Machine Learning with Graphs (MLG)

RecSys: Matrix Factorization

Latent Factor Models

Cheng-Te Li (李政德)

Institute of Data Science

National Cheng Kung University

chengte@mail.ncku.edu.tw

Latent Factor Model

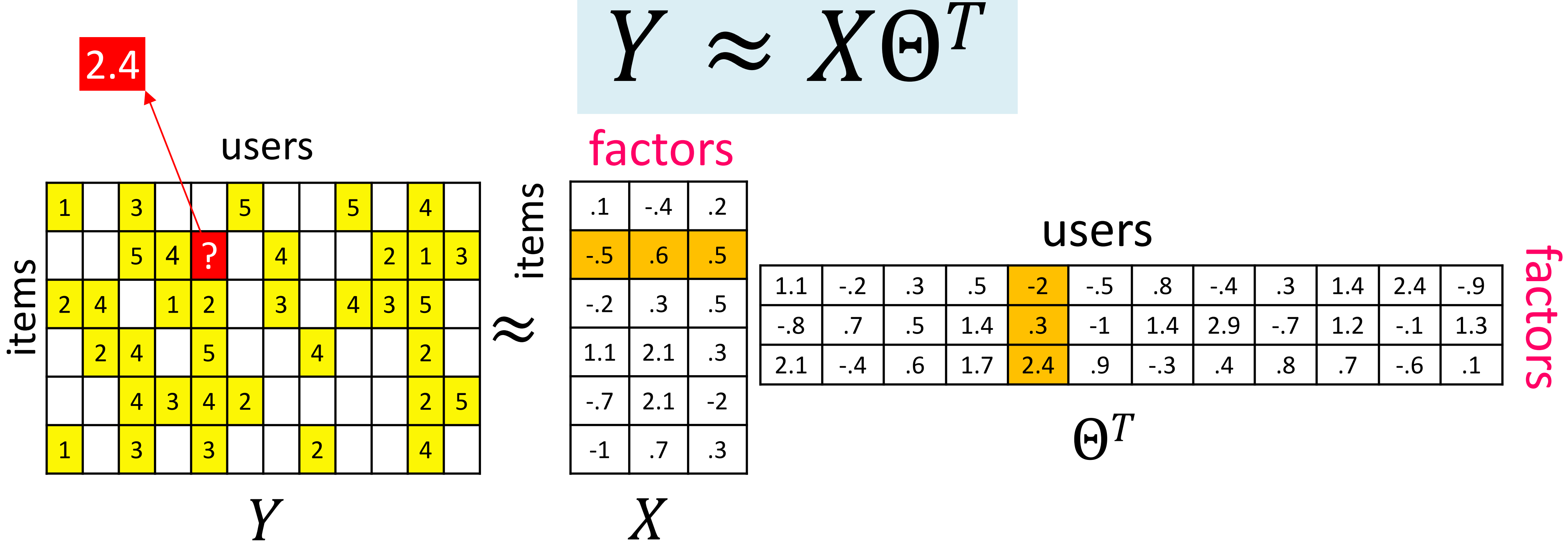
$$Y \approx X\Theta^T$$

$$\underset{Y}{\begin{array}{c|cccccccccccc} & \multicolumn{12}{c}{\text{users}} \\ \hline \text{items} & 1 & & 3 & & & 5 & & & 5 & & 4 & \\ & & & 5 & 4 & & & 4 & & & 2 & 1 & 3 \\ & 2 & 4 & & 1 & 2 & & 3 & & 4 & 3 & 5 & \\ & & 2 & 4 & & 5 & & & 4 & & & 2 & \\ & & & 4 & 3 & 4 & 2 & & & & & 2 & 5 \\ & 1 & & 3 & & 3 & & & 2 & & & 4 & \end{array}} \approx \underset{X}{\begin{array}{c|ccc} & \multicolumn{3}{c}{\text{factors}} \\ \hline \text{items} & .1 & -.4 & .2 \\ & -.5 & .6 & .5 \\ & -.2 & .3 & .5 \\ & 1.1 & 2.1 & .3 \\ & -.7 & 2.1 & -2 \\ & -1 & .7 & .3 \end{array}} \underset{\Theta^T}{\begin{array}{cccccccccccc|c} \multicolumn{12}{c}{\text{users}} & \\ \hline 1.1 & -.2 & .3 & .5 & -2 & -.5 & .8 & -.4 & .3 & 1.4 & 2.4 & -.9 & \text{factors} \\ -.8 & .7 & .5 & 1.4 & .3 & -1 & 1.4 & 2.9 & -.7 & 1.2 & -.1 & 1.3 & \\ 2.1 & -.4 & .6 & 1.7 & 2.4 & .9 & -.3 & .4 & .8 & .7 & -.6 & .1 & \end{array}}$$

