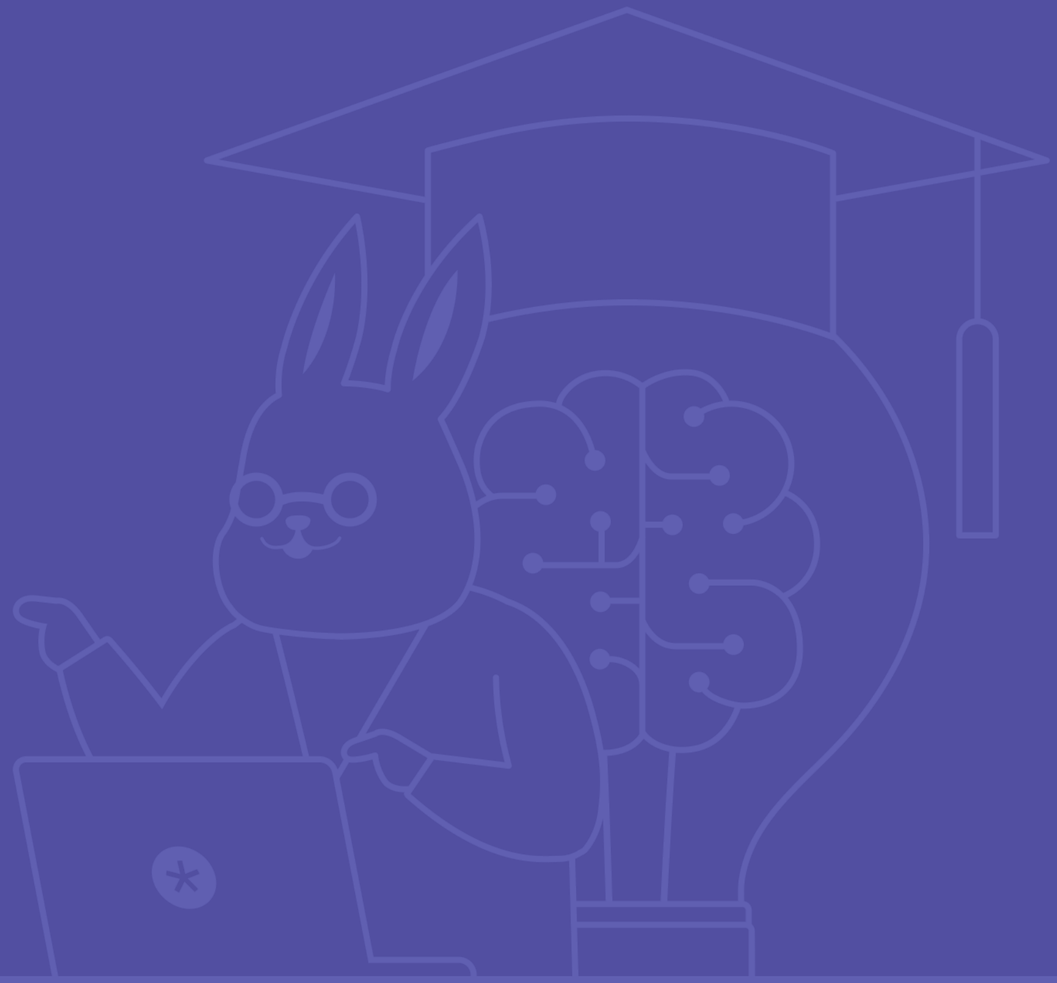
2020 AI College

9장 추천 알고리즘

정민수 강사



Confidential all right reserved

/* elice */

Contents

Target

추천 시스템의 필요성을 이해한다.

추천 시스템이 왜 필요하고 어떤 효과를 주는지 이해한다.

Content-based Recommendation을 이해한다.

Content-based Recommendation 기법을 이해한다.

Collaborative Filtering을 이해한다.

Collaborative Filtering 기법을 이해한다.

Latent Factor Model을 이해한다.

Latent Factor Model 기법을 이해한다.

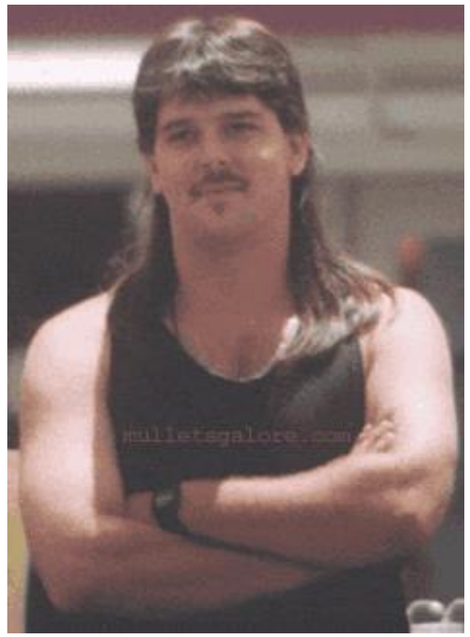
01

추천 알고리즘의 필요성

Confidential all right reserved

01 추천 시스템의 필요성

추천 시스템 예제



Customer X
- Metallica CD 구매
- Megadeth CD 구매

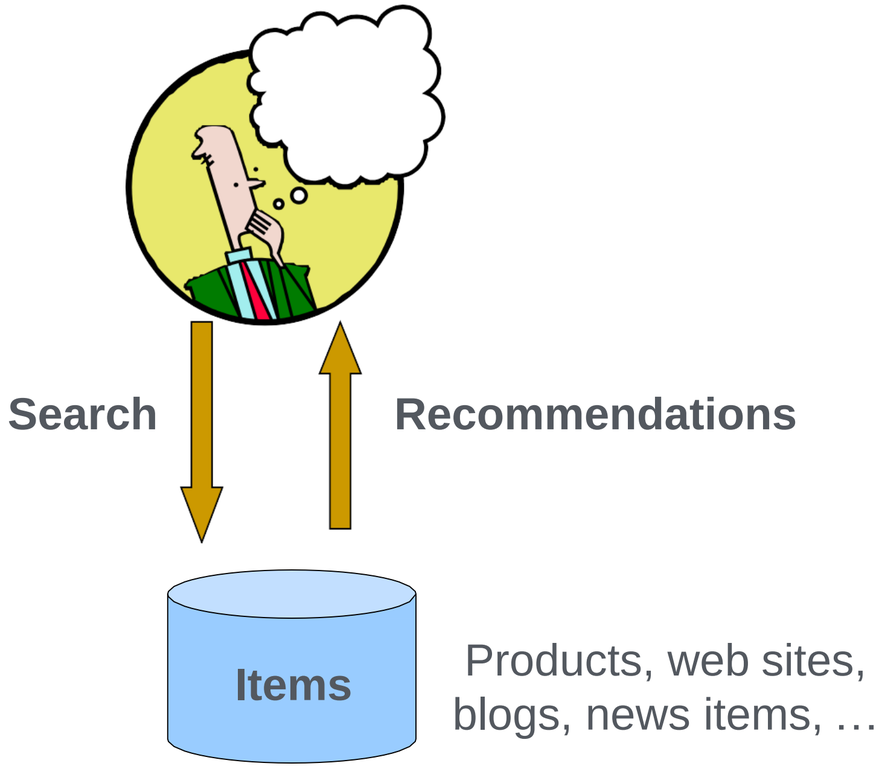
Customer Y
- Metallica를 찾아봄
- 추천 시스템이 Customer X를 참고해서 Megadeth를 추천해줌

/* elice */

01 추천 시스템의 필요성

추천 시스템이란?

- 사용자가 검색하는 검색어에 따라 적절한 상품을 추천해주는 시스템



Products, web sites,
blogs, news items, …

Examples:

amazon.com
PANDORA
StumbleUpon
del.icio.us
NETFLIX
movielens
helping you find the *right* movies
last·fm
the social music revolution
Google News
YouTube
XBOX LIVE

01 추천 시스템의 필요성

Offline과 Online 추천의 차이

- Offline 추천
  - Offline 매장에서는 인기 있는 상품을 추천함

- Online 추천
  - Online에서는 거의 cost 없이 상품에 대한 정보 추천이 가능
  - 인기가 없는 item에 대한 추천이 가능해짐

- 인기가 없더라도 사용자의 취향에 맞춰 상품을 추천한다면 매출을 높일 수 있음
  - Into Thin Air (1998)라는 책이 Touching the Void (1988)라는 책을 bestseller로 만든 유명한 사례가 있음

01 추천 시스템의 필요성

추천의 종류

1. Editorial and hand curated
- 여행에 필수적인 상품 리스트 등

2. 간단한 통계를 통한 추천
- Top 10, 인기 순 추천, 최근 출시된 상품 등

3. 사용자 맞춤 추천
- Youtube, Netflix 등의 사용자 맞춤 추천 시스템

Formal Model

- 이번 강의 내 사용할 추천 모델은 다음과 같이 정의된다.

