# Yelp Dataset Recommender System Final Report

Yuqin Shen, Maggie Liu, Xiang Li, Xi Liu

## 1. Introduction

Given the explosive information available online, users are often overwhelmed with countless choices of movies, restaurants and products. Companies like Amazon, Netflix and other web services leverage recommender system to create a personalized and delightful user experience to drive revenue. On the other hand, users and customers are able to find better opportunities for good products.

Recommender system in short uses user's information to provide product or content recommendation to the user. Recommendation lists are generated based on user preferences, item feature, user-item past interactions, and some other additional information such as temporal and spatial data [9].

As recommender system plays significant roles in our daily lives, in this project, we build a naive recommender system with Yelp dataset. We will perform sentiment analysis to understand user's preferences and explore different algorithms in building a recommender system to provide recommendations to yelp users.

## 2. Related Work

Sentiment analysis is the process of understanding an opinion about a subject written or spoken in a language. In recent years, the research area of sentiment analysis, opinion mining, sentiment mining, and sentiment extraction has gained great popularity. Hu et al. performs sentiment classification of a document at a sentence level instead of the whole document and extracts features on which opinions have been expressed, identifying opinion words by proposing a technique that uses WordNet lexical database [1]. Salinca, A presented a sentiment analysis approach to business reviews classification using a large reviews dataset provided by Yelp: Yelp Challenge dataset. The author proposed several approaches for automatic sentiment classification, using two feature extraction methods and four machine learning models [2].

The goal of recommendation systems is suggesting to a user the items that might be of interest to him/her. In the literature, there are two basic approaches to give recommendations, namely content based[3], which is mostly used with text documents, and collaborative filtering[4], which is based on the user preference. Collaborative filtering has two main sub-categories, namely memory-based[4], using K nearest neighbors' preference and model-based[5], dealing with large dataset. Recently, to improve the accuracy of the recommendation systems, many novel concepts are introduced to utilize more information, such as hybrid weighted filtering[6], risk aware[7], location[8] and social network[9] aware collaborative filtering.

A similar recommender system was proposed by Chao Shi, Sam Mullane, Sean Kickham, Reza Rad and Andrew Rubino in 2017 using Yelp dataset [10]. An interactive web based recommender system was built using four different recommending approaches, including content based, collaborative filtering, social network and location based. The work focuses on implementing a front-to-end system. In our project, we wanted to explore different approaches in recommender system using content-based filtering, collaborative filtering and neighborhood collaborative filtering. We will apply different algorithms in building the recommender system and examine the advantages and disadvantages between them with real user data.

## 3. Problem Definition

With the information overload problem in the digital era, the use of recommender system is necessary to both e-businesses and customers. Our project aims to build a recommender system that help to identify restaurants that interest the most to the customer based on his or her past interactions as well as the individual's friends' opinion.

The project is two-fold. First, we will perform sentiment analysis with Natural Language Processing to understand user's preferences on business with different attributes. Second, we will utilize different approaches, including content-based filtering, collaborative filtering, neighborhood collaborative filter and a hybrid approach to predict user's preference and thus generate recommendations to the user.

Four tables in the yelp dataset will be explored. User table includes user's friend mapping and all metadata associated with the user. Business table contains business data including location data, attributes and categories. Review table contains full review text data including the user_id that wrote the review and business_id that review is written for. Tip data table includes tips written by a user on a business. We will load four tables into Postgres database. We will be pulling data from Postgres database for later analysis and system building.

## 4. Algorithms

### 4.1. NLP: Latent Semantic Analysis(LSA), Latent Dirichlet Allocation (LDA), Convolution Neural Network

#### 4.1.2 LSA

Before word embedding technique, LSA and LDA are good approaches to deal with NLP problems. Both LSA and LDA have the same input which is Bag of words in matrix format. LSA focus on reducing matri dimension while LDA solves topic modelling problems.

LSA is one of the foundational techniques in topic modelling. The core idea is to take a matrix of what we have – documents and terms – and decompose it into a separate document-topic matrix and a topic-term matrix.

We have m documents and n words as input. An m * n matrix can be constructed while column and row are document and word respectively. We use TF-IDF score, since TF-IDF is better than count occurence in most of time as high frequency do not make a lot of sense for better classification. Based on TF-IDF, word importance will be increased if the number of occurrences within the same document, on the other hand, it will be decreased if it occurs in corpus.

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

**TF-IDF**

Term $x$ within document $y$

$tf_{x,y}$ = frequency of $x$ in $y$

$df_x$ = number of documents containing $x$

$N$ = total number of documents

Fig 1. Equation of TF-IDF
(source: http://mropengate.blogspot.com/2016/04/tf-idf-in-r-language.html)

Besides, we might need to use truncated SVD to reduce matrix dimension as the matrix W is very sparse and noisy. The idea of SVD is finding the most valuable information and using lower dimension to represent the same thing.

#### 4.1.3 LDA

LDA is a Bayesian version of pLSA. In particular, it uses dirichlet priors for the document-topic and word-topic distributions, lending itself to better generalization.

Intuitively, we could imagine that we have two layers of aggregations. One is the distribution of categories or topics, the other is the distribution of words within the category or topic. Before train topic modelling, we need to do data preprocessing like stopwords, pronunciation removal to achieve a better result.