- For now let's assume we can approximate the rating matrix Y as a product of "**thin**" $X \cdot \Theta^T$
  - Y has missing entries but let's ignore that for now!
    - Basically, we want the reconstruction error to be small **on known ratings** and don't care about the missing ones

Ratings as Products of Factors

$$Y \approx X\Theta^T$$



Y

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	

X

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

Θ^T

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

- How to estimate the missing rating of user u for item i?

$$\hat{y}_{ui} = x_i \cdot \theta_u = \sum_k x_{ik} \cdot \theta_{uk}$$

x_i = row i of X

θ_u = column u of Θ^T

Predicting Movie Ratings

- User rates movies using zero to five stars

Movie	Alice	Bob	Carol	Dave
我的少女時代	5	5	0	0
派特的幸福劇本	5	?	?	0
你的名字	?	4	0	?
黑暗騎士	0	0	5	4
神力女超人	0	0	5	?

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & ? \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & ? & ? & 1 \\ ? & 1 & 1 & ? \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & ? \end{bmatrix}$$

$r^{(i,j)}$: 1 if user j rates movie i (0 otherwise)

$y^{(i,j)}$: rating given by user j to movie 1 (defined only if $r(i,j) = 1$)

n_u: number of users

n_m: number of movies

Predicting Movie Ratings

		$\theta^{(1)}$	$\theta^{(2)}$	$\theta^{(3)}$	$\theta^{(4)}$	Features/Factors		
						$x_0 = 1$	x_1	x_2
	Movie	Alice	Bob	Carol	Dave	*virtual*	*romance*	*action*
$x^{(1)}$	我的少女時代	5	5	0	0	1	0.99	0
$x^{(2)}$	派特的幸福劇本	5	?	?	0	1	1.0	0.01
$x^{(3)}$	你的名字	?$_{4.95}$	4	0	?	1	0.99	0
$x^{(4)}$	黑暗騎士	0	0	5	4	1	0.1	1.0
$x^{(5)}$	神力女超人	0	0	5	?	1	0	0.9

Let $x^{(i)}$ be the feature vector of movie i

Let $\theta^{(j)}$ be the parameter for user j

For each user j, learn a parameter $\theta^{(j)} \in \mathbb{R}^{n+1}$

Predict user j as rating movie i with $\left(\theta^{(j)}\right)^T x^{(i)}$ stars

$$x^{(1)} = \begin{bmatrix} 1 \\ 0.99 \\ 0 \end{bmatrix} \quad \theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$$

$$\left(\theta^{(1)}\right)^T x^{(3)} = 5 \times 0.99 = 4.95$$

n: number of features

Problem Formulation

$x^{(i)}$: feature vector of movie i

$\theta^{(j)}$: parameter vector for user j $\theta^{(j)} \in \mathbb{R}^{n+1}$

$r^{(i,j)}$: 1 if user j rates movie i (0 otherwise)

$y^{(i,j)}$: rating given by user j to movie 1 (defined only if $r(i,j) = 1$)

$m^{(j)}$: number of movies rated by user j

- For each user j and movie i, predicted rating $\left(\theta^{(j)}\right)^T x^{(i)}$
- Given that $x^{(i)}$ is known, the goal is to learn $\theta^{(j)}$
- Apply "Linear Regression with Regularization" X: movie ratings, y: $x^{(i)}$

$$\min_{\theta^{(j)}} \frac{1}{2\cancel{m}^{(j)}} \sum_{i:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2\cancel{m}^{(j)}} \sum_{k=\mathbf{1}}^{n} \left(\theta_k^{(j)}\right)^2$$

