Final Project Report

Zhipeng Zhu, Meixian Wu, Yuxi Fan

1. Problem & Background Description

Searching for restaurants has been a major need for many people. To address the problem of recommending restaurants to customers, platforms such as Yelp and Google Maps use a recommendation system that recommends specific restaurants to users based on data. As the popularity of recommendation systems increases, it is worth discussing what specific machine learning models or recommender mechanisms can be implemented to achieve such functionality. The two common methods are content-based filtering and collaborative filtering. Our group believes that it is meaningful to compare different implementations of recommender systems. We have implemented both methods to see how they perform on Yelp restaurant datasets. We will discuss the details of implementations and their advantages/disadvantages compared to the others in the following sections.

Our objective is to implement a restaurant recommendation system, specifically for users in the U.S, by using data-mining techniques including data transformation, data analysis, and machine learning models (collaborative filtering and content-based filtering). The tasks of the project are exploratory data analysis of user & restaurant data, recommendation of relevant restaurants based on a specific user profile, and evaluation of model accuracy.

2. Dataset Description

Yelp Open Dataset: *https://www.yelp.com/dataset*

We used an open dataset provided by Yelp, which contains detailed restaurants and user information. It consists of six separate datasets: restaurant dataset, customer dataset, check-in dataset, tip dataset, photo dataset, and review dataset. For our project, we worked on three of the datasets: user, business, and review. The user table contains information about yelp users such as id, name, review_count, etc. The business table contains information about restaurants including id, name, location, stars, etc. Finally, the review table contains reviews of restaurants written by

users. The attributes include id, user_id, business_id, starts, and the review text. All the datasets are in the form of JSON. We retrieved it with Python and filtered out the restaurants outside the United States.

3. Description of method used

Data processing

When we loaded the raw data for exploratory analysis, the running time of our program was very long because of the size of the original dataset. In this case, we decided to randomly sample the dataset to reduce the runtime. The method we chose is Reservoir Sampling. Reservoir Sampling is a family of randomized algorithms for randomly choosing k samples from a list of n items, where n is either a very large number or an unknown number. We prepared data using the following steps:

1. Filter out all the businesses that are not in the U.S. by using regex to match the pattern of business ZipCode. 2. Drop records that do not contain “restaurant” in their categories. 3. Using Reservoir Sampling Algorithm to randomly extract a subset of restaurants. 4. Filter the Review dataset and only keep the reviews relevant to our chosen restaurants. 5. Filter the User dataset and only keep the users who are relevant to our chosen restaurants. 6. Flatten attributes in nested JSON format to individual features.

Exploratory analysis

All tables were processed in Jupyter Notebook and loaded as Pandas Dataframe objects. First, data of non-US restaurants were removed from the datasets. Exploratory analysis was performed in each table. We primarily focused on the distribution of restaurant overall ratings, restaurant categories, and user ratings. In addition, we also analyzed user reviews to extract insights from the texts. Notable visualizations and insights are discussed in the “Results” section.

Recommendation Models

a. **Content-based**

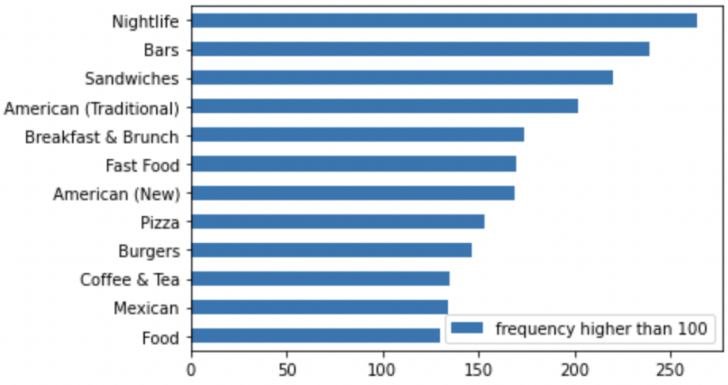
The content-based filtering model gives recommendations based on the similarity of items. This means that if a user likes a certain restaurant, the model would recommend other items which are similar to this one. In this scenario, the model does not consider other users' data, but only focuses on the attributes of the restaurant, such as category and stars. In our case, the content implies the restaurant's features. During data processing, we have converted the categorical attributes to dummy variables. The very first step is to build the model by finding similarities between all the item pairs. A similarity matrix was created with a pairwise cosine similarity function with the restaurant attributes and rating input. The resulting matrix will reflect the similarity of any two restaurants. In this way, the top K similar restaurant can be selected based on the similarity values rankings.

b. Collaborative filtering

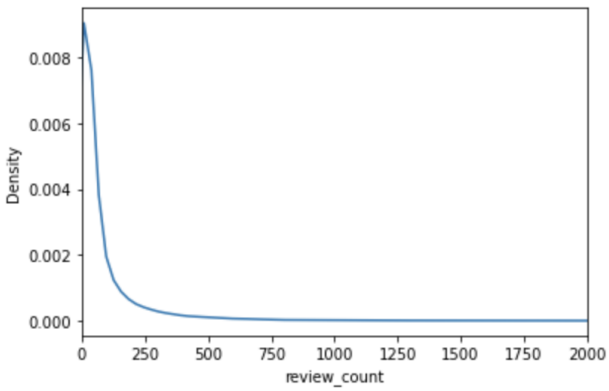
Collaborative filtering is a user-based approach. It is called user-based because it predicts unknown ratings by using the similarities between users. When building a recommendation system with collaborative filtering, users are provided with the recommended items that people with similar tastes and preferences like in the past. The final model for collaborative filtering is Singular Value Decomposition (SVD). It uses a matrix structure, where each row represents a Yelp user, and each column represents a restaurant. The elements of this matrix are the star ratings given to the restaurant by users.

