recommender systems collaborative filtering

working example: predicting movie ratings

users will rate movies using ~~one~~ zero (for computational efficiency) to five stars

movie	user$_1$	user$_2$	user$_3$	user$_4$
movie$_1$	5	5	0	0
movie$_2$	5	**?** *(4)*	**?** *(0)*	0
movie$_3$	**?** *(5)*	4	0	**?** *(0)*
movie$_4$	0	0	5	4
movie$_5$	0	0	5	**?** *(4)*

n_u = number of users

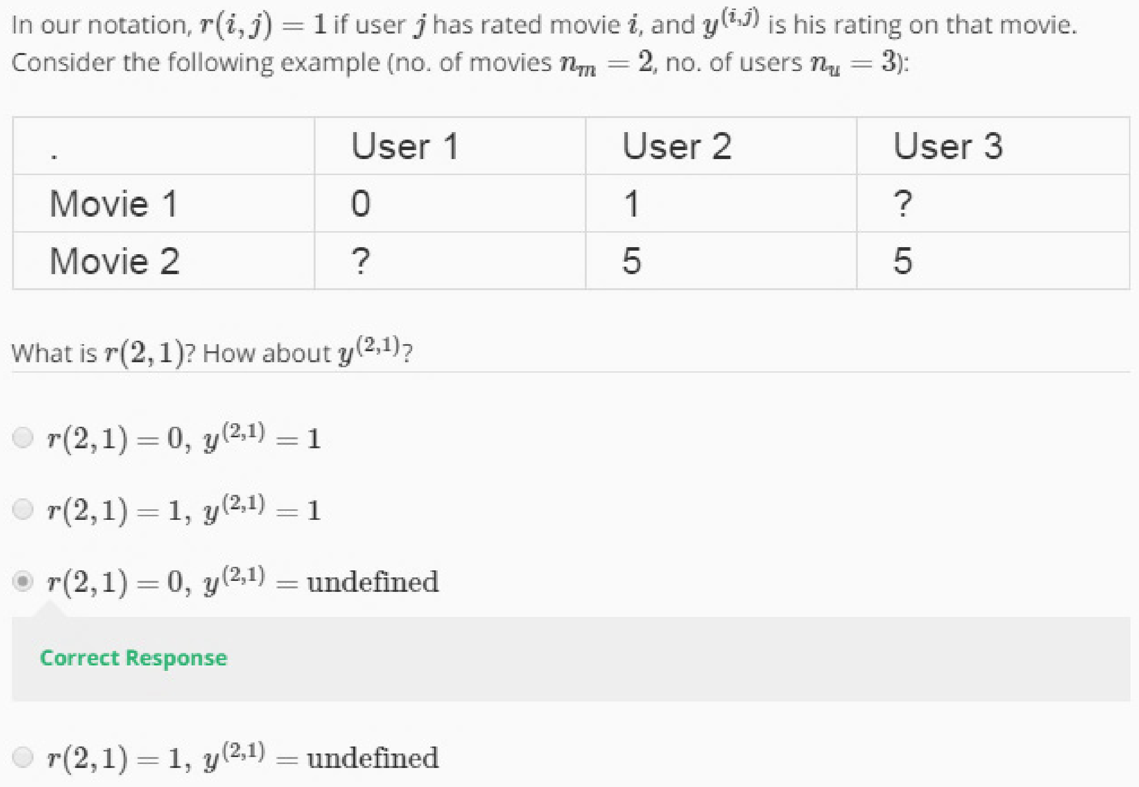
n_m = number of movies

$n_u = \mathbf{4}$

$n_m = \mathbf{4}$

$r(i,j) = 1$ if user j has rated the movie i

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



In our notation, $r(i,j) = 1$ if user j has rated movie i, and $y^{(i,j)}$ is his rating on that movie. Consider the following example (no. of movies $n_m = 2$, no. of users $n_u = 3$):

.	User 1	User 2	User 3
Movie 1	0	1	?
Movie 2	?	5	5

What is $r(2,1)$? How about $y^{(2,1)}$?

