DA-Final-7) Recommender system

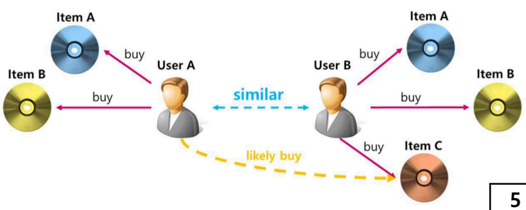
1. Recommender system introduction

1-1. Recommender system 정의

- Basic idea:
  - if two people A and B are similar to each other and A prefers an item X, then B is also likely to prefer X (need to measure user similarity).
  - if two items X and Y are similar to each other and A prefers an item X, then A is also likely to prefer Y (need to measure item similarity).
- How to measure the similarity?
  - Represent users and items as feature vectors and compute correlation coefficient or cosine similarity between them.
- How to represent users and items as feature vectors?
  - **Content-based approach (CB)**:
    - Use contents – domain specific, e.g.,
    - Movie domain – actor, genre, director, year, synopsis, etc.
    - Music domain – singer, genre, composer, lyrics.
  - **Collaborative filtering (CF)**:
    - Rating information – domain independent, e.g.,
    - Users' ratings on items (explicit feedback)
    - Users' purchase records or click records on items (implicit feedback)
- CF has shown commercial success and has been actively researched in the community.
- Recently, hybrid approaches using deep learning technologies has gained popularity, which is beyond the scope of this course.

1-2. Collaborative filtering (CF)

- Use the relationship between users and items



- Such relationships are represented as a rating matrix

Items

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

User가 Item에 내린 평점.

2. Collaborative filtering approaches

2-1. K nearest neighbor

2-1-(1). User-based K nearest neighbor

- User-based nearest neighbor collaborative filtering [Resnick 94]

target user

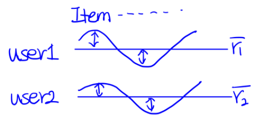
	Item 1	Item 2	Item 3	Item 4	Item 5
User A	5	3	4	4	?
User 1	3	1	2	3	3
User 2	4	3	4	3	5
User 3	3	3	1	5	4
User 4	1	5	5	2	1

계산!

- Pearson correlation between two users:

correlation coefficients

$$sim\ (u_1, u_2) = \frac{\sum_{i \in I^{\{1.2\}}} (r_{1i} - \overline{r_1})(r_{2i} - \overline{r_2}\ )}{\sqrt{\sum_{i \in I^{\{1,2\}}} (r_{1i} - \overline{r_1})^2} \sqrt{\sum_{i \in I^{\{1,2\}}} (r_{2i} - \overline{r_2})^2}}$$



⇒ Item에 따른 variation의 similarity

- $I^{\{x,y\}}$: A set of items, rated by both user x and user y
- r_{ij} : a rating on item j by user i
- $\overline{r_i}$: an average rating of user i

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

-cosine similarity X
└ 평균값에서 Item별 rating값 사이의 편차가 유사한 User끼리 → correlation

- $sim\ (u_1, u_2) = \frac{\sum_{i \in I^{\{1.2\}}} (r_{1i} - \overline{r_1})(r_{2i} - \overline{r_2}\ )}{\sqrt{\sum_{i \in I^{\{1,2\}}} (r_{1i} - \overline{r_1})^2} \sqrt{\sum_{i \in I^{\{1,2\}}} (r_{2i} - \overline{r_2})^2}}$
- $I^{\{x,y\}}$: A set of items, rated by both user x and user y
- r_{ij} : a rating on item j by user i
- $\overline{r_i}$: an average rating of user i

Example:

- $\overline{r_A} = \frac{5+3+4+4}{4} = 4$
- $\overline{r_1} = \frac{3+1+2+3+3}{5} = 2.4$
- $\overline{r_2} = \frac{4+3+4+3+5}{5} = 3.8$
- $\overline{r_3} = \frac{3+3+1+5+3}{5} = 3.2$
- $\overline{r_4} = \frac{1+5+5+2+1}{5} = 2.8$
- $sim\ (u_A, u_1) = \frac{(5-4)(3-2.4)+(3-4)(1-2.4)+\cdots}{\sqrt{(5-4)^2+\cdots}\sqrt{(3-2.4)^2+\cdots}} \approx 0.84$
- 1-NN, $sim\ (u_A, u_1) = 0.84$
- 2-NN, $sim\ (u_A, u_2) = 0.42$

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

Example:

- 1-NN, $sim\ (u_A, u_1) = 0.84$
- 2-NN, $sim\ (u_A, u_2) = 0.42$
- $\hat{r}_{A5} = \overline{r_A} + \frac{0.84 \cdot (3-2.4) + 0.42 \cdot (5-3.8)}{0.84+0.42} = 4.88$

2-NN; nearest 2 user의 similarity-weighted mean
of target item에 대한 rating의 편차
+ target user의 평균 점수

- Final prediction:

$$\hat{r}_{ui} = \overline{r_u} + \frac{\sum_{k \in N} sim\ (u,k) * (r_{ki} - \overline{r_k})}{\sum_{k \in N} sim\ (u,k)}$$

- where N is a k-nearest neighbor set

2-1-(2). Item-based K nearest neighbor

- Item-based nearest neighbor collaborative filtering [Sarwar 2001]

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

좋은 평점을 줄수록 확률↑

- Find kNN items that are similar to item 5.
- Cosine similarity $sim\left(I_i, I_j\right) = \frac{I_i \cdot I_j}{|I_i||I_j|}$
- Prediction: Take the ratings of the most similar items to predict on item 5.

2-1-(3). K nearest neighbor의 장단점

- Pros
    - Intuitive
    - No training required
    - Easy to explain to user
- Cons
    - Not scalable nor accurate
    - Especially bad for sparse matrix

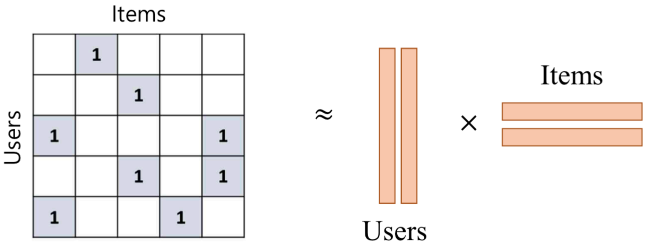
	Item 1	Item 2	Item 3	Item 4	Item 5
UserA	5	1	?	?	?
User 1		1			3
User 2		3		3	
User 3			5	5	
User 4	3		5		1

일반적인 table matrix는 sparse하다!

2-2. Matrix factorization

2-2-(1). Matrix factorization 개요

- MF(matrix factorization) factorizes the rating matrix into user and items matrices, which become the latent model that produces the ratings.
- What is latent model? (내재된)

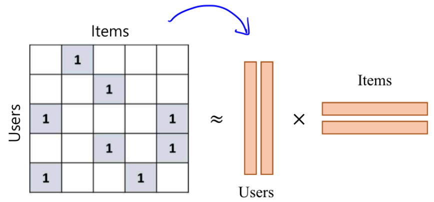


- Latent model is a hidden model which well describes phenomena or observations.

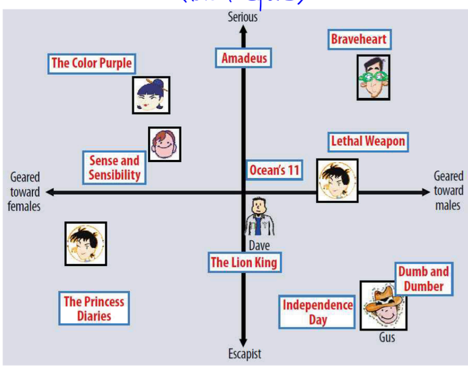
- Users and Items can be represented as vectors in the shared latent space
- Rating score is generated by dot product of user latent vector (p_i) and item latent vector (q_j)

$$\hat{r}_{ij} = p_i \cdot q_j$$

factorized



<latent space>



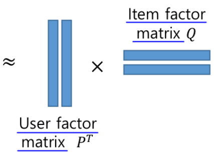
2-2-(2). Matrix factorization formal description

- Latent model:

Rating matrix

$$\begin{array}{cccc} & Avatar & The\ Matrix & Up \\ Alice & & & \\ Bob & & & \\ Charlie & & & \end{array} \left(\begin{array}{ccc} ? & 4 & 2 \\ 3 & 2 & ? \\ 5 & ? & 3 \end{array}\right) \approx$$