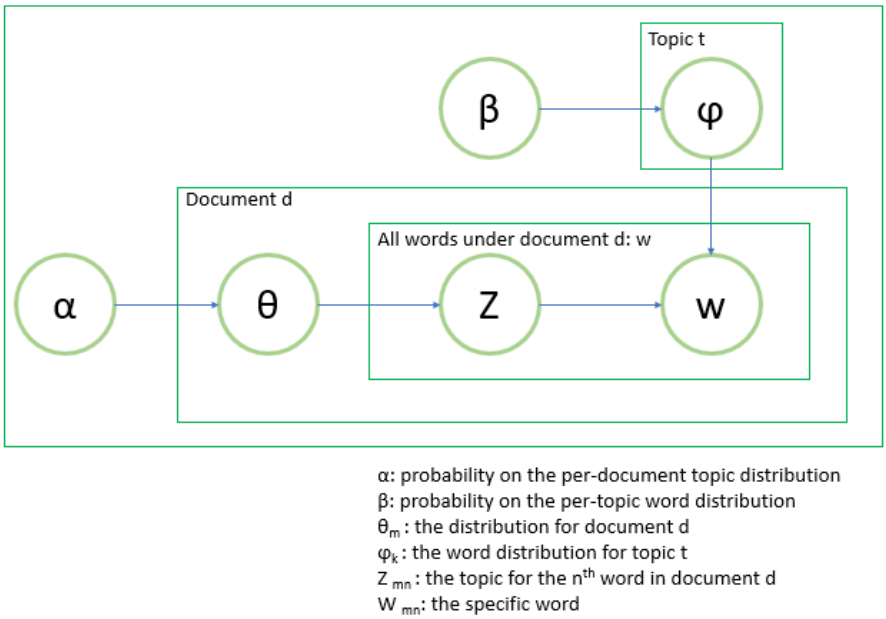



Fig 2. LDA workflow
(source:https://towardsdatascience.com)

From a dirichlet distribution Dir(α), we draw a random sample representing the *topic distribution*, or topic mixture, of a particular document. This topic distribution is θ. From θ, we select a particular topic Z based on the distribution.

Next, from another dirichlet distribution Dir($\beta$), we select a random sample representing the *word distribution* of the topic Z. This word distribution is φ. From φ, we choose the word w.

Formally, we could generate each word from a document.

**4.2. Recommendation systems**

Based on different information sources used for searching and prediction, there are two general methods of recommendation: Content-based filtering and Collaborative filtering.

**4.2.1. Content-based filtering**: It recommends based on the comparison between the content of the items and a user profile. The content of the item in the projects is the restaurant descriptor, the restaurants' categories. Each restaurant is stored as a vector of its attributes in an n-dimensional space and the cosine of the angle between the vectors were calculated to determine the similarity between vectors.

$$K(X, Y) = <X, Y>/(||X||*||Y|)$$

If the user visited a  restaurant in the past and gave a high rating, the similar restaurant would be recommended based on the visited restaurant categories.

**4.2.2. Collaborative filtering**: The above method can be an accurate recommender towards restaurant type. However, it only considered user's own information. To better utilize all the data information in the database, collaborative filtering takes into account all the user rating scores and makes prediction as a linear, weighted combination of other user preferences. We define a user-product matrix Y which describes user preferences. Yij represents the preference of Use ri to Product j.

$$Y = \begin{bmatrix} y_{11} & y_{12} & y_{13} & ? & ? & \cdots & y_{1m} \\ y_{21} & y_{22} & ? & y_{24} & ? & \cdots & ? \\ & & \vdots & \vdots & \ddots & \vdots & \vdots \\ y_{n1} & ? & y_{n3} & ? & y_{n5} & \cdots & y_{nm} \end{bmatrix}$$

**4.2.2.1. Matrics factorization**: To fill in those unknown records, we can decompose *Y* into two low rank matrics Z and W: one representing the latent features of all users and the second represent the latent features of all products:

$$Y \approx \begin{bmatrix} z_{11} & z_{12} & z_{13} & \cdots & z_{1k} \\ z_{21} & z_{22} & z_{23} & \cdots & z_{2k} \\ & & \vdots & \ddots & \vdots \\ z_{n1} & z_{n2} & z_{n3} & \cdots & z_{nk} \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & w_{13} & \cdots & w_{1k} \\ w_{21} & w_{22} & w_{23} & \cdots & w_{2k} \\ & & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & w_{m3} & \cdots & w_{mk} \end{bmatrix}^T$$

Since we do not know these latent features, we build a model to learn and optimize the overall matrix:

- Step1: Normalize the rating to be zero centered.

$$Y_{ij} = Y_{ij} - \mu_j$$

- Step2: Define the cost function as mean square error MSE with L2-regularization.

$$J(W, Z) = \sum_i \sum_j (w_j^T z_i - y_{ij})^2 + \frac{\lambda_1}{2} \|W\|_f^2 + \frac{\lambda_2}{2} \|Z\|_f^2$$

- Step3: Optimize W and Z iteratively using gradient descent

This approach is deep learning-based, which is powerful but computational complex. Here are also some traditional and direct approaches we could consider.

