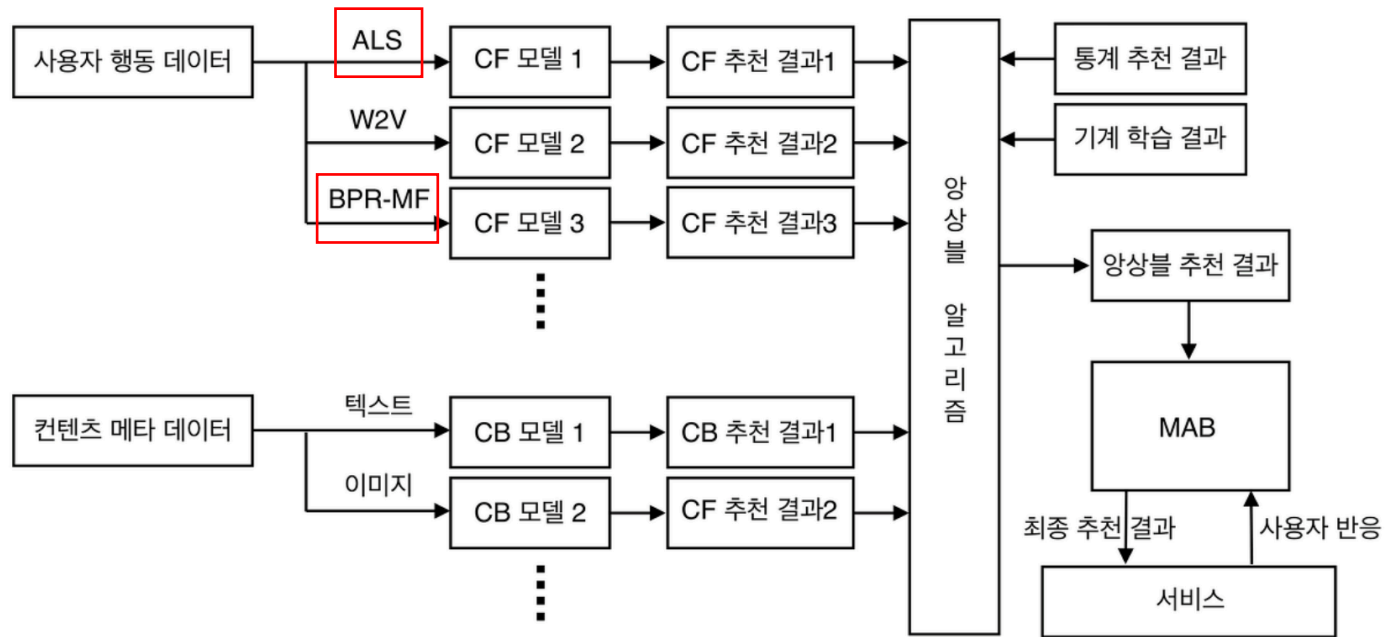
	1.2	1.5
	1.4	0.9
	1.5	1.0
	1.2	0.8

$Embedding_{artist}$

아티스트의 특성 행렬

	1.5	1.9
	2.4	1.0
	1.0	2.4
	0.8	0.4

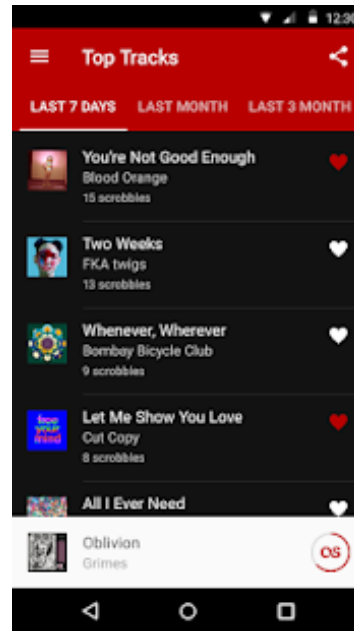
카카오 추천시스템



오늘 다룰 데이터 : Last.FM

	user_id	artist_name	plays
8593694	70361	emiliana torrini	50
2558827	344136	t.a.t.u.	550
10856902	152173	alanis morissette	267
16279545	6746	mc yogi	24
2583290	306930	amy winehouse	127
4628228	89236	decapitated	42
15679098	323248	bachelors of science	61
8468985	300260	great lake swimmers	312
10647188	61247	flying lotus	59
6104662	204729	the rasmus	3295
15960449	257961	gore beyond necropsy	90
14295862	118693	punto omega	178
13767597	27337	carlos do carmo	27
8688251	194746	anthrax	143
7924894	28586	rilo kiley	36
4664396	27235	30 seconds to mars	59
6311667	253294	this heat	25
2237086	82138	george michael	221
10564881	11862	converge	20
3120123	292899	the tough alliance	12

실제 음악 감상 서비스의 데이터



어떤 유저(user_id)가 어떤 아티스트(artist_name)의 음악을 몇 번이나(plays) 들었는지 에 대한 정보

Last.FM (데이터의 특징)

	user_id	artist_name	plays
8593694	70361	emiliana torrini	50
2558827	344136	t.a.t.u.	550
10856902	152173	alanis morissette	267
16279545	6746	mc yogi	24
2583290	306930	amy winehouse	127
4628228	89236	decapitated	42
15679098	323248	bachelors of science	61
8468985	300260	great lake swimmers	312
10647188	61247	flying lotus	59
6104662	204729	the rasmus	3295
15960449	257961	gore beyond necropsy	90
14295862	118693	punto omega	178
13767597	27337	carlos do carmo	27
8688251	194746	anthrax	143
7924894	28586	rilo kiley	36
4664396	27235	30 seconds to mars	59
6311667	253294	this heat	25
2237086	82138	george michael	221
10564881	11862	converge	20
3120123	292899	the tough alliance	12

