Analysis of Geo-tagged Yelp Data

CS562: Advanced Database Systems - Spring 2018

Final Report

https://github.com/ZhiyuWang95/CS562_Geo-tagged_data

https://drive.google.com/drive/folders/0AAHUTO87o0myUk9PVA

Chunxi Wang
Boston University
tracycxw@bu.edu

Zhiyu Wang
Boston University
wangzy95@bu.edu

Chuan Xing Zheng
Boston University
czheng78@bu.edu

May 3, 2018

1. Abstract

Today, one of the most influential leading consumer review sites is Yelp. Yelp contains a large dataset that computer scientists can use to analyze the social relationships and trends between the star rating (from 1 to 5) and reviews, the number of reviews, the type of business, and the number of customers. Our plan is to store, query, and analyze Yelp's large dataset. The main purpose of this project is to use various data analysis and mining tools to attain optimal accuracy of classifying and summarizing the data. Since the Yelp dataset only contain 10 cities around the world, we decided to use Las Vegas as our main city to do the analysis. Optimally, we are finding interesting patterns between the type of restaurant and the star rating of the restaurants in Las Vegas. The data analysis tools we used in this project were K-means++ cluster, Dynamic Time Warping, Singular-Value Decomposition, Latent Semantic Analysis and Random Forest Classifier.

2. Introduction

Nowadays, people travel differently than they use to do. Modern travelers use several geo-social networks to figure out where they may want to go or eat. They want to "understand the feel of the area" and to "live like a local", instead of just "seeing the tourist sights". As the population in cities rises in the 21st century, people will want to know which neighborhood they would feel at home in and business owners will need to know where to expand their market. Applications like Facebook, Instagram, Twitter, Foursquare, and Yelp have turned photos sharing from an atypical practice into a common one. People can tag their photos, tweets, reviews, ratings, and other kind of posts to real-world locations. In our project, we are looking for patterns in these reviews and ratings, all tied to locations and time. Specifically, we chose the Yelp dataset so that we hope to find patterns that could help travelers understand the restaurants they are planning to go to. In this project, we detailed a research through design exploration into questions, which is listed below.

2.1. Problem Definitions

- Where are the highest rated restaurants located in Las Vegas? We want to cluster the restaurants by the geo-tagged data provided in the Yelp dataset, and calculate the average star of each cluster.
- Can we predict the star rating if given some feature of the restaurant?
- What is the relationship between locations and the star ratings, review amounts and other attributes in the data-set?
- What patterns can we visualize for time versus the evolution of Yelp rating of the restaurants in Las Vegas?
- What can text reviews tell us about the restaurants in Las Vegas? Our goal here is to cluster the text reviews of the restaurants and extracting categories in an unsupervised fashion.

3. Data Sets Used for Evaluation

The data-sets we retrieved and worked with is the Yelp data-set and were downloaded from from the following link: *https://www.yelp.com/dataset*. This data has been made available by Yelp for the purpose of

the Yelp Dataset Challenge. Specifically, the challenge dataset contains the following data:

- 481K business attributes, e.g., hours, parking availability, ambience
- 1.6M reviews and 500K tips by 366K users for 61K businesses
- Aggregated check-ins over time for each of the 61K businesses

The dataset only contain 10 cities around the world:

- U.S.: Pittsburgh, Charlotte, Urbana-Champaign, Phoenix, Las Vegas, Madison
- Canada: Montreal and Waterloo
- Germany: Karlsruhe
- U.K.: Edinburgh

4. Methods & Results

4.1. Relationship between Ratings and Location

We first simply extracted all the restaurant data from the *business.json* to DataFrames in Python. The DataFrames consisted of business_id, stars, review_count, latitude, longitude, neighbourhood and category as attributes. We clustered the restaurants in Las Vegas into eight clusters by using the K-Means++ clustering algorithm to group restaurants based on rating of all the restaurants and their corresponding latitude and longitude [Fig. 1]. We then calculated the average star rating of the restaurants in each cluster to find if the distribution would be interesting.

