Location Based Restaurant Recommendations using Yelp dataset

Deepti Gururaj Baragi and Akshharaa Tharigonda

May 3, 2023

1 Introduction

When it comes to choosing a restaurant, there are many factors to consider. What kind of food do you want? What's your budget? What's your location? And if you are new to an area, it can be even more difficult to know where to start.

That's where a restaurant recommendation system comes in. A restaurant recommendation system is a tool that can help people find the best restaurants according to their needs.It does this by taking into account the past ratings and reviews, as well as the location of the people.

In this project, we have developed a restaurant recommendation system for the state of Idaho. The system will take into account the user's location, so that it can recommend restaurants that are close by.

We believe that this system will be a valuable tool for both users and businesses. For users, it will provide a way to find new and interesting restaurants that they might not have otherwise found. For businesses, it will provide a way to reach new customers and increase sales.

2 Problem Statement

The problem that this project is trying to solve is the difficulty of finding good restaurants, especially for people who are new to an area. The system will help people find restaurants that are close to them, and that match their preferences.

3 Domain

The domain of this application is restaurants. The intended users are people who want to find new restaurants to eat at. They are solving the problem of not knowing what to eat next, or of not being able to find restaurants that they will enjoy. They will interact with the system by browsing the catalog, searching for restaurants, and looking at the restaurant rating. They will expect recommendations to be accurate, relevant, and timely. They will evaluate the effectiveness of recommendations by reading them and deciding whether or not they are interested in the restaurants that are recommended.

4 User Story

- User: A 25-year-old woman who is looking for a new restaurant to eat at.
- Problem: She is not sure what to eat next, and she is tired of eating at the same old restaurants.
- Solution: She uses the recommendation system to find new restaurants that she might enjoy.
- Interaction: She browses the catalog, searches for restaurants, and looks at the ratings.
- Recommendations: The system recommends restaurants that are in her location and also having cuisines according to her preference.
- Evaluation: She looks at the recommendations and decides whether or not she is interested in the restaurants that are recommended. If she finds a restaurant that she is interested in, she decides to go to that place to try.

5 Process Overview

The following steps are followed to do the Location-based Restaurant Recommendations to the user.

- Both the business and review data from the Yelp dataset are collected
- The data is cleaned so that there are no missing values present in the data before processing.
- The data is filtered using state, categories, and some other features like latitude, longitude, user ratings, and overall business ratings are also selected. The data is the final data.
- For doing the hyperparameter tuning, the surprise library and the algorithms available in it is used. To do this, the final data is filtered to have only the userid, businessid and the user ratings. This is now the ubr data.
  - Using the ubr data available, the user ratings are predicted by applying the Matrix Factorization algorithms. The algorithm with the best RMSE is selected.
  - To estimate how well the model is performing on the new data, we used a set of user-item pairs that the model has not seen before. This is done by building an anti-test test from a full-train object.
  - The absolute difference between the overall business rating and the predictions is calculated and stored in a column b_diff to use in the future.
  - The data with the user-item pairs that the model has not seen before and the final data in step 3 are merged so that now the data has the predictions, its difference with the overall business rating, latitude, longitude, and other features required.
- K-means clustering with silhouette scoring is used to get the best number of clusters for grouping the businesses based on their locations.
- The data is sorted in the ascending order of the b_diff. Because, the less the difference between the predictions and the actual rating, the better the restaurant.
- Now, to get the nearby restaurant recommendations in Idaho state based on the user's food preference and location, we defined a function that takes the input of the user who is outside of the data set.
- This function returns the restaurant recommendations to the used based on his food preference and the locations co-ordinates.

6 Data Description

The data set used for the project Location-based Restaurant recommendations is the Yelp data set. If it is built, our recommender's ideal data would be the user-generated data and business data combination. The user-generated data would have details about the user's interactions with the business, check-in details, and the review text. Along with this, though the business data provides enough details, we would want to collect, the other details of the business-like price range, hours of operation, or if the takeout or the delivery is available. This information on the Yelp platform would help us to get insights into user preferences. Using this information, the patterns in user behavior can be observed and also provides more context for recommendations using which more personalized recommendations can be generated. We can also collect other data that gives more context to the user's preferences like the weather data, as based on whether the user may have different preferences the food categories or business. For the experimental data, we used the Yelp dataset. It has information about various data like business, reviews, users, check-in, etc. For our application, we used the business and review data.

