Hide Code

IV. Objective 3: Recommendation System

In this part, I will build recommendation system with trying three popular recommendation methods and one deep learning approach(Artificial Nerual Network). The target of this part is to recommend 3 more different restaurants to each customer in the Hoboken Restaurants Reviews dataset. The recommendation system will help business owner as well as yelp to create profit. The methods I will try showed as follows:

1. Co-occurrence Matrix
2. Collaborative Filtering
3. Matrix Factorization
4. Deep learning (Neural Network)

Part 1. Data Understanding and Cleaning

Before building the model, I will introduce the data I will use. The raw data in this part is the Hoboken Restaurants Reviews dataset, which includes 74,611 rows and 8 columns. First, I will clean the data and prepare required data for following recommendation system methods.

Part 1 - 1 Basic Understanding of data

Definition of each variable:

1. user_id: Unique id for each customer
2. user_name: customer's name
3. user_raing: original rating for one restaurant per review
4. user_text: text of customer's review for one restaurant per review
5. restaurant_name: unique name for each restaurant
6. restaurant_rating: the integrated rating of each restaurant
7. restaurant_price: degree of cheap or expensive of one restaurant
8. restaurant_type: the style and theme of one restaurant

Table 1 show a sample dataset and basic information of raw data. **Chart 1** shows the datatype of each column. There are some columns like user_rating, restaurant_rating and restaurant_price should be convert into numerical data. In addition, there are some repeated comma in restaurant_type column, so I will do data cleaning first and try to use this variable in Neural Network part. In addition, it use '$$' symbol represent the degree of price of each restaurant and the symbol should be converted to integer, which could be recognized by the computer.

. . .

	user_id	user_name	user_rating	user_text	restaurant_name	restaurant_rating	restaurant_price	restaurant_type
0	dRuCO4NYO7zyAF8-CeJmZg	Jason L.	5.0 star rating	We booked Grand Vin as our brunch location to ...	Grand Vin	4.0 star rating	$$	Wine, Bars,, Italian,, Cocktail, Bars
1	f36YZ1cA291bNtMHXWtu1Q	Danyale W.	4.0 star rating	Sooooo for date night it was his turn to pick ...	Grand Vin	4.0 star rating	$$	Wine, Bars,, Italian,, Cocktail, Bars
2	-xYUKfWQTaB-7BeizsQA3w	Robin G.	5.0 star rating	Adorable little wine bar with outdoor seating ...	Grand Vin	4.0 star rating	$$	Wine, Bars,, Italian,, Cocktail, Bars
3	tt1vLgAP5UpRXAKJLT2KWg	Alec K.	4.0 star rating	One of the top restaurants in Hoboken. Well ma...	Grand Vin	4.0 star rating	$$	Wine, Bars,, Italian,, Cocktail, Bars
4	-K79Xep4lElqIChsJYWuiQ	Robbie O.	5.0 star rating	Great space-service is on point - short rib ...	Grand Vin	4.0 star rating	$$	Wine, Bars,, Italian,, Cocktail, Bars

Table 1: Sample of raw data

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74611 entries, 0 to 74610
Data columns (total 8 columns):
user_id              74611 non-null object
user_name            74611 non-null object
user_rating          74611 non-null object
user_text            74611 non-null object
restaurant_name      74611 non-null object
restaurant_rating    74611 non-null object
restaurant_price     74479 non-null object
restaurant_type      74611 non-null object
dtypes: object(8)
memory usage: 4.6+ MB
```

Chart 1: Datatype

We could find some basic information of the raw data from **Table 2**. It shows that there are 74,611 reviews in total, while only 44,949 unique users leave the reviews, which means multiple users attended to more than one restaurants and it is valuable to build the recommendation system. Also, it shows that there are only 24,781 unique user's name, so in order to reduce the ambigurity, I will use user_id instead of user name in the recommendation system.

	user_id	user_name	user_rating	user_text	restaurant_name	restaurant_rating	restaurant_price	restaurant_type
count	74611	74611	74611	74611	74611	74611	74479	74611
unique	44949	24781	5	69542	302	7	4	230
top	QxTNaHoxTL8f7jAK5nwQ1g	Nicholas G.	5.0 star rating	I've been coming to STK downtown since they ha...	Morimoto	4.0 star rating	$$	Japanese,, Sushi, Bars
freq	101	103	27702	3	2740	31568	38323	3620

Table 2: Basic Information

After checing the null value, we could find that there are null value in restaurant_price column. **Table 3** shows that there are some missing restaurant_price for some restaurants, since yelp have not updated the price for these restaurants. In following part, I will refill these missing value accordindly.

```
user_id              False
user_name            False
user_rating          False
user_text            False
restaurant_name      False
restaurant_rating    False
restaurant_price      True
restaurant_type      False
dtype: bool
```

	user_id	user_name	user_rating	user_text	restaurant_name	restaurant_rating	restaurant_price	restaurant_type
5273	i_WxJpoxfsddmBne6l8cKQ	Durva L.	4.0 star rating	This is such a small place, but some great rea...	Funjabi	2.5 star rating	NaN	Indian,, Chinese
5274	zr0jkmEJLnaZxWZi7cp09Q	Wasbir R.	1.0 star rating	Giving 1 star because theres no 0 star. This i...	Funjabi	2.5 star rating	NaN	Indian,, Chinese
6782	cnNddUd4fn7h-Pb_Cma-9Q	mako y.	5.0 star rating	Wonderful!!! It was traditional with unique tw...	奥田	4.5 star rating	NaN	Japanese
6783	q1P19WvpTXFTmfU7oXqkaq	Chester G.	2.0 star	When I heard Chef Toru Okuda	奥田	4.5 star rating	NaN	Japanese