어떤 유저(user_id)가 어떤 아티스트(artist_name)의 음악을 몇 번이나(plays) 들었는지에 대한 정보

고객의 취향이 간접적으로 반영된 정보

암묵 데이터 (Implicit)

명시적 데이터 VS 암묵적 데이터

지난 시간까지의 데이터 : 영화 별점 데이터

user_id		title	rating
9516636	43422	Pan's Labyrinth (Laberinto del fauno, El)	1.5
8244048	38771	Space Jam	3.0
15900924	100652	Home Alone 2: Lost in New York	3.5
11652080	42727	For a Few Dollars More (Per qualche dollaro in...	5.0
11004179	105943	Kramer vs. Kramer	1.0
19978640	80920	Messenger of Death	3.0









명시적 데이터 (별점 데이터)

고객 정보 →

제품 정보 →

MODEL

→ 별점 정보

음악 유저	   			
		4.5	2.0	
	4.0		3.5	
		5.0		2.0
		3.5	4.0	1.0

명시적 데이터 VS 암묵적 데이터

지난 시간까지의 데이터 : 영화 별점 데이터

user_id		title	rating
9516636	43422	Pan's Labyrinth (Laberinto del fauno, El)	1.5
8244048	38771	Space Jam	3.0
15900924	100652	Home Alone 2: Lost in New York	3.5
11652080	42727	For a Few Dollars More (Per qualche dollaro in...	5.0
11004179	105943	Kramer vs. Kramer	1.0
19978640	80920	Messenger of Death	3.0









명시적 데이터 (별점 데이터)

고객 정보 →

제품 정보 →

MODEL

→ 별점 정보

음악 유저				
				
		4.5	2.0	
	4.0		3.5	
		5.0		2.0
		3.5	4.0	1.0

행렬 분해 시,
User Item Matrix의 값 = 별점데이터

명시적 데이터 VS 암묵적 데이터

이번 시간의 데이터 : 음악 행동 데이터

	user_id	artist_name	plays
8593694	70361	emiliana torrini	50
2558827	344136	t.a.t.u.	550
10856902	152173	alanis morissette	267
16279545	6746	mc yogi	24
2583290	306930	amy winehouse	127

암묵 데이터 (행동 데이터)

고객 정보 →

제품 정보 →

MODEL

→ 재생 횟수 ?

명시적 데이터 VS 암묵적 데이터

이번 시간의 데이터 : 음악 행동 데이터

	user_id	artist_name	plays
8593694	70361	emiliana torrini	50
2558827	344136	t.a.t.u.	550
10856902	152173	alanis morissette	267
16279545	6746	mc yogi	24
2583290	306930	amy winehouse	127









암묵 데이터 (행동 데이터)

고객 정보 →

제품 정보 →

MODEL

→ 재생 횟수 ?

음악 유저				
		120	32	
	67		95	
		102		56
		35	40	25

재생 횟수가 높다고 해서 꼭 더 선호하는 것이 아님!

- 해당 아티스트의 곡이 대중적일 수도 있고,
- 어쩌다 보니 플레이리스트에 곡이 올라왔을 수도 있고
- 다른 목적으로 곡을 반복적으로 틀어놨을 수도 있고
- 과거에는 음악을 많이 들었는데 최근에는 적게 들었을 수도 있고

.....

명시적 데이터 VS 암묵적 데이터

이번 시간의 데이터 : 음악 행동 데이터

	user_id	artist_name	plays
8593694	70361	emiliana torrini	50
2558827	344136	t.a.t.u.	550
10856902	152173	alanis morissette	267
16279545	6746	mc yogi	24
2583290	306930	amy winehouse	127

암묵 데이터 (행동 데이터)






고객 정보 →

제품 정보 →

MODEL

재생 횟수 ?

재생 유무

음악					
유저					
		✓	✓		
	✓		✓		
		✓			✓
		✓	✓	✓	

암묵적 데이터에서는 유무에 대한 정보가 가장 중요

명시적 데이터 VS 암묵적 데이터

이번 시간의 데이터 : 음악 행동 데이터

	user_id	artist_name	plays
8593694	70361	emiliana torrini	50
2558827	344136	t.a.t.u.	550
10856902	152173	alanis morissette	267
16279545	6746	mc yogi	24
2583290	306930	amy winehouse	127


암묵 데이터 (행동 데이터)

고객 정보 →

제품 정보 →

MODEL

→ 재생 유무

음악 유저				
	0	1	1	0
	1	0	1	0
	0	1	0	1
	0	1	1	1

암묵적 데이터에서는 유무에 대한 정보가 가장 중요
0과 1로 표현

Matrix Factorization의 어려운 점

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \approx \begin{bmatrix} ? & ? \\ ? & ? \\ ? & ? \\ ? & ? \end{bmatrix} \cdot \begin{bmatrix} ? & ? \\ ? & ? \\ ? & ? \\ ? & ? \end{bmatrix}^T$$

특이값 분해 (Singular Vector Decomposition)

$$\begin{bmatrix} \text{gray} & \text{gray} & \text{gray} & \text{gray} \\ \text{gray} & \text{gray} & \text{gray} & \text{gray} \\ \text{gray} & \text{gray} & \text{gray} & \text{gray} \\ \text{gray} & \text{gray} & \text{gray} & \text{gray} \end{bmatrix} = \begin{bmatrix} \text{teal} & \text{green} & \text{blue} & \text{green} \\ \text{teal} & \text{green} & \text{blue} & \text{green} \\ \text{teal} & \text{green} & \text{blue} & \text{green} \\ \text{teal} & \text{green} & \text{blue} & \text{green} \end{bmatrix} \begin{bmatrix} \text{orange} & 0 & 0 \\ 0 & \text{yellow} & 0 \\ 0 & 0 & \text{yellow} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \text{light blue} & \text{light blue} & \text{light blue} \\ \text{purple} & \text{purple} & \text{purple} \\ \text{pink} & \text{pink} & \text{pink} \\ \text{pink} & \text{pink} & \text{pink} \end{bmatrix}$$

$$\mathbf{M}_{m \times n} = \mathbf{U}_{m \times m} \mathbf{\Sigma}_{m \times n} \mathbf{V}^*_{n \times n}$$