4. Results

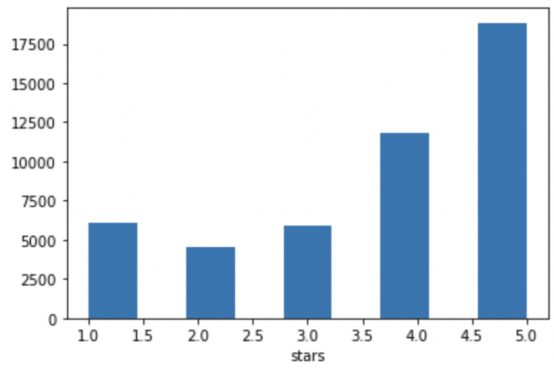
Results from exploratory data analysis



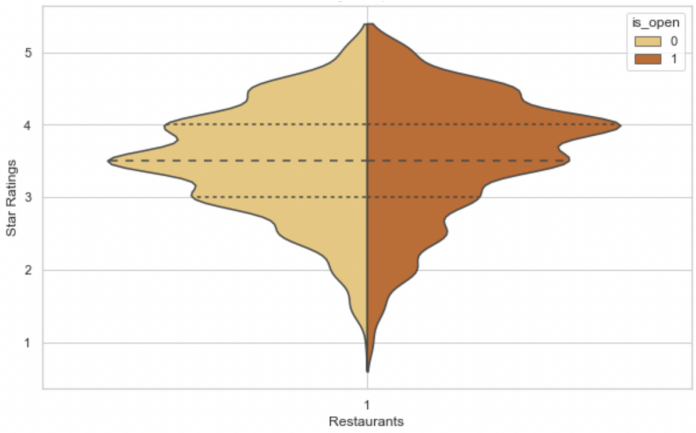
(a) Most frequent restaurant categories



(b) Distribution: the number of reviews each user wrote



(c) Number of reviews for each rating



(d) Star Ratings Distribution of Open and Closed Businesses

The visualizations above show the most common restaurant categories, distribution of the number of reviews each user wrote, and the distribution of ratings. Here we can see that the distribution of review count is very skewed (b), meaning that a significant proportion of users only post a few reviews. This is important because it affects the performance of a user-based recommendation model since many users do not actually share interests.

(e) 5-star reviews

(f) 1-star reviews

The word clouds show the most frequent words appearing in both 5-star reviews and 1-star reviews.

Results of Model Implementations

After implementing the content-based model and the collaborative filtering model, we found that these two models can achieve different functionalities in terms of user inputs and

recommendations. Detailed functions are listed below, and comparisons of the models will be discussed in the next section.

Function 1: Predict top K restaurants that are similar to a selected one (content-based)

When a user inputs a restaurant, we are able to give top k restaurants that are most similar to the given one, in the hope our user may like it also. The method looks for restaurants that are similar to the one that the user has provided and recommends the most similar restaurants. Originally, we have a dataframe with high sparsity, because restaurants each have different attributes and most of the attributes don't overlap. Thus, in the dataframe where we kept all attributes that appeared, most of the data instances are empty. Values in the resulting similarity matrix are all extremely close to 1, meaning that the function can not distinguish the difference between restaurants with such sparse data input. However, when we filtered out attributes with less than 200 occurrences among the 2000 restaurants, the attributes number cut from 592 to 70, the resulting similarity matrix is much more informative and can therefore make reasonable predictions.

Here is our prediction for a sample input:

```
item_similarity = pd.DataFrame(cosine_similarity(business_dummies_highfreq), index = business_dummies.business_
top_k_recommend = item_similarity[business_input].sort_values()[:k].index
top_k_name = GetName(top_k_recommend)
print('Your input restaurant: {} \n'.format( input_name))
print('The top {} restuarants recommended to you: \n{}'.format(k, top_k_name))
```

```
Your input restaurant: Caribbean Delight

The top 5 restuarants recommended to you:
["McDonald's", 'Tiny One', 'Waves Coffee House', "T's Tavern", 'Chillax Cafe']
```

Function 2: Predict top K restaurants based on users' past behavior (collaborative filtering)

The data we used filtered out the users who had written less than three comments. We then split the dataset into a training set and a testing set. Our model performs nicely after turning the SVD parameter with an RMSE = 1.0861.

```
##Model with smallest RMSE
model = SVD(n_epochs = 50,n_factors= 10, biased = True, lr_all = 0.005)
```

The model can be used to provide users recommendations based on the restaurant they have rated in the past. Here we create a sample user who rates Athenian Bar & Grill and Applebee Bar & Grill 5.0 stars, but Starbucks and Dunkin' only 2.0 stars.