**4.2.2.2 Neighborhood collaborative filtering:** This is an unsupervised cluster method for recommendation based on user/item's rating score's pattern, which computes the similarity between users/items using their rating behavior feature space, then uses K nearest neighbor's average rating score to determine the final likelihood of one specific product.

- User-based collaborative filtering: First we define a user vector for each user rating profile and define similarity between each user using pearson correlation similarity:

$$\text{simil}(x, y) = \frac{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r_x})(r_{y,i} - \bar{r_y})}{\sqrt{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r_x})^2} \sqrt{\sum_{i \in I_{xy}} (r_{y,i} - \bar{r_y})^2}}$$

  where Ixy is the set of items rated by both user x and user y. After we find U most similar neighbors u' to the user u, the new score for item i of user u can be:

$$r_{u,i} = \bar{r_u} + k \sum_{u' \in U} \text{simil}(u, u')(r_{u',i} - \bar{r_{u'}})$$

- Item-based collaborative filtering: This approach clusters similar items based on all the user rating records for this item. So it different from content-based method by using a different item feature vector, which would related to user preference(i.e. User-restaurant rating) rather than item content (i.e. Restaurants information).

**4.2.3 Collaborative filtering combining social network:** We will consider adding social network information into our recommendation systems, because social network is an important factor for user to make decisions on the internet. So we can estimate the score of a new restaurant based on information provided by one user's nearest neighbors and best friends.

## 5. System

We will first preprocess and visualize our system. The sentiment analysis will be performed to understand user. Three methods will be explored for the recommendation system, collaborative filtering, neighbor filtering and content filtering.

1. Sentiment Analysis: Natural Language Processing (NLP)

2. Collaborative Filtering: Alternating Least Squares algorithm
3. Neighborhood Filtering: K-Cluster algorithm
4. Content Filtering: NLP similarity computation

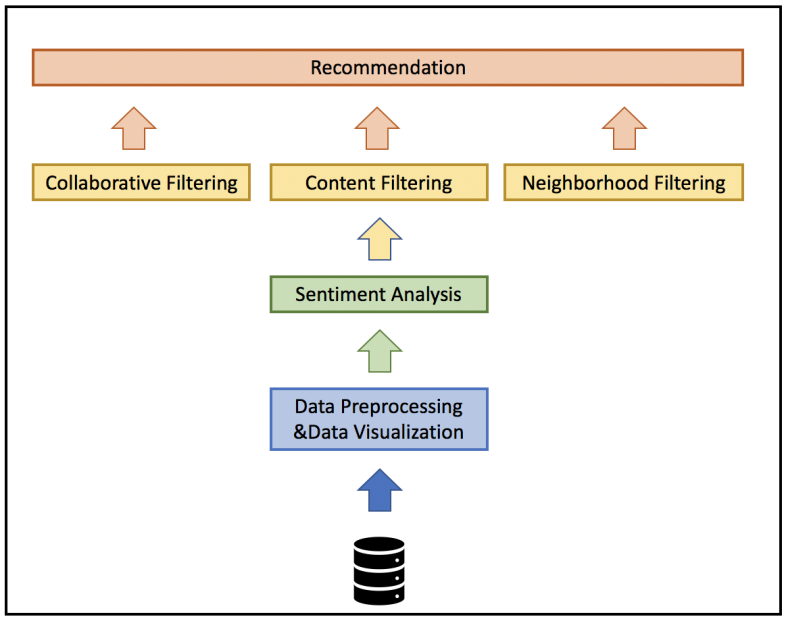



Fig 3. System Architecture

## 6. Experiments

The dataset contains the following number of tables and the following number of tuples and attributes.

Table 1. Dataset Summary

| | Business | Review | User | Tip |
|---|---|---|---|---|
| Tuples | 192609 | 6685900 | 1637138 | 1223094 |
| Attributes | 60 | 9 | 22 | 5 |

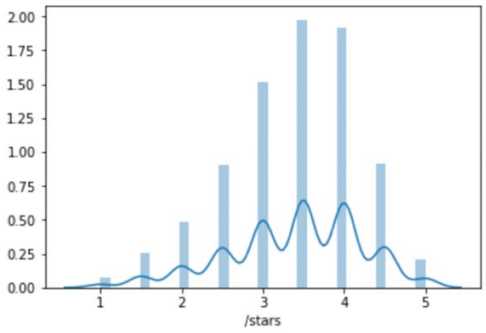
### 6.1 Business Dataset

Original business file has 192609 rows and 60 columns such as business_id, name, address, city, state and so on. In this project, we will focus on the businesses in NC state and 6 useful features were selected. The selected business data has 14720 rows and 6 columns. A sample table is displayed below.

