Exploring Restaurants' Text Reviews *vs.* Rating using a Yelp Dataset

Zeya Kong, Song Chen*, Mao Zheng
Department of Computer Science
*Department of Mathematics and Statistics
University of Wisconsin-La Crosse
La Crosse WI, 54601
mzheng@uwlax.edu

Abstract

This project aims to predict a customer's rating for a restaurant based on his/her text reviews and the ratings for other restaurants previously entered into a dataset in a restaurant recommendation system. It serves as a basis for a restaurant recommendation system, to recommend similar restaurants that a customer likes based on his/her past text reviews and ratings. This project uses an open dataset from Yelp for the restaurants only located in the state of Wisconsin due to the large volume of data. In total, the dataset contains 1619 restaurants corresponding to 26552 customers and 82510 reviews. The current system was developed as a web application running on a single PC using Python and MySQL.

Mainly in this project, we used the Collaborating Filtering Algorithm (sometimes called the nearest-neighbor algorithm) to find similar customers and similar restaurants in this project. More specifically, in order to calculate the similarity, we used three methods: Euclidean distance, Cosine similarity and Pearson correlation coefficient. We choose 200 records from the given dataset as the test set, and we calculated the Mean Absolute Error (MAE) between the prediction rating and the actual rating to evaluate the results.

We conducted a series of experiments including predicting a customer's rating for a restaurant based on his/her ratings for other restaurants in the given dataset. The prediction results based on text reviews were better than the prediction results based on the ratings alone. We also worked on improving the prediction results based on both a customer's text reviews and ratings.

Current findings show that the information behind the text is valuable and we hope to work on a more precise text-based recommendation system in the future.

1 Introduction

Recommendation engine is one of the hottest areas in the field of artificial intelligence. A recommendation engine is a kind of model that can generate a recommendation for a user based on his/her information. There are many impressive recommendation engines such as Yelp, Amazon, Google, LinkedIn, etc. Those systems generate the recommendations based on their own datasets that are usually private. Fortunately, the yelp company has some data competitions that provide open dataset for developers to use in academic. We plan to use the yelp dataset to explore some natural language processing work. This project aims to predict a customer's rating for a restaurant based on his/her text reviews and ratings for other restaurants previously entered into a dataset in a restaurant recommendation system. It serves as a basis for a restaurant recommendation system, to recommend similar restaurants that a customer likes based on his/her past text reviews and ratings. This project uses an open dataset from Yelp for the restaurants only located in the state of Wisconsin due to the large volume of data. In total, the dataset contains 1619 restaurants corresponding to 26552 customers and 82510 reviews. The current system was developed as a web application running on a single PC using Python and MySQL.

Mainly in this project, we used the Collaborating Filtering Algorithm (sometimes called the nearest-neighbor algorithm) to find similar customers and similar restaurants in this project. More specifically, in order to calculate the similarity, we used three methods: Euclidean distance, Cosine similarity and Pearson correlation coefficient. We choose 200 records from the given dataset as the test set, and we calculated the Mean Absolute Error (MAE) between the prediction rating and the actual rating to evaluate the results.

We conducted a series of experiments including predicting a customer's rating for a restaurant based on his/her ratings for other restaurants in the given dataset. The prediction results based on text reviews were better than the prediction results based on the ratings alone. We also worked on improving the prediction results based on both a customer's text reviews and ratings.

2 The Collaborating Filtering Algorithm

In this section, we will briefly introduce the Collaborating Filtering algorithm used in our project.

The collaborative filtering algorithm (sometimes called the nearest-neighbor algorithm) is one of the most basic algorithms in the recommendation system area. In this application, we implemented the collaborative filtering algorithm by using star ratings as well as the text reviews.

The collaborative filtering algorithms are divided into three categories: 1) the user-based collaborative filtering algorithm, 2) the item-based collaborative filtering algorithm and 3) the model-based collaborative filtering algorithm. In this project, we only focused on

first two algorithms because the model-based collaborative filtering algorithm is too complex, and it requires different types of data such as the user's search history that is not included in the dataset we obtained from yelp.

2.1 User-based Collaborative Filtering

The **U**ser-based **C**ollaborative **F**iltering (UserCF) algorithm mainly includes two steps:

- For a target user, finding a collection of users who have similar interests.
- Recommending the items that are the most popular/high-rated for the users of the collection. However the target user did not choose those items before.