```
number of missing value: 132
```

Table 3: Missing value

Part1 - 2 Data Cleaning and Preprocessing

In this part, I will check each column one by one and prepare the dataset for recommendation system part.

- user_id

Table 4 list the top 10 users who left most reviews to restaurants in Hoboken. Multiple customers left reviews for different restaurants more than one time which is valuable for the recommendation system.

	user_id	count
0	QxTNaHoxTL8f7jAK5nwQ1g	101
1	5aZX8bTiD0k9vR60SG588Q	76
2	GRY2acZtI5q4P1KdCWhcUQ	66
3	LcWOq7p7Mhtv9hIlDrhy9A	62
4	Y59HQNazSLR1EUMUmMOaFg	61
5	NWLqOKl0Vxi7qK-6EoBLbg	60
6	A5dqSwriUs8cV4DEzS_V9A	59
7	svFycHjXZYpNVutZ_0_gDQ	59
8	dl4ENy4Bk6-lCu59A8vxbg	57
9	94V-snVUg2Gd-OPd7rcZEg	54

Table 4: Top 10 users left most reviews

- user_name

Since different customers might have same name, in order to reduce the ambigurity, I will use user_id instead of user name in the recommendation system. There is nothing to do with the user_name column.

- user_rating

I will convert the string into numerical format as I mentioned and include user_rating variable to calculate the weights or scores the given customer rated. **Table 5** shows the count of number of each rating star in the Hoboken_restaurants_reviews dataset.

	rating	count
0	5	27702
1	4	23140
2	3	10916
3	1	6608
4	2	6245

Table 5: Count of rating

- Restaurant_name

The restaurant's names are unique in this dataset. So I will use the restaurant name represent each restaurant in the following part. There are 302 different restaurants in the dataset in total. **Table 6 and Table 7** shows that some restaurants received more than 2 thousands revies in Yelp and some only received 1 or 2. This information might influence the accuracy of the recommendation system, since some types of recommendation methods will be limited by data sparsity. A restaurant with too seperated reviews might not be selected as a recommended result.

```
Number of restaurants: 302
```

	restaurant_name	count
177	Morimoto	2740
261	The Spotted Pig	2640
153	Los Tacos No.1	2100
258	The Park	1640
287	Wafels & Dinges	1500
46	Catch	1480
90	Employees Only	1300
15	Artichoke Basille's Pizza & Bar	1300
71	Del Posto	1260
93	Fig & Olive	1220

Table 6: Restaurants received most reviews

	restaurant_name	count
165	Manhattan Bar Grill & Lounge	1
18	Azteca Taqueria Restaurant	1
159	MONDO - DOGS	1
81	El Cantante	1
289	Wah Yoan	1
92	Field House Grill	1
1	52 Restaurant	1
87	El Salvador Restaurant	1
240	Taquitos Mexicanos	2
99	Funjabi	2

Table 7: Restaurants received least reviews

- restaurant_rating

I will convert the restaurant_rating column into numerical data. **Table 7** shows the distribution of different rating scores. The restaurant_rating variable might influence the customer's decision making, so I will take this variable into account in the Neural Network method.

```
['4.0 star rating' '4.5 star rating' '3.5 star rating' '3.0 star rating'
 '2.5 star rating' '5.0 star rating' '1.5 star rating']
```

	restaurant_rating	count
4	4.0	31568
3	3.5	26130
5	4.5	8866
2	3.0	7263
6	5.0	484
1	2.5	267
0	1.5	33

Table 7: Rating Distribution

- Restaurant_price

I will convert the restaurant_price column into numerical data. **Table 8** shows the distribution of degree of restaurant's price. We could find that the level 2 degree is the most popular degree of restaurant's price.