Table 2. Selected Business Data

| | /business_id | /name | /categories | /review_count | /stars | /city |
|---|---|---|---|---|---|---|
| 2 | gnKjwL_1w79qoiV3IC_xQQ | Musashi Japanese Restaurant | Sushi Bars, Restaurants, Japanese | 170 | 4.0 | Charlotte |
| 35 | BvYU3jvGd0TJ7lyZdfiN2Q | Manzetti's Tavern | Sandwiches, Italian, American (Traditional), A... | 16 | 3.5 | Charlotte |
| 58 | _J_x_RaYTqAqAuCwgRhnRQ | Kabob House | Coffee & Tea, Hookah Bars, Nightlife, Persian/... | 15 | 3.0 | Charlotte |
| 141 | L0aSDVHNXCl6sY4cfZQ-5Q | Mai Thai II | Restaurants, Thai | 108 | 4.0 | Concord |
| 157 | U3kygJOTITQFIfaZS7sQjA | JJ's Red Hots - Dilworth | Caterers, Hot Dogs, Restaurants, Vegetarian, A... | 380 | 4.0 | Charlotte |

We visualized the distribution of '/stars' feature and '/review_count' feature for our further analysis.



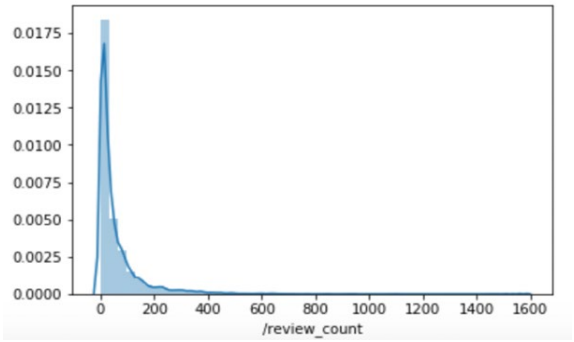




Fig 4. (a) Distribution of 'stars' feature (b) Distribution of 'review_count feature'

Top 30 businesses categories were visualized below.

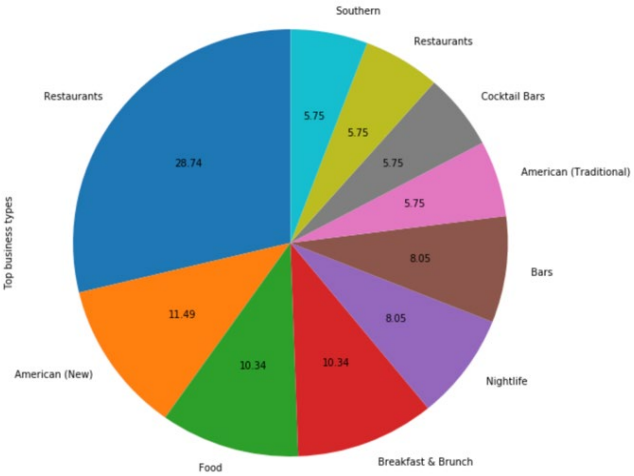
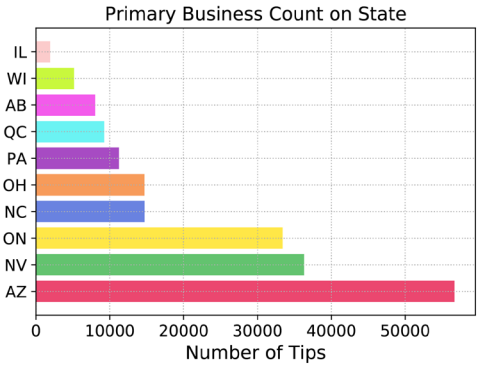



Fig 5. Top 30 Businesses Categories

### 6.2 User & Tips dataset



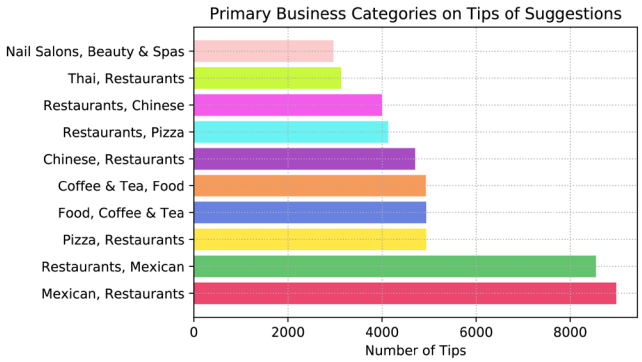
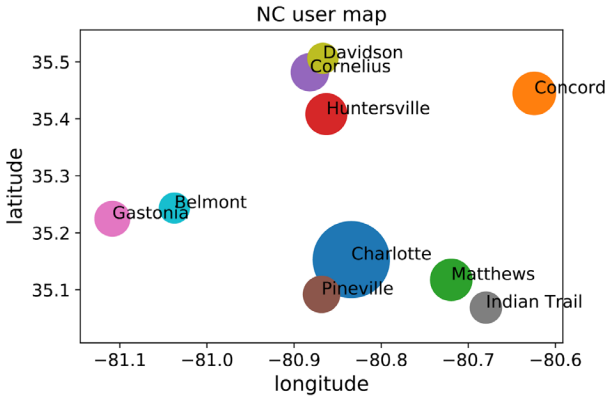




Fig 6. (a) Business Count on State (b) Number of Tips on Business Categories

We could see from Fig 6.(a) that NC states holds medium amount of business, which made it reasonable to analyze reviews, user, tip information on NC states, as the total number of reviews is around 0.66 million, which needs expensive computation.

From Fig 6.(b) we could observe that most of the tips of suggestions are related to restaurants, therefore we will analyze only dataset on restaurants.



