

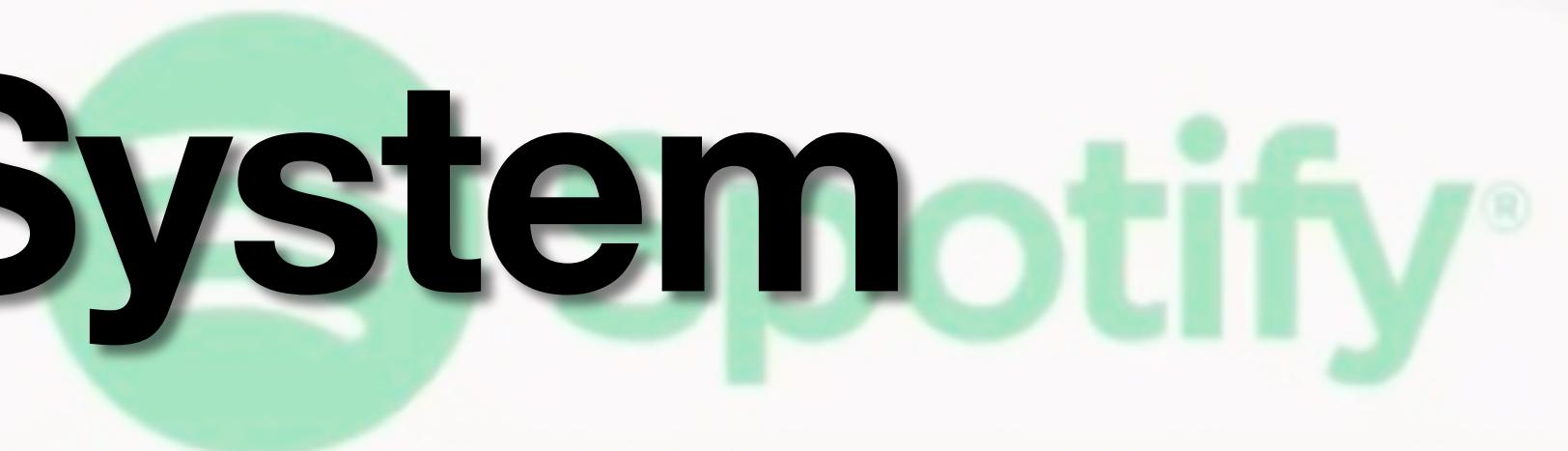


NETFLIX

# Recommendation System

GOOGLE

amazon



Spotify®

1.

## Overview

- What is the Recommendation System?
- Demographic Filtering
- Content based Filtering
- Collaborative Filtering

2. Neural Collaborative Filtering (2017)

# 1. Overview

J W · 6일 전  
유튜브 이거 나한테 왜 추천한거야? 고마워  
1.7천 34

답글 34개 보기

아줌 · 2일 전  
유튜브 알고리즘이 이렇게 고마웠던 적은 없었다  
383 1

답글 보기

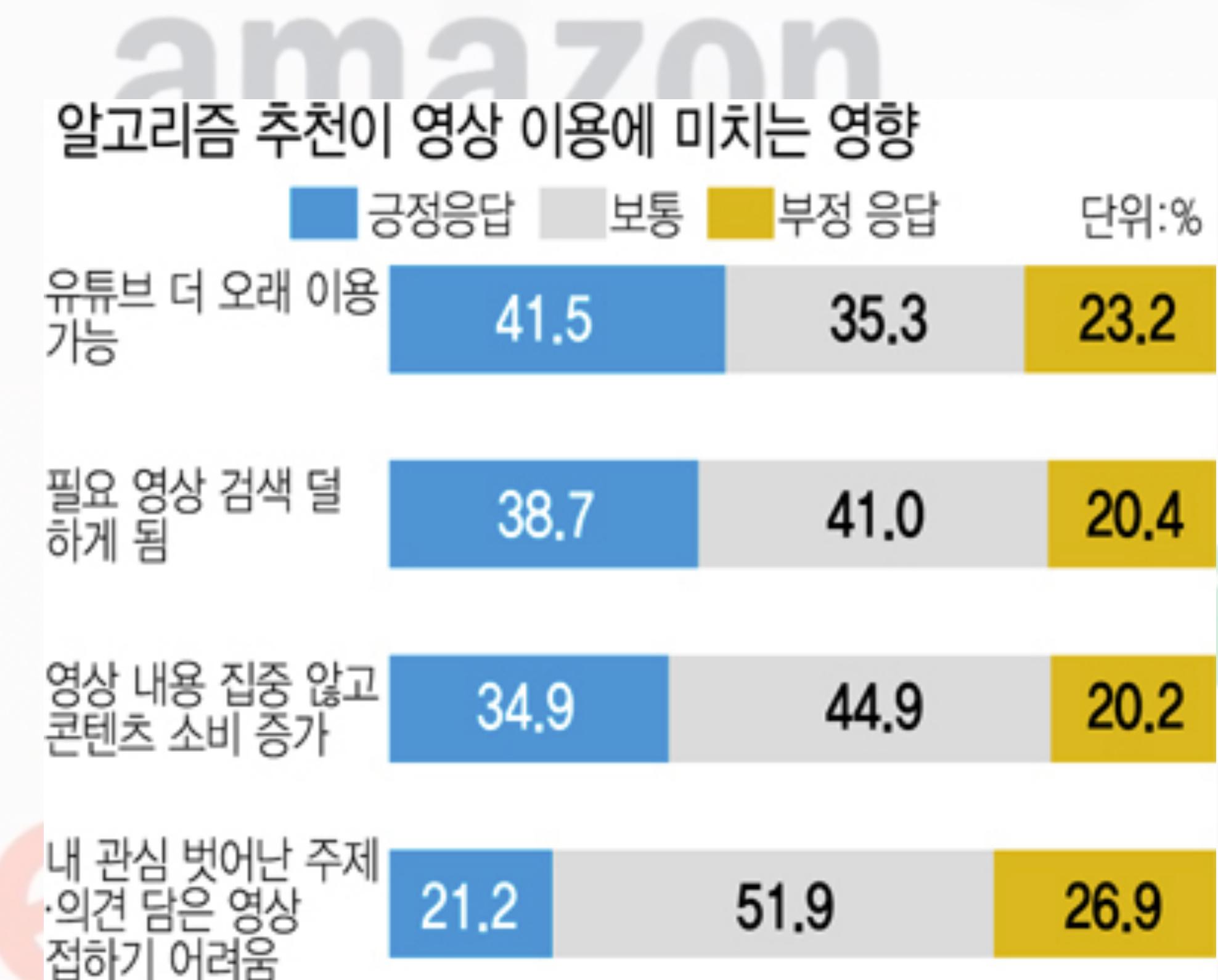


도표 출처: <http://m.knnews.co.kr/mView.php?idxno=1325476&gubun=>

# 1. Overview

**추천 시스템 Recommendation System:** 사용자의 선호 및 과거의 행동을 바탕으로 개인에게 알맞는 아이템(컨텐츠)를 제공하는 시스템

1. 인구통계학적 **Demographic 필터링**: 동일한 인구통계학적 특징을 가진 모든 유저에게 동일한 아이템을 추천
2. 컨텐츠 기반 **필터링**: 어떤 유저  $x$ 에 대하여, 그 유저가 특정 아이템  $i$ 를 선호한다면, 그  $i$ 와 유사한 아이템  $j$ 를 유저  $x$ 에게 추천
3. 협업적 **Collaborative 필터링**: 유저  $x, y, z$ 가 유사한 아이템 군  $S$ 에 관심을 갖는다면, 이  $x, y, z$  간의 매칭을 기초로 하여, 그들에게  $S$  군 내의 아이템을 추천
4. 하이브리드 **필터링**: 컨텐츠 기반 필터링과 협업적 필터링을 혼합한 형태