- X : set of Customers
- S : set of Items

- Utility function $u: X \times S \to R$
- R : set of ratings
- R은 totally ordered set임을 가정
- Rating 예시 : 별점 (0 ~ 5), 0부터 1 사이의 실수값

01 추천 시스템의 필요성

Utility Matrix 예시

	Avatar	Closer	Matrix	Terminator
Alice	1		0.2	
Bob		0.5		0.3
Carol	0.2		1	
David				0.4

/* elice */

01 추천 시스템의 필요성

Key Problems

1. Utility Matrix 구성을 위해 rating을 모으는 방법

- Utility Matrix 구성을 위해 rating data를 수집하는 방법이 필요함.

2. 모르는 rating을 다른 정보를 통해 추론하는 방법

- 사용자가 흥미로워 하는 아이템을 알기 위해 Utility Matrix 내 비어 있는 부분 중 높은 Rating이 어디에 있는지 알아야 함.

3. 추론한 rating에 대한 평가 방법

- 추천 방법이 성공했는지에 대한 평가가 필요함.
- 추론한 rating을 성공적으로 추론했는지를 평가할 방법이 필요함.

01 추천 시스템의 필요성

데이터 수집 (rating 수집 방법)

- **Explicit**
  - 영화 추천상품에 대한 rating을 사용자에게 직접 요청
  - 사용자 입장에서 귀찮기 때문에 제대로 수집이 안됨

- **Implicit**
  - 사용자의 action을 통해 rating을 추론
  - ex) 특정 상품을 구매를 했다는 것은 해당 상품에 대한 rating이 높은 것임
  - 하지만 상품을 구매하지 않는다고 해서 low rating인 것은 아님

모르는 rating 추론 방법

- 일반적으로 Utility Matrix는 sparse함
  - 대부분의 사람들이 상품에 대한 rating을 하지 않기 때문

- Cold Start Problem
  - 새로운 상품은 rating이 아예 없음
  - 새로운 사용자는 이용 기록이 아예 없음

- 아래 접근 방법들을 통해 모르는 rating에 대한 추론이 가능
  - Content-based
  - Collaborative
  - Latent Factor Model

02

Content-based Recommendation

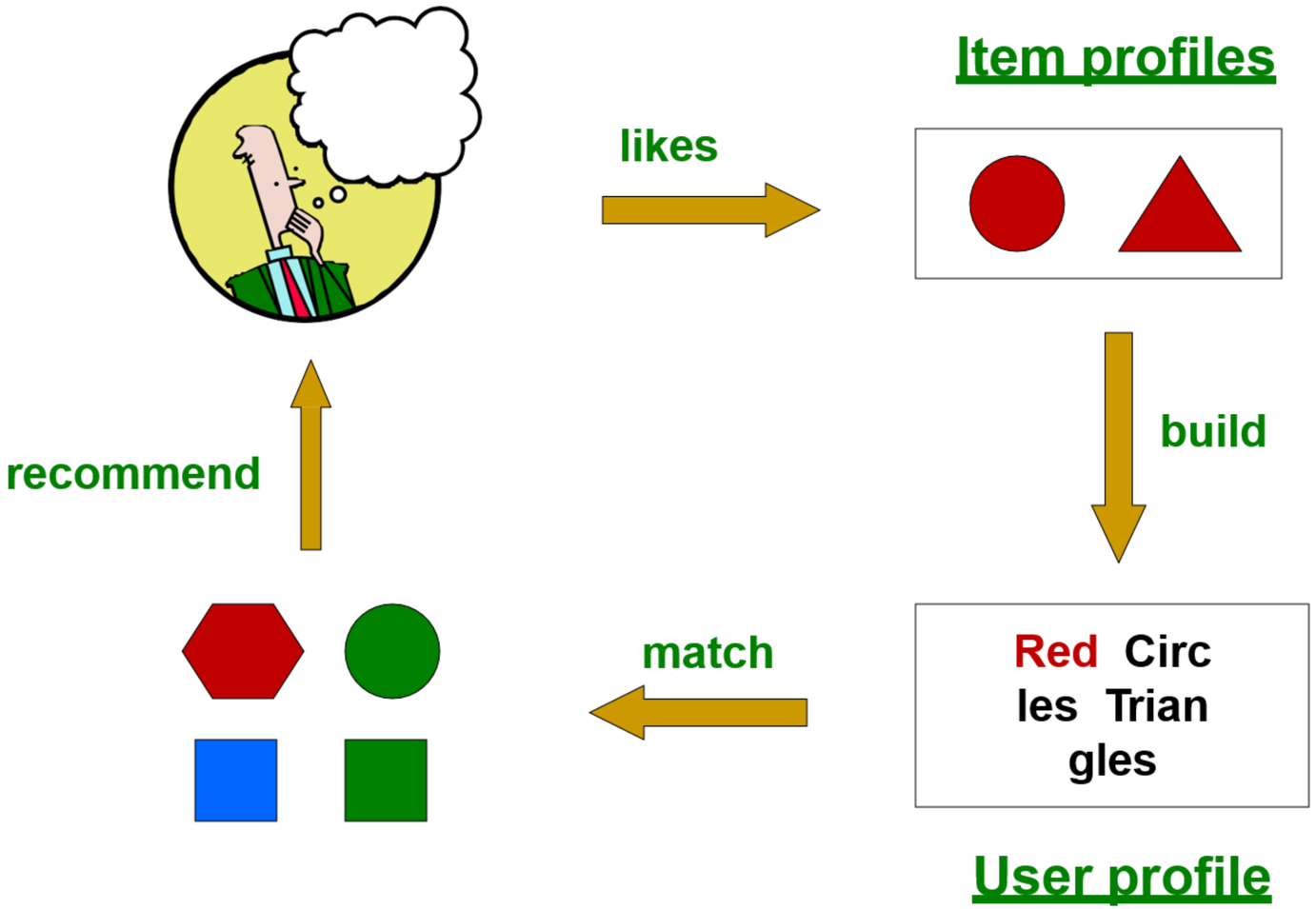
Confidential all right reserved

Content-based Recommendation이란?

- 핵심 idea
  - 사용자가 이전에 높게 rating한 상품과 유사한 상품을 추천함 (상품의 특성을 분석해서 유사 상품을 추천)

- 예시
  - 영화 추천
    - 같은 장르, 배우, 감독의 영화 추천
  - 웹사이트, 블로그, 뉴스 추천
    - 비슷한 내용의 사이트, 기사 추천

Content-based Recommendation 시스템



Item Profiles

- 각 item에 대해 item profile을 구성

- Profile은 feature의 set으로 구성됨
  - 예시
    - 영화 : 배우, 제목, 감독
    - 텍스트 : 문서 내 중요한 단어들의 set

- 중요한 feature 고르는 방법
  - 텍스트의 경우 TF-IDF 기법이 많이 사용됨

(심화 자료) TF-IDF

$$TF_{ij} = \frac{f_{ij}}{\max_k f_{kj}} \quad IDF_i = \log \frac{N}{n_i}$$

Note: we normalize TF to discount for “longer” documents

- f_{ij} = frequency of term (feature) *i* in doc (item) *j*
- n_i = number of docs that mention term *i*
- *N* = total number of docs

- TF-IDF score $w_{ij} = TF_{ij} \times IDF_i$
- **Doc profile**은 가장 높은 TF-IDF 스코어를 가진 단어들로 구성됨

User Profiles

- 각 사용자에 대한 User profile을 구성
- 사용자가 rating한 item들의 profile에 대한 weighted average로 계산

Prediction heuristic

- user profile x와 item profile i가 있을 때 예상되는 rating:

$$u(\boldsymbol{x}, \boldsymbol{i}) = \cos(\boldsymbol{x}, \boldsymbol{i}) = \frac{\boldsymbol{x} \cdot \boldsymbol{i}}{||\boldsymbol{x}|| \cdot ||\boldsymbol{i}||}$$

02 Content-based Recommendation

장점

- 다른 사용자의 profile이 필요 없음
  - cold start problem과 sparsity problem이 없음
- unique한 취향을 지닌 사용자에게 추천이 가능
- 유명하지 않은 새로운 item을 추천해줄 수 있음
  - first-rater problem이 없음
- 설명 가능함
  - 추천된 item이 어떤 이유로 추천되었는지 설명이 가능함