- $r(2,1) = 0,\ y^{(2,1)} = 1$
- $r(2,1) = 1,\ y^{(2,1)} = 1$
- [✓ $r(2,1) = 0,\ y^{(2,1)} = \text{undefined}$]

Correct Response

- $r(2,1) = 1,\ y^{(2,1)} = \text{undefined}$

content based recommendation systems

$n_u = 4, n_m = 5$
$x_0 = 1$

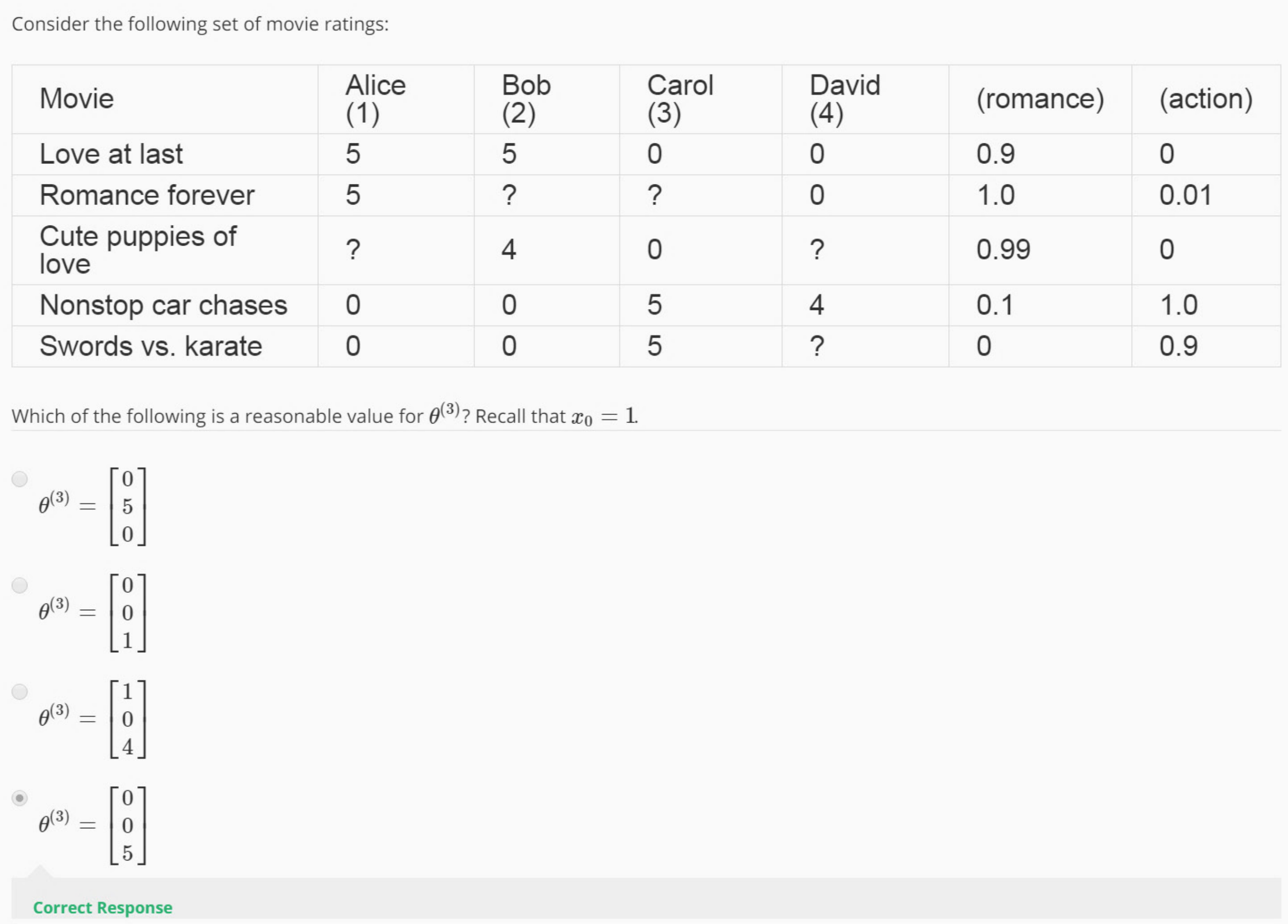
$x^{(1)} = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix}$

movie	user1 $\theta^{(1)}$	user2 $\theta^{(2)}$	user3 $\theta^{(3)}$	user4 $\theta^{(4)}$	x_1	x_2
movie1 $x^{(1)}$	5	5	0	0	0.9	0
movie2 $x^{(2)}$	5	**?**	**?**	0	1.0	0.01
movie3 $x^{(3)}$	**? 4.95**	4	0	**?**	0.99	0
movie4 $x^{(4)}$	0	0	5	4	0.1	1.0
movie5 $x^{(5)}$	0	0	5	**?**	0	0.9

for each user j, learn a parameter $\theta^{(j)} \in \mathbb{R}^3$ $\longrightarrow$ $\theta^{(j)} \in \mathbb{R}^{n+1}$

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

$$x^{(3)} = \begin{bmatrix} 1 \\ 0.99 \\ 0 \end{bmatrix} \leftrightarrow \theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix} \qquad \left(\theta^{(1)}\right)^T x^{(3)} = 5 \times 0.99 = \mathbf{4.95}$$



Consider the following set of movie ratings:

Movie	Alice (1)	Bob (2)	Carol (3)	David (4)	(romance)	(action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

Which of the following is a reasonable value for $\theta^{(3)}$? Recall that $x_0 = 1$.

