# Amazon Fine Foods Recommendation

Haoming Zhang
University of California, San Diego
haz141@eng.ucsd.edu

Guoren Zhong
University of California, San Diego
gzhong@eng.ucsd.edu

## Abstract

*Recommendation system has been a popular topic in the past few decades due to the increasing amount of information in our life. It is an important approach in solving the information overloading problem, and has been applied to many areas, such as product recommenders for online stores and content recommenders for social media platforms[1][2]. In this work, two algorithms are introduced to solve the Amazon fine food recommendation problem, which are collaborative filtering[3] and latent factor model[4]. These algorithms are used to predict the ratings of users to products, and generate a recommendation list for users. The results of both models are shown in the end.*

## 1. Introduction

Recommendation systems are now a popular topic due to the need of managing increasing amount and size of data. It has been applied to many area, such as online shopping and media recommendations. In the past decade, people show a rising preference on online shopping rather than physical shopping, and it is predicted that by 2021, online shopping would have dominated online sales, reaching 54% of all sales [5]. Among all online shopping options, Amazon is one of the most popular shopping platforms in the united states. According to a survey [6], over 89% of the customers would like to do online shopping with Amazon again. Thus, a reliable algorithm of recommending suitable products to specific customers would be of great importance due to its strong correlations to the profits.

### 1.1. Dataset

The data is "Amazon Fine Food Reviews"[7] from Kaggle, which includes 568,454 reviews, 256,059 users and 74,258 products. Among those users, 260 of them have more than 50 reviews. The original dataset have 10 columns of data in total. To cleanse up the dataset, the uncorrelated columns like time and summary are dropped. Besides, the users with less than 20 reviews are also ignored for the purpose of reducing redundancy. After all the manipulations, the new dataset has 3 columns (**ProductID**, **UserID** and **Score**) and 68,015 reviews left. During training, the dataset is separated randomly into 80% of training and 20% of testing.

### 1.2. Tools and models

In this work, two algorithms are introduced to solve the Amazon fine food recommendation problem, which are collaborative filtering (CF)[3] and latent factor model (LFM)[4]. CF is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences information from other users (collaborating). There are different types of collaborative filtering models, such as memory (similarity)-based CF [8], model-based CF [9], and deep-learning CF [10]. Latent factor model is a statistical model that relates a set of observable (manifest) variables to a set of latent variables. In the following sections, these models will be discussed and derived in detail, and the performance of them are shown at last.

## 2. Problem Formulation

The problem a recommendation system problem divided into two parts, rating predictions and product recommendations.

### 2.1. Rating predictions

Complete data is typically not available in reality. For example, as shown in Table.1, there are some missing links between users and products. The reason for these blanks is either the user has not bought this product before or he did not rate it after using. Thus, our first objective is to predict the missing ratings and fill all the blanks in a given dataset. Both collaborative filtering model and LFM would be used in this task, and their performances are evaluated by mean-squared-error (MSE).

### 2.2. Product recommendations

After filling all the blanks in the table, a recommended products list can be formed based on the ratings. In this part,

| | Product 1 | Product 2 | Product 3 | Product 4 |
|---|---|---|---|---|
| User A | 5.0 | | 4.0 | 3.4 |
| User B | 3.0 | 3.6 | 3.5 | |
| User C | | 4.6 | | 4.8 |
| User D | 5.0 | | 3.5 | |

Table 1: Example of recommendation system problems

only the model with better performance in rating predictions would be used.

## 3. Method

In this section, two approaches would be discussed in detail.

### 3.1. Baseline

The baseline in this project is the results predicted by average. In other words, the average rating of all reviews in the training is calculated, then it is used as the estimation for all data in the test set.

### 3.2. Collaborative filter

In this work, we focus on the memory-based collaborative filter (CF), which includes two different types, user-based and item-based.

#### 3.2.1 User-based CF

A user-based CF first finds the similarities between user pairs, then computes the predicted ratings for from a user to products. The similarity can be measured by multiple rules, in which the Jaccard similarity[11] is the most simple one. For two non-empty finite sets $U$, $V$, the Jaccard similarity coefficient is the number of elements in the intersection $U \cap V$[12]:

$$J(U,V) = \frac{|U \cap V|}{|U \cup V|} = \frac{|U \cap V|}{|U|+|V|-|U-V|}. \quad (1)$$

