the yelper helper: Providing recommendations to Yelp users based on their prior review history

Aditi Goyal[1*]; Akayla Hackson[2*]; Neeraj Mathur[2*]
[1]Stanford University School of Medicine, [2]Stanford University School of Engineering | *all authors are primary authors and contributed equally

Abstract

Yelp is a widely used platform designed to recommend businesses and services, ranging from restaurants to nail salons. Despite its popularity, the platform can be cumbersome to use. Searching for new restaurants and services can be tedious and time consuming for yelp users. They must search numerous options and read multiple reviews per option to determine if it is a good fit.

Here we use machine learning techniques to perform 3 experiments:
1. predict the star rating of a restaurant based on a set of reviews;
2. recommend a restaurant to a user based on their star ratings
3. explore the potential of OpenAI in performing these tasks.

Experiment 1: Star Prediction

We begin by using 5 different ML algorithms to predict the star rating of a review based on the raw text: Naïve Bayes (NB), Linear Support Vector Machines (SVMs) Multiclass Logistic Regression (MLR), K Nearest Neighbors (KNN), and Random Forest Classifiers (RFC). The following table shows the precision scores for each model's ability to classify a review into a star rating class

Star Rating	NB	SVM	MLR	KNN	RFC
1	0.47	0.53	0.7	0.54	0.71
2	0.19	0.28	0.46	0.26	0.48
3	0.23	0.29	0.44	0.25	0.44
4	0.38	0.40	0.48	0.29	0.39
5	0.70	0.68	0.7	0.5	0.57

Our results do show that recommending highly rated restaurants may be an effective strategy, as most of our results perform decently well at deciphering 5-star reviews. We associate this trend to the fact that there are a higher number of 5-star reviews in our dataset. This corresponds with our dimensionality reduction analysis of the review text, which finds that there is considerable overlap between the different classes of reviews.

