Product rating prediction based on RentTheRunway clothing data

Ke Wan, Zehui Jiao, Le Yang, Qiwen Zhao

Abstract

As an increased saturation of celebrity worship – social platforms were making people more aware than ever before of socialites and pop-culture stars like Kim Kardashian, and what luxury brands they were wearing. Women increasingly wanted to develop their own personal brands via social media to show off their luxury. To make the bargain cheap and satisfied what they need, Rent the Runway was born. Rent the Runway is a popular e-commerce platform that allows users to rent, subscribe, or buy designer apparel and accessories. In this report, we dive deep into the RentTheRunway dataset and explore the statistics, we try to find out what influence the users' rating on the clothes most and how can we predict the rating based on the characteristics of the user base? We applied sentimental analysis model to predict the rating, and also implemented linear regression, latent factor model, factorization Machine on several different features of the user base. We compared different models and our method over-performed the baseline by 41.4% which is also better than other compared model

Keywords: regression, latent factor model, factorization machine, sentimental analysis

1 DATASET

For this assignment, we use the dataset of renttherunway provided by the company. This dataset provides us with records about the ratings on clothes. Rent the Runway is a popular e-commerce platform that allows users to rent, subscribe, or buy designer apparel and accessories. The customers rent clothes and wrote comments on the rented clothes. Most importantly, they will rate the rented clothed which may influence other's willingness. The dataset also records other information about the renter, which gives us hints the characteristics about the reasons why they tended to give higher or lower ratings. We conduct deep research about the characteristic about this data and do some prediction in this report.

1.1 Overall Review on Data

We have 192544 rows in the dataset. That's to say, we have 192544 comments from users on the clothes. Every comment record has many features, including the figure size of the renter, such as weight, height, body type, and the information about the clothes such as category. More detailed information about each comment in the dataset is displayed in the Table 1.

Table 1: Data formula

name	description
fit	fitness of the rented clothes, with three categories to describe: fit, larger, and small
age	age of the renter
bust_size	the size of the renter's bust
weight	the weight of the renter
height	the height of the renter
size	the size of the clothes
category	the category of the clothes
body_type	which type is the renter's figure
rented_for	the purpose of renting the clothes
review_text	the user's comment on the rented cloth
review_summary	the user's comment summary on the rented cloth
user_id	user ID
item_id	the ID of the rented cloth

Among all these features, intuitively the most valuable features should be the fitness and review text. The fitness represents the feeling of the user and that will directly influence their ratings. And the content in the review text will also directly display their emotion and preferences to the cloth. However, the comments include renter's self figure information, and whether that is a good fit to the clothes feature is another index that needed consideration.

1.2 Dataset analysis

There are 192,462 reviews in the dataset with 105,508 users and 5,850 products. Before we perform predict tasks, we want to explore some useful hints from the dataset, such as the rating, weight, height, body type and category distribution.

1.2.1 Rating Distribution

Intuitively, to perform predict tasks on rating, we want to know the distribution of users' rating and how people tend to rate their rented clothes. Figure 1 shows the result. The histogram tells that most users are tend to rate with Five and very few users would like to give an one rating. This fact indicates that most of the users are content with the clothes they rented. And we can also conclude that if a renter gave a rating below 3, that means the clothes he rent is awful and uncomfortable

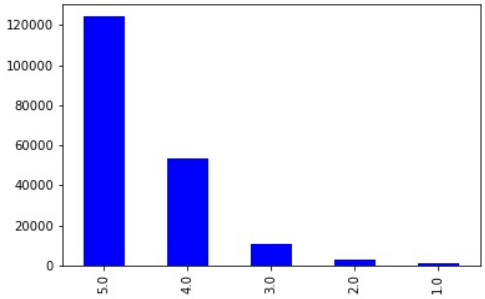

Figure 1: Rating Distribution

1.2.2 Fit vs Rating

In the next step, we'd like to see the fitness distribution across users base. Fitness is one important index to weight the rating. It will directly influence the renters' rating. From the graph, we can find that most clothes are fit for the renters. And that's a reason why most customers will give a five rating.