현업에서 가장 많이 사용 되는 방법? 협업적 필터링, 하이브리드 필터링

# 1. Overview

## 1. 인구통계학적 Demographic 필터링:



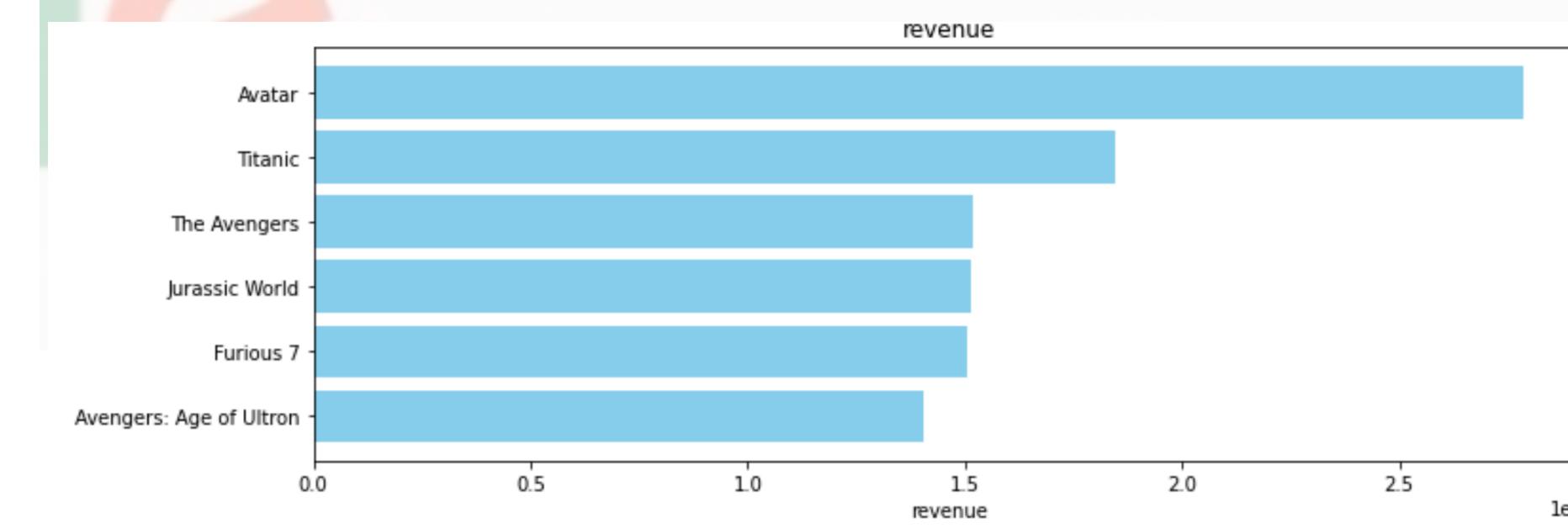
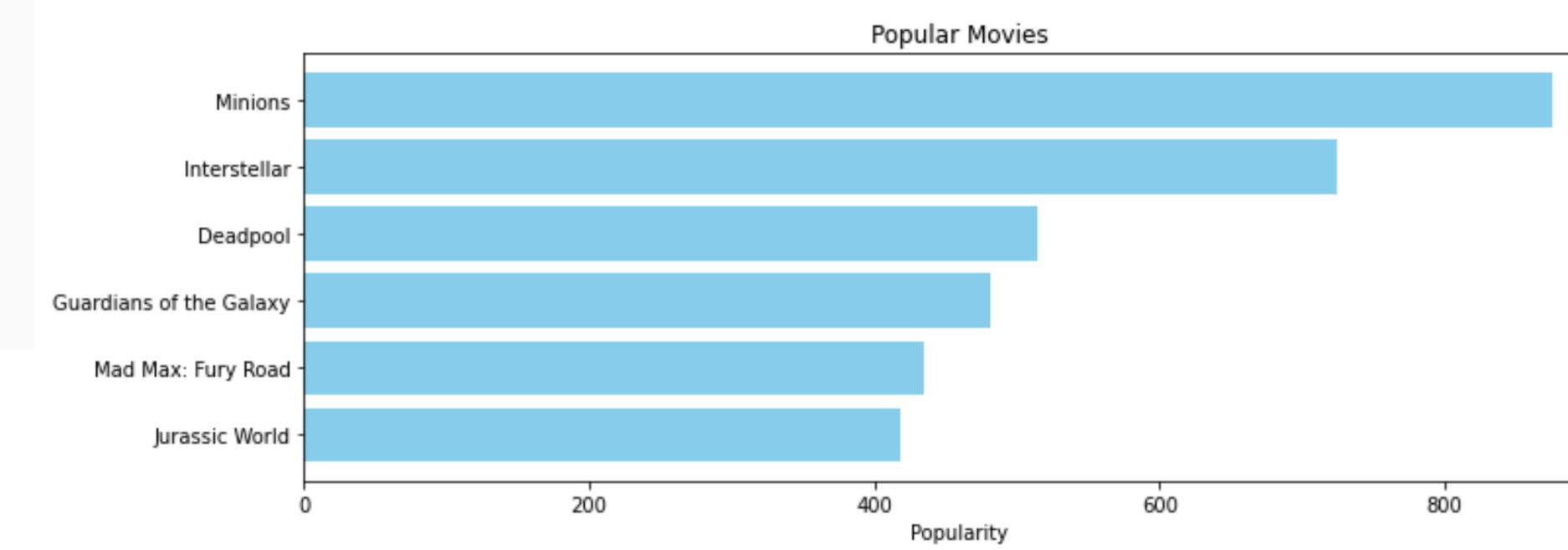
1. 영화 평점을 매길 측정 지표가 필요하다

$$\text{Weighted Rating (WR)} = \left( \frac{v}{v+m} \cdot R \right) + \left( \frac{m}{v+m} \cdot C \right)$$

v is the number of votes for the movie;  
m is the minimum votes required to be listed in the chart;  
R is the average rating of the movie; And  
C is the mean vote across the whole report

2. 모든 영화에 대해 위 측정에 따라 점수를 매긴다.

3. 가장 평점이 좋은 영화를 유저에게 추천해준다.



# 1. Overview

## 2. 협업 필터링

### 2.A 이웃 방법 협업 필터링

#### 2.1 유저 기반 협업 필터링

#### 2.2 아이템 기반 협업 필터링

### 2.B 잠재 요인 모델 방법 협업 필터링

	 liking button		 liking button	 liking button
	 liking button	 liking button		
			 liking button	 liking button
		 liking button		 liking button

# 1. Overview

## 2. 협업 필터링

### 2.1 유저 기반 협업 필터링

### 2.2 아이템 기반 협업 필터링

Movies	아이언맨	트와일리잇	007 콜드레이트	엑스맨
Users		1	0	1
		1	1	0
		0	0	1
		0	1	0

# 1. Overview

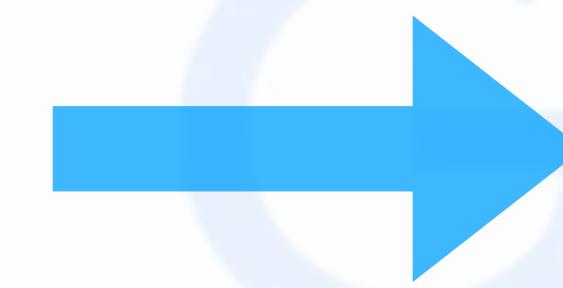
## 2. 협업 필터링

### 2.1 유저 기반 협업 필터링

### 2.2 아이템 기반 협업 필터링

Movies Users	아이언맨	트와일리잇	007 콜드레이트	엑스맨
User1	1	0	1	1
User2	1	1	0	0
User3	0	0	1	1
User4	0	1	0	1

User3에게  
어떤 영화를 추천해줄까?



# 1. Overview

## 2. 협업 필터링

### 2.1 유저 기반 협업 필터링

### 2.2 아이템 기반 협업 필터링

NETFLIX

User3에게  
어떤 영화를 추천해줄까?

유저 간 Cosine 유사도 구하기

$$sim(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Movies	아이언맨	트와일리잇	007 콜드레이트	엑스맨
Users	1	0	1	1
	1	1	0	0
User3	0	0	1	1
	0	1	0	1