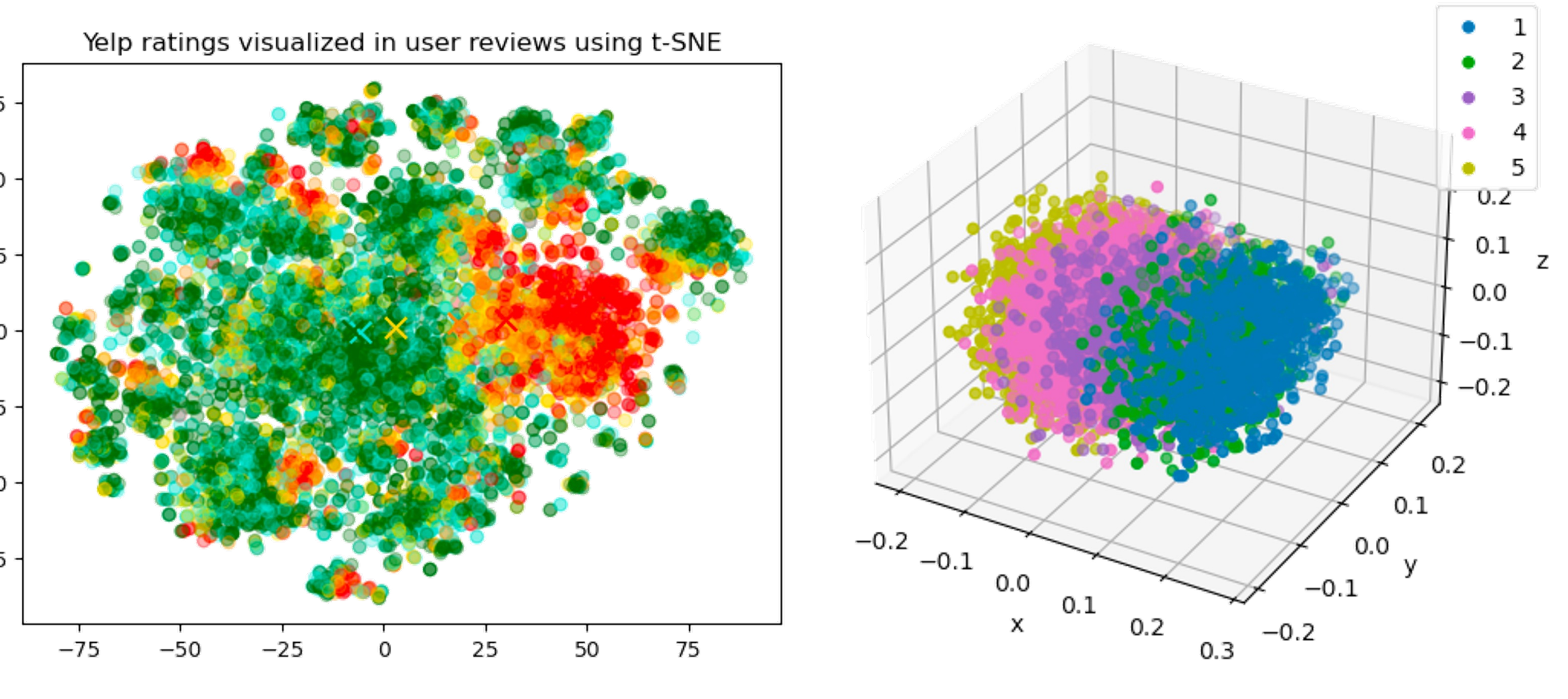


Figure 1a (left) 2D t-SNE representation of reviews: 1=red, 2=dark orange, 3=gold, 4=turquoise, 5=dark green. Figure 1b (right) 3D representation of review separation on 3 PC's.

Conclusions

Through our 3 experiments, we explored the ability for ML to improve the user experience on Yelp.
Our first experiment indicates that we can identify extremely good and bad reviews well, which indicates it will be easier to identify restaurants that should and should not be recommended. As we would want to recommend strong matches, our ability to identify 5-star ratings helps us, despite our inability to identify 2-4 star restaurants.
Our second experiment demonstrates that Collaborative Filtering similarity-based recommendation based on user can perform well for large and small Yelp datasets.
Our third experiment uses OpenAI embeddings and generates strong recommendations for users. We show this method can work with a direct search, or by processing a user's previous review history. This removes the need for a rigorous review history per user and increases the real-world viability of our project.

Dataset

We utilize the publicly available Yelp dataset. This dataset contains nearly 7 million reviews for over 150,000 businesses.
We note that our dataset contains many users with a short review history (less than 10 reviews per person). We also observe that most reviews are quite short, and that 5-star reviews are most prevalent in our dataset.

We conducted some preprocessing by selecting for restaurants and removing restaurants and users with no review history. From there, we sampled the dataset to create different subsets for each of our 3 tasks.

For Star prediction, we randomly selected 250,000 reviews from the set of all restaurant reviews. These reviews were tokenized using CountVectorizer, and then scaled to avoid bias, prior to being used in any ML algorithm. We then split these reviews into a 200K train set and 50K test set.

For Restaurant Recommendation, We only included users with 10+ reviews and business with 100+ reviews. This left 1.5M reviews on 7,000 restaurants from 25,000 users. This data was randomly split 80%/20% train/test.

For testing the OpenAI embeddings, we constrained our search for restaurants in Santa Barbara (zip code 93101) to simulate real world conditions. To that end, we also dropped the 10+ review history per user and 100+ reviews per business requirements.

Experiment 2: Restaurant Recommendations

CF models are a class of recommendation systems commonly used today. CF focuses on analyzing user behavior and preferences to identify patterns and recommend items based on similarities between users. In this project, we use K nearest neighbors, an example of a CG algorithm. We aim to analyze reviewer preference in restaurants, and use that to suggest new restaurants.

