STAT 8056

Homework

He Zhou (zhou1354@umn.edu)

April 10, 2022

Question 1. (Recommender systems): Define user-item peference matrix: $R = [r_{ui}]_{U\times M}$, with rating score r_{ui}. When R is complete, denote $\Theta_{U\times M} = A_{U\times k}B_{k\times M}$, $\|\cdot\|_F$ is Frobenius norm, and $k \le \min\{U, M\}$. Let $\|\Theta\|_* = \sum_u \sigma_u$ be the nuclear norm and σ_u's be the singular values of Θ. Prove that

$$\arg\min_{A,B} \frac{1}{2}\|R - AB\|_F^2 + \lambda(\|A\|_F^2 + \|B\|_F^2) \tag{1}$$

is equivalent to

$$\arg\min_{\Theta} \frac{1}{2}\|R - \Theta\|_F^2 + \lambda\|\Theta\|_*, \tag{2}$$

where $\lambda > 0$ is a tunning parameter.

Proof: First prove a proposition (**Proposition 1.**) which is given in Mazumder et al. (2010).

Proposition 1. *Suppose $W_{m\times m}$, $\tilde{W}_{n\times n}$, $Z_{m\times n}$ satisfies*

$$\begin{pmatrix} W & Z \\ Z^T & \tilde{W} \end{pmatrix} \ge 0.$$

Then $tr(W) + tr(\tilde{W}) \ge 2\|Z\|_*$.

Proof. Let $Z = L\Sigma R^T$ be the SVD of Z, where $\Sigma = diag\{\sigma_1, \cdots, \sigma_r\}$, $r = rank(Z) \le \min\{m, n\}$, $L : m \times r$, $R : n \times r$, and $L^T L = R^T R = I_r$. Define

$$M_1 = \begin{pmatrix} LL^T & -LR^T \\ -RL^T & RR^T \end{pmatrix}, \; M = \begin{pmatrix} W & Z \\ Z^T & \tilde{W} \end{pmatrix}.$$

Notice that $M_1 = \begin{pmatrix} L \\ -R \end{pmatrix}\begin{pmatrix} L \\ -R \end{pmatrix}^T$, then its a positve semidefinite matrix. Since $M \ge 0$, we have

$$\begin{pmatrix} L \\ -R \end{pmatrix}^T M \begin{pmatrix} L \\ -R \end{pmatrix} \ge 0,$$

which implies that its egenvalues are all non-negative. Thus,

$$\begin{aligned} tr(M_1 M) &= tr\left(\begin{pmatrix} L \\ -R \end{pmatrix}\begin{pmatrix} L \\ -R \end{pmatrix}^T M\right) \\ &= tr\left(\begin{pmatrix} L \\ -R \end{pmatrix}^T M \begin{pmatrix} L \\ -R \end{pmatrix}\right) \\ &\ge 0. \end{aligned}$$

On the other hand, we have

$$M_1 M = \begin{pmatrix} LL^T W - LR^T Z^T & LL^T Z - LR^T \tilde{W} \\ -RL^T W + RR^T Z^T & -RL^T Z + RR^T \tilde{W} \end{pmatrix}$$

then by the above SVD of Z, $Z = L\Sigma R^T$, we have

$$\begin{aligned} 0 \le tr(M_1 M) &= tr(LL^T W - LR^T Z^T) + tr(-RL^T Z + RR^T \tilde{W}) \\ &= tr(LL^T W) - tr(LR^T Z^T) - tr(RL^T Z) + tr(RR^T \tilde{W}) \\ &= tr(L^T WL) - tr(LR^T R\Sigma L^T) - tr(RL^T L\Sigma R^T) + tr(R^T \tilde{W} R) \\ &= tr(L^T WL) - tr(LI_r \Sigma L^T) - tr(RI_r \Sigma R^T) + tr(R^T \tilde{W} R) \\ &= tr(L^T WL) - tr(L\Sigma L^T) - tr(R\Sigma R^T) + tr(R^T \tilde{W} R) \\ &= tr(L^T WL) - tr(\Sigma L^T L) - tr(\Sigma R^T R) + tr(R^T \tilde{W} R) \\ &= tr(L^T WL) - tr(\Sigma) - tr(\Sigma) + tr(R^T \tilde{W} R) \\ &= tr(L^T WL) + tr(R^T \tilde{W} R) - 2\|Z\|_*. \end{aligned}$$

