Recommender System using Latent Factor Collaborative Filtering on Yelp Restaurant Data

Eddie Ho
University of California, San Diego
boho@ucsd.edu

Jian-peng Li
University of California, San Diego
jil258@ucsd.edu

Jessica Song
University of California, San Diego
jwsong@ucsd.edu

Jiarui Xu
University of California, San Diego
jix028@ucsd.edu

ACM Reference Format:
Eddie Ho, Jian-peng Li, Jessica Song, and Jiarui Xu. 2018. Recommender System using Latent Factor Collaborative Filtering on Yelp Restaurant Data. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Recommender System is a common algorithm that we use on a daily basis in modern life. We wanted to explore how a recommender system algorithm can pick a ranked list of items to recommend to a specific user's needs. In this project, we narrowed our scope to the Yelp open data set that contains business reviews, user reviews, and separate ratings [1].

In this paper, we performed EDA on the data set and created a Recommendation System using Latent Factor Collaborative Filtering. A user can type in any desired restaurant in a String along with the current location, and our model predicts a list of 5 nearby restaurants that match the user's needs. Then, we evaluate this model's accuracy and discuss further improvements.

Our code is hosted at github.com/ChristineXu0924/cl_recommander

2 DATA SET

2.1 Identify data set

We chose to train a recommendation system using the Yelp data set for the final assignment. This data set represents a subset of the companies, reviews, and user data on Yelp, according to the company's official description. It was initially created for the Yelp Data set Challenge, which gives students the opportunity to explore or analyze Yelp's data and report their findings. Information about firms in 8 metropolitan regions in the USA and Canada may be found in the most recent data set. We selected this data set because of its credibility, volume, and diversity. Millions of user reviews from businesses including restaurants, coffee shops, and spas are included in the Yelp data set. We may leverage the data set's diversity to our advantage by making predictions and suggestions about the future.

Also, the massive scale of the Yelp data set enables us to develop a recommendation system that is more precise. We chose to concentrate on the Restaurant category because the original Yelp data set from the Kaggle website is too vast for us to analyze. The company data set's undesirable categories are removed, leaving us with 52,286 stores and restaurants. The validity of the data set comes last but certainly not least. It greatly improves the reliability of the data because the data sets are made available by the Yelp business itself. As a result, we are more confident that the information we are using accurately reflects the comments and reviews of Yelp users.

2.2 Characteristics of the Data set

We decided to use two data sets. First, we retrieved the data set "yelp_academic_dataset_business.json" in the Yelp public repository. This data set records 150346 unique businesses with 14 features, including name, address, location, star ratings, number of reviews, categories, and other attributes. On the other hand, wanting to filter on the reviews for each business, we also loaded the user reviews data set. However, it was too large for us to handle efficiently in this project, so we explored the notebooks analyzing this data set on Kaggle, and found a Yelp review subset. This subset was much smaller and easier to handle, but it still had 1268552 entries. This subset contains business, review rating (stars), the review text, and time of the review.

2.3 EDA statistics

In this section, we loaded in the business and review datasets, performed some basic statistics operations and to extract insights from the data. The information we have discovered are the following:

- There are 150,346 unique business names present in the business dataset
- There are 1,268,552 unique review posts present in the review dataset
- Average stars for all businesses is 3.596, median is 3.5, minimum is 1.0, and maximum is 5.0.
- Average stars for all reviews is 3.740, median is 5.0, minimum is 1.0, and maximum is 5.0.
- Proportion of restaurants which are open is 0.796.
- Most popular category is Restaurant.
- Most popular restaurant is Starbucks.
- Restaurant with most 5-star reviews from users

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

2.4 Data Cleaning

Data cleaning to guarantee training data consistency is an important step in sustaining model performance since inconsistencies and inaccuracies in training data can prevent algorithms from recognizing patterns.

In the process of data cleaning, we gained the following insights:

- Exploring the data set, we realize that the `funny`, `useful`, and `cool` are not particularly useful for our recommendation model, as it indicates that if a review post is funny or not, useful or not, and cool or not.
- There are values in the categories column which are `nan` which are useless to our recommendation model. Thus, we are removing businesses that have `nan` in the categories column.
- We filtered out the businesses which are in restaurant categories. Ending up with a data frame containing 52,286 stores and restaurants.
- We noticed that the `is_open` attribute in the business data set indicates if a business is still open currently. It makes no sense for our recommendation system to recommend a business which is permanently closed. Thus, we only want business which `is_open` has value of True.
- Reading from the data set file, the `date` column is stored as a string data type. So, we cast the values to datetime objects.

