EE219 Project 3

Collaborative Filtering

Winter 2017

Xiongfeng Hu, 304753117

Yanming Zhang, 004761717

Cong Peng, 904760493

CONTENT

INTRODUCTION

In this report we study a matrix representation of a Recommendation System that utilizes Collaborative Filtering. "Collaborative Filtering" refers to methods of predicting a user's opinion on an entity using other users' opinion. These methods are based on the notion that there exist other users with similar behavior as the target user and finding them and observing their actions will reveal information that we could use to predict the target user's behavior. The problem we will study in detail is the following: based on feedback data from users indicating a rating of items, we construct a matrix, where the element corresponds to the rating given by user on item.

We would like to extrapolate this matrix to infer which unrated items that a user will likely enjoy. Therefore, we adopt the following matrix factorization: $R_{n\times m} = U_{n\times k}V_{k\times m} + \varepsilon_{n\times m}$ where the rows of U are vectors that characterize a particular user, the columns of V characterize a particular item, and represents the error, parameterized by a factor k .

The factorization algorithm used throughout this report is the *Alternating Least Squares Method* which seeks to minimize the following cost function:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(r_{ij} - (UV)_{ij})^2 \tag{1.1}$$

We will build the recommendation system based on a dataset consisting of 100K movie ratings collected by GroupLens[1].

PART 1: A SIMPLE FACTORIZATION USING ALS

Our Recommendation System uses the NMF MATLAB toolbox to perform the matrix factorization that minimizes the cost function in equation (1.1). In this section, we used the NMF tools to factorize R . The metric we used to determine the closeness of the factorization was the squared error (1.1). We only considered entries that of the prediction matrix that corresponded to actual prediction the user actually made, so equation (1.1) was modified by an additional weight matrix $\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}(r_{ij} - (UV)_{ij})^2$ where $w_{ij} = 1$ if user i rated movie j and 0 otherwise. We also varied the value of parameter k to determine the optimal size of matrices U and V . Small values of k encode less information which makes the reconstruction of less accurate but large values of could result in over-fitting the dataset. The results for squared error versus parameter are shown in Table 1 below:

Table 1: Squared Error vs k

k	10	50	100
sq. error	2.49e+04	9.02e+03	2.73e+03

Then, we apply the same cost function but this time, we reverse the roles of the (weight) matrices in the factorization step. We again applied the ALS factorization algorithm for different values of parameter and measured the squared error. The results are summarized below in Table 2.

Table 2: Squared error vs k

k	10	50	100
sq. error	5.71E+04	2.50Ee+03	1.18e+03

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

PART 2: 10-FOLD CROSS VALIDATION

In this section, we show results from the same system built using 10-Fold Cross validation. The original dataset is transformed via random permutation, then divided into a testing and training set. The training set is used to construct the matrix R which is factorized into U and V. This process is repeated for ten iterations (10 folds). The values of the prediction matrix are validated against the ratings in the testing set. We define a new metric based on this prediction error shown below in equation (2.2).

$$\sum_{i,j \in S} | r_{ij}^{\rho} - r_{ij} | \tag{2.2}$$

Where S is the testing set and R^{ρ} is the prediction matrix defined by U and V. The results of the 10-Fold Cross Validation is shown below in Table 3.

Table 3: Ten-Fold Cross Validation Prediction Error

fold	error
1	1.1983
2	1.1419
3	1.0866
4	1.0951
5	1.1185
6	1.2494
7	1.0880
8	2.1028
9	1.1279
10	1.0941

The minimum prediction error occurred in the 3th fold and the maximum error occurred in the 8th fold. The average prediction error across all ten folds is 1.245.

This process is repeated for the weighted ALS factorization matrices from Part 1 (factorize using the {0, 1} matrix). These results are shown below (note that the errors were normalized by the weight matrix to be comparable with the previous results).

Table 4: Prediction Error for ALS on 1/0 Matrix

fold	error
1	2.5384
2	2.5372
3	2.5362
4	2.52
5	2.5397
6	2.5404
7	2.5248
8	2.5422
9	2.5174
10	2.539

In this case, the 8th fold provided the largest Prediction error while the 9th fold had the lowest.