So it remains to show that $tr(L^T WL) \le tr(W)$ and $tr(R^T \tilde{W} R) \le tr(\tilde{W})$. This is because let $\mathcal{L} = \begin{pmatrix} L & \tilde{L} \end{pmatrix}$ be $m \times m$ orthogonal matrix expanded from L, i.e., $\mathcal{L}^T \mathcal{L} = \mathcal{L}\mathcal{L}^T = I_m$, then

$$\begin{aligned} tr(W) &= tr(\mathcal{L}\mathcal{L}^T W) \\ &= tr(\mathcal{L}^T W \mathcal{L}) \\ &= tr\left[\begin{pmatrix} L^T \\ \tilde{L}^T \end{pmatrix} W \begin{pmatrix} L & \tilde{L} \end{pmatrix} \right] \\ &= tr\left[\begin{pmatrix} L^T WL & L^T W\tilde{L} \\ \tilde{L}^T WL & \tilde{L}^T W\tilde{L} \end{pmatrix} \right] \\ &= tr(L^T WL) + tr(\tilde{L}^T W\tilde{L}) \\ &\ge tr(L^T WL), \end{aligned}$$

where the last inequality is because $\begin{pmatrix} W & Z \\ Z^T & \tilde{W} \end{pmatrix} \succeq 0$, thus, $W \succeq 0$, which implies that $\tilde{L}^T W\tilde{L} \succeq 0$. Similarly, with $\tilde{W} \succeq 0$, we have $tr(\tilde{W}) \ge tr(R^T \tilde{W} R)$. Therefore, we proved that

$$0 \le tr(W) + tr(\tilde{W}) - 2\|Z\|_*,$$

as desired. □

Proof of Question 1: Define the objective functions of optimization problem (1) as

$$f_1(A, B, \lambda) = \frac{1}{2}\|R - AB\|_F^2 + \lambda(\|A\|_F^2 + \|B\|_F^2) \tag{3}$$

and the objective functions of optimization problem (2) as

$$f_2(\Theta, \lambda) = \frac{1}{2}\|R - \Theta\|_F^2 + \lambda\|\Theta\|_*. \tag{4}$$

- First show that $\forall \lambda \geq 0$,

$$\min_{A,B} f_1(A, B, \lambda) \leq \min_{\Theta} f_2(\Theta, 2\lambda) \tag{5}$$

Proof. If Θ^* is the minimizer of optimization problem (2), then do SVD of Θ^*: $\Theta^* = U^*\Sigma^* V^{*T}$, where $\Sigma^* = diag\{\sigma_1^*, \cdots, \sigma_r^*\}$, $\sigma_i^* > 0$, $U^* : U \times r$, $V^* : M \times r$, $r = rank(\Theta^*) \leq \min\{U, M\}$, and $U^{*T}U^* = V^{*T}V^* = I_r$. Let

$$\tilde{A} = U^*\Sigma^{*1/2}, \ \tilde{B} = \Sigma^{*1/2}V^{*T}$$

where $\Sigma^{*1/2} = diag\{\sqrt{\sigma_1^*}, \cdots, \sqrt{\sigma_r^*}\}$, then $(\tilde{A}, \tilde{B})$ is a feasible solution to optimization problem (1) and we have

$$\begin{aligned}
f_1(\tilde{A}, \tilde{B}, \lambda) &= \frac{1}{2}\|R - \tilde{A}\tilde{B}\|_F^2 + \lambda(\|\tilde{A}\|_F^2 + \|\tilde{B}\|_F^2) \\
&= \frac{1}{2}\|R - \Theta^*\|_F^2 + \lambda(tr(U^*\Sigma^* U^{*T}) + tr(V^*\Sigma^* V^{*T})) \\
&= \frac{1}{2}\|R - \Theta^*\|_F^2 + \lambda(tr(U^{*T}U^*\Sigma^*) + tr(V^{*T}V^*\Sigma^*)) \\
&= \frac{1}{2}\|R - \Theta^*\|_F^2 + \lambda(tr(\Sigma^*) + tr(\Sigma^*)) \\
&= \frac{1}{2}\|R - \Theta^*\|_F^2 + 2\lambda\|\Theta^*\|_* \\
&= f_2(\Theta^*, 2\lambda).
\end{aligned}$$

where the second equality is because $\|X\|_F^2 = tr(XX^T) = tr(X^TX)$. Thus, given a tunning parameter $\lambda \geq 0$,

$$\min_{A,B} f_1(A, B, \lambda) \leq f_1(\tilde{A}, \tilde{B}, \lambda) = f_2(\Theta^*, 2\lambda) = \min_{\Theta} f_2(\Theta, 2\lambda),$$

as desired in inequality (5). □

- Show that $\forall \lambda \geq 0$,

$$\min_{\Theta} f_2(\Theta, 2\lambda) \leq \min_{A,B} f_1(A, B, \lambda). \tag{6}$$