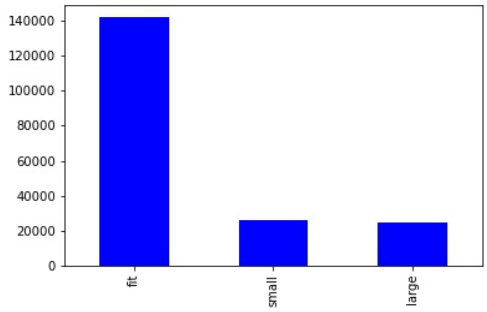

Figure 2: Fitness Distribution

We also want to explore the relationship between the rating and the fitness of the feedback. It is proper to use pie chart to present how the five ratings are distribute under three fitness of the feedback.

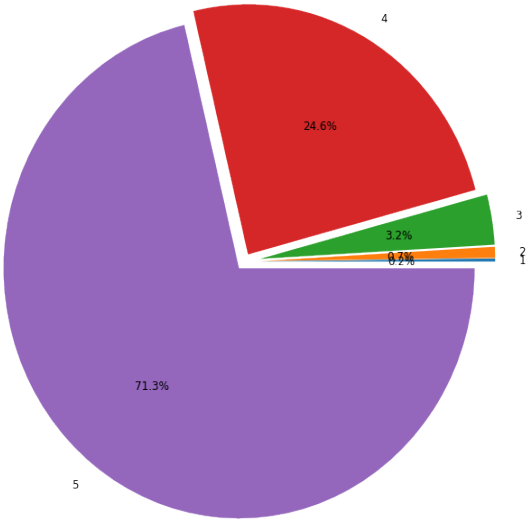

Figure 3: Fitness Distribution

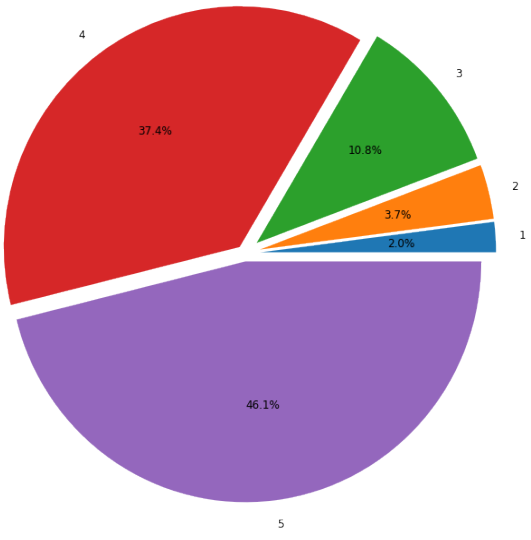

Figure 4: Fitness Distribution

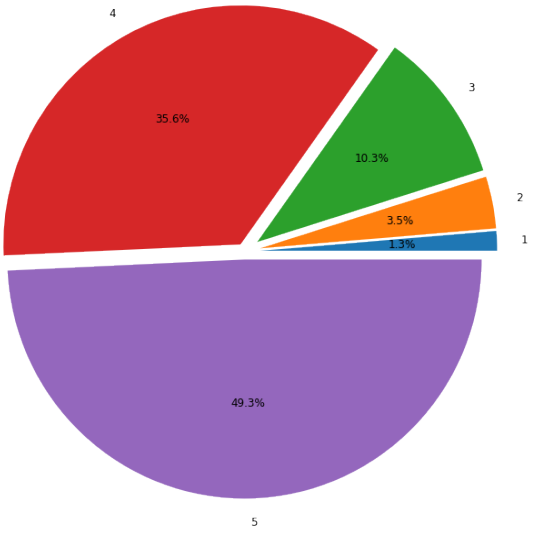

Figure 5: Fitness Distribution

From Figure 3, Figure 4 and Figure 5 we can find that the renters tend to give high ratings to the items when they give 'fit' fitness to the items. In this case, we may think 'fit' can be an important factor that influence the renters' rating.

1.2.3 Age vs Rating

Then, we want to explore the relationship between rating and the user's self, such as age. Is age an import index that influence the users' rating? That's to say, if the renter is older, will they tend to give a higher/lower rating? From Figure 6, we can see the for each rating, the age distribution is almost the same. Therefore, we may think age is not an important factor that will influence the renters' rating.

