User-Customized Restaurant Recommendation using Natural Language Processing on Yelp Data

Somdut Roy*
Georgia Institute of Technology
Atlanta, Georgia, USA
somdut.roy@gatech.edu

Devanshee Shah
Georgia Institute of Technology
Atlanta, Georgia, USA
dshah330@gatech.edu

Vitaly V. Marin
Georgia Institute of Technology
Atlanta, Georgia, USA
vmarin3@gatech.edu

ABSTRACT

Using machine learning tools for natural language processing (NLP) has been prevalent in different avenues across the globe. With numerous sources of available data in textual format, it would be interesting to leverage such information to gain useful insights. Yelp data is one such readily available and insightful data avenue. This study uses a set of a reviews and the corresponding ratings for a restaurant for one city and creates a rating prediction tool using i) Linear SVM and ii) Logistic Regression. At the same time, the provided ratings are used to recommend relevant restaurants to a user mentioned his/her favourite restaurant. Considering Computation Time and Accuracy for both the models, logistic regression was chosen over SVM. Finally, rating prediction and Item based collaborative filtering were merged to devise a tool to recommend a said user, who has a favorite restaurant, with a list of restaurants.

ACM Reference Format:
Somdut Roy, Devanshee Shah, and Vitaly V. Marin. 2018. User-Customized Restaurant Recommendation using Natural Language Processing on Yelp Data. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/1122445.1122456

1 DATA DISCOVERY AND PROPOSED METHODOLOGY

In this section, we discuss the rationale behind choosing a particular data-set, the process of gathering initial insights on the data and detailed description of the proposed methodology.

1.1 Rationale behind choosing particular dataset

We found good amount of organized Yelp data in Kaggle in JSON format [2]. It had detailed Yelp reviews for businesses including but not limited to restaurants from all users from 2005 to 2018 in different cities across the country. Out of all businesses, restaurants had the biggest fraction of reviews. Therefore, from a JSON file of 6 gigabytes size, only businesses tagged as "restaurant" were extracted out. Our aim in this study was to possibly explore different avenues in NLP without excessive computation load. At the same time, having a really small data could make for a very limited and non-generalized study. So, we decided to choose a city with 10k-20k data-points. Avondale, AZ turned out to be one of them. The final extracted CSV file with the relevant information is 7 megabytes in size.

1.2 Data Discovery

In our finalised dataset, there are seven relevant attributes : Review ID, User ID, Business ID, Restaurant Name, Stars, Text Reviews, Date. After filtering out data for restaurants in Avondale, AZ with reviews and ratings, as shown in the Table 1, there are 12662 reviews by 8031 unique users for 163 unique restaurants. Text Review is used for sentiment analysis; rating is the ground truth label for measuring the accuracy of the model as well as used to find the cosine similarities among restaurants; business id and user id served as the key for data wrangling.

Table 1: Information on Our Dataset- Restaurants' Reviews of Avondale, AZ

Number of Reviews	Features Included	Number of Unique users	Number of Unique Restaurants
12662	reviewID, userID, businessID, restaurant name, stars, Text Reviews, Date	8031	163

In this section, Some of the interesting features of our data-set are presented. Figure 1 shows distinct feature - 20 most reviewed restaurants and their counts. *Flavors of Louisiana* is the most reviewed restaurant in Avondale with more than 700 reviews. In another feature, Figure 2(a) shows its ratings over the years.