Proof. On the other hand, if (A^*, B^*) is the minimizer of optimization problem (1), let $\tilde{\Theta} = A^*B^*$. Then

$$\begin{pmatrix} A^*A^{*T} & \tilde{\Theta} \\ \tilde{\Theta}^T & B^{*T}B^* \end{pmatrix} = \begin{pmatrix} A^* \\ B^{*T} \end{pmatrix} \begin{pmatrix} A^* \\ B^{*T} \end{pmatrix}^T \geq 0$$

By the **Proposition 1**, we have

$$\begin{aligned}
2\|\tilde{\Theta}\|_* &\leq tr(A^*A^{*T}) + tr(B^{*T}B^*) \\
&= \|A^*\|_F^2 + \|B^*\|_F^2
\end{aligned}$$

Then

$$\begin{aligned} f_2(\tilde{\Theta}, 2\lambda) &= \frac{1}{2}\|R - \tilde{\Theta}\|_F^2 + 2\lambda\|\tilde{\Theta}\|_* \\ &\leq \frac{1}{2}\|R - \tilde{\Theta}\|_F^2 + \lambda\left(\|A^*\|_F^2 + \|B^*\|_F^2\right) \\ &= f_1(A^*, B^*, \lambda). \end{aligned}$$

Thus given a tunning parameter $\lambda \geq 0$,

$$\min_{\Theta} f_2(\Theta, 2\lambda) \leq f_2(\tilde{\Theta}, 2\lambda) = f_1(A^*, B^*, \lambda) = \min_{A,B} f_1(A, B, \lambda),$$

as desired in inequality (6). □

Combining two inequalities (5) and (6), we've shown that for any given tunning parameter $\lambda \geq 0$,

$$\min_{A,B} \frac{1}{2}\|R - AB\|_F^2 + \lambda(\|A\|_F^2 + \|B\|_F^2) = \min_{\Theta} \frac{1}{2}\|R - \Theta\|_F^2 + \lambda\|\Theta\|_*,$$

i.e., the two optimization problems (1) and (2) are equivalent. This ends the proof of the question.

Question 2. (Movielens data): Movielens data are comprised of movie ratings and are available at https://grouplens.org/datasets/movielens/. The data was collected through the MovieLens website (movielens.umn.edu) during the seven-month from September 19th, 1997 throught April 22nd, 1998. This data has been cleaned up - users who had less than 20 ratings or did not have complete demographic information were removed from this data set. Detailed descriptions of the data file can be found at the end of README.txt on the website. The 100K MovieLens data consists of 1,000,000 anonymous ratings on a five-star scale from 1,000 users on 1,700 movies. Here are four categorical covariates and one continuous covariate, including four user-related covariates, gender, age, occupation, and zip-code, as well as one content-related covariate, genres. For simplicity, we treat gender, age, and occupation as continuous variabls, inaddition to 19 different movie genres that are reparametrized into 19 binary covariates encoding if a certain movie belongs to a particular genre, where only the first digit is used for the zip-code.

1. Use any method of your choice (SVM, boosting, deep learning) to predict the preference scores of each user. There are five pairs of training and testing data sets for, denoted by (u1.base, u1.test), $\cdots$, (u5.base, u5.test) there. It is natural to use 5-fold cross-validation for training and testing. Compute the root mean square error based on cross-validation.

2. For this dataset, many ratings are not observed or missing. Do you have any evidence to suggest that missing does not occur at random? (Hint: Supply suitable plots to argue.) If the goal is to build a recommender system with high prediction accuracy, then demonstrate that the predictive performance can be enhanced by incorporating the missing pattern.

Solution:

The '*u.user*' file contains the demographic information about the users including `age`, `gender`, `occupation` and `zip-code` and there are $U = 943$ users in total. The '*u.item*' file contains the information about the movies including `movie-title`, `release-date`, and one-hot encodings of 19 fields of genre. There are $I = 1682$ movies in total. The '*u.data*' file contains the 100,000 ratings by 943 users on 1682 items. Each user has rated at least 20 movies.

