

**MIE 465 Analytics in Action**

**Final Report**

**Personalised Restaurant Recommendation System for Yelp**

**April 12th, 2019**

**Team 5**

| **Team Members** | **UTORid** | **Student Number** |
| --- | --- | --- |
| Fan Jia | jiafan2 | 1001297198 |
| Huiqing Lu | luhuiqin | 1001122858 |
| Jiayue Niu | niujiayu | 1001122894 |
| Xiarui Xie | xiexiaru | 999587022 |

## 

## **1.0 Introduction**

**1.1 Problem Statement and Motivation**

In modern days, eating out with family or friends has become one of the most common entertaining and social activities [1]. This trend raises a frequently asked question “What to eat for dinner?”. To answer this question, many people turn to Yelp for help as it collects a large number of restaurant reviews including ratings, photos, and comments [2]. However, due to the enormous number of restaurants on the system, it is difficult and inefficient for users to discover restaurants they would like.

To solve this problem, we decided to build a personalized restaurant recommendation system to help Yelp users quickly target restaurants they might like. A recommendation system was developed based on the assumption that the more a user likes a restaurant, the higher the rating would be. Utilizing restaurant information and historical reviews, we tried to predict ratings for each user on restaurants that he/she has never reviewed before. Then, five restaurants with the highest predicted ratings would be recommended to each user.

**1.2 Related Work**

Recommendation system is a popular topic in the machine learning research area. Recommendation systems filter out the most important information from a massive database based on the target user’s interest, preference, and past behaviours [3]. A good recommendation system can greatly benefit the service provider on increasing amount of users and sales. Additionally, it can help users save time on searching and improve the quality of decision making. Currently, the three most commonly used methods for building recommendation systems are collaborative filtering (CF), content-based filtering (CBF), and hybrid filtering (combination of the previous two) [4].

Collaborative filtering predicts from historical ratings. Models assume that a user would give similar ratings to similar items and a given item would be given similar ratings by similar users [5]. This approach has some major limitations such as cold-start, sparsity and scalability problems mainly due to data density issues [6]. Content-based filtering makes predictions by matching item features to user preferences. In this case, restaurant and user information are taken into account when building models. The major challenge for this approach is that as the size of the system grows, it could be labor intensive to perform feature selection [7].

**2.0 Data**

**2.1 Data Sources**

The datasets used for this project were selected from the Yelp Open Dataset [8] as listed below.

*Table 1. Datasets and selected features*

| **Dataset Name** | **Selected Features** |
| --- | --- |
| Business | business\_id, business attributes (price range, alcohol, wifi), food categories (e.g. pizza, American, fast food) |
| User | user\_id |
| Review | user\_id, business\_id, review\_id, rating, review date |

**2.2 Exploratory Data Analysis (EDA)**

Given the above three datasets, the first step was to inner join the three datasets based on business\_id and user\_id. After obtaining the combined dataset, we filtered out missing and invalid data entries. Because the project aimed to develop a personalized restaurant recommendation system, we filtered out other businesses on Yelp such as hair salons and only kept the restaurant review data. Moreover, since all team members live in Toronto, we decided to use Toronto data to build our recommendation system.

Afterward, we obtained a cleaned dataset which contained around 45 thousand reviews left by 12,039 users for 1,798 restaurants dated from 2008 to 2018. We found that more than 60% of the users only left less than 3 reviews (Appendix A). In order to build more accurate models, only users with more than 20 reviews were selected. The modified final dataset contains 8,632 reviews left by 263 users for 1,395 restaurants, which is a decent size for model building. Lastly, we calculated and added some new features to the dataset, including the number of reviews each user left and each restaurant received, as well as the average rating each user gave and each restaurant received. These features were used in the random forest model for the content-based filtering method as described in the Methods section below.