We measure review similarity using cosine_similarity, and error with RMSE.

$$\text{cosine\_similarity}(a,b) = \frac{\sum_i r_{ai} \cdot r_{bi}}{\sqrt{\sum_i r_{ai}^2}\sqrt{\sum_i r_{bi}^2}} \qquad RMSE = \sqrt{\frac{1}{n}\sum_{u,r}(O_{u,r} - P_{u,r})^2}$$

The hyperparameter k in k-NN represents the number of nearest neighbors to consider when making a prediction. We tested this model by directly substituting values for k, and through Matrix Factorization which solves for the hyperparameter k. The results for both are below:

k-Nearest Neighbor

Test	Dataset	Sample 1	Sample 2	Sample 3	Avg RMSE
kNN, k=5	Train	1.072	1.033	1.016	1.04
	Test	1.525	1.458	1.229	1.404
kNN, k=10	Train	1.081	1.014	1.091	1.062
	Test	1.372	1.375	1.15	1.299
kNN, k=20	Train	1.009	0.924	1.007	0.98
	Test	1.128	1.041	1.289	1.153

Matrix Factorization

Test	Dataset	Sample 1	Sample 2	Sample 3	Avg RMSE
MF, k=5, α=.01, λ=.02	Train	1.279	1.173	1.158	1.203
	Test	1.322	1.01	1.176	1.169
MF, k=10, α=.01, λ=.02	Train	1.216	1.077	1.172	1.155
	Test	1.12	1.018	1.174	1.104
MF, k=20, α=.01, λ=.02	Train	1.004	1.277	1.007	1.096
	Test	1.202	1.296	1.161	1.22