2.5 EDA through figures

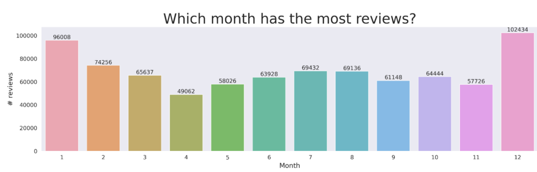

Figure 1: Reviews by Month.

In the above graph, we group all reviews by the month in their respective date value, and aggregate them with count. We can see which month has the highest or lowest review count. And so, we would be able to get a sense of the pattern of how users on Yelp platform write reviews. According to the distribution, we notice that users tend to post more reviews in January and December. April has the lowest number of reviews throughout the year. It goes up again during summer season, this may be due to summer vacation for students which indirectly increases people going out for food and post reviews after their visit.

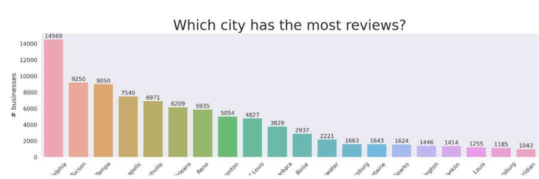

Figure 2: Most popular cities.

The above graph demonstrates cities that have the most businesses located. We have noticed that the imbalance distribution of businesses in different cities, may cause our recommendation system to have a larger weighting on specific areas. Moreover, we could also get a sense of which area has more restaurants located.

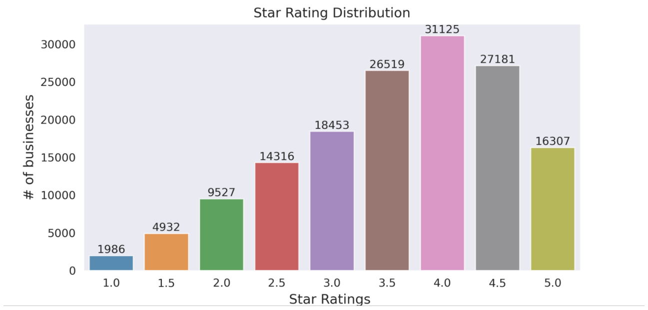

Figure 3: Distribution of rating (stars).

The above graph demonstrates distribution of stars among all restaurants. The distribution peaks at 4.0, and 1.0 has the lowest proportion. We have noticed that users tend to have a higher star rating, the probability that a user leaves a star rating below 3.0 is much lower than a star rating greater than or equal to 3.0.

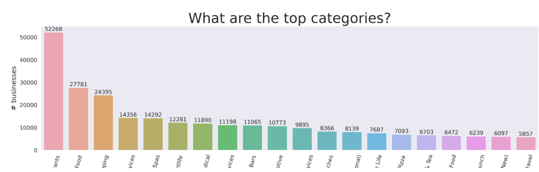

Figure 4: Categories.

The above graph shows that restaurants are the category with the most number of businesses. Therefore, we selected the restaurant entries in this data set to recommend.

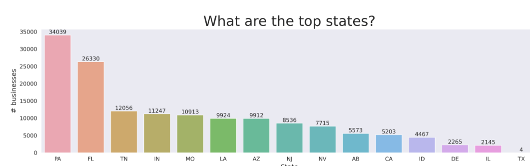

Figure 5: Location Distribution.

The above graph demonstrates states that have the most businesses located. We have noticed that the imbalance distribution of businesses in different states, may cause our recommendation system to have a larger weighting on specific areas. Besides, this may also be caused by each state having a different population which leads to the difference in the number of restaurants in such areas. In order to increase the accuracy of our model and limit the number of data points we train on, we later selected PA and FL (Pennsylvania and Florida) as the states we will investigate.

Figure 6: Common words in user reviews.

The most common words appear in reviews related to the restaurant category. Based on the Word Cloud graph, reviews related to restaurants category tend to include words like "food", "delicious", "order", etc.

