Project Report

Part I Fundamentals

IEOR E4571
Personalization Theory and Application

Tian Pan	tp2517
Yiying Huang	yh2870
Zhichao Yang	zy2280

1. Abstract

This project focused on developing two specific recommendation algorithms, one is item-based collaborative filtering, which utilizes item similarity to make predictions based on available ratings, the other is model-based collaborative filtering, a matrix factorization algorithm is used to derive latent features that associate with users and items. By performing cross-validation on the models, this report seeks to obtain data that supposedly may clarify ambiguities regarding the accuracy of different algorithms. All of the model comparisons and evaluations are done with the RMSE metric.

2. Dataset

A dataset from RecSys2013, a LBS contest organized by Yelp, was used for our research. The raw dataset includes three subsets for businesses, users and reviews respectively. Sample data from the three tables are as follows. To scale down the size of the dataset, we firstly selected only currently open restaurants in Phoenix with a review count larger than 135. Then from the review dataset, all the reviews for the selected business including user id and ratings were filtered out. Next, a rating matrix with user id as index and business id as columns were set up and the cells were filled with corresponding ratings. The main issue with the dataset is its high sparsity. To reduce the sparsity of the dataset, users who rated less than two among all of the selected restaurants were filtered out. After the above preprocessing steps, the final rating matrix has 100 items/restaurants and 3873 users and 4.5% of the user-item ratings are available.

Figure 1. Sample Data


```
{"business_id": "rncjoVoEFUJGCUoC1JgnUA",
"full_address": "8466 W Peoria Ave\nSte 6\nPeoria, AZ 85345",
"open": true,
"categories": ["Accountants", "Professional Services", "Tax Services", "Financial Services"],
"city": "Peoria",
"review_count": 3,
"name": "Peoria Income Tax Service",
"neighborhoods": [],
"longitude": -112.241596,
"state": "AZ",
"stars": 5.0,
"latitude": 33.581867000000003,
"type": "business"}
```


Business


```
{"votes":
    {"funny": 0, "useful": 5, "cool": 2},
"user_id": "rLtl8ZkDX5vH5nAx9C3q5Q",
"review_id": "fWKvX83p0-ka4JS3dc6E5A",
"stars": 5, "date": "2011-01-26",
"text": "My wife took me here on my birthday for breakfast and it wa
"type": "review",
"business_id": "9yKzy9PApeiPPOUJEtnvkg"}
```


Review


```
{"votes":
    {"funny": 0, "useful": 7, "cool": 0},
 "user_id": "CR2y7yEm4X035ZMzrTtN9Q",
 "name": "Jim",
 "average_stars": 5.0,
 "review_count": 6,
 "type": "user"}
```


User

To implement cross validation(CV) for our models, we firstly randomly split user-item rating pairs into train set and test set with percentage 80%, 20% of the cleaned dataset respectively.

3. Objective

In this project, our goal is to use the correlation between users and restaurants to explore effective models to predict users ratings for restaurants they have not been to and therefore recommend top new restaurants for users based on the predicted ratings. We utilize CV to find a best set of hyper-parameters for each algorithm in order to maximize the accuracy. Since we selected restaurants having more than 135 ratings to reduce the sparsity of our data, the model may have a bad generalizability when applied to larger sets of data. Therefore, we may sacrifice a small improvement in accuracy during the training process to avoid overfitting. Considering the run-time cost, we would also sacrifice small improvement in accuracy for efficiency.

4. Algorithms

Given the cleaned user-restaurant dataframe for restaurants in Phoenix, we aimed to fill in the missing ratings in our sparse matrix. We looked at both nearest neighbor and matrix factorization methods in order to assess their predictive performance. However, since most collaborative filtering methods start off with a simple baseline model. We began by implementing the baseline model to our data:

$$r_{ui} = \bar{x} + b_u + b_i$$
$$\bar{x} = global\ average\ rating$$
$$b_u = user\ bias$$
$$b_i = restaurant\ bias$$

This model takes into account the mean rating given by each user and the mean rating received by each restaurant in order to remove the presence of any possible bias when predicting because some users may have tendency to rate everything higher or lower. We firstly used training set to train the model, for the users whose bias is NaN we considered this user bias to be 0, and finally got an RMSE of 1.146 on the test o set.

