Food recommender system on Amazon

CSE 255 Assignment 2

Tao Huang
A53095200
tah012@eng.ucsd.edu

Huan Zhou
A53091001
huz051@eng.ucsd.edu

Kai Zhou
A53101200
kaz040@eng.ucsd.edu

Abstract

This article describes the process of building a recommender system for grocery and gourmet food. Based on people's reviews on amazon.com, we could predict whether the user may or may not like the food by learning the user's taste, the product's features and the correlation between the user and the product. Several models are applied to the predictive task, including Linear Regression, Basic Latent Factor, Bias-SVD and SVD++. Based on the characteristics of our data, we train these models on the different data sets, and evaluate their performance in terms of mean squared error (MSE) on the test set respectively. The results show that linear regression is a suitable model for inexperienced users (those who have reviewed only several items), and Latent Factor Model, especially SVD++ model, is most suitable for experienced users(those who have reviewed many items).

Keywords recommender system, linear regression, latent factor, SVD++

1. Introduction

Recommender systems are software tools and techniques providing suggestions for items to be of use to a user. The suggestions provided are aimed at supporting users in various decision-making processes, such as what items to buy, what music to listen, or what news to read. Food, of course, never fails to be one of the most popular products that people are interested in. Recommending delicious and healthy food to customers can not only help them keep balance in their lives, but also bring about commercial profits for retailers. Established in 1995, Amazon has grown to be one of the world's most popular online shopping websites. It provides a great number of user reviews on various products, including all kinds of foods. Based on that, we will try to find a model which recommends foods for various kinds of users.

2. Dataset Description and Exploration

2.1 Dataset Description

The dataset we use is provided by Julian McAuley on his website[6]. It contains 143.7 million reviews from Amazon website. We will use the pre-categorized Grocery and Gourmet Food dataset which contains 1,297,173 reviews, including 768,450 users and 166,049 different items. Table 1 describes the features of each review:

In addition to the review data, the dataset also provides meta data about the products, as shown in Table 2.

Before we go further, some explorations are performed in order to find the characteristics of the data, e.g. some features might be related to others, the data may contain some patterns. These characteristics would provide us with helpful hints and intuition on building the recommend system. Thus, we first choose some features which might be related to the users' rating, and then visualize their correlations to validate our assumption.

reviewerID	ID of the reviewer
asin	ID of the product
reviewerName	name of the reviewer
helpful	helpfulness rating of the review
reviewText	text of the review
overall	rating of the product
summary	summary of the review
unixReviewTime	time of the review (unix time)
reviewTime	time of the review (raw)

Table 1. Features of reviews

asin	ID of the product
title	name of the product
price	price in US dollars
imUrl	url of the product image
related	related products
salesRank	sales rank information
brand	brand name
categories	list of categories the product belongs to

Table 2. Features of products

2.2 Original Data Exploration

2.2.1 Description Length and Rating

Figure 1 shows the correlation between the user's rating and the length of description of the product. We assumed the products with longer descriptions should have higher ratings. The size of each circle in the figure represents the number of products with that description length. As we can see in the figure, the correlation between product's description length and average rating is not significant, so we will not use this feature in our future prediction models.

2.2.2 Price and Rating

People tend to give high ratings to the product which is affordable or with high quality. So we thought people will give higher rating to the products with lower price. But the results show that people are willing to give higher ratings to expensive food. In Figure 2, the size of each circle represents the number of products with that price. We concluded that the food with higher price have better quality and people are willing to give them higher rates. So we will include price in our prediction models.