In recommendation system problems, each user's preference is considered as a finite set, and the similarity between two users is measured using Eqn.1. Besides, another popular rule is called cosine similarity[13]. It treats two users' preferences into vectors, and measures their similarity by means of a cosine function:

$$\begin{aligned} C(u,v) = \cos(\vec{u},\vec{v}) &= \frac{\vec{u}\cdot\vec{v}}{\|\vec{u}\|\|\vec{v}\|} \\ &= \frac{\sum_{p\in P_{uv}} r_{u,p} r_{v,p}}{\sqrt{\sum_{p\in P_{uv}} r_{u,p}^2 \sum_{p\in P_{uv}} r_{v,p}^2}}, \end{aligned} \quad (2)$$

where $P_{uv}$ is the set of products that both user $u$ and user $v$ have rated, and $r_{u,p}$, $r_{v,p}$ are the ratings from users $u$ and $v$ to the product $p$. Apart from the above two, Pearson correlation coefficient[14] is also a widely-used rule for measuring similarities, defining as the covariance of the two variables divided by the product of their standard deviations:

$$\begin{aligned} \rho_{u,v} &= \frac{\mathrm{cov}(u,v)}{\sigma_u \sigma_v} \\ &= \frac{\sum_{p\in P_{uv}} (r_{u,p}-\bar{r}_u)(r_{v,p}-\bar{r}_v)}{\sqrt{\sum_{p\in P_{uv}} (r_{u,p}-\bar{r}_u)^2 \sum_{p\in P_{uv}} (r_{v,p}-\bar{r}_v)^2}}, \end{aligned} \quad (3)$$

where $\bar{r}_u$ and $\bar{r}_v$ are the average ratings of users $u$ and $v$ respectively.

With the selected similarity rule, the second step of CF is to make predictions using a weighted sum of ratings from other users to a specific product $p$. According to this, the user-based collaborative filtering model is defined as follow:

$$r(u,p) = \bar{r}_u + \frac{1}{Z} \sum_{v\in U_p \setminus \{u\}} (r_{v,p} - \bar{r}_v) \cdot \mathrm{sim}(u,v), \quad (4)$$

where $\mathrm{sim}(u,v)$ is the similarity between $u$ and $v$ mentioned above, $U_p$ is the set of users that have rated the product $p$, and $Z$ is the normalization constant:

$$Z = \sum_{v\in U_p \setminus \{u\}} |\mathrm{sim}(u,v)|. \quad (5)$$

#### 3.2.2 Item-based CF

The idea of item-based CF is similar to that of user-based CF. The only different is that an item based CF finds the similarities between item pairs instead of users. The similarity rules also focus on items now:

$$J(P,Q) = \frac{|P \cap Q|}{|P \cup Q|} = \frac{|P \cap Q|}{|P|+|Q|-|P-Q|}. \quad (6)$$

$$\begin{aligned} C(p,q) = \cos(\vec{p},\vec{q}) &= \frac{\vec{p}\cdot\vec{q}}{\|\vec{p}\|\|\vec{q}\|} \\ &= \frac{\sum_{u\in U_{pq}} r_{u,p} r_{u,q}}{\sqrt{\sum_{u\in U_{pq}} r_{u,p}^2 \sum_{u\in P_{pq}} r_{u,p}^2}}, \end{aligned} \quad (7)$$