```
['$$' '$' '$$$' nan '$$$$']
```

	restaurant_price	count
1	2.0	38323
2	3.0	19809
0	1.0	10567
3	4.0	5780

Table 8: Price Distribution

The restaurant_price variable might influence the customer's decision making. There are some missing restaurant_price for some restaurants, since yelp have not updated the price for these restaurants. I will use 2 to refill these cells according to the statistical result that there are most restaurants with restaurant price in level 2.

. . .

- Restaurant_type

I will do data cleaning for the restaurant_type column. The restaurant_type variable is significant to influence customer select a restaurant. So I will focus on this variable in the Neural Network method. **Table 9** shows the most popular restaurant type in this dataset, 4000 users have left reviews to a 'sushi japanese bars' restaurant.

. . .

	restaurant_type	count
204	sushi bars japanese	4080
169	pizza italian	3348
133	italian	3070
151	mexican	2988
122	gastropubs burgers	2640
98	cocktail new spaces venues american bars event	1640
116	food belgian trucks waffles	1500
186	seafood bars fusion sushi asian	1480
10	american new bars	1451
103	cuban	1320

Table 9: Restaurant Type with most reviews

Now, I finish data cleaning and preprocessing part. I will store the cleaned dataset into 'Hoboken_restaurants_reviews_cleaned.csv'. This dataset will also be used in the method 4 - Neural Network methods.

	user_id	user_name	user_rating	user_text	restaurant_name	restaurant_rating	restaurant_price	restaurant_type
0	dRuCO4NYO7zyAF8-CeJmZg	Jason L.	5	We booked Grand Vin as our brunch location to ...	Grand Vin	4.0	2.0	cocktail bars wine italian
1	f36YZ1cA291bNtMHXWtu1Q	Danyale W.	4	Sooooo for date night it was his turn to pick ...	Grand Vin	4.0	2.0	cocktail bars wine italian
2	-xYUKfWQTaB-7BeizsQA3w	Robin G.	5	Adorable little wine bar with outdoor seating ...	Grand Vin	4.0	2.0	cocktail bars wine italian
3	tt1vLgAP5UpRXAKJLT2KWg	Alec K.	4	One of the top restaurants in Hoboken. Well ma...	Grand Vin	4.0	2.0	cocktail bars wine italian
4	-K79Xep4lElqIChsJYWuiQ	Robbie O.	5	Great space-service is on point - short rib ...	Grand Vin	4.0	2.0	cocktail bars wine italian

Table 10: Clean dataset and Dataset for Method 4

Part 2. Data Preparation

In this part I will prepare the data for co-occurence, collaborative filtering and matrix factorization methods. I will only keep the user_id, restaurant_name and user_rating column to build 3 basic recommendation system. **Table 11** shows a sample of this dataset.

	user_id	restaurant_name	user_rating
0	dRuCO4NYO7zyAF8-CeJmZg	Grand Vin	5
1	f36YZ1cA291bNtMHXWtu1Q	Grand Vin	4
2	-xYUKfWQTaB-7BeizsQA3w	Grand Vin	5
3	tt1vLgAP5UpRXAKJLT2KWg	Grand Vin	4
4	-K79Xep4lElqIChsJYWuiQ	Grand Vin	5

Table 11: Sample of dataset

Customers might rate more than one time for one restaurant. I will use the average rating scores for the cases that customer rated more than one time. This group-by table, **Table 12**, will also be used in collaborative filtering recommendation system.

	user_id	restaurant_name	user_rating
0	---xAZNw9fFPBoy7jmkA2A	Chef Of India	4
1	--68ZwhCrUJUmCXXkMTMKw	Prime Food Market	5
2	--8M2DZ9JkDwTveuRhLPTQ	Del Posto	5
3	--ARr3m5JsxaX3DTUVQW7w	Morimoto	2
4	--CZJeSlpxwQ0VULjnM57w	The Brass Rail	1

Table 12: A group-by Dataset(Method 2)

Based on the group-by dataset, I use pivot() function to convert it into a matrix like dataset. Each row represent each user and each column represent each restaurant. **Table 13** shows the sample of the matrix like dataset, which will be used for method 3, the matrix factorization recommendation system.

restaurant_name	10th & Willow Bar & Grill	52 Restaurant	8th Street Tavern	Adoro Lei	Aether Game Cafe	Ahri's Kitchen	Ainsworth Hoboken	Aldys Restaurant	Ali Baba	Amanda's Restaurant	...	White Star Bar	Wicked Wolf Tavern
user_id													
---xAZNw9fFPBoy7jmkA2A	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--68ZwhCrUJUmCXXkMTMKw	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--8M2DZ9JkDwTveuRhLPTQ	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--ARr3m5JsxaX3DTUVQW7w	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--CZJeSlpxwQ0VULjnM57w	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0

5 rows × 302 columns

Table 13: A Matrix Like dataset with rating(Method 3)

The **Table 14** is similar with the **Table 13**, the only difference is that I convert all of the rating scores into 1, which represent the customers attended to the restaurants.

restaurant_name	10th & Willow Bar & Grill	52 Restaurant	8th Street Tavern	Adoro Lei	Aether Game Cafe	Ahri’s Kitchen	Ainsworth Hoboken	Aldys Restaurant	Ali Baba	Amanda’s Restaurant	...	White Star Bar	Wicked Wolf Tavern
user_id													
---xAZNw9fFPBoy7jmkA2A	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--68ZwhCrUJUmCXXkMTMKw	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--8M2DZ9JkDwTveuRhLPTQ	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--ARr3m5JsxaX3DTUVQW7w	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
--CZJeSlpxwQ0VULjnM57w	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0

5 rows × 302 columns

Table 14: A matrix Like dataset without rating(Method 1)

After preparing the dataset for following part, I will store these dataset as 'Method_1_dataset.csv', 'Method_2_dataset.csv' and 'Method_3_dataset.csv'.

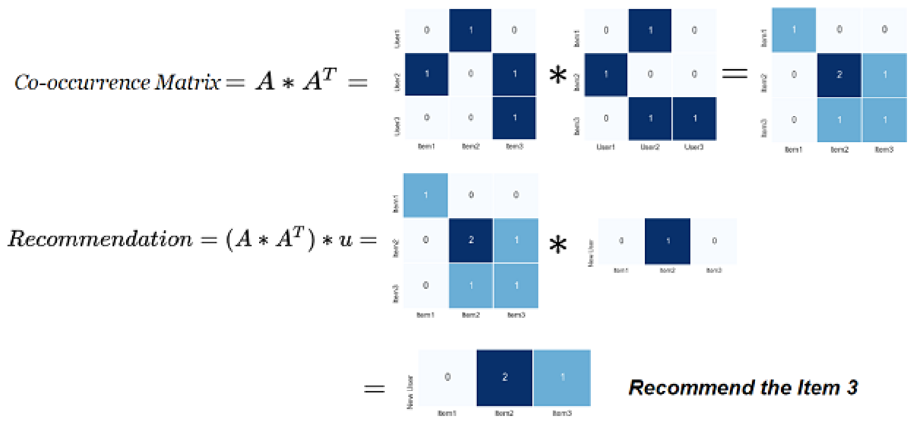
Part 3 Recommendation System Methods:

In this part I will use following methods to build recommendation system model:

- co-occurrence matrices
- collaborative filtering
- matrix decomposition
- Neural Network

Part3 - 1 co-occurrence Matrices

- Background: This is the simpliest method I use to build the recommendation system. The underlying assumption is that the customers will be interested in the restaurants that the other customers who have similar interest.
- Algorithms/equations:
  - Co-occurrence Matrix = A.T * A
  - Recommendation = Co-occurrence Matrix * u
    Following chart will show a simple sample to apply co-occurrence matrix for recommendation system.



Co-occurrence Matrix = A * A^T =
Recommendation = (A * A^T) * u =
Recommend the Item 3
User1
User2
User3
Item1
Item2
Item3
New User

    'A' represent the user-item matrix, which summarizes the interactions between users and items. The matrix is just from the raw data, with one row for each user, and one column for each item. 'A.T' is the transpose of this matrix. In 'A.T', rows means item and the column means user. In the first step, we should multiple 'A' and transposed 'A' and will get the co-occurrence Matrix, which is also the recommender to recommend the item to user. Assume that we have a new user, this new user have bought the item 2. We multiple the co-occurrence matrix and the array represent the new user. Finally, we will obtain the recommendation result which shows we should recommend the Item 3 to this new user. The reason why we ignore the item 2 is that this user have already bought it.
- Tools:
  - Specifically, I will use numpy, pandas in python to build the recommendation model.

- Step 1: Data Loading

The data I input is the **Table 14**, a matrix like dataset without rating scores.

- Step 2: build co-occurrence matrix

I use numpy to convert the matrix like dataset into numpy matrix and then obtain the transpose of this numpy matrix. Now I have two matrices, one original matrix and one transposed matrix. I multiple these two matrices and then eliminate the value in the diagonal cell of the result matrix to avoid repeadedly take attended restaurants into account. And then, I obtain the co-occurrence matrix like **chart 2**.

```
matrix([[ 0.,  0.,  9., ..., 11.,  3.,  0.],
        [ 0.,  0.,  0., ...,  0.,  0.,  0.],
        [ 9.,  0.,  0., ...,  6.,  0.,  0.],
        ...,
        [11.,  0.,  6., ...,  0., 13.,  0.],
        [ 3.,  0.,  0., ..., 13.,  0.,  0.],
        [ 0.,  0.,  0., ...,  0.,  0.,  0.]])
```

Chart 2: co-occurrence matrix

- Step 3: Recommend restaurant for each user

In order to reach our target, to provide 3 more restaurants to each customer, I build a function could use co-occurrence matrix to provide recommended restaurants for each user. The underlying method is multiple the numpy array represented each user and the co-occurrence matrix we obtained in step 2.

I designed two options to select the user, user's index in the dataset or user's id. Following list show a sample result to recommend 3 more restaurants for a user with index 2. For this user, Morimoto is the first restaurant we will recommend with highest scores in 118.0 and so on.