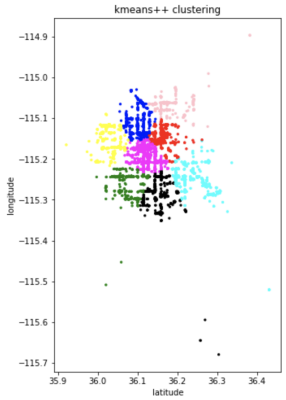


Figure 1. A K-means++ clustering plot of the restaurants in Las Vegas based on star ratings and the restaurants' locations (longitude and latitude). The average star rating for each cluster or color: Blue = 3.33, Green = 3.56, Red = 3.35, Cyan = 3.19, Magenta = 3.60, Yellow = 3.46, Black = 3.56, Pink = 3.49

After the clustering of the restaurants, we also collect data from *review.json*. The x coordinate is the stars level, and the y coordinate is amount of reviews. Each point in the graph represents one restaurant. From this tendency, in the first 500 restaurants, stars between 3.0 and 4.0 got more reviews from costumers.

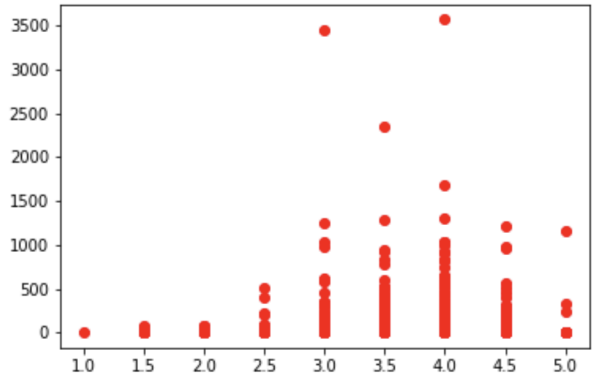


Figure 2. A plot that shows the relationship between the star rating and review counts for 500 restaurants. (X-axis is the star ratings and Y-axis is the review counts)

Then we extended the analysis range from 500 restaurants to 2000 restaurants. From the statistical graph Figure 3, 4.0 star restaurants got the most reviews than others. And 1.0 star restaurants got the least reviews.

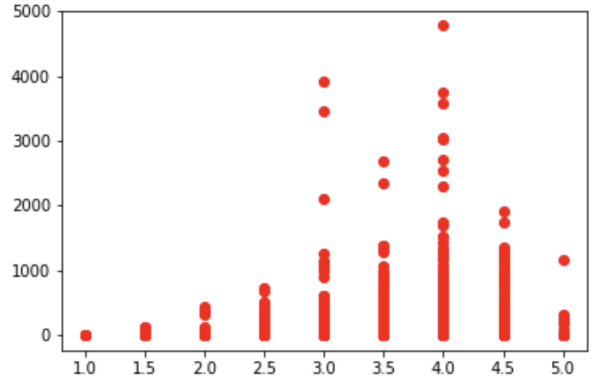


Figure 3. A plot that shows the relationship between the star ratings and review counts for 2000 restaurants.(X-axis is the star ratings and Y-axis is the review counts)

Last, we analyzed the relationship between restaurant stars and its distance to Las Vegas downtown. The result of this analysis provides the suggestion about where to find the higher stared restaurants. First, we got the latitude and longitude of Las Vegas downtown from *https* : *//www.latlong.net/*, which is $(36.114647, -115.172813)$. Because in the

Business.json, every restaurant record provides the location of the restaurant, by calculating the L2 distance between downtown point and restaurant location, we could get the Euclidean distance from each restaurant to downtown. From the graph, Figure 4, it is obvious that the restaurants which are 10 miles from downtown got highest stars, about 3.53. On the opposite, restaurants which are 5 miles from downtown got lowest stars, about 3.47. This tendency shows if we want to find higher stared restaurants, we'd better go to the areas where 10 miles from downtown.

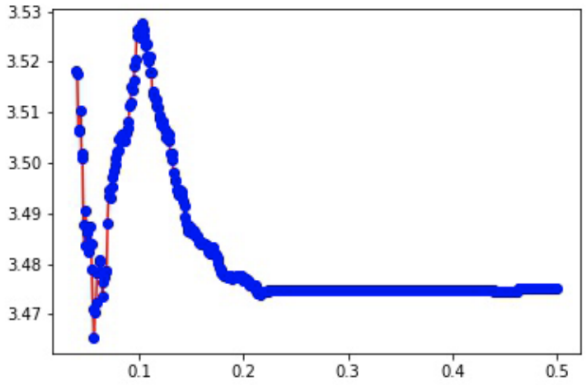


Figure 4. A plot that shows the relationship between the distance away from Downtown and the average star rating for each restaurant. (X-axis is the distance away from Downtown (unit: 100 mile) and Y-axis is the average star rating.)