4.1 Algorithm 1: Item-based collaborative filtering

The Item-based collaborative filtering approach compares users and their ratings to a certain set of items. The algorithm calculates the nearest neighbors to a given user. Every neighbor's previously rated items are compared to the item in focus with an item similarity computation. When the most similar items have been determined, the prediction is computed by taking the weighted average of every neighbor's ratings of those items. The calculation of item similarities can be done with several methods, in this project we choose adjusted cosine similarity so that we

take in to the fact that different users have different rating schemes. The similarity of two restaurants are calculated as follows:

$$\text{AdjustedCosine}(i,j) = \frac{\sum_{u \in U_i \cap U_j} s_{ui} \cdot s_{uj}}{\sqrt{\sum_{u \in U_i \cap U_j} s_{ui}^2} \cdot \sqrt{\sum_{u \in U_i \cap U_j} s_{uj}^2}}$$

Then we can predict the rating for any user-item pair by using the idea of weighted sum. First we took all the restaurants similar to our target restaurant, and from those similar restaurants, we picked restaurants which the active user has rated. We weighted the user's rating for each of these restaurants by the similarity between that and the target restaurant. After that we scaled the prediction by the weighted sum of similarities of top k restaurants to get a reasonable value for the predicted rating. If the target restaurant has no neighbor, we assigned average rating of this restaurant by other users from the dataset to it:

$$\hat{r}_{ut} = \frac{\sum_{j \in Q_t(u)} \text{AdjustedCosine}(j,t) \cdot r_{uj}}{\sum_{j \in Q_t(u)} |\text{AdjustedCosine}(j,t)|}$$

We evaluated this model through cross validation RMSE, where we firstly trained the model by using 75% of the data in the train set, and then tuned the model by using the remaining 25% data in the train set in order to do cross validation to find the best neighborhood size. We tested neighborhood size from 1 to 20 and found that the best neighborhood size was 10 with error rate(RMSE).We finally got an RMSE of 1.207 on our test set for the item-based collaborative filtering model.

4.2 Algorithm 2: Model-based collaborative filtering

Singular-value decomposition was used to estimate the missing rating in the rating matrix. This algorithm reduces the dimension of the original $m \times n$ matrix to two smaller matrices $U_{m \times k}$ and $V_{k \times n}$ where U represents the strength of the association between users and the k latent features while V represents the strength of the associations between each item and the k latent features.

4.2.1 Python Scipy Package - SVD

In order to perform the SVD algorithm using Python Scipy package, blank cells needed to be filled first. We selected to fill the missing cells with user mean rating calculated from the rating matrix.

The parameter in the model is the number of latent space (k) needed. A set of k, [3,5,10,15,20,25,30], was tested through cross-validation. The original dataset was splitted to training dataset (80%) and test dataset (20%). For the training dataset, 5-fold cross-validation was applied so each of the parameter k was evaluated in each iteration.

4.2.2 Matrix Factorization by Stochastic Gradient Descent

Considering that the packaged above is not compatible with missing values and the selection of input value may lead to inaccuracy, the stochastic gradient descent method was also applied to solve the matrix factorization problem. Firstly, a model with L2 regularization was used to avoid overfitting at the cost of introducing bias.

$$\widehat{r_{ui}} = U_u^T V_i$$

Cost Function

$$\min \sum (r_{ui} - U_u^T V_i)^2 + \frac{\lambda}{2}(||U_u||^2 + ||V_i||^2)$$

$$e_{ui} = r_{ui} - \widehat{r_{ui}}$$

$$U_{uk} = U_{uk} + \Upsilon(e_{ui} V_{ik} - \lambda U_{uk})$$

$$V_{ik} = V_{ik} + \Upsilon(e_{ui} U_{uk} - \lambda V_{ik})$$

The stochastic gradient descent was implemented with 400 steps, gamma = 0.02, lambda = 0.3. A simple cross-validation setup with a training set, a tuning set and a test set was used to evaluate the number of k needed due to the run-time limit. The RMSE value converged to a relatively stable value after 400 iterations. Not many choices of lambda was tested because of the slow run-time. In practice, the expensive calculation cost limits the optimization of the learning rate.

Secondly, user and item bias terms were added to the model in order to improve the accuracy. The user and item bias were dynamically updated along with the two matrices U and V. Different from the previous SGD, the two matrices were extended to k+2 columns to include the two bias.

$$\begin{aligned} u_{i,k+1} &= o_i \quad \forall\, i \in \{1, \ldots\ldots, m\} \\ u_{i,k+2} &= 1 \quad \forall\, i \in \{1, \ldots\ldots, m\} \\ v_{j,k+1} &= 1 \quad \forall\, j \in \{1, \ldots\ldots, n\} \\ v_{j,k+2} &= p_j \quad \forall\, j \in \{1, \ldots\ldots, n\} \end{aligned}$$