Original matrix R

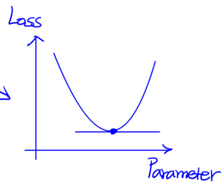


$$r_{ij} \approx \hat{r}_{ij} = [P^T Q]_{ij}$$

- Goal: Find P and Q which minimizes the error (RMSE)

$$\underset{P,Q}{\operatorname{argmin}} RMSE(R, P^T Q)$$

- RMSE:
  - $\sqrt{\frac{1}{|R|}\sum_{(i,j)\in R}\left(r_{ij} - \widehat{r_{ij}}\right)^2}$
- Minimizing RMSE is equal to minimizing unnormalized MSE
  - $\sqrt{\frac{1}{|R|}\sum_{(i,j)\in R}\left(r_{ij} - \widehat{r_{ij}}\right)^2} \Rightarrow \sum_{(i,j)\in R}\left(r_{ij} - [P^T Q]_{ij}\right)^2$
- The final objective function
  - ☆ $\underset{P,Q}{\operatorname{argmin}} \sum_{(i,j)\in R}\left(r_{ij} - [P^T Q]_{ij}\right)^2$ objective function
  - △ $\underset{P,Q}{\operatorname{argmin}} \|R - P^T Q\|_2^2 = \underset{P,Q}{\operatorname{argmin}} \operatorname{Tr}\left((R - P^T \cdot Q)\cdot(R - P^T \cdot Q)^T\right)$



- 2-norm
  - $\left\| \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix} \right\|^2 = x_1^2 + x_2^2 + x_3^2 + x_4^2$
- Trace
  - $Tr\left(\begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}\begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}^T\right) = Tr\left(\begin{bmatrix} x_1^2 + x_2^2 & x_1x_3 + x_2x_4 \\ x_3x_1 + x_4x_2 & x_3^2 + x_4^2 \end{bmatrix}\right) = x_1^2 + x_2^2 + x_3^2 + x_4^2$

2-2-(3). Matrix factorization에서의 Gradient descent

Gradient Descent (GD)

- First-order optimization algorithm to minimize an objective function:
  - $f(x)$: objective function to minimize in terms of x
- Start with a random x and take steps proportional to the negative of the gradient of the function at the current point
  - $x_{n+1} = x_n - \eta \nabla f(x_n)$
  - η: step size

gradient

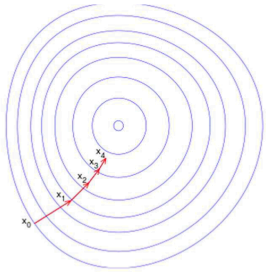


Figure. Illustration of gradient descent

- Objective function for MF
  - $\|R - P^T \cdot Q\|_2^2 = \mathrm{Tr}\left((R - P^T \cdot Q) \cdot (R - P^T \cdot Q)^T\right)$
- Derivative rule objective function
  - $\frac{\partial}{\partial X} Tr[(C + AXB)(C + AXB)^T] = 2A^T(C + AXB)B^T$
- Gradient for the objective function
  - $\frac{\partial f(P,Q)}{\partial P} = -2(R - P^T \cdot Q)Q^T$
  - $\frac{\partial f(P,Q)}{\partial Q} = -2P(R - P^T \cdot Q)$
- Updating rule for gradient descent
  - $x_{n+1} = x_n - \eta \nabla F(x_n)$
  - $P^T = P^T - \eta \frac{\partial f(P,Q)}{\partial P} = P^T + \eta'(R - P^T \cdot Q)Q^T$
  - $Q = Q - \eta \frac{\partial f(P,Q)}{\partial Q} = Q + \eta' P(R - P^T \cdot Q)$

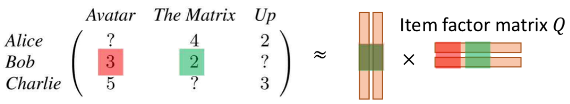
$Tr((R+(-I)P^TQ)(R+(-I)P^TQ)^T)$

$Tr((R+(-P^T)QI)(R+(-P^T)QI)^T)$

Dimensionality check:

- $R = m \times n$
- $P = k \times m$
- $Q = k \times n$
- $P^T = m \times k$
- $Q^T = n \times k$
- $R - P^T \cdot Q = m \times n$
- $(R - P^T \cdot Q)Q^T = m \times k$
- $P^T(R - P^T \cdot Q) = k \times n$

2-2-(4). Matrix factorization에서의 Stochastic Gradient descent



Rating matrix R User factor matrix P^T

- Computing the standard GD is expensive.
- SGD is a lightweight algorithm which repeats following procedure.
  1. Randomly pick a data instance
  2. Calculate a gradient associated with the instance
  3. Update the solution by the gradient of the instance

- Objective function with the regularizers

Regularizer

샘플 (user i, item j)

$$\min_{P,Q} \sum_{(i,j)\in R} \left\{ \underbrace{\left(r_{ij} - \boldsymbol{p}_i^T \cdot \boldsymbol{q}_j\right)^2}_{\text{loss function}} + \lambda_P \|\boldsymbol{p}_i\|_2^2 + \lambda_Q \|\boldsymbol{q}_j\|_2^2 \right\}$$

P,Q size constraints! >커질수록 좋다!