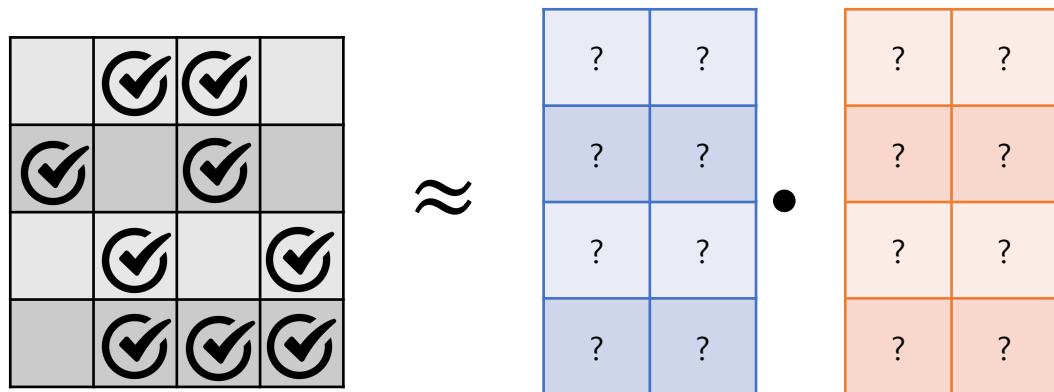
특이값 분해(SVD)

행렬 분해하는 대표적인 수학적 방법

But

데이터가 Sparse한 상황에서는 잘 동작하지 않음

Matrix Factorization (For 암묵 데이터)



ALS(Alternating Least Square)

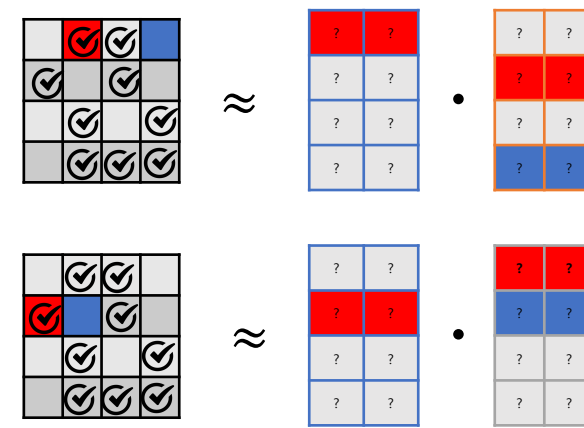
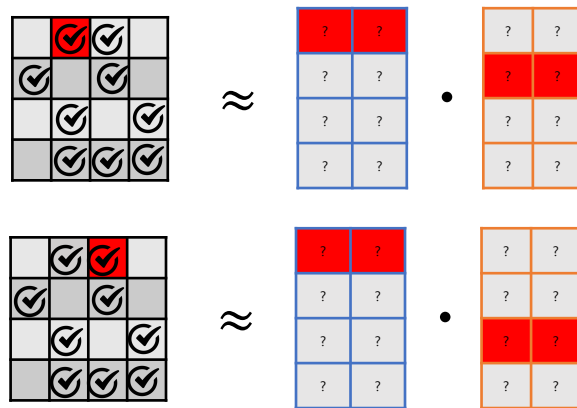
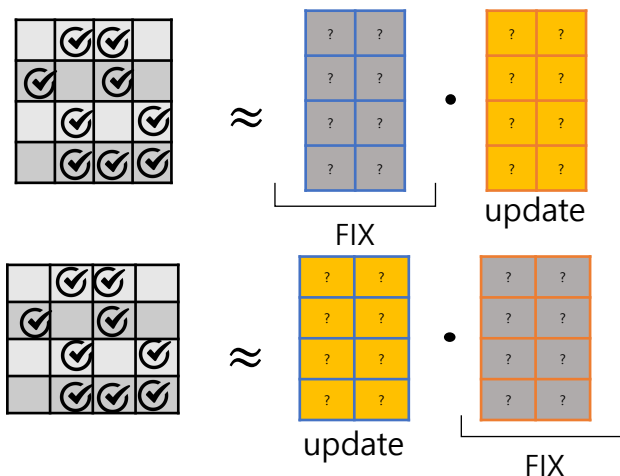
SGD(Stochastic Gradient Descent)

BPR(Bayesian Personalized Ranking)

전체 데이터로
교대로 행렬을 업데이트하기

각 케이스 별로
행렬 값을 업데이트하기

케이스 쌍 별로
행렬 값을 업데이트하기



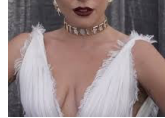

























BPR 알고리즘

(1) 유저 - 아티스트 플레이 행렬을 구성하고, 유저 행렬과 아티스트 행렬을 초기화(무작위)한다.

