Data C200 Graduate Project

Emerging Research And Technology Recommender System

Tian Zhu,
Yifan Wang

Contents

1. Introduction

Background

In the real business world, by doing reviews analysis, we can learn the customer decision-making process before purchase, as this information often is informative about what products they are satisfied with and what they dislike .

Relying on reviews, we can build recommendation systems to foresee what items the users will like and provide personalized services to customers so that we can improve customer satisfaction. With this being said, most consumers are unlikely to support goods that engage in some sort of review censorship. The key to a thorough review would be analyzing uncensored, authentic reviews that directly from real customers. This, however, is extremely challenging as there are millions of product reviews per day. It is practically impossible for retailers to go through each customer review manually without compromising other benefits over time.

Recommendation system is a specific type of information filtering method that seeks to present information items (such as videos, songs, websites, news) that are likely to be of interest to users. In today's world, it's a key component of e-commerce and IT sector success, and it's becoming more and more widespread. Basically, recommendation systems compile a user's profile based on their past records and compare it to certain reference characteristics in order to predict the rating they would give to something they have not yet evaluated.

According to the characteristics of the reference material, a recommendation system can be based either on a content-based approach or collaborative filtering approach, or both. As their names suggest, content-based approaches are based on matching user profiles with specific characteristics of items, while collaborative filtering approaches are based on filters created based on collaborative efforts or similarity between items.

Abstract

This project focuses on predicting the ratings that a list of users will give to the items based on some similar users. We try to use algorithms to find out some items that are liked by other similar users. Our goal is to build a recommender system that can give users some suggestions based on the likes and dislikes of other similar users.

In this project, we are facing the following problems and have figured out:

1. How can we find other users who have similar preferences?

2. How can we predict the rating that a user will give to an item based on other other users who have similar preferences?
3. How can we measure the performance of our models?

Data Description

The dataset used for this project is provided by an academic staff from UC San Diego, which is an updated version of the Amazon review dataset released in 2014. The dataset is divided into categories like Amazon Fashion, All beauty, Appliances, etc.. Each category has two smaller datasets, Five-score and Ratings. “5-score” represents the data that have been reduced to extract 5-score and each of the remaining users and items have five reviews each. “Ratings only” are the datasets that exclude metadata or reviews. In this project, we choose the two datasets (5-score dataset and ratings only dataset) of All beauty and merge them on the ID of the product to do deeper analysis. In this updated dataset, there are 11 variables and the meaning of each variable is explained by UCSD and summarized in the following list:

Field	Data Type	Description	Sample
reviewerID	string	ID of the reviewer	A3SCYP3MYJEY9S
asin	string	ID of the product	B018WCT01C
reviewerName	string	name of the reviewer	Mariee
vote	int	helpful votes of the review	11
style	string	a dictionary of the product metadata	{'Color:': ' Black'}
reviewText	string	text of the review	Nice!
overall	float	rating of the product	5.0
summary	string	summary of the review	Five Stars
unixReviewTime	string	time of the review (unix time)	1457136000
reviewTime	string	time of the review (raw)	02 19, 2015
image	string	images that users post after they have received the product	[https://images-na.ssl-images-amazon.com/image...

verified	boolean	whether the reviewers bought the products on Amazon platform	True

This report involves K-Nearest Neighbors Algorithms. For this purpose, the original data set is split into a training and a test dataset. The training dataset randomly contains 80% observations and the test dataset randomly contains 20% observations from the original dataset.

All the analyses have been performed in the Python environment. The analysis processing and codes are provided in the file *DS200_FP_tian_yifan.pdf.*

2. Exploratory Data Analysis

Before exploratory analysis, we check whether the data is in the proper format and do the following changes. First, we convert reviewTime to a standardized timestamp format. Then, we remove unnecessary variables and missing data in some columns. Among these variables. I choose overall as the response and other variables as predictors, since the goal of this project is to predict users' preferences.

First, we want to view the distributions of rating and verifying status. We find about 13.4% of the reviews haven't been verified. Some of them might be detected as fake reviews by amazon; the others may have haven't be processed by amazon review system.The distribution of ratings in verified and unverified reviews are similar.