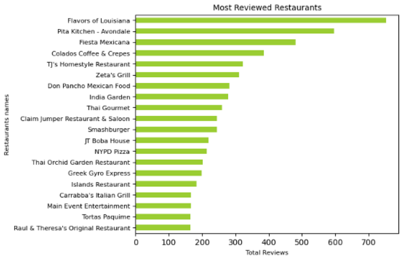


Figure 1: Twenty Most Reviewed Restaurants in Avondale

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

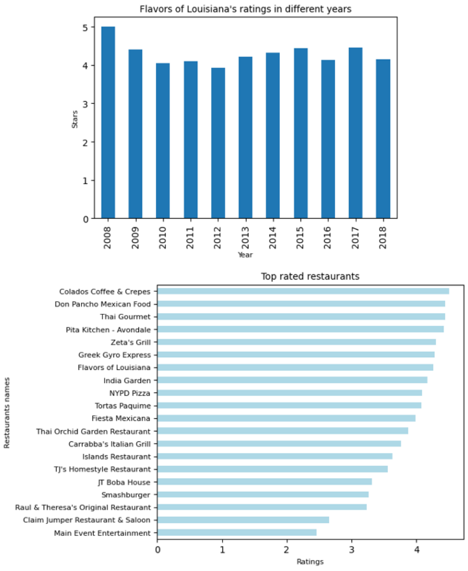


Figure 2: (a) Ratings of "Flavors of Louisiana" over the years (top), (b) Twenty Top Rated Restaurants in Avondale (bottom)

Table 2: Star Distribution of Restaurants Ratings

Stars	1.0	2.0	3.0	4.0	5.0
Distribution in Percentage	16.2	9.8	11.0	21.6	41.4

Figure 2 (b) represents twenty top rated restaurants and their average ratings. *Colados Coffee and Crepes* is the top-rated restaurant with around 4.7 average stars. It is interesting to know that the star ratings (out of 5) for the restaurant reviews are not uniformly distributed. As shown in Table 2, about 60 percentage of these reviews rate the corresponding restaurants very highly (at least 4 stars); the other classes are smaller. Moreover, distribution of rating frequency of all the restaurants as well as the distribution of frequency of ratings given by the users follow the long tail property. In case of restaurants, there are very few restaurants that are rated frequently - called popular restaurants. In figure 3(a), it can be seen that only 40 restaurants are rated more than 100 times. Similarly, Rating Frequency per user is also plotted in Figure 3(b). It can be seen that very less users are interested in rating restaurants.

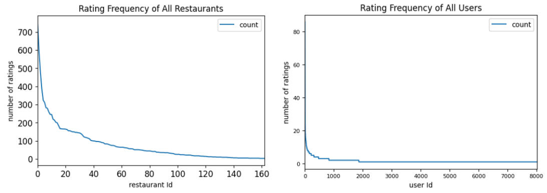


Figure 3: Rating Distributions (a) Rating Frequency against Restaurants (left), (b) Rating Frequency per User (right)

1.3 Big Picture

As discussed before, the big picture of this study involves performing sentiment analysis on all the reviews on one hand and building an item-based collaborative filtering model to recommend restaurants based on favourite restaurant on the other. Eventually, the two aspects are to be merged together into one platform where a given user with a given "favourite restaurant" will get a list of restaurants along with predicted ratings based on the built model. Hence, the entire procedure can be broadly classified into three subsections.

1.4 Methodology: Baseline Discovery and High-Level Work Flow Description

If we could simply use reviews to predict ratings of a restaurants, that would be relatively simple. A study by Yu et al. (2017) used "bag of words" approach for the reviews and then did sentiment analysis using SVM [1]. However, the scenario here is relatively more complicated. Since we will be recommending a user some restaurant which he/she has not paid to visit to or has written a review about beforehand, there would not be a review to implement a rating-prediction tool on. Hence, we will try to create user and restaurant "profiles" based off of all the reviews made by an user or all the reviews placed under a restaurant respectively. Then we intend to form a vector representing both the user and the restaurant profile to represent the event of a given user visiting a given restaurant.

An attempt of combining sentiment analysis with item-based collaborative filtering was done by Jayashree and Kulkarni (2017). They combined (1) sentiment analysis by SVM with (2) item based collaborative filtering using kNN and Naive Bayes'. For our current study we use this method to create one baseline. Since the paper did not mention anything about the way the reviews were converted into training matrix, we take the liberty of experimenting with different techniques learnt in class (in the subsection that follows) and choose the best performing technique as baseline. Going by a finding in the study performed by Yu et al. (2017), that advises use of linear-based classifier like SVM or Logistic Regression for sentiment analysis to yield maximum accuracy [3], our study limits itself to exploring SVM with "linear" kernel and Logistic Regression for the sentiment analysis segment and discussing findings with respect to different aspects of performance.