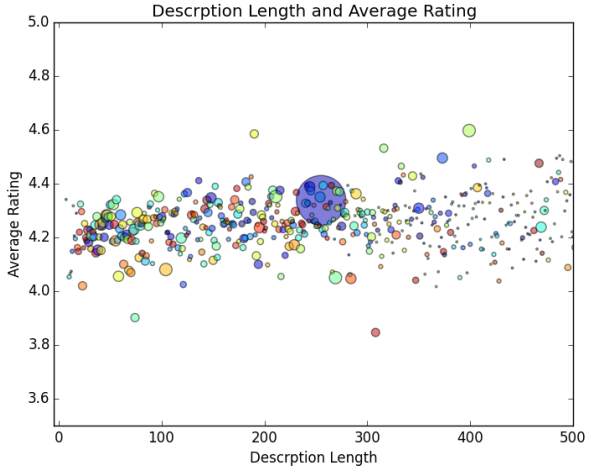

Figure 1. Description Length and Rating

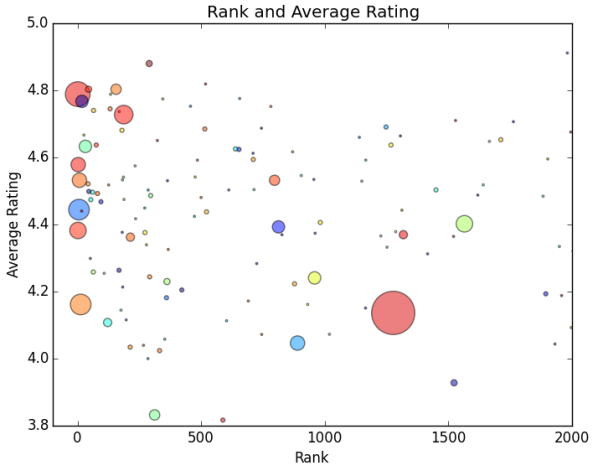

Figure 3. Sales Rank and Rating

Figure 2. Price and Rating

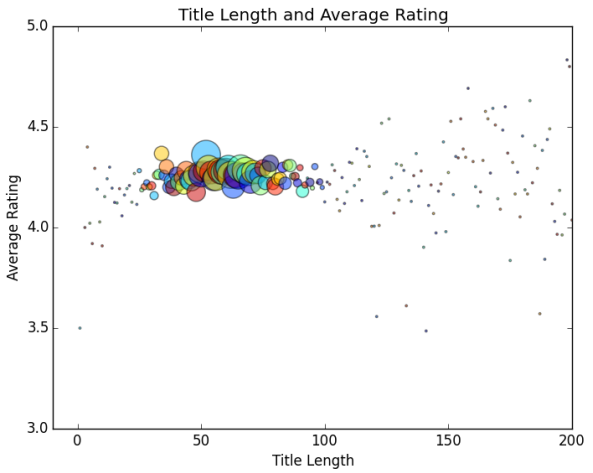

Figure 4. Title Length and Rating

2.2.3 Rank and Rating

We expected the food with higher sales rank will get higher ratings. However, as shown in Figure 3, the sales rank and average rank are not correlated. So we will also exclude this feature.

2.2.4 Title Length and Rating

The figure 4 also shows no relationship between title length and average rating. The size of the circle corresponds to the amount of reviews with that product title length.

2.2.5 Item and Rating

Different products may have very different average rating according to its quality. The figure 5 confirms our expectation. The average rating of different food diverges greatly.

2.2.6 User and Rating

People's standard of rating differs from each other. In figure 6, we eliminate the users with only one reviews to reduce the coincidence. We can tell that although most of users are very generous, some other users are still very strict about rating. We also guessed that