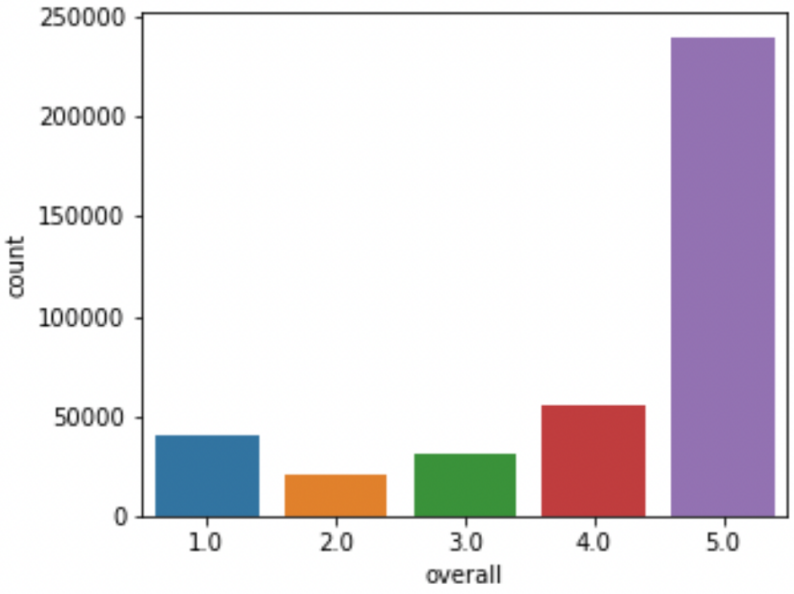

figure 1: Histogram of distributions of overall fitting

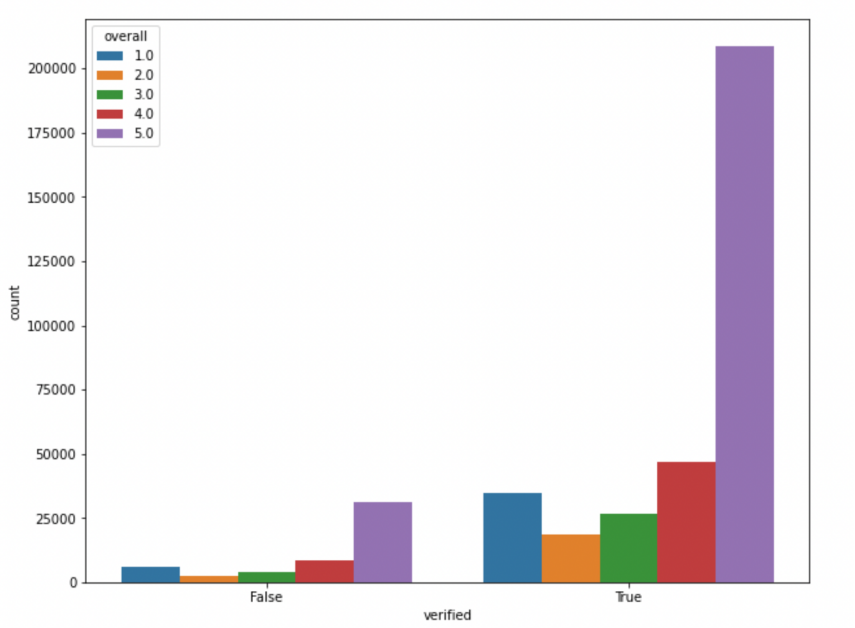

figure 2: Histogram of distributions of overall fitting by verification status

Then, we find some interesting points in the reviews. From the product perspective, the product B000FOI48G has been reviewed by 8671 reviewers. It is a Waterpik dental water jet.The top 100 popular products in the beauty department have been reviewed by 392 or more unique reviewers. Furthermore, 15673 of the products do not have a specific brand. Among these have a specified brand, 252 of the products were from VAGA, and 113 of the products were from L'Oreal Paris. The most reviewed product in the beauty department, surprisingly, is Waterpik Ultra Water Flosser (asin B000FOI48G).

figure 3: The most reviewed product: Waterpik Ultra Water Flosser

From the reviewer perspective , there're 323687 unique reviewers in our dataset. The reviewer A2GJX2KCUSR0EI reviewed 27 products, which was the most across all reviewers. Besides,

the top 100 reviewers in the beauty department reviewed 8 or more products. Data on the level of reviewers are much sparser than data on the product level. The following histogram showed the top 15 reviewer and product also demonstrate the difference.

