Comparing the Staples in Latent Factor Models for Recommender Systems

Cheng Chen
University of Victoria
Victoria, Canada
cchenv@uvic.ca

Lan Zheng
University of Victoria
Victoria, Canada
lanzheng@uvic.ca

Alex Thomo
University of Victoria
Victoria, Canada
thomo@cs.uvic.ca

Kui Wu
University of Victoria
Victoria, Canada
wkui@ieee.org

Venkatesh Srinivasan
University of Victoria
Victoria, Canada
venkat@cs.uvic.ca

ABSTRACT

Since the Netflix Prize competition, latent factor models (LFMs) have become the comparison "staples" for many of the recent recommender methods. The performance improvement of LFMs over baseline approaches, however, hovers at only low percentage numbers. Therefore, it is time for a better understanding of their real power beyond the overall RMSE (root-mean-square error), which as it happens, lies at a very compressed range, without providing too much chance for deeper insight. This paper provides a detailed experimental study regarding the performance of classical staple LFMs on a classical dataset, Movielens 1M[1], that sheds light on a much more pronounced excellence of LFMs for particular categories of users and items, for RMSE and other measures. In particular, LFMs exhibit surprising and excellent advantages when handling several difficult user and item categories. By comparing the distributions of the test and predicted ratings, we show that the performance of LFMs is influenced by the rating distribution. We then propose a method to estimate the performance of LFMs for a given rating dataset. Also, we provide a very simple, open-source, library that implements staple LFMs achieving a similar performance as some very recent (2013) developments in LFMs, and at the same time being more transparent than some other libraries in wide use.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Keywords

Recommender systems, latent factor models, evaluation

1. INTRODUCTION

Since the Netflix one-million-dollar prize competition, latent factor models (LFMs) have gained an immense popularity for implementing recommender systems. One can find hundreds of citations to the central articles of Bell, Koren, and Volinsky (BKV) describing their LFM methodology for movie recommendation [5, 6, 7]. The basic BKV methods have a great appeal of simplicity that has also contributed to their popularity. The metric of choice for comparing recommenders is the root mean squared error (RMSE), and improving on it by devising new adaptations and additions has become a "sport" in the quest for better recommender systems. One can notice, however, that the main improvements come from the basic LFM methods, with the more elaborate methods improving by only a very slim margin, if at all[2].

Notwithstanding their popularity, the fact is that LFMs improve RMSE against naive baselines by only a low margin. For instance, the LFMs described in (cf. [7]) show an improvement of about 0.05 against baseline approaches on the classical MovieLens 1M dataset (with ratings in a 1-5 scale). This is somewhat discouraging and naturally raises the question if the popularity of LFMs is well-founded. Here we provide evidence towards a positive answer to this question and exhibit the excellence of LFMs over baseline approaches by providing a detailed study on the Movielens 1M dataset. Our study shows the advantage of LFMs over baseline approaches, especially for difficult to handle categories of users and items.

Inspired by [11], we define Coldstart, Heavyrater, Opinionated, and Blacksheep users, and Controversial, and Niche items. Except for Heavyrater users, the other categories are typically difficult to handle well by recommender systems. We obtain results not only for RMSE, but also for precision, recall, F measure, and accuracy for each of the aforementioned categories. Our results are revealing. For instance, for Blacksheep users, which are those who go against the mainstream, we surprisingly observed a RMSE improvement of about 12%, which is more than 56% better than the improvement observed for the general case. Impressive improvements can also be seen for other categories, such as Opinionated users and Controversial items. Some more attractive examples are the improvements on recall we observed for Controversial items and Blacksheep users that are in the order of 30% over the baseline, or more than 6 times better than improvements observed for the general (overall) case.

[1]http://www.grouplens.org/node/73, accessed on September, 2013

[2]For example, using a very simple library we implemented ourselves, we were able to achieve a similar RMSE as [9].

There are several ways to combine latent factors with the overall mean rating, and the user and item biases, resulting in various LFM approaches. The biases capture "how lenient" and "how liked" the users and items are, respectively, and they have been claimed to have a high impact on the quality of the predicted ratings. We show, for each metric and for each category, the approach that achieves the best performance. For instance, for F measure, it turns out that ignoring user and items biases works better than including them, in contrast to our observations for RMSE. Finally, we show that the performance of LFMs is influenced by the underlying rating distribution.

