Assignment 5

Team name: M2 Robo

Lee Chun Yin	Chiu Yu Ying	Chan Kwan Yin
3035469140	3035477630	3035466978
		Team leader

1. Introduction

1.1. collaborative filtering (CF)

CF is a technique for recommendation system, in which historical feedback data are used to infer connections between users and products [3]. While additional features can be introduced to offset certain bias effects [1], two inputs (user and product) and one output (user rating on the product) are generally sufficient to train a CF model without involving domain-specific data.

Two major approaches for CF include neighbourhood models and latent factor models. Neighbourhood models compare the similarity between users and recommend products positively rated by similar users, while latent factor models perform dimensional reduction on both users and movies to a common, smaller set of feature attributes such that users are recommended with movies of more coherent features.

1.2. The Netflix Prize dataset

The Netflix Prize is a competition for the prediction of users' favour of movies. The dataset provides existing ratings of users on given movies, and models are trained to predict new ratings.

1.2.1 Dataset format

The dataset contains 100480507 rows of data structured in the following format:

User ID	490189 discrete values
Movie ID	17770 discrete values
Rating	1, 2, 3, 4, 5
Date	Dates from 1999 to 2005

1.2.2 Distribution of ratings

Ratings are mostly distributed around 3 and 4, as shown in Figure 1.

Except for some extreme cases, the number of ratings per user over the 7 years mostly follow an exponential relation for users from 10 to 1000 ratings, as shown in Figure 2. The top 10 users with the highest number of ratings range from 17651 to 8877.

For movies with at least 100 ratings (which is the case for the majority), their numbers of ratings demonstrate a similar but more concave relationship, as shown in Figure 3.

1.2.3 Evaluation process

To evaluate performance, 1425333 rows (about 1.42% of all data) are speified as the standard "probe". We perform analysis in the following procedure:

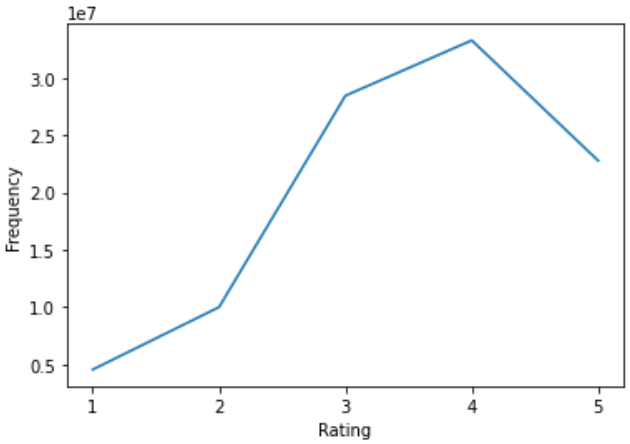

Figure 1. Frequency of ratings

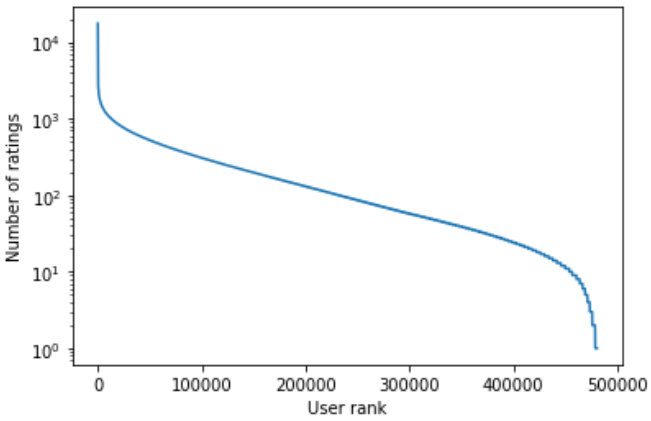

Figure 2. Number of ratings per user

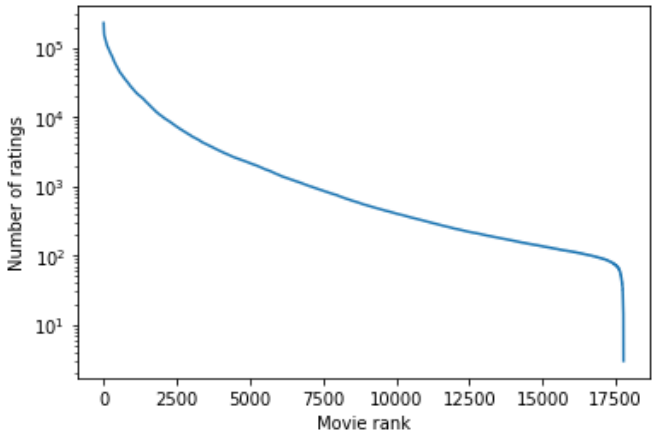

Figure 3. Number of ratings per movie

1. Train the model with the 99055174 non-probe rows.
2. Predict user ratings with the 1425333 probe rows.
3. Compute the root mean squared error (RMSE) between the predicted data and actual data.