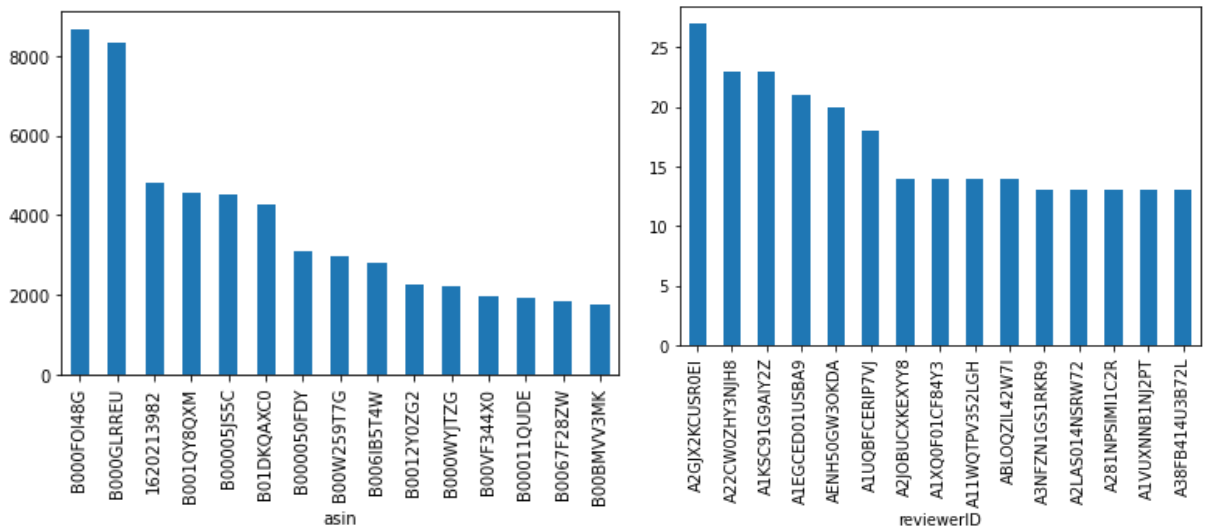

figure 4: The top 15 reviewed products (left) and reviewer (left) and the number of reviews

Finally, we do sentiment analysis on beauty reviews on Amazon. I set overalls larger than 3 as positive reviews and overalls smaller or equal to 3 as negative reviews. Then, I classify positive and negative reviews to 1 and 0. To find the most frequent words in the review texts, I remove the punctuation and stopwords from each text so we can analyze the words. Then I make a wordcloud (figure 5) and present the top 25 frequent words in positive reviews (figure 6) and negative reviews (figure 7) respectively.