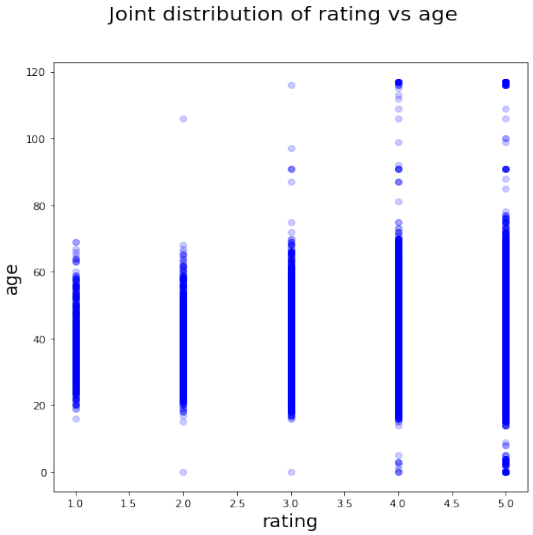

Figure 6: Age vs Rating

1.2.4 Review Date vs Rating

Finally, we want to explore the relationship between review date and rating. That's to say, are goods more likely to be perfect with our development of technology? Thus, a customer is willing to give a high star. From the following figure, we could say that if we have plenty of samples, the rating is higher as year increases.

From the figure, the user who bought goods in 2017 was willing to give a high star than the user in 2013. We can hardly to tell the real reason, but the advanced technology might be an important reason. It might be helpful for our task cause there exists some relationships between year and rating.

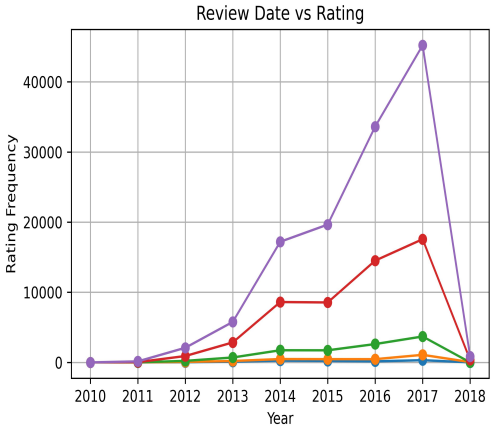

Figure 7: Review Date vs Rating

2 PREDICTIVE TASK

2.1 Task Description

We will identify a predictive task which could be trained on our dataset. We would like to predict the rating of the rent cloth that a renter is tend to give.

$$f(userid, itemid, height, weight, age, review\_text) = rating \tag{1}$$

In the coming sections, we will analyze the features and generate its numerical expression.

2.2 Data Pre-processing

The only ratings in the data set is 2,4,6,8,10, also we check on the original website and find that the user can rating 1,2,3,4,5, so we divide the original rating by 2. To evaluate the predictive task and assess the validity of our model, we divide the training set, validation set and test set according to the ratio of 6:2:2. We use MSE(Mean Square

Error) to evaluate the performance of the model, which is:

$$MSE = \sum (R_e - R_r)^2 \quad (2)$$

Where R_e is the estimate rating and R_r is the rating.

3 MODELS

3.1 Baseline

For the rating prediction task, we will compute averages for each user, or return the global average if we've never seen the user before. The global average is 4.545234115884549, and the Mean-Square-Error(MSE) using the baseline model is 0.6262598959223842.

3.2 Our Selected Model - Sentimental Analysis Model

3.2.1 Model Introduction

In this experiment, we mainly built a sentiment analysis model that estimates ratings. Before this assignment, we spent long time thinking about how to make predictions for users' ratings as precise as possible. We implemented some models such as regression models, latent factor models and so on. But the performance is not satisfactory. Then we made a proposal: maybe it's more effect to make predictions mainly based on the content of users' comments because the comments will contain much information, including the users' attitudes, the users' preferences and so on. Therefore, if we try to do text mining based on users' comments and make predictions, the results will be very satisfactory. We tried and succeeded. The sentimental analysis model performs the best among three other models. The test results of models are shown in table 2. The performance of sentimental analysis model is far greater than any other models. And it beats the baseline greatly. Therefore, we will choose this model.