3 PREDICTIVE TASK

3.1 Baseline Model

3.1.1 Initial Setting. In our baseline model, we first found `business_id`, `user_id`, `stars`, and `text`. We created two dataframes from this dataset, one containing each users' review, and one containing every review for each business. They're named userid-df and business-df, respectively.

3.1.2 Dataframes used. Then, we utilized the `nltk` library to remove the stop words from the text column in order to remove the low-level information to give more focus to the important information. We tokenized both texts in both userid-df and business-df. We also created a matrix representing user rating from the original review dataframe, with one axis representing users and one axis representing businesses. It contains the true rating (stars).

3.1.3 Matrix Factorization. Referring to Theo Jeremiah's recommendation system tutorial [2], we performed a similar matrix factorization on the rating matrix, userid-df, and business-df. In this function, we use gradient descent to tune two matrices P, Q (here as userid-df, and business-df) and discover the relationship between each business and each word. Note that in our baseline model, these three matrices are dense matrices, and the runtime of matrix factorization was $\Theta(n^3)$.

3.1.4 Search for optimal matrices. The formula $\hat{r}_{iu} = Q_i^T P_u$ provide us a rating predictions through P and Q. After decomposing the matrices, we use gradient descent to find the optimal P and Q.

In the case of the reviews, we have users who only left a few comments while other users left more than hundred comments. In order to avoid model overfitting, we add a regularization to the matrix factorization step

$$\lambda |\sum_x \|p_x\|^2 + \sum_i \|p_i\|^2|$$

3.2 Evaluation

The `category` field in the business dataframe indicates the type of food offered by a restaurant, such as boba, hamburger, tea, etc. This field is separate from the text reviews provided by users, so we did not use this feature at all in our training. To assess the accuracy of our model, we determine how likely it is that a recommended restaurant actually has a category similar to the input we are searching for.

To do this, we first identify the unique categories in the `category` column and then generate a string with a combination of categories, appending the target category. We input this string into our recommendation model, which returns the top five restaurant selections. We then assess the likelihood that these restaurants include our target category and record the top recommendations for each target category for manual inspection. Every recommended restaurant contained the feature we are searching for.

SFKjUQ1gmfwm7cJhMCFmkA	ZigZag Scallop	4417 Calienta St	Hernando Beach	FL	34607	28.4977	-82.650015	4.0	214	1	{'Alcohol': "'full_bar'", 'GoodForMeal': '{'de...	Nightlife, Seafood, Bars, Restaurants, America...

Figure 7: "The top recommendation for input Nightlife".

We also manually check the recommended restaurants to see what our model suggests. We wrote three potential user searches with decreasing levels of detail, shown in Figure 8,9 and 10.

```
top 5 result based on Request:

St. Pete Bagel Co.: Sandwiches, Restaurants, Bagels, Food, Donuts,
Coffee & Tea, with Rating of .

East Coast Pizza: Nightlife, Salad, Bars, Restaurants, Caterers, C
hicken Wings, Sandwiches, Event Planning & Services, Pizza, Italia
n, with Rating of .

Ninth Street Deli at Howards: Restaurants, Sandwiches, Bars, Ameri
can (New), Food, Breakfast & Brunch, Delis, Nightlife, American (T
raditional), with Rating of .

Buddy Brew Coffee : Food, Coffee & Tea, with Rating of .

Kettner Coffee Supply: Tacos, Mexican, Restaurants, Coffee & Tea,
Food, with Rating of .
```

Figure 8: Output for search "brunch with sandwiches, bagels, coffee and a vegan option"

```
Pontilly Coffee: Coffee & Tea, Food, Bagels, Bakeries, with Rating of
.

Cafe Du Monde City Park: Coffee & Tea, Food, with Rating of .

Kettner Coffee Supply: Tacos, Mexican, Restaurants, Coffee & Tea, Food, w
ith Rating of .

The Blind Tiger Cafe - Ybor City: Shopping, Vegan, Vegetarian, Beer, Wine
& Spirits, Sandwiches, Food, Bagels, Bakeries, Coffee & Tea, Men's Clothi
ng, American (Traditional), Restaurants, Women's Clothing, Breakfast & Br
unch, Acai Bowls, Fashion, Cafes, with Rating of .

August Rhodes Bakery: Event Planning & Services, Food, Sandwiches, Bakeri
es, Restaurants, American (New), Caterers, with Rating of .
```