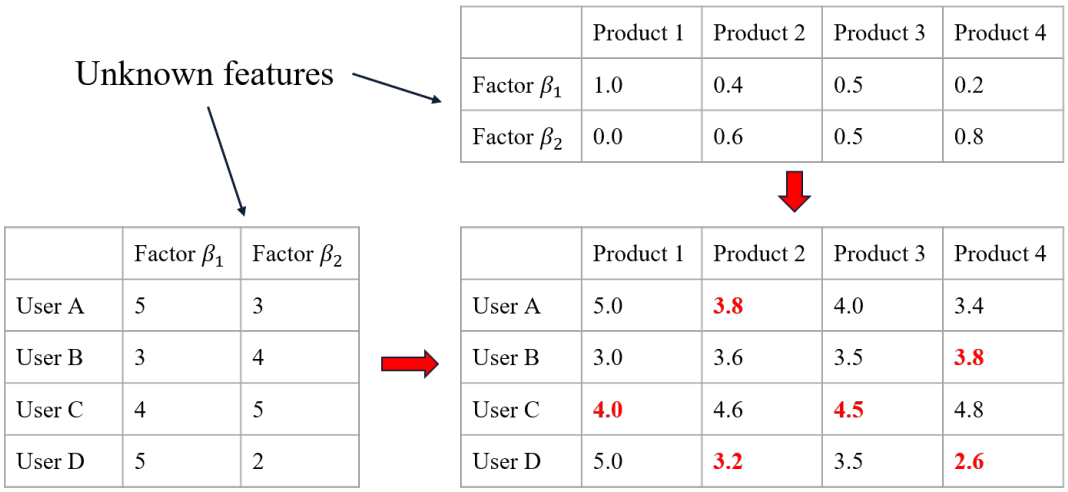


Figure 1: LFM example

$$\rho_{p,q} = \frac{\mathrm{cov}(p,q)}{\sigma_p \sigma_q} = \frac{\sum_{u \in U_{pq}} (r_{u,p} - \bar{r}_p)(r_{u,q} - \bar{r}_q)}{\sqrt{\sum_{u \in U_{pq}} (r_{u,p} - \bar{r}_p)^2 \sum_{u \in U_{pq}} (r_{u,q} - \bar{r}_q)^2}}, \tag{8}$$

In the above equations, $P$, $Q$ are two different product sets, $p$, $q$ represent two products, and $U_{pq}$ is the set of users that rated both $p$ and $q$. Then, the item-based collaborative filter is defined as:

$$r(u,p) = \frac{1}{Z} \sum_{q \in P_u \setminus \{p\}} r_{u,q} \cdot \mathrm{sim}(p,q), \tag{9}$$

with

$$Z = \sum_{q \in P_u \setminus \{p\}} \mathrm{sim}(p,q). \tag{10}$$

### 3.3. Latent factor model

The second model implemented in this project is the Latent factor model (LFM). LFM is a model based on matrix factorization as shown in Fig.1, where $\beta_1$, $\beta_2$ are two unknown factors. The model in general is to learn the ration of the unknown factors $\beta$ for both users and products:

$$f(u,p) = \alpha + \beta_u + \beta_p \tag{11}$$

where $\alpha$ is the global average, and $\beta_u$ and $\beta_p$ are the unknown factors for a user and a product respectively. In fact, $\beta_u$ describes how much a user tends to rate above (or below) the average and the $\beta_p$ describes the how much a product itself tend to be rated above (or below) the average.

Based on Eqn.11, this can be considered as an optimization problem as:

$$\arg\min_{\alpha,\beta} \sum_{u,p} (\alpha + \beta_u + \beta_p - r_{u,p})^2 + \lambda \left[ \sum_u \beta_u^2 + \sum_p \beta_p^2 \right], \tag{12}$$

where $r_{u,p}$ is the rating for user to the specific product and $\lambda$ is the regularization-parameter to be tuned. Eqn.12 can be solved by constructing the Lagrangian and setting all derivatives to 0. Hence, the update rules become:

$$\alpha = \frac{\sum_{u,p \in training} (r_{u,p} - (\beta_u + \beta_p))}{N_{training}}. \tag{13}$$

$$\beta_u = \frac{\sum_{p \in P_u} (r_{u,p} - (\alpha + \beta_p))}{\lambda + |P_u|}. \tag{14}$$

$$\beta_p = \frac{\sum_{u \in U_p} (r_{u,p} - (\alpha + \beta_u))}{\lambda + |U_p|}. \tag{15}$$

In this above equations, $U_p$ represents the set of the users that rated the product $p$ and $P_u$ represents the set of products that are rated by the user $u$.

## 4. Experiment and Result

In this section, the rating predictions results from both model are presented, and the recommendations based on these results are also shown in the end.

### 4.1. Rating predictions

As is mentioned above, the "predict-by-average" model is used as the baseline in the rating prediction task. The MSE of this method on the test set is 1.37, which would be the baseline for this project.

For the collaborative filtering model, we applied all three similarity rules on two different types of models. The final

| | Jaccard | Cosine | Pearson |
|---|---|---|---|
| User-based | 1.002 | 0.842 | 0.736 |
| Item-based | 0.742 | 0.626 | 2.206 |

Table 2: MSE for different collaborative filters.