단점

- 적절한 feature를 찾기가 힘듦
- 새로운 사용자에 대한 추천이 불가능함
  - user profile 생성이 불가능
- Overspecialization 문제
  - 사용자의 content profile 외의 새로운 item을 추천해줄 수 없음
    - 추출된 feature 기준으로 유사하지 않을 수 있으나 무조건 사용자가 선호하지 않는다고 할 수 없음
    - 사용자는 가끔 아예 새로운 유형의 item을 추천 받기를 원함
  - 다른 사용자의 rating을 사용할 수 없음
    - item에 대한 다른 사용자의 rating에 따라 상품의 quality 평가가 가능하나 이를 활용할 수 없음

03

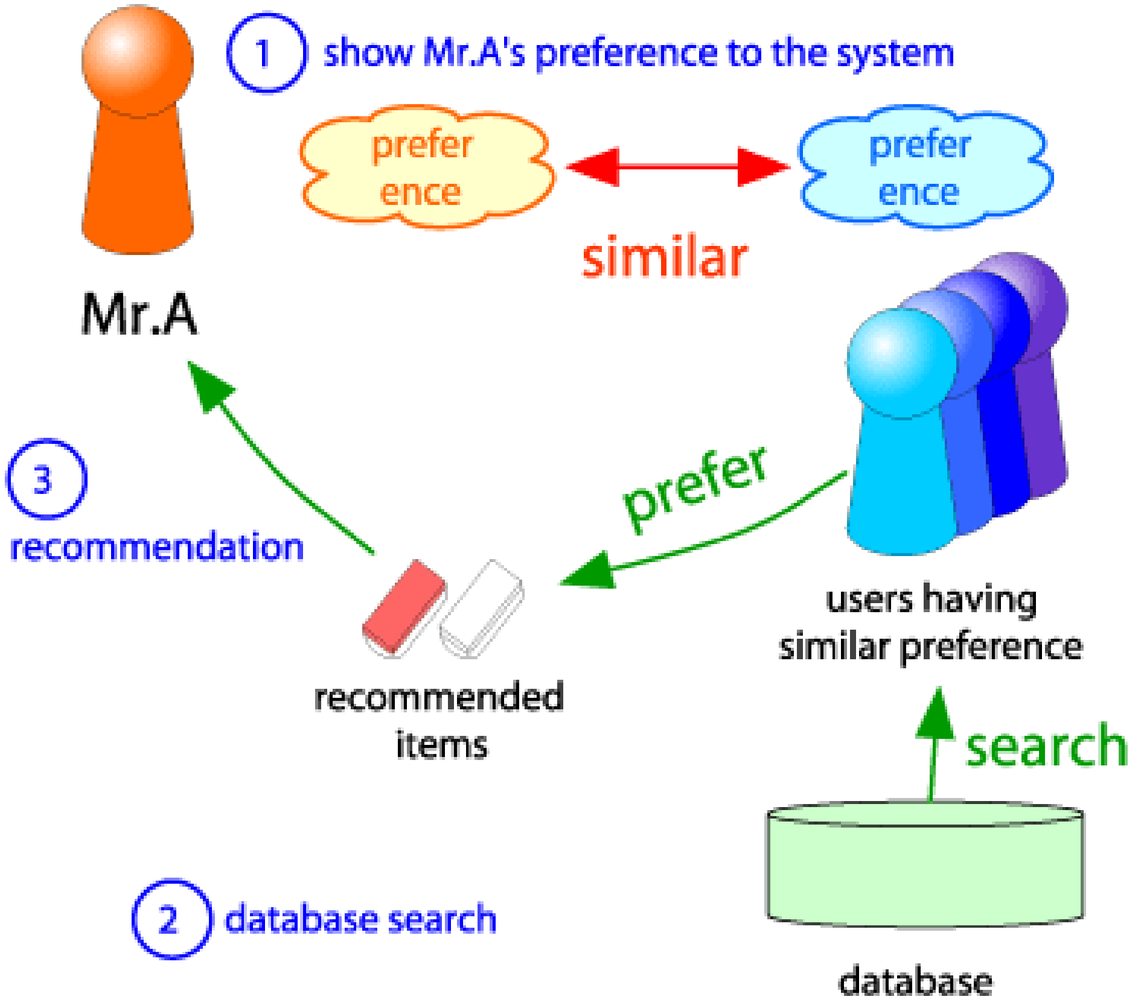
Collaborative Filtering

Confidential all right reserved

03 Collaborative Filtering

User-user Collaborative Filtering

- 사용자 x가 있다고 할 때
- item들에 대한 rating 패턴이 사용자 x와 유사한 다른 N명의 사용자를 찾아보자
- N명의 유사한 사용자들의 rating을 기반으로 x가 rating하지 않은 item에 대한 rating을 추측한다.



/* elice */

유사 사용자 찾기

- r_x : user x의 각 item에 대한 rating vector
- Jaccard similarity measure
  - Problem : rating value가 반영되지 않음
- Cosine similarity measure

$$\text{sim}(\boldsymbol{x}, \boldsymbol{y}) = \cos(\boldsymbol{r_x}, \boldsymbol{r_y}) = \frac{r_x \cdot r_y}{||r_x|| \cdot ||r_y||}$$

  - Problem : low rating에 대한 penalty가 작음
- Pearson correlation coefficient

$$sim(x, y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})(r_{ys} - \overline{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \overline{r_x})^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \overline{r_y})^2}}$$

  - S_{xy} : user x와 y 모두에게 rating 된 item
  - $\overline{r_x}$: average rating of r_x

$r_x = [*, \_, \_, *, ***]$
$r_y = [*, \_, **, **, \_]$

r_x, r_y as sets:
$r_x = \{1, 4, 5\}$
$r_y = \{1, 3, 4\}$

r_x, r_y as points:
$r_x = \{1, 0, 0, 1, 3\}$
$r_y = \{1, 0, 2, 2, 0\}$

/* elice */

Similarity Metric

- 목표 : $sim(A, B) > sim(A, C)$
- Jaccard similarity: 1/5 < 2/4
- Cosine similarity: 0.38 > 0.322
  - Problem: low rating에 대한 penalty가 작음

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

/* elice */

Similarity Metric

- 목표 : $sim(A, B) > sim(A, C)$
- Jaccard similarity: 1/5 < 2/4
- Cosine similarity: 0.38 > 0.322
  - Problem: low rating에 대한 penalty가 작음
  - Solution: 각 row의 평균을 뺀다

sim (A,B) vs. (A,C):
0.092 > -0.559

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	$2/3$			$5/3$	$-7/3$		
B	$1/3$	$1/3$	$-2/3$				
C				$-5/3$	$1/3$	$4/3$	
D		0					0

/* elice */

Rating Predictions

- item i에 대한 사용자 x의 r_x 계산 방법

$$r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$$

$$r_{xi} = \frac{\sum_{y \in N} s_{xy} \cdot r_{yi}}{\sum_{y \in N} s_{xy}} \qquad s_{xy} = sim(x, y)$$

- r_x : user x의 각 item에 대한 rating vector
- N : x와 rating 패턴이 가장 비슷한 k명의 사용자 그룹

- 다른 계산법을 적용해도 된다.

