Final Project: Matrix Factorization for Recommender Systems

Junhao Hua *

June 1, 2015

1 Introduction

This is an implementation document for the final project[1] of *Introduction to Data Mining* instructed by professor Zhongfei Zhang[2]. Broadly speaking, recommender systems are based on one of two strategies: the *content filtering* approach which creates a profile for each user or product to characterize its nature, and the *collaborative filtering* which only relies on past user behavior. The later is quite appealing due to its benefit of domain free. The two primary areas of collaborative filtering are the *neighborhood methods* and *latent factor models*. In this paper, we use the latent factor models for recommender systems and realize them by *matrix factorization*. The algorithm is implemented using MATLAB.

2 Algorithm

Suppose there are$M$ items and N users. let $Y \in \mathbb{R}^{M\times N}$ represent the rating matrix: $Y_{ij} \neq 0$ means that item i is rated by user j with the rating Y_{ij}; $Y_{ij} = 0$ means that item i has not been rated by user j. Let $R \in \{0,1\}^{M\times N}$ be the corresponding indicator matrix. Matrix factorization models [1] map both users and items to a joint latent factor space of dimensionality p, such that user-item interactions are modeled as inner products in that space. Let $X \in \mathbb{R}^{M\times p}$ and $\Theta \in \mathbb{R}^{N\times p}$ be the latent feature matrices of items and users, respectively. Each row vector $X_i \in \mathbb{R}^{1\times p}$ and $\Theta_j \in \mathbb{R}^{1\times p}$ represents the ith item-specific and jth user-specific latent feature vector, respectively. The resulting dot product $X_i\Theta_j^T$ captures the interaction between item i and user j - the user's overall interest in the item's characteristics.

Recent work [2] suggested modelling directly the observed rating only, while avoiding overfitting through a regularized model. The recommender system minimizes the regularized squared error on the set of known ratings:

$$\min_{X,\Theta} \frac{1}{2}||R\cdot(Y - X\Theta^T)||_F^2 + \lambda(||X||_F^2 + ||\Theta||_F^2) \tag{1}$$

*Junhao Hua, PhD candidate, Collage of ISEE, Zhejiang University, huajh7@gmail.com, http://huajh7.com/.

[1] http://10.13.71.118/index.php/projects/.

[2] http://www.isee.zju.edu.cn/dsec/zhongfei_ch.html.

where $\|\cdot\|_F$ is the Frobenius norm, the operator $\cdot$ means the dot product, and λ is the regularization parameter. Using a (stochastic) gradient descent, We can solve this optimization problem with respect to X and Θ simultaneously. Take derivative of cost function (1) w.r.t. X and Θ, we get the gradients

$$\Delta X = R \cdot (X\Theta^T - Y)\Theta + \lambda X, \tag{2a}$$

$$\Delta\Theta = (R \cdot (X\Theta^T - Y))^T X + \lambda\Theta. \tag{2b}$$

The gradient descent update equations for X and Θ are

$$X^{t+1} = X^t - \mu\Delta X^t, \tag{3a}$$

$$\Theta^{t+1} = \Theta^t - \mu\Delta\Theta^t, \tag{3b}$$

where μ is the learning rate, which can be simply set as a constant small number (such as 0.002). After learning the map of the joint latent factor space $\{X, \Theta\}$, the missing ratings in matrix Y can be approximately filled by the $P = X\Theta^T$.

The gradient descent algorithm for matrix factorization is summarized in the following.

Algorithm 1 A Gradient Descent algorithm for Matrix Factorization

Input: Given the rating matrices Y and R. The latent factor matrices X^0, Θ^0 are randomly initialized, and the learning rate $\mu = 0.001$. $MaxIters = 500$.

Set regularization parameter λ appropriately.
for $t \leftarrow 0, 1, 2, \ldots, MaxIters$ **do** ▷ t: time step
  $X^{t+1} = X^t - \mu\big(R \cdot (X^t\Theta^{t,T} - Y)\Theta^t + \lambda X^t\big)$.
  $\Theta^{t+1} = \Theta^t - \mu\big((R \cdot (X^t\Theta^{t,T} - Y))^T X^t + \lambda\Theta^t\big)$.
  if $\frac{\|X^{t+1} - X^t\|_F^2 + \|\Theta^{t+1} - \Theta^t\|_F^2}{\|X^t\|_F^2 + \|\Theta^t\|_F^2} < \epsilon$ **then** ▷ Stop Criterion
    break.
  end if
end for
return $P = X^{t+1}\Theta^{t+1,T}$

3 Experimental Results

3.1 Data Preparation

There are 80k ratings from 943 users ($N = 943$) on 1682 items in the given dataset. The rating value is in the range of $[1, 5]$ in integer. This dataset is partial data from MovieLens 100k dataset[3], which has 100k ratings totally. We use the given 80k ratings data for training and the left 20k ratings for test. The feature number p is set as 10.

3.2 Cold Start

We find that the training data does not leave any history prediction data for 32 items and 0 users. It is difficult to predict such a "new" item. This is called a *cold start* problem. In the latent factor

[3] http://grouplens.org/datasets/movielens/

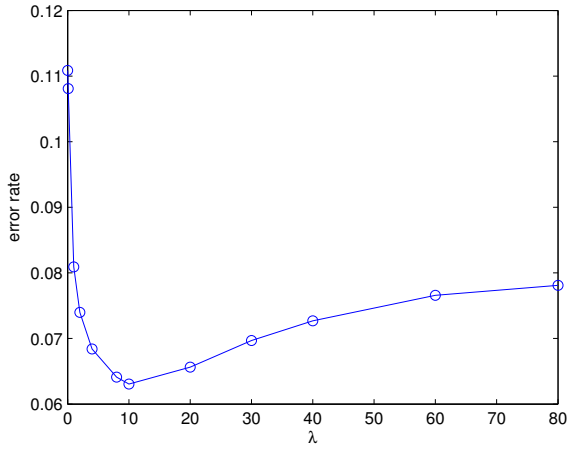

Figure 1: The error rate versus regularization parameter.

model described in the previous section, the corresponding latent factor vector which item/user has no history data will tend to 0 under the role of the regular term. To deal with this problem, we simply use the mean normalization technique. Mathematically, we use $Y - mean(Y)$ to train the model, and use $P = P + mean(Y)$ to predict the ratings.

3.3 The Selection of regularization parameter λ

We evaluate the algorithm performance by exploring a variety of regularization parameter λ, $[0.01, 0.1, 1, 2, 4, 8, 10, 20, 30, 40, 60, 80]$, and choice the one with the lowest error rate. The error rate is defined as

$$error\_rate = \frac{\|R_{test} \cdot (P - Y_{test})\|_F^2}{\|Y_{test}\|_F^2}, \tag{4}$$

where R_{test} and Y_{test} are the responding test rating matrices. As shown in Figure 1, we select $\lambda = 10$ in the following simulation.

3.4 Prediction of ratings

We set the number of feature as 10, and $\lambda = 10$. The program is started from the script file "main.m", and the main function "mf_resys_func", which performs the gradient descent for matrix factorization, is called in this script. The final result is written into the "pred_ratings.txt" in an required format.

References

[1] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[2] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.