In this project, we evaluate three models, namely:

- k-nearest neighbours (KNN), a simple neighbourhood model
- singular value decomposition (SVD), a latent factor model that accounts for user bias
- neural collaborative filtering (NCF), a latent factor model that represents features with neural network weights

Originally, we proposed three models KNN, SVD and SVD++ (a SVD model with time features). Due to the high similarity between SVD and SVD++, we proposed the use of NCF instead, while the additional time effect bias in SVD++ is left out as a to-do.

2. Conclusion

2.1. KNN model

2.1.1 Methodology

The original KNN model we proposed constructs an $m \times n$ matrix (where m, n are the numbers of users and movies) to store all ratings, and an $m \times m$ matrix to store user-user similarity. It occurred that this algorithm was too expensive in terms of memory usage, and computation of $O(m^2)$ similarity was too time-consuming. We instead adopted a technique called Top Q optimization proposed by Hong et al [2], which only selects neighbours from the Q users with the most ratings.

Nevertheless, we are concerned that this would lead to unreliable results due to potential spamming or otherwise "light-hearted" ratings. As observed from Figure 2, the top-most user has 17651 ratings, implying that he/she has on average rated almost 7 movies per day. We are doubtful about the integrity of such data, whether they are genuinely representative of the rest of the users. This is also vulnerable to attacks from malicious robots who attempt to hijack recommendation systems through automatic rating. Even if each rating requires completion of a captcha, it only requires 800 manual completions per day to completely fill the top 100 users. This optimization is also unable to cater for interests from minority groups.

2.1.2 Findings

We discovered that the result using cosine with mean-corrected ratings has the best performance among all (RMSE = 1.1046) while the result using cosine with 3-corrected ratings does not work (with RMSE = 1.1800). The Netflix rating is a 5-star rating which is like a Likert scale commonly used in research surveys. In fact, such a scale has a known issue called the central tendency bias, which means people having surveys tend to have a neutral or middle rating and avoid the endpoints of scale [4]. The poor performance of 3-corrected rating shows the dataset does not exactly follow the same situation of the bias mentioned.

However, according to the frequency graph in 1, we can see the ratings accumulate at 3-4,. This may be because viewers, who are potentially giving unfavour rating, will simply close the movie and choose the other one after clicking and watching the preview of certain movie. The potential raters only remain those who are still interested in the movie after watching the preview. With the central tendency bias, those remaining viewers have a high tendency to give a rating among the favorable rating range (i.e. between 3 and 5). In this case, 4 is the "middle" to be selected. Apart from that, we also observe that there are fewer unfavorable (less than 3) ratings. This may be because only those viewers, who watched most of the movie, feel disappointed will give unfavorable ratings. This may be the reason for the less counts for unfavorable ratings.

Therefore, based on these cases, the "middle" of the rating is not just 3 and also 4. The mean-corrected ratings can consider the mentioned situation.

2.2. SVD model

2.2.1 Methodology

Contrary to the originally proposed model, simple gradient descent was not feasible due to memory constraints. We opted an stochastic gradient descent (SGD) algorithm with minibatch optimization instead. Due to performance reasons, vectorization of the algorithm was necessary, but this results in less readable representations of the descent formula, which has caused issues with diverging RMSE in some cases, which are more difficult to debug since the numeric output is not convenient to evaluate.

2.2.2 Findings

In the SVD model, we observed that train and test RMSE are inversely related over different selections of hyperparameters. Figure 4 visualizes the results of different hyperparameters we tested with. This suggests that the issue of overfitting is very prevalent in latent factor models.

In particular, the two models with the lowest test RMSE reduced the factorization dimension greatly down to $k = 50$ and $k = 10$ respectively. This suggests that the model starts memorizing specific users once the the number of users is comparable to the number of dimensions. Regularization was similarly observed to have the effect of sacrificing train RMSE for test RMSE.

