Matrix Factorization Techniques for Recommender System

`20230329 (수), 양윤정`

- ref-1 (SGD) : Recommender-Systems-[Netflix].pdf (datajobs.com)
- ref-2 (ALS) : cf.pdf (yifanhu.net)

요약

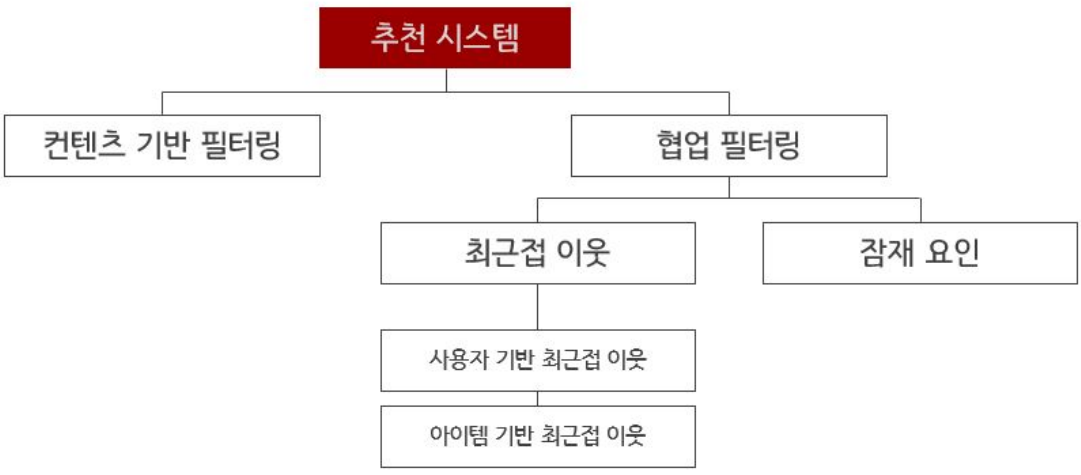
- 추천 시스템 중, 협업 필터링에 기반한 잠재적 요인 모델의 일환인 행렬 인수분해 (**Matrix Factorization, MF**) 에 대한 다양한 방법 설명
- MF 중에서도 Netflix Prize Competition 에서 제안한 방법과 그 결과 소개

---

1. 추천 시스템의 이해/전략

추천 시스템의 종류

- 컨텐츠 기반 필터링 (Content-based filtering)
- 협업 필터링 (Collaborative filtering) ✔️
  - 최근접 이웃 협업 필터링
    - 사용자 기반
    - 아이템 기반
  - 잠재 요인 협업 필터링 → 2009년 넷플릭스 추천 컴퍼티션 (MF-행렬 분해-를 이용한 잠재 요인 협업 필터링 우승)



컨텐츠 기반 필터링 (Content-based filtering)

- 각 사용자나 아이템에 대해 프로필을 만들고, 그 특성을 구체화하는 방식
- 기본 아이디어 : 어떤 사용자가 특정 아이템을 선호할 때, 그 아이템과 비슷한 컨텐츠를 가진 다른 아이템을 추천

협업 필터링 (Collaborative filtering)

- 컨텐츠 기반처럼 구체적으로 프로필을 만들지 않고, 과거 사용자의 행동(예: 이전 구매 기록이나 제품 평가 기록 등)에만 의존하여 시스템을 구축
- 사용자-아이템 간의 상관 관계를 찾는 것이 주 목적
- 장점 : Domain-free 방식 → 즉, 프로필과 같은 관련 지식이 필요하지 않음
- 단점 : 새로운 사용자와 아이템을 다루기에 부적합 → Cold start problem 발생

> 🔷 Cold start problem : 새로운 사용자나 새로운 아이템 등장 시, 기존의 관련 경험이나 행동 방식이 없기 때문에 추천(예측)이 어려워지는 문제