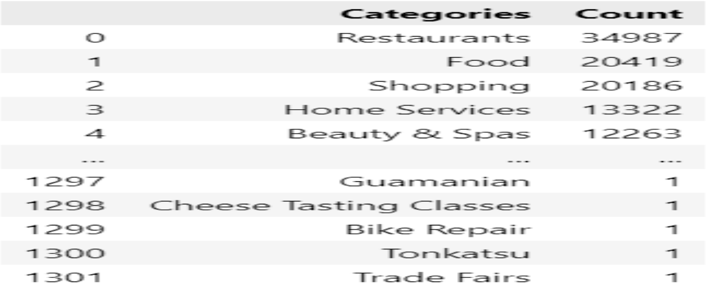
- Business data: This is a business.json file that has details about the unique businesses, their ratings, location coordinates, categories if the business is open or not, city, state, postal code, and other attributes.
- Reviews data: This review.json dataset has information about the user, review Id, user ratings, and the user review text.

For our experimental data, we filtered out the businesses that are open and based on the 'Restaurants' category. We filtered out the data with the details related to a particular state, Idaho. Later the data is merged with the review. json data, on the business Id, so that our data has the user ratings given to a particular business and made sure all the business Id has the respective ratings i.e., no business from the business data is missing the reviews in the merged data. Then we used the recommendations algorithms from the surprise library to generate the user rating predictions using which we tried to generate the recommendations based on the user preference and his location coordinates.

7 Data Analysis

This section discusses in detail, how the data is analyzed including data cleaning and feature selection. Firstly, the business data is read into the memory. The restaurant recommendations will be valid only if the business is open. So, all the businesses that are open ($is\_open == 1$) are selected. Each business comes under different categories. All the categories that are not null are selected. Of the top 30 categories available in the data, the 'Restaurants' category is selected as it has the highest count in the categories. For this analysis, all the cities in Idaho state with the respective businesses that are open are selected.

The following figures show the distribution of the categories and the businesses per city in Idaho state.



	Categories	Count
0	Restaurants	34987
1	Food	20419
2	Shopping	20186
3	Home Services	13322
4	Beauty & Spas	12263
...	...	...
1297	Guamanian	1
1298	Cheese Tasting Classes	1
1299	Bike Repair	1
1300	Tonkatsu	1
1301	Trade Fairs	1

Figure 1: categories Distribution

From the above figure, we can say that the restaurants have the highest count in the categories followed by Food and Shopping.

Table 1: cities-Businesses count

City	Business count
Boise	601
Meridian	236
Eagle	63
Garden City	32
Boise City	5

The Figure with the cities-Business count, shows the top 5 cities and the businesses count in the state of Idaho.

Secondly, the review data has the details about the *review_id*, *user_id*, *business_id*, useful, funny cool, etc columns. Both the reviews data and the filtered data are merged in the *is_business*. From here, we made sure that the review count in the filtered business data and the shape of the merged data is the same. Here, we made sure that all the unique businesses in Idaho have the review available and that there are no missing values present. The data is cleaned and filtered by replacing the Nan values with the empty dictionary till there are no null values present in the data.

Finally, the columns that are required are selected by dropping the unnecessary columns like funny, cool, etc. For our project, the most important columns are *user_id*, *business_id*, latitude, longitude, city, state, categories, user rating, and overall business rating.

8 Metric Used: RMSE

In the context of location-based restaurant recommendations considering the user food preference and location, we choose RMSE as a metric to evaluate the accuracy of the predicted user ratings.

Rmse is a metric that is used to predict continuous variables like ratings. Our application of recommendations involves predicting the ratings using the user ratings for a business. By using RMSE, the root mean squared difference between the actual ratings and predicted ratings is measured.

In the context of our application, RMSE is the appropriate metric as it provides an estimate of how well the model predicts the ratings compared to the actual ratings.

The accuracy of the recommender can be increased by including the location coordinates as it captures the spatial dependencies between the users and restaurants.

9 Hyperparameter Tuning

For the hyperparameter tuning, we used the GridSearchCV from the surprise library. We chose to tune the factors and epochs using the SVD over the grid of hyperparameters, using cross-validation. This gives the estimation of the best performance of the hyperparameters. We defined a dictionary *param_grid* that has the values of the hyperparameters that we wanted to tune. We created a search object using the GridSearchCV and passed the algorithm that needs to be used for tuning, the parameters grid with the hyperparameters and their respective values, scoring measure RMSE, and the number of folds for the cross-validation. The fit method returns the best hyperparameters by training and testing the algorithm on all the combinations of the grid by using the cross-validations and the RMSE performance metric. The best hyperparameters obtained are: $n\_factors$: 10, *n_epochs*: 10 with a best RMSE score of 1.40518

I apologize for the mistake. The figure is still in the SVD section because I did not move it to the Algorithms section. Here is the updated code with the figure moved to the Algorithms section:

10 Algorithms

Before generating recommendations, we chose three algorithms:

- SVD
- SVDPP
- NMF

Table 2: Algorithms-RMSE

Algorithm	RMSE
SVD	1.392865
SVDpp	1.410567
NMF	1.632652

The figure shows the RMSE scores for each algorithm. As you can see, SVD has the lowest RMSE score, which means that it is the most accurate algorithm.