```
[['1', 'Morimoto', 118.0],
 ['2', 'The Spotted Pig', 106.0],
 ['3', 'EN Japanese Brasserie', 66.0]]
```

In addition, I build a function with using for loop to recommend the restaurant for each user in the dataset. So the co-occurrence matrix could provide recommend for each customer in the Hoboken_restaurants_reviews_cleaned.csv dataset.

- Step 4: Check the result

Table 15 shows a sample of the recommendation result dataset for co-occurrence matrix method. Each user will receive recommendation of restaurants specifically.

	user_id	recommendation
0	---xAZNw9fFPBoy7jmkA2A	[['1', 'Gino’s Pizzeria', 9.0], ['2', 'Noodlef...
1	--68ZwhCrUJUmCXXkMTMKw	[['1', 'The Cheesecake Factory', 3.0], ['2', '...
2	--8M2DZ9JkDwTveuRhLPTQ	[['1', 'Morimoto', 118.0], ['2', 'The Spotted ...
3	--ARr3m5JsxaX3DTUVQW7w	[['1', 'The Spotted Pig', 143.0], ['2', 'EN Ja...
4	--CZJeSIpxwQ0VULjnM57w	[['1', 'La Isla Restaurant', 48.0], ['2', 'Ama...

Table 15: co-occurrence matrix result

These are two samples of recommendation result for the user with id '---xAZNw9fFPBoy7jmkA2A' and '--ARr3m5JsxaX3DTUVQW7w'. The model recommended three more restaurants for each user which stored in the recommendation column. Each cell is a list include three results. The first item (1, 2, 3) in the list means the ranking, the second item represent the name of restaurant, and the third item(9.0, 8.0, 6.0) means the weights/scores for each restaurant.

The weights/scores for different users are in different range according to the shape and values of co-occurrence matrix. For example, for the user with id '---xAZNw9fFPBoy7jmkA2A', the weights are 9.0, 8.0 and 6.0, while the weights are 143.0, 127.0 and 118.0 for the user with id '--ARr3m5JsxaX3DTUVQW7w'.

```
[['1', 'Gino’s Pizzeria', 9.0], ['2', 'Noodlefan', 8.0], ['3', 'Fox and Crow', 6.0]]
```

```
[['1', 'The Spotted Pig', 143.0], ['2', 'EN Japanese Brasserie', 127.0], ['3', 'Del Posto', 118.0]]
```

co-occurrence matrix summary:

In general, the co-occurrence matrix is easier to build than the other methods and it could build with dataset in any size, so it could be used for sparse dataset. However, the limitations for co-occurrence matrix is obvious. For example, It can only be built based on the user's past behaviors and it will ignore the user's rating for restaurants.

Part 3 - 2 Collaborative Filtering

- My second recommendation system is collaborative filtering recommednation system. This method is tring to find similar users or items, evaluate the scores of those users/items, and recommend given user the items they will probably rate high. There are two different approaches in Collaborative Filtering: user-based collaborative filtering and item-based collaborative filtering. Basically, all of those two methods contains two steps:
  - First Step: Find out how many users/items in the database are similar to the given user/item.
  - Second Step: Assess other users/items to predict what grade you would give the user of this product, given the total weight of the users/items that are more similar to this one. I will include the user-based collaborative filtering in this project.
- Algorithms/Equations:
  - Similarity Calculation: The first step for collaborative filtering is calculate the similarity between the given user and the users in the dataset. There are multiple different methods to calculate the similarity and I will compare three popular methods, cosine similarity, jaccard similarity and pearson similarity.

    1. cosine similarity:

    $$\text{Cosine similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}}$$

    2. Jaccard Similarity:

    $$\text{Jaccard similarity} = J(\mathbf{x}, \mathbf{y}) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)}$$

    3. Pearson Similarity:

    $$\text{Pearson similarity} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

  - Recommend_items: I will use weighted arithmetic mean according to the degree of similarity to fill empty cells in the recommendation-result table. The equation will be showed as follows:

    $$\bar{x} = \frac{w_1 x_1 + w_2 x_2 + \cdots + w_n x_n}{w_1 + w_2 + \cdots + w_n}.$$

- Tools:
  I will use graphlab libaray in Python to build the recommendation system.

- Step 1: Data Loading

The data I input is the **Table 12**, a groub-by dataset for collaborative filtering.

Since I use the graphlab library to build the collaborative filtering recommendation, I convert the pandas dataframe into graphlab dataframe, which is similar with the pandas dataframe but is suitable for function in graphlab library. In order to evaluate the performance of different methods to calculate similarity, I split the dataset into training set and test set with a ratio in 80/20.

```
This non-commercial license of GraphLab Create for academic use is assigned to nman@stevens.edu and will expire on
April 25, 2019.
```

- Step 2: calculate similarity

In this part I will test three different methods to calculate similarity and compare the precision and recall for each method. The three methods will be showed as follows:

```
1. cosine similarity;
2. Jaccard Similarity;
3. Pearson Similarity.
```

The definition of precision and recall will be showed as follows:

- Recall: It represent the ratio of items that a user likes were actually recommended.
- Precision: It means the user actually liked how many items out of all the recommended items.

- Cosine similarity

Similarity is the cosine of the angle between the 2 vectors of the item vectors of A and B.

$$\text{Cosine similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

```
Precision and recall summary statistics by cutoff
+--------+-----------------+-----------------+
| cutoff |  mean_precision |   mean_recall   |
+--------+-----------------+-----------------+
|   1    |  0.029708085766 | 0.0227225509452 |
|   2    | 0.0267372771894 | 0.0412972804151 |
|   3    | 0.0233646201096 | 0.0540945097843 |
|   4    | 0.0222164815293 | 0.0675257182038 |
|   5    | 0.0208731593903 | 0.0792679885142 |
|   6    | 0.0196475214558 |  0.089786578108 |
|   7    | 0.0189442865754 |  0.10113746566  |
|   8    | 0.0180293636442 |  0.109458753029 |
|   9    | 0.0172699178124 |  0.117481097992 |
|   10   | 0.0170240247998 |  0.128617790437 |
+--------+-----------------+-----------------+
[10 rows x 3 columns]
```

Table 16: Cosine precision and recall

...

In **Table 16**, along with the increase of training epoches, the scores of precision and recall are enhance accordingly. In the cutoff 10, the precision is 0.0170 and recall is 0.1286.

- Jaccard Similarity

Jaccard similarity is based on the number of users which have rated item A and B divided by the number of users who have rated either A or B.clicked.

$$\text{Jaccard similarity} = J(\mathbf{x}, \mathbf{y}) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)}$$