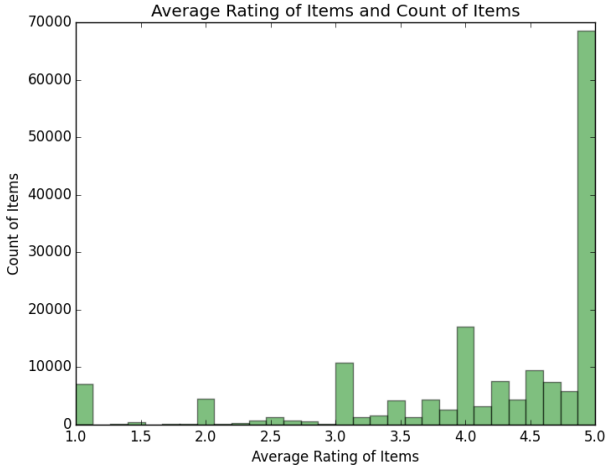

Figure 5. Item and Rating

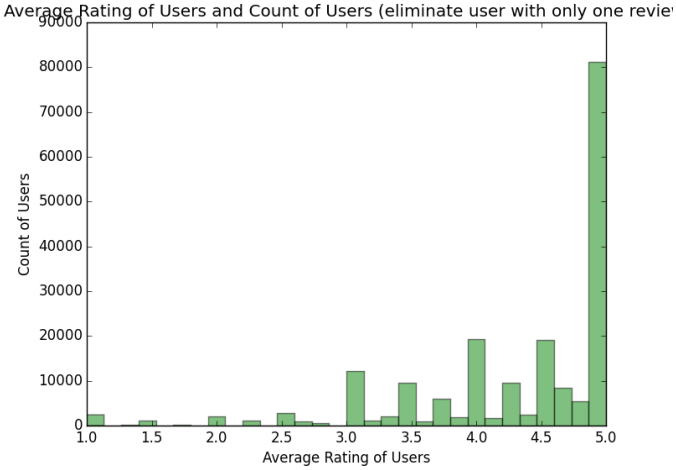

Figure 6. User and Rating

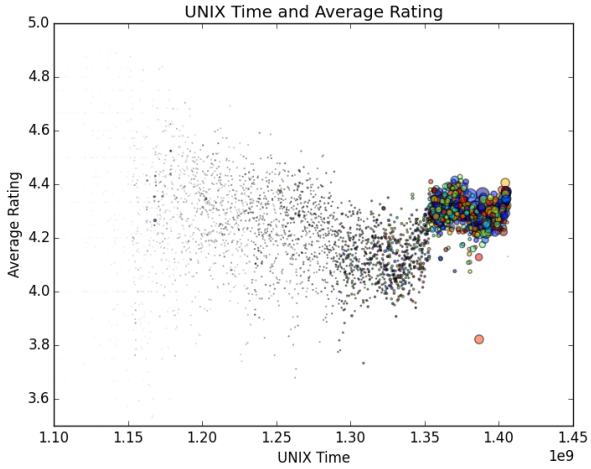

Figure 7. Unix Time and Average Rating

some users tended to give reviews to the products they were not satisfied with.

2.2.7 Unix Time and Average Rating

Next thing we would like to know is whether review time matters in rating. Figure 7 shows the tendency of average rating with time with the size of each circle representing the number of reviews at that time. At the beginning, the number of reviews is small and the average ratings are divergent and decrease. The turning point occurs around year 2012. Both the number of reviews and average rating increase. We don't know the exact reason but we guessed that Amazon might have a new policy that only allows products with high quality appear on its website.

2.2.8 Month and Average Rating over ten years

We also want to know if month matters in the prediction of average rating. Figure 8 shows the trends of average rating along with months in ten years from 2004 to 2013. We expected people will give higher rates around holiday such as Christmas and Thanksgiving but the result did not support our expectation.

