Matrix Factorization with Stochastic Gradient Descent

James Michelson

July 3, 2017

Abstract

Solving matrix factorization with stochastic gradient descent.

1 Basic Matrix Factorization

Within the context of a recommendation problem, let use denote the set of users U and the set of items D. Let $\mathbf{R}$ be the size of $|U| \times |D|$ and assume we wish to discover K latent features. Thus, we want to discover two matrices $\mathbf{P}$ (a $|U| \times K$ matrix) and $\mathbf{Q}$ (a $|D| \times K$ matrix) such that their product approximates $\mathbf{R}$:

$$\mathbf{R} \approx \mathbf{P} \times \mathbf{Q}^T = \hat{\mathbf{R}} \tag{1}$$

We can consider each row of $\mathbf{P}$ to represent the strength of the association between a user and the latent features. Likewise, each row of $\mathbf{Q}$ can be considered the strength between an item and the latent features.

To get the prediction of an item q_j by u_i, we can take the dot product of the two vectors coresponding to p_i and q_j:

$$\hat{r}_{ij} = p_i^T q_j = \sum_{k=1}^{K} p_{ik} q_{jk} \tag{2}$$

With stochastic gradient descent we can initialize $\mathbf{P}$ and $\mathbf{Q}$ with arbitrarily values and iteratively calculate the predicted difference between $\hat{r}_{ij}$ and r, aiming to find a (local) minimum of the difference.

We can estimate the error as follows:

$$e_{ij}^2 = (r_{ij} - \hat{r}_{ij})^2 \tag{3}$$

$$\hat{r}_{ij} = \sum_{k=1}^{K} p_{ik} q_{kj} \tag{4}$$

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2 \tag{5}$$

$$e_{total} = \sum_{i=1}^{P} \sum_{j=1}^{Q} (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2 \tag{6}$$

To minimize this error, we need to know the gradient at the current values and therefore we differentiate the above equation with respect to these two variables separately. As per (4) above, we can derive the gradient with respect to the error for p_{ik} and q_{kj}:

$$\frac{\partial e_{total}}{\partial p_{ik}} = -2 \sum_{j=1}^{Q} (r_{ij} - \hat{r}_{ij}) q_{kj} = -2 \sum_{j=1}^{Q} e_{ij} q_{kj} \tag{7}$$

$$\frac{\partial e_{total}}{\partial q_{kj}} = -2 \sum_{i=1}^{P} (r_{ij} - \hat{r}_{ij}) p_{ik} = -2 \sum_{i=1}^{P} e_{ij} p_{ik} \tag{8}$$

Thus we have the update rules for both p_{ik} and q_{kj}:

$$p'_{ik} = p_{ik} - \alpha \frac{\partial e_{total}}{\partial p_{ik}} = p_{ik} + 2\alpha \sum_{j=1}^{Q} e_{ij} q_{kj} \tag{9}$$

$$q'_{kj} = q_{kj} - \alpha \frac{\partial e_{total}}{\partial q_{kj}} = q_{kj} + 2\alpha \sum_{i=1}^{P} e_{ij} p_{ik} \tag{10}$$

Where α is constant (termed step-size) which determines the rate at which we approach the local minimum.

2 Regularization

Additionally, we can add a regularization parameter to reduce overfitting. The can be acheived by adding a parameter λ and modifying the squared error as follows:

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2 + \lambda \sum_{k=1}^{K} (||P||^2 + ||Q||^2) \tag{11}$$

This parameter λ is used to ensure that our latent feature vectors for users and items (i.e., P and Q) give approximations of R without containing large numbers. Thus, our new update rules are:

$$p'_{ik} = p_{ik} + 2\alpha(\sum_{j=1}^{Q} e_{ij} q_{kj} - \lambda \sum_{j=1}^{Q} q_{kj}) \tag{12}$$

$$q'_{kj} = q_{kj} + 2\alpha(\sum_{i=1}^{P} e_{ij} p_{ik} - \lambda \sum_{i=1}^{P} p_{ik}) \tag{13}$$

3 Acknowledgements

- QuuxLabs writeup
- John Davis