So, if we can find the similar users for a target user, we can easily recommend the item that has the highest average rating/most popular to the target user from the UserCF algorithm.

2.2 Item-based Collaborative Filtering

The **Item**-based **C**ollaborative **F**iltering algorithm (ItemCF) gives users recommendations for items that are similar to the items they liked before.

The item-based collaborative filtering algorithm is mainly divided into two steps:

- For all items, finding the relationship between the two.

For a target user, finding items his/her liked and recommend similar items for this user.

2.3 Similarity Calculation

There are many ways to calculate the similarity between two different users or items. In order to compare the two different things, we transform the user/item information into a one-dimensional numeric vector. In this project, there are two different ways to represent a user/item as a vector. One is to use the star rating information to generate the vector. The other is to use the text-review information to generate the vector.

In this project, we mainly used three different methods to calculate the similarity.

2.3.1 Euclidean Distance

The Euclidean distance or Euclidean metric is the "ordinary" straight-line distance between two points in Euclidean space. We can use the Euclidean distance to calculate how far the two vectors are. The closer the two are, the more similar.

For n-dimension vector q and p.

$$d(\mathbf{p},\mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}.$$

where p_1, p_2 ,…, p_n are the value of the current dimension of vector p, same as q_1, q_2 ,…, q_n .

2.3.2 Cosine Similarity

The cosine can represent the angle between the two vectors. Smaller the angle, more similar. Here is the formula:

$$\cos\theta = \frac{x_1 x_2 + y_1 y_2}{\sqrt{x_1^2 + y_1^2} \times \sqrt{x_2^2 + y_2^2}}$$

where vector1: (x_1, y_1), vector2: (x_2, y_2)

2.3.3 Pearson Correlation Coefficient

The Pearson Correlation Coefficient and the Cosine Similarity have the similar information, and both are invariant to scaling. The Pearson correlation is also invariant to add any constant to all elements [3]. The formula is:

$$r = \frac{\sum_{i=1}^{n} (X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n} (X_i - \overline{X})^2} \sqrt{\sum_{i=1}^{n} (Y_i - \overline{Y})^2}}.$$

where Xi: the value in ***i***th position of the first vector; Yi: the value in ***i***th position of the first vector; $\overline{X}$ and $\overline{Y}$ are the mean value.

3 Build and Test Model

We implemented the user-based collaborative filtering algorithm and item-based collaborative filtering algorithm by using Euclidean distance, cosine similarity and Pearson correlation coefficient as the similarity calculation method.

We started the model with two parts: rating-based only and text-based only.

3.1 Rating-based model

From Yelp data, we have business table, user table and review table. These tables allow us to generate the customer-restaurant-rating matrix as shown in Table1:

	Customer 1	Customer 2	…	Customer n
Restaurant 1	1 (rating)	5	…	3
Restaurant 2	3	(empty)	…	4
…	…	…	…	…
Restaurant n	6		…	

Table 1. The customer-restaurant-rating matrix example

- **User-based Collaborative Filtering Recommendation**

In Table 1, for example, we can choose the whole column of customer 1 which is a one-dimensional vector. Similarly, we can choose the customer 2 column as a vector. The similarity of these two vectors can be calculated by using above one of the three similarity mathematical formulas. This model will calculate the similarity using all three methods to find which similarity calculation is the best one for this dataset.

- **Item-based Collaborative Filtering Recommendation**

The only difference between the UserCF and the ItemCF is we choose a whole row for one restaurant as a vector and calculate the similarity between the different restaurants in the ItemCF. For example, in Table 1, we can get restaurant 1's row and restaurant 2's row as two one-dimensional vectors. By using the same approach to calculate the similarity for the two vectors we can determine the degree of the similarity.

3.2 Text-based model

Compared to the rating-based model, the text-based model is very similar. The only difference between the rating-based model and text-based model is the similarity calculation. For the rating-based model, we generated a restaurant-customer-rating matrix and selected columns or rows to calculate the similarity. But for the text-based model, we could not use the previous rating matrix. We tried to generate a new matrix that contains the information of the text review instead of the restaurant-customer-rating matrix. Basically we need to convert text information into numeric information in the matrix.