Table 2: Best MSE under different models

Model Name	Best MSE
Sentimental Analysis Model	0.3674
Latent Factor Model	0.4828
Factorization Machine	0.5487
Regression Model	0.5109
Baseline	0.6263

Sentiment analysis model is a text mining model which is used to make predictions based on sentimental words. Firstly, we extracted "review_text" of users from training set, counted the number of unique words, and counted the time of each word's appearance. Secondly, we ignored capitalization and removed punctuation from the set. Thirdly, we built feature vector by taking the most popular words. After building the feature and getting the labels from training set, we extracted bag-of-word features and trained the model. Finally, we used the model to make predictions on the test set. Initially, we set the number of popular words as 1000 and set the l2 regularization as 1.0. The mean square error (MSE) equals to 0.3766.

3.2.2 Optimizations

Then we tried to make some optimization to improve the performance of this model. The first thing we tried was to try different values of l2 regularization. We set the value as 5.0, 1.0, 0.001, and 0.00001, but the MSE does not have much difference. The MSE for different model is shown in the Table 3.

Table 3: MSE under different models (where number of popular words equals to 1000)

l2 regularization	MSE
5.0	0.3765
1.0	0.3766
0.001	0.3766
0.00001	0.3766

We chose the value of l2 regularization as 5.0 because it performs the best among the other values. And then we tried many values for number of popular words. We set the value as 1000, 5000, and 10000. The MSE for different model is shown in the Table 4. We can see that when the number of popular words is set to 5000, the MSE is the smallest, which equals to 0.3674. At this time, we have optimized the model well and reduced its MSE by about 2.5%.

Table 4: MSE under different models (where value of l2 regularization equals to 5.0)

number of popular words	MSE
1000	0.3765
5000	0.3674
10000	0.3732

3.2.3 Problems and Solutions

When we were doing the experiment, we faced a problem of over fitting. We found that if we set the l2 regularization with a small value (less than 0.00001), the performance of the model on test set will be much poorer compared to the performance on the training set. For instance, when we assign the l2 regularization with 0.000001, the

MSE on test set equals to 0.3832. But the MSE on training set equals to 0.3721. However, if we replaced the value of l2 regularization with a larger one, for example, 1.0, the performance of the model on test set and training set will be almost the same. The implies that if we set the l2 regularization too small, we will get an over fitting model. In order to eliminate over fitting, we have to give a larger value to l2 regularization.

3.2.4 Strength and Weakness

The strength of this model is very obvious. It outperforms the baseline by 41.4%, meaning that text mining is a powerful tool to make predictions to the ratings. Although the performance of sentimental analysis model is very impressive, the weakness of this model is obvious. Compared with other models, sentimental analysis model only takes the comments from users into consideration without including other features such as 'fit', 'age', and so on. Maybe it will have more prediction errors when most of the comments from users are neutral. In this case, the comments will not contain many words that express the users' preferences, and the prediction value of the model will be approximately equal to the average value of ratings of the training set, which means that the model will be a trivial model. Fortunately, the sentimental information in this data set is good enough that it will help train a well performed model.

3.3 Linear Regression Model

3.3.1 Model Introduction

There are other kinds of models that we tried. The first model that we tried is Linear Regression Model. The model is very simple - just extract the features from the training set and make a linear regression, then we will get the model. We can use the model to make predictions. The formula of the model is:

$$y = X\theta \tag{3}$$

where y is the rating vector, and X is the feature matrix.

We tried three kinds of feature vectors for the model. The first feature only contains the length of review text. The second feature only contains fitness feature (fitness feature means the clothes is small, fit or large, which is provided by consumers directly). The third feature contains the properties of users, including age, height, and weight. We trained each of three feature vectors separately. Then we did ablation experiments to determine contributions of the features to the overall.

3.3.2 Selected Features

The first feature contains len(review_text) only. We extracted the review_text from the training set, then we got the length of every review text and constructed feature vectors. Then we used linear regression to train and test the model. The MSE of the model on the test set is 0.5109.

The second feature contains 'fit' feature only. We extracted the 'fit' feature from the training set, then we used one-hot encoding to map the 'fit' string to integers. Then we used linear regression to train and test the model. The MSE of the model on the test set is 0.5109.

The third feature contains age, height, and weight. This is used to analyze whether or not the property of users will have effect to the ratings. We extracted the age, height, and weight data from the training set, then we constructed feature vectors and used linear regression to train and test the model. The MSE of the model on the test set is 0.5116.