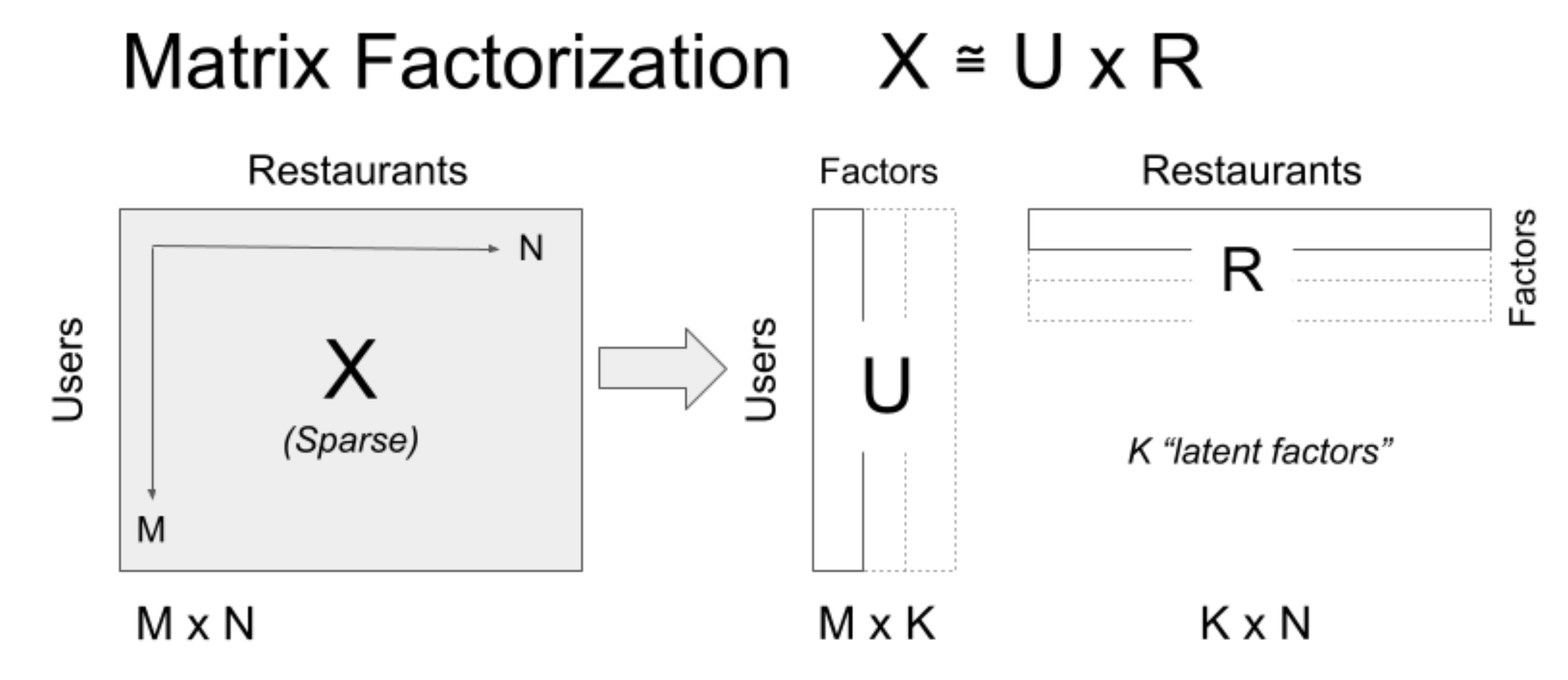


Figure 2: Matrix Factorization takes the large (sparse) matrix of users x restaurants and decomposes into two lower-rank matrices of users x latent factors and restaurants x latent factors. The rating of a user for a restaurant (a cell in the large matrix) can be computed as the inner product of the two lower-rank matrices. This improves efficiency and identifies latent factors.