Artist Matrix

User Matrix

User Matrix			음악								
			유저								
											
											
											
											

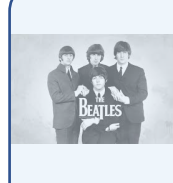
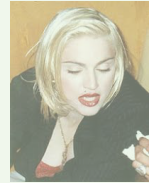
BPR 알고리즘

(2) 사용자가 들어본 아티스트와 들어보지 않은 아티스트를 쌍으로 뽑는다

Artist Matrix

User Matrix

음악
유저



BPR의 핵심 가정

고객이 구매, 감상한 작품에 대한 선호도가
고객이 구매, 감상하지 않은 작품에 대한 선호도보다 높다





BPR 알고리즘

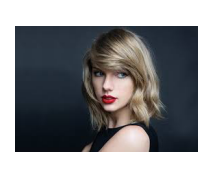
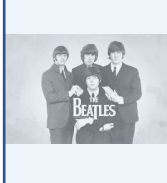
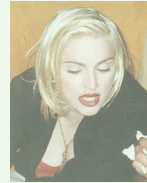
(3) 들어본 아티스트와 유저의 유사도와 들어보지 않은 아티스트와 유저의 유사도를 계산한다

Artist Matrix

User Matrix

음악
유저



BPR의 핵심 가정

고객이 구매, 감상한 작품에 대한 선호도가
고객이 구매, 감상하지 않은 작품에 대한 선호도보다 높다

$$\text{Similarity} \left(\text{User Icon}, \text{Madonna Image} \right) \gg \text{Similarity} \left(\text{User Icon}, \text{The Beatles Image} \right)$$

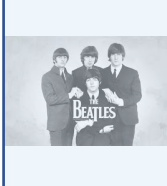
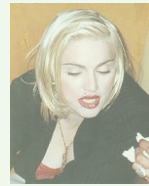
BPR 알고리즘

(4) 그 차이가 증가하는 방향으로 임베딩 행렬의 값을 변경한다.

Artist Matrix

User Matrix

음악
유저



차이가 증가하는 방향(기울기) 값을 구해,
기울기 값에 따라 임베딩 벡터의 값을 변경

$$U := U - \alpha \frac{\partial L}{\partial U}$$

$$I_{pos} := I - \alpha \frac{\partial L}{\partial I_{pos}}$$

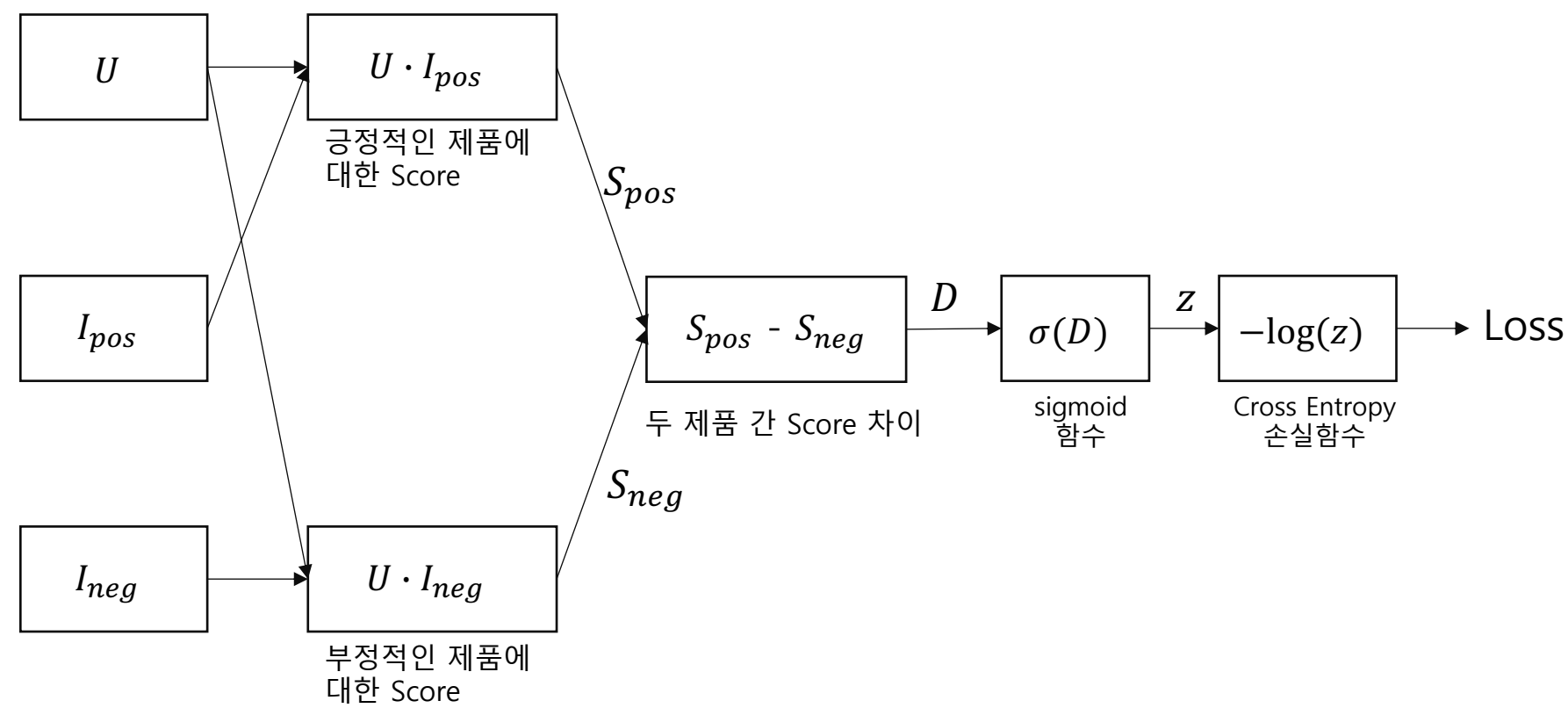
$$I_{neg} := I - \alpha \frac{\partial L}{\partial I_{neg}}$$

기울기를 통해 값을 찾아가는 방식 : 경사하강법(Gradient Descent)