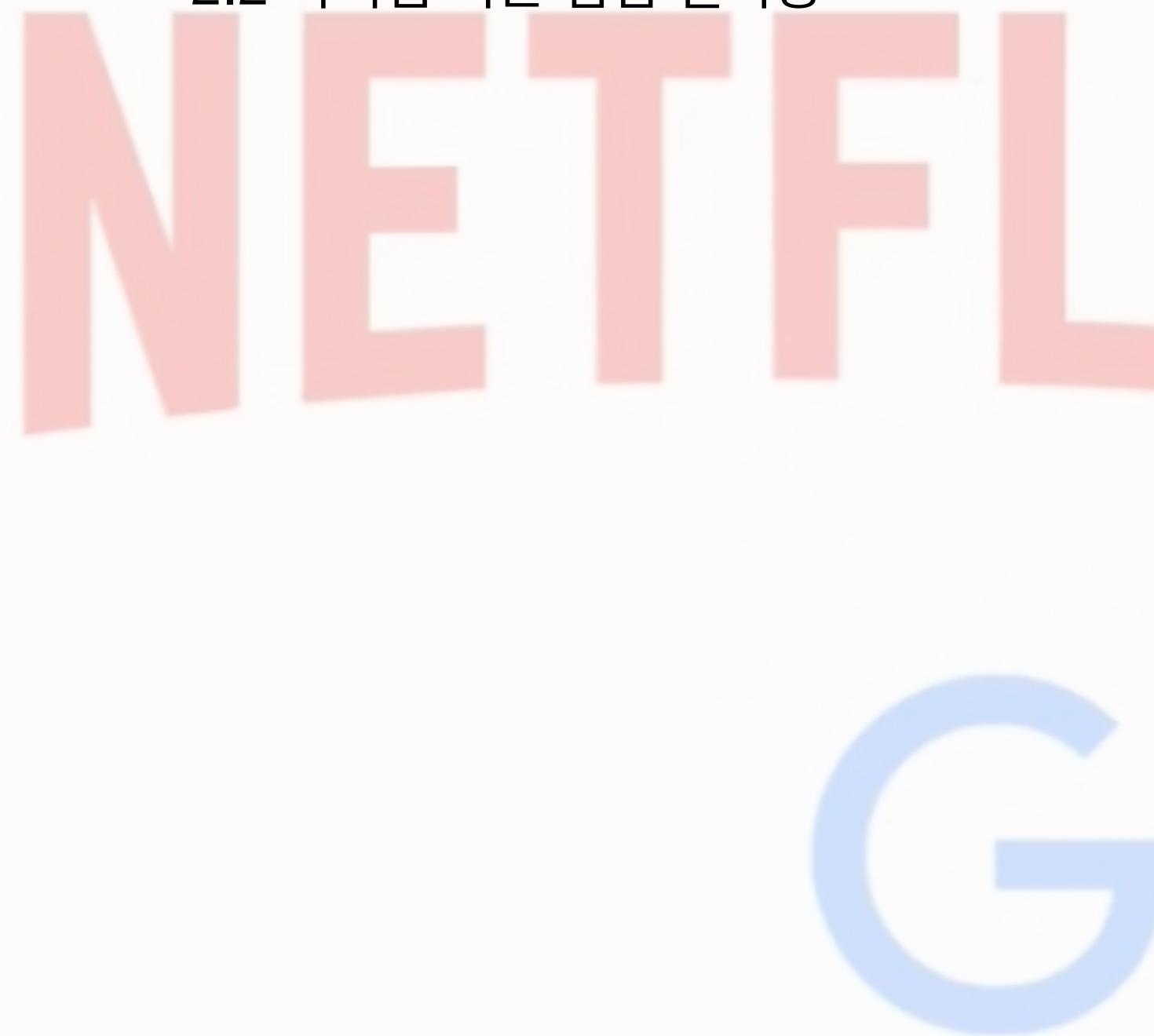
Cosine 유사도

# 1. Overview

## 2. 협업 필터링

### 2.1 유저 기반 협업 필터링

### 2.2 아이템 기반 협업 필터링



```
[1] 1 import numpy as np  
  
[4] 1 mat = np.array([[1,0,1,1],[1,1,0,0],[0,0,1,1],[0,1,0,1]])  
2 print(mat)  
3 print('수염 유저: ', mat[2])  
  
[[1 0 1 1]  
 [1 1 0 0]  
 [0 0 1 1]  
 [0 1 0 1]]  
수염 유저: [0 0 1 1]
```

```
1 def cos(a,b):  
2  
3     dot = np.dot(a,b)  
4     norm_a = np.linalg.norm(a)  
5     norm_b = np.linalg.norm(b)  
6  
7     return dot/(norm_a * norm_b)
```

```
1 print(f'user1과의 코사인 유사도: {cos(mat[0],mat[2])}')  
2 print(f'user2와의 코사인 유사도: {cos(mat[1],mat[2])}')  
3 print(f'user4와의 코사인 유사도: {cos(mat[3],mat[2])}')
```

user1과의 코사인 유사도: 0.8164965809277259  
user2와의 코사인 유사도: 0.0  
user4와의 코사인 유사도: 0.4999999999999999

# 1. Overview

## 2. 협업 필터링

### 2.1 유저 기반 협업 필터링

### 2.2 아이템 기반 협업 필터링

NETFLIX

User3에게  
어떤 영화를 추천해줄까?

유저 간 Cosine 유사도 구하기

$$sim(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Movies	아이언맨	트와일리잇	007 콜드레이트	엑스맨	Cosine 유사도
Users	1	0	1	1	0.81
User1	1	1	0	0	0.00
User2	0	0	1	1	0.50
User3	0	1	0	1	

# 1. Overview

## 2. 협업 필터링

2.1 유저 기반 협업 필터링

2.2 아이템 기반 협업 필터링

Movies	아이언맨	트와일리잇	007 콜드레이트	엑스맨	Cosine 유사도
Users	1	0	1	1	0.81
Users	0	0	1	1	0.00
Users	0	0	1	1	0.50

A diagram illustrating collaborative filtering. It shows a matrix where users (Rows) rate movies (Columns). The matrix is partially filled with binary values (0 or 1). A red arrow points from the '1' in the first user's row to the '007' movie column, indicating a recommendation. The diagonal line from top-left to bottom-right is labeled 'Movies' and the vertical line on the left is labeled 'Users'. The matrix has 4 rows (users) and 5 columns (movies). The columns represent movies: Iron Man, Twilight, GoldenEye, and X-Men. The rows represent users: a man in a suit, a woman with blonde hair, a man with a beard, and a man in a blue shirt.

아이템 기반 협업 필터링을 사용할 때, 영화 007을 좋아하는 유저에게 어떤 영화를 추천할까?

# 1. Overview

## 2. 협업 필터링

2.1 유저 기반 협업 필터링

2.2 아이템 기반 협업 필터링

Users Movies					Cosine 유사도
	User 1	User 2	User 3	User 4	
아이언맨	1	1	0	0	0.49
트와일라잇	0	1	0	1	0.00
스페셜 007	1	0	1	0	
기생충	1	0	1	1	0.82

# 1. Overview

아이템 기반 협업 필터링을 사용할 때, 영화 ‘엑스맨’을 좋아하는 유저에게 어떤 영화를 추천할까?

## 2. 협업 필터링

2.1 유저 기반 협업 필터링

2.2 아이템 기반 협업 필터링

		Users					Cosine 유사도
		Movies					
			1	1	0	0	0.41
			0	1	0	1	0.41
			1	0	1	0	0.82
			1	0	1	1	

# 1. Overview

아이템 기반 협업 필터링을 사용할 때, 영화 ‘엑스맨’을 좋아하는 유저에게 어떤 영화를 추천할까?

## 2. 협업 필터링

2.1 유저 기반 협업 필터링

2.2 아이템 기반 협업 필터링

		Users	Movies	Cosine 유사도			
				Iron Man	X-Men	Spider-Man	
				Twilight	James Bond	Avengers	0.41
				1	1	0	0.41
				0	1	0	0.41
				1	0	1	0.82
				1	0	1	0.82

# 1. Overview

## 2. 협업 필터링

### 협업 필터링이 지닌 곤경

1. **Cold start:** 새로운 영화 ‘아이언맨 4’가 나왔다고 해보자. 이 경우 어떤 유저도 이 영화에 대해 평점을 매기지 않았을 것이다. 새 아이템이 추가된 경우에 협업 필터링을 사용하기는 어렵다.

2. **Scalability:** 많은 수의 사용자, 아이템이 있는 경우에, 많은 양의 계산이 요구된다. 이 계산을 수행하기 위해, 많은 양의 Computing Power가 필요하다.

3. **Long-tail:** 추천 시스템이 관리하는 아이템이 많은 경우에, 특정 아이템에 대한 선호가 쏠리는 현상이 발생할 수 있다. 즉, 대다수의 유저가 소수의 아이템에 대해 선호하고, 나머지 비교적 선호되지 못하는 다수의 아이템들은 추천 목록에서 제외될 수 있다.

(e.g.) 유저 X,Y,Z,W와 아이템 a,b,c,d,e가 있다고 해보자. 유저 X,Y는 a,b만을 선호하고 Z는 a,d를 선호한다고 해보자. 그리고 새로 가입한 유저 W는 d에 대해 선호를 표시했다고 해보자. 이 경우 c,e는 추천 시스템에서 고려되지 않을 것이다. 이 작은 사고 실험이 큰 사이즈로 확대된다면, 추천 시스템 내에서 선호되는 아이템들 간의 **비대칭적 쏠림 현상**이 나타날 수 있다.

# 1.Overview

다음 장에서 소개할 컨텐츠 기반 필터링은 협업 필터링이 가진 곤경들을 피해할 수 있다!

# 1. Overview

## 3. 컨텐츠 기반 필터링

컨텐츠 기반 필터링은 오직 설명description에 대한 정보와 유저가 과거에 선호한 아이템과의 상호성만을 사용한다.  
즉, 이 방법은 추천 시스템이 유저가 이전에 선호한 아이템과 유사한 아이템을 추천하게 해준다.

어떻게 컴퓨터가 아이템 간의 유사성을 판단하는가?