Afterwards, we did ablation experiments to determine contributions of the features to the overall. The result is shown in the table 5 below.

Table 5: Ablation Results

features included	**MSE**
feature 2 and 3	0.5119
feature 1 and 3	0.5118
feature 1 and 2	0.5112
all included	0.5103

From table 15, we can find that the length of review text and 'fit' feature matters almost the same in the Linear Regression Model. But the age-weight-height feature matters the least. That means that the properties of users will have less effect on the ratings.

3.3.3 Strengths and Weaknesses

One of the strengths of this model is that it is very simple, and it can be easily trained. We can spend less time training an acceptable model. However, because the model is simple, it cannot predict the ratings precisely. Compared to Sentimental Analysis Model, the performance of Linear Regression Model is quite trivial.

3.4 Latent Factor Model

3.4.1 Model Introduction

In recommender systems, the latent factor models are mainly used to allow the model to discover the latent features between items by itself. The basic idea of latent factor models is that we can assume there exists an unknown low-dimensional

representation of users and items, and the affinity between the users with the items can be modeled accurately[3]. Matrix factorization techniques are are a widely successful class of latent factor models that attempt to find weighted low-rank approximations to the user-item matrix, where weights are used to hold missing entries. Using the latent factor model, the rating prediction $f(u, i)$ can be set as:

$$f(u,i) = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i \tag{4}$$

Where α is initially set to the mean ratings, β_u and β_i are user biases and item biases, γ_u and γ_i are user factors and item factors. In this case, the optimization problem becomes to minimize the following regularized squared error:

$$\sum_{u,i}(\alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i - R_{u,i})^2 + \lambda[\sum_u \beta_u^2 + \sum_i \beta_i^2 + \sum_i ||\gamma_i||_2^2 + \sum_i ||\gamma_u||_2^2] \tag{5}$$

Where λ is the regularization coefficient, and $R_{u,i}$ is the ground truth of the rating.

3.4.2 Hyperparams

Let k denote the dimensions of γ_u and γ_i. We set k from 0 to 4 to do experiments and compare the results.

Let λ denote the regularization coefficient. We set k at 0.001, 0.0001 and 0.00001 to do experiments and compare the results.

To compare the results under different configuration of hyperparams, we use the grid search method to do the experiments. To prevent the over-fitting, in additional using the regularization method, we also set the max iterations to 30.

Here we show the MSE(Mean Square Error) result on the test set under different configuration of hyperparams:

Table 6: MSE under different λ with $k = 0$

λ	**MSE**
0.00001	0.5128
0.0001	0.4828
0.001	0.4960

Table 7: MSE under different λ with $k = 1$

λ	**MSE**
0.00001	0.5131
0.0001	0.4829
0.001	0.4960

Table 8: MSE under different λ with $k = 2$

λ	**MSE**
0.00001	0.5140
0.0001	0.4828
0.001	0.4960

Table 9: MSE under different λ with $k = 3$

λ	**MSE**
0.00001	0.5197
0.0001	0.4829
0.001	0.4960

Table 10: MSE under different λ with $k = 4$

λ	**MSE**
0.00001	0.5152
0.0001	0.4829
0.001	0.4960

During the experiments, we find that the bigger the k is, it is more possible for the model to be over-fitting. When the k is set to 2, and the λ is set to 0.0001, the result is best. Under this configuration, we can get the MSE on the train set for 0.3952, the MSE on the validation set for 0.4835, and the MSE on the test set for 0.4828.

3.4.3 Strengths and Weaknesses

Latent factor model can Reveal hidden features that explain why the corresponding prediction scores are given. Such features may not be directly describable in words. Latent factor model is very data-hungry, therefore, if you feed it with a huge amount of data, it can fit any features. However, in the experiments we find that latent factor model is easy to be over-fitting and sometimes you do not know which dimensions of the factors is appropriate and have to try many potential dimensions. Moreover, if we meet some user or item we have not met before, the latent factor model may become useless.