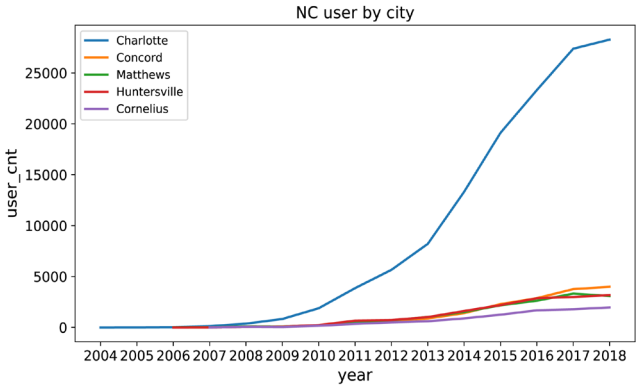




Fig 7. (a) Total User Count by Cities in NC (b) User Count Trend by Year in NC

From Fig 7 we could see that Charlotte occupied most of market share since it has the highest amount of users, and it also takes the strongest of momentum to expand the business.

### 6.3 NLP Preliminary Data Processing & Visualization

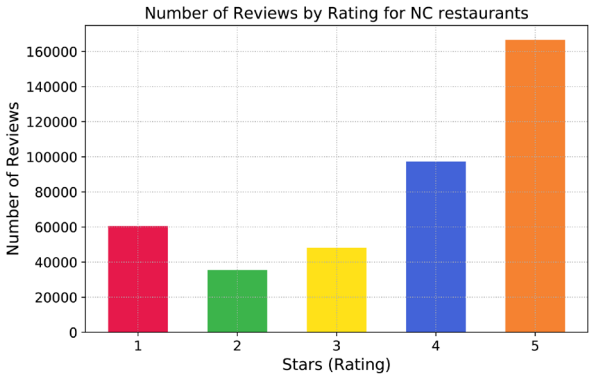



Fig 8. Review Count by Stars in NC

From Fig 8 we noticed that we have more positive reviews than negative reviews. To make the problem easier to analyze, here we categorized stars more than 3 as positive reviews, and stars less than 3 as negative reviews. Then we will make sentimental analysis based on this rule in the future.

Based on this condition, we have: Positive Reviews: 263,859; Negative Reviews: 96,064

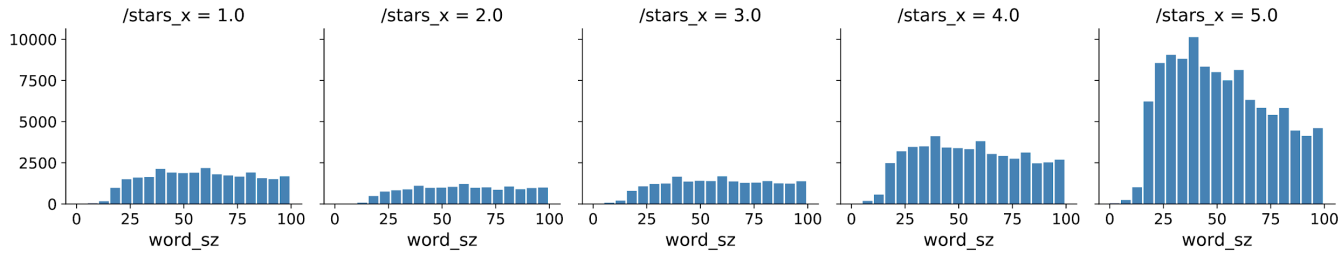



Fig 9. Word Size of Review by Stars Rating

From Fig 9 we could see that positive reviews usually takes longer length of words, and it is usually within 100 words. More work will be done in the future.

### 6.4 Review Dataset

The location, business name and review content were visualized in this section. It is found that the popular restaurants with high ratings (rating>3) cluster around the Charlotte area.