Item-item Collaborative Filtering

- item i에 대해서 사용자 x에 의해 rating 된 item 중 다른 유사한 item들을 찾는다.
- 찾은 유사한 item에 대한 rating을 이용하여 사용자 x의 item i에 대한 rating을 추측
- User-user 모델에서 사용한 prediction function을 유사하게 적용

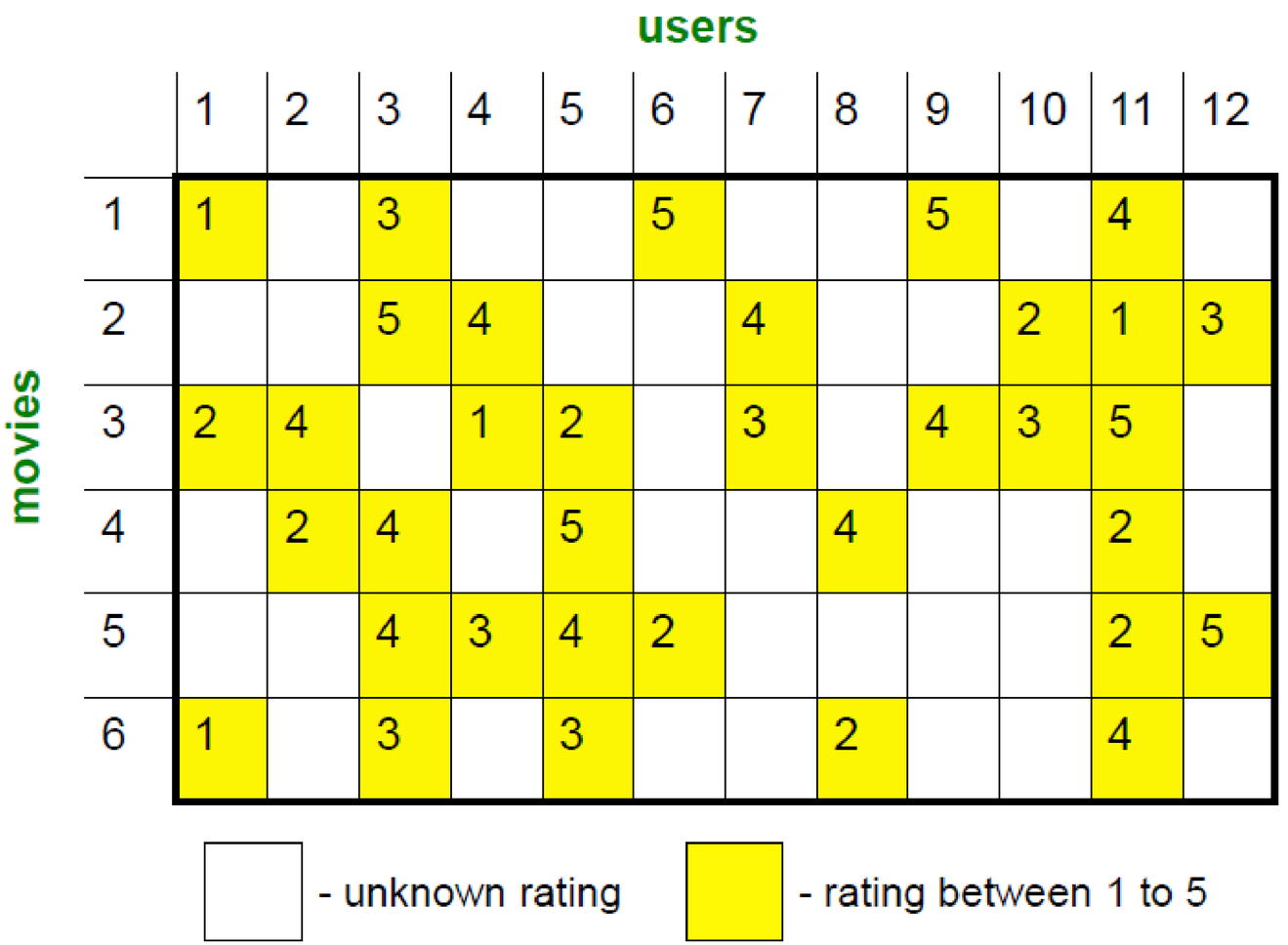
$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

s_{ij}... similarity of items ***i*** and ***j***
r_{xj}...rating of user ***x*** on item ***j***
$N(i;x)$... set items rated by ***x*** similar to ***i***

/* elice */

Item-item CF 예시

- |N| = 2라고 가정



/* elice */

Item-item CF 예시

- |N| = 2라고 가정

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12
1	1		3		?	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

- estimate rating of movie 1 by user 5

Item-item CF 예시

- |N| = 2라고 가정

Similarity computation:

1) Subtract mean rating m_i from each movie *i*
   $m_1 = (1+3+5+5+4)/5 = 3.6$
   row 1: *[-2.6, 0, -0.6, 0, 0, 1.4, 0, 0, 1.4, 0, 0.4, 0]*
2) Compute cosine similarities between rows

users

movies	1	2	3	4	**5**	6	7	8	9	10	11	12	**sim(1,m)**
1	1		3		?	5			5		4		**1.00**
2			5	4			4			2	1	3	**-0.18**
3	2	4		1	2		3		4	3	5		**0.41**
4		2	4		5			4			2		**-0.10**
5			4	3	4	2					2	5	**-0.31**
6	1		3		3			2			4		**0.59**

Neighbor selection:
Identify movies similar to movie **1**, **rated by user 5**

/* elice */

Item-item CF 예시

- |N| = 2라고 가정

Compute similarity weights:
$s_{1,3}$=0.41, $s_{1,6}$=0.59

users

movies	1	2	3	4	5	6	7	8	9	10	11	12	sim(1,m)
1	1		3		?	5			5		4		1.00
2			5	4			4			2	1	3	-0.18
3	2	4		1	2		3		4	3	5		0.41
4		2	4		5			4			2		-0.10
5			4	3	4	2					2	5	-0.31
6	1		3		3			2			4		0.59

/* elice */

Item-item CF 예시

- |N| = 2라고 가정

Predict by taking weighted average:

$r_{1.5}$ = (0.41*2 + 0.59*3) / (0.41+0.59) = 2.6

$$r_{ix} = \frac{\sum_{j \in N(i;x)} s_{ij} \cdot r_{jx}}{\sum s_{ij}}$$

users

movies	1	2	3	4	5	6	7	8	9	10	11	12
1	1		3		2.6	5			5		4	
2			5	4			4			2	1	3
3	2	4		1	2		3		4	3	5	
4		2	4		5			4			2	
5			4	3	4	2					2	5
6	1		3		3			2			4	

Common Practice

- 일반적으로 rating을 추측하는 공식은 다음과 같음

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N(i;x)} s_{ij} \cdot (r_{xj} - b_{xj})}{\sum_{j \in N(i;x)} s_{ij}}$$

Before:

$$r_{xi} = \frac{\sum_{j \in N(i;x)} s_{ij} r_{xj}}{\sum_{j \in N(i;x)} s_{ij}}$$

baseline estimate for r_{xi}

$$b_{xi} = \mu + b_x + b_i$$

- μ = overall mean movie rating
- b_x = rating deviation of user x
  = (*avg. rating of user x*) – μ
- b_i = rating deviation of movie i
  = (*avg. rating of movie i*) – μ

- s_{ij} : item i와 j의 similarity
- $N(i;x)$: k nearest neighbor
  - x에 의해서 rating된 item 중 item i와 가장 유사한 item들

/* elice */

Item-item vs User-user

- 일반적으로 item-item이 user-user보다 성능이 좋다고 보고됨.
- user는 복잡하고 다양한 취향을 갖고 있으나 item은 그보다 단순하기 때문

Complexity

- k개의 Nearest neighbor를 찾는 과정이 가장 cost가 큼: O(|X|)
  - X : 전체 사용자의 수 (item의 경우 item의 수)
- runtime에 사용하기엔 무거움
  - pre-compute할 필요가 있음
- computation cost를 줄이는 여러가지 방법들
  - Near neighbor search in high dimensions (LSH)
  - Clustering
  - Dimensionality reduction 후 찾기

03 Collaborative Filtering

장단점

- 장점
  - 어떤 종류의 item에 대해서도 동작
- 단점
  - Cold Start Problem
    - 충분한 user와 rating이 필요
    - Utility matrix가 어느정도 구성되지 전까지는 동작하기 힘듦
  - Sparsity
    - user/rating matrix(utility matrix)가 sparse
    - 같은 item에 대해 ratin한 사용자를 많이 찾기 힘듦
  - First rater
    - 이전에 rating이 없었던 신규 item 혹은 user에 대한 추천이 불가능
  - Popularity bias
    - 특이 취향이 있는 사용자에 대한 추천이 힘듦
    - 일반적으로 인기 있는 item들에 대한 추천이 주로 가능