- Clustering Analysis
- Artificial Neural Net
- TF-IDF

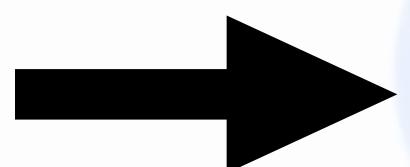
# 1. Overview

## 3. 컨텐츠 기반 필터링

### TF-IDF (Term Frequency - Inverse Document Frequency)

- TF : 특정 단어가 특정 문서document 내에서 몇번 등장했는가. 높을수록 빈번히 나타나는 단어임을 의미
- DF: 특정 단어가 여러 문서들 내에서 등장한 빈도. ['호랑이'가 doc1,doc20 에서 등장했다면, '호랑이'의 df=2]
- IDF:  $\log\left(\frac{\text{num\_words}}{\text{df}(\text{certain\_word})}\right)$ . 값이 클수록 해당 certain word가 특이한 단어임을 의미

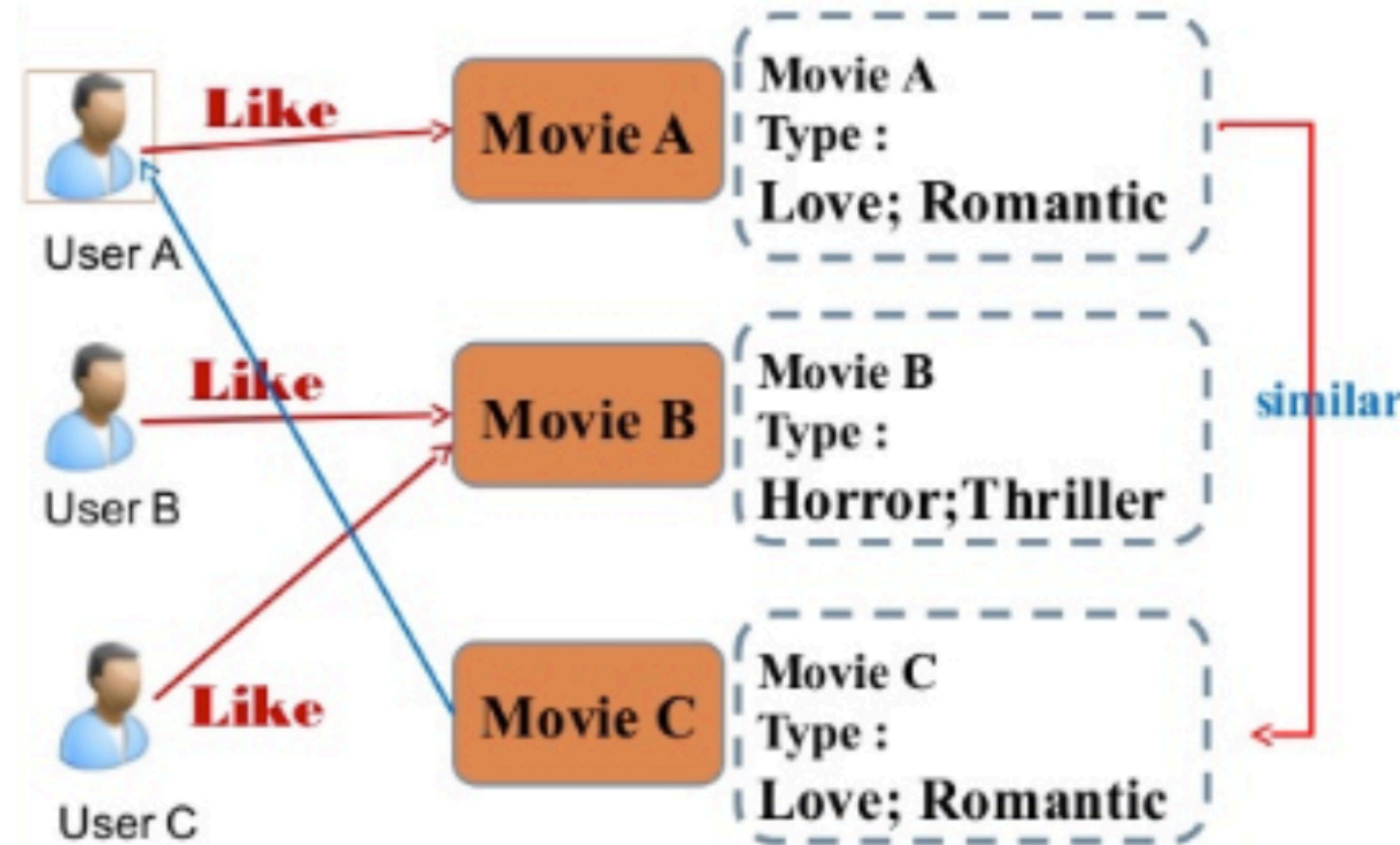
$$TF - IDF(w) = tf(w) \times \log\left(\frac{N}{df(w)}\right)$$



TF-IDF는 (1) 해당 단어가 얼마나 등장하는가, (2) 얼마나 특이한가 (사용되지 않았는가)를 모두 반영한 수치

# 1. Overview

## 3. 컨텐츠 기반 필터링



## 2. Neural Collaborative Filtering (2017)

- 1. Intro
- 2. Preliminaries
- 3. Neural Collaborative Filtering
- 4. Experiments

**배경:** 2009년에 기존의 Netflix의 추천 시스템의 성능을 10% 이상 개선해주는 알고리즘이 Netflix 공모전에서 수상작이 되었다. 이것은 행렬 인수분해 **Matrix Factorization**를 이용한 방법이다. 이것이 소개된 이후, 이 MF는 협업 필터링 방법 중 가장 인기있는 방법으로 여겨진다.



### [6] 저자들의 MF에 대한 지적

사용자와 아이템 간의 상호성을 표현하기에 inner product는 너무 단순한 선택이다. 이것은 복잡한 그들 간의 상호성을 포착하기에 충분치 않다.

### [6] 저자들의 제안

MF 뿐만 아니라, 최근 분야에서 사용되는 DNN을 도입해보자.

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
4. Experiments

### Matrix Factorization

$$\hat{y}_{ui} = f(u, i | \mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u^T \mathbf{q}_i = \sum_{k=1}^K p_{uk} q_{ik},$$

$\hat{y}_{ui}$ : 예측된 상호성 정도 interaction

**u**: users

**i**: items

$p_u$ : user에 대한 latent vector (유저가 어떤 factors를 소유한 정도를 측정)

$q_i$ : item에 대한 latent vector (유저가 어떤 factors를 소유한 정도를 측정)