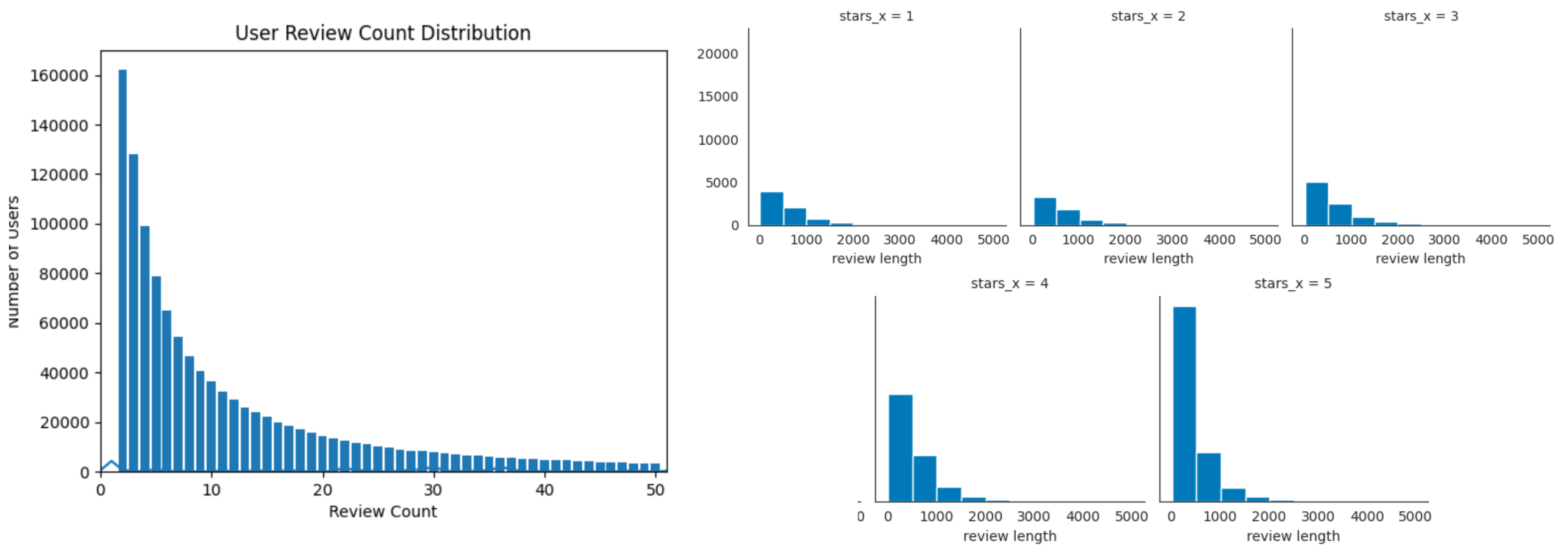
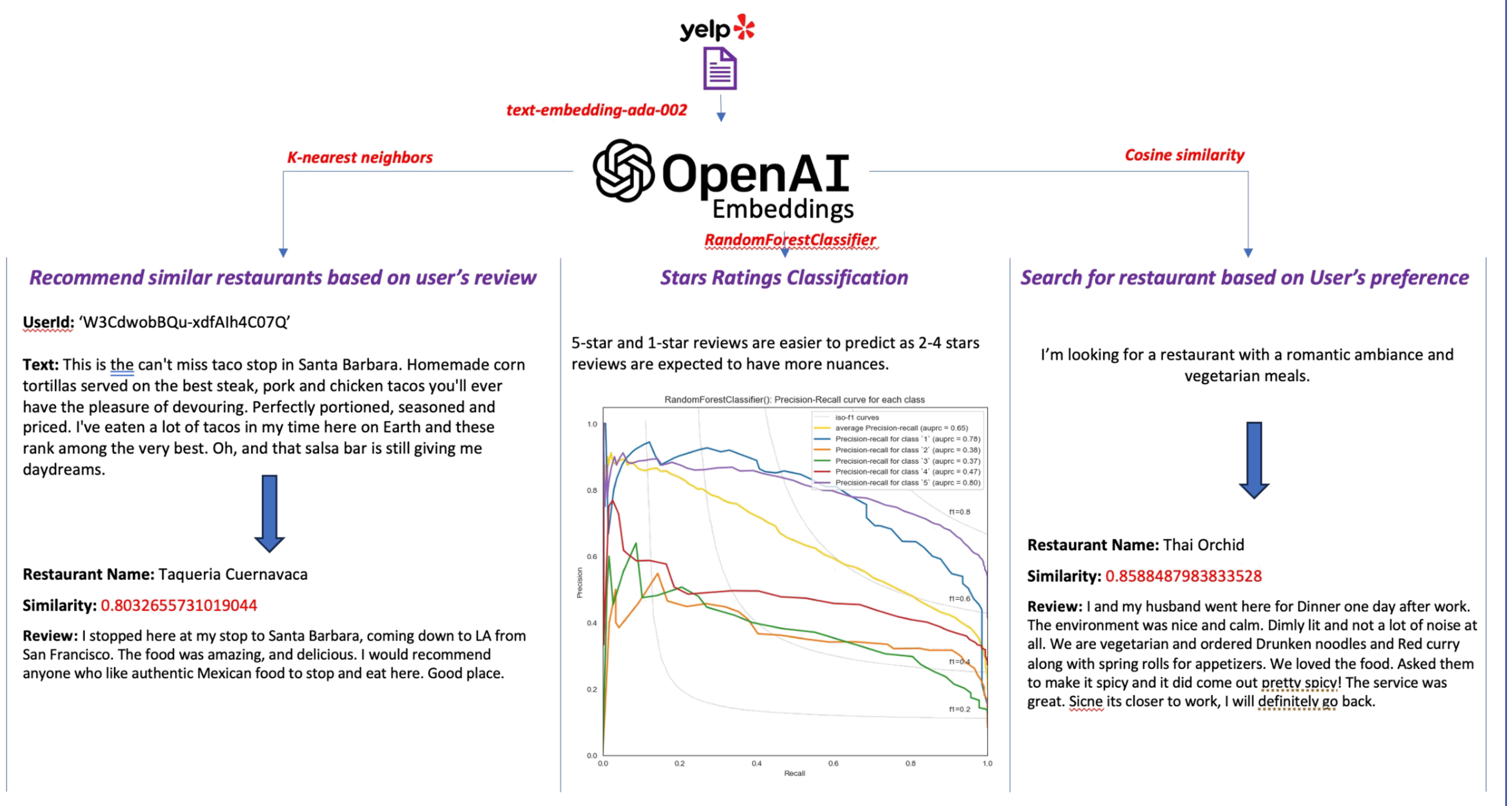


Figure 3a) distribution of number of reviews per user. Most users have left less than 10 reviews. Figure 3b) Distribution of the length of reviews per class. We see that most reviews are quite short, and that 5 star reviews are the most common reviews in the dataset.