More specifically the contributions of this work are as follows.

1. We describe four different combinations of latent factors with the mean, user and item biases, and provide a simple library implementation that is more transparent than other available software.

2. We show that the performance of the LFMs over the MovieLens 1M (without considering user and item categories) exhibits a similar behavior as that of the baseline. Therefore, for the general case, the benefit from latent factors is not particularly strong.

3. We define user and item categories, some of which are difficult to handle by recommender systems (RMSE is high for them), and show that it is for some of these categories that the latent factors really excel and produce impressive results.

4. We compare the groundtruth rating (i.e., real-world data) distribution as well as the predicted rating distribution for each category and show that LFMs have better performance on a Gaussian-like distributed rating dataset. We then propose a method to estimate the performance of LFMs for a given rating dataset.

2. PRELIMINARIES

We use the following notation: (1) r_{ui}: rating of user u for item i, (2) $\hat{r}_{ui}$: predicted rating of user u for item i, (3) m: the mean rating, (4) b_u: bias for user u, (5) b_i: bias for item i, (6) $\mathbf{p}_u$: vector of latent factor values for user u, and (7) $\mathbf{q}_i$: vector of latent factor values for item i.

The user and item biases b_u, b_i are simple real numbers that capture how lenient user u is and how liked item i is. $\mathbf{p}_u$ and $\mathbf{q}_i$ are vectors of d real values. These values capture the importance of d latent factors in characterizing user u and item i in the space of these latent factors. All b_u, b_i, $\mathbf{p}_u$, and $\mathbf{q}_i$ are typically learned by a stochastic gradient descent (SGD) procedure that strives to minimize the squared error $e_{ui}^2 = (r_{ui} - \hat{r}_{ui})^2$ for each existing r_{ui} entry. We consider the following combinations for computing the predicted rating:

$$\hat{r}_{ui} = m + b_u + b_i \quad (1)$$

$$\hat{r}_{ui} = \mathbf{p}_u \cdot \mathbf{q}_i \quad (2)$$

$$\hat{r}_{ui} = m + \mathbf{p}_u \cdot \mathbf{q}_i \quad (3)$$

$$\hat{r}_{ui} = m + b_u + b_i + \mathbf{p}_u \cdot \mathbf{q}_i \quad (4)$$

We call (1) the *baseline* approach, (2) the *pure factor* approach, and (3) and (4) the *mixed* approaches, with (4) being advocated by Koren and Bell [7]. The SGD rules for updating b_u, b_i, $\mathbf{p}_u$, and $\mathbf{q}_i$ at each known r_{ui} are

$$b_u \leftarrow b_u + \gamma(e_{ui} - \lambda_1 b_u) \quad (5)$$

$$b_i \leftarrow b_i + \gamma(e_{ui} - \lambda_2 b_i) \quad (6)$$

$$\mathbf{p}_u \leftarrow \mathbf{p}_u + \gamma(e_{ui}\mathbf{q}_i - \lambda_3 \mathbf{p}_u) \quad (7)$$

$$\mathbf{q}_i \leftarrow \mathbf{q}_i + \gamma(e_{ui}\mathbf{p}_u - \lambda_4 \mathbf{q}_i) \quad (8)$$

where r_{ui} is involved in computing e_{ui}, γ is the learning rate, and λ_1, λ_2, λ_3, λ_4 are regularization constants. We provide a simple, open-source, implementation of the above LFM methods, where we strive for conciseness and transparency [1] [3].

In addition, we also investigate Bayesian Probabilistic Matrix Factorization(BPMF) [12]. As a fully Bayesian treatment of probabilistic matrix factorization, this model places hyperpriors over the hyperparameters and uses Markov Chain Monte Carlo to perform approximate inference for factors.

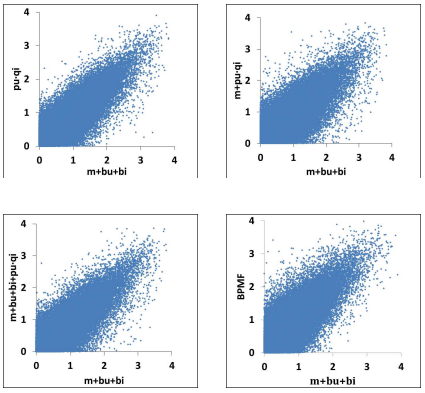


Figure 1: Absolute error correlation between baseline and LFMs

3. OVERALL PERFORMANCE

In this section we show that the overall behavior of recommenders (2), (3), (4) and BPMF (5) is very similar to the baseline recommender (1).