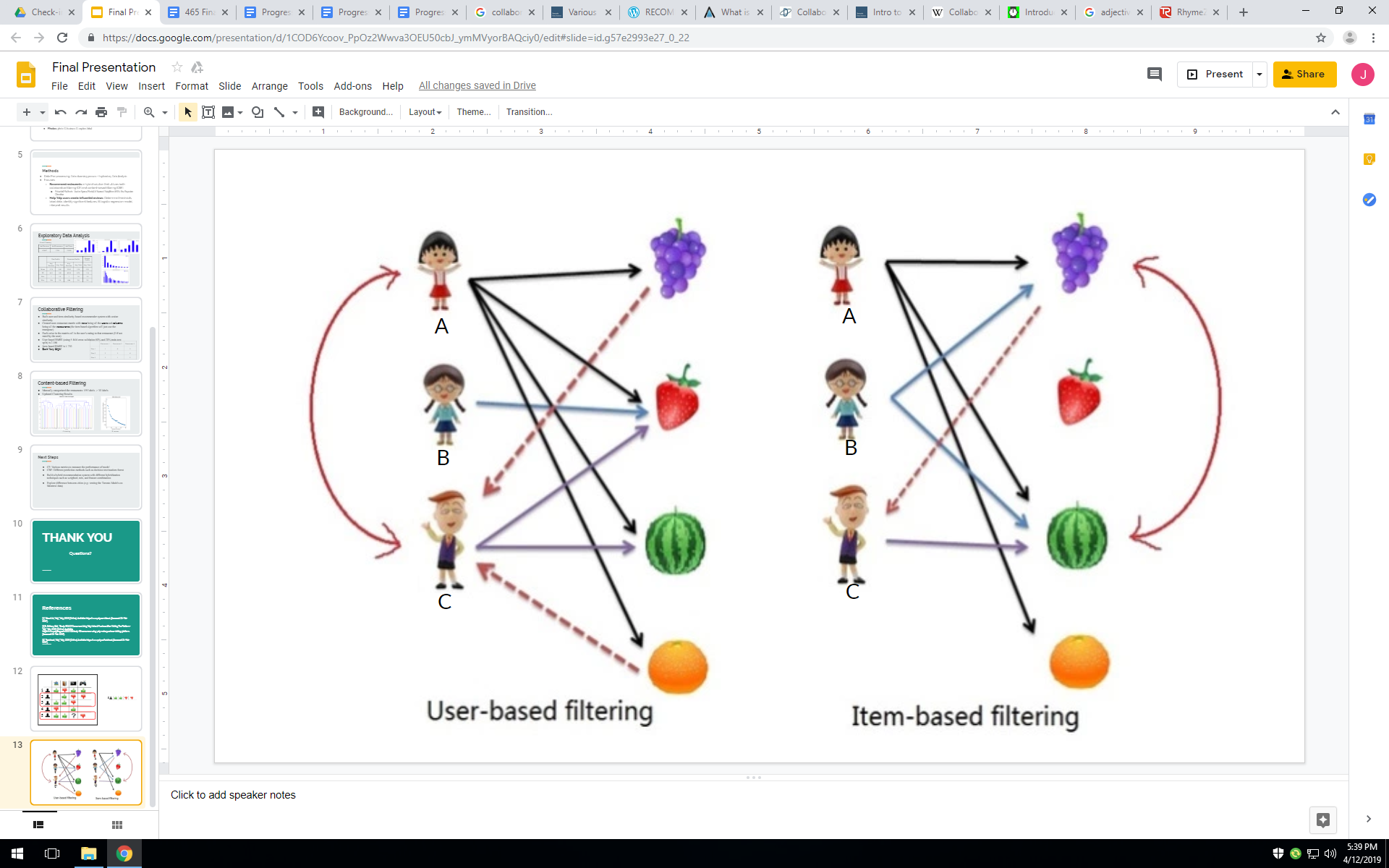
## **3.0 Methods**

### **3.1 Data Preparation**

To improve performance, we used a continuous value between 1 to 5 for rating prediction. The team randomly split the dataset into a training set (80%), and a testing set (20%) for the model building. The training set including 6,582 reviews was used to train the models. The testing set including 1,624 reviews was used to validate and compare the models’ performances as out-of-sample testing data. We also conducted checks to ensure the two data sets have a balance of different ratings (Appendix B).

### **3.2 Collaborative Filtering**

Collaborative filtering is a method of predicting future user preferences for items by collecting preferences from other similar users. Collaborative filtering is a reasonable method to try because our dataset contains enormous explicit ratings which give the potential of extracting user preference patterns. The major underlying assumptions are as follows: 1. Users will like items that are liked by users who share similar tastes/interests; 2. Users will like the items that are similar to the things they like. Following these assumptions, we have adopted two collaborative filtering methods, user-based and item-based filtering (Figure 1).

****

*Figure 1. User-based and item-based filtering [9]*

**3.2.1 User-based Collaborative Filtering**

User-based filtering has applied the first assumption: “users will like items that are liked by users who share similar tastes/interests”. As shown in the left part of Figure 1, User A and C are considered similar because they both liked strawberry and watermelon. We can expect User C to like grape and orange (the two fruits User A likes) with the assumption that similar people will like similar things.

To implement this idea in our case, we needed to work with numbers and formula to compute the predicted ratings. In order to predict User X’s rating for Restaurant X, mainly two factors were considered in computation: other users’ ratings to the restaurant and the similarity between the current user and other users. If two users have a high similarity, they tend to have similar taste and give similar ratings for the restaurant. Cosine similarity was used as our similarity measurement. More details about how to form rating matrix and compute cosine similarity can be found in Appendix C. The result is further divided/normalized by the sum of all similarities to keep the prediction rating within 1 to 5. The formula is shown below:



Finally, the top 5 restaurants with highest predicted ratings will get into the recommendation list.

**3.2.2 Item-based Collaborative Filtering**

Unlike user-based approach, item-based collaborative filtering generally follows the assumption that “users will like the items that are similar to the things they like”, which indicates a focus on item similarities. Referring back to Figure 1, on the right part, because both User B and User C like watermelon and User B also likes grape, grape is recommended to User C. In general, if one restaurant received good ratings from both User 1 and 2, then another restaurant received high rating from User 1, will be predicted to receive a good rating from User 2.

In our context, to predict restaurant rating by users, two factors were considered in computation: other users’ ratings to the restaurant and the similarity between the restaurant and other restaurants. If two restaurants are given high rating by the same user, they are considered similar. As restaurant is the core subject now, the formula will be based on restaurant similarities rather than user similarities. Other implementation remained the same.

### **3.3 Content-based Filtering**

The second recommendation system utilized content-based filtering and random forest prediction model. Content-based filtering method used attributes of a item, in this case, the characteristics of a restaurant to make recommendations. Firstly, we used hierarchical clustering to group restaurants into different clusters based on their characteristics. Secondly, after the clusters were formed, they were used as an input feature to the random forest model to predict user ratings for restaurants in different clusters. Finally, top five restaurants with the highest rating will be be recommended.