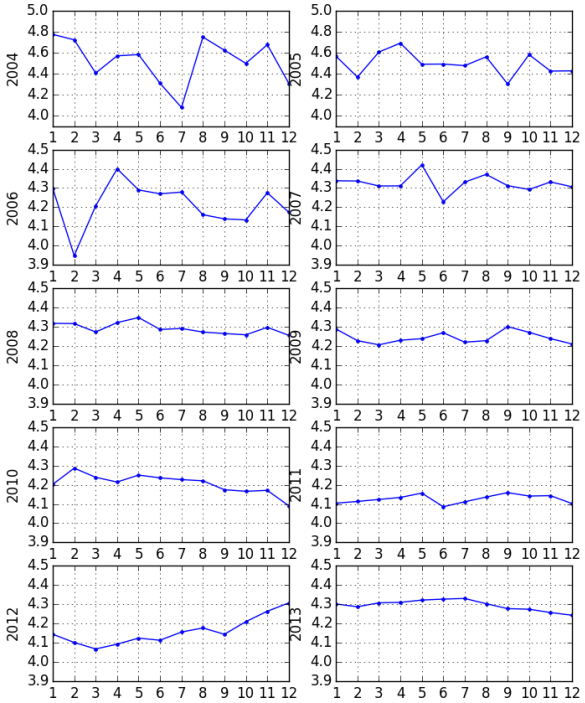

Figure 8. Month and Average Rating over ten years

3. Prediction Task

3.1 Task Description

To recommend a product to a specific user, we need to predict the users evaluation towards this product. Naturally, the rating value of the product in the review data would be a perfect quantified representation of the users evaluation about this product. It seems paradoxical that we are recommending products to users who have bought them before, since the review would only exist after purchase. However, by evaluating the error between the predicted rating and the ground truth, we could generalize the model to recommend products to users who have never bought them by predicting the users rating, given the users id and the products id. Hence, the predictive task is define as:

$$\hat{r}(user, item) = rating_{predict}$$

To evaluate the performance of the model, Mean Squared Error (MSE) is calculated on the test data.

$$MSE = \frac{\sum_t [\hat{r}(user, item) - r(user, item)]^2}{T}$$

3.2 Data Pre-processing

From section 2 we find that the dataset we will use is extremely sparse. For instance, there are in total 1,297,173 data points, in which we find 768,450 distinct users and 166,049 distinct items. Therefore, each user reviews only 1.69 items on average and each item receives 7.8 reviews. The reason why each user has so few reviews on average is that a majority of users review only one item, which largely decreases the average reviews per user. In order to make better use of our data and to improve the accuracy, we divide

the data into two parts: the first part is the subset of data points in which each user has reviewed more than 10 items; the second part is the subset of data points in which each user has reviewed less or equal than 10 items. More specifically, the first part consists of experienced users and the second part consists of inexperienced users. For different types of users, we will use different models to do the predictions and also for different models, we will choose different datasets to train, which will be described in the following sections in detail.

3.3 Baseline

To better evaluate the model, we set up a baseline which is simply predicting the user's average rating. If the user is unseen in the training data, then the global average rating is used as the prediction. Equation 1 defines the baseline, where I_u defines products bought by the user, R_{ui} defines the rating value which user u gives to product i, T is number of training examples.

$$predict_{baseline} = \begin{cases} \frac{\sum_{i \in I_u} R_{ui}}{|I_u|} & \text{if user is seen in training set} \\ \frac{\sum_{u,i} R_{ui}}{T} & \text{otherwise} \end{cases} \quad (1)$$

We calculate MSE of baseline model on the two different parts data sets described in section 3.2. For the dataset in which each user reviews more than 10 items, the baseline achieves $MSE = 1.097$. For the dataset in which each user reviews no more than 10 items, the baselien achieves $MSE = 1.698$.

4. Related Literature

In recommender systems, rating prediction is one of the most important and challenging problems. The Netix Prize has a big impact on this topic and brought many interesting solutions. Simon Funk, one of the participants in this competition, firstly posed a revised SVD method on his website [1], which attracted researchers' attention to the matrix factorization techniques. Different from conventional Singular Value Decomposition, his method used an approximate way to compute the low-rank approximation of the matrix by minimizing the squared error loss.

Since then, many other SVD-based models have been created. A simple one is to add bias terms into the basic matrix factorization model. It is desribed by winner Yehuda Koren , the winner of Netflix Prize in his paper [5]. Koren has also proposed SVD++ model [3], which considers the implicit user feedback, such as the rating actions of users or their browsing history on the website. With the additional implicit information of users, this model improves prediction accuracy to some extent. Furthermore, temporal effects is also an important information in the rating prediction tasks[2, 4]. It follows the idea that user's preference may change over time and item's popularity may also change. For example, a kind of food may become more popular if it is advertied by a pop star.

Some other techniques are also applied to improve accuracy. Julian McAuley[7] combines latent rating dimensions with latent review topics, which makes good use of review information to improve the rating prediction. This model is also suitable for new products and users who could still provide substantial informaition from limited number of review texts. In another paper[8], he modifies latent factor model by introducing user experience as a function of time. Each user can learn individually with a different rate at which their experience progresses.