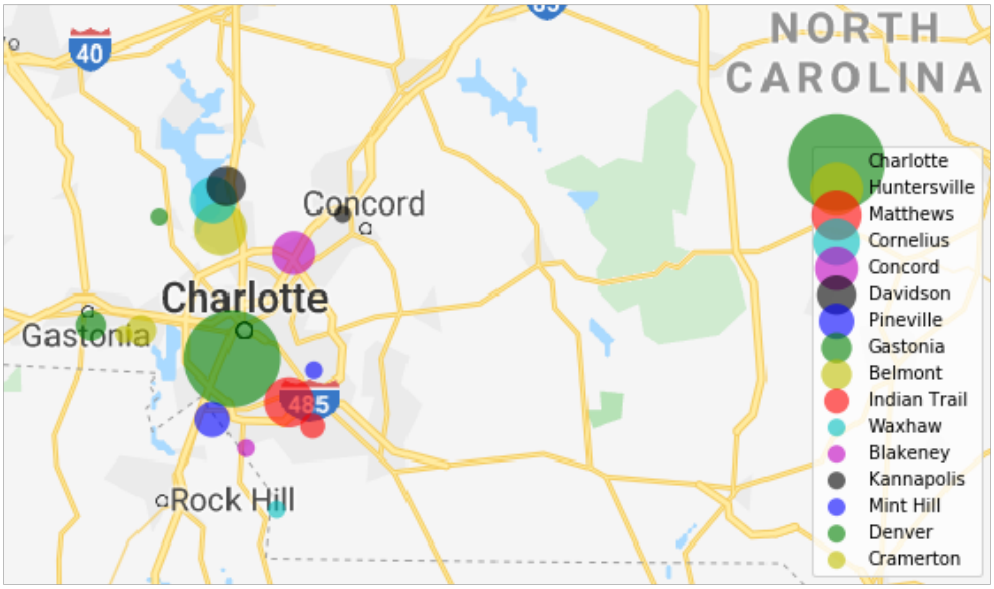



Fig 10. Location of top 500 popular restaurants (star>3) rated by reviewers

The most popular reviewed business includes, “The Cowfish Sushi Burger Bar”, “Mildwood Smokehouse” and “Amelia’s French Bakery & Cafe”. Most reviewed businesses are restaurants or cafes and they span across different restaurants categories. The most frequent words in the positive reviews are “good”, “great”, “restaurant” which are expected that are associated with positive sentiment.

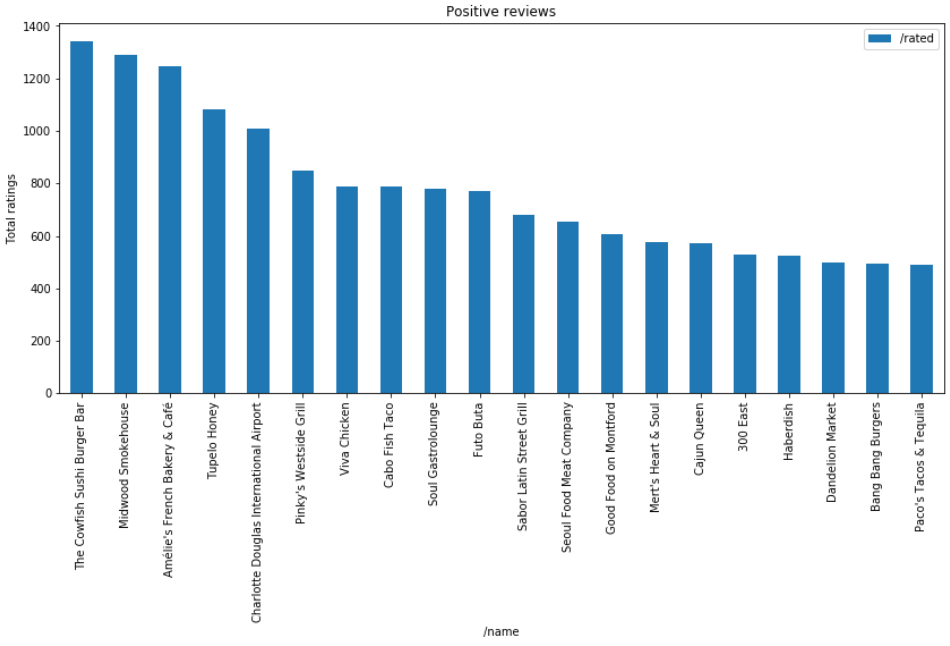



Fig 11. Top 20 reviewed Restaurants with ratings higher than 3

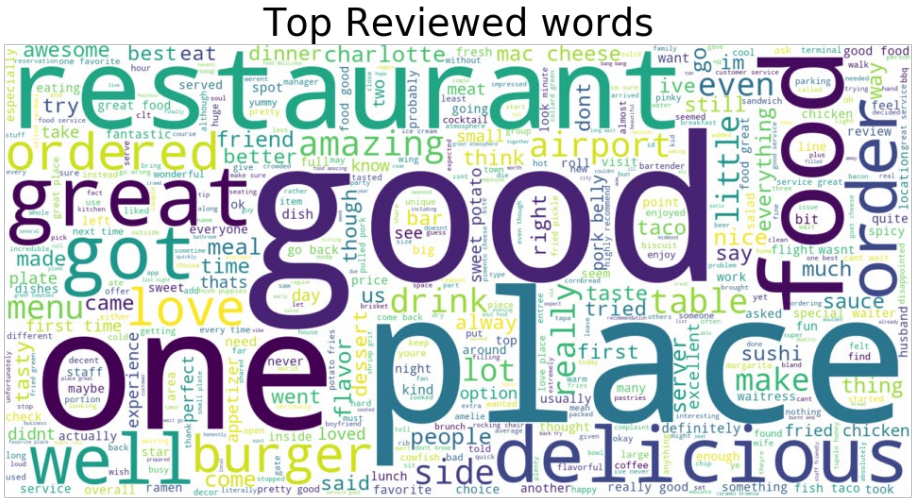



Fig 12. Word cloud for top 20 reviewed businesses

### 6.5 Sentimental Analysis

#### 6.5.1 Data Preprocessing

All the following work is based on this step. To start with, we will try to clean our text data as much as possible. The idea is to remove the punctuation, numbers, and special characters. Though the data cleaning algorithm is slightly different, it works well for the corresponding analysis method. We will remove shorter words which normally don't contain useful information. It is also a good practice to remove the stop-words from the text as they are mostly clutter and hardly carry any useful information. Finally, we make all characters lowercase to nullify case sensitivity.