4.2. Star prediction based on location

In 4.1, according to the experiment about relationship between ratings and locations, we found that the average rating has this tendency: the restaurants located about 10 miles from Downtown got the highest average stars. Therefore, now we already know where to find the good restaurants in Las Vegase. However, we were more curious on where to open a restaurant and what kind of restaurant to open to receive higher ratings.

First, we split into training and test sets. Due to the worry that only latitude and longitude may not be enough to make reliable prediction, we also included the price-range and the opening time of the restaurant as another 2 attributes into our input vector. In which, one input instance should be a four dimensional vector, which is like $[36.0669136, -115.1708484, 20, '' 8:30-22:30'']$.

It was obvious that using all these vectors as training data, we needed to relabel the open time with integer values. We mapped the opening time with integer number labels from 0 to 357, as shown below.

```
8:30-22:30 0
11:00-0:00 1
0:00-0:00 2
10:00-16:00 3
8:00-0:00 4
11:00-21:00 5
11:00-22:00 6
7:00-2:00 7
10:00-23:00 8
9:00-0:00 9
8:00-22:00 10
5:00-23:00 11
10:00-22:00 12
11:00-20:00 13
11:00-21:30 14
```

Figure 5. label all the open time

Now we have collected the data we are interested. The second step were building learning module and trained them. However, what needs to be mentioned is there are only 2775 restaurant records provide all 4 information. In this manner, the shortage of the data limited the performance of the learning module.

At the beginning, I built a neural network with just one hidden layer. There were 4 nodes in the input layer, 5 nodes in the hidden layer and 5 nodes in the output layer. Sigmoid function were used as a activation function and the training procedure finished with 20000s step and 0.1 learning rate. 2,575 records were used as training data and 200 records were used as testing data. The performance of the neural network was not good, every star prediction was almost around $3(\pm 0.15)$.

After consulting our professor, George Kollios, we decided to try Random Forest on this dataset. *Sklearn* library provided a set of built-in functions to implement the Random Forest classifier and so we didn't build the module. I loaded the data into built-in function, *RandomForestClassifier()* and linked outputs. By setting the error to 1, the random forest could get an accuracy of 0.7 in 200 size test set.

```
#accuracy print.
print("the size of the test set is", len(test))
print(count,"predictions out of ", len(test),"test are correct.")
print("So, the accuracy is", count/len(test))
```

```
the size of the test set is 200
140 predictions out of  200 test are correct.
So, the accuracy is 0.7
```

Figure 6. prediction performance of random forest

Obviously, 0.7 prediction accuracy was still not high enough. In order to improve the performance of the module, we thought of several future works. First, to make the neural network feasible, we need bigger dataset. Only 2,775 records were not enough to train a reliable neural network classifier. Second, to improve the accuracy of the module, more features should be selected. In our current module, only 4 features (latitude, longitude, price, Sunday open time) were taken

into consideration. To have better prediction, load in more relevant features is unavoidable. For example, the type of food should be used, but we didn't find an effective way to label the type of food of a restaurant. This is because most restaurants are pretty versatile; one restaurant could be labeled with Chinese, Japanese and even New American food at the same time. If we can label the food type effectively and train the module with bigger data-set, then the performance of the module should improve greatly.

4.3. Analysis of Restaurant Reviews with LSA

We clustered restaurants based on the text reviews from the *review.json* file and extracted categories in an unsupervised fashion. First, we create a vector list in Python to store the business_id with its three features and we extracted from both the business.json and review.json data. The three features we extracted are the latitude, longitude, and all the reviews for all the restaurants in Las Vegas. Next, we used the Latent Semantic Analysis (LSA) on the text reviews to cluster the restaurants. We took into account the first 1,000 reviews per restaurant and computed the document-term matrix, so that we could apply the principle component analysis (PCA) to it. We ignored the terms that appeared in more than 70% and less than 20% of the documents. Then, we used the Python library called *sklearn.decomposition.TruncatedSVD()* to sparse the data. In order to improve the clustering accuracy, we excluded the less significant concepts from the documents' feature vectors. At the end, we used 6 as our principle component as shown in Figure 7.