Figure 1 shows the scatter plots of the absolute error of the baseline against the LFMs over the whole test set. Absolute error is equal to the absolute difference between the true rating and the predicted one. We can observe that the points on the plots clearly demonstrate a positive correlation between the baseline and four LFMs, suggesting that if a test case is difficult for the baseline (the absolute error is high), it is not necessarily easier for the LFMs.

Is this the whole story about these models? The answer is no. In the following, we further investigate the performance for different user and item categories.

4. USER AND ITEM CATEGORIES

The performance picture changes in quite interesting ways once we consider particular categories of users and items. Similarly as [11] suggests, we define user and item categories for the MovieLens 1M dataset

[3]We are not claiming that popular libraries, such as MyMediaLite ([4]) and GraphLab ([10]), contain problems. We are only implying that building a simpler library is sometimes easier than tuning a more general purpose one.

Heavyrater users(HR) who provided more than 270 ratings.

Opinionated users(OP) who provided more than 135 ratings and whose standard deviation of ratings is larger than 1.2.

Blacksheep users(BS) who provided more than 135 ratings and for which the average distance of their ratings from the mean rating of the corresponding items is greater than 1.0.

Coldstart users(CS) who provided no more than 135 ratings.

Controversial items(CI) which received ratings whose standard deviation is larger than 1.1.

Niche items(NI) which received less than 135 ratings.

Our reasoning for the above numbers is as follows. MovieLens 1M is a very dense dataset. 270 is the median of the number of each user's ratings in the training set. We chose half of it as the threshold for **Coldstart users** as well as **Niche items**. This number covers approximately half of the total users and items, respectively. The choices of standard deviation were inspired by [11]. Similar groups have also been used in [2].

5. EXPERIMENTS

We conducted the experiments on the MovieLens 1M dataset in a manner of 10-fold cross validation. In our experiments we used factor dimension $d = 25$, and tuned the learning rate and regularization constants by grid search and cross-validation. Specifically, $\gamma = 0.005$, $\lambda_1 = \lambda_2 = 0.02$, $\lambda_3 = \lambda_4 = 0.03$. Other tested combinations of these hyperparameters performed worse in the cross-validation. For BPMF, the number of samples from Markov Chain is around 20. The number is not fixed because we evaluate RMSE for each sample (user and item factors) on the training set and we stop the sampling procedure if the RMSE starts to increase.

In addition to RMSE, we also evaluate the following four performance metrics with 3.5 as the threshold to classify positive and negative samples:

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

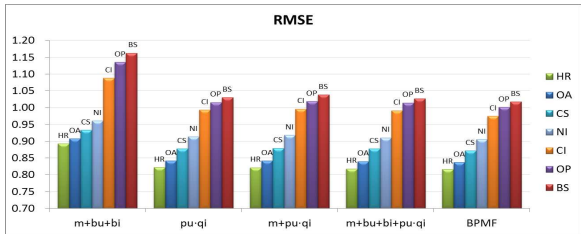
$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$F\ measure = 2 * \frac{Precision * Recall}{Precision + Recall}$$

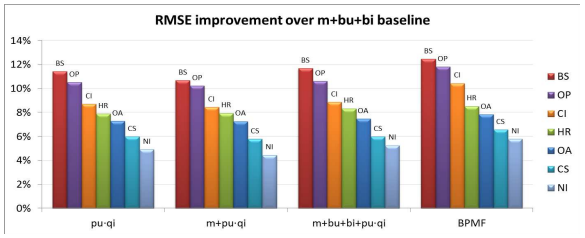
$$Accuracy = \frac{TrueNegative + TruePositive}{TotalNumberofUsers}$$

The four metrics and RMSE are computed as the average of the 10-fold test.

