Learning from Sets

***Abstract*—Existing item recommendation methods rely on the availability of the users preferences on the items consumed in the past. Additionally, the performance of these recommendation methods is proportional to the amount of information available to the recommender system, i.e., the more availability of users preferences on the items, the better the quality of the recommendations for the user. However, it is not easy to elicit the users preferences on items to improve the recommendations for the user, particularly when we have a large number of items. In this paper, we propose a method in which we elicit the users preference on a set of items rather than on the individual items that constitute the set. We use this preference on the set to estimate the users preference on the items that constitute the set. This method, compared to the process in which the user has to provide sequentially his/her preference for the individual items, can quickly gather the users preferences on a few sets that span a large number of items. In the experiments on real-world data, we show that the quality of recommendations generated from these sets is comparable to the recommendations generated using the individual preference on the items.**

I. Introduction

II. Notations and Definitions

III. Related Work

IV. The Movielens set ratings dataset

A. Data collection

Movielens[1] is a recommender system that utilizes collaborative filtering algorithms to recommend movies to their users based on their preferences. We collected ratings from the Movielens users on certain sets of movies. These sets were generated by selecting five movies from the movies that a user has already rated. There was no overlap among the sets presented to the user in a session. The user can rate at most 50 such sets in a session. An example of the questionnaire eliciting a user's rating on a set of movies is shown in the Figure 1. The rating widget in the interface could be rated from 0 to 5 with a precision of 0.5. For the purpose of data collection, we selected users who were active since January 2015. The data was gathered from the user's responses to the questionnaire from Feb 2016 to April 2016.

B. Data processing

Before analyzing the dataset, we removed users who have rated sets within a time interval of less than one second to avoid users who might be guessing the ratings at random. After this data pre-processing step, we were left with ratings from 854 users over 29,516 sets containing 12,549 items. The Figure 2 shows the distribution of the number of sets rated by the user, and we can see that roughly half of the users, i.e., 428 users have rated more than 90% of the sets, i.e., at least 45 sets in a session.

[1]www.movielens.org

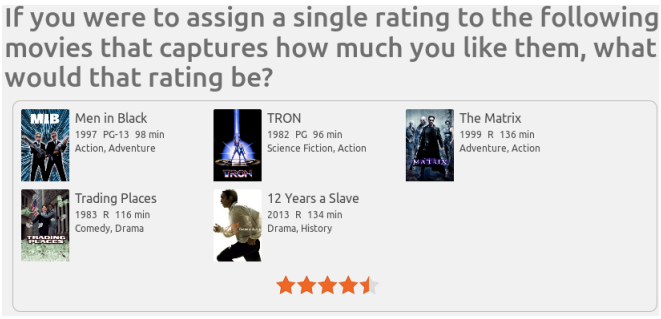

Fig. 1: The interface used to elicit users' ratings on a set of movies.

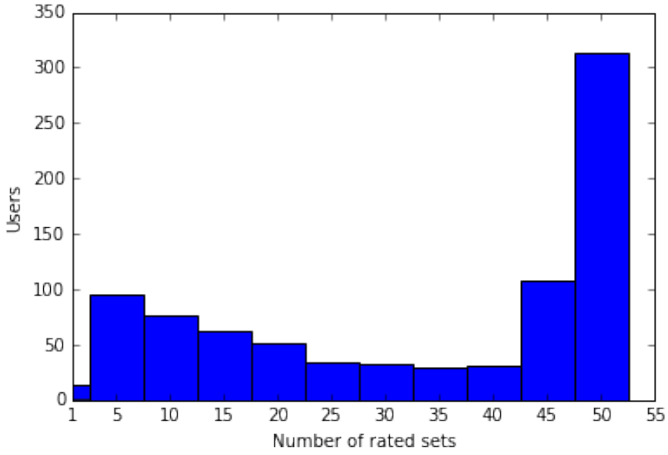

Fig. 2: The distribution of number of sets rated by the users.

We investigated whether ratings are distributed uniformly or if some ratings tend to appear more than the others. Figure 3 depicts the distribution of the collected ratings over all the sets. The majority of the ratings lies between 3.0 and 4.0. Also, ratings less than 2.0 were relatively few when compared with the rest indicating that the users tend to rate sets favorably, or most of the sets contain favorable items.

