11-741/641/441 Machine Learning for Text Mining
Homework 6: Collaborative Ranking

TA in charge: Guoqing Zheng

Due: April 26, 11:59 PM

In Homework 4 you have learnt how to implement a collaborative filtering system for movie ratings. In this assignment, we ask you to complete the same task but with the technique called "Collaborative Ranking".

1 Background

In the collaborative filtering homework we asked you to implement a CF system and evaluate your system performance in Netflix dataset with root mean squared error (RMSE). However, RMSE may not be the the optimal evaluation metric. In practice our system may only generate top-k list of items to a user, thus the user will never see items that the system believes she will not give a high score. RMSE or the equivalent regression-based metric put the same emphasis on high score items and low score ones, enforcing the system to do a lot of extra work. For example, if we know that Netflix will only recommend items with 4 or 5 stars to the user, requiring the system to predict ratings with 1 star accurately is actually not necessary. Another reason we want to reformulate CF as a ranking problem is that our focus will be precisely on the items' position in the ranked list, instead of their actual ratings. For these reasons we now consider NDCG as evaluation metric and use probabilistic matrix factorization and learn-to-rank(LETOR) methods to solve CF problem again. For more detailed background of collaborative ranking, please refer to [1].

2 Problem Formulation

Recall that LETOR in web search essentially involves generating a ranked list of documents given a query. Now we view a user in the recommendation task as a query in the web search setting and items as the documents to rank. Unlike LETOR for web search, we do not have explicit user-item features with which to train the models. Fortunately, this is exactly what we get as a result of training a latent factor model. In this task, we ask you to firstly apply Probabilistic Matrix Factorization [2] to user-item matrix $R \in \mathbb{R}^{m \times n}$($m$ users and n items) and get user factors $\{\boldsymbol{a}_u\}_{u=1,..,m}$ and item factors $\{\boldsymbol{b}_i\}_{i=1,..,n}$. In this latent model, we learn d-dimensional vectors representing "factors" for each user and item, which models our belief that a small number of unobserved factors are sufficient to provide accurate ratings. We learn each latent vector by minimizing the following primary objective function:

$$\min_{\boldsymbol{u},\boldsymbol{v}} \sum_{\{u,i\} \in R} (R_{ui} - \boldsymbol{a}_u^\top \boldsymbol{b}_i)^2 + \lambda (\sum_{i=1}^{n} \|\boldsymbol{b}_i\|^2 + \sum_{u=1}^{m} \|\boldsymbol{a}_u\|^2)$$

We then use feature vector $\mathbf{x}_{ui} = [\boldsymbol{a}_u . * \boldsymbol{b}_i]$ for the pair of user u and item i, here the symbol ".*" denotes point-wise multiplication (hence $\mathbf{x}_{ui}$ is also a vector). You are asked to use these feature vectors and apply RankSVM and LR-based LETOR (We call it LR-LETOR hereafter) algorithm to generate ranked list for missing elements in user-item matrix R.

Note that PMF can be used as baseline CF method as well. We can make prediction for a missing rating $\hat{R}_{u,i}$ by taking the inner product of the factors of user u and item i:

$$\hat{R}_{u,i} = \boldsymbol{a}_u^\top \boldsymbol{b}_i$$

3 Implementation Requirement

3.1 The Dataset

The dataset is the same as the one you used in Homework 4. The output format is also the same. Note that although we ask you to predict score for each user-item pair in test set, this time in evaluation we don't care about the actual values of the ratings but rather the relative ranking order of the items given each user as a query.

3.2 Evaluation

This time we use NDCG as the evaluation metric. We provide you a script to evaluate your system output.

3.3 Comparing methods

We ask you to implement PMF as a baseline and then use user-item feature pairs from PMF to implement RankSVM and LR-LETOR. In your report please also include results from Homework 4 (User-user and item-item similarity) and evaluate them using NDCG. Please compare and analyse the performance of these methods in your report.

Following are the detailed implementation requirements about RankSVM and LR-LETOR:

1. **Training set construction**
   Given the observed rating matrix R, construct the training set T as

   $$T = \left\{ \left(\mathbf{v}_{ij}^{(u)}, y_{ij}^{(u)} \right) \middle| \mathbf{v}_{ij}^{(u)} = \mathbf{x}_{ui} - \mathbf{x}_{uj},\ y_{ij}^{(u)} = sgn(R_{ui} - R_{uj}),\ \text{for } (u,i,j) \in \Omega \right\} \tag{1}$$

   where

   $$\Omega = \{(u,i,j) |\ |R_{ui} - R_{uj}| = 4,\ \text{for } (u,i,j) \in R\} \tag{2}$$

   and $sgn()$ is the sign function.

   Essentially, you are only asked to consider pairs of items given a user where the difference of the item scores is 4, which is trying to discriminate highest-rated items from lowest-rated items as much as possible.

   After constructing the training set as described above, the pair-wise learning to rank problem turns to a binary classification problem over T.

2. **RankSVM**
   In this part, we use LIBLINEAR (Version 2.1) again for soving the binary classification problem on T. For the training, again you should use **L2-regularized L2-loss support vector classification**. Report your experiments as required by the report template.

3. **LR-LETOR**
   In this part, implement a binary regularized LR for LETOR. (You can reuse your code of RMLR from HW5). Report your experiments as required by the report template.

4 Grading and Submission

What to Turn In
You must submit a .zip file with the name `HW6-YourAndrewID.zip` to Blackboard that contains a report, your test set predictions, and your source code by April 26, 11:59PM.

1. **Report (80 pts)**: You must describe your work and your analysis of the experimental results in a written report. A report template is provided. Please address all the questions in this template report within the specified section. Do not change the section headings. Do NOT list your source code in the report. Please make it easy for the TA to see how you have addressed each of the requirements described for each section. Name the report `HW6-YourAndrewID.pdf`.

2. **Test set predictions (10 pts) + Bonus (5 pts)**: You must provide the rating file for the test set using your best algorithm (out of PMF, RankSVM and LR-based LETOR) - use your best judgment as to which system is "best" (again NDCG on the development set is typically a good indicator). Its performance will be graded in terms of NDCG@20. The top 5 submissions in evaluation will receive the bonus 5 pts. Name the prediction file `predictions.txt`.

3. **Source code (10 pts)**: Your submission must include all of your source codes, and any files necessary to make and run your source code. Your source code must run within our testing service, so please check that it does before you make your final submission. The TA will look at your source code, so make sure that it is legible, has reasonable documentation, and can be understood by others. Put all your source codes under the `src/` directory.
   Note: Although code documentation is worth only 10 pts, failing to submit source code will result in a zero grade.

Restrictions

- Your system must do this task entirely automatically. No manual intervention is allowed. The TA **will not** modify your source code in order to change the parameters or run a different experiment.
- You must write all of the software yourself. (You can re-use your own code from previous homeworks; however, you are NOT allowed to use other's code in the case that you discussed previous HWs with other students.)

References

[1] Suhrid Balakrishnan and Sumit Chopra. Collaborative ranking. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 143–152. ACM, 2012.

[2] Andriy Mnih and Ruslan Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2007.