Optimization Objective

Given that $x^{(1)}, x^{(2)}, \dots, x^{(n_m)}$ is known

To learn $\theta^{(j)}$ (parameter for user j):

$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{i:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

To learn $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}$ (parameters for all users):

$$\min_{\theta^{(1)},\theta^{(2)},\dots,\theta^{(n_u)}} \frac{1}{2} \boxed{\sum_{j=1}^{n_u}} \sum_{i:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \boxed{\sum_{j=1}^{n_u}} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

Gradient Descent

$$\min_{\theta^{(1)},\theta^{(2)},\ldots,\theta^{(n_u)}} \underbrace{\frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2}_{\to J\left(\theta^{(1)},\theta^{(2)},\ldots,\theta^{(n_u)}\right) \text{ : loss function}}$$

Gradient descent update:

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \sum_{i:r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right) x_k^{(i)} \qquad (\text{for } k = 0)$$

$$(\text{for } k \neq 0)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\underbrace{\sum_{i:r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right) x_k^{(i)} + \lambda\theta_k^{(j)}}_{\to \frac{\partial}{\partial\theta_k^{(j)}} J\left(\theta^{(1)},\theta^{(2)},\ldots,\theta^{(n_u)}\right)} \right)$$

From User to Movie Vectors

- However, cannot know each movie i's feature vector $x^{(i)}$

Features/Factors

unknown

		$\theta^{(1)}$	$\theta^{(2)}$	$\theta^{(3)}$	$\theta^{(4)}$	$x_0 = 1$	x_1	x_2
	Movie	Alice	Bob	Carol	Dave	*virtual*	*romance*	*action*
$x^{(1)}$	我的少女時代	5	5	0	0	1	0.99	0
$x^{(2)}$	派特的幸福劇本	5	?	?	0	1	1.0	0.01
$x^{(3)}$	你的名字	?	4	0	?	1	0.99	0
$x^{(4)}$	黑暗騎士	0	0	5	4	1	0.1	1.0
$x^{(5)}$	神力女超人	0	0	5	?	1	0	0.9

- If we can know each user's preference vector $\theta^{(j)}$, we will be able to estimate each movie i's feature vector $x^{(i)}$

$$\theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} \quad \theta^{(2)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} \quad \theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} \quad \theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

$$\left(\theta^{(1)}\right)^T x^{(1)} \approx 5 \quad \left(\theta^{(2)}\right)^T x^{(1)} \approx 5$$

$$\left(\theta^{(3)}\right)^T x^{(1)} \approx 0 \quad \left(\theta^{(4)}\right)^T x^{(1)} \approx 0$$

$$x^{(1)} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

Optimization Objective

Given that $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$ is known

To learn $x^{(i)}$ (parameter for user j):

$$\min_{\theta^{(j)}} \frac{1}{2} \sum_{j:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

To learn $x^{(1)}, x^{(2)}, \ldots, x^{(n_m)}$ (parameters for all movies):

$$\min_{x^{(1)}, x^{(2)}, \ldots, x^{(n_m)}} \frac{1}{2} \boxed{\sum_{i=1}^{n_m}} \sum_{j:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \boxed{\sum_{i=1}^{n_m}} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

Latent Factor (LF) Model

To learn $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}$ (parameters for all users):

$$\min_{\theta^{(1)},\theta^{(2)},\dots,\theta^{(n_u)}} \frac{1}{2} \boxed{\sum_{j=1}^{n_u}} \sum_{i:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \boxed{\sum_{j=1}^{n_u}} \sum_{k=1}^{n} \left(\theta_k^{(j)}\right)^2$$

To learn $x^{(1)}, x^{(2)}, \dots, x^{(n_m)}$ (parameters for all movies):

$$\min_{x^{(1)},x^{(2)},\dots,x^{(n_m)}} \frac{1}{2} \boxed{\sum_{i=1}^{n_m}} \sum_{j:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \boxed{\sum_{i=1}^{n_m}} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

Guess:

$$\theta^{(j)} \rightarrow x^{(i)} \rightarrow \theta^{(j)} \rightarrow x^{(i)} \rightarrow \theta^{(j)} \rightarrow x^{(i)} \rightarrow \cdots$$