C. Analysis of set ratings

Since we know the actual rating of the users over the movies in the sets, these ratings can help us understand the users' behavior when assigning a single rating to a set of movies. To this end, we computed the deviation of the rating assigned by a user to the set from the average of the user's ratings of the items constituting the set. To calculate this deviation we computed the Root Mean Square Error (RMSE) between the ratings assigned to the sets and the average of the user-item ratings in the sets. As can be seen in the Table I, this

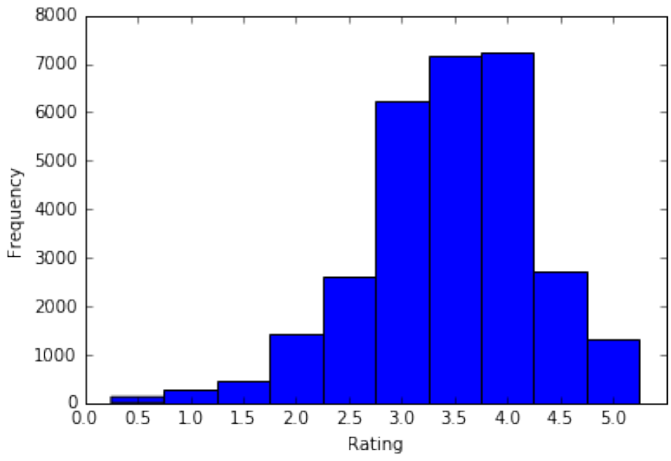

Fig. 3: Ratings distribution.

TABLE I: RMSEs across different type of sets.

Type	% of Sets	Number of Sets	RMSE
Over-rated	17.68	5,220	0.962
Under-rated	22.26	6,571	0.810
Neither	60.05	17,725	0.241
All	100	29,516	0.597

RMSE over all the sets is 0.597. Subsequently, we looked at the fraction of sets that are rated below the average of the ratings of the items in the set. We refer to these sets as the under-rated sets. Similarly, the over-rated sets are those sets that are rated above the average of the ratings of the items in the set.We used a margin of 0.5 while identifying these sets, i.e., sets rated within the margin of 0.5 of their average rating were considered neither as over-rated or under-rated sets.

The Table I shows that the majority of the sets were neither over-rated or under-rated and had a significantly lower RMSE of 0.241 when compared against under-rated or over-rated sets. Further, we selected users who have rated at least 50 sets and computed the fraction of under-rated and over-rated sets. As can be seen from the Figure 4, some users tend to over-rate or under-rate sets and by the Kolmogorov-Smirnov 2 sample test this behavior was found to be statistically different (P-value $<$ 1e-16) from that of random.

Further, we examined the effect of diversity of the items' rating in the set, i.e., whether the presence of movies with varying degree of ratings has any effect on the rating assigned by the user to the set. For each set, we computed the difference between the item's rating and the average of the ratings in the set. We defined set-deviation as the square root of the mean of this difference of the items' ratings in the set. The sets having deviation greater than 1.0 were characterized as "Diverse" sets and rest as "Non-Diverse" sets. Following this characterization in the dataset, nearly 20% of the sets are diverse sets having RMSE of 0.725 while remaining sets have lower RMSE of 0.559. This indicates that user seems to under-rate or over-rate diverse sets more in comparison to non-diverse sets.

We also investigated whether the presence of movies that are rated recently or that are rated in the past has an effect on the rating assigned to the set. To this end, for each set we computed the difference between the timestamp of the earliest rating and 2016, i.e. when the users were asked to rate the sets. Interestingly, the under-rated sets contained ratings that were provided on average 1,800 days,i.e., five years before the survey while remaining sets contained ratings that were provided approximately 1,450 days, i.e., four years before the survey. This difference among the sets was found to be statistically significant by T-test, i.e. P-value $<$ 1e-16. This suggests that the user's preference for a movie in a set appears to decay temporally leading to under-rating of the set containing that movie.

V. Methods

A. Modeling rating on the sets

To estimate the preferences on individual items from the preference of the user on the sets, we need to understand how the user rate a given set of items. A user may rate a set of items by considering each item in the set. When a user consider all the items in the set before assigning a rating to a set then we can safely assume that the user most likely gives the set an average score of his preferences for all the items that constitute the set. This assumption can be validated by the data analysis in the previous section where we observed that most of the ratings on the sets are close to the average of the ratings of the items in the set. Under this assumption the rating of the user u on a set S is given by:

$$r_{us} = \frac{1}{|S|} \sum_{i \in \mathcal{S}} r_{ui}, \tag{1}$$

where $\mathcal{S}$ denotes the set containing the items and r_{ui} is the rating of the user u on the item i.

Since we do not know the original ratings of the items in the sets, i.e., r_{ui} we can estimate the rating on the set, i.e., $\tilde{r}_{us}$ as the average of the estimated ratings of the items in the set.

$$\tilde{r}_{us} = \frac{1}{|S|} \sum_{i \in \mathcal{S}} \tilde{r}_{ui}, \tag{2}$$

where $\tilde{r}_{ui}$ is the estimated rating of the user u on the item i.