**K**: latent space의 차원

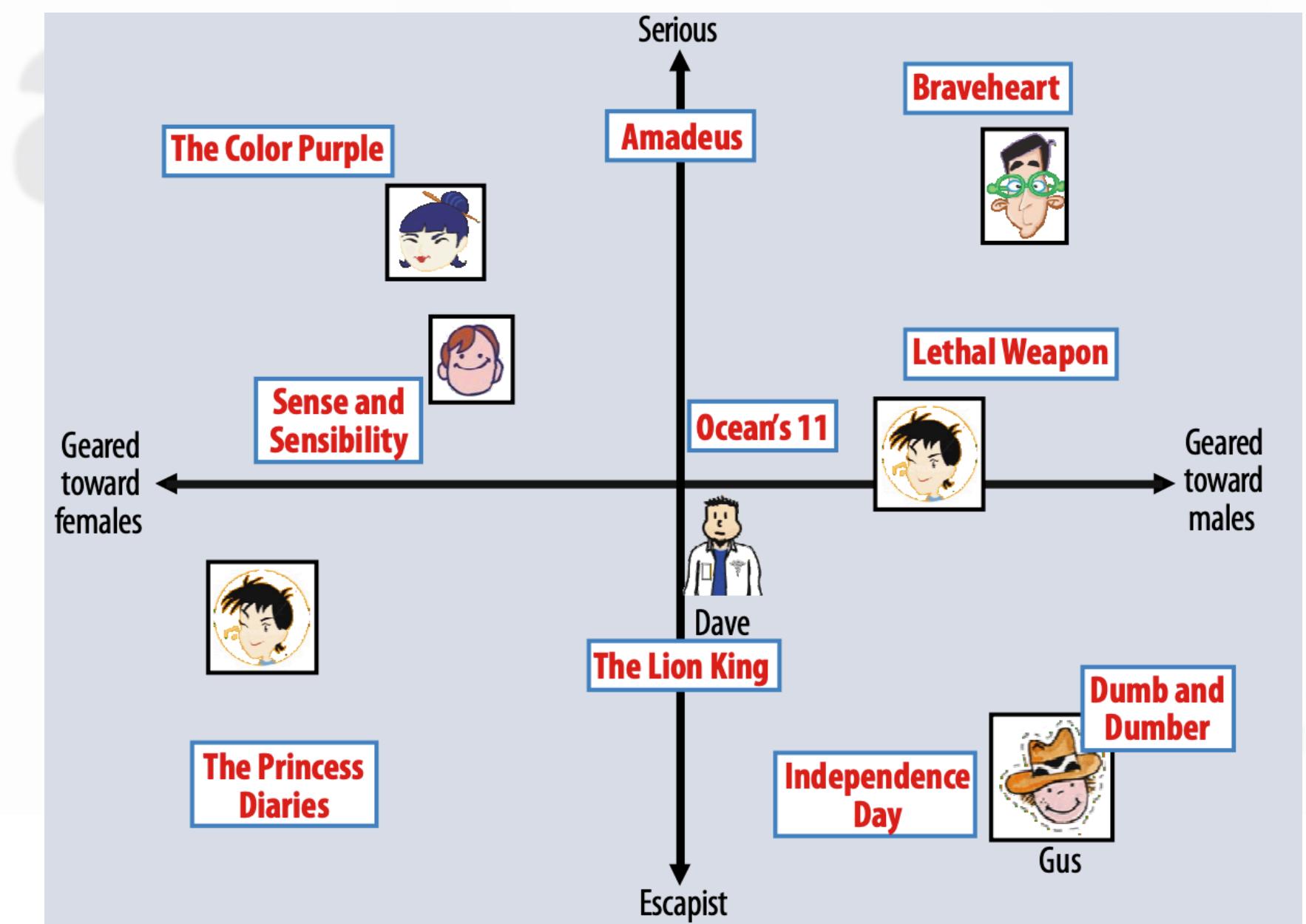


Figure 2. A simplified illustration of the latent factor approach, which characterizes both users and movies using two axes—male versus female and serious versus escapist.

# 보충: Matrix Factorization

	아이템1	아이템2	아이템3	아이템4	아이템5
사용자1	4.0			2.0	
사용자2		5.0		3.0	1.0
사용자3			3.0	4.0	4.0
사용자4	5.0	2.0	1.0	2.0	

사용자가 각 아이템에 대한 선호를 인지하지 못하는 경우가 있다.  
이 경우, Interaction Matrix는 Sparse Matrix(empty를 0으로 간주)  
형태를 갖는다.

	요인1	요인2	요인3		아이템1	아이템2	아이템3	아이템4	아이템5
사용자1	0.96	0.47	-0.76		1.62	-0.79	1.04	1.07	1.43
사용자2	-0.03	0.84	-2.47	*	1.51	0.45	-0.06	0.12	-0.21
사용자3	2.38	0.11	-1.20		-2.22	-1.85	0.43	1.18	-0.50
사용자4	0.59	1.10	-1.06						

MF의 수행에서는, users, items 간의 (유의미한) 잠재된  
요인이 있다고 가정한 뒤, MF 행렬 인수분해를 통해 그 요  
인을 찾아낸다.

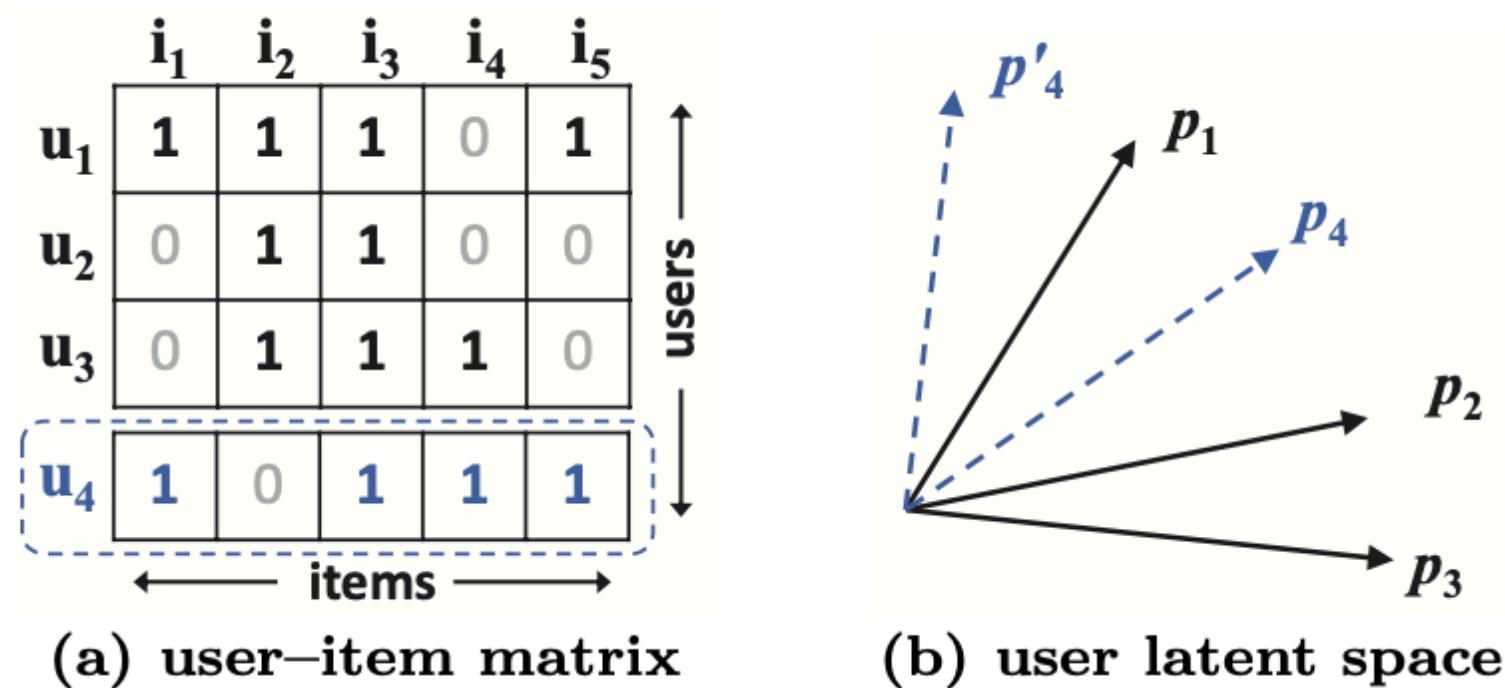
	아이템1	아이템2	아이템3	아이템4	아이템5
사용자1	3.99	0.90	1.31	2.00	1.66
사용자2	6.70	4.98	0.98	3.00	1.00
사용자3	6.68	0.39	3.00	3.98	3.99
사용자4	4.97	2.01	1.01	2.02	1.14

위에서 분해된 두 행렬을 내적하여 원행렬을 예측

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
4. Experiments

내적 함수는 Matrix Factorization의 표현성을 제약할 수 있다



**Figure 1: An example illustrates MF's limitation.**  
From data matrix (a),  $u_4$  is most similar to  $u_1$ , followed by  $u_3$ , and lastly  $u_2$ . However in the latent space (b), placing  $p_4$  closest to  $p_1$  makes  $p_4$  closer to  $p_2$  than  $p_3$ , incurring a large ranking loss.

(a)는 interactions matrix. 여기서 두 유저 간 유사도는 inner product, cosine 유사도로 측정될 수 있다.

Step 1

(a)에서,  $u_1, u_2, u_3$  간의 유사도를 구하면 (b)의 검은 선들 간의 관계로 나타나낼 수 있다.

u<sub>2</sub>와 u<sub>3</sub>의 유사도: 0.8164965809277259  
u<sub>1</sub>와 u<sub>2</sub>의 유사도: 0.7071067811865475  
u<sub>1</sub>와 u<sub>3</sub>의 유사도: 0.5773502691896258

$u_n$ 에 대한 벡터를  $p_n$ 이라 할 때, 위 계산 결과에 따라,

$p_1$ 은  $p_3$  보다  $p_2$ 와 더 가깝고,  $p_2$ 는  $p_1$  보다  $p_3$ 와 더 가깝다.

Step 2

이제,  $u_4$ 와 나머지 간의 유사도를 구해보자.

u<sub>1</sub>와 u<sub>4</sub>의 유사도: 0.75  
u<sub>2</sub>와 u<sub>4</sub>의 유사도: 0.35355339059327373  
u<sub>3</sub>와 u<sub>4</sub>의 유사도: 0.5773502691896258

위 계산은  $p_4$ 는  $p_1$ 과 가장 가깝고,  $p_2$  보다  $p_3$ 와 더 가깝다는 것을 의미한다.

그러나 문제는  $p_4$ 를  $p_1$ 에 가장 가깝게 둔다면,  $p_4$ 는  $p_3$  보다  $p_2$ 에 더 가까울 것이다.