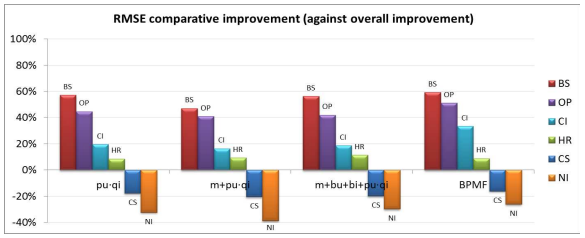
In the following, we describe Figures 2-6 and conclusions drawn from them. In these figures, **OA** stands for overall performance.



(a)

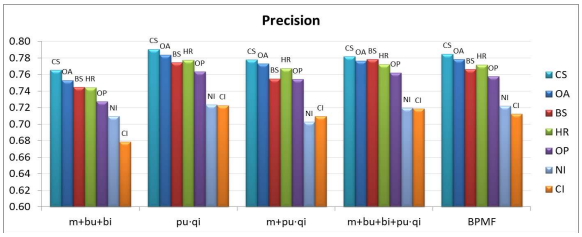


(b)

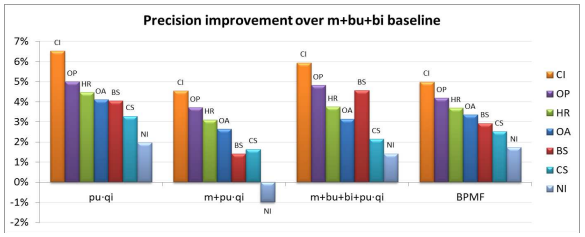


(c)

Figure 2: RMSE



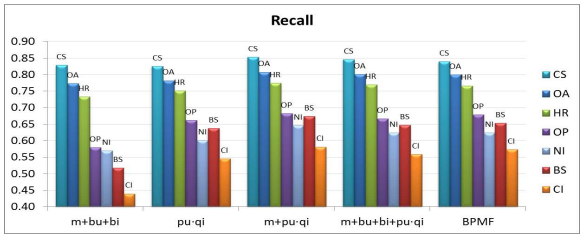
(a)



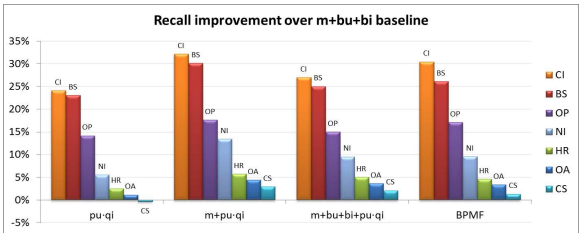
(b)

Figure 3: Precision

1. The RMSE values for each category are shown in Figure 2 (a). The more on the left a bar is, the smaller (better) the RMSE is. We can see that the most difficult categories in terms of RMSE are Blacksheep and Opinionated users, and Controversial items. We see improvement for $\mathbf{p}_u \cdot \mathbf{q}_i$ (2), $m + \mathbf{p}_u \cdot \mathbf{q}_i$ (3), $m + b_u +$

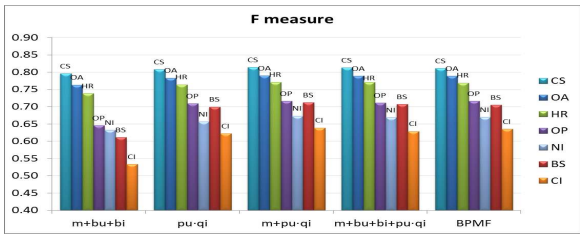


(a)

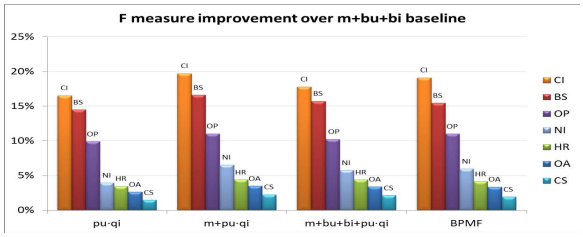


(b)

Figure 4: Recall



(a)



(b)

Figure 5: F measure

$b_i + \mathbf{p}_u \cdot \mathbf{q}_i$ (4) and BPMF (5) compared to the baseline $m + b_u + b_i$ (1). Figure 2 (b) gives the amount of improvement over the baseline. We see that the improvement for Blacksheep users is more than 11% and for the Opinionated users is more than 10%. This is quite impressive if we recall that the Neflix competition was for improving the RMSE by 10%. We can also observe that the difference in improvement between (2), (3), (4) and (5) is not too pronounced, with (5) doing only slightly better than the other three. Figure 2 (c) shows how much greater is the improvement for (2), (3), (4) and (5), for each category, compared to the overall improvement. For example, for Blacksheep users, the improvements for (2), (4) and (5) are close to 60% greater than the overall improvement. Specifically, the improvement for Controversial items for (5) is now more than 10%, which is a significant change over the other methods.