Hybrid Methods

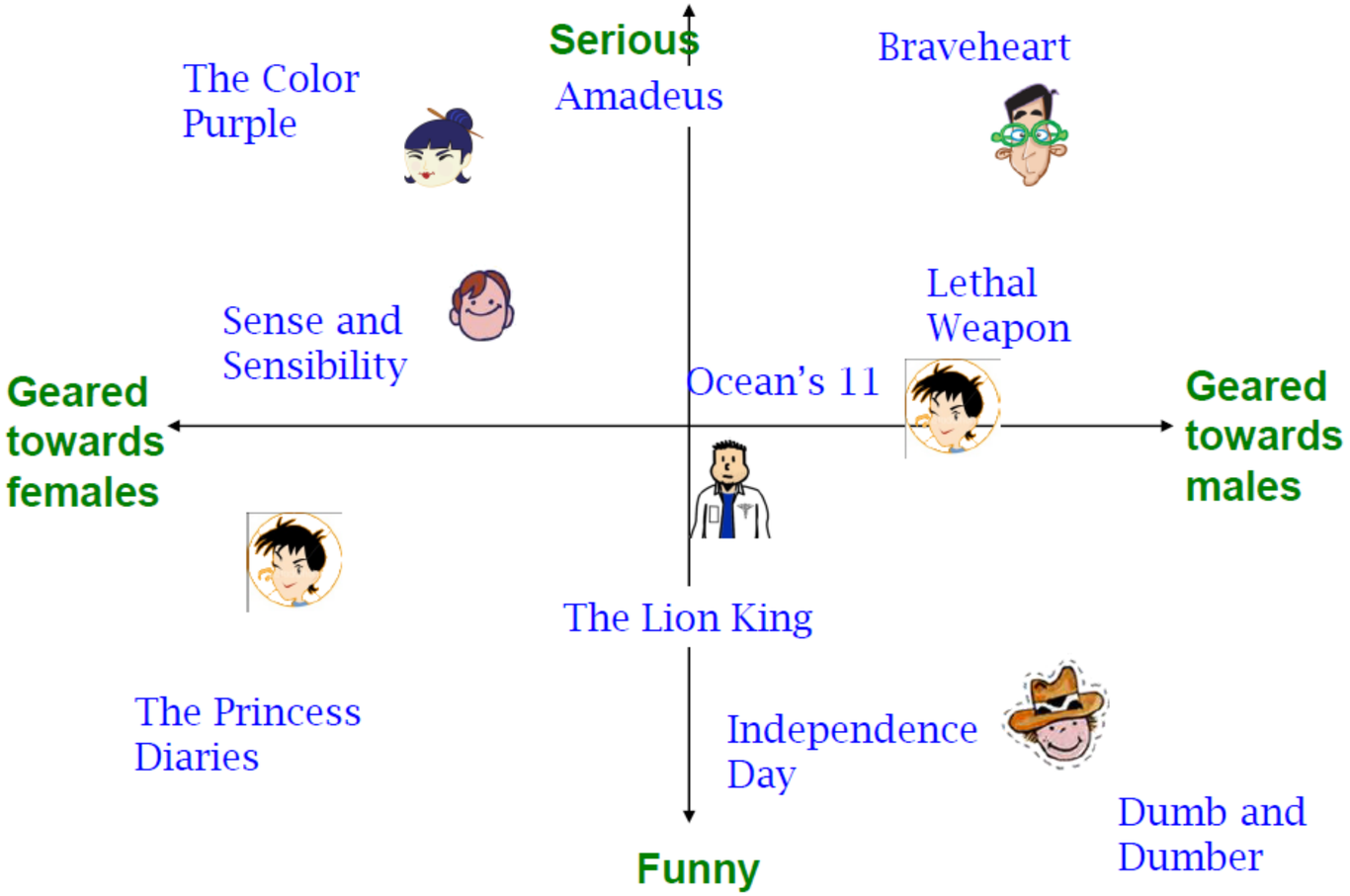
- 2개 이상의 다른 recommend system을 구성하고 예측을 결합
- content-based method에 collaborative filtering을 추가하여 사용
  - 서로의 단점을 보완할 수 있음
  - 신규 item에 대한 문제를 item profile을 통해 해결
  - 신규 user에 대한 문제는 일반적인 popular item을 추천해주는 방식으로 진행

04

Latent Factor Model

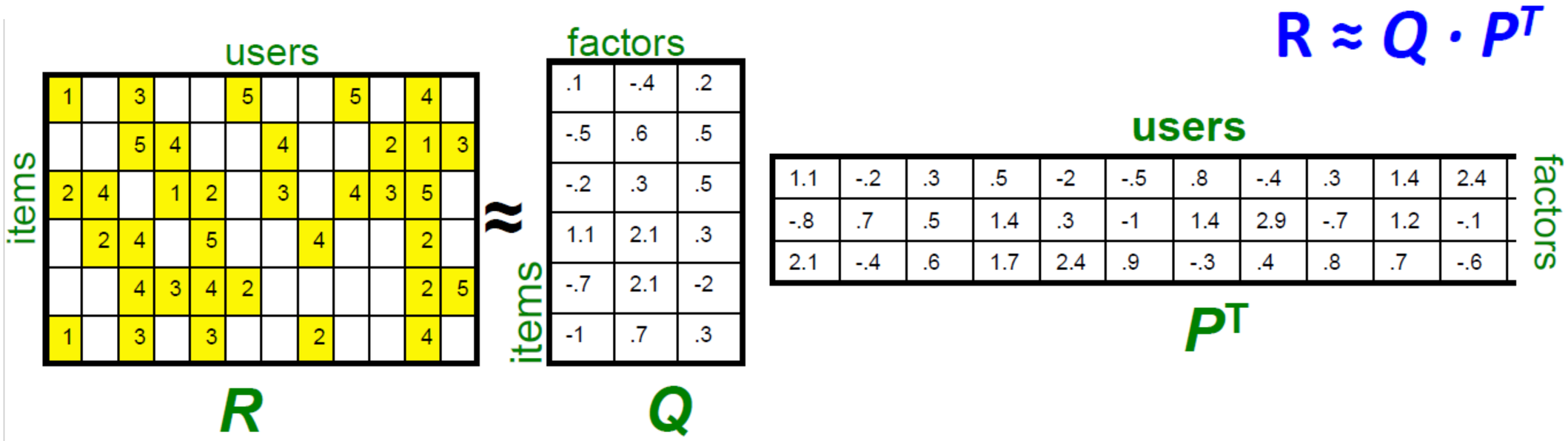
Confidential all right reserved

Latent Factor Model



Latent Factor Model

- SVD on Utility Matrix!



1		3			5			5		4
		5	4			4			2	1	3
2	4		1	2		3		4	3	5
	2	4		5			4			2
		4	3	4	2					2	5
1		3		3			2			4	

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

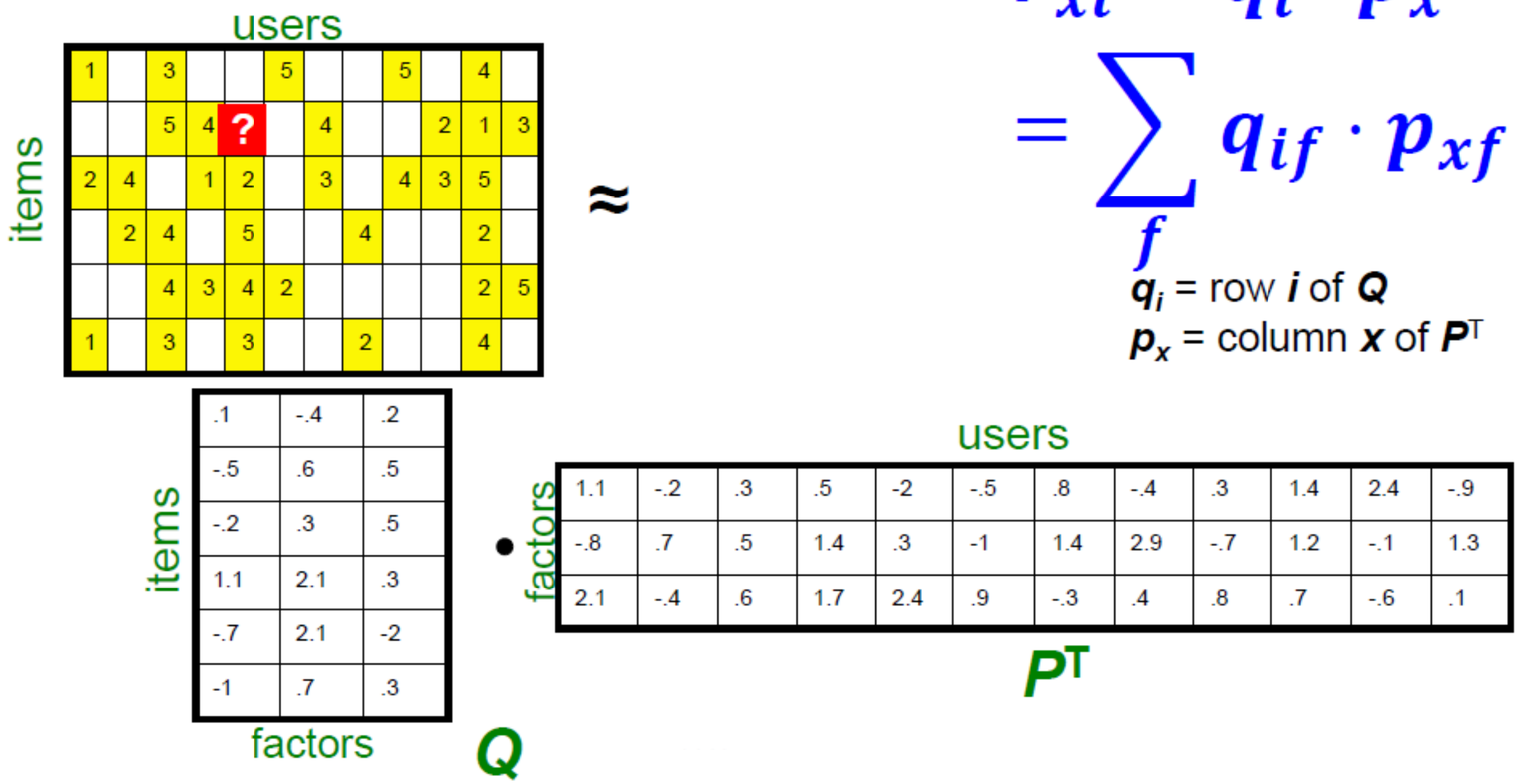
1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6