#### **3.3.1 Clustering**

**Feature selection**

Selecting proper features is very crucial for the performance of content-based filtering recommendation system. The original yelp dataset contains more than 200 attributes for a restaurant, we manually selected 34 features to cluster restaurants before building the prediction model. These 34 final features include the region of the food (e.g: Canadian), food type (e.g: ramen), and occasion of the restaurant (e.g: steakhouse), and are listed in Appendix D.

**Hierarchical Clustering**

Clustering is a common technique to group more similar objects. To group the 1303 restaurants, we tried both hierarchical and k-means clustering methods, and selected hierarchical clustering as its dendrogram was clear in visualizing the distribution of clusters and could help us to select a proper cut-off point for determining the number of clusters required. Different linkage and distance rules were tested, and we found the complete rule with Euclidean distance yielded the most evenly distributed clusters (Appendix E). After testing with different cut off points, a cut-off distance 3.1 was selected, as it returns 9 clusters with reasonable sizes (Figure 2).

******

*Figure 2. Nine clusters and their sizes*

#### **3.3.2 Random Forest Model**

After the restaurants were grouped into clusters, we tried the random forest model to predict user ratings for the restaurants, by considering the cluster number as an input feature variable. Other features, as listed in Table 2, were also used as the random forest input variables. The actual users’ ratings were the target variables for model training.

*Table 2. The input feature variables for the random forest prediction model.*

|  | **Input variables** | **Relevance** |
| --- | --- | --- |
| 1 | Restaurant average stars | This feature gives a general idea of a restaurant’s quality and popularity. |
| 2 | Restaurant reviews count | The large or small number of reviews a restaurant received might be relevant to its quality. For example, a restaurant with a lot of reviews might suggest that it is very popular; a restaurant with a few reviews might suggest it is just ordinary. |
| 3 | User reviews count | This feature can suggest whether the user is very active in exploring restaurants which may be related to his/her reviewing behaviour. |
| 4 | User average stars | This feature suggests whether a user is more accepting or more picky when reviewing restaurants. |
| 5 | Cluster number | Restaurants in the same clusters might have more similar ratings. |

Since the cluster numbers for the restaurants are categorical, we applied one-hot encoding to transform the nine cluster numbers into nine sets of binary values for the restaurant clusters.

To build the random forest model, the earlier training data set was further splitted into 80% training set (5418 data points) and 20% testing set (1355 data points). We applied five-fold cross validation in training our model to prevent overfitting. Grid search was used for parameter tuning to determine the best parameters. The best parameters returned by the grid search are listed in Appendix F.

In addition, as the input rating features and review count features are in different scales (the ratings are from 1 to 5, while the review count ranges from 3 to 6314), we tried data normalisation. Normalisation of the data was expected to increase the model performance. However, the RMSE of the normalised model increased by only 0.01, which is insignificant. Hence we decide to proceed with the random forest model without normalisation.

### **3.4 Building Hybrid Model**

Researches have shown a hybrid recommendation system that combines both collaborative and content-based filtering methods could be more effective in certain cases [10]. Hence, the team tried combining the predicted results from both similarity-based model and the random forest model to compare the performance of the two separate models as well as possible hybrid models on the testing set. A linear regression model and two weighted models were developed to integrate the similarity-based and the random forest models. Only the training set mentioned in Section 3.1 was used to train the linear regression model and to determine the weight factors for weighted models. The testing set were used to compare and evaluate different models’ performance.

**Linear Regression Model**

A linear regression model was developed with predicted ratings from the similarity-based model and the random forest model as input variables. However, our target values were discrete, and the predictions from the user-similarity model did not follow a normal distribution. Hence the linearity and normality assumptions were not met. This is proven by our testing result (in Appendix G). The linear regression was not a good model to predict user ratings, and hence it is excluded from further analysis.

**Weighted Model**

Inspired by taking the average outputs of the two single models, the team tested assigning different weights to the output of the previous similarity-based and the random forest models. In addition, a second weighted combination model of the random forest model and the user average rating[[1]](#footnote-0) is also built, in the hope that user average can help adjust the predictions, as the user average rating pseudo model performed better than the user similarity model.

**3.2 Evaluation Metrics**

RMSE and p@k are two commonly used metrics in evaluating performance of recommendation systems. Both metrics were used to evaluate our models and they are introduced in the following:

**Root mean squared error** (**RMSE)**: It tells on average how different the predictions and the true ratings are. Our rating data has a rating from 1 to 5; thus we have set a RMSE less than 1 as our goal.

**Prediction precision at k (P@k):** It indicates among the top k items that are recommended to the user, how many of them are truly liked by the user. We selected k to be five, meaning only the top five restaurants would be recommended. Among them, restaurants with a rating higher than four were defined as true likes. We believed we should not overwhelm users by a large number of recommendations.

## **4.0 Results**

Random forest outperforms all of the other models in predicting users ratings for restaurants as it has the smallest RMSE and largest P@k as shown in Table 4, which compares the performance of different models. If top five restaurants are recommended to a user, there is a probability of 58% (three out of five) that the user will actually like the restaurants. One of the possible reasons for the better performance is that the random forest model is an ensemble model. Via bootstrapping, different aspects of the reviews are studied by a group of various decision trees, and if most trees predict well, the random forest model will be able to make accurate predictions. Secondly, as a stacked model, the information of restaurants has been learned by the clustering model, and contributed to the prediction performance for the random forest.