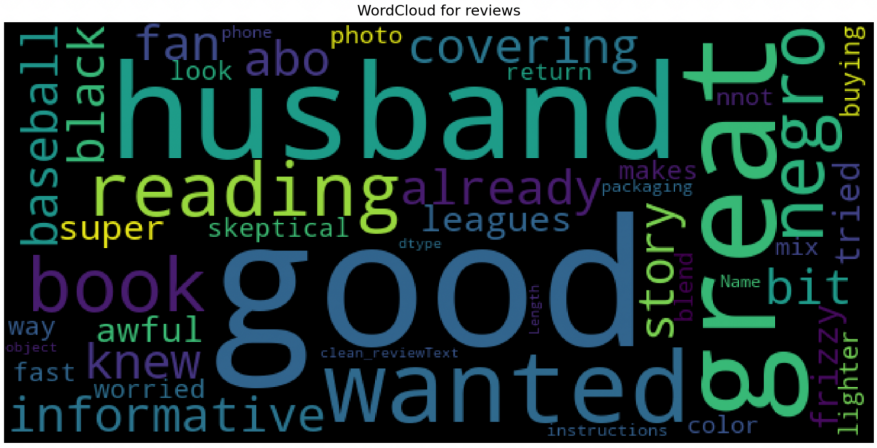

figure 5. WordCloud for reviews

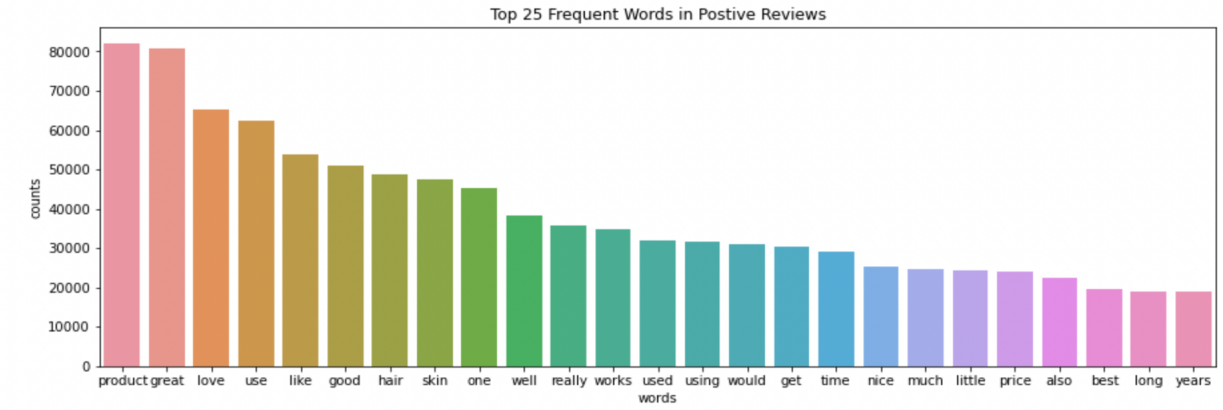

figure 6. Top 25 Frequent Words in Positive Reviews

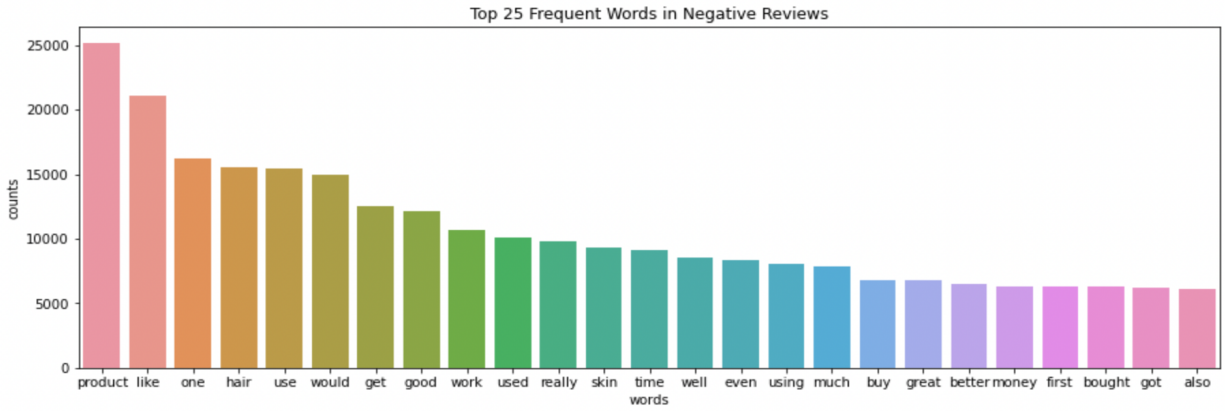

figure 7. Top 25 Frequent Words in Negative Reviews

3. Methodology

3.1 KNN model based on review text

The first model is based on the K-nearest neighbor (KNN) algorithm. The basic idea of the product-based KNN algorithm is: to predict the popularity of the products based on consumers' review texts. We can find some similar products with Y by predicting the ratings based on customer reviews. KNN will find the nearest K neighbors of each product under a similarity function, and use the weighted rating as the prediction of ratings of other products related to product Y. We use CountVectorizer to select the top 400 frequent words in the review texts and find the five neighbors of each product. Then we classify the ratings larger than or equal to 3.5 as 1 (positive reviews), and the ratings less than 3.5 as 0 (negative reviews).

In order to minimize the bias of different users and make the results more objective, we select products which have more than 5 reviews.

First, when we use the KNN algorithm, we can find a group of similar products. For example: Based on reviews, we can get the average rating of product B01E54U1RI is 4.25531914893617. The first similar product is B01E21CEXS average rating is 3.0625. The second similar product is B001AMRQ2W average rating is 4.073394495412844.

Second, we use the model to predict the ratings of each product. In order to improve the accuracy we use 5-fold cross-validation.