- Gradient
  - $\frac{\partial f(r_{ij}, \boldsymbol{p}_i, \boldsymbol{q}_j)}{\partial \boldsymbol{p}_i} = -2\left(r_{ij} - \boldsymbol{p}_i^T \cdot \boldsymbol{q}_j\right)\boldsymbol{q}_j + 2\lambda_p \boldsymbol{p}_i$
  - $\frac{\partial f(r_{ij}, \boldsymbol{p}_i, \boldsymbol{q}_j)}{\partial \boldsymbol{q}_i} = -2\left(r_{ij} - \boldsymbol{p}_i^T \cdot \boldsymbol{q}_j\right)\boldsymbol{p}_i + 2\lambda_q \boldsymbol{q}_j$
- Updating rules

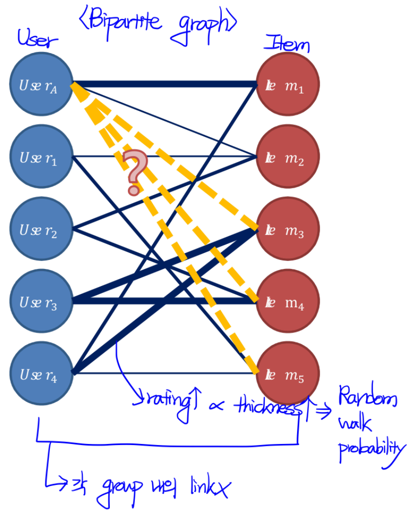
shrinking

샘플
  - $p_i \leftarrow p_i + \eta' \left(\left(r_{ij} - \boldsymbol{p}_i^T \cdot \boldsymbol{q}_j\right) \cdot \boldsymbol{q}_j - \lambda_P \cdot \boldsymbol{p}_i\right)$
  - $q_j \leftarrow q_j + \eta' \left(\left(r_{ij} - \boldsymbol{p}_i^T \cdot \boldsymbol{q}_j\right) \cdot \boldsymbol{p}_i - \lambda_Q \cdot \boldsymbol{q}_j\right)$

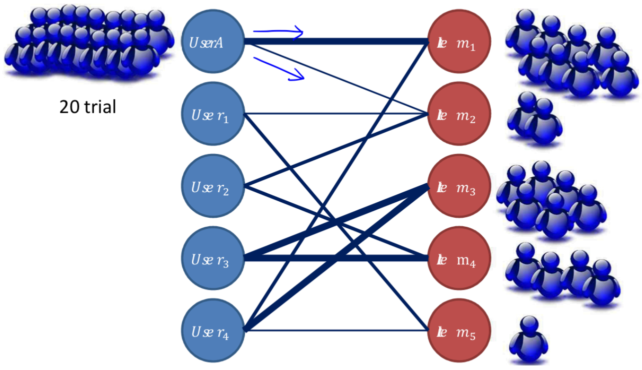
2-3. Random walk on graph

Rating matrix

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



- Users and items are represented as a *bipartite graph*: users and items are nodes and the ratings are links between them.
- Unknown ratings (links) between users and items are estimated by *random walk*, similar to the PageRank algorithm.
- The $s(u, i_{unseen})$ can be proportionally obtained by the fraction of trials reaching i_{unseen} via random walk



3. Recommender system summary

- Rating matrices are extremely sparse in practice!
  - MF (or random walk) also fails
- Hybrid approaches merges CB and CF
  - Multimodal learning
  - Deep learning approaches
  - Cold-start recommendation
- Other issues (SPTN)
  - Scalability : Huge matrix → factorization problem
  - Privacy-preserving : matrix 정보 공유X
  - Timing : 추천 시간
  - Novelty or diversity 단지 유명한 item만 추천X, 익숙지 않은 items도 추천!
- Netflix competition dataset:
  - The number of users: 500k
  - The number of items: 17k
  - The total number of possible ratings
  - 500k X 17k = 8.5 B 필요한 rating
  - The total number of actual ratings = 10 M 실제 rating
  - The portion of non-zero entries = 0.11% ⇒ Very Sparse!

Amazon dataset