*Table 3. The comparison of performance of different models.*

| **Model Description** | **RMSE on training set** | **RMSE on testing set** | **P@k on training set** | **P@k on testing set** |
| --- | --- | --- | --- | --- |
| User average rating model | 0.94 | 0.94 | 0.48 | 0.48 |
| Identical performance on both training and testing set is reasonable because the predictions are fixed (by individual average rating). | | | |
| User-similarity model | 0.58 | 1.16 | 0.97 | 0.44 |
| Significant performance difference in the training and testing set indicated potential overfitting so the team double checked the codes and ensured there was no mistake. The possible reason was that the user preference patterns were distinctive for each user and thus it was hard to generalize when looking at other people’s ratings. | | | |
| Item-similarity model | 0.79 | 1.12 | 0.97 | 0.44 |
| Item-based algorithm performed better than user-based in terms of might be because the restaurant data points were more dense (i.e received ratings from many users). | | | |
| Random forest model | 0.86 | 0.87 | 0.76 | 0.58 |
| Consistent RMSE on training, testing sets. While the P@k for testing set dropped, the potential reason might be prediction ability on a smaller sample (top 5) was measured. | | | |
| Weighted model combining item-similarity and random forest | 0.79 | 1.12 | 0.97 | 0.44 |
| Assigned Weight: Item-similarity model = 1 & Random Forest = 0. It is identical to the item-similarity model. This weight was assigned because the training RMSE for the item-similarity model was smaller than that of random forest model. | | | |
| Weighted model combining user average rating and random forest | 0.86 | 0.87 | 0.76 | 0.58 |
| Assigned Weight: Random Forest = 1 & User average rating = 0. It is identical to the random forest model. This weight was assigned as the training RMSE for the random forest model was smaller than that of user average rating model. | | | |

# **5.0 Discussion**

We have selected the random forest model to be the optimal recommender. Since the restaurant recommender models were built and tested using Toronto data, we wondered how well the final model would perform on other city’s data. Due to similar city characteristics, Montreal data filtered with users having more than 20 reviews were chosen. The final Montreal dataset contains 987 reviews left by 38 users for 371 restaurants and the random forest model had a RMSE = 0.91. The result was slightly worse than the result of Toronto data. It is acceptable because from research on existing Yelp restaurant recommendation system projects, a model with RMSE smaller than 1 was considered fairly good [11].

**6.0 Conclusion**

To help Yelp users to discover restaurants they may like, we built a collaborative filtering system, a content-based filtering system and two hybrid recommendation systems. The proposed model, random forest, outperformed all of other models, with a RMSE of 0.87 and P@k of 0.58 on the testing set for Toronto. Hence, with the help of our recommendation system, three out of five restaurants recommended to yelp users in Toronto will be appreciated. In addition, a validation on Montreal data was also conducted, and returned a slightly worse result compared to Toronto.

# **7.0 Future Directions**

Further exploration on hybrid models combining collaborative and content-based filtering methods, such as incorporating some content-based characteristics into collaborative approach, can be conducted to increase the performance of predicting user ratings [13].

In addition, an optimization model of what restaurants to recommend under certain conditions can be developed. User can provide constraints to discover restaurants they would like. Furthermore, an interface can be developed for the Yelp website and the Yelp application for users to input filters to generate constraints for optimization.

# 

# **8.0 References**

[1] "Eating out is the top financial indulgence for most Canadians", The Globe and Mail, 2019. [Online]. Available:https://www.theglobeandmail.com/globe-investor/personal-finance/household-finances/eating-out-at-restaurants-is-the-top-financial-indulgence-for-most-canadians/article36597841/. [Accessed: 1- Apr- 2019].

[2] "About Us | Yelp", Yelp, 2019. [Online]. Available: https://www.yelp.com/about. [Accessed: 01- Feb- 2019].

[3] Pan C, Li W. Research paper recommendation with topic analysis. In Computer Design and Applications IEEE 2010;4,pp. V4-264.

[4] Isinkaye, F., Folajimi, Y. and Ojokoh, B. (2015). Recommendation systems: Principles, methods and evaluation. Egyptian Informatics Journal, 16(3), pp.261-273.

[5] Herlocker, Jonathan L., et al. ”An algorithmic framework for performing collaborative filtering.” Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 1999.

[6] Lee, Sanghack, Jihoon Yang, and Sung-Yong Park. “Discovery of hidden similarity on collaborative filtering to overcome sparsity problem.” International Conference on Discovery Science. Springer Berlin

Heidelberg, 2004.

[7] Lops, Pasquale, Marco De Gemmis, and Giovanni Semeraro. ”Content-based recommender systems: State of the art and trends.” Recommender systems handbook. Springer US, 2011. 73-105.

[8] "Yelp Dataset", Yelp.com, 2019. [Online]. Available: https://www.yelp.com/dataset/download. [Accessed: 1- Feb- 2019].

[9] "RECOMMENDER SYSTEMS 101", d4datascience.wordpress.com, 2019. [Online]. Available: https://d4datascience.wordpress.com/2016/07/22/recommender-systems-101/. [Accessed: 13- Apr- 2019].

[10] Arxiv.org, 2019. [Online]. Available: https://arxiv.org/pdf/1901.03888.pdf. [Accessed: 1- Apr- 2019].

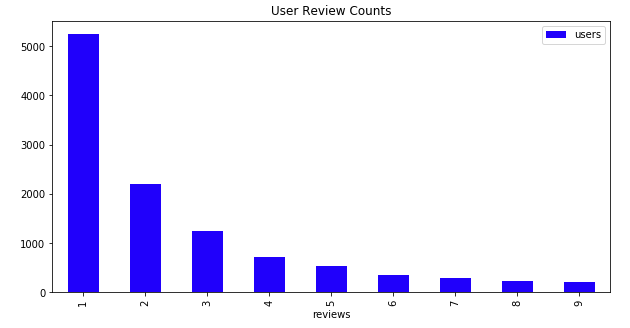
[11] Li, Y. and Song, H. (n.d.). [online] Pdfs.semanticscholar.org. Available at: https://pdfs.semanticscholar.org/5c7a/8f1f630428e7a471e831dde6514e91721492.pdf [Accessed 1 Mar. 2019].