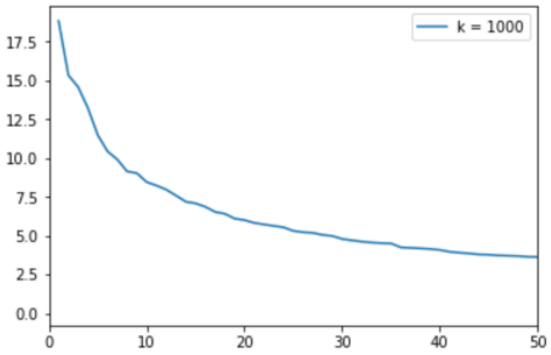


Figure 7. A plot that shows that 8 is the reasonable principle component.

Finally, we visualized the clusters by plotting the longitude and latitude of the restaurants in a scatter plot and labeled each cluster. The clustering algorithm we used was K-Means++ and used the Silhouette Coefficient to find the best score that relates to a model with "better defined" clusters [Fig. 8]. From the Silhouette Score plot we saw that six is a local maximum in the Silhouette Coefficient as the potential number of clusters for our data. We calculated the frequency of terms in each cluster and outputted the five most frequent words in the reviews the restaurants received from their customers to represent the six clusters [Fig. 8].

Figure 8. A Silhouette Score plot shows that the best Silhouette Coefficient is 5.

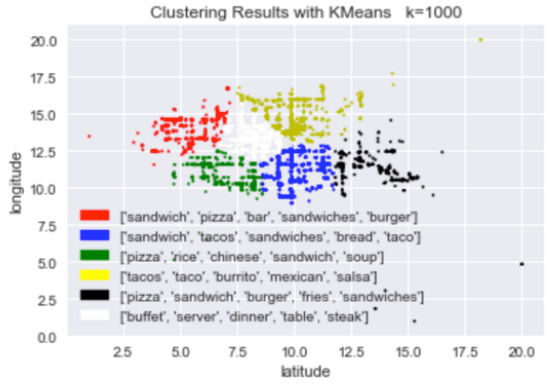


Figure 9. A K-Means++ clustering plot of restaurants based on the reviews that they received.

4.4. Analysis of Restaurant Reviews with Query

Based on the LSA method above, we can get a restaurant-terms matrix with 5,899 rows and 1,711 columns. After we transformed it with Singular Value Decomposition (SVD), we converted the original matrix to 3 matrices (U, S and V^T). U is a restaurant-topic matrix, which tells us how strongly related restaurant i is to the topic represented by semantic dimension and any 2 distinct row vectors are orthogonal to each other. S is a square, diagonal matrix and the diagonal consists of the singular values. The magnitude of the singular value measures the importance of the corresponding semantic dimension. V^T is a topic-

term matrix. Any two distinct column vectors are orthogonal to each other. Each number in the matrix indicates how strongly related terms i is to the topic represented by semantic dimension j. Since values in S is in descending order, we just keep the most significant 50 topics($k = 50$). Then, set all other values to 0 to decrease noise and analyzed the restaurant reviews. We came up with 4 queries with 3 terms in each query:

- Query1 = ['music','rock','drink']
- Query2 = ['pizza', 'bar', 'quiet']
- Query3 = ['bar', 'sushi', 'expensive']
- Query4 = ['chocolate', 'donut', 'cheap']

For each query, we calculated the distance between each query and each restaurant with cosine similarity method. The smaller the values are, the more similar they are. Then for each query, we achieve 20 most similar restaurants and then plot them based on their latitudes and longitude [Fig. 10].

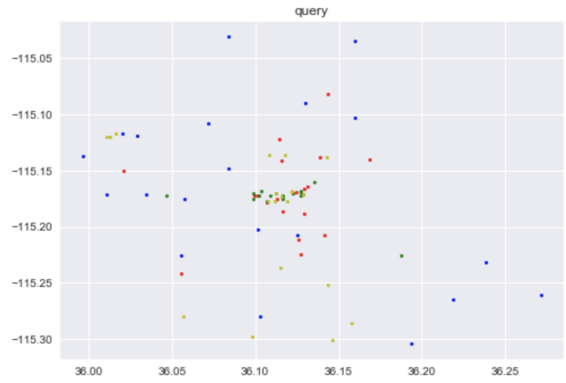


Figure 10. Most admissible restaurants retrieved from our 4 queries.

