# Review Augmented and Geo-Based Multi-User Restaurant Recommendation System

Hongyi Wang, Zhangming Sun, Jiawei Gu, Tzu Ching, Shiau Kang Wang, Shichuan Wang
*School of Engineering and Computer Science*
*University of Texas at Dallas, Richardson, TX, 75080, USA*

***Abstract*—In this project, we aim to find a solution of restaurant recommendation for individuals and groups based on the Yelp dataset. As increasing information and services, it becomes difficult to find a proper place to eat. Additionally recommendation for groups needs to consider more aspects for each member in the group, such as preferences and location. In our attempt, first, filtering a list of restaurant base on the multi-user real-time position. Then, ranking the restaurant with group rating, which is the maximum of the lowest rating in the group. The rating will be generated by using ALS method based on the restaurant average rating. To enhance the evaluation of restaurants we apply NLP analysis on the reviews, provides a reference for users while making the decision.**

## 1. Introduction

Nowadays, restaurant recommendation APP are wildly used, for example Yelp, Foursquare, etc. These APP only shows the nearest restaurant around one user, however, under many scenarios, friends who want to have dinner together are separate. So we try to build a program that allow restaurant recommend based on multi-users real-time location, and find a well-located restaurant by considering all.

## 2. Data Preprocessing

We are using the dataset of Yelp, which including 85901 records of business information, 686556 records of customer information, and records of 2685066 customer reviews. We narrow the task at Las Vegas who has 4928 restaurants, which is the largest number of restaurants among all the cities.

In order to give good recommendation, the restaurants with reviews less than 50 will be ignored. And for customers, in order to give precise result, the ones with reviews more than 10 will be considered as the service targets. It's 2182 restaurant and 23182 customer in total.

## 3. System Design

### 3.1. Overview

Our system mainly in four parts. Distance based filter, this part we will use google map distance matrix API to filter out the restaurants that is too far away. For the second part, ALS method will be use to predict the customer rating. Since most of the customers are friendly, they don't give lower score, so the average rating is biased. What we provide in the third part is a rating method based on the customer review, which reveals the customers' real feed back. For the last part is the presentation of the results, gives a better understanding of the results.

### 3.2. Distance Based Filter

In the recommendation system, we use Google Maps Distance Matrix API and it plays a significant role. It returns information according to routes between start and end points, when calculated by the Google Maps API. It also consists of rows including duration and distance values for each pair. Although this service does not provide detail data of routes, it can help us to achieve to essential purposes, creating matrix and displaying relative position.

Creating matrices is a key function in Google Maps Distance Matrix API and it improves our system to work more efficiently as well. Matrices are used to store users ID and restaurants ID. For example, if six persons are using the recommendation system and N restaurants are suitable for their reunion, a matrix size of $6 \cdot N$ will be created. Each element in this matrix represents the distance between users and restaurants. However, we encountered an obstacle while creating matrices. That is, Google restricts the matrix size per request. It is reasonable since as a free tier API, it can be used as any business related activity. To overcome the problem, we divide the whole matrix to lots of consistent small matrices. To illustrate the case above, we divide $N$ by 30 and then create $N/30$ small matrices. Later, we use Dictionary, one of Python Structure, to access these values. Finally, put each dictionary into List according to the order. By doing so, we are able to avoid the restriction from Google and acquire complete distance information. We also need to set up a boundary for the restaurants within area. The method is intuitive and workable. For example, if two users coordinates are X and Y, we use the center of them and set weight for the boundary to create a rectangle. A rectangle sized 0.08 times 0.08 in Coordinates systems will include around three hundred restaurants. Google Maps Distance Matrix API is not the most powerful one for calculation but it is truly a convenient one for relationship between

TABLE 1. AN EXAMPLE OF MATRIX OF RESTAURANTS AND CUSTOMERS

| | R 1 | R 2 | R 3 | ... | R $N$ |
|---|---|---|---|---|---|
| Customer1 | 1.0 | 1.0 | 1.0 | ... | 1.0 |
| Customer2 | 1.0 | 4.0 | 8.0 | ... | 8.0 |
| Customer3 | 10.0 | 5.5 | 5.5 | ... | 8.0 |

users and restaurants. It can save mutual distance for each group member and restaurants and save them efficiently. We will utilize Google Map Distance Matrix API to solve the problem. Step 1 is to find mathematical centroid of all users, and filtered out restaurant that is too far away. After that, Google Map API will be called to create a distance matrix between users and restaurants using coordinate. By using the maximum distance between a retardant and all user, we sort the restaurants from distance close to far. As is show in Tab1.