[12] "Collaborative filtering", En.wikipedia.org, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Collaborative\_filtering. [Accessed: 13- Apr- 2019].

[13] Ids.csom.umn.edu, 2019. [Online]. Available: http://ids.csom.umn.edu/faculty/gedas/papers/recommender-systems-survey-2005.pdf. [Accessed: 13- Apr- 2019].

# **9.0 Appendices**

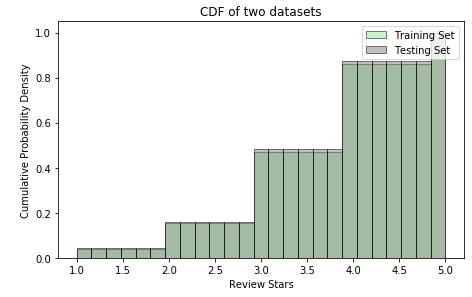
**Appendix A - EDA Finding: Many users only left a few reviews**



*Figure 3: Review counts with corresponding user counts*

The figure above shows review counts (only 1-9 is shown as an example) and the corresponding user counts. We can see that from the total 12,039 users, more than half of the users have left only one review and more than sixty percent of the users have left less than three reviews. In order to increase data density for better model performance, we decided to filter out users with less than 20 reviews which left us with 263 users’ data for model training and testing.