B. Modeling user-item ratings

Assuming that the original user-item rating matrix R is low-rank, then following the classic matrix factorization approach [1] the estimated rating of the user u for the item i is given by,

$$\tilde{r}_{ui} = b_u + b_i + p_u^T q_i, \tag{3}$$

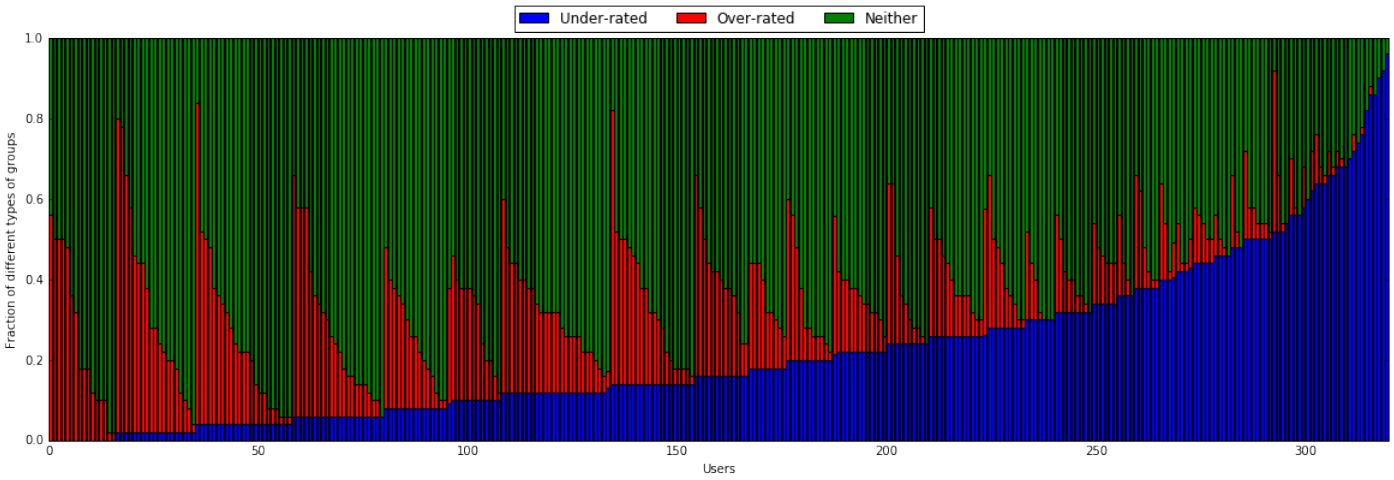

Fig. 4: Fraction of different types of sets across users.

where b_u is the user bias, b_i is the item bias, $p_u \in \mathcal{R}^k$ denotes the latent factor of the user u, $q_i \in \mathcal{R}^K$ denotes the latent factor of the item i and k is the rank of the matrix R.

C. Set rating using matrix-factorization (LFS)

We can rewrite the estimated score of the user u for the set $\mathcal{S}$ in equation 2 using equation 3 as follow:

$$\tilde{r}_{us} = \frac{1}{|S|} \sum_{i \in \mathcal{S}} b_u + b_i + p_u^T q_i, \tag{4}$$

D. Modeling session bias (LFSWSessBias)

Further, the user's ratings on the set could be affected by psychological phenomena or his mood during the session, e.g., a user may rate a set in the context of the sets they have seen before. Also, as seen in our investigations on the data, some users tend to overrate or underrate the sets when compared with the average of the ratings of the items in the set. Such effects can be captured by adding a user specific session bias:

$$\tilde{r}_{us} = b_{us} + \frac{1}{|S|} \sum_{i \in \mathcal{S}} b_u + b_i + p_u^T q_i, \tag{5}$$

where b_{us} denotes the session bias of the user u.

E. Full bias model (LFSWGBias)

Moreover, We can also add the mean of ratings on the sets (a constant) to represent the portion of the rating on the set which is independent of the users' and the items' personalization:

$$\tilde{r}_{us} = \mu + b_{us} + \frac{1}{|S|} \sum_{i \in \mathcal{S}} b_u + b_i + p_u^T q_i, \tag{6}$$

where μ is the mean of ratings on the sets.

F. Model learning

The model parameters, i.e., $\theta = [p_u, q_u, b_u, b_i, b_{us}]$, are estimated by minimizing a loss function. For accurate predictions of ratings for the sets and the items, it is appropriate to minimize a square error loss function to estimate the model. In the Top-n recommendations, the precise rating prediction is irrelevant as we only care about correctly ranking the items for the user. Therefore, a ranking loss function, e.g., Bayesian Personalized Ranking (BPR) [2] is more appropriate to estimate the model. We propose to use both the square error loss function and the BPR loss function to estimate the model parameters.