Method: Low Rank Matrix Factorization with Neural Network

The matrix factorization model (Mazumder et al., 2010) maps users and movies to a joint latent factor space such that user-movie interactions are models as inner products. Given the number of latent factors K, assume that $\boldsymbol{b}_i \in \mathbb{R}^K$ is the intrinsic score for movie i, and $\boldsymbol{a}_u \in \mathbb{R}^K$ is the intrinsic score for user u. Then the model assumes that the expectation of user u's rating for movie i is equal to

$$\mathbb{E}[r_{ui}] = \mu + a_u^T b_i$$

Based on the matrix factorization model, we can further train a neural network by adding a *Fully Connected* hidden layer to introduce non-linearity as well as some *Dropout* layers to introduce intrinsic regularization to the neuron network model. The architecture of the model is as follows:

- Given the number of latent factors $K = 50$, generate the low-rank embedding for users $A \in \mathbb{R}^{U \times K}$ and the low-rank embedding for movies $B \in \mathbb{R}^{I \times K}$ where rows of A are intrinsic scores $\boldsymbol{a}_u$ and rows of B are intrinsic scores $\boldsymbol{b}_i$;
- Add a *Dropout* layer to A with dropout probability $p_{user} = 0.4$ and a *Dropout* layer to B with dropout probability $p_{movie} = 0.4$;
- Do the inner product between the dropped-out embeddings;
- Add a *Fully Connected Layer* with $h = 96$ neurons followed by a *ReLU* layer and a *Dropout* layer with dropout probability $p_{fc} = 0.4$;
- Use the *sigmoid* function followed by some scale-shift linear transformations to make the output in the range $[0.5, 5.5]$.

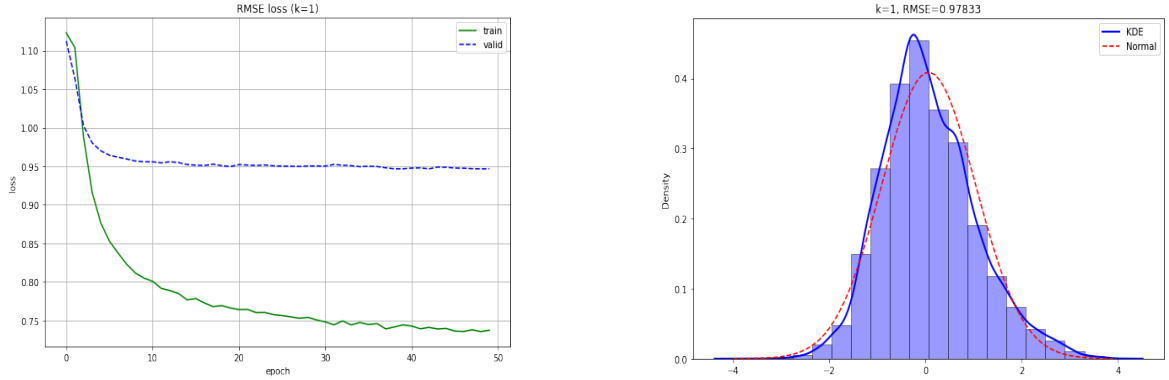
Adding a *Fully Connected* hidden layer introduces non-linearity to the model. To avoid the problem of over-parametrization and over-fitting, *Dropout* layers are used to intrinsicly introduce regularization to the parameters. The objective loss function is the mean-squared-error of predicted ratings. The model is trained using the *Adam* optimization algorithm (Kingma and Ba, 2014) with learning rate 10^{-3} and batch size 128. The summary for the model architecture is given below:

```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
user_input (InputLayer)         [(None, 1)]          0           []

movie_input (InputLayer)        [(None, 1)]          0           []

user_embedding (Embedding)      (None, 1, 50)        47150       ['user_input[0][0]']

movie_embedding (Embedding)     (None, 1, 50)        84100       ['movie_input[0][0]']

FlattenUsers (Flatten)          (None, 50)           0           ['user_embedding[0][0]']

FlattenMovies (Flatten)         (None, 50)           0           ['movie_embedding[0][0]']

user_dropout (Dropout)          (None, 50)           0           ['FlattenUsers[0][0]']

movie_dropout (Dropout)         (None, 50)           0           ['FlattenMovies[0][0]']

Simalarity-Dot-Product (Dot)    (None, 1)            0           ['user_dropout[0][0]',


fc_layer (Dense)                (None, 96)           192         ['Simalarity-Dot-Product[0][0]']

fc_dropout (Dropout)            (None, 96)           0           ['fc_layer[0][0]']

==================================================================================================
```

```
Total params: 131,442
Trainable params: 131,442
Non-trainable params: 0
----------------------------------------------------------------------------------------------------
```