- SVD (Singular Value Decomposition) is a matrix factorization technique that can be used to find latent factors in a matrix. These latent factors can then be used to predict user ratings. SVD is a very popular algorithm for recommender systems because it is relatively easy to understand and implement. It is also relatively accurate, especially for large datasets.

  SVD is helpful for our project because it can be used to find latent factors that can be used to predict user ratings for restaurants. This is important because it allows us to recommend restaurants to users that they are likely to enjoy.

  For example, if a user has rated a number of Italian restaurants highly, SVD can be used to find other Italian restaurants that the user is likely to enjoy. This is because SVD can identify the latent factors that are common to all of the Italian restaurants that the user has rated highly.

- SVDpp (SVD++ with Probabilistic Implicit Feedback) is a variant of SVD that is specifically designed for implicit feedback data. Implicit feedback data is data that does not contain explicit ratings, such as clicks or purchases. SVDPP is able to learn latent factors from implicit feedback data, which makes it a good choice for recommender systems that are used to recommend items that users have not explicitly rated.

  SVDpp is helpful for our project because it can be used to find latent factors that can be used to predict user ratings for restaurants even if the user has not explicitly rated any restaurants. This is important because it allows us to recommend restaurants to users that they are likely to enjoy, even if they are new to the area.

  For example, if a user has clicked on a number of Italian restaurant listings, SVDPP can be used to find other Italian restaurants that the user is likely to enjoy. This is because SVDPP can identify the latent factors that are common to all of the Italian restaurants that the user has clicked on.

- NMF (Non-negative Matrix Factorization) is another matrix factorization technique that can be used to find latent factors in a matrix. NMF is different from SVD and SVDpp in that it is designed to find non-negative latent factors. This makes NMF a good choice for recommender systems that are used to recommend items that are composed of non-negative values, such as colors or ingredients.

  NMF is helpful for our project because it can be used to find latent factors that can be used to recommend restaurants based on their ingredients. This is important because it allows us to recommend restaurants to users that they are likely to enjoy based on their dietary restrictions or preferences.

  For example, if a user has specified that they are vegetarian, NMF can be used to find restaurants that serve vegetarian food. This is because NMF can identify the latent factors that are common to all of the vegetarian restaurants.

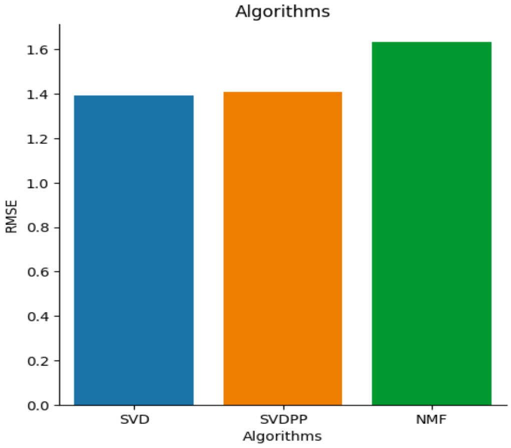


Figure 2: Algorithm Performance

We chose these algorithms because they are all matrix factorization techniques that can be used to find latent factors in a matrix. These latent factors can then be used to predict user ratings.

We evaluated the performance of each algorithm based on its RMSE score, and found that SVD has the lowest RMSE Score. Therefore, we chose SVD as the best-performing model.

We used the Surprise library to implement these algorithms. It is easy to use and provides a variety of features, such as cross-validation and evaluation metrics.

11 SVD

We performed held-out train-test split and full train-test. The held-out train-test split is a method of evaluating the performance of a recommender system by splitting the data into a training set and a test set. The training set is used to train the recommender system, and the test set is used to evaluate the performance of the recommender system. The full train-test is a method of evaluating the performance of a recommender system by using the entire dataset to train the recommender system and then evaluating the performance of the recommender system on the entire dataset.

We found that the SVD algorithm performed best on both the held-out train-test split and the full train-test.

