collaborative filtering ⇩ low rank matrix factorization

vectorization: low rank matrix factorization

movie	user1	user2	user3	user4
movie1	5	5	0	0
movie2	5	?	?	0
movie3	?	4	0	?
movie4	0	0	5	4
movie5	0	0	5	?

$n_m = 5$

$n_u = 4$

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

$y^{(i,j)}$

collaborative filtering → low rank matrix factorization

predicted ratings:

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & 0 \end{bmatrix} \quad \begin{bmatrix} (\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T(x^{(1)}) & \cdots & (\theta^{(n_u)})^T(x^{(1)}) \\ (\theta^{(1)})^T(x^{(2)}) & (\theta^{(2)})^T(x^{(2)}) & \cdots & (\theta^{(n_u)})^T(x^{(2)}) \\ \vdots & \vdots & \vdots & \vdots \\ (\theta^{(1)})^T(x^{(n_m)}) & (\theta^{(2)})^T(x^{(n_m)}) & \cdots & (\theta^{(n_u)})^T(x^{(n_m)}) \end{bmatrix}$$

$$\mathrm{X} = \begin{bmatrix} - & (x^{(1)})^T & - \\ - & (x^{(2)})^T & - \\ - & \vdots & - \\ - & (x^{(n_m)})^T & - \end{bmatrix} \quad \Theta = \begin{bmatrix} - & (\theta^{(1)})^T & - \\ - & (\theta^{(2)})^T & - \\ - & \vdots & - \\ - & (\theta^{(n_u)})^T & - \end{bmatrix} \quad (\theta^{(1)})^T(x^{(1)}) \to \mathrm{X}\Theta^T$$

Let $X = \begin{bmatrix} - & (x^{(1)})^T & - \\ & \vdots & \\ - & (x^{(n_m)}) & - \end{bmatrix}$, $\Theta = \begin{bmatrix} - & (\theta^{(1)})^T & - \\ & \vdots & \\ - & (\theta^{(n_u)}) & - \end{bmatrix}$.

What is another way of writing the following:

$$\begin{bmatrix} (x^{(1)})^T(\theta^{(1)}) & \ldots & (x^{(1)})^T(\theta^{(n_u)}) \\ \vdots & \ddots & \vdots \\ (x^{(n_m)})^T(\theta^{(1)}) & \ldots & (x^{(n_m)})^T(\theta^{(n_u)}) \end{bmatrix} \longrightarrow$$

- $X\Theta$
- $X^T\Theta$
- $X\Theta^T$

Correct Response

- $\Theta^T X^T$

using the learned features to find related movies

for each product i, the algorithm learns feature vector $x^{(i)} \in \mathbb{R}^n$

$x_1 =$ romance, $x_2 =$ action, $x_3 =$ comedy, $x_4 =$ ...

how to find moves j related to movie i:

small $\left\| x^{(i)} - x^{(j)} \right\|$ → movie j and i are 'similar'

5 most similar movies to movie i:

find the 5 movies j with the smallest distance between features $\left\| x^{(i)} - x^{(j)} \right\|$

implementational detail: mean normalization

consideration of an example where a user has not rated any movies

movie	user_1	user_2	user_3	user_4	**user_5**
movie_1	5	5	0	0	**?**
movie_2	5	**?**	**?**	0	**?**
movie_3	**?**	4	0	**?**	**?**
movie_4	0	0	5	4	**?**
movie_5	0	0	5	**?**	**?**

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

$$\min_{\substack{x^{(1)},\dots,x^{(m)} \\ \theta^{(1)},\dots,\theta^{(n_u)}}} \underbrace{\frac{1}{2} \sum_{(i,j):r(i,j)=1} \left(\left(\theta^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right)^2}_{n = 2 \text{ features to learn and } \theta^{(5)} \in \mathbb{R}^2 \qquad \theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}} + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} \left(x_k^{(j)} \right)^2 + \underbrace{\frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} \left(\theta_k^{(j)} \right)^2}_{\text{minimize } \frac{\lambda}{2}\left[\left(\theta_1^{(5)}\right)^2 + \left(\theta_2^{(5)}\right)^2 \right]}$$

minimizing the regularization parameter encourages the algorithm to set $\theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and thus will predict **user_5** as giving all 0 ratings for the examples → $\left(\theta^{(5)}\right)^T \left(x^{(i)}\right) = 0$

this problem is addressed with mean normalization mean normalization

$$Y = \begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \text{ stored in matrix as average } \mu = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix}$$

then subtract off the average rating matrix μ from matrix Y

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

$$\rightarrow \text{learn } \theta^{(j)}, x^{(i)}$$

treat the new matrix Y as the orignial ratings of movies given by users

for user j, on movie i, predict: $\left(\theta^{(j)}\right)^T\left(x^{(i)}\right) + \boldsymbol{\mu_i}$

user₅ : $\theta^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\qquad \left(\theta^{(5)}\right)^T\left(x^{(i)}\right) + \boldsymbol{\mu_i}$

because minimization of the regularization parameter promoted a zero assignment to parameter vector $\theta^{(5)}$, mean normailzation will simplay apply the addition of μ_i vector for each movie rating example predicted for **user₅**

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

mean normalization is a method of feature scaling applied in machine learning algorithms. Unlike other applications of feature scaling, this method did not scale the movie ratings by dividing by the range (max – min value).

this is due to the ratings already having comparable scales (0 to 5 stars)