Latent Factor Optimization Objective

Given $x^{(1)}, x^{(2)}, \ldots, x^{(n_m)}$, estimate $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$:

$$\min_{\theta^{(1)},\theta^{(2)},\ldots,\theta^{(n_u)}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2$$

Given $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$, estimate $x^{(1)}, x^{(2)}, \ldots, x^{(n_m)}$:

$$\min_{x^{(1)},x^{(2)},\ldots,x^{(n_m)}} \frac{1}{2}\sum_{i=1}^{n_m}\sum_{j:r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

Minimizing $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n_u)}$ and $x^{(1)}, x^{(2)}, \ldots, x^{(n_m)}$ simultaneously:

$$J(x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)})$$

Final Loss Function

$$= \frac{1}{2}\sum_{(i,j):\,r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2$$

$$\min_{\substack{x^{(1)},\ldots,x^{(n_m)}\\ \theta^{(1)},\ldots,\theta^{(n_u)}}} J(x^{(1)}, \ldots, x^{(n_m)}, \theta^{(1)}, \ldots, \theta^{(n_u)})$$

Latent Factor Model Algorithm

(1) Initialize $x^{(1)}, \dots, x^{(n_m)}$ & $\theta^{(1)}, \dots, \theta^{(n_u)}$ to small random values

(2) Minimize $J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)})$ using gradient descent (or an advanced optimization algorithm)

$\cancel{x_0 = 1}$

$$\begin{matrix} \cancel{\theta_0} \\ \theta_1 \\ \vdots \\ \theta_n \end{matrix}$$

for every $j = 1, \dots, n_u$ and $i = 1, \dots, n_m$:

$$x_k^{(i)} := x_k^{(i)} - \alpha \left(\sum_{j:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right) \theta_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \left(\sum_{i:r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right) x_k^{(i)} + \lambda \theta_k^{(j)} \right)$$

(3) For a user with parameters θ and a movie with (learned) features x, predict a star rating of $\theta^T x$

$$\hat{y}^{(i,j)} = \left(\theta^{(j)}\right)^T \left(x^{(i)}\right)$$

Latent Factor Model: Matrix Factorization (MF)

Movie	Alice	Bob	Carol	Dave
我的少女時代	5	5	0	0
派特的幸福劇本	5	?	?	0
你的名字	?	4	0	?
黑暗騎士	0	0	5	4
神力女超人	0	0	5	?

$y^{(i,j)}$

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & ? \end{bmatrix}$$

$$\hat{y}^{(i,j)} = \left(\theta^{(j)}\right)^T \left(x^{(i)}\right)$$

$$Y \approx X\Theta^T$$

$$\hat{Y} = \begin{bmatrix} \left(\theta^{(1)}\right)^T\left(x^{(1)}\right) & \left(\theta^{(2)}\right)^T\left(x^{(1)}\right) & \cdots & \left(\theta^{(n_u)}\right)^T\left(x^{(1)}\right) \\ \left(\theta^{(1)}\right)^T\left(x^{(2)}\right) & \left(\theta^{(2)}\right)^T\left(x^{(2)}\right) & & \left(\theta^{(n_u)}\right)^T\left(x^{(2)}\right) \\ \vdots & & \ddots & \vdots \\ \left(\theta^{(1)}\right)^T\left(x^{(n_m)}\right) & \left(\theta^{(2)}\right)^T\left(x^{(n_m)}\right) & \cdots & \left(\theta^{(n_u)}\right)^T\left(x^{(n_m)}\right) \end{bmatrix} = X\Theta^T$$

Low-Rank Matrix Factorization

$$X = \begin{bmatrix} \left(x^{(1)}\right)^T \\ \left(x^{(2)}\right)^T \\ \vdots \\ \left(x^{(n_m)}\right)^T \end{bmatrix} \qquad \Theta = \begin{bmatrix} \left(\theta^{(1)}\right)^T \\ \left(\theta^{(2)}\right)^T \\ \vdots \\ \left(\theta^{(n_u)}\right)^T \end{bmatrix}$$

Find Relevant Movies by MF

When a user just views an item, how to recommend relevant items to her?