Hyperparameter Tuning

We performed a GridSearchCV with 5-fold cross-validation and tried different numbers of n_neighbors. For our dataset, the KNN algorithm performed best when the best n_neighbors value is 21.

Performance Measurement

- Accuracy is 0.7182320441988951
- TPR (True Positive Rate) or Recall: 0.9766536964980544
- TNR (True Negative Rate): 0.08571428571428572
- FPR (False Positive Rate): 0.9142857142857143
- FNR (False negative Rate): 0.023346303501945526
- MSE: 0.281767955801105

The accuracy for text data is around 0.718.

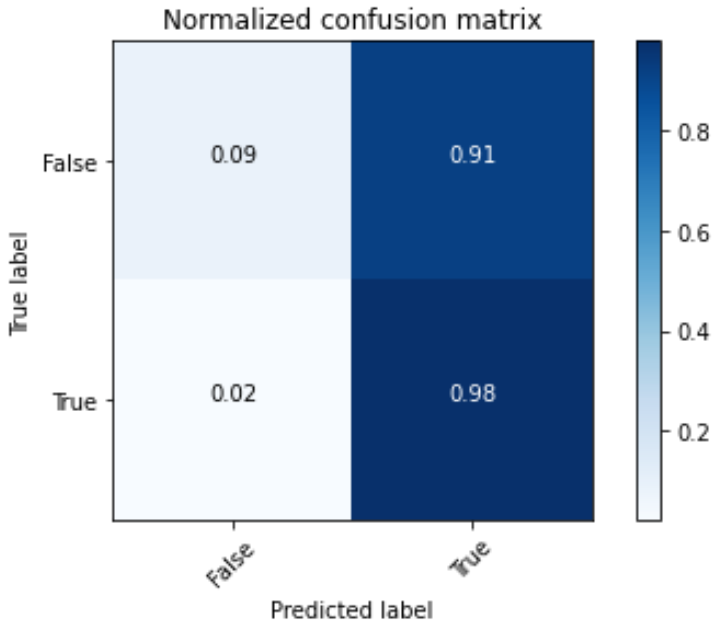

figure 8. Confusion matrix

We find the model performance is not bad. Also, in the real business world, customers' preferences tend to be unique and diverse. To improve our recommendation system and give customized recommendations to users, we decide to build the model and do predictions based on users preferences. More details are provided in the next section.

3.2 Collaborative Filtering Approach

The second model is also based on the KNN algorithm. The basic idea of the item-based KNN algorithm is: to predict the rating of user X on product Y, we can find the ratings given to that product Y by the other users (X1, X2, X3, etc.) and the other products they reviewed similarly. Those products are considered similar products. KNN will find the nearest K neighbors of each product under a similarity function, and use the weighted rating as the prediction of rating of user X on product Y.