Since the main purpose is to identify positive or negative comments, we defined the reviews into two classes:
(1) Positive: comments with stars > 3

(2) Negative: comments with stars < 3

#### 6.5.2 LSA (Latent Semantic Analysis)

Step 1:
Document-Term Matrix. We used TfidfVectorizer to create a document-term matrix.

Step 2:
Vectorize term and document. We used the document-term matrix and decompose it into multiple matrices.

Step 3:
SVD model to get top topics. We used TruncatedSVD to get the most important components. The number of topics is specified by n_components parameter.


Fig 13. Most frequent words in positive and negative reviews

The above figure used wordcloud to plot the most frequently used word in both positive and negative reviews.

We also tokenized the text to get an exact sorted list top words in these reviews, shown as following, which is coherent with the images.

Top 8 words in POSITIVE reviews:['great', 'place', 'food', 'get', 'one', 'service', 'go', 'time']
Top 8 words in NEGATIVE reviews:['food', 'get', 'one', 'place', 'service', 'time', 'back', 'never']

It looks like both positive and negative reviews revolve around place, food, service and time.

#### 6.5.2.1 Case Study of LSA

Given a target restaurant, using LSA, we could get to know what kind of information is most related to the customers attitude. With this information, the restaurant could take corresponding actions to improve the business.

Here we took the restaurant with the most number of reviews as an example. The business_id is yQab5dxZzgBLTEHCw9V7_w.

Table 4

| | /business_id | cnt |
|---|---|---|
| 0 | yQab5dxZzgBLTEHCw9V7_w | 1895 |
| 1 | RAh9WCQAuocM7hYM5_6tnw | 1645 |
| 2 | gG9z6zr_49LocyCTvSFg0w | 1598 |
| 3 | WbJ1LRQdOuYYlRLyTkuuxw | 1561 |
| 4 | RVQE2Z2uky4c0-njFQO66g | 1525 |

Table 5. Top topics + words for **positive** reviews

Topic 0:
['food: 0.22', 'nice: 0.20', 'clean: 0.17', 'charlotte: 0.17', 'chairs: 0.16', 'rocking: 0.16', 'time: 0.16']
Topic 1:
['chairs: 0.25', 'rocking: 0.24', 'navigate: 0.21', 'nice: 0.20', 'easy: 0.18', 'food: 0.17', 'clean: 0.15']
Topic 2:
['easy: 0.44', 'navigate: 0.31', 'clean: 0.22', 'nice: 0.19', 'lots: 0.13', 'shops: 0.11', 'staff: 0.10']
Topic 3:
['easy: 0.42', 'navigate: 0.31', 'parking: 0.17', 'great: 0.16', 'security: 0.12', 'rocking: 0.11', 'check: 0.11']
Topic 4:
['food: 0.31', 'options: 0.26', 'terminal: 0.21', 'nice: 0.21', 'plenty: 0.18', 'wifi: 0.14', 'pretty: 0.14']

Table 6. Top topics + words for **negative** reviews

Topic 0:
['flight: 0.25', 'time: 0.20', 'people: 0.19', 'gate: 0.17', 'charlotte: 0.17', 'food: 0.13', 'line: 0.13']
Topic 1:
['parking: 0.23', 'food: 0.18', 'terminal: 0.18', 'construction: 0.13', 'term: 0.12', 'instead: 0.11', 'security: 0.11']
Topic 2:
['baggage: 0.25', 'parking: 0.25', 'claim: 0.18', 'minutes: 0.17', 'bags: 0.16', 'term: 0.12', 'security: 0.12']
Topic 3:
['worst: 0.27', 'rude: 0.24', 'service: 0.19', 'baggage: 0.17', 'experience: 0.16', 'customer: 0.14', 'employees: 0.13']
Topic 4:
['line: 0.35', 'people: 0.24', 'security: 0.15', 'bags: 0.15', 'food: 0.14', 'lines: 0.14', 'long: 0.13']

The restaurant business id is yQab5dxZzgBLTEHCw9V7_w. We limited the SVD components to 10. From the top 10 topics word possibility distribution, we could tell that:

- Positive reviews revolve around food, rocking, navigate.
- Negative reviews revolve around parking, service, food.

### 6.5.3 LDA (Latent Dirichlet Allocation)

The steps of applying LDA is similar to LSA,  what we need to do is change SVD model to LDA model, which is easy to implement using current sklearn library.

Here we used the previous case as an example.

Table 7.  Top topics + words for **positive** reviews

| |
|---|
| Topic 0:<br>['airport: 513.61', 'clean: 125.72', 'food: 98.08', 'like: 92.35', 'nice: 79.77', 'just: 75.84', 'charlotte: 75.65']<br>Topic 1:<br>['airport: 296.81', 'nice: 114.97', 'food: 83.14', 'like: 80.29', 'people: 78.22', 'rocking: 72.39', 'chairs: 71.76']<br>Topic 2:<br>['airport: 202.32', 'charlotte: 71.23', 'clt: 68.44', 've: 58.50', 'airports: 54.90', 'like: 46.91', 'terminal: 40.82']<br>Topic 3:<br>['und: 22.20', 'der: 16.20', 'ist: 16.20', 'den: 13.20', 'die: 13.20', 'flughafen: 13.20', 'das: 11.20']<br>Topic 4:<br>['airport: 645.36', 'food: 284.37', 'time: 189.42', 'charlotte: 184.74', 'rocking: 169.92', 'like: 165.90', 'chairs: 164.73'] |

Now with the trained model, we could visualize the topics for interpretability. Here we use pyLDAvis, a popular visualization package which is designed to help interactively with:

1. Better understanding and interpreting individual topics, and
2. Better understanding the relationships between the topics.