- For each movie i, we learn a feature vector $x^{(i)} \in \mathbb{R}^n$
  - E.g. x_1: romance, x_2: action, x_3: comedy, x_4:
  - Jointly learn X and Θ using Gradient Descent
- How to find movies j related to movie i?

Euclidean: $\sqrt{\sum_{k=1}^{n} x_k^{(i)} - x_k^{(j)}}$

Cosine: $\frac{x^{(i)} \cdot x^{(j)}}{\sqrt{x^{(i)} \cdot x^{(i)}}\sqrt{x^{(j)} \cdot x^{(j)}}}$

Small $\left\| x^{(i)} - x^{(j)} \right\|$ ⇒ movie j and i are similar

- → Find the top movies j with the smallest $\left\| x^{(i)} - x^{(j)} \right\|$

	$\theta^{(1)}$	$\theta^{(2)}$	$\theta^{(3)}$	$\theta^{(4)}$	$x_0 = 1$	x_1	x_2
Movie	Alice	Bob	Carol	Dave	virtual	romance	action
$x^{(1)}$ 我的少女時代	5	5	0	0	1	0.99	0
$x^{(2)}$ 派特的幸福劇本	5	?	?	0	1	1.0	0.01
$x^{(3)}$ 你的名字	?	4	0	?	1	0.99	0
$x^{(4)}$ 黑暗騎士	0	0	5	4	1	0.1	1.0
$x^{(5)}$ 神力女超人	0	0	5	?	1	0	0.9

How about New Users? The **Cold-Start** Problem

No Ratings (Never Rate)

$n = 2$

		$\theta^{(1)}$	$\theta^{(2)}$	$\theta^{(3)}$	$\theta^{(4)}$	$\theta^{(5)}$	x_0	x_1	x_2
	Movie	Alice	Bob	Carol	Dave	Eva	*virtual*	*romance*	*action*
$x^{(1)}$	我的少女時代	5	5	0	0	? → 0	1	0.99	0
$x^{(2)}$	派特的幸福劇本	5	?	?	0	? → 0	1	1.0	0.01
$x^{(3)}$	你的名字	?	4	0	?	? → 0	1	0.99	0
$x^{(4)}$	黑暗騎士	0	0	5	4	? → 0	1	0.1	1.0
$x^{(5)}$	神力女超人	0	0	5	?	? → 0	1	0	0.9

$$\min_{\substack{x^{(1)},\dots,x^{(n_m)}\\ \theta^{(1)},\dots,\theta^{(n_u)}}} \frac{1}{2}\sum_{(i,j):\,r(i,j)=1}\left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2 + \underline{\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(\theta_k^{(j)}\right)^2}$$

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & ? & ? \end{bmatrix}$$

$$\theta^{(5)} \in \mathbb{R}^{2+1}$$

$$\frac{\lambda}{2}\left[\left(\theta_0^{(5)}\right)^2 + \left(\theta_1^{(5)}\right)^2 + \left(\theta_2^{(5)}\right)^2\right]$$

$$\theta^{(5)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\left(\theta^{(5)}\right)^T\left(x^{(i)}\right) = 0$$

Solve Cold-Start by **Mean Normalization**

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \mu = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix} \rightarrow Y = \begin{bmatrix} 2.5 & 2.5 & -2.5 & -2.5 & ? \\ 2.5 & ? & ? & -2.5 & ? \\ ? & 2 & -2 & ? & ? \\ -2.25 & -2.25 & 2.75 & 1.75 & ? \\ -1.25 & -1.25 & 3.75 & -1.25 & ? \end{bmatrix}$$

Step 1. Find mean values μ for each movie

Step 2. Subtract $Y = [y^{(i,j)}]$ by $\mu^{(i)}$ (for (i,j): $r(i,j) = 1$)

Step 3. Use the new Y to learn $\theta^{(j)}$ and $x^{(i)}$

Step 4. Make prediction by

$$\hat{y}^{(i,j)} = \left(\theta^{(j)}\right)^T \left(x^{(i)}\right) + \mu^{(i)}$$

Assume $\theta^{(5)} \approx \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow \left(\theta^{(j)}\right)^T \left(x^{(i)}\right) \approx 0 \rightarrow \hat{y}^{(i,j)} \approx \mu^{(i)}$