1.4.1 Baseline Discovery. Before creating a baseline, we have to look at different word embedding techniques to represent the reviews. For that purpose, it is essential to explore different ways and discard methods that provide less encouraging accuracy results at this stage. We initially start with using "bag of centroids" approach on word embedding that we learn to use in Assignment 2 of this course. Instead of using word2vec, however we use *FastText*, an open-source library provided by Facebook AI lab [4] because it is believed to retain slightly better syntactical information and perform better for a small data-set like ours [5]. Firstly, we use all the words available in the review list without removing the stopwords to be trained using *FastText*. That creates an array of vectors with each vector representing words in the model vocabulary. Applying k-means clustering on the array of vectors and using elbow method (figure below), we find out that using 10 clusters would be the optimal for this data-set.

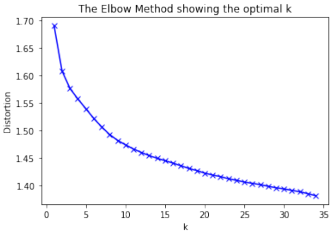


Figure 4: Elbow method with point of inflection around k=10

Using 10 clusters, each of the reviews in the data-set was represented as a bag of centroids where it provides a counter for the number of times the words in the reviews feature in a particular cluster. Once we form the training matrix X, we split that into 80-20 to make dummy train and test sets. For cross-validation, we split the train set data in 80:20 ratio to optimize the regularization parameter C using "linear" kernel for SVM. The values for C are chosen at random ranging from E-4 to E4. For this case, the model is expected to perform best for C=1.19. It must be mentioned that to maintain consistency and retain a scope for a fair comparison, we have same

(1) Train-test split procedure (80:20)
(2) Fixed k-fold cross validation with k=5
(3) Usage of Google Collab in the same machine with minimal interference with other processes.

procedures for test-train-split, cross-validation and hyper-parameter optimization throughout the rest of this study. The test accuracy of the resultant model turned out to be 41.1 percent. While it is better than a random classifier which would have a 20 percent probability of classifying reviews (ratings 1-5) correctly, it is essential that we explore other baselines that would yield better performance. We try using logistic regression with tuning the "Inverse of regularization strength" parameter C in it. This process yielded a test accuracy of 41.4 percent. With respect to accuracy, that was not a significant jump, however it is worth mentioning that the run-time was observed with hyper-parameter tuning process, 30 iterations of Linear SVM (300.46 seconds) took almost 4 times of what same number of Logistic Regression (82.06 seconds) iterations took to compute.

In quest for better performance, it is essential to explore other ways to represent texts. In a blog posted by nadbor (2016), the arithmetic mean of word2vec vector representations of the words in a piece of text is used to represent the said piece of text [6]. In our data-set, we can take the mean of the words in the reviews to create vector representations of the reviews. When we train the resulting matrix for ratings using Linear SVM, the optimum C value was found to be 3.22 and the corresponding F1 score turned out to be 53.2 percent. The test score here was 55.3 percent. That was a significant jump over the accuracy returns from "bag of centroids" approach. This could be because of the fact that bag of centroids generalizes words to one of ten centroids. This causes generalization in review representation and depletes the inherent domain knowledge. In addition, we train the matrix using Logistic Regression and get F1 score of 56.1 percent and test accuracy of 58.3 percent at significantly reduced run-time.

1.4.2 High-Level Work Flow Description. As the Mean-Vector representation showed better performance for rating prediction, we will proceed forward with that technique for creating matrix for rating prediction training. Since a user will be recommended restaurants where he/she has not visited beforehand, we will not have prior information about any review for the user-restaurant pair. So, the plan is to use all the reviews made by a user and create mean-vector representation of that to represent an interpretation of "user profile". In a similar manner, we take all the reviews placed against a restaurant, create mean-vector representation of that and call that the "restaurant profile". Then instead of using the provided reviews, if user U with "user profile" vector V_U makes a Review represented with vector r in a restaurant R which has a "restaurant profile" vector of V_R, instead of using the review r, we use the appended vector $[V_U V_R]$. We call it the association vector ($Assoc(U, R)$) between a user and a restaurant. This provides us with a luxury of associating any user with any restaurant that the user may not have any visit to before.