- 협업 필터링의 분류
  - 근접 이웃 방법 (neighborhood methods) : 사용자 간의 관계, 아이템 간의 관계를 계산 (사용자-아이템 행렬에 의존)
    - 사용자 기반 최근접 이웃 협업 필터링 : 특정 사용자와 유사한 사용자들을 선정하고, 이들을 TOP-N이라고 명명한 뒤 이들이 선호하는 아이템을 특정 사용자에게 추천하는 방식
    - 아이템 기반 최근접 이웃 협업 필터링 : 어떤 사용자가 A라는 아이템을 선호한다고 할 때, 그 사용자는 A와 유사한 B라는 아이템 역시 선호할 것이라는 가정 하에 추천을 진행하는 방식 → 일반적으로 사용자 기반보다 높은 정확도
  - 잠재 요인 방법 (latent factor models) : 평점 패턴에서 추론된 20~100 가지의 요인으로 아이템과 사용자 모두를 특성화하여 평점을 설명하려는 대안적인 접근 방식
    - 사용자-아이템 평점 행렬에 잠재되어 있는 어떤 요인(factor)이 있다고 가정하고, 행렬 분해를 통해 그 요인들을 찾아내는 방식
    - 잠재 요인을 구체적으로 정의하는 것이 어렵지만, 실제 시스템에서는 추천의 근거를 마련하는 데에 있어 큰 역할을 수행
    - 예 : 영화 추천 시스템에서 장르/재미/감동과 같은 것들이 잠재 요인

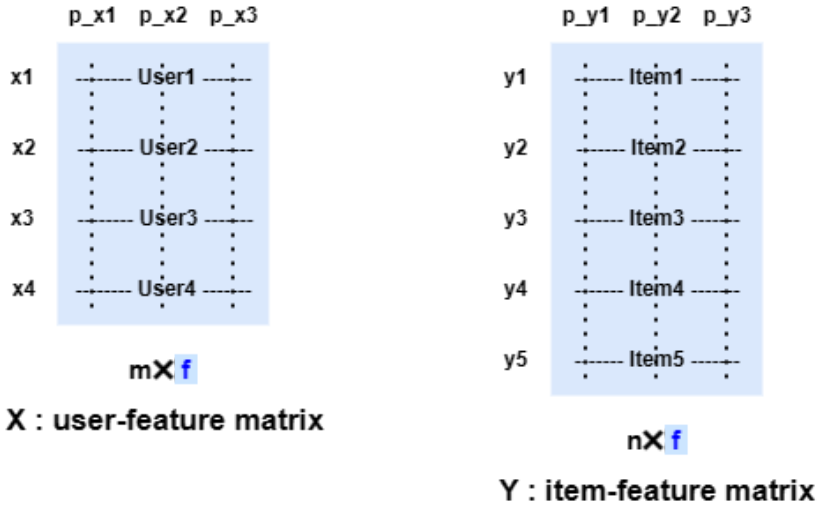
2. Matrix factorization methods

- 행렬 분해는 아이템 평점 패턴으로부터 추론된 요인들의 벡터로부터 아이템과 사용자 모두를 특성화하여 학습시키는 방법
- 아이템과 사용자 간의 높은 상관 관계를 지닐 경우 추천하는 방법
  - 예측 정확도와 우수한 확장성