Recommendation Systems by MF

- For recommender systems, we can represent the relationships between users and items
    - Items: Movies, Songs, Projects, Business, Uses, etc.
- Then the goal is to predict user-item ratings

User	Item	Ratings
1	5	3
1	10	?
1	13	5
…	…	…
u	v	r_{uv}
…	…	…

Number of items

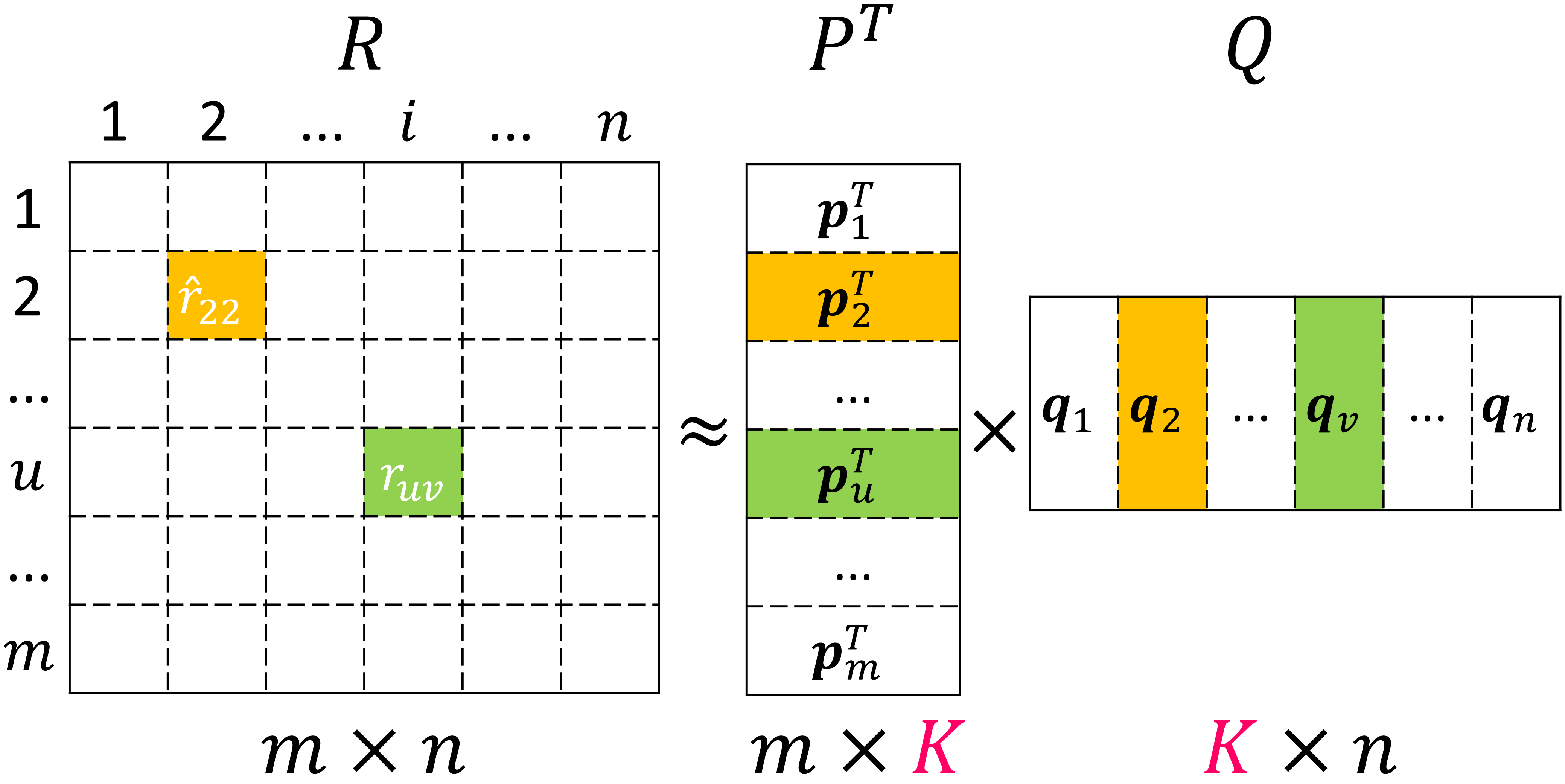
$$\begin{array}{c|cccccc} & 1 & 2 & \dots & i & \dots & n \\ \hline 1 & & & & & & \\ 2 & & ? & & & & \\ \dots & & & & & & \\ u & & & & r_{uv} & & \\ \dots & & & & & & \\ m & & & & & & \end{array}$$