**Appendix B - Checking the balance of different ratings in the training and testing sets**

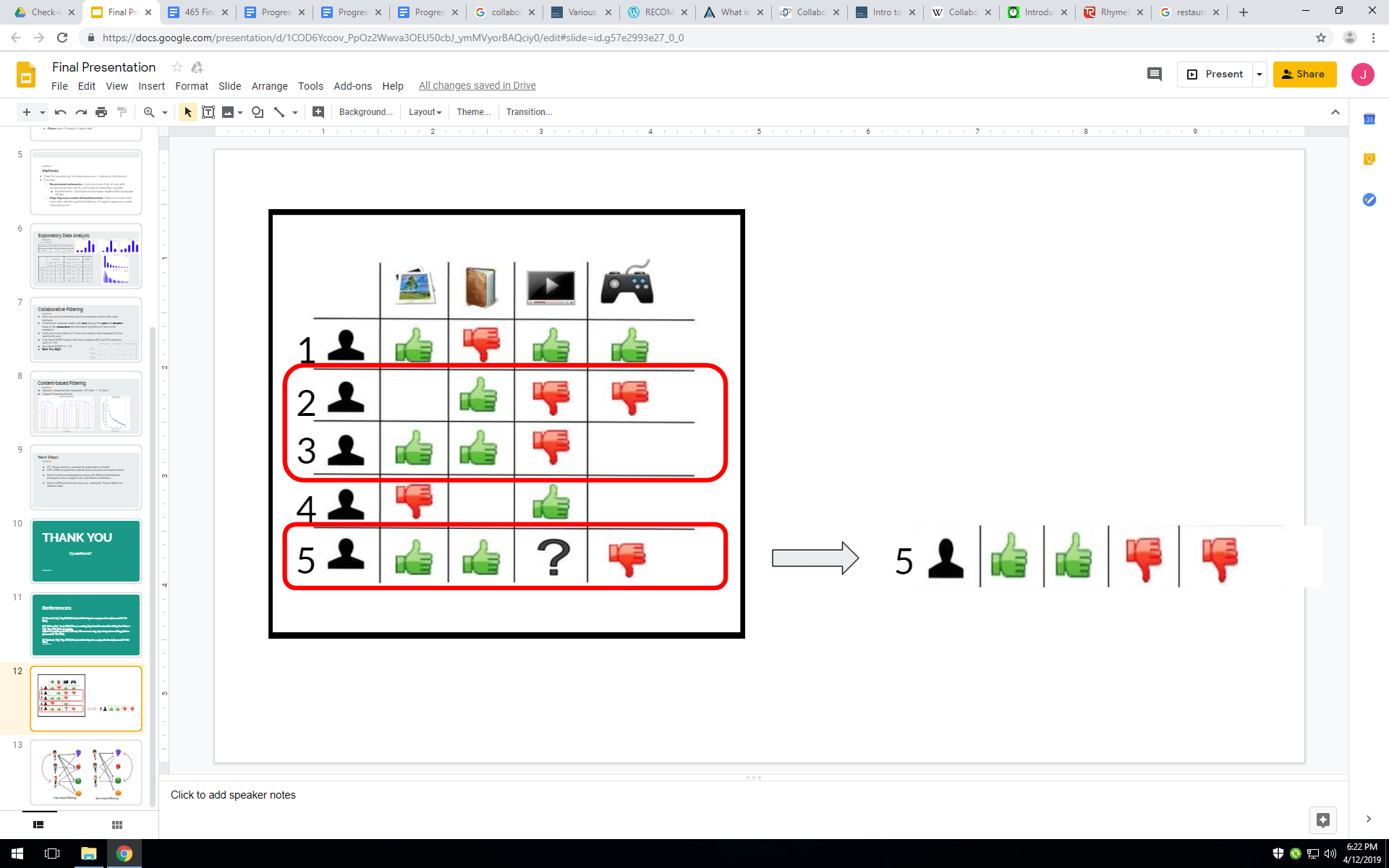


*Figure 4. Distribution of actual ratings in training and testing datasets*

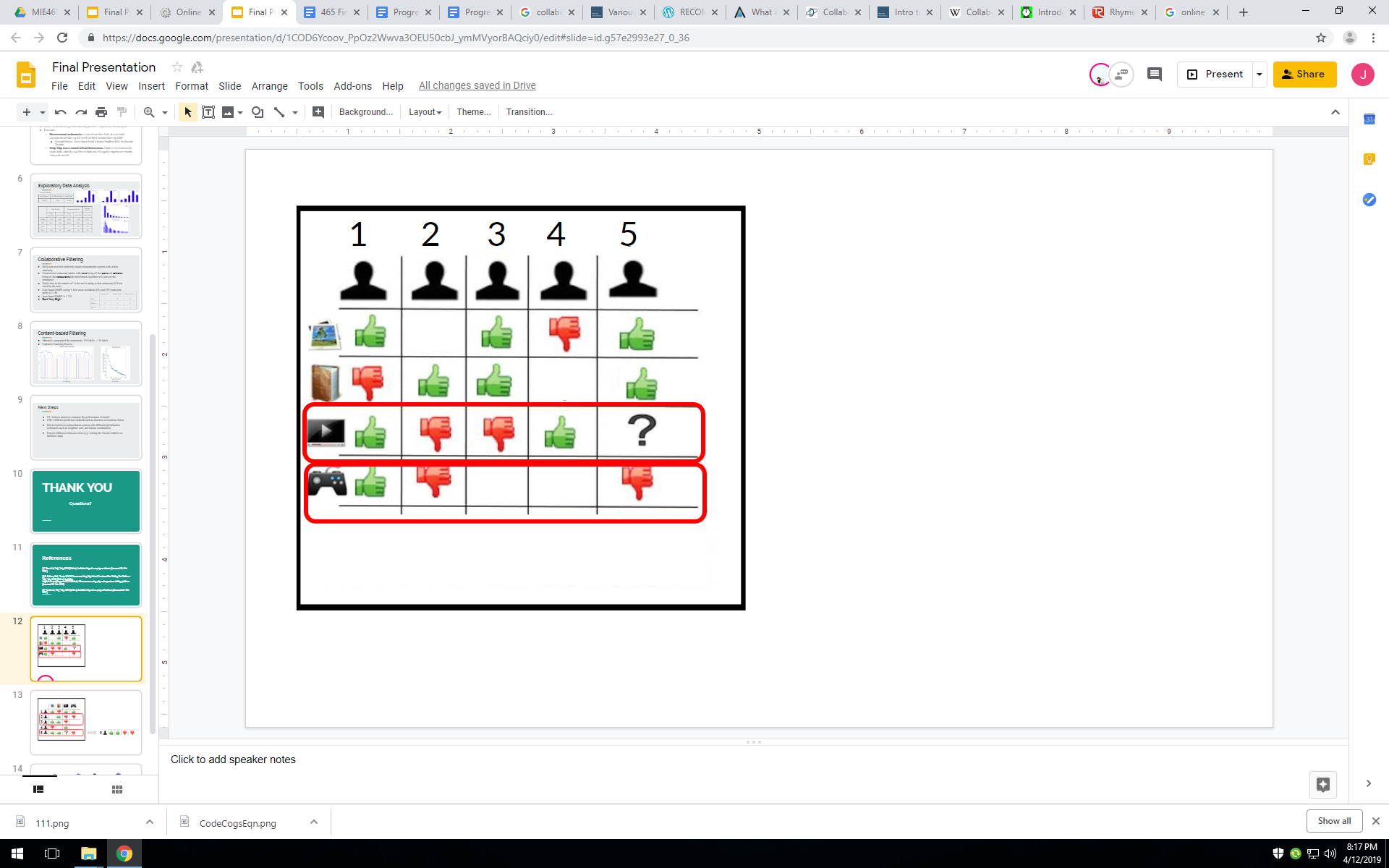
According to this plot, the team ensures that the distribution of different ratings are almost the same within both training and testing sets.

**Appendix C - Detailed Rating Matrix Formation and Cosine Similarity Computation**

To implement collaborative filtering, we have rearranged the data into a matrix. Each row represents a user while each column represents a restaurant. Figure 5 below gives a visualized example of this idea. The goal in the example is to predict whether User 5 will like the TV. After looking at similar users (User 2 and 3), the system will predict User 5 doesn’t like the TV because neither User 2 nor 3 liked the TV. As for item-based approach, the matrix will just be transposed and put in use. Figure 3 provides an example matrix for item based matrix.