$Assoc(U, R) = [V_U V_R]$
$Review(U_i, R_j : Rating_{i,j} —> Assoc(U_i, R_j) : Ratingi, j$

As we train this *Assoc* matrix to predict ratings, the model becomes capable of predicting rating for a given user in a given restaurant. In parallel, item-based collaborative filtering is done using kNN based solely on the ratings following the methodology explained in the GitHub repository of $KevinLiao$ [7]. As we follow the method explained, we find the cosine similarities among restaurants using the ratings given by the users to the restaurants. Additionally, we use fuzzy string matching to find the closest restaurant in our dataset with the "favourite restaurant" given in the query. By finding the 20 nearest neighbors of the closest matching restaurant, we get the top 20 recommendations with each one having distance metric with the given input. Since this part only involves the ratings and does not include information about the reviews, experimenting and fine-tuning this part of the project is out of the scope of this class. We merely use this to create a list of top X restaurants that are to be recommended for a user with given "favourite restaurant" based on the ratings they provide on different restaurants. Once we have the list of restaurants, we can leverage the user and restaurant profiles extracted from the first section of the model to predict the star ratings for the restaurants. We then multiply the distance metric from collaborative filtering (that shows relevance) and predicted rating (that depict restaurant fondness) to create a hybrid rating for the restaurants. We finally sort the restaurants based on the hybrid ratings. The high level architecture looks like **this**.

2 RESULTS

In this section, we present the rating prediction performance of the Assoc Matrix against the baselines discussed in previous section. For clarification, we reiterate the established baselines:

(1) Rating Prediction Model trained using Linear SVM following Jayashree and Kulkarni (2017).
(2) Rating Prediction Model trained using Logistic Regression

We can intuitively imply that both these baselines are expected to do better than the *Assoc* Matrix approach that we devised because the rating prediction coming directly from individual review is expected to have more specific information for a particular user-restaurant pair and combining all reviews of a user to all reviews of a restaurant could be expected to affect the pair specific domain knowledge of the specific user-restaurant pair. So, our aim was to observe if Assoc Matrix trained model can give results that are close enough to the baseline accuracy to justify the trade-off. Performance of the model is not solely defined by the accuracy. In addition, the computation time, should be one of the key reasons to choose one model over the other in form of a tie-breaker.

2.1 Rating Prediction: Accuracy

We created the *Assoc* Matrix for the data and treated them to SVM with varying regularization parameter C as before. After hyper-parameter tuning and cross validation, the C for optimum performance was 41.98 and the F1 score and the test score when treated with that turned out to 51.2 percent and 50.4 percent respectively. We replicated the procedure with Logistic Regression. Logistic Regression with C=590.17 gave a F1 score of 51.05 percent and a test score of 52.5 percent. When compared to the baselines, the accuracy with Logistic regression fares around the same ballpark. Hence we look into the run-time in the next section. Figure 5(a) shows the performance bar-graph, when compared to our baselines.

2.2 Rating Prediction: Computation Time

Comparing the run-time for SVM and Logistic Regression with our baselines, there is an obvious increase in run-time which can be attributed to doubling of dimension of our approach [figure 5(b)]. With our Logistic regression model being significantly faster than our SVM baseline, it is our clear choice.