- $\theta^{(3)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$
- $\theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$
- $\theta^{(3)} = \begin{bmatrix} 1 \\ 0 \\ 4 \end{bmatrix}$
- [✓ $\theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$]

Correct Response

problem formulation

$r(i,j) = 1$ if user j has rated movie i (0 otherwise)
$y^{(i,j)} =$ rating by user j on movie i (if defined)
$\theta^j =$ parameter vector for user j
$x^i =$ feature vector for movie i
for user j, movie i, predicated rating: $\left(\theta^{(j)}\right)^T\left(x^{(i)}\right) \longrightarrow \theta^{(j)} \in \mathbb{R}^{n+1}$
$m^{(j)} =$ number of movies rated by user j

to learn $\theta^{(j)}$:

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

optimization objective:

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

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

to learn $\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(n_u)}$:

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

gradient descent update:

(for $k = 0$)

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

(for $k \neq 0$)

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

collaborative filtering

changing the problem assuming the value of features $x_1, x_2, \dots, x_n$, etc. are unknown:

movie	user1 $\theta^{(1)}$	user2 $\theta^{(2)}$	user3 $\theta^{(3)}$	user4 $\theta^{(4)}$	x_1	x_2
movie1 $x^{(1)}$	5	5	0	0	**?**	**?**
movie2 $x^{(2)}$	5	**?**	**?**	0	**?**	**?**
movie3 $x^{(3)}$	**?**	4	0	**?**	**?**	**?**
movie4 $x^{(4)}$	0	0	5	4	**?**	**?**
movie5 $x^{(5)}$	0	0	5	**?**	**?**	**?**

additional assumption that the values of parameter vectors $\theta^1, \theta^2, \theta^3, \theta^4$ are obtained:

$$\theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}, \theta^{(2)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}, \theta^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}, \theta^{(4)} = \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

the ability to obtain the values for the above parameter vectors make the computation of the unknown features $x_1, x_2, \dots, x_n$, etc. possible

focusing on the first example $x^{(1)}$, the problem sums over all indices θ^j for which we have a feature vector $x_1^{(1)}$:

movie	user1 $\theta^{(1)}$	user2 $\theta^{(2)}$	user3 $\theta^{(3)}$	user4 $\theta^{(4)}$	x_1	x_2
movie1 $x^{(1)}$	5	5	0	0	**? 1.0**	**? 0.0**