Based on the state-of-the-art methods above, we try to choose appropriate models for our prediction task. The dataset we use is $Grocery\,and\,Gourmet\,Food$ reviews, which is provided by Julian McAuley on his website[6].

Features	Description
price	The price of the food (if price is given)
alsoBought	The number of products which the users who bought this product also bought
alsoViewed	The number of products which the users who bought this product also viewed
boughtTogether	The number of products that are bought together with this product

Table 3. Features used in Linear Regression Model

5. Model Description

5.1 Linear Regression

The first model we tried is linear regression. The reason is that whether a user would be fond of a specific kind of food should be much related to the food itself. Thus the user's judgement (rating) should be a function of the food's features. This naturally leads to linear regression model.

As discussed in section 2, the lengths of description text and title text and the sales rank of the product have no correlation with the rating. So we do not include these features in the linear regression model. The features we use in are defined in Table 3

Since the data set is not complete, some products' prices are missing. We separate the train data into two parts, one of them has price feature, and the other does not. And we train linear regression model on the two data set respectively. Our linear regression model achieves MSE of 1.553, which is better than the baseline.

5.2 Linear Regression with Users' Bias

The features described in section 4.1 are purely based on the products' features. We completely ignores the users features when predicting the user's rating.

To introduce users' bias into the model, we add three additional features.

- userAvgRating
  The average rating which the user gives to the products he/she has bought before. If the user is not seen in the training set, use the average value of other users. This feature shows the user's rating behavior in the history and should be taken into account when predicting the user's rating on the current product.
- itemAvgRating
  The average rating which the product has received. If the product is not seen in the training set, use the average value of other products. This feature describes how other users evaluate this product. Since people should have similar judgement on delicious food and unpalatable food, the average rating the product has received should be correlated with a new user's rating.
- userReviewCount
  The count of the reviews that the user has made. This features describes the user's expertise in reviewing food. As the user purchases and reviews more food, the user's taste might change.

After training the linear regression model with these three extra features, the MSE has been reduced to 1.587, which is even worse than the model without user bias features. We believe this is due to the fact that the majority of the users in our dataset have only reviewed one or two items, thus we do not have enough information about the user's behavior.

5.3 Latent Factor Model (Basic)

Latent Factor Model was created by Simon Funk, a participant of Netflix Prize in 2006. Since then, it has been widely used in many prediction tasks. Briefly speaking, this model tries to find latent factors for each user and item and it performs by projecting user features and item features into low-dimensional spaces. A simple form of this model can be written as:

$$\hat{r}(u,i) = p_u^T \cdot q_i$$

where q_i is a vector associated with each item i, p_u is also a vector associated with each user u. For a given item i, the elements of q_i measure the extent to which the item possesses those factors; for a given user u, the elements of p_u measure the extent of interest the user has in items that are high on the corresponding factors. Therefore, their dot product $p_u^T \cdot q_i$ denotes the overall interest of the user in characteristics of the item. Besides, we should also add the regularization terms to the optimization problem as:

$$min \sum_{(u,i)\in Train} [r(u,i) - \hat{r}(u,i)]^2 + \lambda(||p_u||^2 + ||q_i||^2)$$

we can use gradient descent method to train the model and get update rules as follows:

$$p_u \leftarrow p_u + \alpha(q_i - \lambda p_u)$$

$$q_i \leftarrow q_i + \alpha(p_u - \lambda q_i)$$

where α is the learning rate.

5.4 SVD with Bias Terms (Bias-SVD)

We have also used Bias-SVD (also called Funk-SVD) model. This model add bias terms β_u and β_u into the simple Latent Factor Model above. These bias terms indicate the observed deviations of user u and item i from the average. For example, the average rating over all foods is 3.5 (namely $\mu = 3.5$). Furthermore, $Cheesecake$ is better than an average food, so it receives 0.4 stars above the average ($\beta_{Cheesecake} = 0.4$). On the other hand, a user named $Stephen$ rate 0.3 stars lower than the average($\beta_{Stephen} = -0.3$). Thus, 0.4 and -0.3 reflects the deviations of $Cheesecake$ and $Stephen$ from the global average 0.35. Therefore, in this model, a rating is predicted by the rule:

$$\hat{r}(u,i) = \mu + \beta_u + \beta_i + p_u^T \cdot q_i$$

To learn the model parameters, we should also minimize the regularized squared error:

$$min \sum_{(u,i)\in Train} [r(u,i) - \hat{r}(u,i)]^2 + \lambda(\beta_u^2 + \beta_i^2 + ||q_i||^2 + ||p_u||^2)$$