2. The Precision values for each category are shown in Figure 3 (a). Here we observe some surprising facts, such as the precision for Coldstart users (a difficult category for RMSE) being the highest for all approaches, and especially for (2), which is more than 79%. Also, Blacksheep, Heavyrater, and Opinionated users have a higher precision in all four models. The improvements for (2), (3), (4) and (5) are good (see Figure 3 (b)), but not as great as those for RMSE. The precision values for (2), (3), (4) and (5) are better than those for the baseline except for (3) for Niche items.

3. The recall values for each category are shown in Figure 4 (a). The recall for Coldstart users exhibits the highest value for all approaches, and especially for (2), which is (slightly) more than 85%. The improvements shown in the same figure (b), are impressive for (3) for Controversial items and Blacksheep users with bars exceeding 30%!

4. The F measure for each category are shown in Figure 5 (a). Coldstart users still score the highest, with values of more than 0.8, for all approaches. Again, the improvements shown in the same figure (b), are impressive for some categories, such as that for (3), for Controversial items, which is close to 20%. Notably, all (2), (3), (4) and (5) improve over the baseline (1) for all the categories.

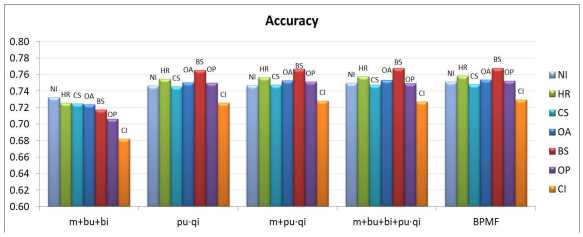
5. Accuracy values are given in Figure 6 (a). All (2), (3), (4) and (5) have similar accuracy values across all categories, with Blacksheep users faring better than the other categories. Accuracy improvements are shown in the same figure (b). The improvements we observe are similar to those for precision.

6. RATING DISTRIBUTIONS OF DIFFERENT CATEGORIES

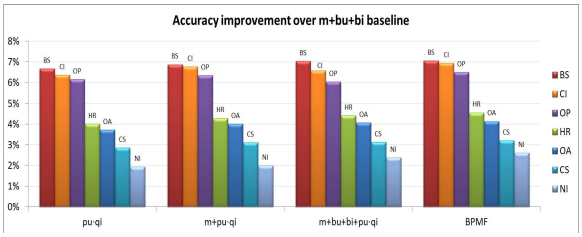
Based on figures in previous sections, we observe that the performance of LFMs are different for different user/item categories. To the best of our knowledge, there hasn't been such fine-grained analysis of LFMs in the related research. We hypothesize that the performance of LFMs is influenced by the distribution of ratings in different categories. In this section, we focus on the relationship between RMSE and the distribution of ratings in different categories. We will analyze the baseline method and three LFMs, specifically, $m+b_u+b_i$ (1), $\mathbf{p}_u \cdot \mathbf{q}_i$ (2), $m+\mathbf{p}_u \cdot \mathbf{q}_i$ (3) and $m+b_u+b_i+\mathbf{p}_u \cdot \mathbf{q}_i$ (4).

We select one train/test split of the 10-fold test to demonstrate rating distributions of different categories. We only present figures for test dataset because training dataset has similar distribution.

We firstly draw histograms of the groundtruth test ratings for each category, as shown in Figure 7.