Figure 9: Output for search "Coffee with view"

```
top 5 result based on Request:

Zukku-San Sushi Bar & Grill: Sushi Bars, Restaurants, with Rating of ⭐
⭐⭐⭐.

Modern Nails: Beauty & Spas, Nail Salons, with Rating of ⭐⭐⭐⭐.

Loco Lucho's Latino Kitchen: Spanish, Restaurants, Latin American, Puerto
Rican, Caribbean, Cuban, with Rating of ⭐⭐⭐.

Landry's Seafood House: Restaurants, Cajun/Creole, Seafood, with Rating o
f ⭐⭐⭐.

Pho Bistro: Food, Restaurants, Chinese, Bubble Tea, Vietnamese, with Rati
ng of ⭐⭐⭐.
```

Figure 10: Output for search "Barry wants rice"

Our conclusion is that our model creates reasonable outputs for detailed inputs. In Figure 8, the recommendations closely match the description. For less specific inputs, the outputs generally make sense. However, some can be irrelevant. For example, for input "Barry wants rice," which only has one word correlated to restaurants, one recommendation deviated to a nail salon.

3.3 Optimization Efforts

We started our baseline model using the first 1000 entries in the restaurant dataset, filtering using user reviews' text data and user's ratings. Our outputs made sense but was not entirely accurate, so we wanted to include more data.

The recommendation process was slow in our baseline model. We want to expand our train dataset to be more than 50000 entries, and also add in more features. Thus we needed to improve the runtime for the recommendation process. In our algorithm, the pivot table R containing a real rating for each user toward each restaurant is very sparse (one user might only review 1 restaurant). Thus instead of searching for rating through each row and column in the matrix factorization step, we convert it into a sparse matrix data type (csr _matrix from scipy.sparse()) [5], which allows us to access the location that has a rating(not-nan/non-zero value) directly. The runtime now is reduced to $\Theta(n)$, which, for sample size = 1000, is a decrease from 20 minutes to 1 minute. Then we were able to train our model using a randomly sampled 50000 data points from the filtered reviews dataset.

3.4 Finalized Model

We also conduct further filters and add location features into our finalized model. First of all, the result from our baseline model contains business that was not open(indicated by the 'is _open' column in the 'yelp_academic _dataset _business' file). We do not want to recommend a restaurant that is closed.

Based on our EDA on most businesses located states, we decide to focus on reviews for restaurants in PA and FL.

After filtration, the final business dataframe contains 13993 restaurants, and the final reviews dataframe contains 309458 reviews. We then use a randomly generated subset of size n instead of selecting the first n reviews. (n=50000)

Another layer of KMeans location-based recommender model to recommend restaurants around a given location (longitude and latitude) was added. The 13993 were clustered according to a KMeans clustering algorithm with hyperparameter n _clusters being set to 83 ($\sqrt{\frac{n}{2}}$ with n = 13993).

The following user/business Tf Idf feature matrix and the user-business-rating pivot table constructions are the same as in the baseline model. The only difference is that instead of a dense matrix, the user-business-rating data frame is converted into a sparse matrix (csr _matrix). Thus the factorization and update of P and Q now only takes O(n) runtime since we now can directly access the indices that contain non-zero values.

After generating restaurants based on the user's description input, the final model can take another location input in [longitude, latitude] format. This location will be clustered into a location group using the trained KMeans. And then a filter of the location cluster will be put on the previously generated restaurants and output the top 5 restaurants sorted by relatedness, overall stars, and number of reviews that the restaurant has.

4 LITERATURE

4.1 Recommender Systems (KNN, SVD, NN-keras) by Zolboo [4]

Our exploration was inspired by Kaggle's "Recommender System (KNN, SVD, NN-keras)" project. Our project began with the same Yelp dataset as Zolboo's. Zolboo created a recommender system based on restaurant features rather than user features by utilizing Content-Based Filtering. The assumption is that if a consumer enjoys a restaurant, they would also like other comparable businesses. On the other hand, Collaborative Filtering, a separate strategy, was also used by Zolboo. It is predicated on the notion that consumers prefer restaurants that are comparable to other restaurants they enjoy, as well as restaurants that are preferred by other people with similar interests. We have noticed that Zolboo's recommender system only performs simple preprocessing tasks, which include extracting and encoding the attributes column in the dataset. Therefore, we came to an inspiration that we may improve the performance of the model by using more relevant features in the content-based filtering model. According to Zolboo's description, his content-based model with K-Nearest-Neighbour, achieved an accuracy score of 0.567 on the training set, and a score of 0.534 on the testing set.