- Utility matrix R에 대해 Q와 P^{T}의 곱으로 표현이 가능하다고 하자
- R이 missing entry들을 가지고 있지만 우선은 무시하고 생각

/* elice */

Latent Factor Model

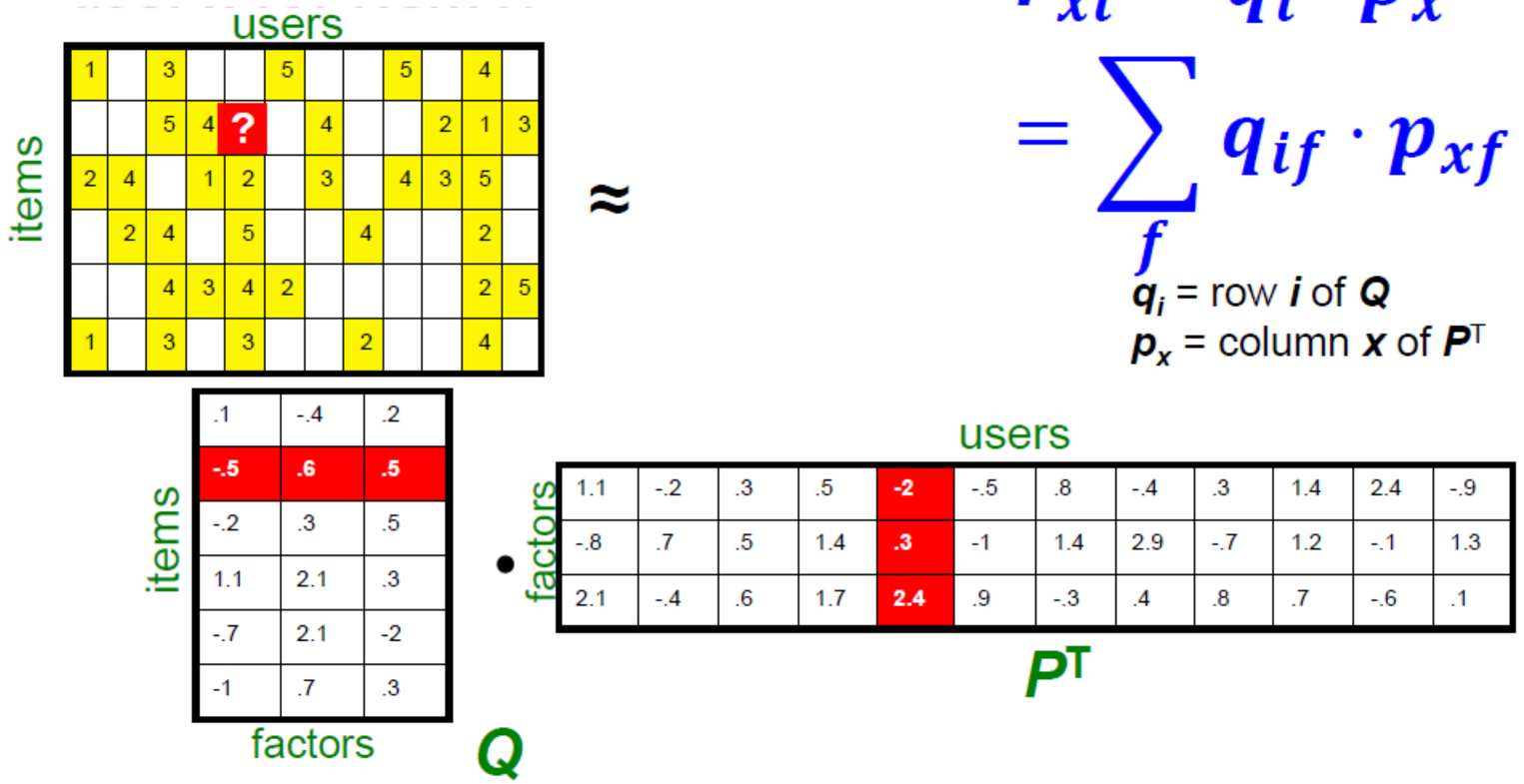
- 어떻게 item i에 대한 user x의 missing rating을 추론할 수 있을까?



/* elice */

Latent Factor Model

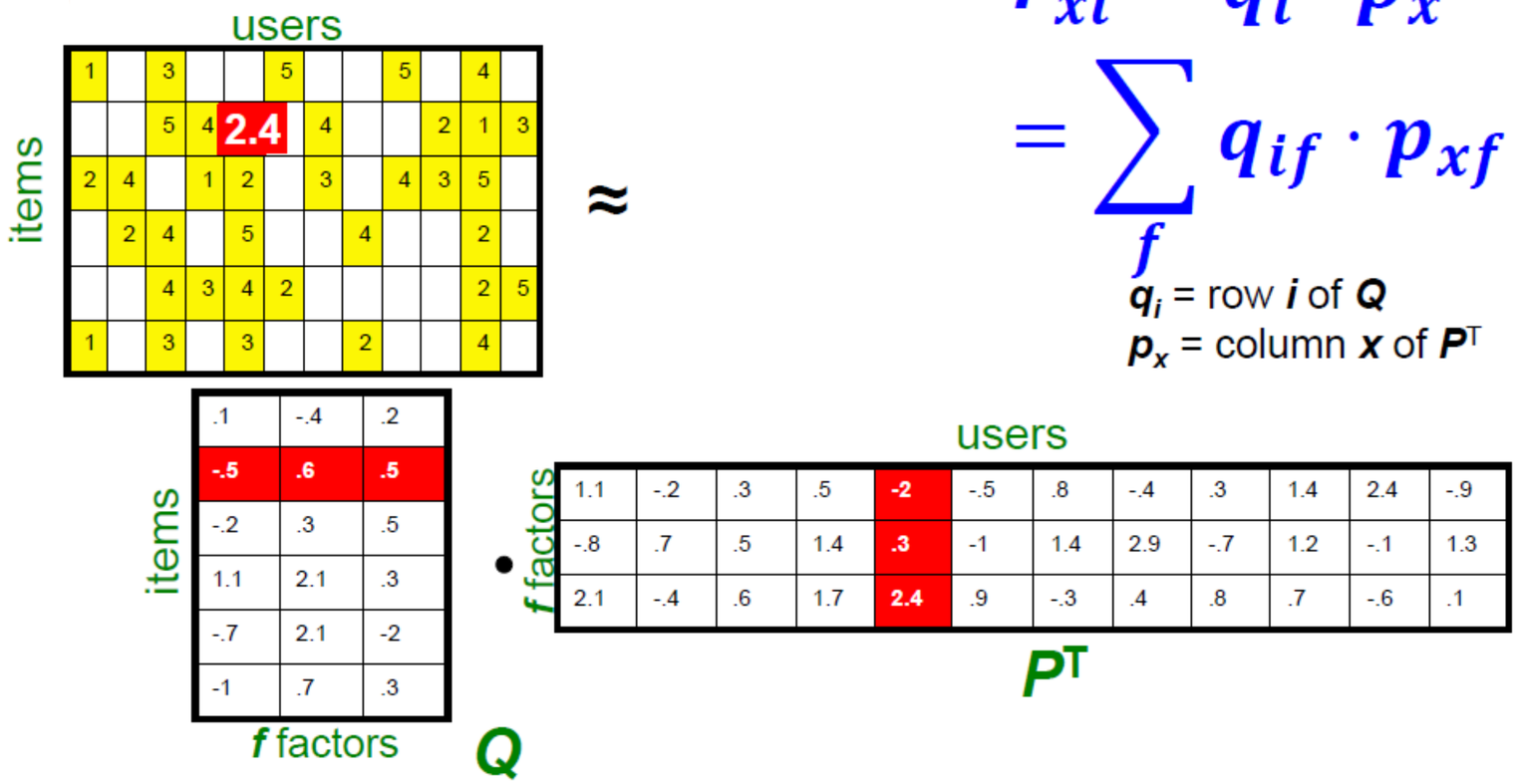
- 어떻게 item i에 대한 user x의 missing rating을 추론할 수 있을까?