Location preference from high to low is: Resturant2 $>$ Resturant3 $>$ Resturnat5 $>$ Resturant1

Then we store the list of python Restaurant object. With a restaurant rating dictionary get from ALS, we will finally be able to choose the top N restaurant for all users.

### 3.3. Alternating Least Squares Method

Recommendation algorithm and procedure The core part of our multi user restaurant recommendation system is to provide user specific ratings for one particular restaurants. With the vector or multi-users preference for one particular restaurants, we can provide better recommendation for a group of customers. In order to solve this problem, here we applied the well-established latent factor model and using the plain alternating least squares (ALS) algorithm. The latent factor model treats the review rating of an item (denoted as $i$) given by a user (denoted as $u$) as the dot product of item vector $q_i$ and user vector $p_u$ (equation 1). The ALS algorithm calculate the user vector $q_u$ and item vector $q_i$ that can minimize the cost function (equation 2) by rotating between fixing the $q_i$'s and $p_u$'s. By applying this strategy, ALS algorithm converts the non-convex problem to two quadratic problems and solves them by least-square algorithm.

$$r_{ui} = q_i^T p_u (1)$$

$$\mathbf{min}_{p,q} \sum_{(u,i)\in k} \left(r_{ui} - q_i^T p_u\right)^2 + \lambda \left(\|q_i\|^2 + \|p_u\|^2\right)$$

The ALS algorithm is chosen because it can be easily parallelized on distributed computation system. Spark is one of the widely used distributed computation framework in big data field. And ALS algorithm is already implied by the spark machine learning library. During applying ALS algorithm on the yelp dataset, there are several challenges. The first is how to deal with multiple reviews given by a customer to a particular restaurant in different dates. In order to get the more accurate review given by the customer, only the last review is revered in the user item interaction matrix. The second is the cold start problem. Latent factor model using collaborative filtering strategy, and collaborative filter strategy have difficult on address new items and new users in the recommendation system. In order to overcome this, only well reviewed restaurants (restaurants with more than 50 reviews) and active users (users with review count more than 10 in well reviewed restaurants) were considered as input to ALS algorithm. In order to get more accuracy prediction, cross validation is applied on training the ALS model. The whole dataset was split into three parts, a training dataset with 60% data, a test dataset with 20% data and a validation dataset with the rest 20% data. Only the training and test dataset was used during the training process and the validation data set was used for the final validation. The rank parameter (the number of latent factors) was tuned during the training process. The final model with rank set to 120, gave a RMSE of 1.118 on test data set and a RMSE 1.113 on validation data set. Then the final model is applied on the whole matrix of all the well-reviewed restaurants and active users in Las Vegas area. The original rate gave by the user were reserved in the matrix, and for the user-item interactions that cannot be predicted by the ALS model (NaNs in the matrix), it will be replaced by the average rating of the restaurant.

### 3.4. Sentimental Analysis for Yelp Customers' Reviews and Classification

The propose of sentimental analysis of users reviews is to provide a new recommend reference for users to choose the restaurant by their preference. Some users may prefer to go to the best ambiance restaurant with friends, the others may like to go to restaurant with great food. These information is hard to obtain from the star rating. Thus, the users reviews could be a very significant source to extract information. In this project, the review score we produced totally stem from the users reviews and be independent with the score of users star rating. The tool we used in this project for sentimental analysis is VADER (Valence Aware Dictionary and Sentiment Reasoner). Vader is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media, and works well on texts from other domains. There are few methods to analyze positive/ negative emotion of a sentence. The easiest way is to use Naive Bayes, and provide a suitable positive and negative training dataset for feature extraction. When, however, people doesnt have sufficient dataset, lexicon based sentiment analysis will be the best way to classify a bunch of un-label sentences. The main steps of this technique are shown in the Fig1.

The Vader Analyzer is able to evaluate a sentiment score for every sentence and provide high precision to reflect users emotion. Thus, this sentiment scores for every restaurants reviews can be useful reference to other users in future. Furthermore, if user need more restaurant information like food, service and atmosphere, we can extract these features from every sentence and gather the statistic of scores in order to give a fair ranking of all the restaurants.