L: 손실함수

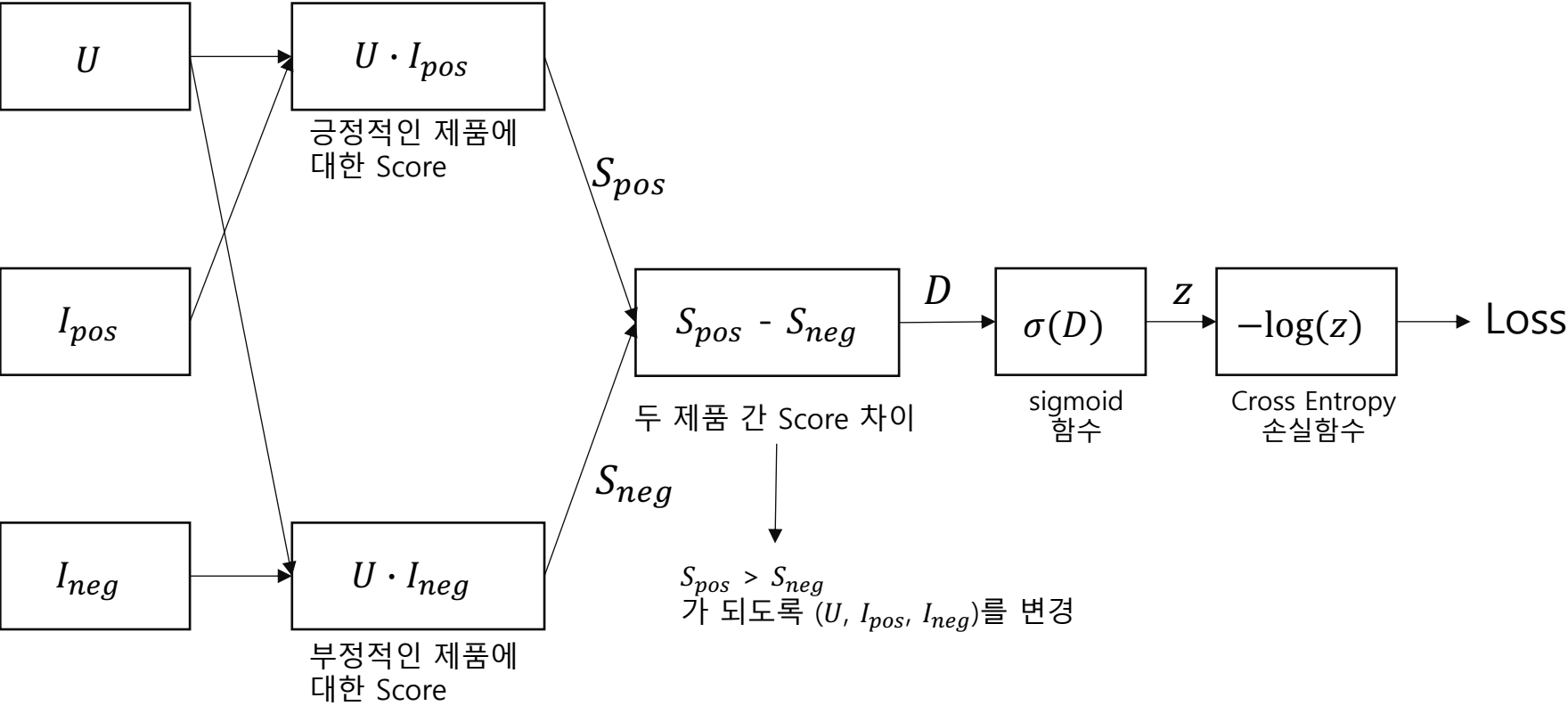
Appendix

BPR의 수식을 계산 그래프로 표현



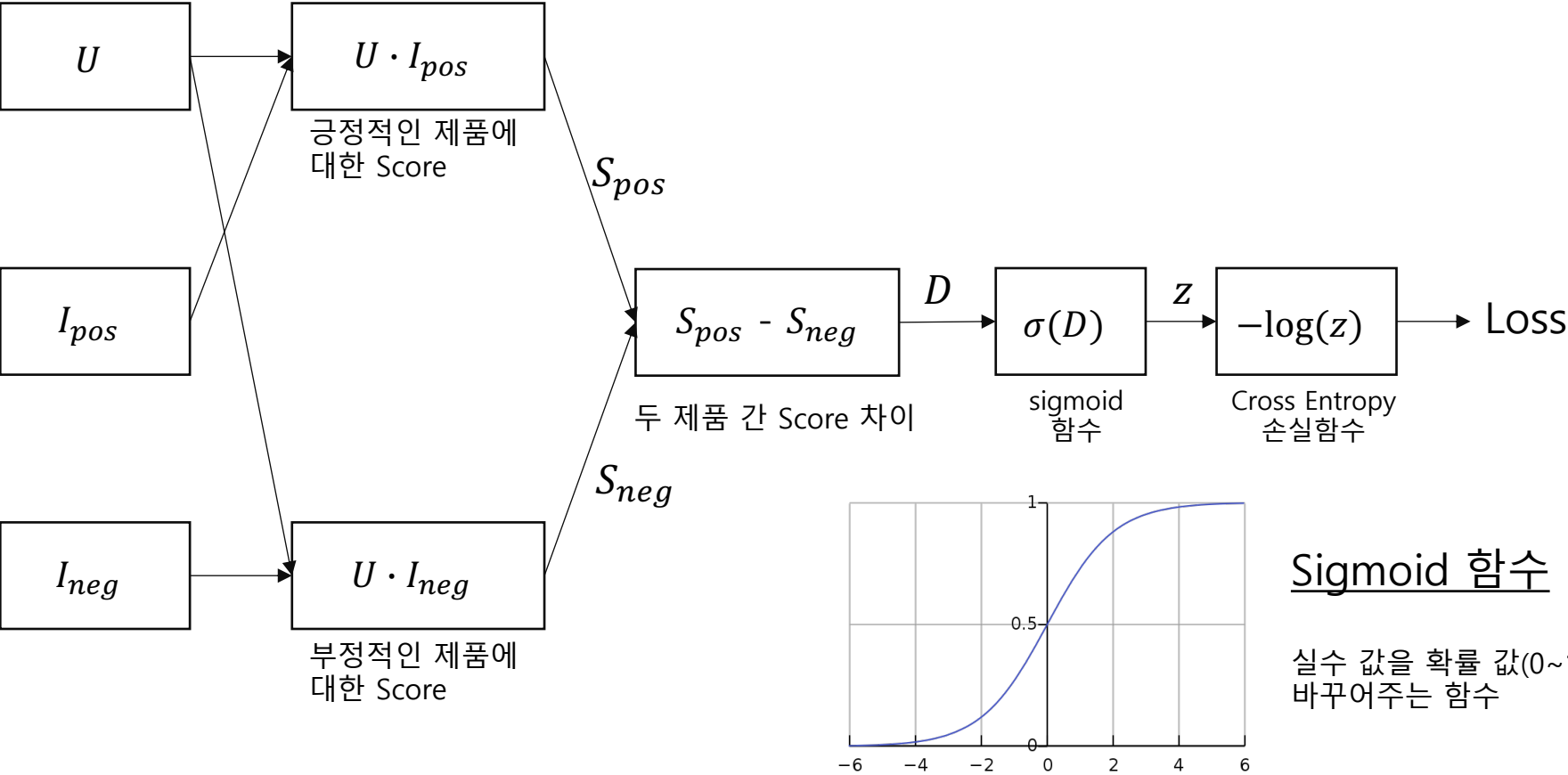
Appendix

BPR의 수식을 계산 그래프로 표현



Appendix

BPR의 수식을 계산 그래프로 표현

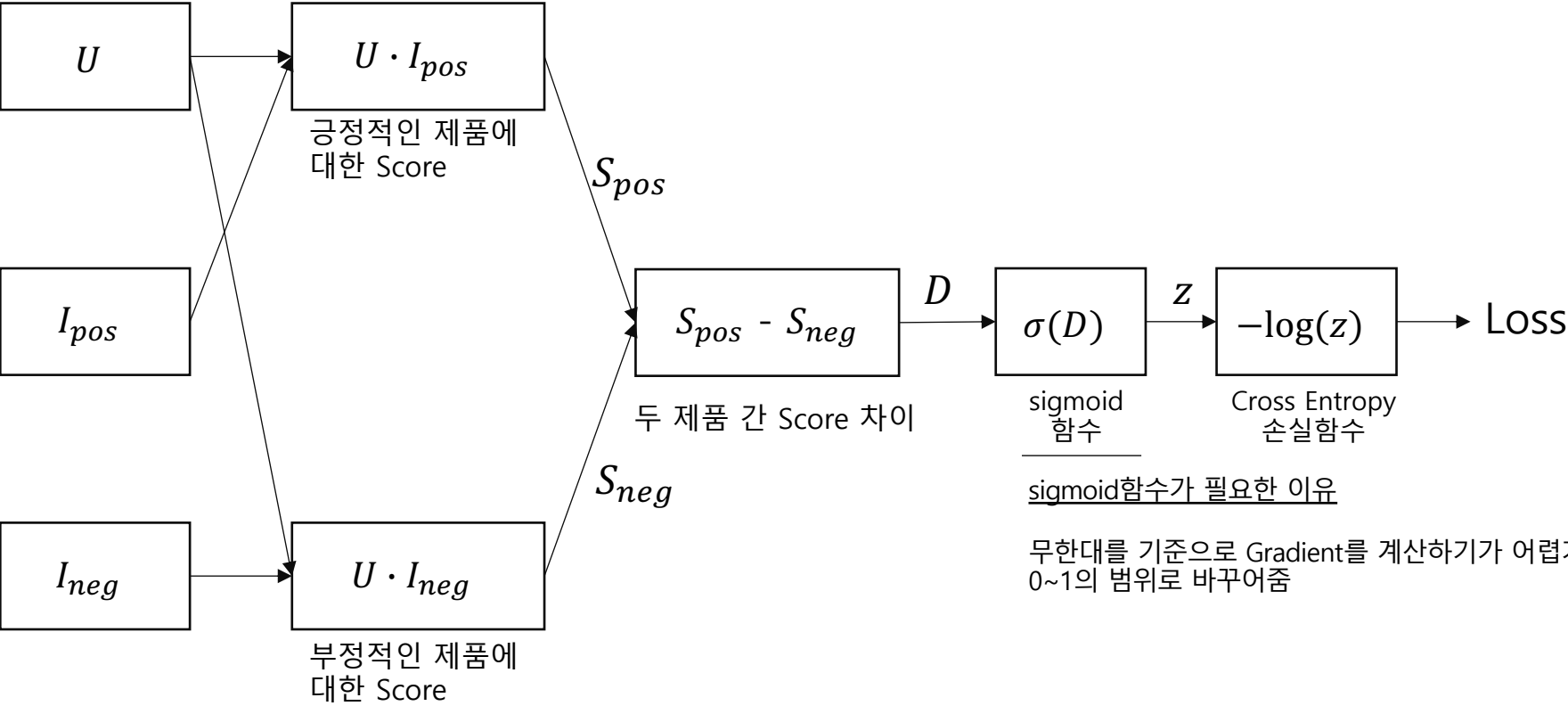


Sigmoid 함수

실수 값을 확률 값(0~1) 사이로 바꾸어주는 함수

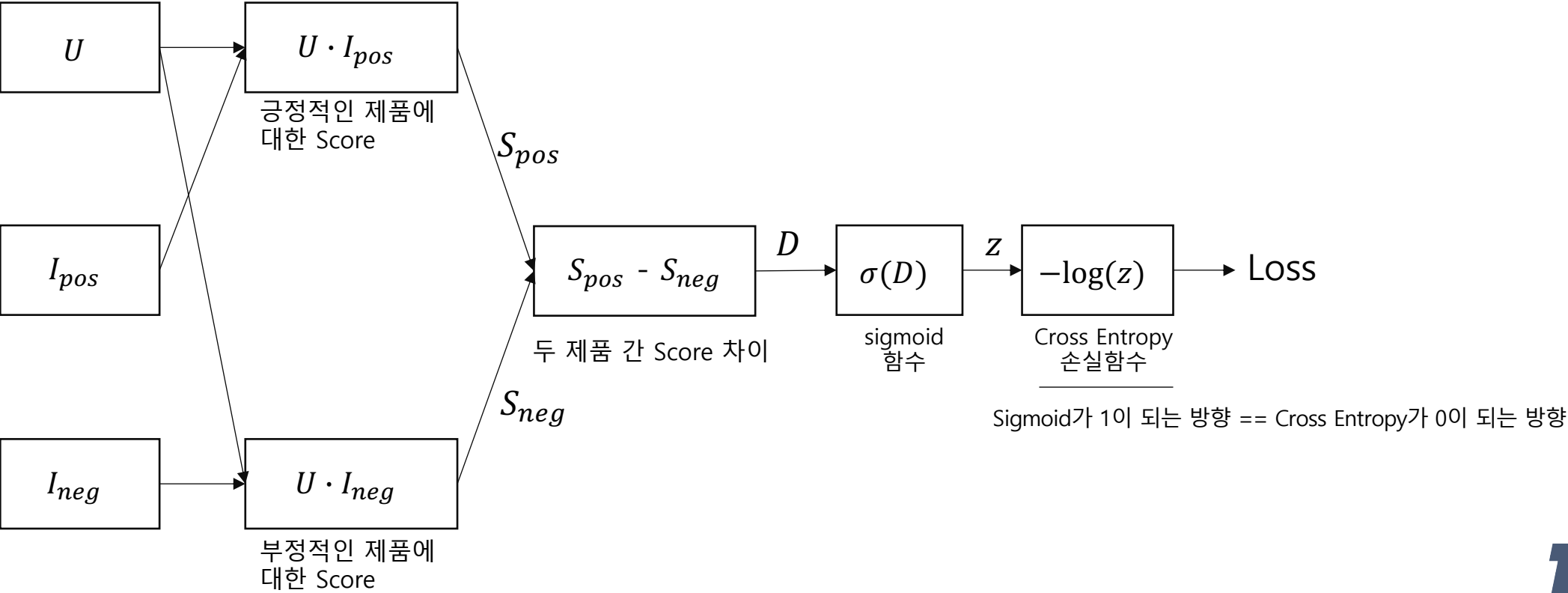
Appendix

BPR의 수식을 계산 그래프로 표현



Appendix

BPR의 수식을 계산 그래프로 표현