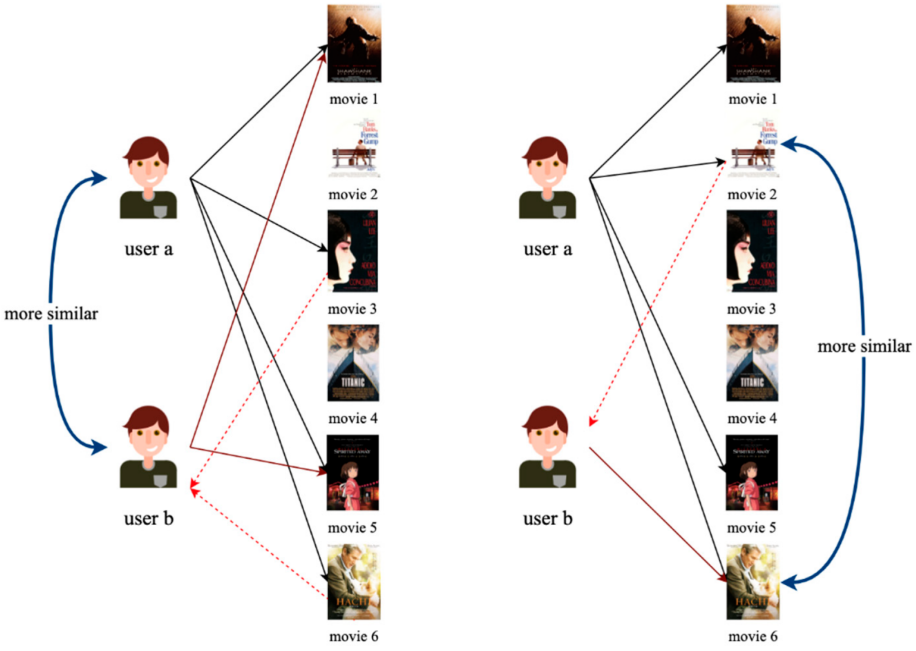

figure 9. User based and Item based filtering

As we have a large dataset, we considered ratings provided by users who have rated more than 2 products, and the products that have more than 2 reviews. The reasons are: Firstly, without enough reviews, we may not be able to understand the users' taste and the prediction may not be accurate. We may not be able to detect similarity between products or reviewers if a product has only received one rating or a reviewer has only reviewed one product. Secondly, memory consideration: collaborative filtering methods could become computationally complex with the local machines on a large dataset, and my computer was incapable of handling the entire dataset. The subset of data has 26738 rows, and each row represents a review made by a user.

Our dataset was randomly split into the train (80%) and test (20%) dataset to better understand the performance of our model. Surprise[1] library is used for modeling.

Hyperparameter Tuning

The parameters we adjusted to achieve optimal performance of the algorithm includes: `k`, `name`, `min_support` and `user_based`.

To identify the 'most similar' products/reviewers, we have a number `k`. As mentioned before, it determines the number of similar products/reviewers used in the prediction; 20, 40 and 60 are considered. To decide which users are 'similar', we define a similarity function. `name` means the similarity function used to compute similarity; msd and cosine functions are compared.

`min_support` is the minimum number of common items needed between users to consider them for similarity; 3, 4, and 5 are compared. The boolean argument `user_based` specifies whether a user-based approach will be used or not, and both are tested.

We performed a GridSearchCV with 5-fold cross-validation. All the possible combinations of parameters above are fitted and evaluated using RMSE.

For our dataset, the KNN algorithm performed best when using the item-based approach and msd as the similarity function, using 20 nearest neighbors to predict the value with a minimum support of 5.

Performance Measurement

With the optimal parameters, the RMSE of the training data with cross validation is 0.6822 and the RMSE of the test data is 0.6764.

We use 3.5 as a cutoff of positive and negative reviews (1-positive, 0-negative), and categorized the true rating and predicted rating to measure the performance. The accuracy of the model is 0.9301. The precision of the model in test data is 0.9434, and the recall is 0.9783, which are fairly high. The confusion matrix of the test model with and without normalization are listed below. Our model performed well in predicting the positive reviews but poorer in the negative reviews.

For those predictions deviated from true ratings, a significant feature is that our model fails to use their neighbors to predict the rating. In this case, our KNN with means model will utilize the average rating of the product, or the average rating of all products (4.476) in the beauty department to predict the rating. This results in some false positives.