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
4. Experiments

저자의 내적 함수가 Matrix Factorization의 표현성을 제약할 수 있다는 지적은 적절한가

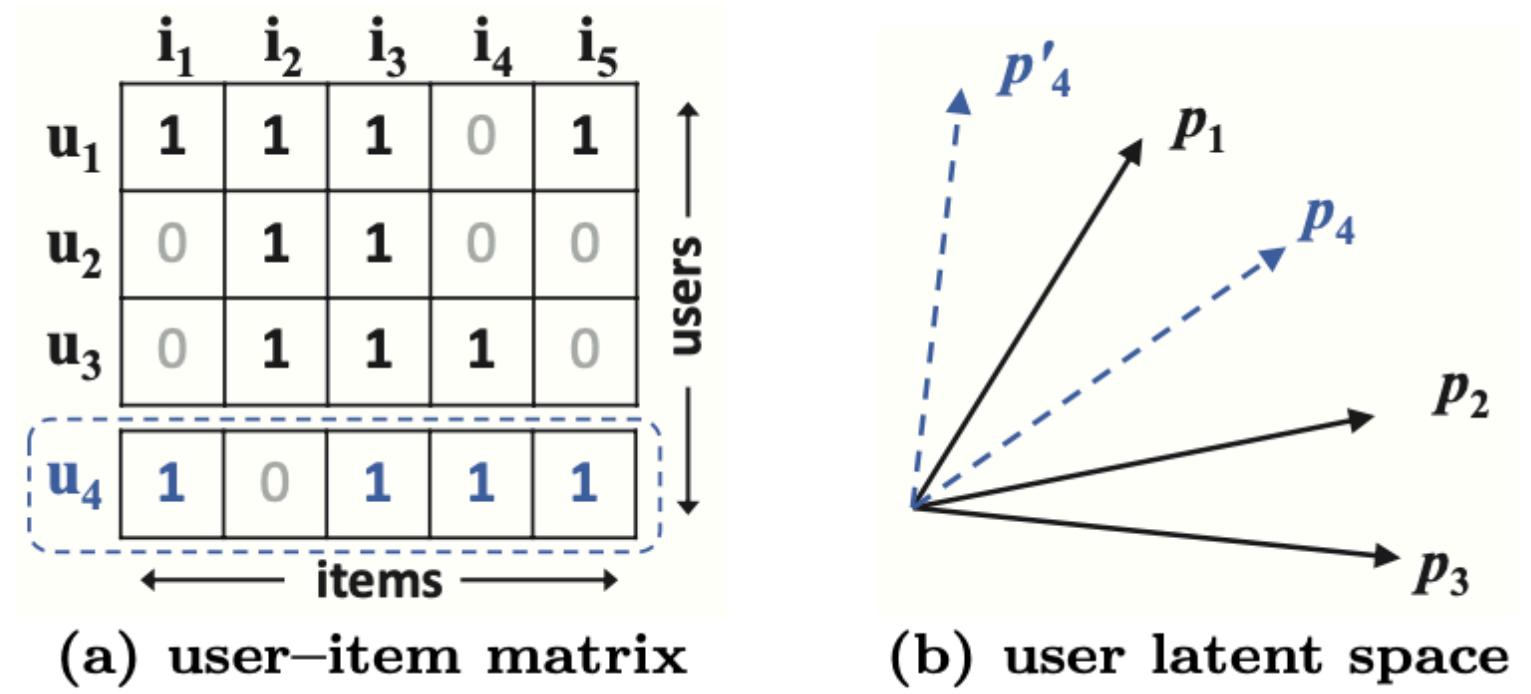
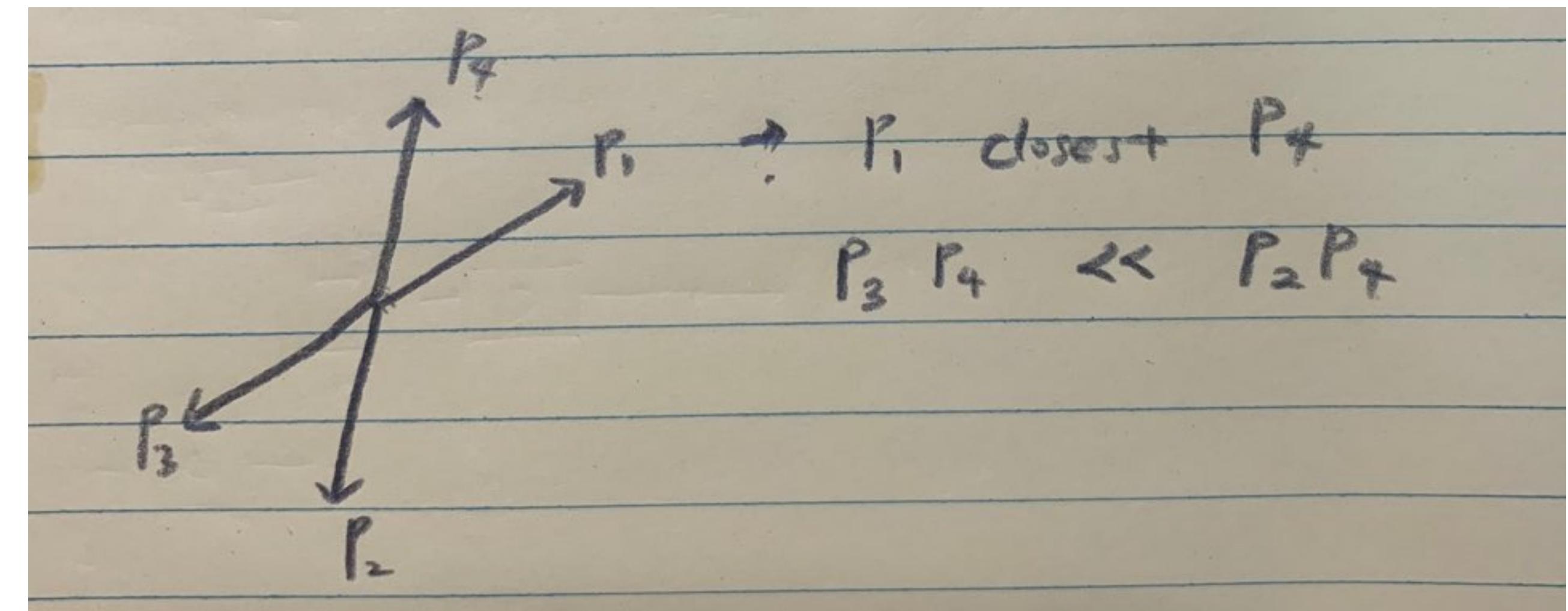


Figure 1: An example illustrates MF's limitation. From data matrix (a),  $u_4$  is most similar to  $u_1$ , followed by  $u_3$ , and lastly  $u_2$ . However in the latent space (b), placing  $p_4$  closest to  $p_1$  makes  $p_4$  closer to  $p_2$  than  $p_3$ , incurring a large ranking loss.



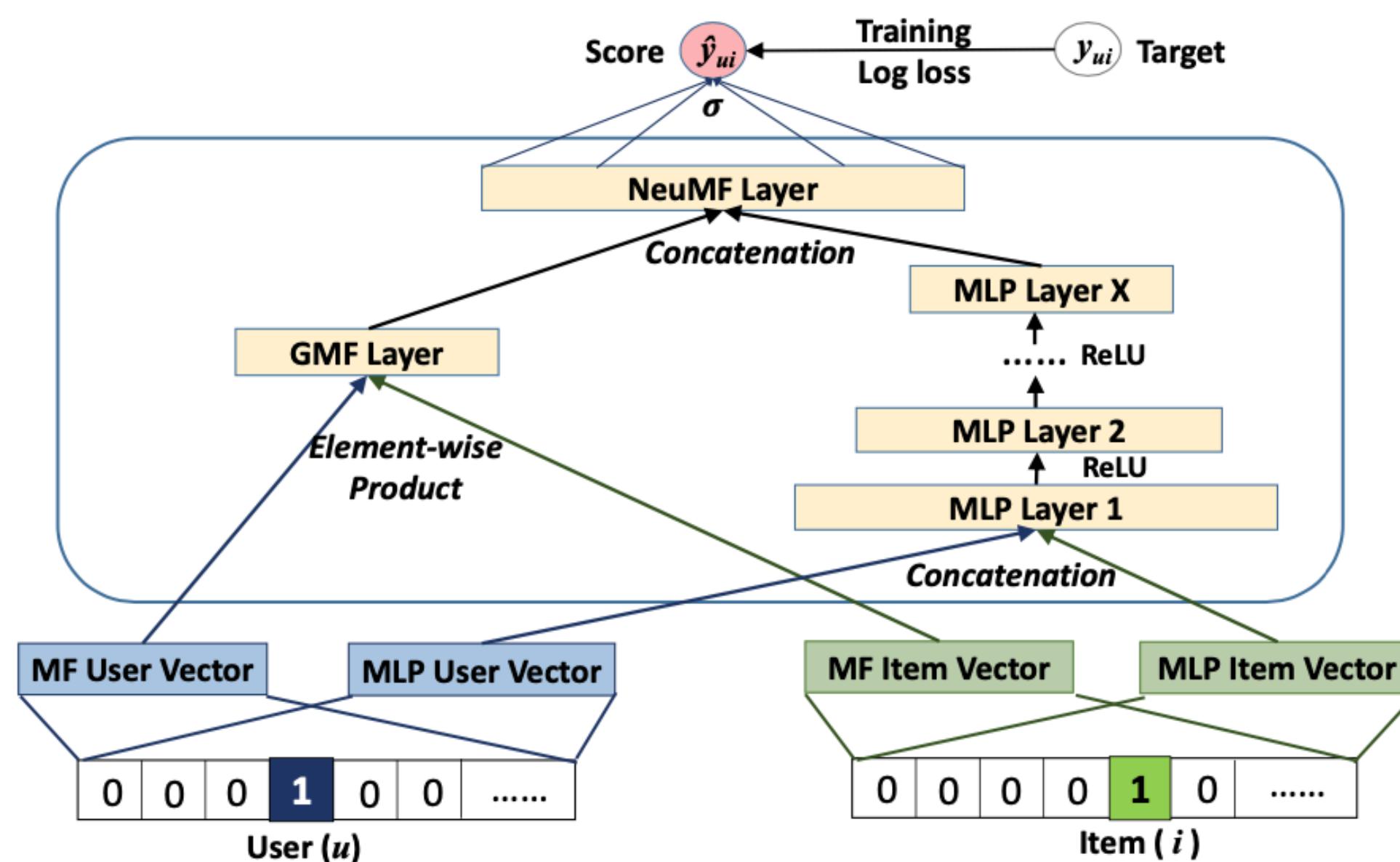
u2와 u3의 유사도: 0.8164965809277259  
u1와 u2의 유사도: 0.7071067811865475  
u1와 u3의 유사도: 0.5773502691896258