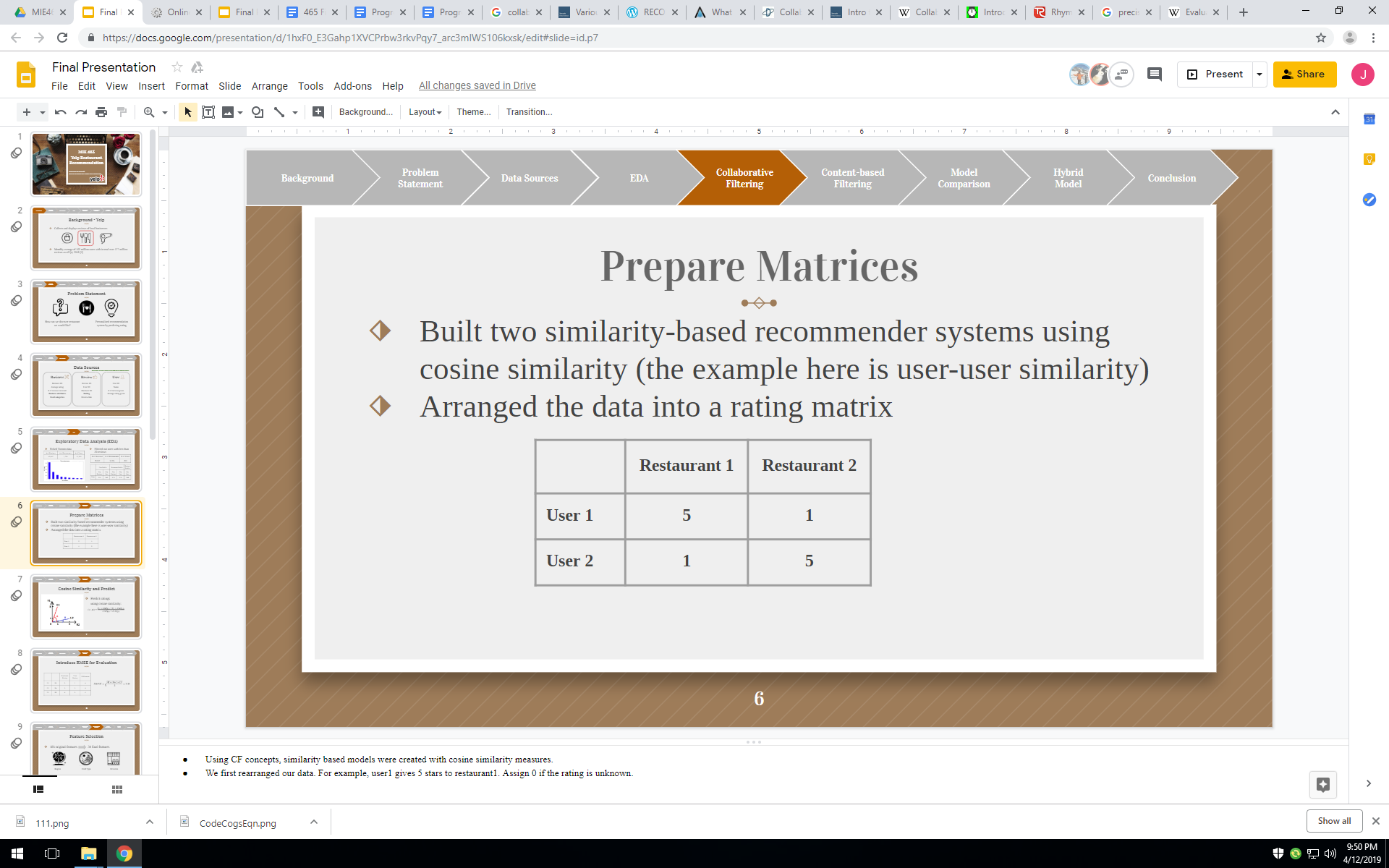


*Figure 5. An example of user based approach matrix [12]*

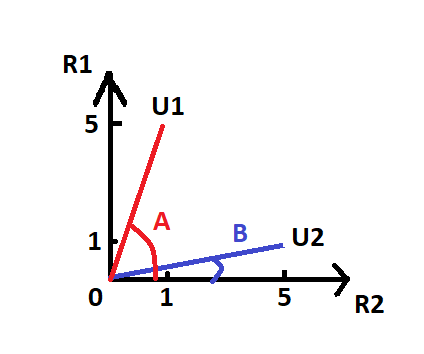


*Figure 6. An example of item based approach matrix [12]*

With the matrix built, the next step was calculate pairwise cosine similarity between each pair of users. Taking the mini matrix in Figure 7 as an example, the rating vectors of User 1 and User 2 to Restaurant 1 and Restaurant 2 were plotted. From a high-level perspective, their cosine similarity is basically comparing the angles (A and B) between those rating vector and the x-axis (Figure 8). In this case, their similarity will be very small due to the large difference in the angles.