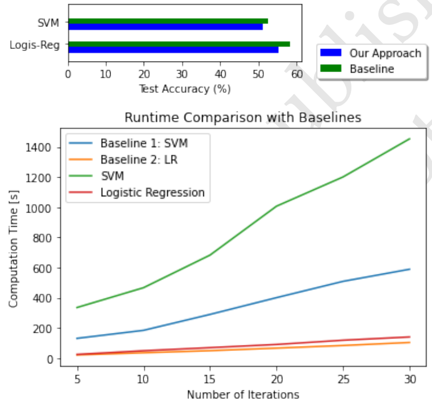
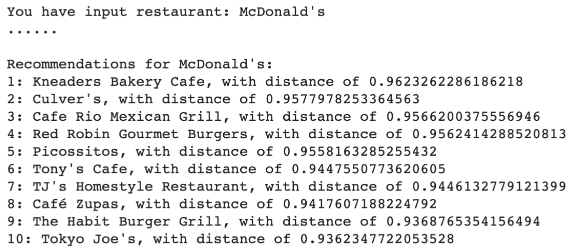


Figure 5: Rating Prediction Comparison with Baselines: (a) Test Accuracy (top), (b) Run-time (bottom)

2.3 Item-based Collaborative Filtering: kNN

As mentioned above, in this section, we recommend using traditional Item-based collaborative Filtering. Following figure shows 10 recommended restaurants in the results along with the distances when favourite restaurant - *McDonald's* is given as an input. Cosine distances of all the recommended restaurants look quite promising. In this filtering, text reviews are not used and only star ratings are used to feed kNN model with k value as 20. Resulting distances will be further used for hybrid rating.



```
You have input restaurant: McDonald's
......

Recommendations for McDonald's:
1: Kneaders Bakery Cafe, with distance of 0.9623262286186218
2: Culver's, with distance of 0.9577978253364563
3: Cafe Rio Mexican Grill, with distance of 0.9566200375556946
4: Red Robin Gourmet Burgers, with distance of 0.9562414288520813
5: Picossitos, with distance of 0.9558163285255432
6: Tony's Cafe, with distance of 0.9447550773620605
7: TJ's Homestyle Restaurant, with distance of 0.9446132779121399
8: Café Zupas, with distance of 0.9417607188224792
9: The Habit Burger Grill, with distance of 0.9368765354156494
10: Tokyo Joe's, with distance of 0.9362347722053528
```

Figure 6: Recommendation Results using item-based collaborative filtering

3 NEXT STEPS

Our model predicts 5 classes (5 stars) as compared to two in the baseline. Additionally, We are going to add our own approach of merging reviews along with the ratings to recommend. Certain steps that we are preparing for:

(1) Building a hybrid system that uses item based collaborative filtering and review based rating prediction to get recommendations. A prototype is shown **here**.
(2) Building a GUI using Python *Tkinter* module to recommend restaurants.

4 CONTRIBUTION

All team members have contributed a similar amount of effort.

5 RESOURCES

[1] R. Jayashree and Deepa Kulkarni. 2017. Recommendation System with Sentiment Analysis as Feedback Component. Advances in Intelligent Systems and Computing Proceedings of Sixth International Conference on Soft Computing for Problem Solving (2017), 359–367. DOI:http://dx.doi.org/10.1007/978-981-10-3325-4_36

[2] Yelp, Inc. 2020. Yelp Dataset. (March 2020). Retrieved March 28, 2020 from https://www.kaggle.com/yelp-dataset/yelp-dataset

[3] Boya Yu, Jiaxu Zhou, Yi Zhang, and Yunong Cao. 2017. Identifying Restaurant Features via Sentiment Analysis on Yelp Reviews. Identifying Restaurant Features via Sentiment Analysis on Yelp Reviews (2017).

[4] Facebookresearch. 2020. facebookresearch/fastText. (March 2020). Retrieved March 28, 2020 from https://github.com/facebookresearch/fastText

[5] Anon. 2016. FastText and Gensim word embeddings. (August 2016). Retrieved March 28, 2020 from https://rare-technologies.com/fasttext-and-gensim-word-embeddings/

[6] Nadbor. 2016. DS lore. (May 2016). Retrieved March 28, 2020 from http://nadbordrozd.github.io/blog/2016/05/20/text-classification-with-word2vec/

[7] KevinLiao159. KevinLiao159/MyDataSciencePortfolio. Retrieved March 28, 2020 from https://tinyurl.com/cse6240-item-based-reference