$\theta^{(j)}$

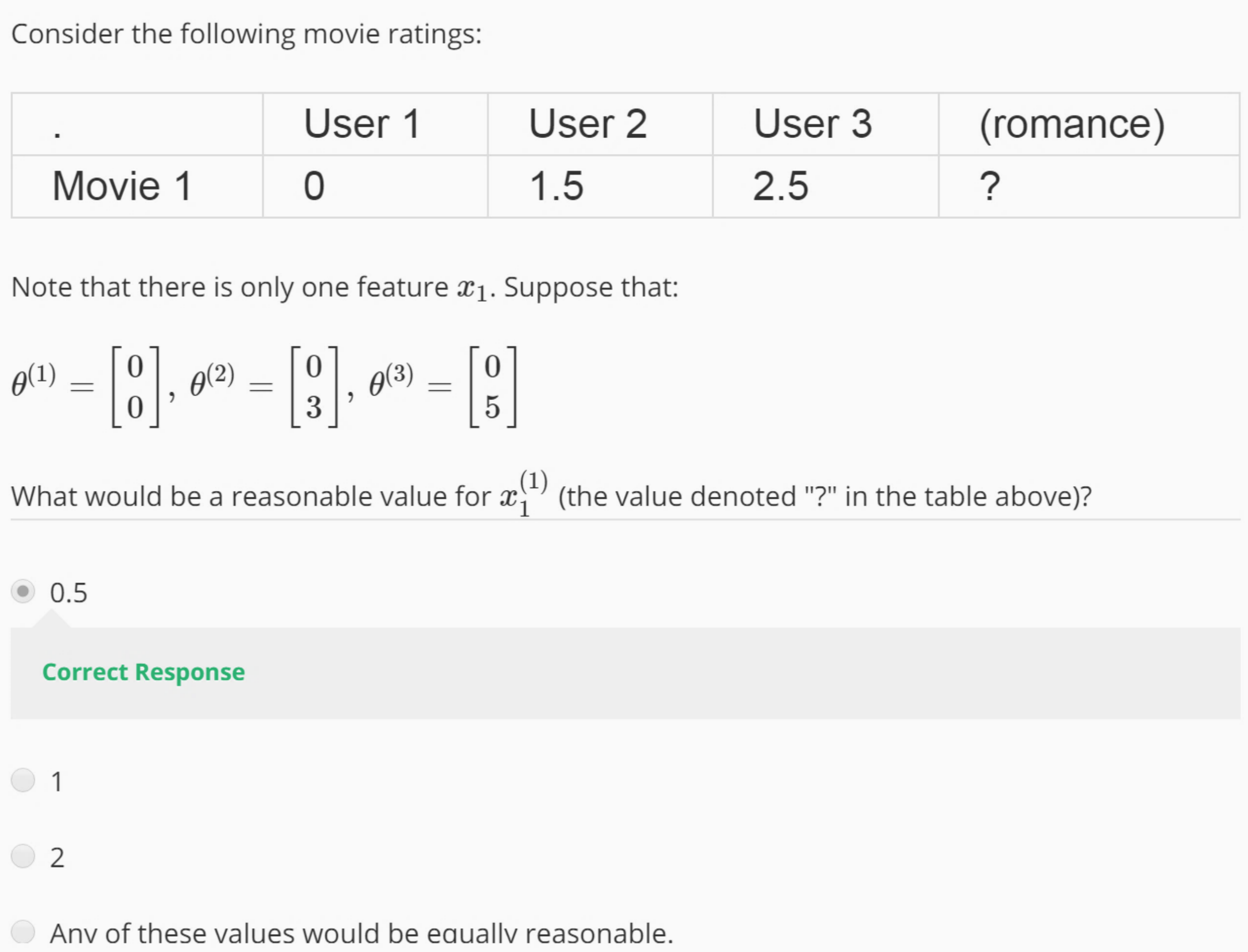
$$\left(\theta^{(1)}\right)^T \left(x^{(1)}\right) \approx 5$$

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

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

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

$$x^{(1)} = \begin{bmatrix} 1 \\ \mathbf{1.0} \\ \mathbf{0.0} \end{bmatrix}$$



Consider the following movie ratings:

.	User 1	User 2	User 3	(romance)
Movie 1	0	1.5	2.5	?

Note that there is only one feature x_1. Suppose that:

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

What would be a reasonable value for $x_1^{(1)}$ (the value denoted "?" in the table above)?

[✓ 0.5]

Correct Response

1

2

Any of these values would be equally reasonable.

optimization algorithm

formalizing the problem of learning $x^{(i)}$

given $\theta^{(1)}, \ldots, \theta^{(n_u)}$, to learn $x^{(i)}$:

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

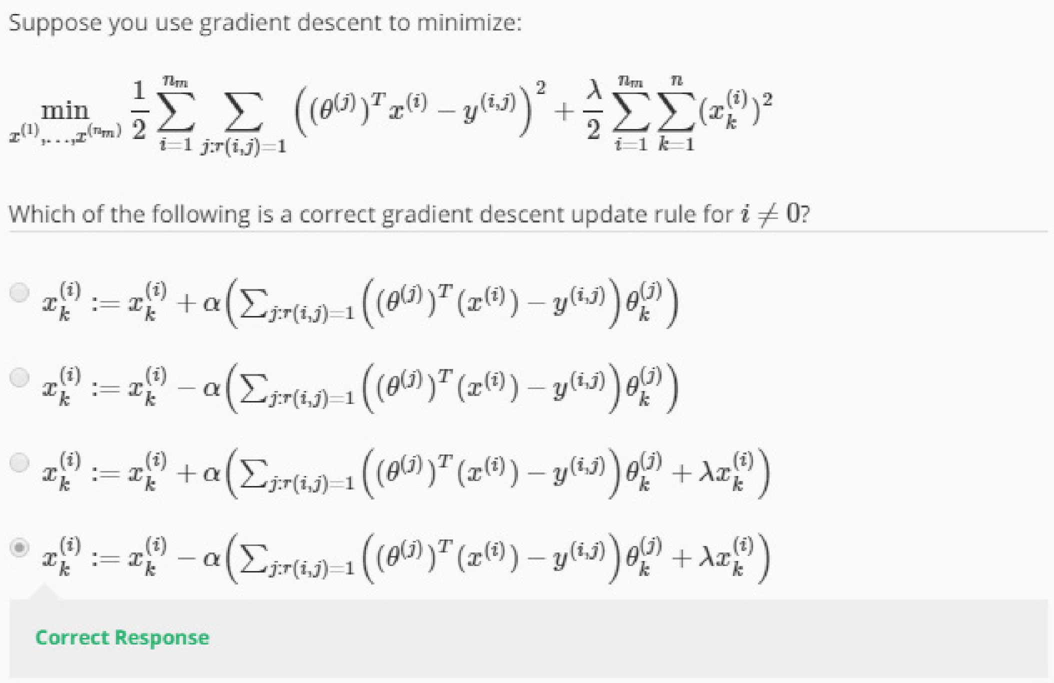
summarize all indices J for which a rating is available for example i and minimize the squared error: choose features $x^{(i)}$ so the predicted value of how J rates example i will be similar to the actual value of $y^{(i,j)}$ that is actually observed in the rating of user J on example i

in summary, the term attempts to choose features $x^{(i)}$ so that for all users J that have provided a rating of example $x^{(i)}$, the algorithm predicts a value for how the user would have rated an example that is not too far from the actual rating observed. regularization can be added to control feature scales.

to expand the algorithm to learn the features for many examples as opposed to single example $x^{(i)}$, an additional summation is added:

given $\theta^{(1)}, \ldots, \theta^{(n_u)}$, to learn $x^{(1)}, \ldots, x^{(n_m)}$:

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


Suppose you use gradient descent to minimize:

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

Which of the following is a correct gradient descent update rule for $i \neq 0$?

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

Correct Response

given the ratings of examples $r^{(i,j)}$ and $y^{(i,j)}$ and features $x^{(i)}, \ldots, x^{(n_m)}$, the parameters can be estimated:

given $x^{(1)}, \ldots, x^{(n_m)}$ (and movie ratings),

can estimate $\theta^{(1)}, \ldots, \theta^{(n_u)}$

given the parameters, the ratings of feature examples can be estimated:

given $\theta^{(1)}, \ldots, \theta^{(n_u)}$,

can estimate $x^{(1)}, \ldots, x^{(n_m)}$

the method is to randomly initialize $\theta^{(i)}$ to learn features $x^{(i)}$ and reiterate to converge to a reasonable set of features for examples and parameters for users:

guess $\theta \rightarrow x \rightarrow \theta \rightarrow x \rightarrow \theta \rightarrow x \rightarrow \cdots$

collaborative filtering refers to the observation that when the algorithm is run with a large set of users, the users are used to *collaborate* in the efforts to learn better features to be used by the algorithm for new users

collaborative filtering algorithm

given features $x^{(1)}, \dots, x^{(n_m)}$, estimate parameters $\theta^{(1)}, \dots, \theta^{(n_u)}$:

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

given parameters $\theta^{(1)}, \dots, \theta^{(n_u)}$, estimate features $x^{(1)}, \dots, x^{(n_m)}$:

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

minimizing $x^{(1)}, \dots, x^{(n_m)}$ and $\theta^{(1)}, \dots, \theta^{(n_u)}$ simultaneously:

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

both θ and x can be minimized simultaneously by combining their expressions seen above. it is important to note that the individual squared error formulas from the above two terms are identical in terms of estimating both $\theta^{(1)}, \dots, \theta^{(n_u)}$ and $x^{(1)}, \dots, x^{(n_m)}$

the first summation is summed over all movies i rated by user j

the second summation is conversely summed over all user j that have rated movie i

thus, the two squared error summations are combined in respects to summing all instances of (i,j) where $r(i,j) = 1$. additionally, the regularization parameters for each expression are comprised in the simultaneous minimization mechanism as they individually regularize both $x_k^{(j)}$ and $\theta_k^{(j)}$ altogether minimizing as one term:

$$\min_{\substack{x^{(1)},\dots,x^{(m)}\\ \theta^{(1)},\dots,\theta^{(n_u)}}} J\left(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}\right)$$

as opposed to the previous approach of random initialization of θ then x then θ etc...