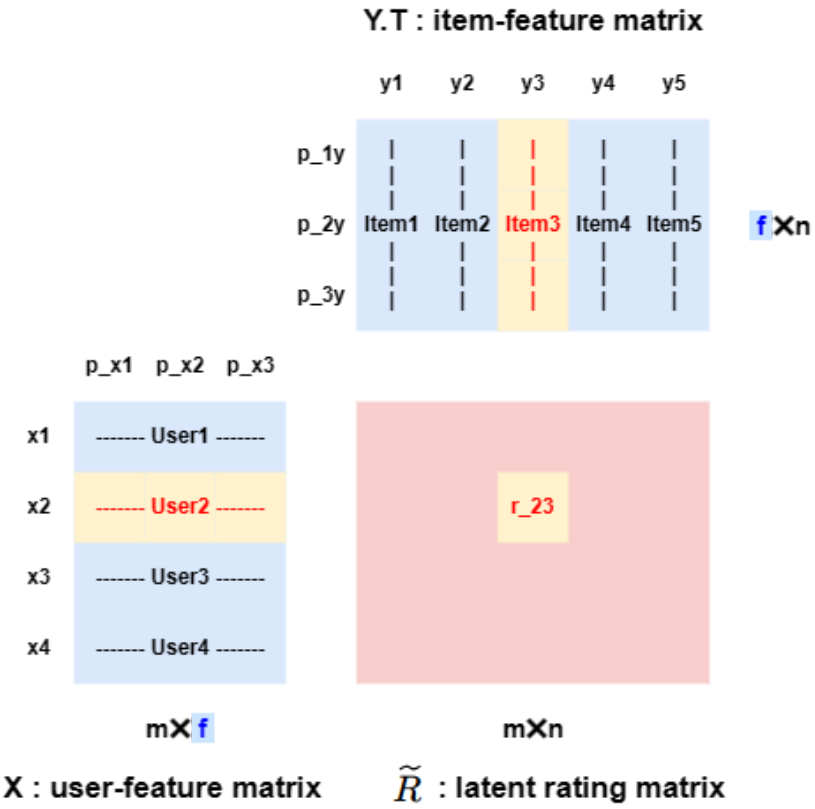
    - 다양한 실상황을 모델링 하기 위한 뛰어난 유연성 제공
- (결측값이 존재하는) 주어진 평점 행렬 : $R \approx \widetilde{R}$: 잠재적 평점 행렬로 예측/근사…

2-1. Latent rating matrix (latent user-item matrix) 의 의미

1. $X = (x_u)_{u=1,\dots,m} \in Mat_{m\times f}(\mathbb{R})$
    - m명의 user가 f개의 feature에 대해 내린 평점들을 행벡터로 쌓은 행렬
    - 즉, u-번째 행 x_u는 u-번째 사람의 평점 벡터 $\in \mathbb{R}^f$
    - 높은 점수를 받은 항목에 대한 사용자의 관심 정도(or 선호도)를 측정한 값
2. $Y = (y_i)_{i=1,\dots,n} \in Mat_{n\times f}(\mathbb{R})$
    - n개의 item이 f개의 feature를 어느정도 보유하고 있는지를 행벡터로 쌓은 행렬
    - 즉, i-번째 행 y_i는 i-번째 item이 각 feature들을 어느 정도 보유하는지를 나타내는 벡터 $\in \mathbb{R}^f$



3. Latent rating matrix $\widetilde{R} := XY^T = \left(\langle x_u, y_i \rangle\right) \in Mat_{m\times n}(\mathbb{R}),\ 1 \le u \le m \text{ and } 1 \le i \le n$
    - **[Notation]** $\widetilde{R} := (\tilde{r}_{ui})$
    - $\tilde{r}_{ui} = \langle x_u, y_i \rangle = {x_u}^T y_i \in \mathbb{R}$: the (u,i)-component of latent rating matrix
        - **[주의]** $x_u \in \mathbb{R}^f \cong Mat_{f\times 1}(\mathbb{R}),\ y_i \in \mathbb{R}^f \cong Mat_{f\times 1}(\mathbb{R})$
        - $\langle$ u-번째 user의 feature vector, i-번째 item의 feature 벡터 $\rangle = {x_u}^T y_i$
        - 의미 : u-번째 user와 i-번째 item의 상호 작용
        - 즉, item이 보유하고 있는 각 특성에 대한 user의 전반적인 관심 정도(선호도, 평점)를 측정



4. Goal : $\widetilde{R} \approx R$ → 즉, $\widetilde{R}$을 학습시키자!!
    - 기존 방식 : 주어진 sparse rating matrix의 결측값을 채워서 dense하게 만들고 학습시킴
    - 문제점 : 계산량 증가, 데이터 왜곡
    - 이 논문이 발표될 당시의 연구 : 정규화된 모델을 사용하여 overfitting을 피하고 관찰된 평점만 이용하여 모델링할 것을 제안
    - To learn the factor vectors (p_u and q_i), the system minimizes the regularized squared error on the set of known ratings.
    - known ratings $R = (r_{ui}) \in Mat_{m \times n}(\mathbb{R})$