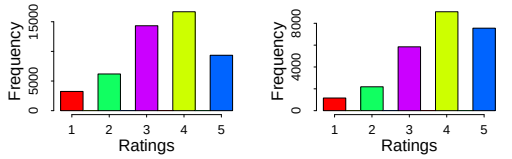


(a)

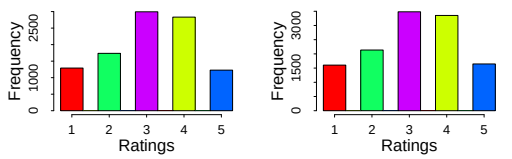


(b)

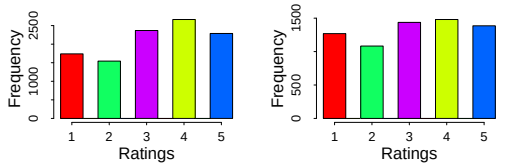
Figure 6: Accuracy



(a) Heavyrater Users (b) Coldstart Users



(c) Niche Items (d) Controversial Items



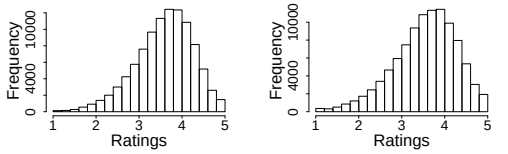
(e) Opinionated Users (f) Blacksheep Users

Figure 7: Histogram of groundtruth ratings

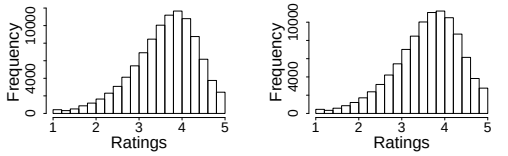
In Figure 7, we see that ratings in Opinionated users and Blacksheep users are more uniformly distributed than the other four categories. Surprisingly, we have already observed that LFMs have the worst RMSE for these two categories in Figure 2. This implies that the difference in rating distributions of different categories may result in different performance for LFMs.

In order to verify this hypothesis from the model side, we plot histograms of the overall ratings produced by $m+b_u+b_i$ and three LFMs in Figure 8. We observe that the four models exhibit a very similar shape of the rating distribution. All of them have a truncated Gaussian-like distribution in the predictions, where the majority of ratings is located around the mean value. It indicates that if the ratings to be predicted are far from Gaussian distribution, then LFMs may fail to give good predictions.

We then plot for each category the predicted ratings produced by $m+b_u+b_i+\mathbf{p}_u\cdot\mathbf{q}_i$ (4) in Figure 9. Since Figure 2 suggests LFMs have the same relative performance for each category, $HR < OA < CS < NI < CI < OP < BS$ in terms of RMSE, we choose this model as an objective example.



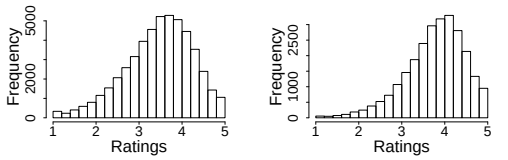
(a) $m+b_u+b_i$ (b) $\mathbf{p}_u\cdot\mathbf{q}_i$



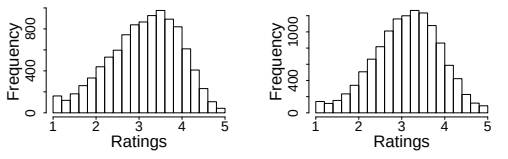
(c) $m+\mathbf{p}_u\cdot\mathbf{q}_i$ (d) $m+b_u+b_i+\mathbf{p}_u\cdot\mathbf{q}_i$

Figure 8: Histogram of predicted ratings

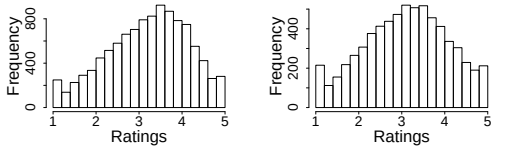
Figure 9 shows that the predicted rating distribution for each category is similar. In other words, LFMs always output Gaussian-like rating distribution regardless of the type of category.



(a) Heavyrater Users (b) Coldstart Users



(c) Niche Items (d) Controversial Items



(e) Opinionated Users (f) Blacksheep Users

Figure 9: Histogram plots of ratings for different categories produced by $m+b_u+b_i+\mathbf{p}_u\cdot\mathbf{q}_i$

Inspired by this observation, we propose a method to estimate the performance of LFMs given a test rating dataset.

This method compares a given discrete rating dataset to a synthetic dataset generated from a Gaussian distribution which is discrete and truncated. Intuitively, the smaller the difference, the better the performance we should expect from LFMs. The sampling algorithm to create the synthetic dataset is described below (Algorithm 1).