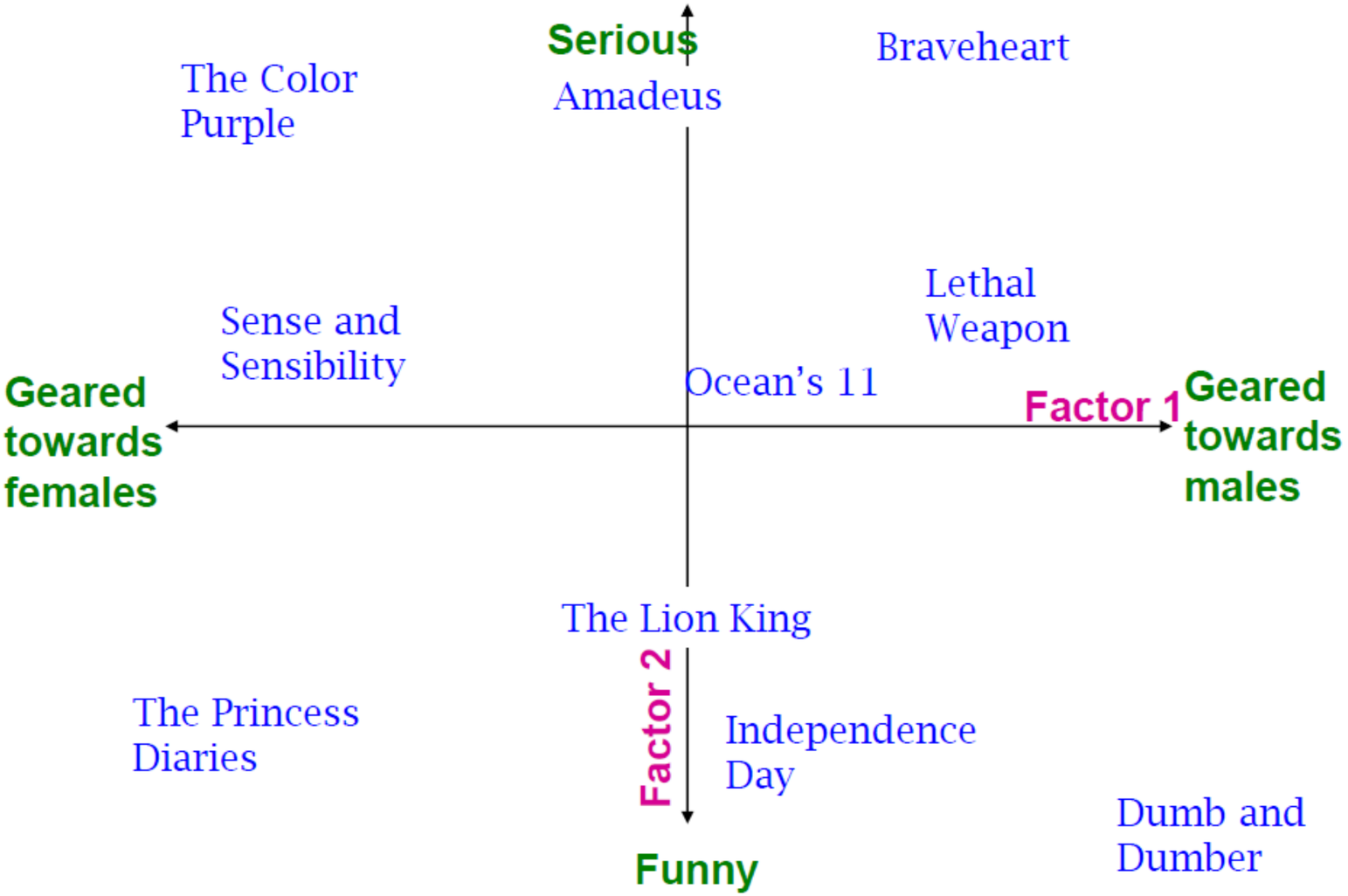
/* elice */

Latent Factor Model

- 어떻게 item i에 대한 user x의 missing rating을 추론할 수 있을까?