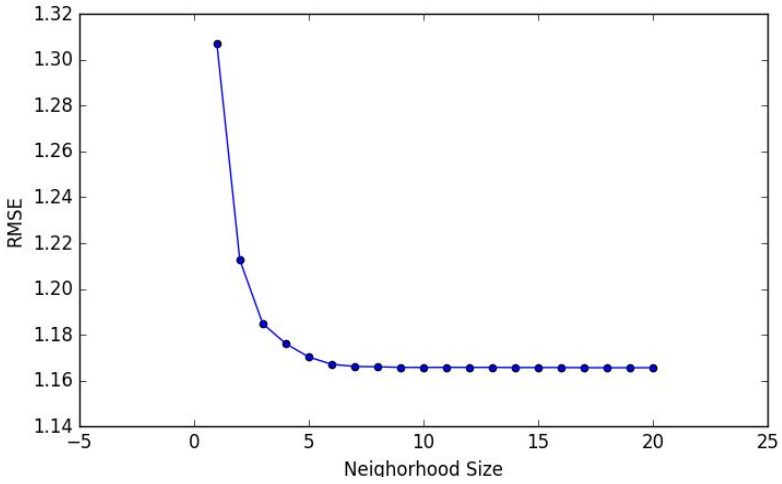
5. Accuracy

5.1 Accuracy of Item-based Collaborative Filtering

The accuracies of using different number for neighborhood sizes were really close to each other when neighborhood size k is larger than 10, the RMSE value varies little and was not constantly decreasing. Although when RMSE is a little smaller(0.0001) when k>10 than when k=10. We

decided to choose k=10 since we do not have regularization in the model and we wanted to avoid overfitting for the tuning dataset. The RMSE on the tuning set is 1.16573 and the the RMSE on the test set is 1.20699. This model didn't perform very well in testing and RMSE is actually even higher than the Baseline. We believe that it is mainly due to the sparsity of the data.

Figure 2. RMSE vs. Neighborhood Size

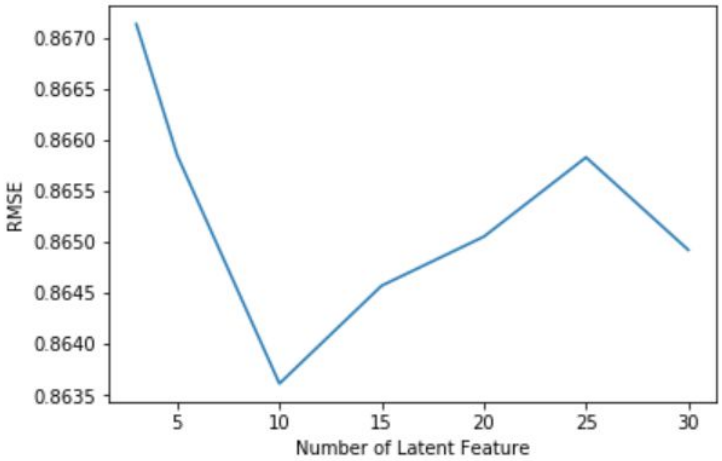


5.2 Accuracy of Model-based Collaborative Filtering

5.2.1 Python Scipy SVD package

Cross-validation shows that k = 10 gives the smallest RMSE value for the tuning dataset but the accuracies of using different number of features were close to each other. The test set RMSE when k = 10 was 0.87, which was close to the training set RMSE values and lower than the baseline model RMSE. The training error is not significantly smaller than the test error, so the model does not have a large overfitting issue . The run-time is roughly 153s.

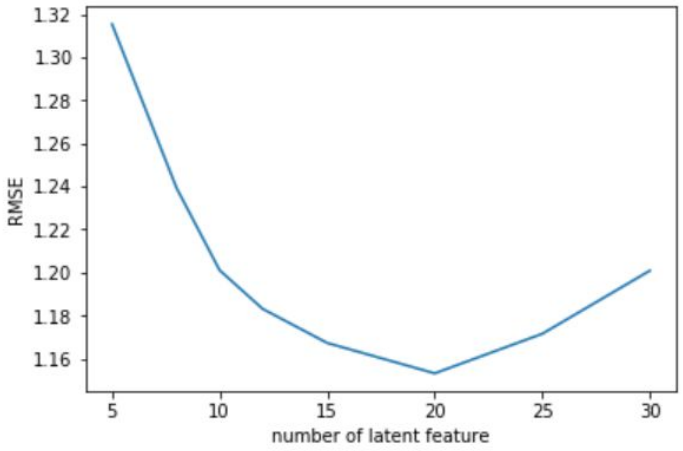
Figure 3. RMSE vs. Number of Latent Features