*Figure 7: Small fragment of the rating matrix*



*Figure 8: Rating vectors and corresponding angles*

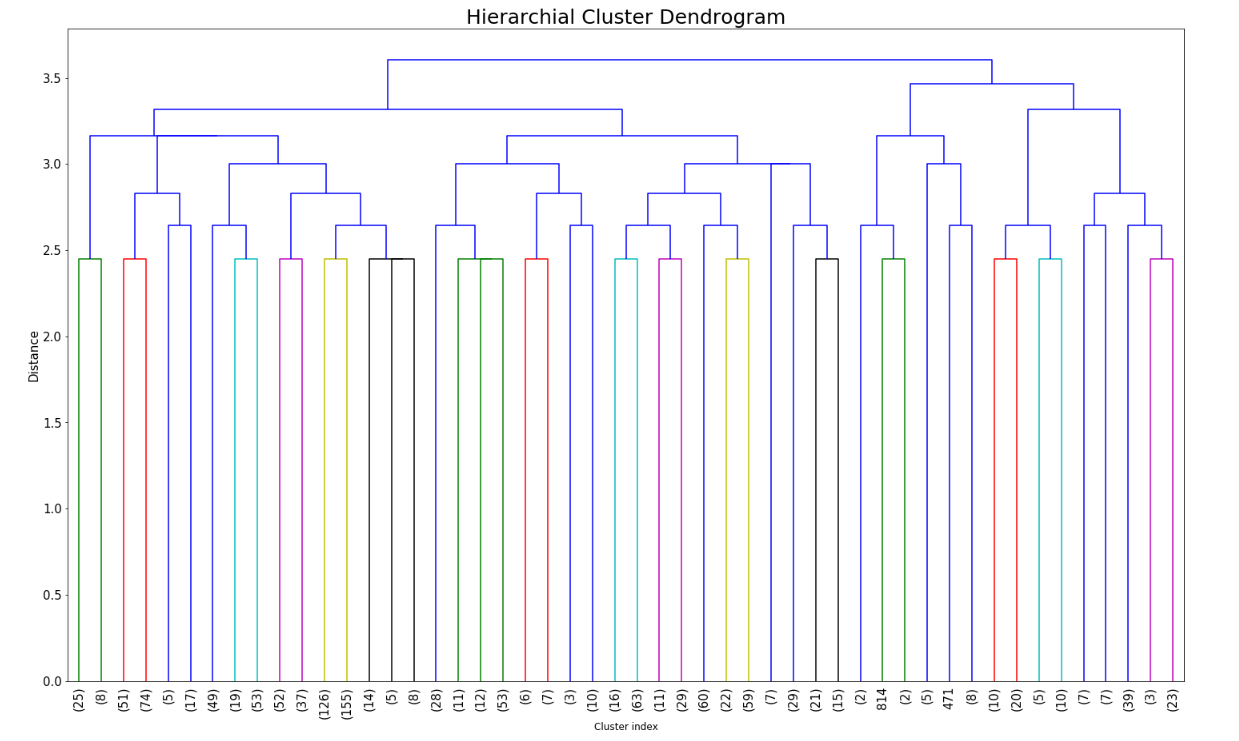
**Appendix D - Clustering technique: Reduce the number of features**

*Table 4. Feature selection for clustering*

| Selected Features | | |
| --- | --- | --- |
| * African * Alcohol * Bakeries * Barbeque * Bars * Breakfast & Brunch * Buffets * Cafes * Delivery * Desserts * Diners * East Asian | * European * Fast Food * Gluten-Free * Halal * Kosher * Latin American * Mediterranean * Modern European * Night * North America * Other Asian | * Poutineries * Price Range * Restaurants * Seafood * South Asia * South Eastern Asian * Special Settings * Steakhouses * Vegan * Vegetarian * Wifi |
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|
|

To improve the performance of clustering model, we manually selected 34 final features from more than 200 original features. Those final features were consist of three major types: region, food type, and occasion.

**Appendix E - Hierarchical clustering result**

****

*Figure 9. Dendrogram for hierarchical clustering*

Above dendrogram shows that the results for hierarchical clustering. We only chose to plot 50 nodes to save the time to render, and the top 50 nodes are good enough for us to pick a cut-off point. According to the tree diagram, most of the clusters are evenly distributed.

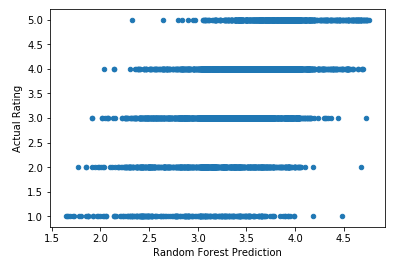
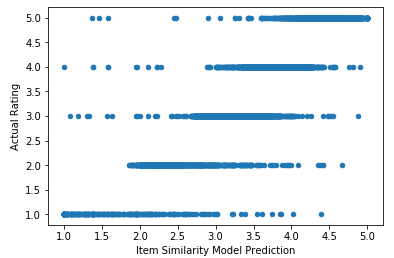
**Appendix F - Random forest model parameters**

*Table 5. Random forest model parameters as a result of grid search*

| **Parameter name** | **Explanation** | **Value** |
| --- | --- | --- |
| max\_depth | The maximum depth of trees | 6 |
| n\_estimators | The number of trees | 200 |
| min\_samples\_leaf | Minimum number of samples required at a leaf in order to further split on this node | 8 |

**Appendix G - Checking assumptions for linear regression**

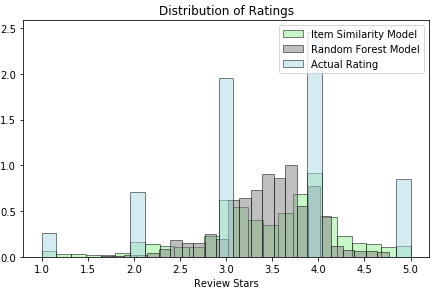
**Checking linearity**

****

*Figure 10. Scatter plots for linearity check*

Above scatter plots are created to check the linearity of the input features (user similarity model prediction, and random forest prediction) and dependent variable (actual rating). As the actual ratings are discrete value, the linearity assumptions are not met.

**Checking normality**



*Figure 11. Distribution of model predictions and actual rating*

According to the plot, the actual ratings, and the predictions from user similarity are not following normal distribution. While the predictions from the random forest model follows a left-skewed normal distribution.

**Testing result for linear regression**

| RMSE on Training Set | RMSE on Testing Set |
| --- | --- |
| 0.47 | 1.79 |

1. User average rating model is a dummy baseline model created to be compared with the other models in terms of the two metrics. It simply predicted a user’s ratings for all restaurants the user’s historical average ratings [↑](#footnote-ref-0)