Similarly, we use the gradient descent method to get the update rule for each parameter:

$$\beta_u \leftarrow \beta_u + \alpha(e_{ui} - \lambda b_u)$$

$$\beta_i \leftarrow \beta_i + \alpha(e_{ui} - \lambda b_i)$$

$$p_u \leftarrow p_u + \alpha(e_{ui} \cdot q_i - \lambda p_u)$$

$$q_i \leftarrow q_i + \alpha(e_{ui} \cdot p_u - \lambda q_i)$$

where $e_u i = r(u,i) - \hat{r}(u,i)$ and α is the learning rate.

Additionally, we can expect better performance by using different learing rates to user bias, item bias and the factors. However, in our model we just use the same learning, in order to make it more simple and concise.

5.5 SVD with Implicit Feedback (SVD++)

SVD++ model makes use of implicit feedback information, which refers to any kinds of users' click, purchase history information that can assist users' preference. This is especially helpful for those users that provided much more implicit feedback than explicit one. SVD++ method integrates the explicit and implicit user feedback and was shown to offer accuracy superior to SVD. In this model, a second set of item factors is added, relating each item i to a factor vector y_i . Those new item factors are used to characterize users based on the set of items that they rated. The model can be described as:

$$\hat{r}(u,i) = \mu + \beta_u + \beta_i + q_i^T \left(p_u + \frac{1}{\sqrt{|N(u)|}} \sum_{j\in N(u)} y_i \right)$$

where $N(u)$ is a set of items rated by u. If we compare it to the previous one on the SVD, we will find that the only difference is the addition of the $\frac{1}{\sqrt{|N(u)|}} \sum_{j\in N(u)} y_i$ factor. The way to interpret this is that it is including the effect of the $implicit$ information as opposed to p_u that only includes the effect of the explicit one. A user rates an item is in itself an indication of preference. In other words, chances that the user $likes$ an item he/she has rated are higher than for a random not-rated item.

To calculate the parameters for this model, we need to minimize the associated regularized squared error function through gradient descent method. Looping over all known ratings in the training set, we update each parameter by the following rules:

$$\beta_u \leftarrow \beta_u + \alpha(e_{ui} - \lambda_1 b_u)$$

$$\beta_i \leftarrow \beta_i + \alpha(e_{ui} - \lambda_1 b_i)$$

$$p_u \leftarrow p_u + \alpha(e_{ui} \cdot q_i - \lambda_2 p_u)$$

$$q_i \leftarrow q_i + \alpha(e_{ui} \cdot (p_u + \frac{1}{\sqrt{|N(u)|}} \sum_{j\in N(u)} y_i) - \lambda_2 q_i)$$

$$\forall j \in |N_u|, y_j \leftarrow y_j + \alpha(e_{ui} \cdot \frac{1}{\sqrt{|N(u)|}} \cdot q_i - \lambda_2 y_j)$$

where we use different regularization parameters λ_1 and λ_2 to bias terms and factors.

Based on the characteristics of the data, our model only uses the history items that a user has rated as implicit feedback factors. If given more information, such as the items a users has clicked or items a user has searched for, we could add more implicit factors into our model.

6. Results and Conclusion

In section 3 we mentioned that based on the characteristics of our data, we will use different sets of data to train different models. Therefore, there seems to be no sense if we compare the results from different models. However, we can still make comparisons within the same type of model and then conclude how to use these models under different situations.

6.1 Results of Linear Regressing Model

The results of two linear regression models and the baseline model is shown below.