2-2. Basic MF model - Loss Function : regularized squared error

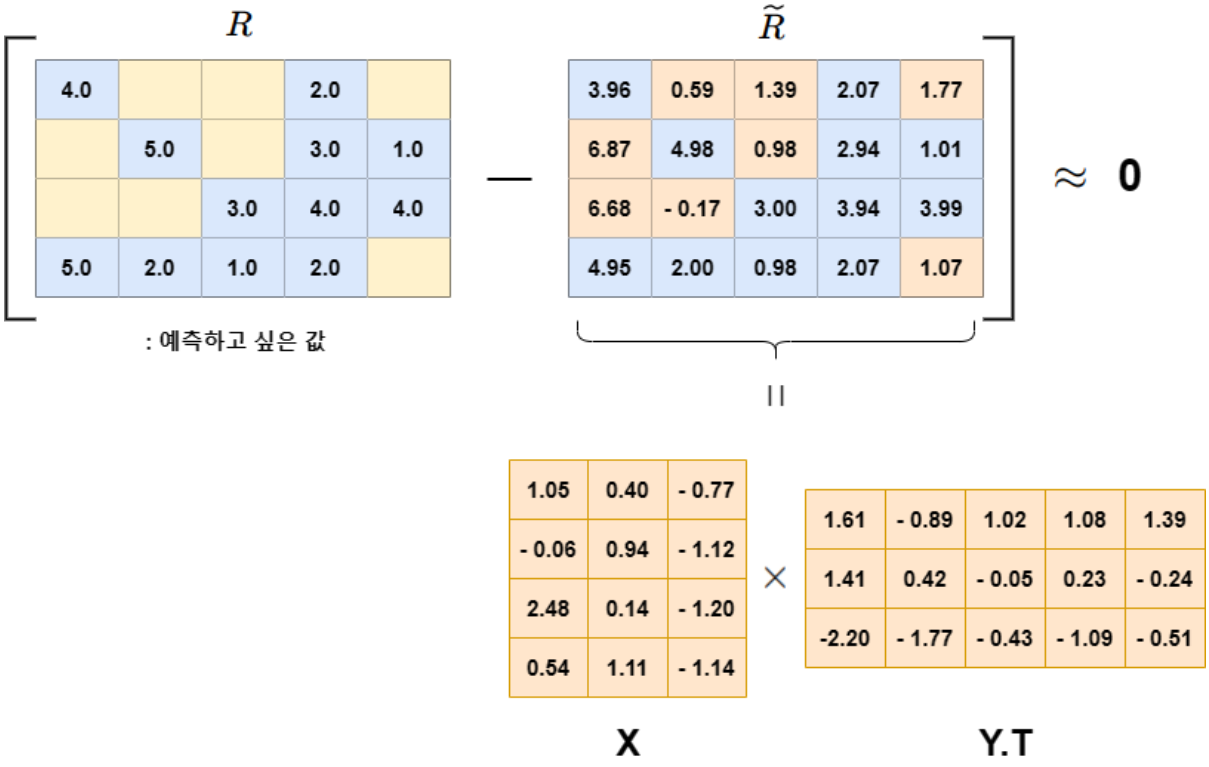
$$\min_{x^*, y^*} \sum_{(u,i) \in \kappa} \left\{ \left(r_{ui} - \langle x_u, y_i \rangle\right)^2 + \lambda \left(\|x_u\|^2 + \|y_i\|^2 \right) \right\}$$

- $\kappa = \{(u,i) \mid 1 \le u \le m,\ 1 \le i \le n\}$: the set of the (u,i) pairs for which r_{ui} is known (the training set)

2-3. Learning Algorithms : Two approaches to minimizing above loss function

2-3-1. Stochastic Gradient Descent (SGD)