```
Precision and recall summary statistics by cutoff
+--------+-----------------+-----------------+
| cutoff |  mean_precision |   mean_recall   |
+--------+-----------------+-----------------+
|   1    | 0.0296219753724 | 0.0229579193542 |
|   2    | 0.0257039524671 | 0.0395347799249 |
|   3    | 0.0231349923936 | 0.0534933772276 |
|   4    | 0.0220227331439 | 0.0681769636154 |
|   5    | 0.0205631619737 | 0.0788656554071 |
|   6    | 0.0193604868108 |  0.08881618977  |
|   7    | 0.0190180954841 |  0.101965200101 |
|   8    | 0.0182338758288 |   0.1114334695  |
|   9    | 0.0173560282059 |  0.118891347166 |
|   10   | 0.0167743046586 |  0.127352106486 |
+--------+-----------------+-----------------+
[10 rows x 3 columns]
```

Table 17: Jaccard precision and recall

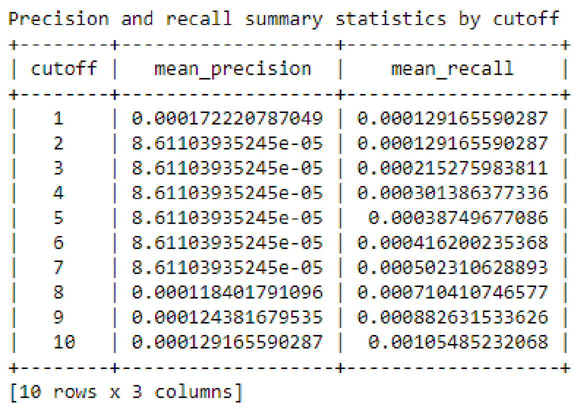
...

Table 17 shows that along with the increase of training epoches, the scores of precision and recall are enhance accordingly. In the cutoff 10, the precision is 0.0167 and recall is 0.1274.

- Pearson Similarity

Pearson Similarity is the pearson coefficient between the two vectors.

$$\text{Pearson similarity} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$



```
Precision and recall summary statistics by cutoff
+--------+------------------+-------------------+
| cutoff |  mean_precision  |    mean_recall    |
+--------+------------------+-------------------+
|   1    | 0.000172220787049 | 0.000129165590287 |
|   2    | 8.61103935245e-05 | 0.000129165590287 |
|   3    | 8.61103935245e-05 | 0.000215275983811 |
|   4    | 8.61103935245e-05 | 0.000301386377336 |
|   5    | 8.61103935245e-05 |  0.00038749677086 |
|   6    | 8.61103935245e-05 | 0.000416200235368 |
|   7    | 8.61103935245e-05 | 0.000502310628893 |
|   8    | 0.000118401791096 | 0.000710410746577 |
|   9    | 0.000124381679535 | 0.000882631533626 |
|   10   | 0.000129165590287 |  0.00105485232068 |
+--------+------------------+-------------------+
[10 rows x 3 columns]
```

Table 18: Pearson precision and recall

...

...

In **table 18**, along with the increase of training epoches, the scores of precision and recall are enhance accordingly. In the cutoff 10, the precision is 0.0001 and recall is 0.0011.

So, with comparison of these three methods to calculate similarity, the cosine similarity perform better than the others, while the Jaccard similarity only slightly lower than cosine similarity. So either cosine similarity or jaccard similarity would be good for the collaborative filtering.

Now I will use cosine similarity and jaccard similarity to recommend restaurants for each customer in the dataset.

- Step 4: Recommend top 3 rating restaurants to each customer

Cosine Similarity Result looks like **Table 19**. Jaccard Similarity result looks like **Table 20**. **Table 21** is the Pearson similarity result. Each customer receive three recommended restaurants ranking by the score.

```
+------------------------+------------------------------+----------------+------+
|        user_id         |       restaurant_name        |     score      | rank |
+------------------------+------------------------------+----------------+------+
| fsq7a4Mog2pJ00er6DuHvw |        Los Tacos No.1        | 0.160737276077 |  1   |
| fsq7a4Mog2pJ00er6DuHvw |            Decoy             | 0.0817471146584 |  2   |
| fsq7a4Mog2pJ00er6DuHvw | Artichoke Basille’s Pizza ... | 0.0772345662117 |  3   |
| J_nwnpiV4ZqohauRp7XgEQ |        Los Tacos No.1        | 0.267895460129 |  1   |
| J_nwnpiV4ZqohauRp7XgEQ |            Decoy             | 0.136245191097 |  2   |
| J_nwnpiV4ZqohauRp7XgEQ | Artichoke Basille’s Pizza ... | 0.12872427702  |  3   |
+------------------------+------------------------------+----------------+------+
[113163 rows x 4 columns]
```

Table 19: Cosine Result

```
+------------------------+------------------------------+----------------+------+
|        user_id         |       restaurant_name        |     score      | rank |
+------------------------+------------------------------+----------------+------+
| fsq7a4Mog2pJ00er6DuHvw |        Los Tacos No.1        | 0.0233147740364 |  1   |
| fsq7a4Mog2pJ00er6DuHvw |            Decoy             | 0.0152284502983 |  2   |
| fsq7a4Mog2pJ00er6DuHvw | Artichoke Basille’s Pizza ... | 0.0133729577065 |  3   |
| J_nwnpiV4ZqohauRp7XgEQ |        Los Tacos No.1        | 0.0233147740364 |  1   |
| J_nwnpiV4ZqohauRp7XgEQ |            Decoy             | 0.0152284502983 |  2   |
| J_nwnpiV4ZqohauRp7XgEQ | Artichoke Basille’s Pizza ... | 0.0133729577065 |  3   |
+------------------------+------------------------------+----------------+------+
[113163 rows x 4 columns]
```

Table 20：Jaccard Result

```
+------------------------+-----------------------------+-------+------+
|        user_id         |       restaurant_name       | score | rank |
+------------------------+-----------------------------+-------+------+
| fsq7a4Mog2pJ00er6DuHvw |       Gold Roast Cafe       |  5.0  |  1   |
| fsq7a4Mog2pJ00er6DuHvw |        52 Restaurant        |  5.0  |  2   |
| fsq7a4Mog2pJ00er6DuHvw | Luna’s Pizzeria & Restaurant |  5.0  |  3   |
| J_nwnpiV4ZqohauRp7XgEQ |       Gold Roast Cafe       |  5.0  |  1   |
| J_nwnpiV4ZqohauRp7XgEQ |        52 Restaurant        |  5.0  |  2   |
| J_nwnpiV4ZqohauRp7XgEQ | Luna’s Pizzeria & Restaurant |  5.0  |  3   |
+------------------------+-----------------------------+-------+------+
[113163 rows x 4 columns]
```

Table 21: Pearson Result

I am curious about why the Pearson similarity result seems similar for different users.