PART 3: PRECISION AND RECALL

Next, we apply the 10-Fold Cross Validation techniques developed in Part 3 to a classification problem where we conclude that ratings above a certain threshold are movies the user "Liked". We use this to quantify the Precision and Recall metrics which are defined as follows:

$$precision = \frac{t_p}{t_p + f_p} \quad (4.1)$$

$$recall = \frac{t_p}{t_p + f_n} \quad (4.2)$$

A true positive (tp) is defined as an element in the matrix R where $r_{ij} > t$ AND $r^{\rho}{}_{ij} > t$ for some threshold t. A false positive (fp) is defined as an element of R where $r_{ij} \leq t$ AND $r^{\rho}{}_{ij} \leq t$ and so on.

For each of the 10 folds, we sweep the threshold between 1 and 5 and obtain a value for precision and recall. We average these values across the folds and plot Precision vs Recall parameterized lby threshold. This result is shown below in Figure 1.

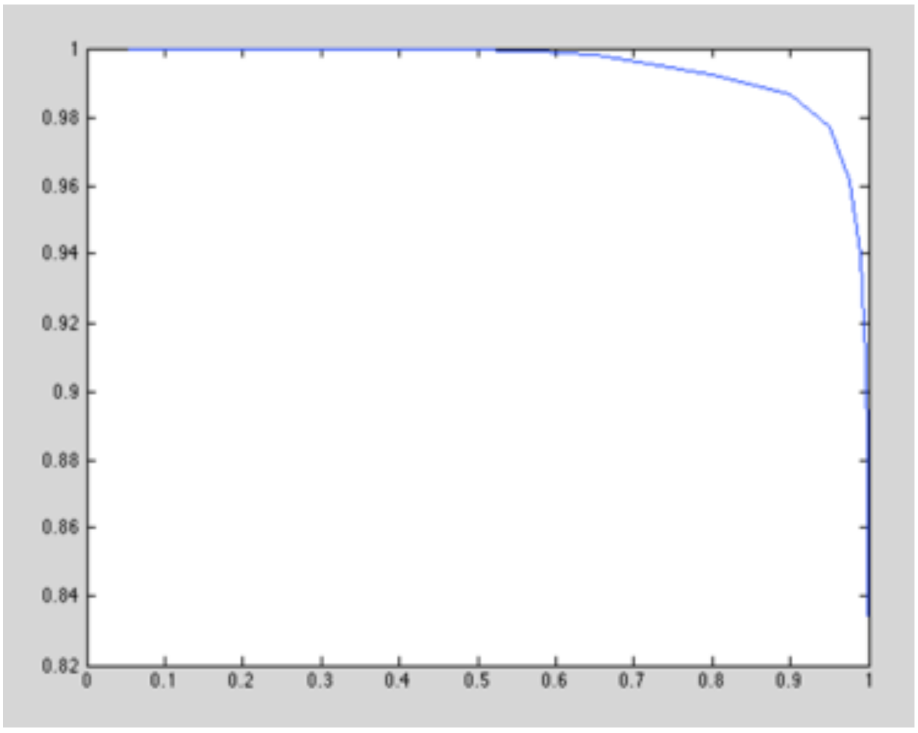

Figure 1: Precision vs Recall for Standard ALS Factorization

We do not repeat the process for the modified factorization using the non-trivial weight matrix from Part 1. Performing classification on this matrix would not produce useful results since we train on a matrix consisting only of 1's and 0's, to which classification of "likes" versus "dislikes" would be meaningless. Therefore, we proceed to the next part of the report, where we analyze the improvement on the ALS algorithm based on introducing a regularization factor.

PART 4: IMPROVED ALS WITH REGULARIZATION

In this section, we introduce an improvement on the ALS cost function, called a Regularization term λ which reduces the effect of over-fitting the data. The new cost function is modeled by the following Equation:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}(r_{ij}-(UV)_{ij})^2+\lambda(\sum_{i=1}^{n}\sum_{j=1}^{m} u_{ij}^{\ 2}+\sum_{i=1}^{n}\sum_{j=1}^{m} v_{ij}^{\ 2}) \qquad (5.1)$$

The NMF code was modified to implement a new update rule based on this cost function and the results were re-evaluated for Parts 1 for $\lambda = 0.01, 0.1, 1$.

Table 5: Squared Error vs k and λ (Part 1)

λ\k	10	50	100
0.01	57420	57363	58712
0.1	25286	24878	26543
1	11636	11738	13549

Again, the process is repeated on the matrix Using Non-Trivial Weights.