4.2 A Very Extensive Data Analysis of Yelp by Bukun [3]

Bukun's exploratory data analysis (EDA) and data visualization influenced our Yelp dataset discovery. Bukun's research looked at the entire business by identifying the most popular categories and cities with the most business parties listed in the Yelp dataset. Bukun also did sentiment analysis on reviews from specific businesses. Considering Bukun's detailed examination of the complete dataset, we opted to narrow our emphasis to the restaurant category subset of the dataset. Given that we are investigating a relatively small and more focused selection of businesses, we may discover some intriguing findings. Furthermore, we noticed that Bukun's analysis is missing the spatial elements of the data. As a result, we focus on the distribution of evaluations over the course of a year. Unexpectedly, we found a trend of review spikes over the winter and summer seasons.

4.3 How to Build a Restaurant Recommendation System Using Latent Factor Collaborative Filtering by Theo Jeremiah [2]

In his article, Theo Jeremiah provides an in-depth discussion on the collaborative filtering recommendation model, which we have employed in this project using the Yelp dataset. Through his work, Theo explains the fundamental concepts and assumptions that underpin this type of recommendation algorithm. We found his model to be compelling and chose to adapt it to fit the specifics of the current Yelp dataset, which we have designated as our baseline model. Building on this foundation, we then proceeded to incorporate additional features and data to create an enhanced version of Theo's recommendation model. Consequently, we view Theo's work in the restaurant recommendation system as an essential starting point for our project. By leveraging his model, we were able to build upon existing knowledge and develop a more effective recommendation algorithm.

5 RESULTS

5.1 Final Model Evaluation and Comparison

Our final model was able to produce results that contain the inputted category from previous categorical evaluation method. Therefore, we continued to qualitatively test our model by entering designated input.

This time, we included a location field in our test. We chose (40.16, -75.12) and (40, -75), respectively Willow Grove in Philadelphia and a small New Jersey town very close to Philadelphia.

```
user_input = "brunch with sandwiches, bagels, coffee and a vegan option"
location = (40.16, -75.12)
```

```
combined_top5 = generation(user_input, location)
```

```
# text output
for bid in combined_top5['business_id']:
    s_rest = business[business['business_id'] == bid].iloc[0]
    print("{}: {}, with Rating of {}. \n".format(
        s_rest['name'], s_rest['categories'], int(s_rest['stars'])*'⭐'))
```

```
HG Bucks Bagels: Sandwiches, Bakeries, Restaurants, Food, Bagels, Delis, Pizza, with Rating of ⭐⭐⭐.

The Bagel & Bread House: Bagels, Restaurants, Bakeries, Food, with Rating of ⭐⭐⭐⭐.

Village Bagel Company: Restaurants, Food, Bagels, with Rating of ⭐⭐⭐⭐.

Buona Via: Mediterranean, Nightlife, Breakfast & Brunch, Seafood, Restaurants, Bars, Greek, Italian, with Rating of
⭐⭐⭐.

Luhv Vegan Bistro: Restaurants, Burgers, Cafes, Vegan, with Rating of ⭐⭐⭐⭐.
```

Figure 11: Recommendations for "brunch with sandwiches, bagels, coffee and a vegan option" on final model

We no longer see irrelevant recommendations in our inputs. Recommendations for a detailed input matches our requirements, and the recommendations for an abstract input were all related to our input (Figure 11, 12, 13). For example, Popeyes may seem irrelevant to the input "Barry wants rice," but Cajun rice is a main side dish at the fast food restaurant.