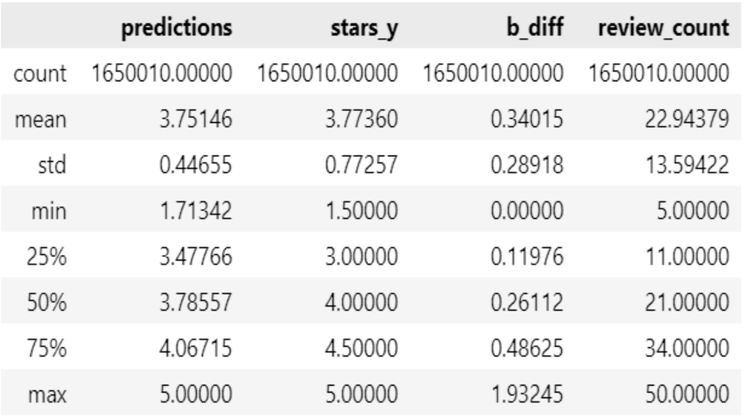
12 Findings

Once the data with the predictions and the original final data are merged, from the statistical data of the data frame, we can see that the maximum b_diff is 1.93 and the minimum is 0. From this and the ratings, we can say that the restaurants are good in Idaho with a maximum difference of 1.93, These details can be found in the figure 3.

Consider a user who is new to the Idaho state and in a location. He wants to know the best nearby restaurants available that satisfy his food preference. Consider the user's food preference as 'pizza' and his location coordinates are, 43.634003 and -116.402057 respectively. The results obtained by this user are in figure 4.

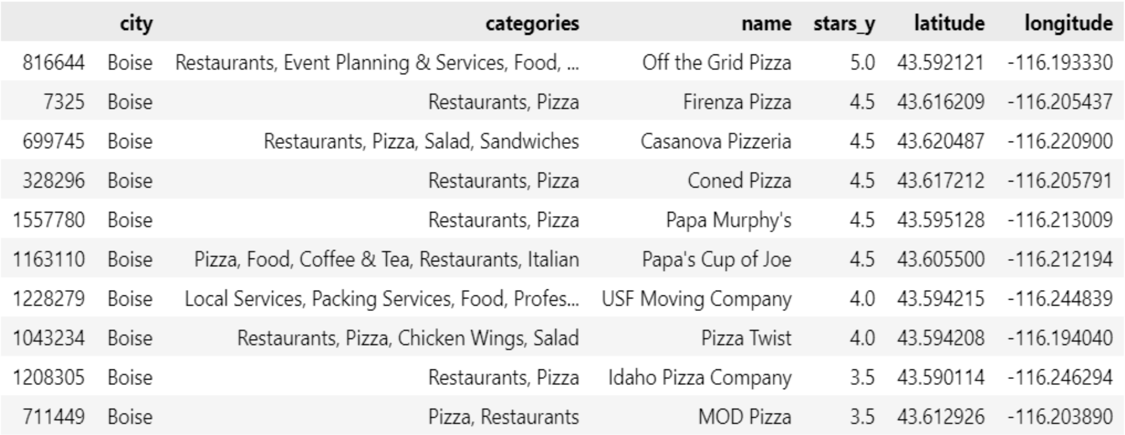
The map plot for the above results can be seen in the figure 5.

From the above results, we can see that the restaurants are recommended to the user near his location, based on his food preference. The details like the name of the restaurant, its overall rating, nearby latitude, longitude, and the categories columns are returned.



	predictions	stars_y	b_diff	review_count
count	1650010.00000	1650010.00000	1650010.00000	1650010.00000
mean	3.75146	3.77360	0.34015	22.94379
std	0.44655	0.77257	0.28918	13.59422
min	1.71342	1.50000	0.00000	5.00000
25%	3.47766	3.00000	0.11976	11.00000
50%	3.78557	4.00000	0.26112	21.00000
75%	4.06715	4.50000	0.48625	34.00000
max	5.00000	5.00000	1.93245	50.00000

Figure 3: Statistical Information of data



	city	categories	name	stars_y	latitude	longitude
816644	Boise	Restaurants, Event Planning & Services, Food, ...	Off the Grid Pizza	5.0	43.592121	-116.193330
7325	Boise	Restaurants, Pizza	Firenza Pizza	4.5	43.616209	-116.205437
699745	Boise	Restaurants, Pizza, Salad, Sandwiches	Casanova Pizzeria	4.5	43.620487	-116.220900
328296	Boise	Restaurants, Pizza	Coned Pizza	4.5	43.617212	-116.205791
1557780	Boise	Restaurants, Pizza	Papa Murphy's	4.5	43.595128	-116.213009
1163110	Boise	Pizza, Food, Coffee & Tea, Restaurants, Italian	Papa's Cup of Joe	4.5	43.605500	-116.212194
1228279	Boise	Local Services, Packing Services, Food, Profes...	USF Moving Company	4.0	43.594215	-116.244839
1043234	Boise	Restaurants, Pizza, Chicken Wings, Salad	Pizza Twist	4.0	43.594208	-116.194040
1208305	Boise	Restaurants, Pizza	Idaho Pizza Company	3.5	43.590114	-116.246294
711449	Boise	Pizza, Restaurants	MOD Pizza	3.5	43.612926	-116.203890