u1와 u4의 유사도: 0.75  
u2와 u4의 유사도: 0.35355339059327373  
u3와 u4의 유사도: 0.5773502691896258

위 같이 표현하면 표현의 제약을 피해갈 수 있지 않을까? 유사도를 고려한다면 그렇지 않아 보인다.  $p_1 p_3$  유사도와  $p_3 p_4$  유사도가 동일하다는 것을 고려하여 위 그림을 다시 그려 보면, 유사도 ranking이 보존되지 않는 경우만이 그려진다.

## 2. Neural Collaborative Filtering (2017)

1. Intro
  2. Preliminaries
  - 3. Neural Collaborative Filtering**
  4. Experiments
- 3.1 GMF
  - 3.2 MLP
  - 3.3 NeuMF



**Figure 3: Neural matrix factorization model**

### Learning NCF

Binary cross-entropy loss

$$\begin{aligned} L &= - \sum_{(u,i) \in \mathcal{Y}} \log \hat{y}_{ui} - \sum_{(u,j) \in \mathcal{Y}^-} \log(1 - \hat{y}_{uj}) \\ &= - \sum_{(u,i) \in \mathcal{Y} \cup \mathcal{Y}^-} y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui}). \end{aligned} \quad (7)$$

NCF에 확률적 설명을 부여하기 위해,  $\hat{y}_{ui}$ 의 범위는 [0,1]이다.

추천 시스템은 이진 분류 문제로 다뤄지게 된다.

### Pre-training

딥러닝 모델에서 초기화의 역할은 중요하다. NeuMF는 GMF와 MLP의 양상을 모델이다. 저자는 GMF, MLP의 사전훈련된 모델을 사용하여 NeuMF를 초기화하기를 제안한다.

1. 수렴 때까지 GMF, MLP를 Adam 최적화 방식으로 두 모델 학습
2. we concatenate weights of the two models with

$$\mathbf{h} \leftarrow \begin{bmatrix} \alpha \mathbf{h}^{GMF} \\ (1 - \alpha) \mathbf{h}^{MLP} \end{bmatrix}, \quad (13)$$

where  $\mathbf{h}^{GMF}$  and  $\mathbf{h}^{MLP}$  denote the  $\mathbf{h}$  vector of the pre-trained GMF and MLP model, respectively; and  $\alpha$  is a hyper-parameter determining the trade-off between the two pre-trained models.

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
- 4. Experiments**

### Datasets

- 1.MovieLens
2. Pinterest

**Table 1: Statistics of the evaluation datasets.**

Dataset	Interaction#	Item#	User#	Sparsity
MovieLens	1,000,209	3,706	6,040	95.53%
Pinterest	1,500,809	9,916	55,187	99.73%

1. MovieLens: 유저가 영화에 대해 평점을 매긴다
2. Pinterest: 유저가 이미지에 대해 평점을 매긴다

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
- 4. Experiments**

### 4.1 NCF 방법은 기존의 최신 CF 방법보다 뛰어난가?

- 4.2 어떻게 제안된 최적화 틀(log loss with negative sampling)이 추천 시스템에서 잘 작동하는가?
- 4.3 더 깊은 심층 신경망을 선택하는 것은 interaction data의 학습에 도움이 되는가?

#### 평가 측정 지표

- HR (Hit Ratio)
- NDCG (Normalized Discounted Culmulative Gain) 자세한 설명: <https://lamttic.github.io/2020/03/20/01.html>

**방법 1.** 각 유저에 대해, 그의 최신 interaction을 test set으로 삼고, 나머지를 training set으로 이용한다.

**방법 2.** 우리는 시간 절약을 위해 모든 아이템에 대해서가 아닌 해당 유저와 interact 되지 않은 100 개의 아이템만을 임의로 선정한다.  
그리고 그 100 개 중에서 test item으로 사용할 것에 우선순위를 부여한다.

**방법3.** 그 item들의 성능은 위의 HR, NDCG로 평가한다.

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
4. Experiments

### 4.1 NCF 방법은 기존의 최신 CF 방법보다 뛰어난가?

- 4.2 어떻게 제안된 최적화 틀(log loss with negative sampling)이 추천 시스템에서 잘 작동하는가?
- 4.3 더 깊은 심층 신경망을 선택하는 것은 interaction data의 학습에 도움이 되는가?

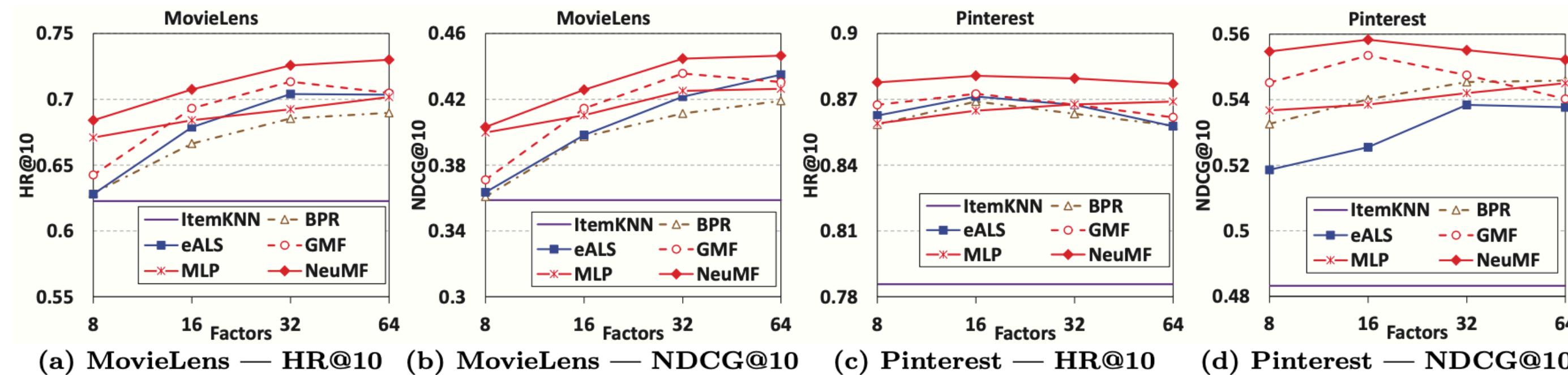


Figure 4: Performance of HR@10 and NDCG@10 w.r.t. the number of predictive factors on the two datasets.