With location, we were able to search the same input and location on Yelp. The inputs above did not produce recommendations that matched with the actual Yelp, but our model was able to do so on other inputs, shown in the Figure 14 and 15.

```
user_input = "coffee with view"
location = (40, -75)
```

```
combined_top5 = generation(user_input, location)
```

```
# text output
for bid in combined_top5['business_id']:
    s_rest = business[business['business_id'] == bid].iloc[0]
    print("{}: {}, with Rating of {}. \n".format(
        s_rest['name'], s_rest['categories'], int(s_rest['stars'])*'⭐'))
```

```
Mayfair Diner: Burgers, Diners, Restaurants, with Rating of ⭐⭐⭐.

Al's Corner Deli and Catering Services: Food, Event Planning & Services, Caterers, Delis, Restaurants, with Rating of
⭐⭐⭐⭐.

Dunkin': Donuts, Food, Coffee & Tea, Restaurants, with Rating of ⭐⭐.

Chick-fil-A: Event Planning & Services, American (Traditional), Restaurants, Food, Caterers, Fast Food, Chicken Shop,
Breakfast & Brunch, with Rating of ⭐⭐⭐⭐.

Originals Italian Restaurant & Bar: Food Delivery Services, Pizza, Restaurants, Sandwiches, Food, Beer, Wine & Spirit
s, with Rating of ⭐⭐⭐.
```

Figure 12: Recommendations for "Coffee with view" on final model

```
user_input = "barry wants rice"
location = (40, -75)
```

```
combined_top5 = generation(user_input, location)
```

```
# text output
for bid in combined_top5['business_id']:
    s_rest = business[business['business_id'] == bid].iloc[0]
    print("{}: {}, with Rating of {}. \n".format(
        s_rest['name'], s_rest['categories'], int(s_rest['stars'])*'⭐'))
```

```
King Food Restaurant: Restaurants, Soup, Chicken Shop, Chinese, with Rating of ⭐⭐⭐⭐.

China Gate: Restaurants, Chinese, with Rating of ⭐⭐.

Jin Jin Chinese Restaurant: Restaurants, Seafood, Soup, Chinese, with Rating of ⭐⭐⭐.

Sweetie's Take Out & Catering: African, Restaurants, Caterers, Event Planning & Services, South African, with Rating
of ⭐⭐⭐⭐.

Popeyes Louisiana Kitchen: Restaurants, Fast Food, American (Traditional), Chicken Wings, with Rating of ⭐⭐.
```

Figure 13: Recommendations for "Barry wants rice" on final model

```
user_input = "cheap burger"
location = (40.16, -75.12)
```

```
combined_top5 = generation(user_input, location)
```

```
# text output
for bid in combined_top5['business_id']:
    s_rest = business[business['business_id'] == bid].iloc[0]
    print("{}: {}, with Rating of {}. \n".format(
        s_rest['name'], s_rest['categories'], int(s_rest['stars'])*'⭐'))
```

```
Sonic Drive-In: Burgers, Food, Fast Food, Restaurants, Ice Cream & Frozen Y

Old School Burgers, Dogs & Shakes: Fast Food, American (Traditional), Burge
⭐⭐⭐⭐.

P.J. Whelihan's: Nightlife, Bars, American (Traditional), Restaurants, with

Mike's York Street Bar & Grill: Nightlife, Restaurants, Sports Bars, Bars,
⭐⭐.

PJ Whelihan's Pub + Restaurant - Horsham: Gastropubs, American (Traditional
ts, with Rating of ⭐⭐⭐.
```

Figure 14: Recommendations for "Cheap burger"

The recall (number of relevant items retrieved / total number of relevant items) is quite low. However, it is quite normal considering how large-scaled and up-to-date the actual yelp dataset is.

Overall, when compared to the baseline model, the finalized model produces results that are more relevant to the search and the location.

5.2 Effectiveness

We built the entire model based on user reviews (String), user rating (float), longitude and latitude. User reviews predicted rating matrix and allowed this recommender model to match input with existing

