Vectorization: Low Rank Matrix Factorization

Having looked at collaborative filtering algorithm, how can we improve this?, by now we're going to see the vectorization implementation of this algorithm and also see a little bit about other things we can do with this algorithm.

- Given one product, can we determine other relevant products?

- We start by working out another way of writing out our predictions
  - So take all ratings by all users in our example of collaborative filtering and group into a matrix Y

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

So here we have:

- n_m = 5 movies,
- n_u = 4 users
- This matrix Y is going to be a $\mathbb{R}^{5x4}$ matrix
  - Where to index the elements we'll use $y^{(i,j)}$
  - $y^{(i,j)}$ = It's the rating given to movie i by user j

Given Y there's another way of writing out all the predicted ratings:

Predicted ratings:

$$\begin{bmatrix} (\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T(x^{(1)}) & \dots & (\theta^{(n_u)})^T(x^{(1)}) \\ (\theta^{(1)})^T(x^{(2)}) & (\theta^{(2)})^T(x^{(2)}) & \dots & (\theta^{(n_u)})^T(x^{(2)}) \\ \vdots & \vdots & \vdots & \vdots \\ (\theta^{(1)})^T(x^{(n_m)}) & (\theta^{(2)})^T(x^{(n_m)}) & \dots & (\theta^{(n_u)})^T(x^{(n_m)}) \end{bmatrix}$$

- With this matrix of predictive ratings
- We determine the (i, j) entry for every movie

We can define another matrix X, just like matrix we had for linear regression:

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

- Take all the X features for each movie and stack them in rows
- Also define a matrix Θ
  - Take each per user parameter vector and stack in rows

Given matrices X (each row containing features of a particular movie) and Θ (each row containing the weights for those features for a given user), then the full matrix Y of all predicted ratings of all movies by all users is given simply by: $Y = X\Theta^T$.

- We can given this algorithm another name - **Low rank matrix factorization**
  - This comes from the property that the $X\Theta^T$ calculation has a property in linear algebra that we create a **low rank matrix**.

Video Question:

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:

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

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

> $X\Theta^T$

- $\Theta^T X^T$

Finally, having run the collaborative filtering algorithm, we can use the learned features to find related films

Finding related movies

For each product i, we learn a feature vector $x^{(i)} \in \mathbb{R}^n$.

- When we learn a set of features we don't know what the features will be - lets we identify the features which define a film:
  - Say we learn the following features
    - x_1 - romance
    - x_2 - action
    - x_3 - comedy
    - x_4 - $\ldots$
- So we have n features all together
- After we've learned features it's often very hard to come in and apply a human understandable metric to what those features are
  - Usually learn features which are very meaning full for understanding what users like

How to find movies j related to movie i?

- Say we have movie i
  - Find movies j which is similar to i, which we can recommend:

Predicting how similar two movies i and j are can be done using the distance between their respective feature vectors x. Specifically, we are looking for a small value of $||x^{(i)} - x^{(j)}||$.

- i.e. the distance between those two movies
- Provides a good indicator of how similar two films are in the sense of user perception
- NB - Maybe ONLY in terms of user perception

e.g. 5 most similar movies to movie i:

- Find the 5 movies j with the smallest: $||x^{(i)} - x^{(j)}||$.