- 행렬분해

일반적으로 작은 수 k (ex k≒10)를 고정 후

각 유저의 니즈 k차원 벡터 x_u로 요약 후, 아이템 i를

k차원 벡터 y_i로 요약한다.

$x_1, \cdots, x_n \in R^k$가 유저들에 대한 factor,

$y_1, \cdots, y_m \in R^k$가 아이템에 대한 factor 들이 되게한다.

그 후, k × n Matrix X × m Matrix Y 를 다음과 같이 정의한다.

$X = [x_1 \cdots x_n]$. $Y = [y_1 \cdots y_m]$

claim: $R \simeq X^T Y$를 추정.

이를 목적함수를 ~~최~~ 최소화하고 최적의 X, Y를 찾는

최적화 문제로 만들자. 특히 observed ratings의

최소제곱오차를 최소화 하는 방향으로.

$$\min_{X,Y} \sum_{r_{ui}\ \text{observed}} (r_{ui} - x_u^T y_i)^2 + \lambda \left(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \right)$$

이 목적함수는 안볼록이다. ($\because$ ~~x~~ $x_u^T y_i$ term 때문에)

경사하강을 쓰면 속도도 느리고, iteration cost도 많이 듦.

그래서 variable X set 고정 후 상수 취급시 목적함수는 Y의

convex function이 되고, 반대 경우도 적용.

결론적으로 Y 고정 후 X 최적화 하고, X 고정 후 Y 최적화 한거를

이를 수렴할 때 까지 반복 ⇒ 이게 ALS 임.

- 계산비용 분석

x_u 업데이트시 $O(n_u k^2 + k^3)$의 비용이 듦. n_u: 유저 u가 rating한 item수.

y_i 업데이트 시 $O(n_i k^2 + k^3)$ 들고, n_i: 아이템 i에 rating한 user 수.

i). 행렬분해. (ALS 버전)

M = playlists. n = songs.

$$M = \begin{array}{c} \\ \text{playlist} \\ \vdots \\ \text{playlist}m \end{array} \begin{array}{c} \text{song}1 \quad \cdots \quad \text{song}\,n \\ \left[\begin{array}{ccccc} 1 & 0 & 1 & 0 & \cdots \\ 0 & 0 & 1 & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots \end{array}\right] \end{array}$$

M_{ij} : playlist i contains song j.

one of decomposition of M.

$M = UV^T$, $U = m \times d$, $V = n \times d$. d is arbitrary dimention.

$\min |M - UV^T|^2$: 최소제곱오차.

$$\begin{bmatrix} u_1^T \\ \vdots \\ u_n^T \end{bmatrix} v_j = \begin{bmatrix} m_{1j} \\ \vdots \\ m_{nj} \end{bmatrix}.$$ v_j : 최소제곱해로 찾는다.

반대로, $u_i^T [v_1 \cdots v_n] = [y_{i1} \cdots y_{in}]$.

예측 가능하다. playlist i 안에 song j를.

만약 $u_i^T v_j$가 충분히 클 경우.

이를 ALS라 한다.

ii). CMF. (collective matrix factorization)

그러나. 지금 문제에서. 우리는 . songs 랑 tags를 . 동시에 예측해야함

지금 제시된 두 행렬. /. 해결을 위한. ALS 사용 . with. 같은 추가된 제한.

$$M_{song} = \begin{array}{c|c} & \text{song1} \cdots \text{song n} \\ \hline \begin{array}{c}\text{playlist1}\\ \vdots \\ \text{playlist m}\end{array} & \begin{bmatrix} 1 & 0 & \cdots & 1 \\ \vdots & & \ddots & \vdots \\ 0 & 1 & \cdots & \end{bmatrix} \end{array}, \quad M_{tag} = \begin{array}{c|c} & \text{Tag1} \cdots \text{Tag l} \\ \hline \text{playlist} & \begin{bmatrix} \vdots & \ddots & \vdots \end{bmatrix} \end{array}$$

$$= U_{song} V_{song}^T \qquad = U_{tag} V_{tag}^T$$

$$\min \left(\underbrace{\| U_{song} V_{song}^T - M_{song} \|^2 + \| U_{tag} V_{tag}^T - M_{tag} \|^2}_{①} \right)$$

우리는 . U_{song} 과 U_{tag} 가. 유사하다고 제한한다.

$$s.t. \; \| U_{song_i} - U_{Tag_i} \|^2 \le \lambda.$$

정규화 form을 두자.

$$\min \left(\sum_{x \in \{song, tag\}} \| U_x V_x^T - M_x \|^2 + \| U_{song} - U_{tag} \|^2 \right)$$

or. $\lambda = 0$, then. $U_{song} = U_{tag} = U$. ?

↳ 우리는 얻는다 (야호!)

$$①식 = \left(U \begin{bmatrix} V_{song} \\ V_{tag} \end{bmatrix}^T - [M_{song} \; M_{tag}] \right)^2.$$

~~[illegible]~~ 어떤 추정이 없는 코드가 오면.

ALS를 써서. 문제를 잘. 해결 가능하다?