- 구현이 쉽고, 계산이 빠름 (그러나 경우에 따라 ALS가 더 좋을 수 있음)
- Given ratings for training set, $R = (r_{ui}) \in Mat_{m\times n}(\mathbb{R})$
- Goal : $\widetilde{R} = XY^T \approx R$, i.e., $\widetilde{R}$ predicts R.
- Compute the associated prediction error :
  - $e_{ui} := r_{ui} - \tilde{r}_{ui} = r_{ui} - \langle x_u, y_i \rangle$
- 기울기의 반대 방향으로 γ 에 비례하는 크기로 매개변수를 수정하여 결과 출력 :
  - $x_u \leftarrow x_u + \gamma(e_{ui}y_i - \lambda x_u)$
  - $y_i \leftarrow y_i + \gamma(e_{ui}x_u - \lambda y_i)$



2-3-2. Alternating Least Squares (ALS)

- x_u, y_i 모두 미지수이므로 **2-2**에서 정의한 loss function은 convex가 아님
- 둘 중 하나의 factor를 번갈아가며 상수로 여기고, 최소 제곱(least square) 문제로 해결 → **i.e., Alternating + Least Squares optimization process**
- 각 단계가 수렴할 때까지 위의 loss function은 감소
- Goal : (각각을 열벡터로 보고 계산…)
  - to find a vector $x_u \in \mathbb{R}^f \cong Mat_{f\times 1}(\mathbb{R})$ for each user $u = 1, \ldots, m$

    → “user-factor”

- to find a vector $y_i \in \mathbb{R}^f \cong Mat_{f\times 1}(\mathbb{R})$ for each item $i = 1, \dots, n$

  → “item-factor”

Cost Function for basic-ALS

$$
\begin{aligned}
L(x_u, y_i) &= \sum_{(u,i)\in\kappa} \Big\{ \big(r_{ui} - \langle x_u, y_i \rangle\big)^2 + \lambda\big(\|x_u\|^2 + \|y_i\|^2\big) \Big\} \\
&= \sum_{(u,i)\in\kappa} \big(r_{ui} - \langle x_u, y_i \rangle\big)^2 + \lambda\Big(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \Big)
\end{aligned}
$$

1. 1st Step : User-factor x_u 업데이트
   - For arbitrary chosen $u \in \{1, \dots, m\}$ and consider the fixed user vector x_u.
   - Goal : Find x_u that minimizes the cost function
     - $\frac{\partial L(x_u)}{\partial x_u} = 0$ 을 만족하는 x_u 찾는 문제
       - $Y \in Mat_{n\times f}(\mathbb{R})$: item-factor matrix
       - $Y^TY \in Mat_{f\times f}(\mathbb{R})$
       - $r(u) = (\langle x_u, y_1 \rangle \;\; \langle x_u, y_2 \rangle \;\; \cdots \;\; \langle x_u, y_n \rangle)^T \in \mathbb{R}^n \cong Mat_{n\times 1}(\mathbb{R})$
   - 위 식을 정리하면 :

     🔷 $x_u = (Y^TY + \lambda I)^{-1} Y^T r(u)$

2. 2nd Step : Item-factor y_i 업데이트
   - For arbitrary chosen $i \in \{1, \dots, n\}$ and consider the fixed item vector y_i.
   - Goal : Find y_i that minimizes the cost function
     - $\frac{\partial L(y_i)}{\partial y_i} = 0$ 을 만족하는 y_i 찾는 문제
       - $X \in Mat_{m\times f}(\mathbb{R})$: user-factor matrix
       - $X^TX \in Mat_{f\times f}(\mathbb{R})$
       - $r(i) = (\langle x_1, y_i \rangle \;\; \langle x_2, y_i \rangle \;\; \cdots \;\; \langle x_m, y_i \rangle)^T \in \mathbb{R}^m \cong Mat_{m\times 1}(\mathbb{R})$
   - 위 식을 정리하면 :