In Figure 10, the red dots represent query1, blue dots represent query2, green dots represent query3, yellow dot represents query4. Noticeably, the green dots gather around Downtown more and whereas the blue dots scatters everywhere. Hence, we acquired the trend from this figure bars which serve expensive sushi are likely to gather in Downtown and quiet bars which serve pizza are seen everywhere in Las Vegas. The reason for this is because area near Downtown has a lot of hotels and great facilities, so most travelers and millionaires prefer to go there to have fun. Since people wandering near Downtown are possibly richer, more restaurants chose to boost the price to make profit. For foods like pizza, they are cheap and delicious, so the needs for them doesn't vary based on location. Therefore, we see bars with pizza in many places in the city.

With the restaurant-terms matrix, if given a query, we can easily calculate the distance between the query and each restaurants. After sorting the distances in descending way, we can return good matches along with its information like business id, latitude, longitude, etc. This method will be really helpful to provide suggestions for people who want to find a specific place to eat.

4.5. Time Series of Star Rating with DTW

Secondly, we visualized the time series of average star rating for the restaurants in Las Vegas. We extracted the businesses from the *review.json* file because this json file contains the date of when the users rate a restaurant. Then, we parsed out all the restaurant reviews and extracted the important features we want to analyze, e.g. stars, date, business_id, review_count, and categories. We ended up with 92,9606 restaurant reviews. Since we are looking for interesting trends of time versus stars rating, we extracted only the reviews that have stars and the date they were given to the restaurant. In addition, 92,9606 of reviews is a big number and so we only took the restaurants with at least 4,000 reviews, which we did not consider restaurants with small review counts. In the end, we ended up with only seven restaurants with at least 4,000 reviews and plotted their time series of average star rating from 2005 to 2017 [Fig. 11]. Then, we picked the two restaurants with the same amount of years. We used dynamic time warping (DTW) with the Euclidean distance to find similarity between the two temporal sequences [Fig. 12, 13].

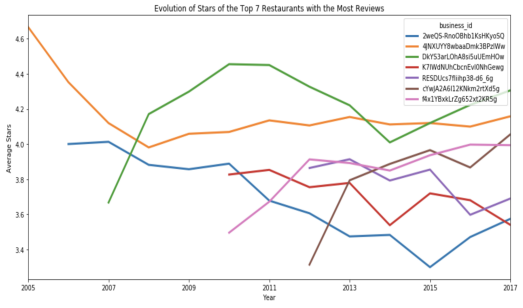


Figure 11. A plot that shows the evolution of stars received from 2005 to 2017 for the 7 restaurants in Las Vegas that received more than 4000 reviews.

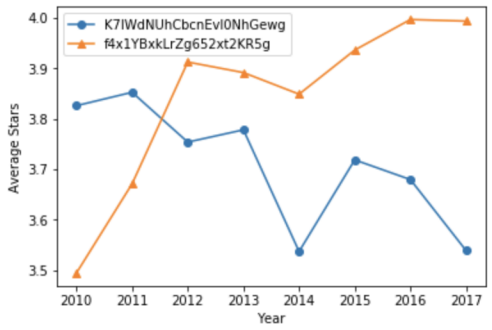


Figure 12. A plot that shows the evolution of stars received from 2010 to 2017 for two time sequences.

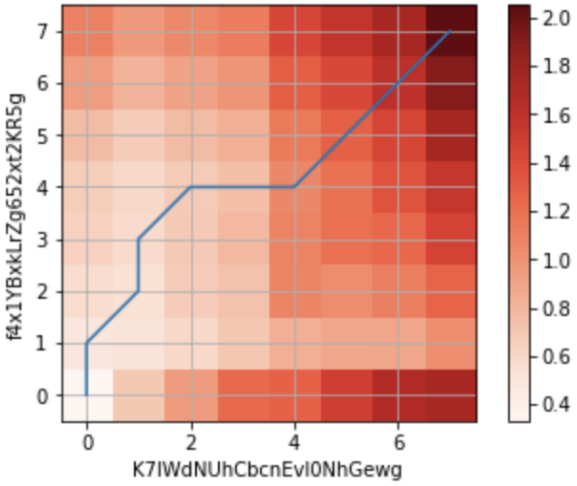


Figure 13. A plot shows the optimum warping path, which minimizes the sum of DTW distance along the path and incorporates the DTW distance between the two restaurants in Figure 6.