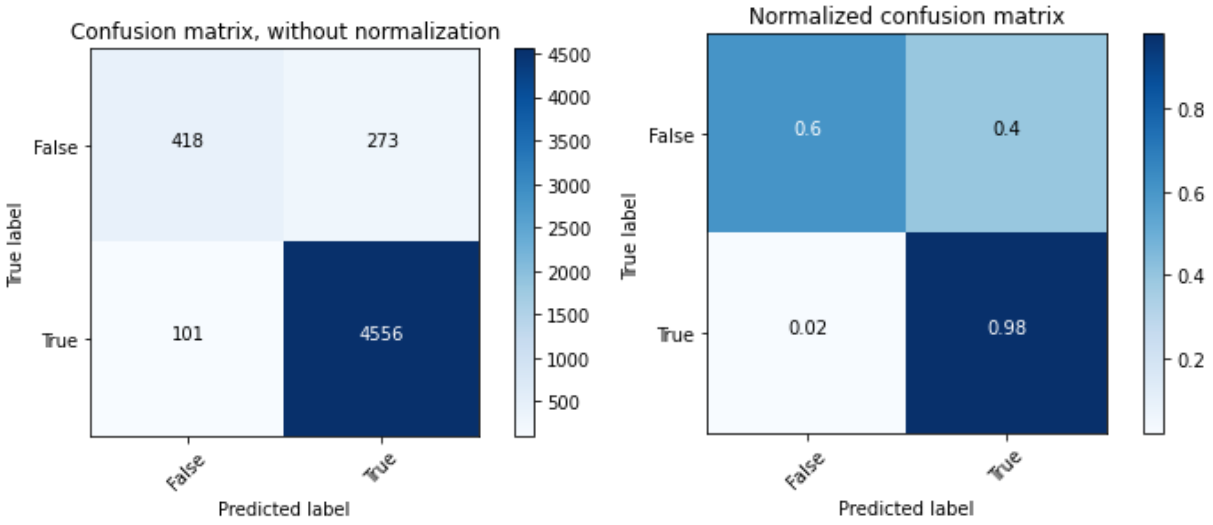

figure 10. Confusion matrix

When we look into specific measures, the predicted ratings of some observations in test dataset are similar to the actual ratings, which indicates that our model could be a valid model as recommendation system: by predicting the ratings of a reviewer on all of the products in the beauty department, we are able to rank the ratings and present the product in accordance with the rank (higher ratings) to ensure that our customer would be satisfied with the products recommended.

For those predictions deviated from true ratings, a significant feature is that our model fails to use their neighbors to predict the rating. In this case, our KNN with means model will utilize the average rating of the product, or the average rating of all products (4.476) in the beauty department to predict the rating. This results in some false positives.

The results of the hyperparameter tuning shows that the performance of user-based algorithms is always poorer than the item-based KNN algorithm with the same other parameters (RMSE: user-based ~0.73, item-based ~0.68). Considering the nature of our review data (and also the EDA part), the reason should lie in that the user-based data appears to be more sparse. [2] For example, the top 100 reviewers in the beauty department reviewed only 8 or more products while the top 100 popular products in the beauty department have been reviewed by 392 or more unique reviewers. This makes it easier to find similar products than similar users. A user-based KNN also presents computational challenges since there are typically much more users/reviewers (323687 in the original dataset) than products (32571 in the original dataset).

4. Discussion and Conclusion

Strength

Both of the two approaches use the K-Nearest Neighbors Algorithm. The KNN algorithm is a non-parametric algorithm, which means it does not require certain assumptions about the distribution of our data for implementation. KNN is a memory-based approach [4]. New training data is immediately incorporated into the KNN classifier. Thus, real-time changes in the input can be handled quickly by the algorithm. This allows our algorithm to apply to new reviews and new ratings on Amazon so that customers can always get updated recommendations.

Limitation and next step

This recommender system can be applied in a variety of areas, such as search engines, online shopping platforms, etc. However, there are also some limitations:

1. The two models of this recommendation system are based on the review text and ratings, so we may fail to recommend some new items until customers review or rate them. One of the possible solutions is to utilize the attributes and descriptions of the products to find similarity on a content-based approach.
2. In the second model, we used a subset of the data and only reviewers with more than 2 reviews were included. We may fail to recommend products to new customers as we could not compare the similarity and predict the customer's preference. An alternative method could be applying a popularity ranking model and recommend the most popular products when we have no or limited information about the customer.

Conclusion

The recommender system can predict the preference a user would give to an item and it will be very helpful in the e-commerce platforms. The recommendation system achieved a low RMSE and high accuracy in prediction, which showed that our model is able to rank the ratings and present the product in accordance with the rank (higher ratings), which ensures that our customer would be satisfied with the product recommendations.

Giving personalized suggestions can improve customer satisfaction and loyalty which will become an outbreak of some retailers. This machine learning tool could help a lot of companies like Amazon, Netflix, Google, etc to filter through massive customer information and get customers' perferences.

References

[1] Hug, N., (2020). Surprise: A Python library for recommender systems. Journal of Open Source Software, 5(52), 2174,
https://doi.org/10.21105/joss.02174

[2] G. Linden, B. Smith and J. York, (2003). Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing, vol. 7, no. 1, pp. 76-80, doi: 10.1109/MIC.2003.1167344.

[3] Allibhai,E., Towards Data Science,
https://towardsdatascience.com/building-a-k-nearest-neighbors-k-nn-model-with-scikit-learn-51209555453a

[4] Grčar, M., Fortuna, B., Mladenič, D., & Grobelnik, M. (2006). kNN versus SVM in the collaborative filtering framework. In Data Science and Classification (pp. 251-260). Springer, Berlin, Heidelberg.