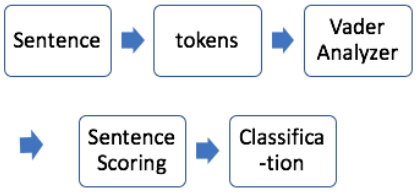


Figure 1. Result of Recommendation System

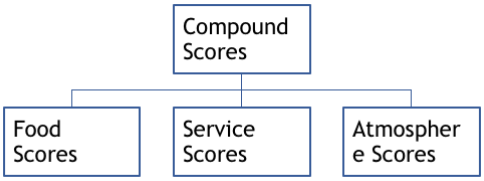


Figure 2. Result of Recommendation System

### 3.5. Visualization

We want to create user-friendly GUI for users based on Python. It must be able to display the location for current users and 10 ten recommended restaurants on Google map. As Google provide only UI for Web service. We need to find another solution.

First we found an open source code from Github called GooMPy, however, it does not have functions for adding annotations to map through Googles API. So we modified the source code and successfully presents retrieved result on static google map as Fig1.

GooMPy provides a Python interface to the Google Static Maps API (https://github.com/simondlevy/GooMPy ). It provides a Python interface to the Google Static Maps API witch automatically download and stitch together map tiles into a single image that you can zoom and pan dynamically. After we retrieved all information of recommended restaurant including id, name, coordinate, score, we will display it on GUI as final step of our project.

## 4. Experiment

For the experiment, we select 3 customers, and corresponding location. The output including average distance
**Input format:** python dirctions.py [user2Id] [coordinate1] [user2Id] [coordinate2] ...
**Output format:** [distance]: distance between this restaurant and furthest user [name]: restaurant name [star]: ALS rating [nlp_score]: restaurant score calculate by review analysis using NLP
**Experiment Output** The ouput of this experiment is shown in Fig3 and Fig4.

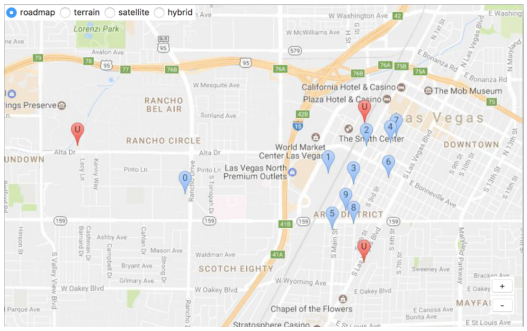


Figure 3. Visualization of the Result of Experiment

```
distance: 3246  name: Anthony & Marios Broadway Pizzeria  star: 3.80401674906
nlp_score:  3.17689302981 resturant ID: KdJJFvYI9w5xF0EVYPbXMA

distance: 3504  name: Slidin' Thru  star: 3.70620926221
nlp_score:  3.14820883159 resturant ID: LnCbpxipUrA165zvfaePGw

distance: 3561  name: MTO Café  star: 4.5
nlp_score:  3.30124281444 resturant ID: d4bXnfbUc1SLu3jVcr-w1w

distance: 3800  name: El Sombrero Mexican Bistro  star: 4.5
nlp_score:  3.20968788981 resturant ID: 9eYvD9isKnk1TvZ5LX5C1Q

distance: 3973  name: Espression Cafe  star: 4.0
nlp_score:  3.18308360656 resturant ID: spgzzx5RbKhBvAlZLAxo_A

distance: 4008  name: Makers & Finders Coffee  star: 4.5
nlp_score:  3.30683186027 resturant ID: sphMxGkYY5OR7uleFxk_WQ

distance: 4020  name: Le Pho Vietnamese Kitchen  star: 4.0
nlp_score:  3.16589340102 resturant ID: u5bGaiZGosBf-L9lYLCg5g

distance: 4078  name: Anthony's New York Pizza & Deli  star: 4.0
nlp_score:  3.12287916667 resturant ID: HPDhf68IVvvO7nX0iUVlxg

distance: 4162  name: Rock'N'oodles  star: 4.5
nlp_score:  3.23591534392 resturant ID: 6OJV—BjqPpJjnTueniqVw

distance: 4171  name: Mingo Kitchen & Lounge  star: 4.0
nlp_score:  3.25919479554 resturant ID: IfaLsdrW6b5u-6yFHLVjhA
```

Figure 4. Result of Experiment in Console

## References

[1] Yelp Dataset. https://www.yelp.com/dataset_challenge

[2] Google Maps Distance Matrix API. https://developers.google.com/maps/documentation/distance-matrix/intro

[3] GooMPy: Python interface to the Google Static Maps API. https://github.com/simondlevy/GooMPy

[4] Hutto, Clayton J and Gilbert, Eric, *Vader: A parsimonious rule-based model for sentiment analysis of social media text*. Eighth International AAAI Conference on Weblogs and Social Media. 2014.