4.6. Annual Recurrences

In this section, we examined the yearly patterns, or people monotonous behavior all throughout the year. For example, we took the means of the star ratings for every month and plot them on a scatter plot [Fig. 14]. We can see that there are higher star ratings in February compared to all other early months in a year. The reasons behind this might be because there are more holidays around this time of the year (e.g. Birthdays and Valentine's Day). Surprisingly, the ratings are not high during the summer months (June, July, August). We were expecting higher ratings during this time of the year because people tend to go out more. On the contrast, the highest rating received is during the Fall season (September and October). And the lowest rating received is during the month December.

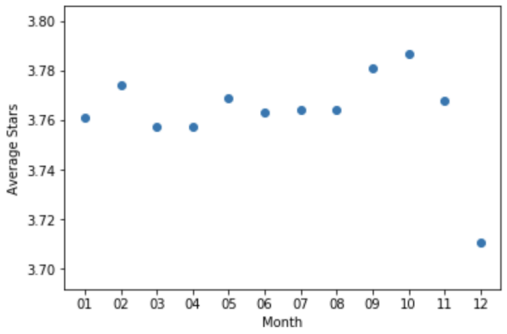
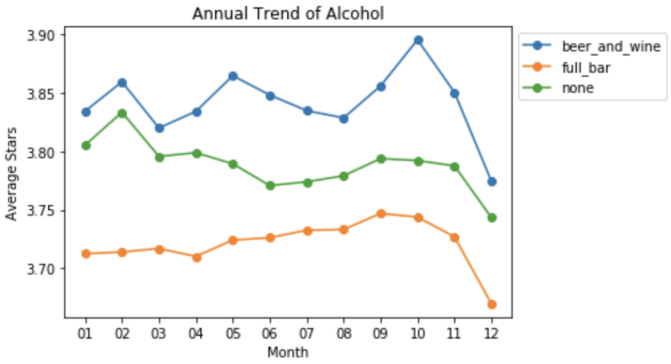
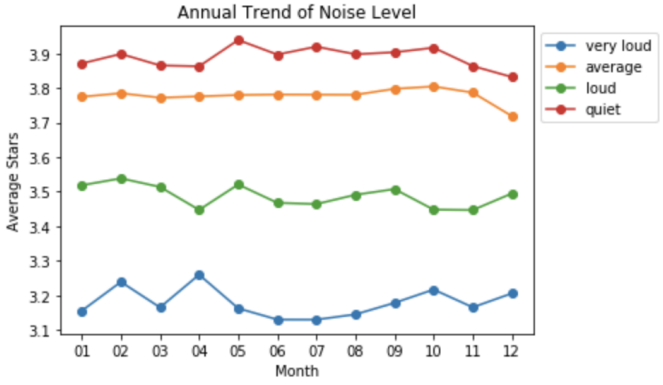


Figure 14. Average Ratings per Month

Figure 15 shows how the 3 different features (Alcohol, Noise Level, and WiFi) were rated across the months of the year. In common, the differences are not noteworthy, in any case, we were able to form some interesting perceptions. For example, people tend to give higher ratings to restaurants that serve alcohol in winter more than summer. And they seem to dislike very loud restaurants in summer more than any other seasons.





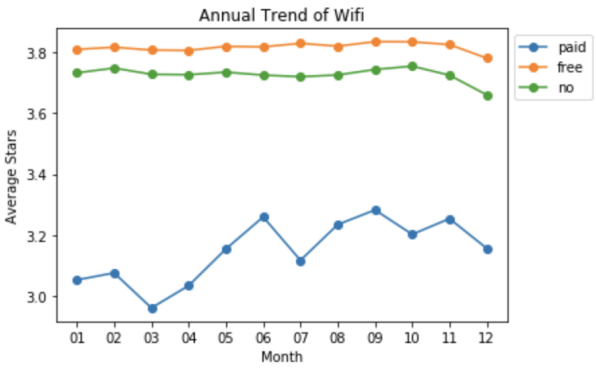
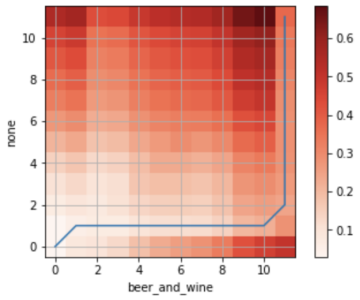


Figure 15. Plots that shows how various feature are rated annually.