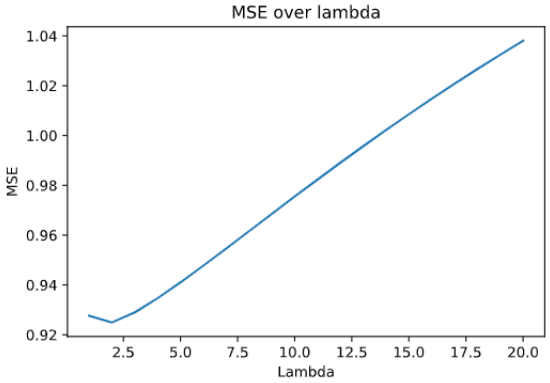


Figure 2: MSE vs. $\lambda$ in latent factor model

results are shown in Table2. Nearly all models are better than the baseline except for the user-based model equipped with Pearson correlation coefficient as the similarity rule. One possible reason for this is that many items are only rated by one or two users. Thus, the item vector may become $\mathbf{0}$ after subtracting the mean from it, ending up with singularities. On the other hand, the item-based CF with cosine similarity has the lowest MSE at around 0.626. Therefore, we conclude that Jaccard similarity and Cosine similarity work better with item-based model, while Pearson correlation coefficient prefers user-based model.

For the latent factor model, after fine tuning the regularization parameter $\lambda$, the best MSE for the model is 0.92 when $\lambda$ is 2 (Shown in Fig. 2). Though this result is not as good as that of the collaborative filter, it is still a lot better than the baseline.

### 4.2. Recommendations

Since the item-based collaborative filter with Cosine similarity performs the best among all models, it would be used for the final recommendations.

In this work, the recommendations are done in the "products-per-user" perspective. In other words, every user would receive recommendations of products that has ratings grater than 3. The main idea of this approach is to recommend as many products to a user as possible since wrong recommendations in fact would have little influence. On the other hand, the chance that a user would check and buy a product could be increased greatly with more the recommendation coming. The full results can be found on our Github repository [15].

## 5. Conclusion

In this project, multiple models for recommendation system are built based on collaborative filtering and latent factor model. Most models show great improvement from the baseline, and item-based collaborative filter equipped with Cosine similarity has the highest accuracy. Future works can be focused on improvements by using different similarity rules for the collaborative filter and using different optimizers for the latent factor model. Besides, introducing more data or more parameters may also help.

## Acknowledgement

We would like to thank Professor Manuela Vasconcelos for introducing our team into the topic of statistical learning. We would also like to thank the team members in this project for the amazing work. Haoming was responsible for analysing the data and building the latent factor model. Guoren took charge of cleaning up the data and implementing the collaborative filtering model. Without their contribution, this project wouldn't be accomplished.

## References

[1] Goel, Ashish, et al. "The who-to-follow system at Twitter: strategy, algorithms, and revenue impact." Interfaces 45.1 (2015): 98-107.

[2] Qu, Zhiguo, et al. "A novel quantum image steganography algorithm based on exploiting modification direction." Multimedia Tools and Applications 78.7 (2019): 7981-8001.

[3] Goldberg, David, et al. "Using collaborative filtering to weave an information tapestry." Communications of the ACM 35.12 (1992): 61-70.

[4] Simon, F. "Try This at Home". (2006). http://sifter.org/ simon/journal/20061211.html

[5] Ouellette, C. (2021, January 06). "Online Shopping Statistics You Need to Know in 2021". Retrieved March 08, 2021, from https://optinmonster.com/online-shopping-statistics/

[6] Mohsin, M. (2021, February 05). "10 Amazon statistics you need to know in 2021". Retrieved March 08, 2021, from https://www.oberlo.com/blog/amazon-statistics

[7] Project, S. (2017, May 01). "Amazon fine food reviews". Retrieved March 08, 2021, from https://www.kaggle.com/snap/amazon-fine-food-reviews

[8] Yu, Kai, et al. “Probabilistic memory-based collaborative filtering.” IEEE Transactions on Knowledge and Data Engineering 16.1 (2004): 56-69.

[9] Xue, Gui-Rong, et al. “Scalable collaborative filtering using cluster-based smoothing.” Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. 2005.

[10] He, Xiangnan, et al. “Neural collaborative filtering.” Proceedings of the 26th international conference on world wide web. 2017.

[11] Jaccard, Paul. “Lois de distribution florale dans la zone alpine.” Bull Soc Vaudoise Sci Nat 38 (1902): 69-130.

[12] Levandowsky, Michael, and David Winter. “Distance between sets.” Nature 234.5323 (1971): 34-35.

[13] Adomavicius, Gediminas, and Alexander Tuzhilin. “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions.” IEEE transactions on knowledge and data engineering 17.6 (2005): 734-749.

[14] Resnick, Paul, et al. “Grouplens: An open architecture for collaborative filtering of netnews.” Proceedings of the 1994 ACM conference on Computer supported cooperative work. 1994.

[15] Zhang, H., and Zhong, G. (2021, March). haomingvince/ECE271B_Group3. GitHub. https://github.com/haomingvince/ECE271B_Group3