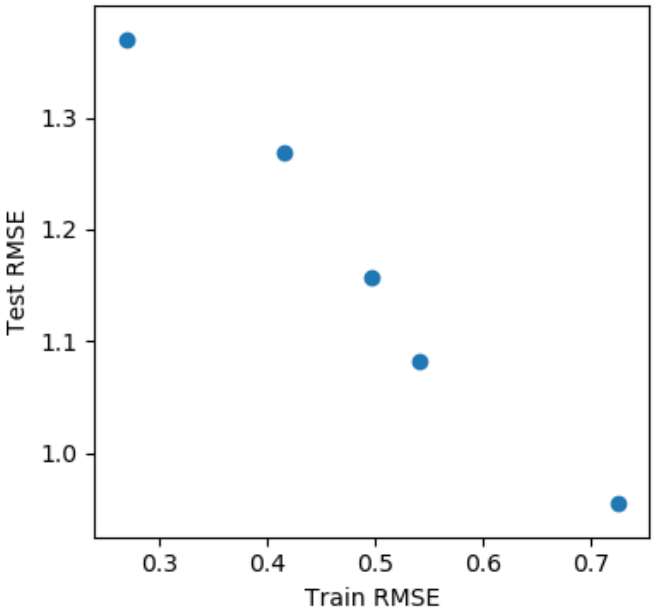

Figure 4. SVD Train RMSE vs Test RMSE

2.2.3 Further enhancement

BellKor suggested that rating date has a bias effect on the rating value [1]. We plan to adopt this into the model so that the exact bias effects can be visualized more clearly.

Further interpretation and analysis on the values trained in the model would also be helpful in investigating the overfitting issue involved. In particular, it would be useful to find the existence of any complementary pairs of user-feature/movie-feature weights, which imply the excess of the k hyperparameter selection, or singleton weights, which imply direct memorization of a specific user/movie.

2.3. NCF model

2.3.1 Methdology

For the neural network model, we mainly faced difficulties in the training process. Firstly, due to the complexity of the neural network model, we faced difficulties in the training process for the whole dataset, and had to use optimization techniques such as increasing batch size in order to complete the training in reasonable time.

Second, for the matrix factorization part of the neural network model, issues were faced for the training loss, where the training loss was unable to converge to zero even without model regularization and was stuck at RMSE ≥ 1. More investigation on the model training and neural network design needs to be done to make sure the network is designed such that the loss can converge to a low value.

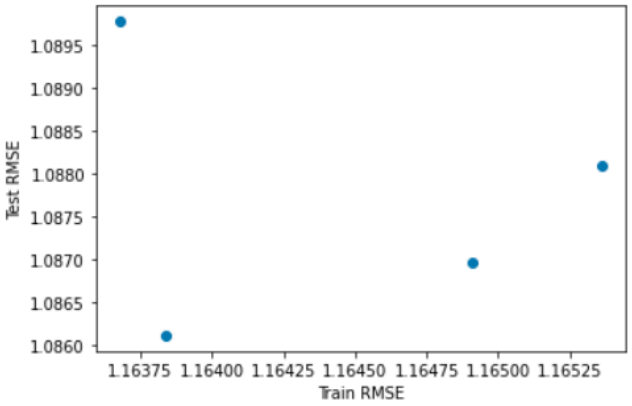

Figure 5. NCF Train RMSE vs Test RMSE

2.3.2 Findings

Generally the different combinations of hyperparameters do not have a huge impact on the test RMSE ranging from 1.08611 to 1.08977. One exception is the case with a huge batch size ($N = 1048576$) and less number of epoch ($x = 8$) when compared with the other case with $N = 65535$ and $x = 9$. The test RMSE of the former case is 2.23113, which shows a significant difference in performance with the other case. This proves that the smaller batch size can allow the model to have more cycles to learn and improve its performance.

Another interesting finding is that the train RMSE values in all models are higher than their corresponding test RMSE value, as shown in Figure 5. A possible improvement is to further reshuffle the entire dataset to avoid the abnormal issue.

References

[1] Robert M Bell, Yehuda Koren, and Chris Volinsky. The bellkor 2008 solution to the netflix prize. *Statistics Research Department at AT&T Research*, 1(1), 2008. 2, 4

[2] Ted Hong and Dimitris Tsamis. Use of knn for the netflix prize. *CS229 Projects*, 2006. 3

[3] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008. 2

[4] Stanley S Stevens. Issues in psychophysical measurement. *Psychological review*, 78(5):426, 1971. 3