Figure 4: Results: restaurant Recommendations

13 Learnings

SVD performed best among SVD, SVDpp, and NMF. This is because SVD is a simple and efficient algorithm that can be used to generate accurate recommendations. SVDpp is a more complex algorithm that can improve the accuracy of recommendations by taking into account the user's past behavior. NMF is a more complex algorithm that can improve the accuracy of recommendations by taking into account the user's interests. However, NMF is more computationally expensive than SVD and SVDpp.

The number of latent factors is important. The number of latent factors is the number of dimensions that are used to represent the users and items in the recommendation system. A higher number of latent factors can improve the accuracy of recommendations, but it can also make the recommendation system more computationally expensive.

The regularization parameter is important. The regularization parameter is a hyperparameter that controls the amount of regularization that is applied to the recommendation system. Regularization helps to prevent overfitting, which is when the recommendation system learns the noise in the data instead of the underlying patterns.

The evaluation metric is important. There are a variety of evaluation metrics that can be used to measure the accuracy of a recommendation system. The most appropriate evaluation metric will depend on the specific application.

14 Conclusion, Limitations and Future works

In conclusion, we can say that the restaurant recommendations would be more effective if the locations and food preferences of the user are considered. Using the algorithms in the surprise library, we predicted the user ratings and determined the number of clusters to be used using silhouette scoring with the K-means.

The results of RMSE obtained using the various algorithms showed that the SVD algorithm worked better

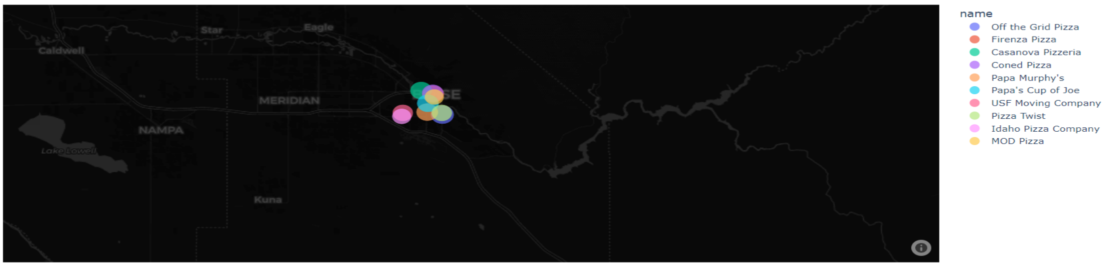


Figure 5: Results: restaurant Recommendations Map

when compared to the others. The performance of the algorithm changes depending on the size of the dataset.

The resulting recommendations are given to the external user who is not present in the data using his food preference and the location coordinates. Overall, our project demonstrates the effectiveness of restaurant recommendations using the locations with the food preference of the user. The recommendations would be more personalized with better accuracy.

Though through this project, we can say that we can give more personalized recommendations to the users, to check this we further need to perform some tests with the user groups. Only then we estimate to how many levels the recommendations were useful to the users. In this work, we did not include the threshold value to the business while recommending as the max threshold itself was less than 2. This may change the results in the future if there are businesses with greater differences between the predicted ratings and the actual rating of the business.

Overcoming the above limitations, For future works on this project, we want to use the user data available in the Yelp data set to get more information about the user and his reviews. Considering other factors like weather, hours when the business is open, and delivery or takeout options will help in getting more personalized recommendations to the users. The analysis of the review text provided by the user can be done as the continuation of this project as it will help in understanding the user needs and preferences in a more detailed way.

15 Reflection

Through this project,

- We learned that it is important to have a good understanding of the problem you are trying to solve before you start developing a solution. In this case, we needed to understand what factors people consider when choosing a restaurant, and what data we would need to collect in order to build a recommendation system that would be relevant to users.

- We also learned that it is important to be patient and persistent when developing a recommendation system. It can take a lot of time and effort to collect and clean data, develop an algorithm, and evaluate the performance of the algorithm. It is also important to be willing to iterate on the solution based on user feedback.