5.2.2 Matrix Factorization by Stochastic Gradient Descent (SGD)

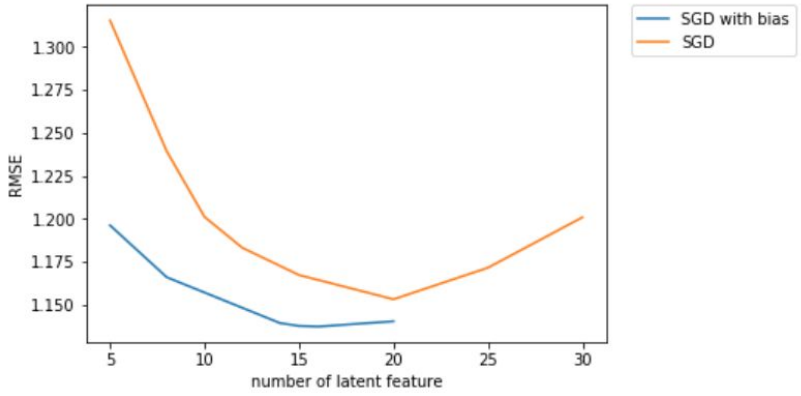
The modeling training using SGD algorithm started with L2 regularization terms and the result shows that the optimal number of latent feature is 20. The RMSE values obtained were generally larger than the SVD method mentioned above. The test set RMSE is 1.1398, which is lower than the baseline model RMSE. The run-time of the test set was 1355s, which is much longer than the SVD method.

Figure 4. RMSE vs. Number of Latent Feature (without user/item bias)



To further improve the the accuracy, user and item bias terms were added to the model. The result turned out to be slightly improved and the minimal RMSE was achieved around k = 15. However, the run-time for this model was slowed to about 4600s. Therefore, the improvement in the accuracy is not efficient considering the run-time cost.

Figure 5 . Comparison between SGD with and without bias terms



Based on the results above, SVD is a better algorithm for this dataset regarding to accuracy and runtime. Therefore, the coverage test below for model-based collaborative filtering was applied to the SVD method only.

6. Coverage

Since we originally selected restaurants having more than 135 ratings to reduce the sparsity, the models were expected to have a poor performance when applied to larger datasets which will be more sparse in general. The four expanded datasets contain restaurants having more than 95, 105, 115 and 125 reviews respectively. This means that as the dataset size increases, the dataset is more sparse (Table1).

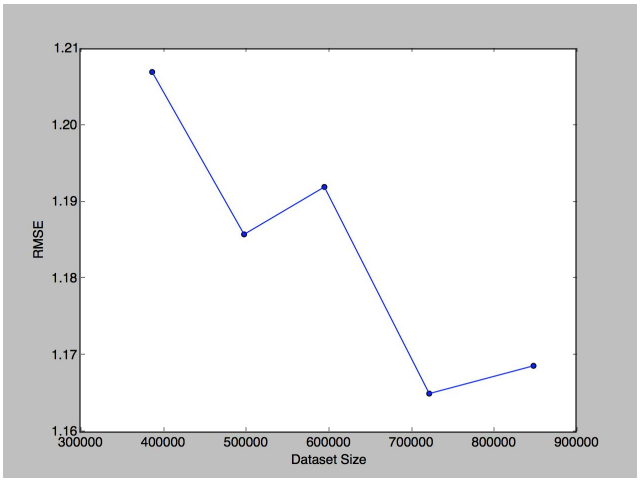
Table 1. Four expanded datasets with sparsity statistics.

Dataset	Dataset Size	Valid user-item pair	Valid user-item pair %
data95	848598	24430	2.88%
data105	722455	22872	3.17%
data115	595620	21176	3.56%
data125	498491	19679	3.95%
Data135 (original)	387300	17651	4.56%

6.1 Coverage of Item-based Collaborative Filtering

The accuracy of the item-based collaborative filtering goes up and down as we increase the size of the dataset but the over trend show that the accuracy is increasing because the RMSE decreases overall as the size of dataset or the number of rating increases even though the sparsity increased as well. The run-time also increases when we increases the size of the dataset.

Figure 6. Item-based collaborative filtering: RMSE vs. Dataset Size



6.2 Coverage of Model-based Collaborative Filtering

The RMSE values for larger and more sparse datasets are slightly larger than the original dataset, which means, as the percent of valid ratings increases, the RMSE values tend to have a decreasing trend. While doubling the size of the dataset, run-time increases by more than doubled.