```
restaurant_name
Manà Restaurant                    5.000000
Wah Yoan                           5.000000
Taquitos Mexicanos                 5.000000
Bocata’s Deli                      5.000000
Azteca Taqueria Restaurant         5.000000
Ujala Kabab                        5.000000
Manhattan Bar Grill & Lounge       5.000000
MONDO - DOGS                       5.000000
Ventura Restaurant                 5.000000
Terranova Deli                     5.000000
El Salvador Restaurant             5.000000
52 Restaurant                      5.000000
Gold Roast Cafe                    5.000000
Griot Cafe                         5.000000
Taste of Greece                    4.857143
Andrea Salumeria                   4.840000
Toscana                            4.789474
Royal Grill Halal Food             4.785714
Salerno Salumeria                  4.700000
Losurdo’s Italian Bakery & Deli    4.700000
Name: user_rating, dtype: float64
```

We could find that all the results are almost same when using pearson model. Since all the recommended restaurant have an average rating of 5. So the pearson recommender is not accurate enough.

Collaborative Filtering Summary:

The collaborative filtering method seems better than the co-occurrence matrix, since it take the user's rating into account, which means it would be more relative to customers' real demand and it is still easy to understand. In addition, the collaborative filtering system will perform better and better with more and more users in the dataset. However, it also only focus on the privious user's behaviors like the co-occurrence matrix method, so it doesn't have the ability to update accordingly. Moreover, it will be limited by data sparsity. If we could not collect enough data and the data sparsity is too low, the collaborative filtering will not perform well.

Part 3 - 3 Matrix Factorization

- Background: The Matrix Factorization is the third method I used to build recommendation system. The assumption of matrix factorization is that each user have tendency to like different features of item. For example, in Hoboken restaurant dataset, the features might be theme and style of the restaurant, environment, service quality, food quality of restaurants and so on. Different user have different expectation and preferences for different features. In addition, each item, restaurant in this case, have features in different degree. We assume that the user will rate higher to a new restaurant with more features this user like and will rate lower to a new restaurant with less features this user like.
- Algorithms/equation:
  - Basic Algorithm:

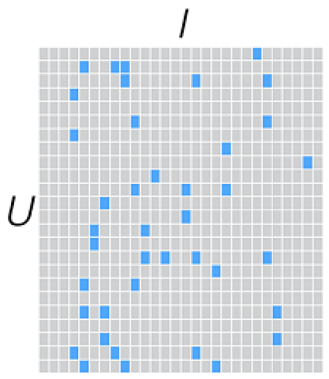
$$\hat{r}_{ui} = \sum_{k=1}^{K} U_{uk} V_{ki}$$

```
- R_ui: user u's rating for item i
- U_uk:item i's similarity with a feature k
- V_ki:user u's preference for a feature k
```

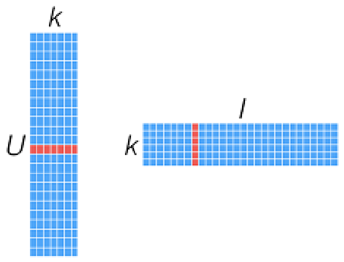
  - Algorithm with bias:

$$\hat{r}_{ui} = \sum_{k=1}^{K} U_{uk} V_{ki} + \mu + b_u + b_i$$

  - This algorithm could also include more parameters like customer's feedback, taste changes according to time and so on.
- Tools:
  - I will use sklearn and numpy in python to build the matrix factorization recommendation system.
  - The input data looks like the example as follows:



  - And then I will seperate this user and item matrix into two marix look like follows:



  - K means features in this case so each row in the users and features matrix will represent the customer's preference for different features and each column in the features and items matrix will represent the degree of a restaurant have such features.

- Step 1: Data Preparation

In this step, I will convert the input data, **Table 13: A matrix like table with rating scores** into a numpy matrix.

...

- Step 2: Extract features k

I could use sklearn library to extract features k and seperate the matrix into two matrices, the user and features matrix as well as the features and items matrix. In this part, we reduce the dimension of original matrices and got two new matrices.

```
- new matrix 1: represent user i's preference of different features k
- new matrix 2: represent the item j's similarity to features k
```

After building the function, I randomly use features_no= 10 at the first try to create two new matrices. The shape of first matrix is (44949, 10) and second matrix is (10, 302).

- Step 3: Recommender

In this part, I will get the recommender matrix by dot product of these two matrices with using numpy.dot()function. The recommender matrix will look like **table 22**.

	user_id	10th & Willow Bar & Grill	52 Restaurant	8th Street Tavern	Adoro Lei	Aether Game Cafe	Ahri's Kitchen	Ainsworth Hoboken	Aldys Restaurant	Ali Baba	...	White Star Bar
0	---xAZNw9fFPBoy7jmkA2A	0.003127	8.309046e-07	0.000713	0.000139	0.000200	0.000706	0.001725	0.000039	0.001400	...	0.002331
1	--68ZwhCrUJUmCXXkMTMKw	0.000378	9.981413e-08	0.000086	0.000032	0.000024	0.000096	0.000209	0.000005	0.000168	...	0.000285
2	--8M2DZ9JkDwTveuRhLPTQ	0.000000	0.000000e+00	0.000000	0.005092	0.000000	0.000156	0.000000	0.000000	0.000000	...	0.000000
3	--ARr3m5JsxaX3DTUVQW7w	0.000000	0.000000e+00	0.000000	0.000000	0.000000	0.001011	0.000159	0.000000	0.000000	...	0.000000
4	--CZJeSIpxwQ0VULjnM57w	0.007670	2.037205e-06	0.001752	0.000336	0.000491	0.001732	0.004230	0.000096	0.003434	...	0.005721

5 rows × 303 columns

Table 22: recommender matrix

- Step 4: Export the Result

In order to recommend new restaurants to each customer, I will remove the score for the restaurant each customer have already attended. And then each customer will receive three recommended restaurants. A sample result will look like following list:

. . .

```
[['1', 'The Cuban Restaurant and Bar', 0.0017525621561418668],
 ['2', 'La Isla Restaurant', 0.0015292799793646238],
 ['3', 'Morimoto', 0.0014210628091436324]]
```

I build a function with using for loop to recommend the restaurants for each customer in the dataset and save the result as 'matrix_factorization_result.csv'.

- Step 5 Check the result

The **Table 23** shows a sample result of matrix factorization recommendation. The first item in the result list represent the ranking, the second item is the restaurant's name and the third item is the scores of each restaurant.

	Unnamed: 0	user_id	Recommendation
0	0	---xAZNw9fFPBoy7jmkA2A	[['1', 'The Cuban Restaurant and Bar', 0.01458...
1	1	--68ZwhCrUJUmCXXkMTMKw	[['1', 'The Cuban Restaurant and Bar', 0.00175...
2	2	--8M2DZ9JkDwTveuRhLPTQ	[['1', 'Perry St', 0.23603528958327633], ['2',...
3	3	--ARr3m5JsxaX3DTUVQW7w	[['1', 'STK Downtown', 0.037760289982801606], ...
4	4	--CZJeSIpxwQ0VULjnM57w	[['1', 'The Cuban Restaurant and Bar', 0.03576...