guess $\theta \rightarrow x \rightarrow \theta \rightarrow x \rightarrow \theta \rightarrow x \rightarrow \cdots$

the $x_0 = 1$ convention is removed and thus $x \in \mathbb{R}^n$ as opposed to previously $x \in \mathbb{R}^{n+1}$
the same is assumed for θ_0 and thus $\theta \in \mathbb{R}^n$

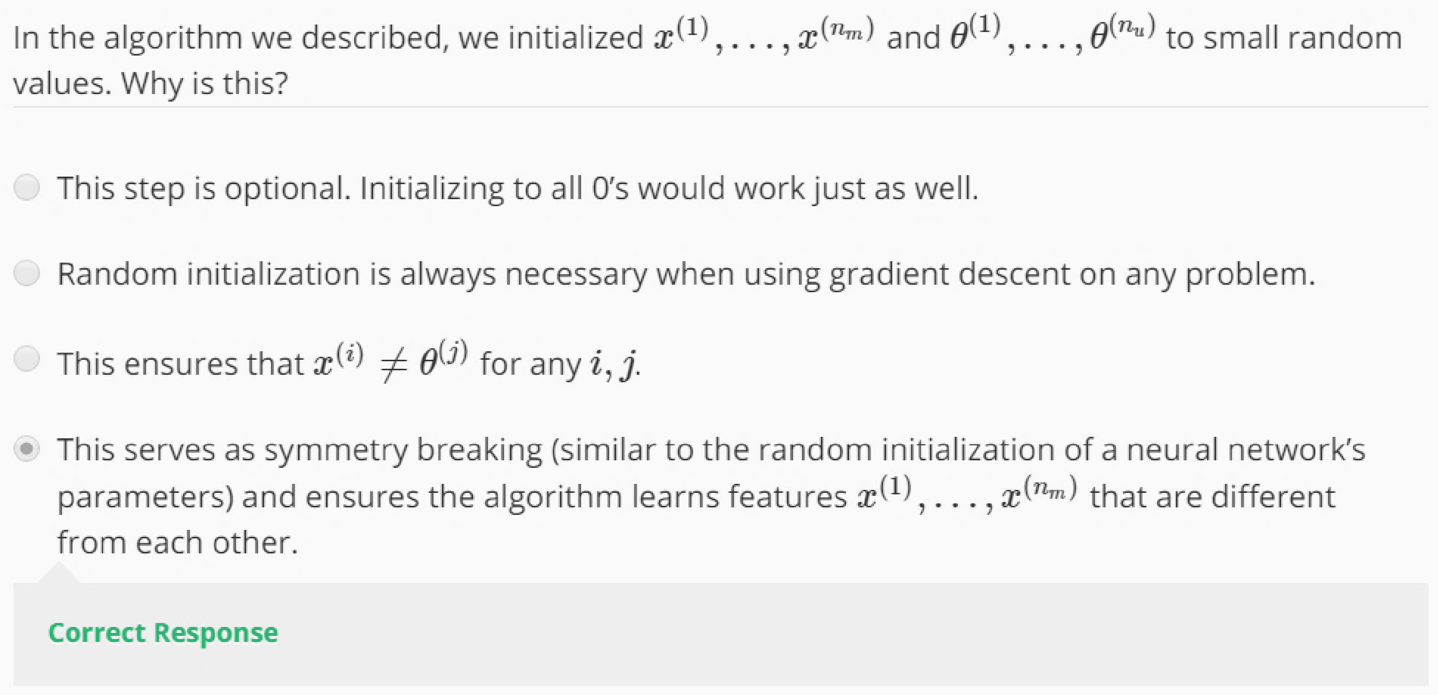
collaborative filtering algorithm application

1. initialize $x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}$ to small random variables
2. minimize $J\left(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}\right)$ using gradient descent
    1. (or an advanced optimization algorithm)
    2. e.g. for every $j = 1, \dots, n_u, i = 1, \dots, n_m$:

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

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

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



In the algorithm we described, we initialized $x^{(1)}, \dots, x^{(n_m)}$ and $\theta^{(1)}, \dots, \theta^{(n_u)}$ to small random values. Why is this?

- This step is optional. Initializing to all 0's would work just as well.
- Random initialization is always necessary when using gradient descent on any problem.
- This ensures that $x^{(i)} \neq \theta^{(j)}$ for any i, j.
- [✓ This serves as symmetry breaking (similar to the random initialization of a neural network's parameters) and ensures the algorithm learns features $x^{(1)}, \dots, x^{(n_m)}$ that are different from each other.]

Correct Response