Experiment 3: OpenAI Embeddings


yelp
text-embedding-ada-002
K-nearest neighbors
OpenAI Embeddings
RandomForestClassifier
Cosine similarity
Recommend similar restaurants based on user's review
Stars Ratings Classification
Search for restaurant based on User's preference
RandomForestClassifier(): Precision-Recall curve for each class
iso-f1 curves
average Precision-recall (auprc = 0.65)
Precision-recall for class `1` (auprc = 0.78)
Precision-recall for class `2` (auprc = 0.38)
Precision-recall for class `3` (auprc = 0.37)
Precision-recall for class `4` (auprc = 0.47)
Precision-recall for class `5` (auprc = 0.80)
Precision
Recall
UserId: 'W3CdwobBQu-xdfAlh4C07Q'

Text: This is the can't miss taco stop in Santa Barbara. Homemade corn tortillas served on the best steak, pork and chicken tacos you'll ever have the pleasure of devouring. Perfectly portioned, seasoned and priced. I've eaten a lot of tacos in my time here on Earth and these rank among the very best. Oh, and that salsa bar is still giving me daydreams.

Restaurant Name: Taqueria Cuernavaca

Similarity: 0.8032655731019044

Review: I stopped here at my stop to Santa Barbara, coming down to LA from San Francisco. The food was amazing, and delicious. I would recommend anyone who like authentic Mexican food to stop and eat here. Good place.

5-star and 1-star reviews are easier to predict as 2-4 stars reviews are expected to have more nuances.

I'm looking for a restaurant with a romantic ambiance and vegetarian meals.

Restaurant Name: Thai Orchid

Similarity: 0.8588487983833528

Review: I and my husband went here for Dinner one day after work. The environment was nice and calm. Dimly lit and not a lot of noise at all. We are vegetarian and ordered Drunken noodles and Red curry along with spring rolls for appetizers. We loved the food. Asked them to make it spicy and it did come out pretty spicy! The service was great. Sicne its closer to work, I will definitely go back.

We experiment with OpenAI's text embeddings to perform our restaurant prediction task. We use the KNN method to identify similar restaurants to what a user likes. We do this by calculating the distance between a favorable review left by a user and other favorable reviews, then recommend the top n restaurants with similar reviews.
We also searched through reviews semantically by embedding the search query and then finding the most similar reviews. We compared the cosine similarity of the embeddings of the search query and all reviews and then selected top n best matches.
Our results show a striking similarity between the input (either a user review or search parameters) and the recommended restaurants.
Finally, we also performed star classification using a Random Forest model and the OpenAI embeddings. As discovered in experiment 1, 5-star and 1-star reviews are easy to predict, while the medium ratings are more difficult.

References

[1] Behera, Gopal, and Neeta Nain. "Collaborative Filtering with Temporal Features for Movie Recommendation System." Procedia Computer Science 218 (2023): 1366-1373.
[2] Schafer, J., Frankowski, D., Herlocker, J., & Sen, S. (n.d.). Collaborative Filtering Recommender Systems. https://faculty.chas.uni.edu/~schafer/publications/CF_AdaptiveWeb_2006.pdf
[3]Breese, J.S., D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI). 1998. Madison, Wisconsin. Morgan Kaufmann.
[4] Ajitsaria, Abhinav. Build a Recommendation Engine With Collaborative Filtering – Real Python. Realpython.com. http://realpython.com/build-recommendation-engine-collaborative-filtering/
[5] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009): 30-37.
[6]. Yelp Dataset. (n.d.). www.yelp.com. http://www.yelp.com/dataset