3.5 Factorization Machine

3.5.1 Model Introduction

In real world, features extracted from the dataset are not independent. So linear regression may not work well. Thus, we decide to apply factorization machines to this dataset. Firstly, we will explain the theory of factorization machine. Factorization Machines (FM) are generic supervised learning models that map arbitrary real-valued features into a low-dimensional latent factor space and can be applied naturally to a wide variety of prediction tasks including regression, classification, and ranking. FMs can estimate model parameters accurately under very sparse data and train with linear complexity, allowing them to scale to very large data sets — these characteristics make FMs an ideal fit for real-world recommendation problems[1].
The FM model equation is composed of n-way

interactions between features. A second-order model includes weights for each base feature as well as interaction terms for each pairwise feature combination.

$$f(x) = w_0 + \sum_{p=1}^{P} w_p x_p + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} w_{p,q} x_p x_q \quad (6)$$

This model formulation may look familiar — it's simply a quadratic linear regression. However, unlike polynomial linear models which estimate each interaction term separately, FMs instead use factorized interaction parameters: feature interaction weights are represented as the inner product of the two features' latent factor space embeddings[2]:

$$f(x) = w_0 + \sum_{p=1}^{P} w_p x_p + \sum_{p=1}^{P-1} \sum_{q=p+1}^{P} \langle v_p, v_q \rangle x_p x_q \quad (7)$$

This greatly decreases the number of parameters to estimate while at the same time facilitating more accurate estimation by breaking the strict independence criteria between interaction terms.

As shown in the table, we could say that the best model is that we only select user_id and item_id as the features. In the research process, I found the result was bad when we chose all features. So I tried every combination of features. The feature other_items may be more important than others but still not useful.
The data distribution might be the reason. As stated before, the data distribution is not even. Most user is more likely to give a high rating regardless of clothes.
Besides, FM works better when features are sparse. But in this dataset, features are dense. So if we don't select extra features, the result is the best.

3.5.3 Hyperparams

init_stdev This parameter is the initialization of stdev. The result is shown as follows:

Table 11: MSE of predicting rating about init_stdev

init_stdev	MSE
0.01	0.5481
0.1	0.5505
0.2	0.5490
0.3	0.5487
0.4	0.5471
0.5	0.5477

rank This parameter is the rank of the factorization used for the second order interactions. The result is shown as follows:

3.5.2 Selected Features

We selected the following features and transformed them:

- user_id
- item_id
- review_date
- category
- fit
- other_items

In review date, we just chose the review year and transformed it. Other items mean all items the user has commented. In other features, we just transformed them like one-hot encoding. The result comparison is shown as follows.

Table 12: MSE of predicting rating about rank

rank	MSE
3	0.7385
4	0.6585
5	0.6159
6	0.5848
7	0.5711
8	0.5915

l2_reg_w This parameter is L2 penalty weight for linear coefficients. The result is shown as follows:

Table 13: MSE of predicting rating about l2_reg_w

l2_reg_w	MSE
0.0001	0.7138
0.001	0.6997
0.01	0.6364
0.1	0.5615
0.5	0.5642

l2_reg_V This parameter is L2 penalty weight for pairwise coefficients. The result is shown as follows:

Table 14: MSE of predicting rating about l2_reg_V

l2_reg_V	MSE
0.0001	1.0743
0.001	0.8782
0.01	0.6099
0.1	0.5505
0.5	0.5892

summary So we set parameters as following:

Table 15: parameters setting

parameter	value
init_stdev	0.01
rank	7
l2_reg_w	0.1
l2_reg_V	0.1

4 LITERATURE REVIEW

The RentTheRunway dataset was collected by Julian McAuley. In their previous research, they focused on how to predict the fitness for the cloth through customer's feedback. They proposed a predictive framework to tackle the product fit problem. The framework will capture the semantics information hidden in the feedback and utilize a metric learning technique to resolve label imbalance issues. To fit semantics, they adopt a lantent factor formulation to decompose semantics of fit feedback. In their paper, extensive experiments was conducted, they compare the performance of five methods(1-LV-LR, K-LV-LR, K-LF-LR, K-LV-ML, K-LF-ML). The proposed method K-LV-ML and K-LF-ML used metric learning approach to produce the final classification. The experiments show their capacity of capturing fit sentiments of "true" sizes.

The task of the paper is to predict the fitness of clothes which is different from our rating prediction task. However, the idea to implement semantics analysis and metric learning is enlightened. When it comes to rating predicting task, the basic method is the offline experiment method, using average ratings of items or user, the neighborhood-similarity-based algorithms using Pearson coefficient, the implicit semantic model and matrix factorization model, SVD and Funk-SVD, lantent factor model with the influence of neighborhood. However, in this task, one important feature is the review text from the feedback. The semantics information in review text can display the emotional preference to rate high or rate low.