Algorithm 1: SYNTHETIC Gaussian distributed data generation based on a given dataset

Input: A rating dataset for a certain category
Output: Synthetic ratings

```
Compute features of the input, length, mean, sd;
Initialize Gaussian parameters by the statistical
  features, μ = mean, σ = sd;
Initialize an empty list, R_synthetic;
for i in 1 : length do
    Sample a random number rn from a Gaussian
      distribution rn ~ N(μ, σ);
    Round rn to an integer number;
    while rn < 1 || rn > 5 do
        Sample a rn, rn ~ N(μ, σ);
        Round rn to an integer number;
    Add rn into R_synthetic;
return R_synthetic;
```

Once we have the synthetic dataset, we use Kullback-Leibler (KL) divergence [8, 3] to evaluate the difference between the two discrete probability distributions.

Figure 10 shows the KL divergence from each user/item category to its corresponding discrete and truncated Gaussian distribution.

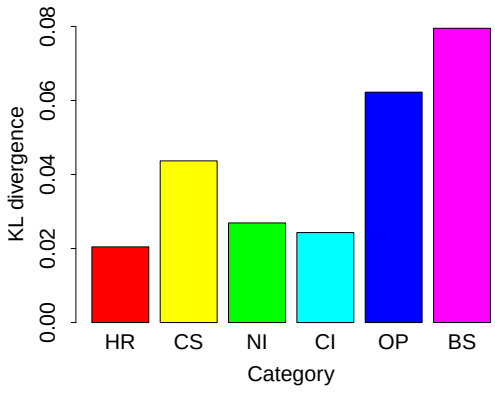


Figure 10: KL divergence for different categories

We can clearly see the groundtruth ratings of Blacksheep users and Opinionated users are less likely to be Gaussian-like distributed than those of the other four categories. In addition, the rating of Heavyrater users has the smallest KL divergence among all categories. This fact verifies what we have observed in Figure 2, where LFMs have the best performance for Heavyrater users and the worst RMSE for Blacksheep users and Opinionated users. However, for Cold-start users, Controversial items and Niche items, due to insufficient samples, the estimation of KL divergence is not accurate enough to confirm the hypothesis. We will explore possible measures other than KL divergence in future work.

7. CONCLUSIONS

We conducted detailed experiments to verify the capability of LFMs and found that whereas LFMs improve RMSE against the baseline method by a small percent over the whole dataset, LFMs show very promising advantages when dealing with certain difficult categories of users and items. We also present the relationship between the groundtruth rating distribution and the predicted rating distribution for each category. We conclude that LFMs perform better on Gaussian-like distributed rating dataset. Future work involves continuing to investigate rating distribution as well as other possibilities so that we can have a better understanding of the performance of LFMs. We will also conduct similar analysis using different datasets to further validate the hypothesis.

8. REFERENCES

[1] Java library for latent factor models, November 2013.

[2] M. Chowdhury, A. Thomo, and W. W. Wadge. Trust-based infinitesimals for enhanced collaborative filtering. In S. Chawla, K. Karlapalem, and V. Pudi, editors, *COMAD*. Computer Society of India, 2009.

[3] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.

[4] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Mymedialite: a free recommender system library. In B. Mobasher, R. D. Burke, D. Jannach, and G. Adomavicius, editors, *RecSys*, pages 305–308. ACM, 2011.

[5] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In Y. Li, B. Liu, and S. Sarawagi, editors, *KDD*, pages 426–434. ACM, 2008.

[6] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[7] Y. Koren and R. M. Bell. Advances in collaborative filtering. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 145–186. Springer, 2011.

[8] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 1951.

[9] J. Lee, S. Kim, G. Lebanon, and Y. Singer. Matrix approximation under local low-rank assumption. In *ICML*, 2013.

[10] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. Distributed graphlab: A framework for machine learning in the cloud. *PVLDB*, 5(8):716–727, 2012.

[11] P. Massa and P. Avesani. Trust-aware recommender systems. In J. A. Konstan, J. Riedl, and B. Smyth, editors, *RecSys*, pages 17–24. ACM, 2007.

[12] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In W. W. Cohen, A. McCallum, and S. T. Roweis, editors, *ICML*, volume 307 of *ACM International Conference Proceeding Series*, pages 880–887. ACM, 2008.