     🔷 $y_i = (X^TX + \lambda I)^{-1} X^T r(i)$

Cost Function for weighted-ALS

$$
\begin{aligned}
L(x_u, y_i) &= \sum_{(u,i)\in\kappa} \left\{ c_{ui}\left(p_{ui} - \langle x_u, y_i \rangle\right)^2 + \lambda\left(\|x_u\|^2 + \|y_i\|^2\right) \right\} \\
&= \sum_{(u,i)\in\kappa} c_{ui}\left(p_{ui} - \langle x_u, y_i \rangle\right)^2 + \lambda\left(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \right) \\
&\text{where} \quad p_{ui} = \begin{cases} 1 & \text{if } r_{ui} > 0 \\ 0 & \text{if } r_{ui} = 0 \end{cases}
\end{aligned}
$$

- $\min_{u,i} L(x_u, y_i)$
- $\lambda(\star)$: regularization term → overfitting 방지

> 🔷 Exact value of the parameter λ is data-dependent and determined by cross validation.

- 여기서 제시한 loss function (cost function) 은 가능한 모든 (user,item) pair에 대하여 계산하기 때문에 식은 mn-term으로 이루어져 있음

- $P = (p_{ui})$: Binary rating matrix (= bipartite adjacency matrix) → 아이템을 소비한 적이 있으면 선호한다고 생각하고, 소비한 적이 없으면 선호도가 없다고 생각

> 🔷 which indicates the preference of user u to item i

- 평점 r_{ui} 가 측정하는 신뢰도 개념을 고려하기 위한 c_{ui} **confidence (신뢰도 측정 변수) 도입**
  - 가정 : 선호도의 정도는 다양한 원인에서 비롯
  - 예를 들어, 선호도 $p_{ui} = 0$ 이라는 것은 낮은 신뢰도와 관련있고, 이는 사용자가 아이템의 존재를 몰랐다거나 가격 등의 이유로 소비하지 않음
  - 또는 어떤 상품을 좋아하지 않지만 누군가의 선물을 위해 소비하는 경우 → 즉, 노이즈 있음
  - 이처럼 선호하는 아이템이라 할지라도 신뢰도 수준이 다름
  - 일반적으로 평점 r_{ui}가 클수록 아이템에 대한 선호도 커짐
  - 따라서, 주어진 p_{ui}에 대한 신뢰도를 측정하는 변수인 c_{ui} 도입하겠다!!

> 🔷 The numerical value of explicit feedback indicates preference, whereas the numerical value of implicit feedback indicates confidence.

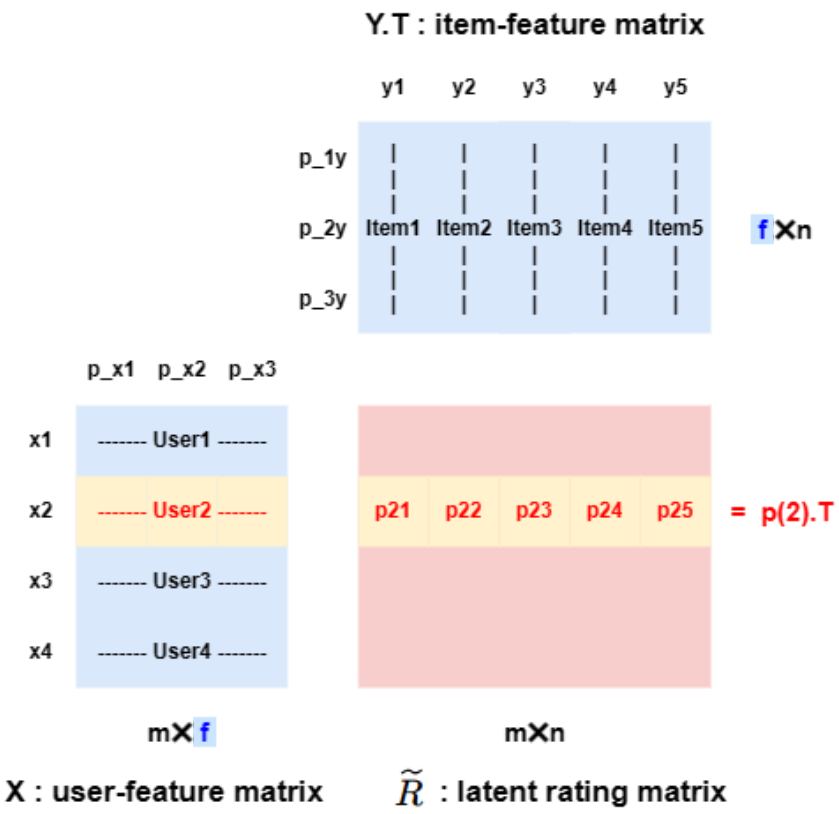
- $c_{ui} = 1 + \alpha r_{ui},\ \alpha$: increasing rate constant (실험에 의하면 $\alpha = 40$ 일 때 성능 좋음)
- $\tilde{p}_{ui} = \langle x_u, y_i \rangle = {x_u}^T y_i \in \mathbb{R}$: preference (SGD에서 $\tilde{r}_{ui}$ 에 해당, 예측하고 싶은 값)