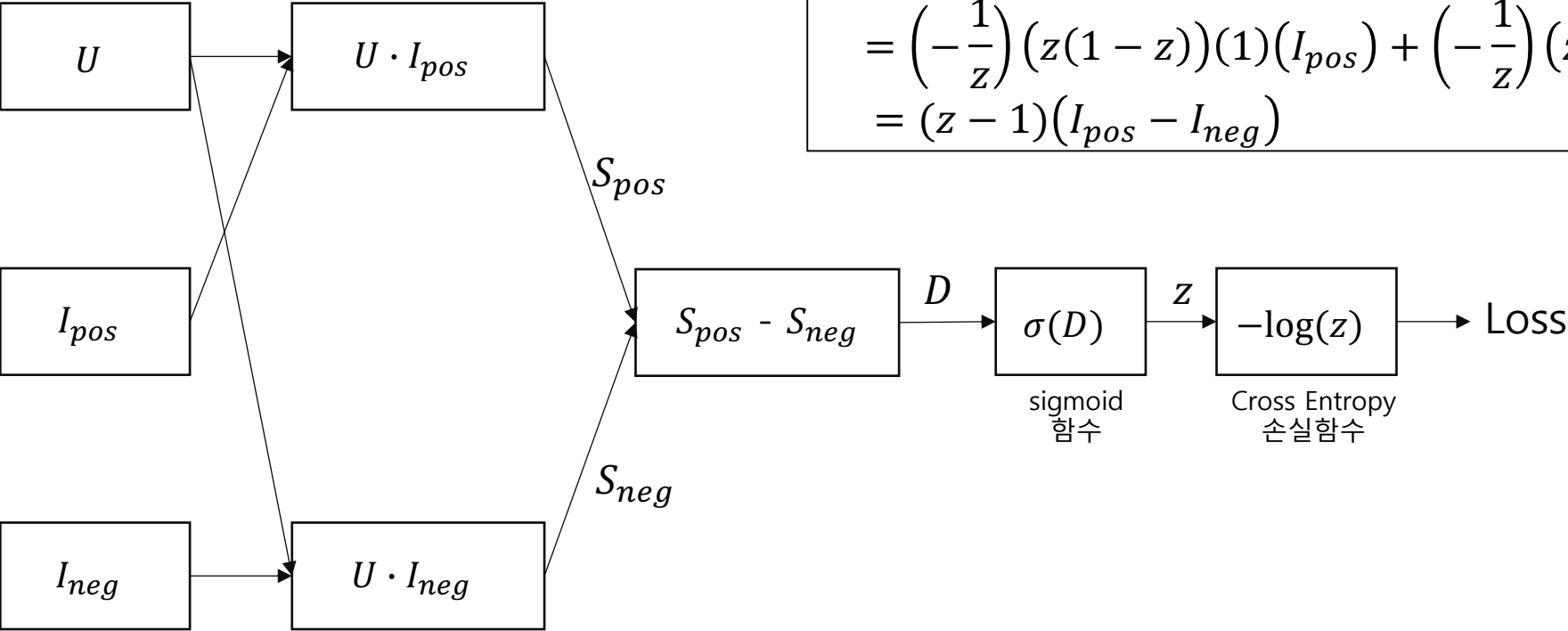


Appendix

기울기 값을 찾는 방식

유저에 대한 기울기 값 찾기

$$\begin{aligned}\frac{\partial L}{\partial U} &= \frac{\partial L}{\partial Z} \frac{\partial Z}{\partial D} \frac{\partial D}{\partial S_{pos}} \frac{\partial S_{pos}}{\partial U} + \frac{\partial L}{\partial Z} \frac{\partial Z}{\partial D} \frac{\partial D}{\partial S_{neg}} \frac{\partial S_{neg}}{\partial