Here are the steps used to generate the matrix:

1) For all text reviews (in our project, the total text reviews are 80000+), find the top 2000 words that appear most frequently

   [like, dislike, taste, happy, …….]

2) Generate the words-frequency vector for each text review, see Figure 1 below.

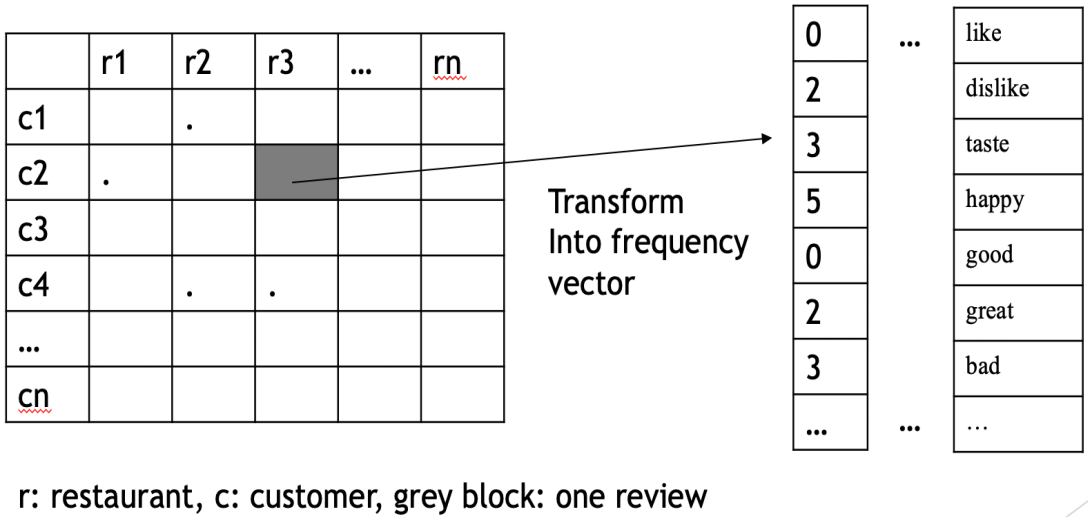

Figure 1 Word Frequency Vector

3) Then we compared the different words-frequency vector to calculate the similarity as shown in Figure 2 below.

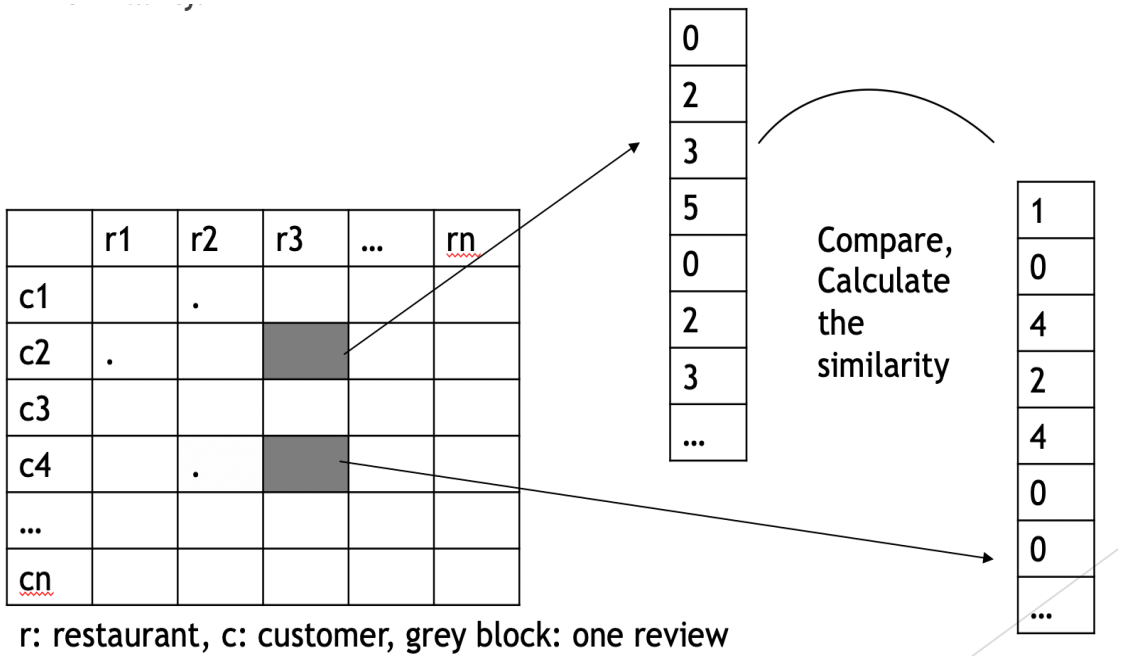

Figure 2 Calculate the similarity

After we found similar users or similar restaurants, we can predict a customer's rating for

a restaurant by calculating the average score or frequency.