Unfortunately, we cannot inference a lot from the the plots shown above. We have attempted to form the same plots for all other features, apparently people are changing. It is probably because in the early days when Yelp website was made there were not enough reviews, so one outlier made a big impact on the overall mean. However, an interesting observation to point out is that people tend to review restaurants that serves alcohol more over time. In essence, we wanted to find similarity between the 3 alcohol features (beer_and_wine, full_bar, and none). We used DTW with Euclidean distance again to measure the similarity between the temporal sequences of the alcohol features [Fig. 16]. We wanted to compare the "beer_and_wine" feature with the other 2 features because restaurants that have the attribute as "beer_and_wine" get higher ratings throughout the year, which is shown in Figure 16. From the DTW plots shown below, interestingly, the optimal warping path between the "beer_and_wine" and "none" sequences are monotonously similar throughout the year until when it gets to the very end of the year. On the other end, the optimal warping path is a straight diagonal line between "beer_and_wine" and "full_bar". In this case, there is no close relationship between the two temporal sequences, which says the trend of ratings received for restaurants have "full_bar" are not closely related to the trend of ratings received for restaurants with "beer_and_wine" as attribute.



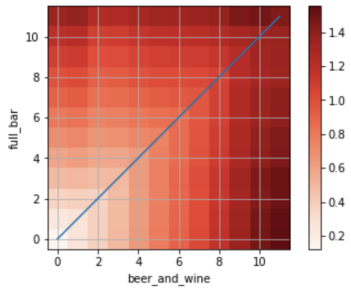


Figure 16. DTW plots that shows how various feature are related and they consist of the warping paths.

5. Conclusion

In summary, we have found many interesting patterns and trends from the yelp dataset.

First, we get use K-means++ to cluster the restaurants into 8 parts, each cluster has different average rating star. We also get the relationship between average star rating and the distance away from downtown. We built two machine learning modules to predict the star of a restaurant based on its location, price range and open time. One module used Neural network, the other is based on random forest. Because only 2,775 records are provided, neural network can not provide effective result, and random forest could make 70 percent accurate prediction.

Second, we utilize LSA to analyze restaurant reviews. For labeling each cluster, we calculate the frequency of terms in each cluster and choose the five most frequent words in reviews as label. After transforming the restaurant-terms matrix by SVD, we can retrieve the most similar restaurants according to our query by calculating the distance. We get some interesting patterns by testing some queries. For example, expensive bars which serve sushi can often be found near downtown while quiet bar which serve pizza can be found everywhere in Las Vegas. This model is really useful for answer the query from people who want to find some specific place to eat.

At last, the temporal sequences were used with the Dynamic Time Warping method and we didn't see any major patterns for the evolution of star ratings between the 7 restaurants in Las Vegas that received more than 4,000 reviews from 2005 to 2017. Moreover, our DTW plot show that the optimum warping path did not show a close relationship between them, specifically for the 2 restaurants that consist with the same amount of years. Subsequently, we used DTW to analyze the annual temporal sequences for restaurants that serve alcohol. The 3 features that we compared in the alcohol attributes were "beer_and_wine", "full_bar", and "none". The stars received by the restaurants with the feature

"beer_and_wine" is most similar to the feature "none". On contrary, it is not similar to the feature "full_bar". This was an outcome we didn't expect because we thought restaurants that serve "beer_and_wine" should be similar to restaurants that serve "full_bar"; you can have beer and wine in full bar too.

6. Sources

- https://simonpaarlberg.com/post/latent-semantic-analyses/
- http://repository.cmu.edu/cgi/viewcontent.cgi?article=2022&context=dissertations
- https://cseweb.ucsd.edu/jmcauley/cse255/reports/wi15/Wa'el_Farhan.pdf
- http://cs229.stanford.edu/proj2015/301_report.pdf
- http://scikit-learn.org/stable/
- http://www.numpy.org