Table 6: Squared Error vs k and λ (Part 1)

λ\k	10	50	100
0.01	65384	66265	66462
0.1	31888	31772	32271
1	17027	17329	18012

PART 5: MOVIE RECOMMENDATIONS

In this part, the dataset we use is different from the previous problems, the dataset we played with is in the package '*ml-latest-small'*.

In this problem, we first convert R to a matrix where the entries are 1 for available data points and 0 for the missing ones. Then we perform a 10 cross validation on R, then we sort the ratings for each users in a descending order to get the top L movies recommendations. In this case, L = 5.

On calculation to average precision, we first set a threshold to make a correct decision. We count the number of movies we consider in each fold, and compute the number of select movies recommendations. The average precision in our case is 0.94 given L = 5. The table below shows the different precision given different L values.

Table 7: Precision over different L

L:	Precision: (%)
1	95.59
2	95.64
3	95.45
4	96.55
5	96.45

According to the requirements, we also need to find hit rate and false rate of our algorithm. Hit rate represents what fraction of the test movies liked by the users are suggested by our system, and the false rate represents the opposite. We set L starting from 1 to 5, and plot these values as points in a two-dimensional space with hit rate on the y axis and false rate on the x axis. The plot is given as follows.

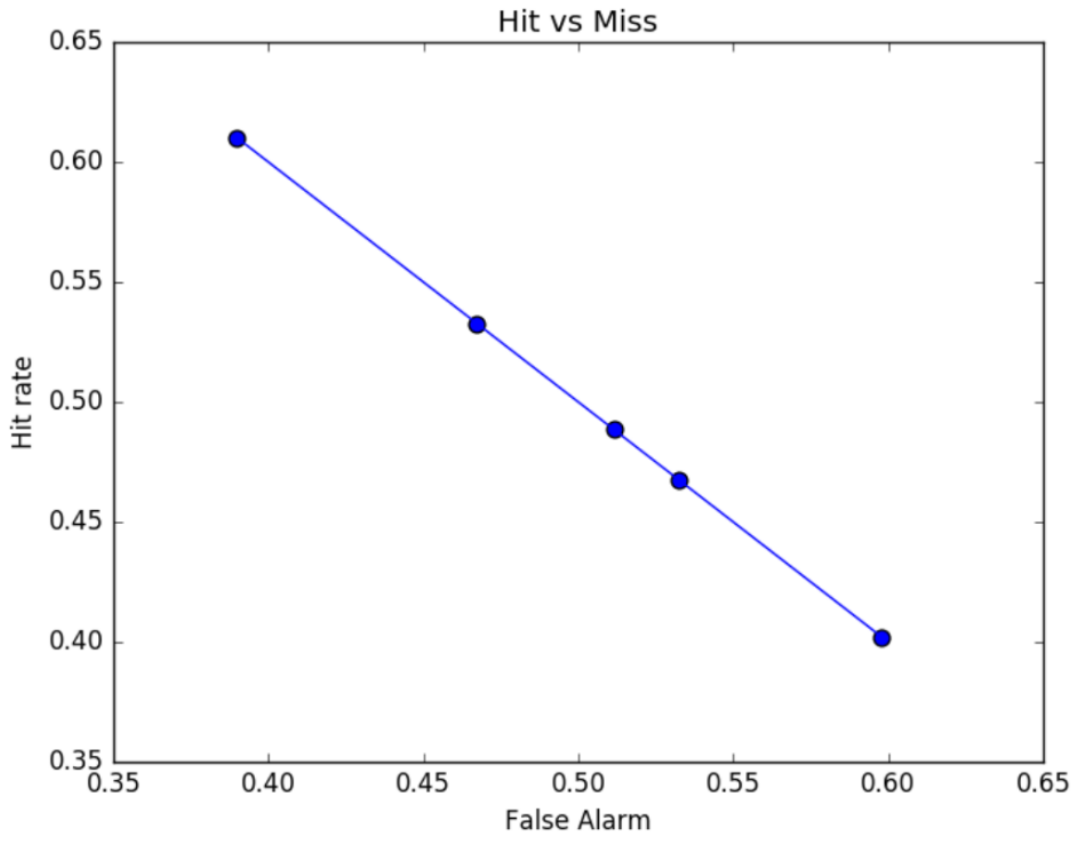

Figure 2: Hit rate vs Miss rate