Once the model is built, we decided to use an approach called cross validation to test our model.

3.3 Test Model

The aim of cross validation is to eliminate the impact of the data overfit when we conduct testing. The dataset is divided into two parts: the training dataset and the testing dataset. The two parts are independent, which means when you test your model by using testing dataset, the testing dataset does not have any influence on the training dataset.

In our project, we randomly chose 200 samples from the whole dataset as the test dataset. One sample is a record that contains a user's rating as well as the text review for the restaurant. For each test, we hid the real rating star and the text review corresponding to the rating and used collaborative filtering algorithms to predict the rating. The variance of the real rating compared to the predicted rating can reflect the accuracy of the model. For all of the testing, the test dataset is fixed, which allows us to compare the different algorithms and different similarity calculations.

There are two standard ways to calculate the variance between the predicted value and the real value. **M**ean **A**bsolute **E**rror (MAE) and **R**oot **M**ean **S**quare **E**rror (RMSE) are commonly used by many machine-learning projects to measure their model's accuracy. Both are used to calculate average error for the entire testing. The difference between those two is that MAE calculates the absolute error but RMSE calculates the squared error for each testing, which means RMSE is more sensitive when there is a larger difference between the real value and the predicted value. In our project, we decided to use MAE to measure our model because the testing dataset contains some outliers which could be magnified by using RMSE to calculate. In addition, because the range of each error is from zero to five, MAE is more suitable for this range. The smaller MAE value, the smaller of the variance between the prediction and real value. Hence the smaller of the MAE, the better of the prediction.

4 Experiments Results

We conducted total of twenty seven testing, the results are shown in Table 2:

Test case #	Test size	Type	Similarity	Words Vec type	Top Words #	Time(s)	MAE
1	200	rating	Euclidean			0.1466	1.723
2	200	rating	Cosine			0.1532	1.727
3	200	rating	Pearson			0.1671	1.728
4	200	text	Euclidean	Bag-of-words	200	0.1929	1.667
5	200	Text	Euclidean	Bag-of-	500	0.1959	1.659

				words			
6	200	Text	Euclidean	Bag-of-words	1000	0.1892	1.656
7	200	Text	Euclidean	Bag-of-words	2000	0.1885	1.651
8	200	Text	Cosine	Bag-of-words	200	0.1954	1.662
9	200	Text	Cosine	Bag-of-words	500	0.1935	1.656
10	200	Text	Cosine	Bag-of-words	1000	0.1887	1.651
11	200	Text	Cosine	Bag-of-words	2000	0.1986	1.651
12	200	Text	Pearson	Bag-of-words	200	0.2134	1.656
13	200	Text	Pearson	Bag-of-words	500	0.1950	1.653
14	200	Text	Pearson	Bag-of-words	1000	0.2051	1.650
15	200	Text	Pearson	Bag-of-words	2000	0.1970	1.649
16	200	Text	Euclidean	TF-IDF	200	0.1929	1.660
17	200	Text	Euclidean	TF-IDF	500	0.1959	1.657
18	200	Text	Euclidean	TF-IDF	1000	0.1892	1.654
19	200	Text	Euclidean	TF-IDF	2000	0.1885	1.653
20	200	Text	Cosine	TF-IDF	200	0.1954	1.663
21	200	Text	Cosine	TF-IDF	500	0.1935	1.657
22	200	Text	Cosine	TF-IDF	1000	0.1887	1.655
23	200	Text	Cosine	TF-IDF	2000	0.1986	1.654
24	200	Text	Pearson	TF-IDF	200	0.2134	1.655
25	200	Text	Pearson	TF-IDF	500	0.1950	1.651
26	200	Text	Pearson	TF-IDF	1000	0.2051	1.646
27	200	Text	Pearson	TF-IDF	2000	0.1970	1.649