The recommendations provided for the User are Gonzo At Bar XV, Pokeworks, Pomodoro, Cold

Prediction Result:

```
print(df_with_name[df_with_name.business_id.isin(favor.iid)].name.unique())
```

```
['Arepazo Tapas & Wine' 'Gonzo At Bar XV' 'Pokeworks' 'Cold Beer'
 'Pomodoro']
```

Beer, and Arepazo Tapas & Wine. The recommendations make sense because our sample user is a person who loves bar and grill and dislikes coffee stores.

5. Observation and Conclusion

Content-based Filtering

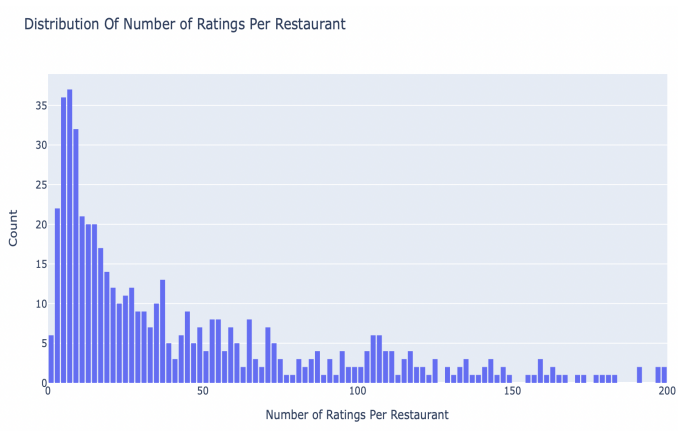
The content-based filtering model is straightforward since it only calculates the similarity of restaurants. The advantage of this model is that it does not need all user data to calculate user-user similarity. This feature can ensure better performance and scalability if the datasets are large. In addition, the recommendation results would not be affected by popular user preferences. Since the system can make recommendations solely based on the preference of a specific user, restaurants that only a few users like could also be recommended.

This model also has a significant drawback. We found that it is difficult to evaluate the accuracy of the model due to the fact that cosine similarity is calculated from attribute parameters without human interpretation. This means that the content-based model does not have a testing mechanism. However, in realistic practices, it would be possible to evaluate recommendations by looking at product metrics or different forms of user feedback.

Collaborative Filtering Observation

Algorithm	test_rmse	fit_time	test_time
SVD	1.070310	0.121615	0.004679
KNNBasic	1.207693	0.015763	0.015894
CoClustering	1.322392	0.092453	0.003898
NormalPredictor	1.531878	0.002070	0.004512

While doing research on collaborative filtering, we found that SVD is the most popular method among several machine learning models. By applying SVD, KNN, Normal prediction (predicts a random rating based on the distribution of the training set, which is assumed to be normal), and Co-clustering to our Yelp Review Dataset, we found out that SVD indeed has the smallest RMSE.



We also looked at the users with the smallest and largest prediction errors in the test dataset. The user with the smallest error has written many reviews and rated all the restaurants with 5.0 stars. In this case, it is not very hard to predict his rating for other restaurants based on his behavior. The user with the largest prediction error only wrote three reviews before, and two of the restaurants he reviewed had very few other reviews. As a result, it is hard to find users that have similar tastes to him accurately. Dataset quality is crucial to the performance of the SVD Model. Luckily, with the high volume of Yelp reviews, we have enough data to train well-performance recommendation models.

Model Comparison

During the implementation and testing of the two models, both models successfully helped to predict the restaurants that the user may like. For the content-based model, the algorithm is intuitive and context-independent. Yet the model is sensitive to sparsity. When the attributes have few overlaps, the data will have high sparsity, and the cosine similarity calculation may fail to tell the difference in similarity. The advantage is that the input requirement is simple, which only requires the user to identify a restaurant to make a recommendation. For the collaborative filtering model, the recommendation results are more customized and we can test

the accuracy of the model. However, it is highly dependent on the quality of the users' rating dataset. When using a users-based model, problems like fake reviews and bias ratings can largely impact the performance of the model.

6. References

Sharma A. (2020, May 29) Python Recommender Systems: Content-Based & collaborative filtering recommendation engines. Retrieved October 4, 2021, from https://www.datacamp.com/community/tutorials/recommender-systems-python

Rehberg, J. (2021, June 8). Recommend using Scikit-learn and TensorFlow. Medium. Retrieved October 4, 2021, from https://towardsdatascience.com/recommend-using-scikit-learn-and-tensorflow-recommenDer-bc659d91301a

Kumar, V.(2020, Mar 25) V.Singular Value Decomposition (SVD) & Its Application In Recommender System. Retrieved December 4, 2021, from https://analyticsindiamag.com/singular-value-decomposition-svd-application-recommender-system

Mavuduru, A. (2020, December 24). How you can build simple recommender systems with surprise. Medium. Retrieved December 5, 2021, from https://towardsdatascience.com/how-you-can-build-simple-recommender-systems-with-surprise-b0d32a8e4802.