```
"[['1', 'The Cuban Restaurant and Bar', 0.01458922419511693], ['2', 'La Isla Restaurant', 0.012734421654190975],
['3', 'Pilsener Haus & Biergarten', 0.00954516675514815]]"
```

Table 23: A sample of matrix factorization result

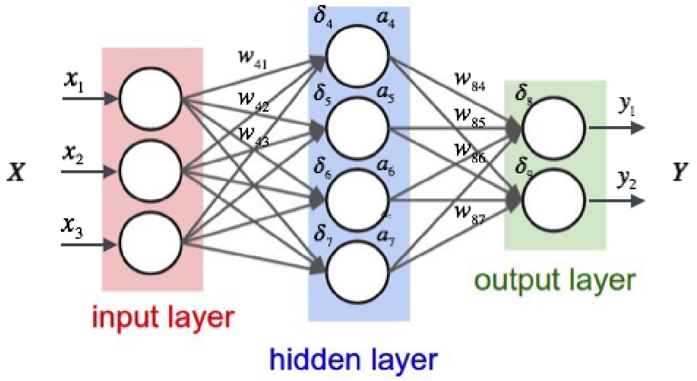
Matrix Factorization Summary：

In general, matrix factorization method is more elegant than previous two methods. It could easily apply to masive dataset and we could add more new parameter(bias, historical feedback, time changes and so on) to the algorithms accordindly. However, the data sparisity will also limit its performance.

Part 3 - 4. Deep Learning - neural network:

I have already build three recommendation systems with popular recommendation algorithms. Now, I will try to use deep learning method - Neural Networks to build a recommendation system. The objective is same as previous methods. I will recommend 3 resturants each customer most likely to be interested to them.

Neural Network method simulate the processes that human or animal recognized the object. An artificial neural network includes one input layer, at least one hidden layer with nodes(kernel) and bias, one output layer including defined target classes. Each layers and nodes will be randomly generated when launching the training process. Or if we apply transfer learning, the layers and nodes could be like a pre-trained model. Input variables will be normalized firstly, multiple with each neural weights, sum together, plus an bias and connect with the next layer. The target of each training process is trying to reduce the differences between real labels and the predicted results. We could use sigmoid, adam or other methods to optimize the reducing process. The revised information will transfer back to each layer and nodes as well as modify the weights and bias. Finally, we will get a neural network with relatively accurate weights and bias for each node on each layer.



Before fiting the variable into neural network, we should normalize the input variables for each row first, since the Neural Network Algorithm assume that the input variable is an number between 0 to 1.

Tools:

- I will use sklearn in python to build the NN recommendation system.

Model design:

- Label:
  - Restaurants Name (302 classes in total)
- According to the EDA part, I will include some variables in the model.
- Input variables:
  - user_rating
  - restaurant rating
  - restaurant price
  - restaurant type
- The first three variables are numerical data and the restaurant type is data in text format. So I will also apply NLP to convert the restaurant_variables into the data type that NN could recognize.
- I design this model because the Neural Network model will return a restaurant that most likely match with the input variables. In a situation that accuracy eaqual to 100%, NN model will return the label (restaurant) we inputed. At the same time, it will also give a probability for each class in the dataset, which will show how does one restaurant is likely to be the accurate one. In another word, in order to build a recommendation system based on NN, I could rank both of these classes(302 restaurants) by the probability rate. And then, I could remove the accurate label in the ranking list and return other classes from higer probability to lower. So, the top n restaurants in this ranking will be the restaurants that are most similar to the accurate one.

Part 3 - 4 - 1: Data Preparation

Table 24 is the dataset I use to build the Neural Network recommendation system. I will use the restaurant_name as the label and use user_rating, restaurant_rating, restaurant_price and restaurant_type as the independent variables.

	user_id	restaurant_name	user_rating	restaurant_rating	restaurant_price	restaurant_type
0	dRuCO4NYO7zyAF8-CeJmZg	Grand Vin	5	4.0	2.0	cocktail bars wine italian
1	f36YZ1cA291bNtMHXWtu1Q	Grand Vin	4	4.0	2.0	cocktail bars wine italian
2	-xYUKfWQTaB-7BeizsQA3w	Grand Vin	5	4.0	2.0	cocktail bars wine italian
3	tt1vLgAP5UpRXAKJLT2KWg	Grand Vin	4	4.0	2.0	cocktail bars wine italian
4	-K79Xep4lElqIChsJYWuiQ	Grand Vin	5	4.0	2.0	cocktail bars wine italian

Table 24: Neural Network input data

Step 1: Normalized numerical data

In this step, I use 'max and min' normalized method to encode the user_rating, restaurant_rating and restaurant_price column into normalized value.

Step 2: Normalized text data by applying NLP- tf-idf

In this step, I use 'tf-idf' to convert the restaurant_type data into weights.

Step 3: Combine step 1 and step 2 and store the new matrix like dataframe

Table 25 shows the new matrix like dataframe. Each row represent each customer. The 0 to 128 columns(129 in total) represent 129 features of restaurant type and the weight of preferences that each customer tend to like each feature. User_rating, restaurant_rating and restaurant_type columns are the encoded value according to original dataframe. Output column is the restaurant_name and will also be the label for each observation.

	0	1	2	3	4	5	6	7	8	9	...	123	124	125	126	127	128	user_rating	restaurant
user_id																			
dRuCO4NYO7zyAF8-CeJmZg	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.309521	0.0	0.0	...	0.0	0.0	0.0	0.670605	0.0	0.0	0.004578	0.
f36YZ1cA291bNtMHXWtu1Q	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.309521	0.0	0.0	...	0.0	0.0	0.0	0.670605	0.0	0.0	0.003663	0.
-xYUKfWQTaB-7BeizsQA3w	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.309521	0.0	0.0	...	0.0	0.0	0.0	0.670605	0.0	0.0	0.004578	0.
tt1vLgAP5UpRXAKJLT2KWg	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.309521	0.0	0.0	...	0.0	0.0	0.0	0.670605	0.0	0.0	0.003663	0.
-K79Xep4lElqlChsJYWuiQ	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.309521	0.0	0.0	...	0.0	0.0	0.0	0.670605	0.0	0.0	0.004578	0.

5 rows × 133 columns

Table 25: Neural Network dataset

Part 3 - 4 - 2: Build Neural Network model and tune to find a best model

Step 1: Data loading and spliting

The first target in this step is to figure out a best model, so I split the input data into two set with ratio, 80/20.

Step 2: Build the model

At the first try, I randomly generate a neural network model with following parameters:

```
- hidden layer siz e= (16,6)
- solver = adam
- learning rate = 0.001
```

After training the model with training set and predicting the model with test set, the accuracy rate is 86.28% for the test set.