User u gives a rating r_{uv} to item v

Number of users

$m \times n$

Matrix Factorization



K: number of latent dimensions/factors (e.g. topics, categories)

$$\hat{r}_{ui} = \boldsymbol{p}_u^T \boldsymbol{q}_i \qquad \hat{r}_{22} = \boldsymbol{p}_2^T \boldsymbol{q}_2$$

Matrix Factorization

- MF solves the following equation:

$$\min_{P,Q} \frac{1}{2} \sum_{(u,i)\in R} (r_{ui} - \boldsymbol{p}_u^T \boldsymbol{q}_i)^2 + \overset{\text{Regularization term}}{\underline{\frac{\lambda}{2}(\|\boldsymbol{p}_u\|^2 + \|\boldsymbol{q}_i\|^2)}}$$

or: $+\frac{\lambda_1}{2}\|\boldsymbol{p}_u\|^2 + \frac{\lambda_2}{2}\|\boldsymbol{q}_i\|^2$

$$\hat{r}_{ui} = \boldsymbol{p}_u^T \boldsymbol{q}_i \qquad \hat{R} = P^T Q$$

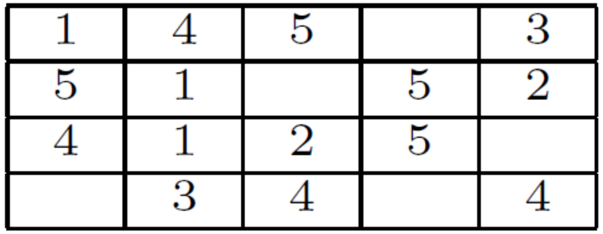
λ: Regularization Parameter

- Stochastic Gradient Descent (SGD) is the most popular optimization method for MF
  - SGD loops over ratings in the training data (existing ratings)

MF with SGD: Example

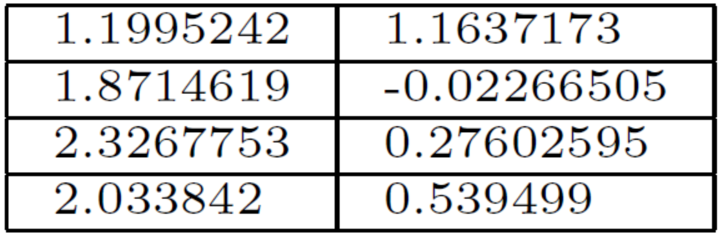
- Hyperparameters: $K = 2, \alpha = 0.1, \lambda = 0.15,$ #iteration=150, initialization $\sim\mathcal{N}(0{,}0.01)$