SGD를 이용한 MF 업데이트 방법과의 차이점

- 다양한 신뢰 수준 고려
  - SGD에서는 각 user-factor, item-factor 를 같은 weight로 측정하였지만, ALS에서는 각 user 또는 item 별로 다른 신뢰도(weight)를 매겨서 계산함
  - 즉, 일종의 weighted user-item rating … 로 생각
- 관찰된 데이터만 고려하는 것이 아니라, 가능한 모든 (u, i) pair를 고려
  - SGD에서는 $r_{ui} > 0$ 인 값들만으로 알고리즘 돌림

1st Step : User-factor x_u 업데이트

- For arbitrary chosen $u \in \{1, \dots, m\}$ and consider the fixed user vector x_u.
- Goal : Find x_u that minimizes the cost function
  - $\frac{\partial L(x_u)}{\partial x_u} = -2 \sum_i c_{ui}(p_{ui} - \langle x_u, y_i \rangle) y_i + 2\lambda x_u = 0$ 을 만족하는 x_u 찾는 문제
    - $Y \in Mat_{n \times f}(\mathbb{R})$: item-factor matrix
    - $Y^T Y \in Mat_{f \times f}(\mathbb{R})$
    - For each user u, let $C^u = (C^u{}_{ii} = c_{ui}) \in Mat_{n \times n}(\mathbb{R})$ be a diagonal matrix.
      - c_{ui} : 고정된 user u의 각 아이템에 대한 가중치 상수
    - $p(u) = (\langle x_u, y_1 \rangle \ \ \langle x_u, y_2 \rangle \ \ \cdots \ \ \langle x_u, y_n \rangle)^T \in \mathbb{R}^n \cong Mat_{n \times 1}(\mathbb{R})$



- 위 식을 정리하면 :

> 🔷 $x_u = (Y^T C^u Y + \lambda I)^{-1} Y^T C^u p(u)$

2nd Step : Item-factor y_i 업데이트

- 위와 마찬가지 방법으로 해결하면
  - $\frac{\partial L(y_i)}{\partial y_i} = -2 \sum_u c_{ui}(p_{ui} - \langle x_u, y_i \rangle)x_u + 2\lambda y_i = 0$ 을 만족하는 y_i 찾는 문제
    - $X \in Mat_{m \times f}(\mathbb{R})$
    - $X^T X \in Mat_{f \times f}(\mathbb{R})$
    - For each item i, let $C^i = (C^i{}_{uu} = c_{ui}) \in Mat_{m \times m}(\mathbb{R})$ be a diagonal matrix.
      - c_{ui} : 고정된 item i의 각 user에 대한 가중치 상수
    - $p(i) = (\langle x_1, y_i \rangle \ \ \langle x_2, y_i \rangle \ \ \cdots \ \ \langle x_m, y_i \rangle)^T \in \mathbb{R}^m \cong Mat_{m \times 1}(\mathbb{R})$우
  - 위 식을 정리하면 :

> 🔷 $y_i = (X^T C^i X + \lambda I)^{-1} X^T C^i p(i)$