```
Accuracy: 0.8628962004958789
Wall time: 3min 10s
```

Step 3:Tuning

In order to get a better model, I tune some parameters and obtain a better model with 87.77% accuracy rate and parameters as follows:

```
- hidden layer siz e= (25,6)
- solver = adam
- learning rate = 0.005
```

```
{'hidden_layer_sizes': (25, 6), 'learning_rate_init': 0.005}
Wall time: 2h 56min 34s

Accuracy: 0.8777055551832741
```

Step 4: Get recommendation result with best estimator

Following table shows a sample result of the neural network model.

	probability	restaurant_name
0	1.504943e-37	10th & Willow Bar & Grill
1	4.619613e-58	52 Restaurant
2	5.929570e-134	8th Street Tavern
3	7.670625e-54	Adoro Lei
4	2.107623e-145	Aether Game Cafe

```
[['1', 'Augustino’s', 0.022578243355864183],
 ['2', 'Hudson Clearwater', 0.0003665265326626242],
 ['3', 'The Lobster Place', 0.00035943076450474433]]
```

Step 5: Check the result

Following table shows the final result that recommend 3 restaurants to each customers. The shape of the result is similar with the co-occurrence result and matrix factarization result. The only different is that the third item means the probability that the recommended restaurant match with the restaurants the user attended.

	user_id	Recommendation
0	dRuCO4NYO7zyAF8-CeJmZg	[['1', 'Barbuto', 0.0034586612971036304], ['2'...
1	f36YZ1cA291bNtMHXWtu1Q	[['1', 'Barbuto', 0.0032257489325294257], ['2'...
2	-xYUKfWQTaB-7BeizsQA3w	[['1', 'Barbuto', 0.0034586612971036304], ['2'...
3	tt1vLgAP5UpRXAKJLT2KWg	[['1', 'Barbuto', 0.0032257489325294257], ['2'...
4	-K79Xep4lElqIChsJYWuiQ	[['1', 'Barbuto', 0.0034586612971036304], ['2'...

```
"[['1', 'Barbuto', 0.0034586612971036304], ['2', 'Zack’s Oak Bar & Restaurant', 0.00043441651480210933], ['3', 'The
Ear Inn', 0.0002991082358883752]]"
```

Neural Network Summary:

Neural Network recommendation system is different from previous three methods. The underlying concept is to predict the probability a restaurant matched with a customer's interests. This method could include more and take into account more parameters like customer's gender, their consumer behaviors, their social network relationship and so on. Also, along with more and more observations are using to feed the model, the accuracy of the model will be better and better. In addition, we could tune the model to find better parameters to build the neural network model. There are still space to tune and find a better model. However, the neural network recommendation system also have limitation that it could not perform well for a sparse dataset.

Part 4: Comparison of the Recommendation System methods

In real world, there are multiple parameters to evaluate the performance of recommendation system, such as customer's satisfaction, diversity of recommendation, creativity, innovation, surprice degree, real time updated and cold start, profit of project. However, this project was built in offline sandbox, it is hard to use the online evaluation to detect the performance of these methods.

So I will compare these four recommendation system methods through describing their limitaions, advantages and application scenarios.

- Co-occurrencce Matrix
  - Advantage： Easy to understand; Start with any size dataset
  - Limitation： Only focus on previous behaviors; ignore customer's rating
- Collaborative Filtering
  - Advantage：Perform well in masive dataset; Take customer's rating into account
  - Limitation: Only focus on previous behaviors; rely on customer's rating ; Limited by data sparsity
- Matrix Factorization
  - Advantage：Take more parameters like bias, customer's feed back, changes of taste into account
  - Limitation: Limited by data sparsity
- Neural Network |
  - Advantage：Easy to add more independent variables to build the NN model; Performance could be elevated with more data; Performance could be enhanced by tuning

    - Limitation: Limited by data sparsity

V. Project Conclusion

Our main target of this project is to explore the insight from customer reviews from Yelp.com. We have two data sources, 1. Customer Reviews of Restaurants at Hoboken (74, 611 rows). We build a Web Scraper to scrape this dataset. 2. Full dataset yelp provided (321 million rows). we use yelp open source API to collect this dataset.

In addition, we have tree sub objectives. The first one is predicting stars based on customer's reviews and historical data. We use Random Forest algorithm in to predict the result. Our second sub objective is figuring out popular topics customer cared about, which will benefit owners of restaurants to improve their service quality. In this part, we applied NLP & Sentiment Analysis with using CV, tf-idf, LDA methods and machine Learning algotithms like Logistic Regression and Random Forest. The thrid sub target is recommending top 3 restaurants to each customer in the Hoboken dataset. In this part, we use four different recommendation methods including Co-occurrence Matrix, Collaborative Filtering, Matrix Factorization and a deep learning method Neural Network.

In general, we reach the main target we setup and figure out the insight from reviews data. We provide scores prediction, help owners of restaurants modify their services and products and create profit for both restaurateurs and yelp.com.

VI. Reference:

- Recommendation System part:
  1. Collaborative Filtering Recommender Based on Co-occurrence Matrix - Marjan Sterjev
  2. Moviex.ai - Movie recommendations with Deep Learning
  3. Deep Neural Networks for YouTube Recommendations - Paul Covington, Jay Adams, Emre Sargin
  4. Matrix Factorization Techniques for recommender system - Yehuda Koren, Robert Bell, Chris Volinsky
  5. https://learnforeverlearn.com/cooccurrence/ (https://learnforeverlearn.com/cooccurrence/)
  6. https://www.analyticsvidhya.com/blog/2016/06/quick-guide-build-recommendation-engine-python/ (https://www.analyticsvidhya.com/blog/2016/06/quick-guide-build-recommendation-engine-python/)
  7. https://hub.packtpub.com/building-recommendation-engine-spark/ (https://hub.packtpub.com/building-recommendation-engine-spark/)