U} \\ &= \left(-\frac{1}{Z}\right)(z(1-z))(1)(I_{pos}) + \left(-\frac{1}{Z}\right)(z(1-z))(-1)(I_{neg}) \\ &= (z-1)(I_{pos} - I_{neg})\end{aligned}$$

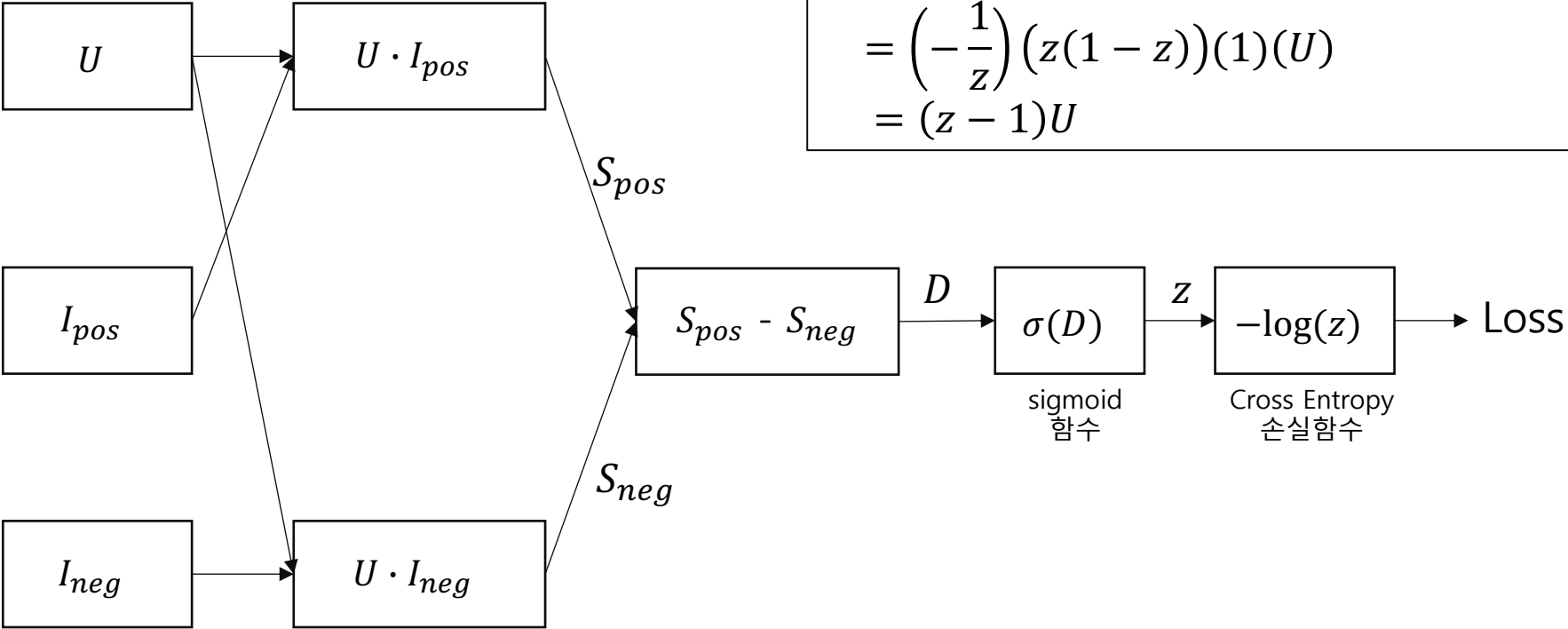


Appendix

기울기 값을 찾는 방식

아이템에 대한 기울기 값 찾기

$$\frac{\partial L}{\partial I_{pos}} = \frac{\partial L}{\partial Z} \frac{\partial Z}{\partial D} \frac{\partial D}{\partial S_{pos}} \frac{\partial S_{pos}}{\partial I_{pos}}$$
$$= \left(-\frac{1}{z}\right) (z(1-z))(1)(U)$$
$$= (z-1)U$$



Appendix

기울기 값을 찾는 방식

아이템에 대한 기울기 값 찾기

$$\frac{\partial L}{\partial I_{neg}} = \frac{\partial L}{\partial z} \frac{\partial z}{\partial D} \frac{\partial D}{\partial S_{neg}} \frac{\partial S_{neg}}{\partial I_{neg}}$$
$$= \left(-\frac{1}{z}\right) (z(1-z)) (-1) (U)$$
$$= -(z-1)U$$