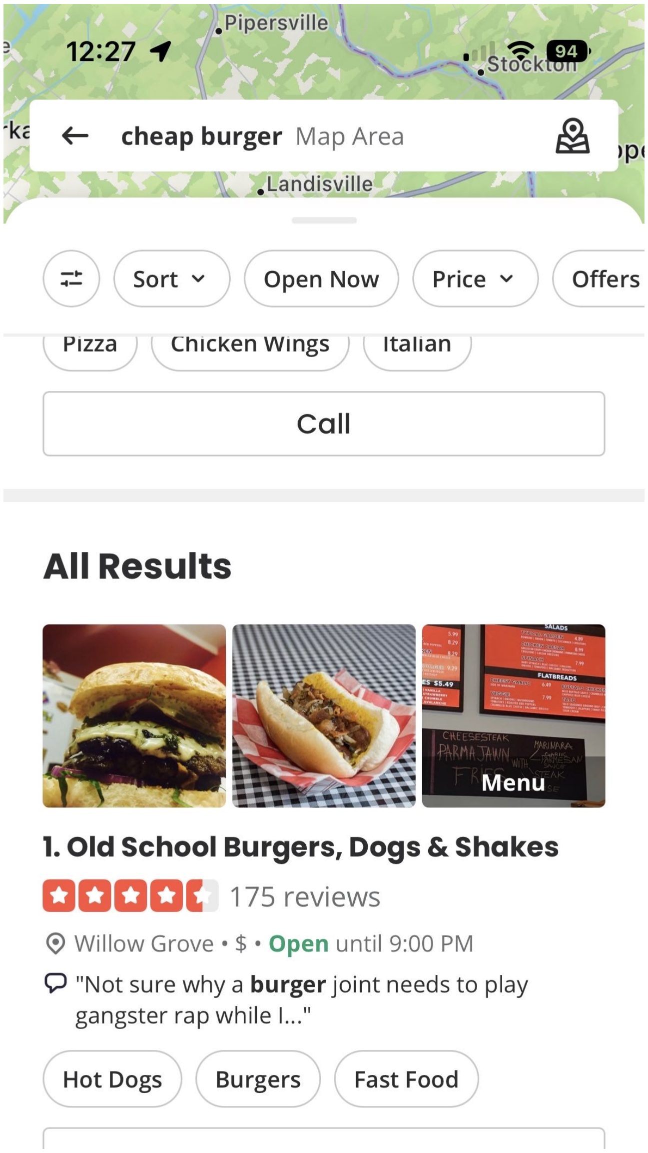

Figure 15: Yelp search in Willow Grove

reviews. Longitude and latitude effectively make our recommendations relevant to the user, since one can only go to restaurants within a reachable distance to their location.

5.3 Ideal Hyper-parameters

We built a recommender system from scratch, so there were no hyper-parameters to tune. For our optimization of the model, please refer to sections Baseline Model and Finalized model.

5.4 Major Takeaways

(1) A recommendation system can take a lot of features into consideration. Before training a system, one should carefully design which features to use, and how to use them.

(2) For example, we wanted to add the price feature and filter input with price categories. We also wanted to incorporate time of the day and the businesses' open hours. For example, if a user is searching at noon, the recommendations should be restaurants that serve lunch well. Some restaurants have a great rating for their breakfast, but not necessarily dinner. In the future, we want to closely investigate certain restaurants to develop algorithms for cases that vary with hours as well. The learning of user's features such as user preferences and their age could also be taken into consideration in the future model.

(3) More data always yield to better results.

(4) Due to time and technical constraints, we were only able to train the recommendation model on 50,000 rows. However, we have 1268552 reviews, and this was only a selected subset. If we had more data and computing power, using the entire dataset that Yelp gives would be a great way to make more accurate predictions in more locations.

(5) A recommender system is best evaluated through interactions with users.
   (a) Our evaluation technique was very limited. Our baseline model evaluation ensured that each recommendation fit the desired category; it does not measure how good a recommendation actually is. We examined certain input strings; however, our examination was a very limited perspective on a very small sample.
   (b) In the future, we can annotate the goodness of recommendations for a larger sample (e.g. 1000) of input strings. This will give a quantitative and more accurate measure of the model performance.
   (c) If actual users search what they want, go to the restaurants, and rate how good the recommendation was, we can receive a large dataset of users' feedback on our prediction, and tune the model accordingly.

6 REFERENCES

(1) https://www.yelp.com/dataset
(2) https://towardsdatascience.com/how-to-build-a-restaurant-recommendation-system-using-latent-factor-collaborative-filtering-ffe08dd57dca
(3) https://www.kaggle.com/code/ambarish/a-very-extensive-data-analysis-of-yelp
(4) https://www.kaggle.com/code/zolboo/recommender-systems-knn-svd-nn-keras
(5) https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.csr_matrix.html