Therefore we research on another paper which also performs a rating prediction task and utilize sentimental analysis. Song, Gao and, etc proposed a text-driven latent factor model for rating prediction, and validate the advantage of their approach over state-of-the-art baseline methods.[4]

The dataset they uses is IMDB, Yelp 2013 and Yelp 2014. These datasets mainly include reviews from audience and customers and real rates that was given. These datasets are quite similar to what we are working on.

Review rating prediction is a fundamental problem in the field of sentiment analysis and opinion mining. The problems are often solved in two ways, the recommender system based approach typically adopts matrix factorization techniques. And the sentimental analysis based approach has largely regarded the problem as a multi-class classification task focusing on text content. The paper combined the two approaches with a text-driven latent factor model for review rating prediction. They modeled user-specific and product-specific LFM components separately and then jointly optimize the two components in a unified framework. The excellent part is that for both the components, there is inner parameters to constrain the sentiment-consistency and text-consistency. By adding the two kinds of consistency parameter which employs sentimental analysis techniques into the euation of LFM, the authors enable the LFM to be text-driven.

Our idea in this report to solve the problem is very similar to the paper's work. We use Latent Factor Model and factorization machine to predict from the angle of recommender system, the feature we used in the model considered the characteristics of users and clothes, besides we employ sentimental analysis to contract semantics features that may influence the renters' rating. According to the experiment results, the model combined with sentimental analysis as an assistant feature beat all other model and baseline. This similarity shows that combined recommender model together with nature language processing tech like sentimental analysis as an asistance will enable the predicting model higher efficiency and accuracy to do the task.

We do not take the cold start situation into our consideration. In the paper we review, they proposed a Pairwise Rating Comparison Strategy to solve cold start problems, the basic idea is to minimize the difference between actual differ-

3.5.4 Strengths and Weaknesses

Factorization machines can estimate interactions even in these settings well because they break the independence of the interaction parameters by factorizing them. In general, this means that the data for one interaction helps also to estimate the parameters for related interactions.
But the model may not work well if the features are dense. And the search progress is time-consuming when the model has higher order. This is why we seldom use a model with order greater than 2.

ence and expected difference via rating comparisons to calibrate the factors of users/products. The strategy is very enlightened in the real-world predicting problems.

5 RESULTS AND CONCLUSIONS

Table 16: MSE under different models

Model Name	MSE
Sentimental Analysis Model	0.3674
Latent Factor Model	0.4828
Factorization Machine	0.5487
Regression Model	0.5109
Baseline	0.6263

From the Table 16 we can find that our sentimental analysis model is better than the other 4 models. The result is so significant that our model outperformed the baseline model by 41.4%, which directly returns the average rating. In our sentimental analysis model, we built feature vector by taking the most popular words. Then we build the feature and getting the labels from train set, we extracted bag-of-word features and train the model. Our final configuration is that we set the value of l2 regularization at 5.0 and the number of popular words at 5000. After discussion, we think the reason why the performance of our model outperformed other model may be that using text mining technology, the review text of the renter can better represent how the renter feel about the item. We see that the latent factor is also good but worse than the sentimental analysis model, because it can be easily over-fitting and it is data hungry, which means it needs much more data to train. As for factorization machine, although it can consider the relationship between features, it still doesn't work well. The possible reason is that this model works better when data is sparse.

At last, we want to thank the professor and all TAs. This is the first web mining and recommendation systems we have ever take, and we believe this course can enable us to proceed further in this field.

References

[1] https://towardsdatascience.com/factorization-machines-for-item-recommendation-with-implicit-feedback-data-5655a7c749db

[2] Rendle, Steffen. "Factorization machines." 2010 IEEE International conference on data mining. IEEE, 2010.

[3] Sammut, C., and G. I. Webb. "Latent Factor Models and Matrix Factorizations." (2010): 571-571.

[4] Song K, Gao W, FENG S F, et al. Recommendation vs sentiment analysis: A text-driven latent factor model for rating prediction with cold-start awareness[J]. 2017.