Table 2 The testing result for the different approaches

- **Type**: the type of the collaborative filtering (rating-based or text-based)
- **Similarity**: the method to calculate the similarity between two vectors
- **Words Vec type**: The type of the word vector for text-based recommendations
- **Top Words #:** the length of the top frequency words vector
- **Time(s):** the average time cost per test case. (unit: seconds)
- **MAE：** mean absolute error
- We used bag-of-words and TF-IDF words matrixes in the Words Vect type

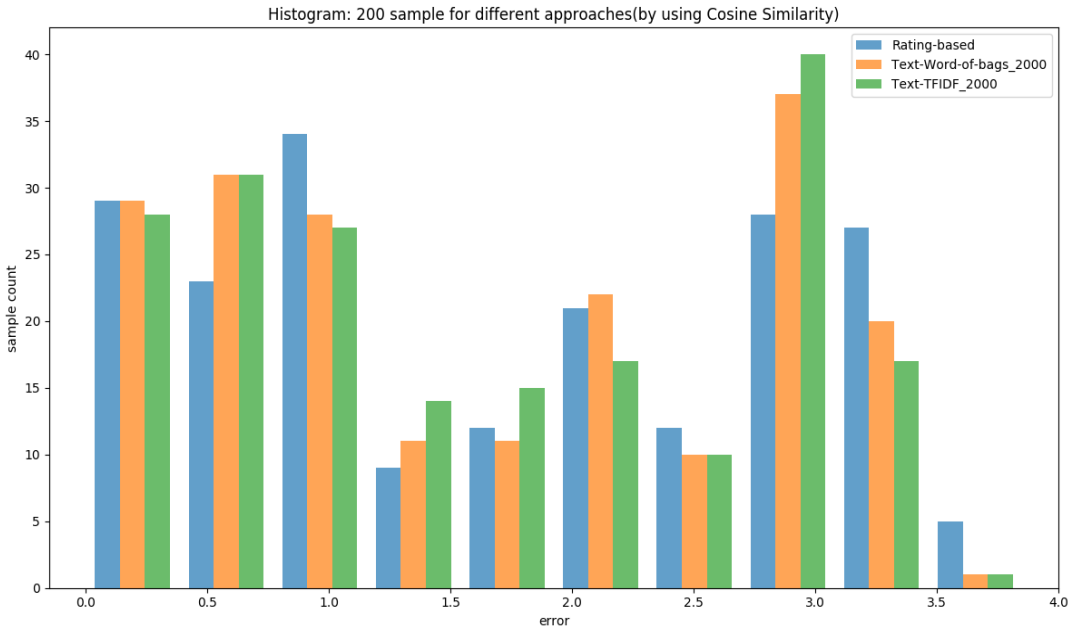

Figure 3 The Error Distributions for Rating-based and Text-based Approaches
x-axis: the error between the real value and the predict value (from 0 to 4); y-axis: the number of samples in which the error is in the current range.

Mean Absolute Error (MAE):

```
rating-based(Cosine): 1.7274787003615506
text-based(words-of-bags,Cosine): 1.651312033694885
text-based(TFIDF,Cosine): 1.654228700361551
```

6 Conclusions

In Figure 3, x-axis value is the variance between prediction and real value. We have discussed the smaller MAE value, the smaller of the variance between the prediction and real value, the better of the prediction. When the x-axis value is small, using text-based approaches have more records; when the x-axis value is bigger than 3.5, using text-based approaches have less records. That means using text-based approaches have better predictions.

Every record from the yelp dataset contains the rating star and the text review. From the testing result we can draw a conclusion, the information behind the text is valuable and our project can use that information to do more precise text-based recommendation. In order to improve our project, we considered using some deep learning tools such as **Word2vec** and **Sentence2vec / Doc2vec** to process our text reviews as the next step.

References

[1] https://en.wikipedia.org/wiki/Machine_learning.
[2] Wikipedia, “Collaborative filtering”
https://en.wikipedia.org/wiki/Collaborative_filtering
[3] Linden Greg,Smith Brent and York Jeremy, “Amazon.com Recommendations: Item-to-Item Collaborative Filtering.” , IEEE Internet Computing, 2003.
[4] https://stackoverflow.com/questions/1838806/euclidean-distance-vs-pearson-correlation-vs-cosine-similarity