Models	Baseline	Linear Model 1	Linear Model 2
MSE	1.698	1.553	1.587

As shown in the results, including user bias in the features is even worse than the linear model which only use product's features, since most of the users review only one or two items. We conclude that new users are hard to predict, compared to experienced users. Their rating behaviors seem to differ significantly from the average rating.

6.2 Results of Latent Factor Model

Based on the idea of latent factors, we have tried three models to train the data, which are Basic-LFM, Bias-SVD and SVD++ models. The training data we use is a subset of data which consists of experienced users(those who have more than 10 reviews). For parameters in these models, we choose $\alpha = 0.02$ as a general learning rate, $\lambda = 1.0$ as a common regularization parameter and $k = 20$ as the dimension for latent factors. we get the best results of all the models as follows:

Models	Baseline	Basic-LFM	Bias-SVD	SVD++
MSE	1.097	4.522	0.908	0.894

Some results are just within my expectations, while others surprise me a lot. Let's make a detailed analysis of these results.

Firstly, we need to note that the Baseline result in this model is different from linear regression model, because we use different dataset for them.

The first model is a basic latent factor model. It gets the MSE result of 4.522, which is even worse than baseline method. The reason may be that this model simply uses two latent factors to predict the ratings, regardless of either the global average rating or bias effects of each user and each item. Since the rating matrix is very sparse, each element of latent vectors can easily be affected by a single value in the matrix. Therefore, the parameters in this simple model are not trained very well. When I use a denser matrix in this model(for example, a matrix consists of users who review more than 30 items), the MSE drops down to nearly 2.5. So I conclude from this experiment that Latent Factor Model is very suitable to predict ratings for experienced users. The reason lies in that experienced users give us more information about their preferences as well as their implicit feedback.

The second is Bias-SVD model, which adds the bias terms and global average into the basic model. This model should be a good choice for this prediction task and it turns out to be. It improves the baseline method by 17%, which is a satisfying result. Furthermore, SVD++ model adds the implicit information into Bias-SVD model and gets a slight improvement of the result. However, SVD++ model has higher computation complexity than Bias-SVD model, which is the price for a sightly better result.

6.3 Summary

We can have an overview of these models. In our recommender system, we use different models to make predictions for different users. For inexperienced users, they have only reviewed several items, giving us very little information about their preferences. So we use linear regression model to predict their ratings and recommend high rating items to them. For experienced users, we use Bias-SVD model or SVD++ models to make predictions. In this way, we can get a comparatively high accuracy for both types of users.

7. Further Work

Given more time, we can do further improvement on our models in following ways:

- **Choose better parameters for Latent Factor Models**
  In our latent factor models, we have only tried several values of parameters, such as regularization parameter λ, learning rate α, dimension of latent factor k. Choosing a better parameter for these models could improve accuracy to some extent.

- **Add temporal information in Latent Factor Models**
  Another idea to improve latent factor model is to add time information. It is based on the fact that item's popularity may change over time and user's preference or bias may also change. According to our own experience, there is a high chance that this may happen, because our taste is totally different from it was ten years ago! It seems to be a good idea to take time information into account.

- **Using Collaborative Filtering for new inexperienced users**
  Collaborative Filtering is widely used in real-world recommender systems. It firstly calculates the similarity between different items(or users), and then recommends to users those items which are highly similar to what they have purchased. Since inexperienced users purchased only several items, we can just recommend similar items to them. It's an easy and effective model.

- **Combining different models**
  It is possible that different models are suitable to different situations. An intuitive idea is to add different weights to those models and combine them together, which can be described as $\hat{r} = \sum_{k=1}^{K} \alpha_k \hat{r}^{(k)}$. Similarly, we can choose the appropriate weights by minimizing MSE.

References

[1] Netflix update: Try this at home. `http://sifter.org/ simon/journal/20061211.html`.

[2] P. B. Kantor, L. Rokach, F. Ricci, and B. Shapira. *Recommender systems handbook*. Springer, 2011.

[3] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[4] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

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

[6] J. McAuley. Amazon product data. `http://jmcauley.ucsd.edu/data/amazon/`.

[7] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM, 2013.

[8] J. J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*, pages 897–908. International World Wide Web Conferences Steering Committee, 2013.