Movie Ratings

Imagine a movie renter's companies such as YouTube, Netflix, HBO... and want to make suggestion to user based their ratings from 0 to 5 stars.

Movie	User 1	User 2	User 3	User 4
Bates Motel	5	4	1	?
Family guys	1	?	5	2
Dexter	?	3	5	4
Notebook	0	5	?	?

Let's define some parameters before we start recommending:

- n_u: $number\ of\ users$
- n_m: $number\ of\ movies$
- m_j: $number\ of\ movies\ rated\ by\ user j\ to\ learn\ \theta^j$
- $r(i,j)$: $1\ if\ user\ j\ has\ rated\ movie\ i$
- $y^{(i,j)}$: $rating\ \ given\ by\ user\ j\ to\ movie\ i\ (defined\ only\ if\ r(i,j)=1)$

The purpose is to predict value for those movies that has not been rated by user j.

Content-Based Recommendation

Let's say for each movie, we have a set of features such as x_1: action, x_2: romance, …

Movie	x_1: action	x_2: romance
Bates Motel	0.8	0.5
Family guys	0.1	0.3
Dexter	0.99	0
Notebook	0.01	1

Then each movie can be represented by feature vectors. We add $x_0 = 1$, so we will have feature vectors like $x_1 =$ [1 0.8 0.1 0.78 0.01]. When n=2, number of features (do not count x_0). For each user j as rating movie i with $(\theta^j)^T x^i$ stars, as separate linear regression problem. Let's take user 1, which will be associated with θ^1. Let's make a prediction for user 1 for Dexter's movie:

$x^3 = \begin{matrix}1\\0.78\\0\end{matrix}$, and imagine $\theta^1 = \begin{matrix}10\\5\\0\end{matrix}$, so the $(\theta^j)^T x^i = 4.95$

By separating calculating the linear regression, we can make a prediction for the movies for users. How can we learn parameter θ^j, we simply use the least squared error (similar to linear regression) with regularization term:

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

Movie Ratings

θ is n+1-dimensional feature. We can delete 1/m from formula above to simplify it further since its only constant term. We can generalize the equation above for all θ as follow:

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

And we may use gradient descent with learning rate of α. Remember that we have slightly different equation for k equal to 0 and non-zero ks.

Collaborative Filtering

For many movies, we may not have features defining the movies. Instead we have values how much each user like romantic or action movies. For example, user, one rated what he likes as $\theta^1 = [0\ 5\ 0]$. We have these value for each user.

Movie	User 1	User 2	User 3	User 4
Bates Motel	5	4	1	?
Family guys	1	?	5	2
Dexter	?	3	5	4
Notebook	0	5	?	?

So far, we know some ratings from the user shown above as well. From user 1, we know he likes action movies based on θ^1. Also, we know for other user what they like and what's their ratings for the movies. $(\theta^1)^T x^i\ i\ approximately\ 5\ for\ user1$, $(\theta^2)^T x^i$ is approximately 4 for user 2. We can approximate x^i at the end and fills the question marks.

Movie	x_1: action	x_2: romance
Bates Motel	?	?
Family guys	?	?
Dexter	?	?
Notebook	?	?

So here is the optimization algorithm:

- Given $\theta^1, \theta^2, \dots, \theta^{n_u}$
- Learn x^i: $min_{x^j}\ \frac{1}{2m^j}\sum_{i:\,r(i,j)=1}^{last}\left((\theta^j)^T x^i - y^{i,j}\right)^2 + \frac{\lambda}{2m^j}\sum_{k=1}^{n}(x_k{}^j)^2$
- We can generalize it for all x by considering equation above for all x.
- Given θ predict x, then improve value of θ, and again predict x... and it's how to improve the rating by time and learn by collaborating of the users.

Movie Ratings

There is more efficient way of calculating x and θ. Instead of going back and forth and calculating them separately, we can calculate them simultaneously.

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

$$\min J\ (x^1, x^2, \ldots . x^{n_m}, \theta^1, \theta^2, \ldots ., \theta^{n_u}) \text{ for } x^1, x^2, \ldots . x^{n_m}, \theta^1, \theta^2, \ldots ., \theta^{n_u}$$

Previously had a convention that we have an $x_0 = 1$ term. When we're using this kind of approach we have no x_0, So now our vectors (both x and θ) are n-dimensional (not n+1). We do this because we are now learning all the features so there is no need to hard code it to be 1, since it can learn to choose 1 for itself. To wrap it up, here is the summary of the algorithm:

- *Initialize $x^1, x^2, \ldots . x^{n_m}, \theta^1, \theta^2, \ldots ., \theta^{n_u}$ to small random initial values. This serves as symmetry breaking to ensure features are different from each other.*
- Minimize J $(x^1, x^2, \ldots . x^{n_m}, \theta^1, \theta^2, \ldots ., \theta^{n_u})$ using gradient descent and we regularize every parameter
- Having minimized the values, given a user (user j) with parameters θ and movie (movie i) with learned features x, we predict a start rating of $\left(\theta^j\right)^T x^i$.

Low Rank Matrix Factorization

If a user has recently watched a product, can we relate similar products to her/him? For this means, let's put all of the ratings for users into a matrix like $Y = \begin{bmatrix} 5 & 5 & 0 & 0 \\ 5 & ? & ? & 0 \\ ? & 4 & 0 & ? \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 5 & 0 \end{bmatrix}$. where we have 5 movies and 4 users. Predicted rating are:

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

We can vectorize it as simply by $X\theta^T$, where x is a matrix containing the x^i values and θ matrix containing θ^i values. This algorithm is called low rank matrix factorization.

Related Movies

We can use learned features to recommend movies to a user. How can we find movies j related to movie i? If we can fine small difference in features of movie i and j, we can put them in similar recommended movies.

Movie Ratings

Mean Normalization

Imagine a user A that has not rated any movies yet. How collaborative filtering will rate movie for that user?

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

The only terms affected by user A will be the middle term. So, we want to minimize the $\frac{\lambda}{2}[\left({\theta_1}^A\right)^2 + \left({\theta_2}^A\right)^2]$ and we get $\theta^A = [0\ 0]$ for any i. Which mean user A will not like any movies! This does not make any sense. For this problem, we pursue as follow:

- Group all of ratings in a matrix (like above for Y), a column with question mark is for user A that has not rated any movies yet.
- Compute the average rating (μ) each movie obtained and stored in an n_m(vector matrix)
- Subtract the matrix Y from μ. So, the first row of Y will be subtracted from μ_1, and so on (normalizing each movie to have an average rating of 0), and imagine this is a matrix we got from user and use it for further analysis.
    - For our prediction of user j on movie i, predict: $(\left(\theta^j\right)^T x^i + \mu_i$

So, for user A, the $\theta^A = [0\ 0]$, $\left(\theta^j\right)^T x^i = 0$ and for user A, we will predict the μ_i, which is average of movies by other users.