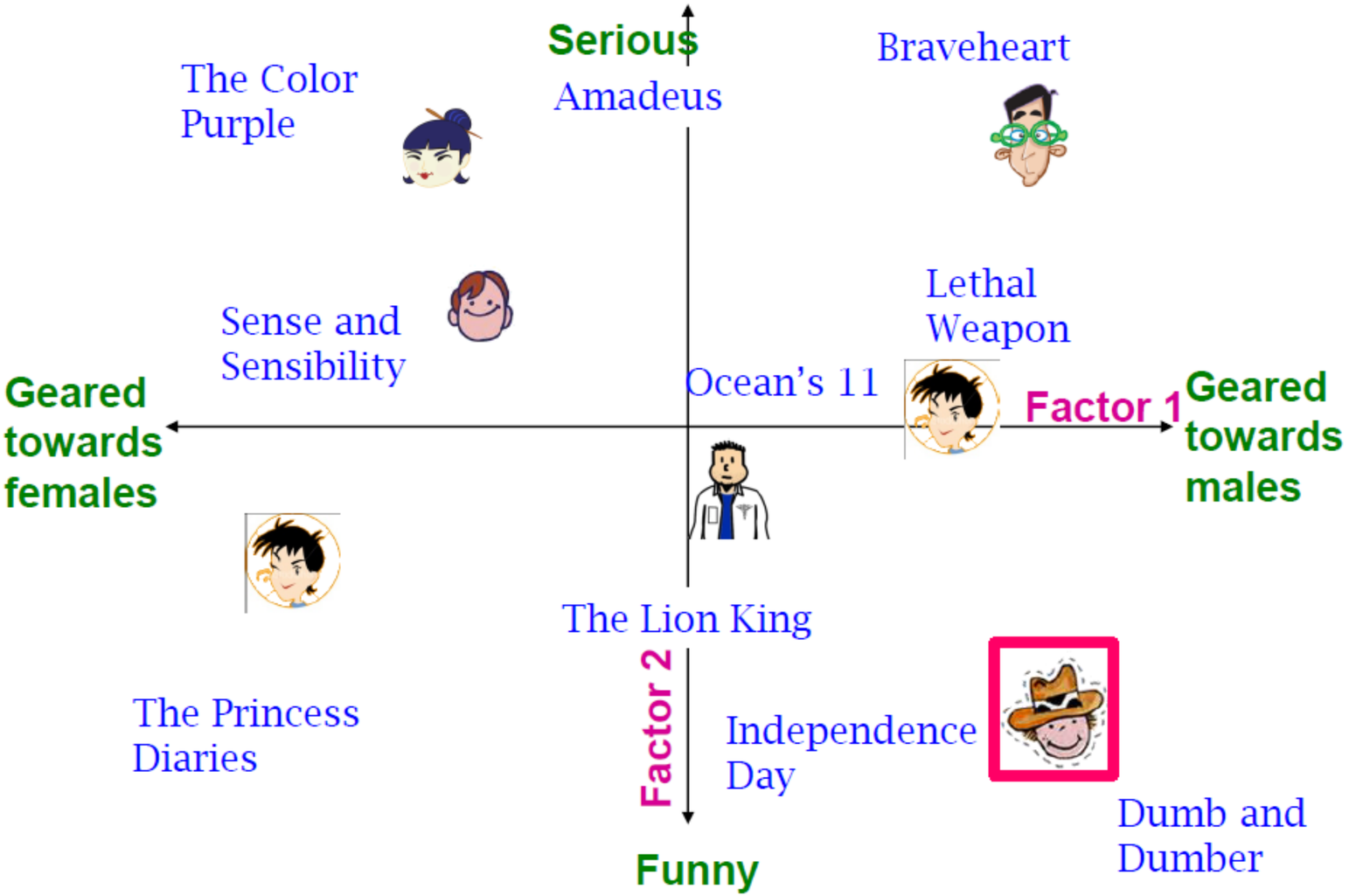


Latent Factor Model



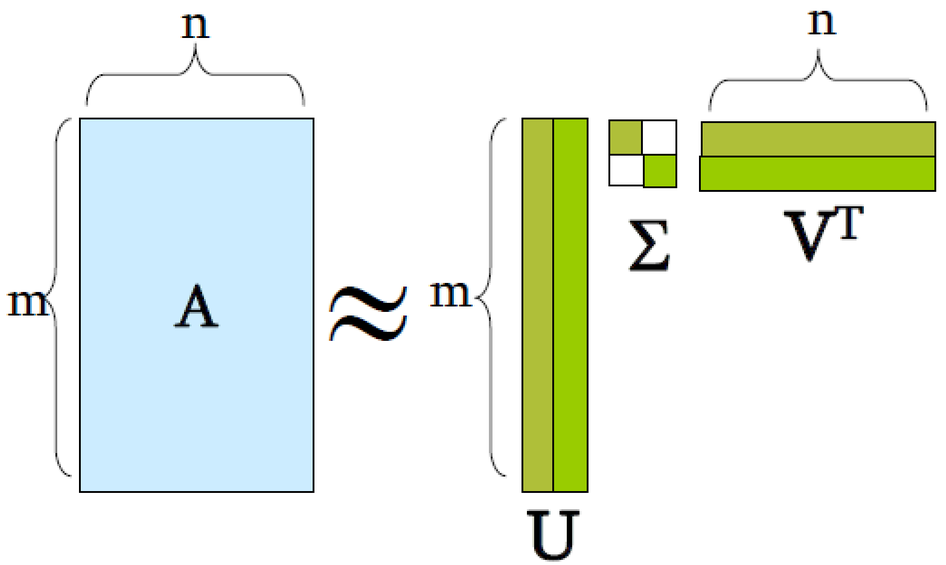
/* elice */

Latent Factor Model



SVD에 적용

- SVD
  - A : 입력된 data matrix
  - U : Left singular vectors
  - V : Right singular vectors
  - ∑ : Singular values



- 우리의 경우 $R \approx QP^{\mathrm{T}}$를 만족해야 함
- $A = R, Q = U, P^{T} = \Sigma V^{\mathrm{T}}$

SVD에 적용

- SVD는 SSE(Sum of Squared Error)를 최소화하기 때문에 최소의 reconstruction error를 제공하여 적합

$$\min_{U,V,\Sigma} \sum_{ij \in A} \left(A_{ij} - [U\Sigma V^{\mathrm{T}}]_{ij}\right)^2$$

- SSE와 RMSE는 단조 비례함

$$RMSE = \frac{1}{c}\sqrt{SSE}$$

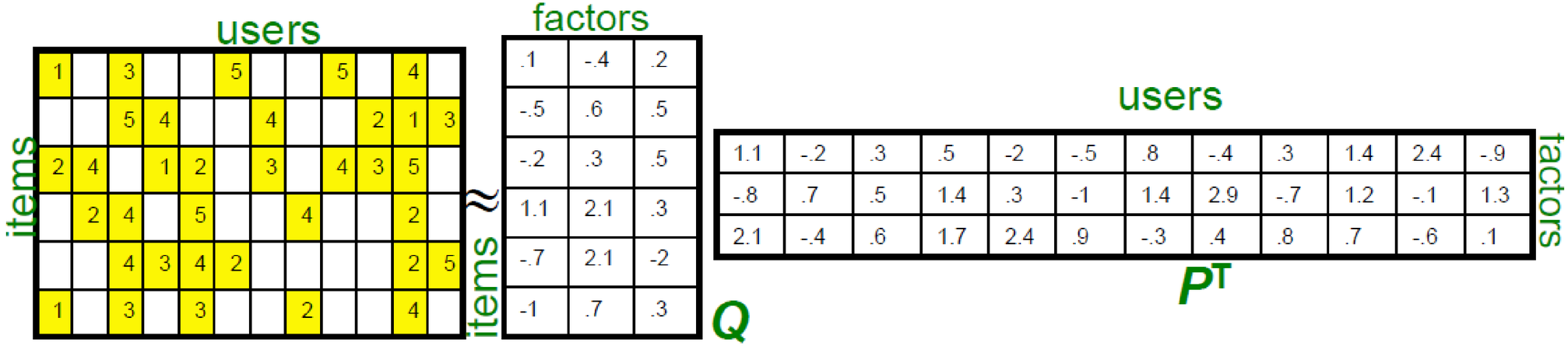
- 하지만 SVD는 missing data에 대한 고려가 없어 바로 적용하기엔 적합하지 않음!
  - error는 missing data를 0으로 가정하고 error를 계산함
  - Utility matrix R은 missing data를 많이 포함하고 있음

Matrix Factorization

- SVD는 missing data에 대한 고려가 전혀 되어있지 않음
- P와 Q를 찾기 위해 문제에 최적화된 기법을 사용

$$\min_{P,Q} \sum_{(i,x)\in R} \left(r_{xi} - q_i \cdot p_x\right)^2 \qquad \hat{r}_{xi} = q_i \cdot p_x$$

- 우리 문제에서는 P와 Q가 orthogonal하고 unit vector로 이루어질 필요가 없음
- P와 Q는 각각 user와 movie에 대한 latent space



/* elice */

05

평가 방법

Confidential all right reserved

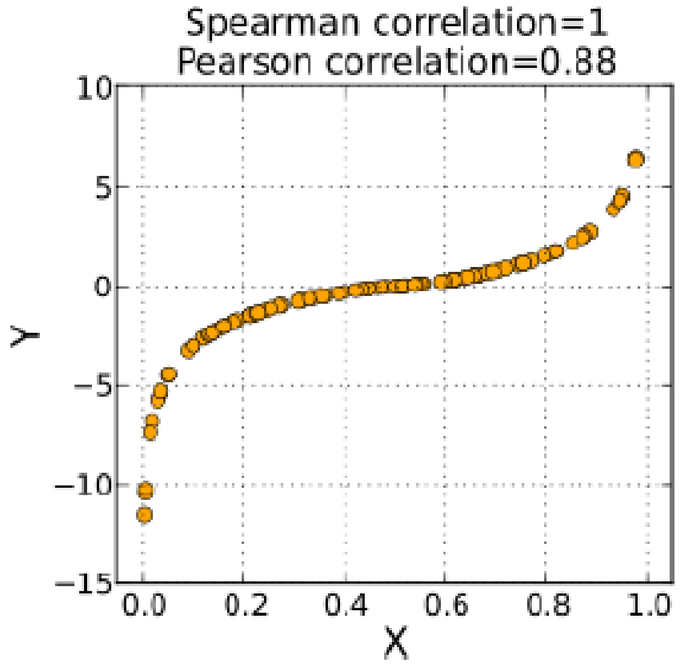
Prediction 평가

- 이미 알고있는 rating을 기반으로 prediction을 평가
- 평가 Metric
- RMSE (Root Mean Square Error)

$$\sqrt{\sum_{xi}\left(r_{xi}-r_{xi}^{*}\right)^{2}}$$

r_{xi} is predicted, r_{xi}^{*} is the true rating of x on i

- Precision at top 10
  - prediction value가 가장 높은 top 10에 대한 error로 평가
- Rank Correlation
  - system와 user의 ranking에 대한 Spearman's correlation
  - Spearman's correlation?
    - 각 값들의 rank에 대한 correlation



05 평가 방법

평가에 대해 주의할 점

- 단순히 accuracy에만 집중하는 것은 조심할 필요가 있음
  - 모델이 어떤 부분에서 잘하고 못하는지를 분석해서 판단해야 함

- 추천에서는 high rating만을 추천해주기 때문에 high rating에 대한 정확도를 평가할 필요가 있음
  - RMSE는 high rating과 row rating에 대한 평가가 같은 중요도로 평가되기 때문에 목적에 맞지 않게 평가하게 될 수 있음

Credit

/* elice */

코스 매니저
이해솔

콘텐츠 제작자
정민수

강사
정민수

감수자
이해솔

디자인

Contact

TEL
070-4633-2015

WEB
https://elice.io

E-MAIL
contact@elice.io