The loss function to estimate the model by minimizing square error loss function is given by

$$\mathcal{L}_{rmse}(\Theta) \equiv \sum_{u \in U} \sum_{s \in \mathcal{R}_{us}} (\tilde{r}_{us}(\Theta) - r_{us})^2, \tag{7}$$

where U represents all the users, $\mathcal{R}_{us}$ contains all the sets rated by the user u, r_{us} is the original rating of the user u on the set s and $\tilde{r}_{us}$ is the estimated rating of the user u on the set s.

Similarly, the model can be learned by minimizing the BPR loss function given by

$$\mathcal{L}_{bpr}(\Theta) \equiv -\sum_{u \in U} \sum_{\substack{s,t \in \mathcal{R}_{us}, \\ r_{us} > r_{ut}}} \ln \sigma(\tilde{r}_{us}(\Theta) - \tilde{r}_{ut}(\Theta)), \tag{8}$$

where s and t are sets rated by user u such that $r_{us} > r_{ut}$.

To control model complexity, we add regularization of the model parameters thereby leading to an optimization process of the following form:

$$\min_{\Theta} \mathcal{L}(\Theta) + \lambda(||\Theta||^2), \tag{9}$$

where λ is the regularization parameter and $\mathcal{L}(\theta)$ is the loss function, i.e., either $\mathcal{L}_{bpr}(\theta)$ or $\mathcal{L}_{rmse}(\theta)$.

Algorithm 1 LFS_{rmse}-Learn

procedure LFS$_{rmse}$-LEARN
  $\eta \leftarrow$ learning rate
  $\lambda \leftarrow$ regularization weights
  iter $\leftarrow 0$
  Initialize Θ randomly
  while iter $<$ maxIter or error on validation set decreases **do**
    for each user u **do**
      Sample a set s s.t. $s \in \mathcal{R}_{us}$
      Compute $\tilde{r}_{us}$ using equation 6
      $e_{us} \leftarrow (\tilde{r}_{us} - r_{us})$
      $v_k \in \mathcal{R}^k \leftarrow 0$
      for each item $i \in s$ **do**
        $v_k \leftarrow v_k + q_i$
      end for
      $p_u \leftarrow p_u - \eta(\frac{e_{us}}{|s|} v_k + \lambda p_u)$
      $b_u \leftarrow b_u - \eta(e_{us} + \lambda b_u)$
      $b_{us} \leftarrow b_{us} - \eta(e_{us} + \lambda b_{us})$
      for each item $i \in s$ **do**
        $q_i \leftarrow q_i - \eta(\frac{e_{us}}{|s|} p_u + \lambda q_i)$
        $b_i \leftarrow b_i - \eta(\frac{e_{us}}{|s|} + \lambda b_i)$
      end for
    end for
    iter $\leftarrow$ iter $+1$
  end while
  return Θ
end procedure

The optimization problem of the equation 9 can be solved by stochastic gradient descent algorithm.

VI. Experimental Evaluation

VII. Results and Discussion

VIII. Conclusion

References

[1] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems, 2009.

[2] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461, Arlington, Virginia, United States, 2009. AUAI Press.

Algorithm 2 LFS_{bpr}-Learn

procedure LFS$_{bpr}$-LEARN
  $\eta \leftarrow$ learning rate
  $\lambda \leftarrow$ regularization weights
  iter $\leftarrow 0$
  Initialize Θ randomly
  while iter $<$ maxIter or error on validation set decreases **do**
    for each user u **do**
      Sample a pair of set $s, t \in \mathcal{R}_{us}$ s.t. $r_{us} < r_{ut}$
      Compute $\tilde{r}_{us}$ and $\tilde{r}_{ut}$ using equation 4
      $\tilde{r}_{ust} \leftarrow \tilde{r}_{us} - \tilde{r}_{ut}$
      $\tau \leftarrow \frac{-1}{1+exp(\tilde{r}_{ust})}$
      $v_k \in \mathcal{R}^k \leftarrow 0$
      for each item $i \in s$ **do**
        $v_k \leftarrow v_k + q_i$
      end for
      for each item $i \in t$ **do**
        $v_k \leftarrow v_k - q_i$
      end for
      $p_u \leftarrow p_u - \eta(\frac{\tau}{|s|} v_k + \lambda p_u)$
      for each item $i \in s$ **do**
        $q_i \leftarrow q_i - \eta(\frac{\tau}{|s|} p_u + \lambda q_i)$
        $b_i \leftarrow b_i - \eta(\frac{\tau}{|s|} + \lambda b_i)$
      end for
      for each item $i \in t$ **do**
        $q_i \leftarrow q_i - \eta(\frac{-\tau}{|s|} p_u + \lambda q_i)$
        $b_i \leftarrow b_i - \eta(\frac{-\tau}{|s|} + \lambda b_i)$
      end for
    end for
    iter $\leftarrow$ iter $+1$
  end while
  return Θ
end procedure