Figure 1 shows the prediction results when trained and validated on '*u1.base*' and tested on '*u1.test*'. Figure 1a shows the RMSE curves for training and validation sets. The training and validation sets are 80%-20% random split of '*u1.base*' set. The loss decreases raplidly and converges after around 20 epochs. Figure 1b shows the distribution of the prediction error, i.e., the difference between predicted ratings and true ratings on the test set '*u1.test*'. The blue curve shows the estimated density using kernel density estimation; while the red dashed curve shows the estimated density under normal assumption (i.e., only mean and standard deviation estimated). The histogram does not show strong evidence of skewness or heavy tail. The RMSE on '*u1.test*' is $RMSE_1 = 0.97406$.

Repeat the experiments on the five pairs of 80%-20% split of the data, (u1.base, u1.test), ..., (u5.base, u5.test), we obtain $RMSE_1 = 0.97406$, $RMSE_2 = 0.96633$, $RMSE_3 = 0.95996$, $RMSE_4 = 0.95582$, $RMSE_5 = 0.96234$, with an average $RMSE_{CV} = 0.9637$.



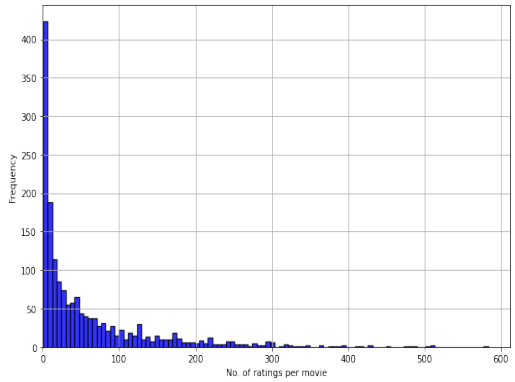
(a) *RMSE curves for train and valid sets.* (b) *Distribution of prediction error.*

Figure 1: *The prediction results when trained on* 'u1.base' *and tested on* 'u1.test'. *(a): RMSE curves for training and validation sets. The loss decreases rapidly with Adam optimizer and after around 20 epochs, the rmse on validation set converges. (b) Histogram: distribution of the prediction errors, i.e., the difference between predicted ratings and true ratings on the test set. Blue curve shows the estimated density using kernel density estimation; red dashed curve shows the estimated density under normal assumption. There's no strong evidence of skewness or heavy tail in prediction error.*

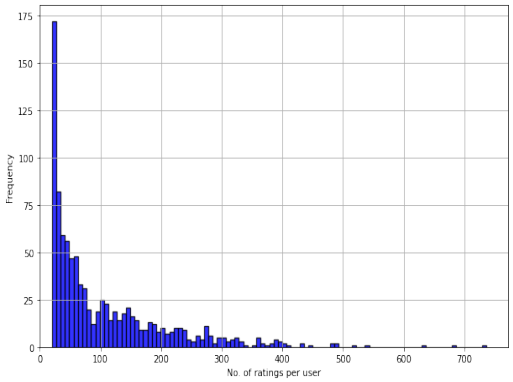
The complete code written in python notebook is available at GitHub: https://github.umn.edu/zhou1354/STAT8056_hw.

Nonignorable Missing

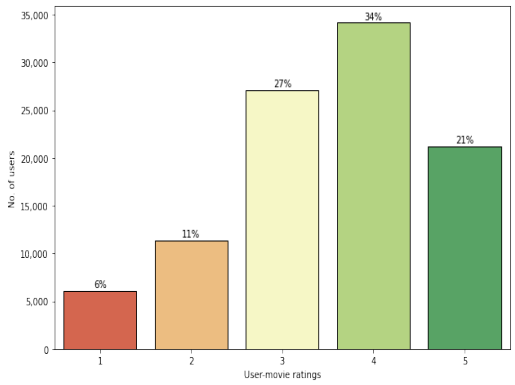
There are 943 users and 1682 movies but only 100,000 (6.3%) ratings in this dataset, that is, 93.7% of the ratings are not observed or missing. Data exploration shows evidence to suggest that missing does not occur at random. Figure 2 illustrates four plots that suggest evidence of nonignorable missing:



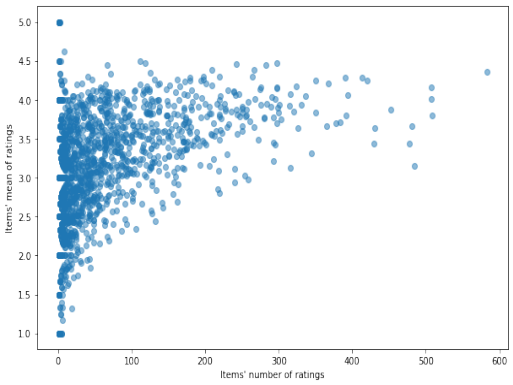
(a) *Histogram of no.ratings per movie.*



(b) *Histogram of no. ratings per user.*



(c) *Distribution of user-movie ratings.*



(d) *Scatterplot of rating mean against rating number.*

Figure 2: *Data exploration: nonignorable missing. (a): Histogram of number of ratings per movie. The distribution is highly skewed with heavy tail and extremely high density at around zero. Most movies are not popular and only a few are popoular. (b): Histogram of number of ratings per user. The distribution is highly skewed with heavy right tail and extremely high density at around 20. Most users rated few movies and only a few rated a lot. (c): Barplot: Distribution of movie-user rating. The above-average score movies are rated much more than below-average score movies. (d) Scatterplot: movie mean of rating v.s. movie number of rating. Popular movies have high ratings.*

(a) Histogram of number of ratings per movie. For this 100k Movielens dataset, the number of movies' ratings has a large range between 0 and 600. If the missing was completely at random, we expect to see a bell-shaped distribution from the histogram. However, the true distribution is highly skewed with extremely high density at around zero. Most of the movies have only a few ratings and not many movies are popular.

(b) Histogram of number of ratings per user. The number of users' ratings also has a large range between 20 and 800. The distribution is highly skewed with heavy right tail and extremely high density at around 20. Most of the users have only a few ratings and not many users rated a huge amount of movies.

(c) Barplot showing distribution of user-movie ratings. If the missing was completely at random, we expect to see a bell-shaped distribution. However, only 17% of the user-movie ratings are at below-average score (i.e., at score 1 or 2) while 55% of the user-movie ratings are at above-average score (4 or 5). If we simply treat above-average score as recommended and below-average

score as not recommended. This means that recommended / good movies are rated much more than not-recommended / bad movies. This is probably because the later users tends to watch the high-rating movies rated by former users.

(d) Scatterplot showing movies' mean of ratings against movies' number of ratings. The mean of movies' ratings has large variation with a pattern. The number of ratings are highly associated with the mean of ratings. Popular movies tend to have high ratings.

Method: Group-specific Recommender System (Bi et al., 2017)

In the group-specific recommender system, the number of user groups N and the number of item groups M are introduced and pre-specified. The model adds additional latent factors to the matrix factorization model and assums that the expectation of rating by user u on movie i is equal to

$$\mathbb{E}[r_{ui}] = (\boldsymbol{p}_u + \boldsymbol{s}_{v_u})^T (\boldsymbol{q}_i + \boldsymbol{t}_{j_i})$$

where $v_u \in \{1, \cdots, N\}$ and $j_i \in \{1, \cdots, M\}$ are group labels for user u and movie i, and $\boldsymbol{s}_{v_u}$ and $\boldsymbol{t}_{j_i}$ are K-dimensional group effects for user u and movie i. With this additional latent group factors, the model could possibly capture the non-ignorable missing and strong dependency among users and movies.

We set the number of groups as $N = 12$ for user group and $M = 10$ for movie group and the results are robust if large values of N and M are chosen as suggested by Bi et al. (2017). The number of latend factors is set as $K = 6$ and the ℓ_2 tuning parameter is set as $\lambda = 10$. Repeat the experiments on the five pairs of 80%-20% split of the data, (u1.base, u1.test), ..., (u5.base, u5.test), we obtain $RMSE_1 = 0.9232$, $RMSE_2 = 0.9121$, $RMSE_3 = 0.9189$, $RMSE_4 = 0.9117$, $RMSE_5 = 0.9138$, with an average $RMSE_{CV} = 0.9159$. Comparing with the average RMSE using the matrix factorization method, the group-specific method has better prediction accuracy.

The implementation of group-specific recommender system was modified from Matlab codes provided at https://sites.google.com/site/xuanbigts/software. The complete Matlab code is available at GitHub: https://github.umn.edu/zhou1354/STAT8056_hw.

References

Bi, X., Qu, A., Wang, J., and Shen, X. (2017). A group-specific recommender system. *Journal of the American Statistical Association*, 112(519):1344–1353.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Mazumder, R., Hastie, T., and Tibshirani, R. (2010). Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11:2287–2322.