Figure 7. SVD: Dataset Size vs. RMSE

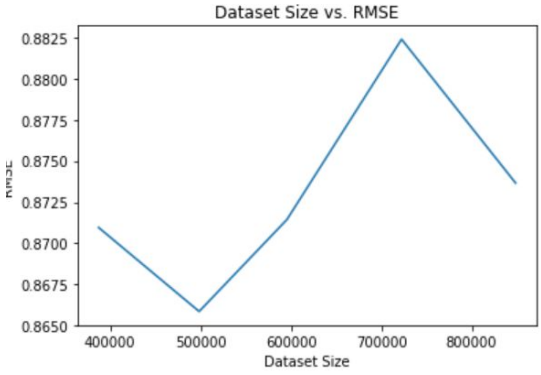


Figure 8. SVD: Sparsity vs. RMSE

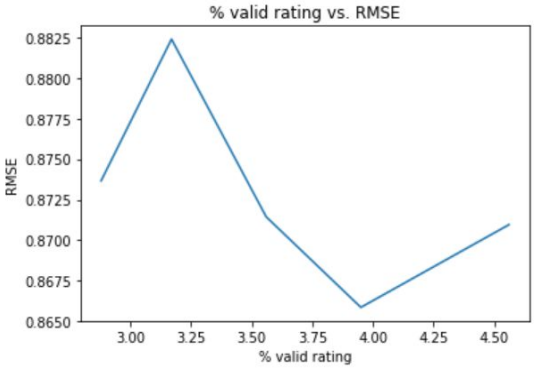
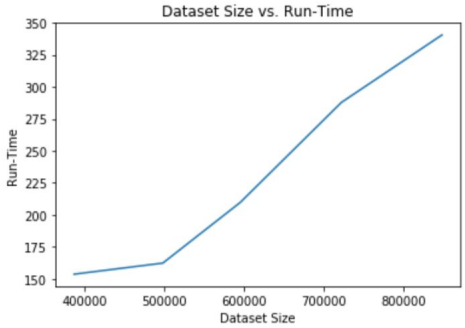
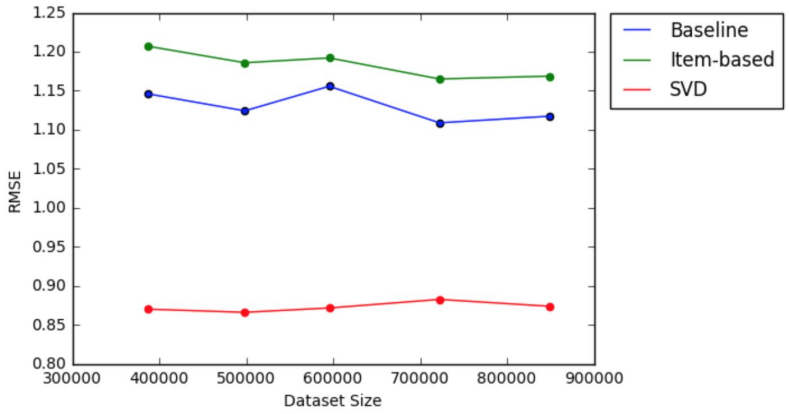


Figure 9. Run-time (s) vs. Dataset Size



7. Conclusion

Figure 10. Accuracy comparison



The prediction accuracy of a recommender system is heavily depend on various factors. The properties of a dataset plays a major role as well as the parameters of the algorithms. As the plot shows, we can easily find that both baseline model and item-based collaborative filtering model are optimized for the smallest dataset while the SVD algorithm performs better on the larger, sparser dataset. We can also notice that the baseline model is slightly better than the item-based collaborative filtering mode. This occurs because of the sparseness of the matrix, since the reliability of vector-based similarity matrices depends on the number of shared ratings and the dataset we have is too sparse. Also, given the sparseness of the data, it was no surprise that the matrix factorization techniques performed a lot better than the item-based collaborative filtering and baseline model. We can conclude that the SVD algorithm is seemingly more accurate than the item-based collaborative filtering algorithm when it comes to scaling from small to large datasets.

8. References

[1] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst., 22(1):143–177, Jan. 2004.

[2] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International Conference on World Wide Web, WWW ’01, pages 285–295, New York, NY, USA, 2001. ACM.

[3] G. Shani and A. Gunawardana. Evaluating recommender systems. Technical Report MSR-TR-2009-159, November 2009.

[4] http://surpriselib.com/

[5] http://blog.csdn.net/recsysml/article/details/12287513

[6] Aggarwal, Charu C. . Recommender Systems: The Textbook, New York: Springer; 1st ed. 2016 edition (March 29, 2016).

[7] https://beckernick.github.io/matrix-factorization-recommender/