$R =$



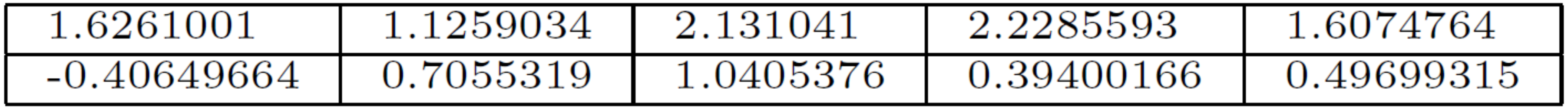
1	4	5		3
5	1		5	2
4	1	2	5	
	3	4		4

$P =$



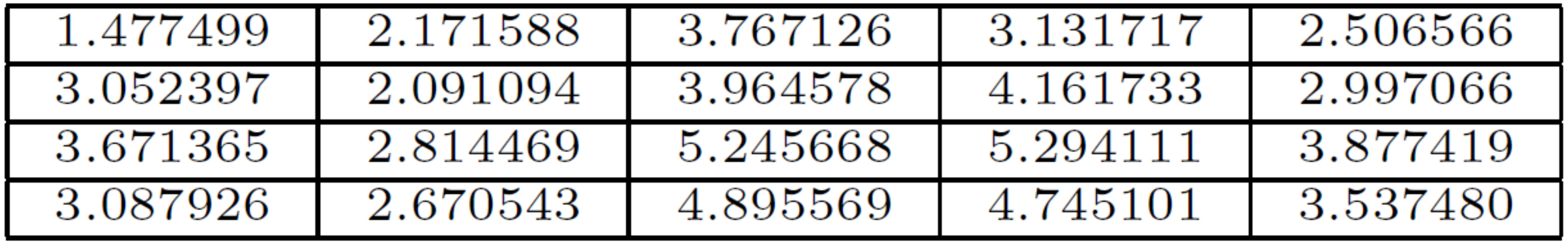
1.1995242	1.1637173
1.8714619	-0.02266505
2.3267753	0.27602595
2.033842	0.539499

$Q^T =$



1.6261001	1.1259034	2.131041	2.2285593	1.6074764
-0.40649664	0.7055319	1.0405376	0.39400166	0.49699315

$\hat{R} = PQ^T =$

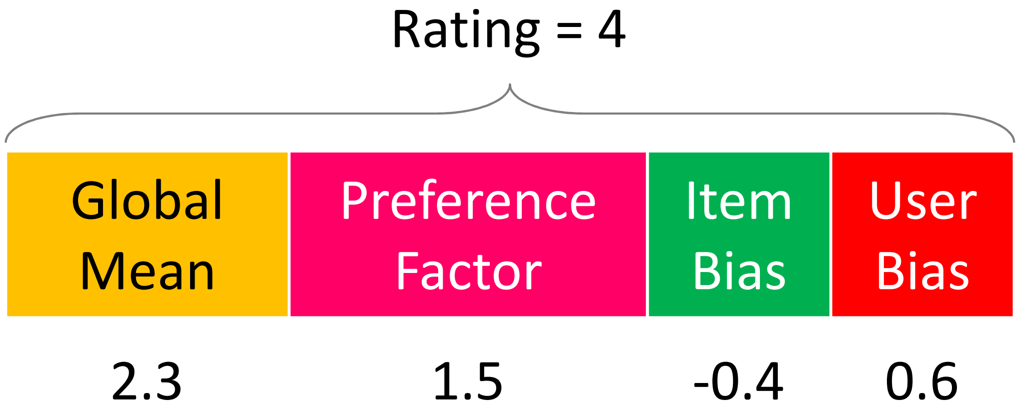


1.477499	2.171588	3.767126	3.131717	2.506566
3.052397	2.091094	3.964578	4.161733	2.997066
3.671365	2.814469	5.245668	5.294111	3.877419
3.087926	2.670543	4.895569	4.745101	3.537480

Adding Biases

- Subtract global mean μ
  - Deal with cold-start users
  - Consider only preference
- Item or user specific rating variations are called biases
  - E.g. Alice rates no movie with more than 2 (out of 5)
  - E.g. Movie X is hyped and rated with 5 only
  - → Some items are significantly higher/lower rated
  - → Some users rate substantially lower/higher
- Matrix factorization needs to allow bias correction
  - Offset per user
  - Offset per movie

New Objective Function with Biases


Rating = 4

Global Mean 2.3

Preference Factor 1.5

Item Bias -0.4

User Bias 0.6

$$\hat{r}_{ui} = \mu + b_u + b_i + \mathbf{p}_u^T \mathbf{q}_i$$

$$\min_{P,Q,b_u,b_i,\mu} \frac{1}{2} \sum_{(u,i)\in R} \left(r_{ui} - (\mu + b_u + b_i + \mathbf{p}_u^T \mathbf{q}_i)\right)^2 + \frac{\lambda}{2}\left(\|\mathbf{p}_u\|^2 + \|\mathbf{q}_i\|^2 + b_i^2 + b_u^2 + \mu^2\right)$$

Apply SGD to find the latent factors

λ can be selected by grid-search

Recap: previously in CF

- μ = mean value over all user-item ratings
- b_x = *(average rating of user x)* – μ
- b_i = *(average rating of item i)* – μ