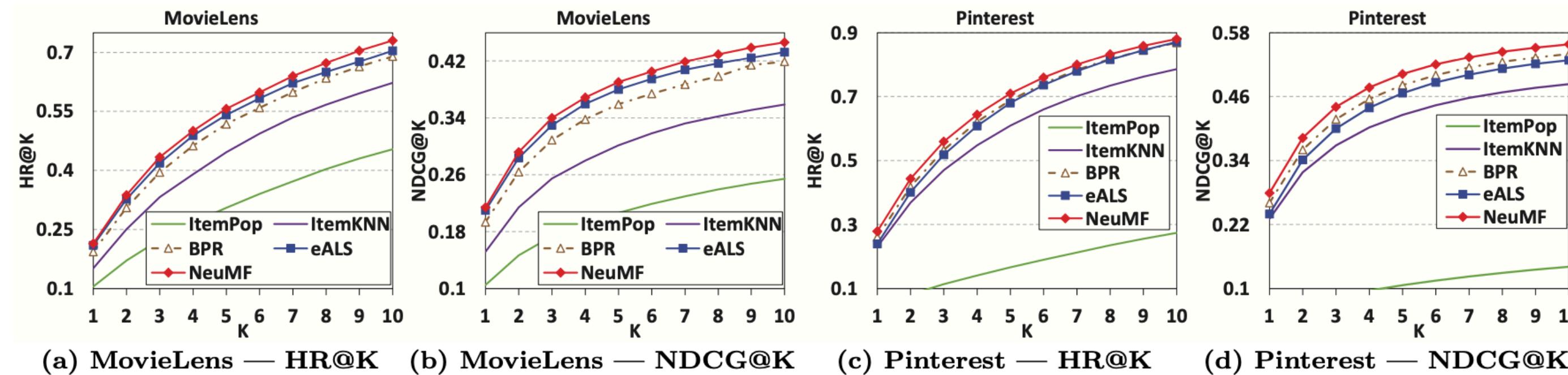


Figure 5: Evaluation of Top-K item recommendation where  $K$  ranges from 1 to 10 on the two datasets.

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
- 4. Experiments**

4.1 NCF 방법은 기존의 최신 CF 방법보다 뛰어난가?

4.2 어떻게 제안된 최적화 틀(log loss with negative sampling)이 추천 시스템에서 잘 작동하는가?

4.3 더 깊은 심층 신경망을 선택하는 것은 interaction data의 학습에 도움이 되는가?

**Table 2: Performance of NeuMF with and without pre-training.**

Factors	With Pre-training		Without Pre-training	
	HR@10	NDCG@10	HR@10	NDCG@10
<b>MovieLens</b>				
8	0.684	0.403	<b>0.688</b>	<b>0.410</b>
16	<b>0.707</b>	<b>0.426</b>	0.696	0.420
32	<b>0.726</b>	<b>0.445</b>	0.701	0.425
64	<b>0.730</b>	<b>0.447</b>	0.705	0.426
<b>Pinterest</b>				
8	<b>0.878</b>	<b>0.555</b>	0.869	0.546
16	<b>0.880</b>	<b>0.558</b>	0.871	0.547
32	<b>0.879</b>	<b>0.555</b>	0.870	0.549
64	<b>0.877</b>	<b>0.552</b>	0.872	0.551

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
4. Experiments

4.1 NCF 방법은 기존의 최신 CF 방법보다 뛰어난가?

4.2 어떻게 제안된 최적화 틀(log loss with negative sampling)은 추천 시스템에서 잘 작동하는가?

4.3 더 깊은 심층 신경망을 선택하는 것은 interaction data의 학습에 도움이 되는가?

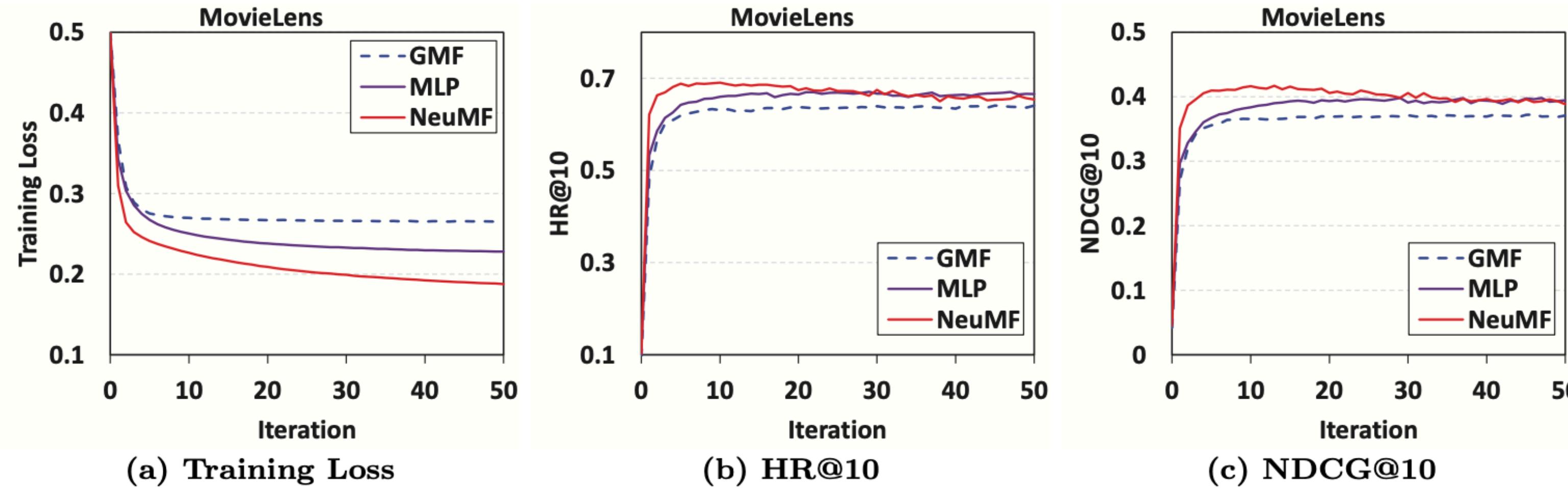


Figure 6: Training loss and recommendation performance of NCF methods *w.r.t.* the number of iterations on MovieLens (factors=8).

- (1) iteration이 높을수록, loss는 감소하고, 성능은 더 좋아지는 경향을 보인다.
- (2) 가장 효율적인 업데이트는 처음에 10 번의 iterations만을 진행하는 것이다.
- (3) 위 그래프에서, iteration=10에서 NeuMF의 모든 성능이 가장 좋음을 확인할 수 있다.

## 2. Neural Collaborative Filtering (2017)

1. Intro
2. Preliminaries
3. Neural Collaborative Filtering
- 4. Experiments**

4.1 NCF 방법은 기존의 최신 CF 방법보다 뛰어난가?

4.2 어떻게 제안된 최적화 틀(log loss with negative sampling)이 추천 시스템에서 잘 작동하는가?

4.3 더 깊은 심층 신경망을 선택하는 것은 interaction data의 학습에 도움이 되는가?

**Table 3: HR@10 of MLP with different layers.**

Factors	MLP-0	MLP-1	MLP-2	MLP-3	MLP-4
<b>MovieLens</b>					
<b>8</b>	0.452	0.628	0.655	0.671	<b>0.678</b>
<b>16</b>	0.454	0.663	0.674	0.684	<b>0.690</b>
<b>32</b>	0.453	0.682	0.687	0.692	<b>0.699</b>
<b>64</b>	0.453	0.687	0.696	0.702	<b>0.707</b>
<b>Pinterest</b>					
<b>8</b>	0.275	0.848	0.855	0.859	<b>0.862</b>
<b>16</b>	0.274	0.855	0.861	0.865	<b>0.867</b>
<b>32</b>	0.273	0.861	0.863	<b>0.868</b>	0.867
<b>64</b>	0.274	0.864	0.867	0.869	<b>0.873</b>

see, even for models with the same capability, stacking more layers are beneficial to performance. This result is highly encouraging, indicating the effectiveness of using deep models for collaborative recommendation. We attribute the im-

# References

- [1] <http://m.knnews.co.kr/mView.php?idxno=1325476&gubun=>
- [2] <https://www.ijert.org/research/recommender-systems-types-of-filtering-techniques-IJERTV3IS110197.pdf>
- [3] <https://www.kaggle.com/ibtesama/getting-started-with-a-movie-recommendation-system>
- [4] Sachi Nandan Mohanty (editor), Jyotir Moy Chatterjee (editor), Sarika Jain (editor), Ahmed A. Elngar (editor), Priya Gupta (editor) - Recommender System with Machine Learning and Artificial Intelligence
- [5] [http://www.kocca.kr/insight/vol05/vol05\\_04.pdf](http://www.kocca.kr/insight/vol05/vol05_04.pdf)
- [6] He, Xiangnan, et al. "Neural collaborative filtering." Proceedings of the 26th international conference on world wide web. 2017.
- [7] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." *Computer* 42.8 (2009): 30-37.
- [8] <https://lamttic.github.io/2020/03/20/01.html>
- [9] [https://greeksharifa.github.io/machine\\_learning/2019/12/17/Recommendation-System/](https://greeksharifa.github.io/machine_learning/2019/12/17/Recommendation-System/)