This is an interactive visualization tool. You could manually select each topic to view its top frequent or relevant terms, and you could also explore the intertopic distance plot to learn about how topics relate to each other, including potential higher-level structure between groups of topic.
Noticed that this topic label is different from the LDA topics order we print, but the term for each topic is the same, we could relate the graph with the printing list based on the term weights.

#### 6.5.3.1 Case Study of LDA

We could see from the printing list and the graph of both positive and negative reviews of the restaurant whose business_id is yQab5dxZzgBLTEHCw9V7_w, that positive reviews revolve around airport, food, time,  while negative reviews revolve around flight, time, gate. The word ‘airport’ appears on both positive and negative models, it might be that this restaurant is located in the airport.

It is a useful tool to investigate what kind of elements impact the business.

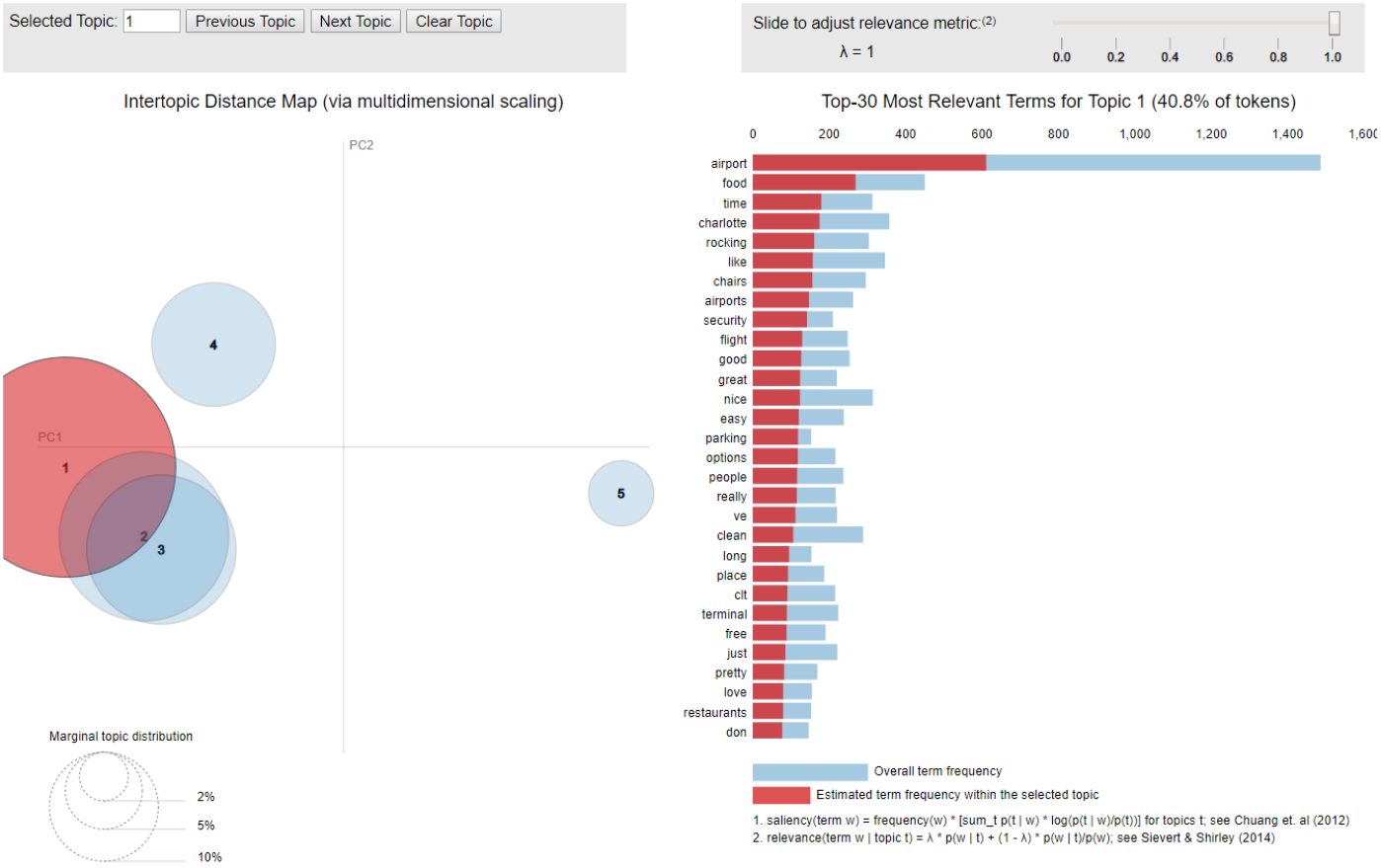



Fig 14. **Positive** Reviews for Business_id: yQab5dxZzgBLTEHCw9V7_w

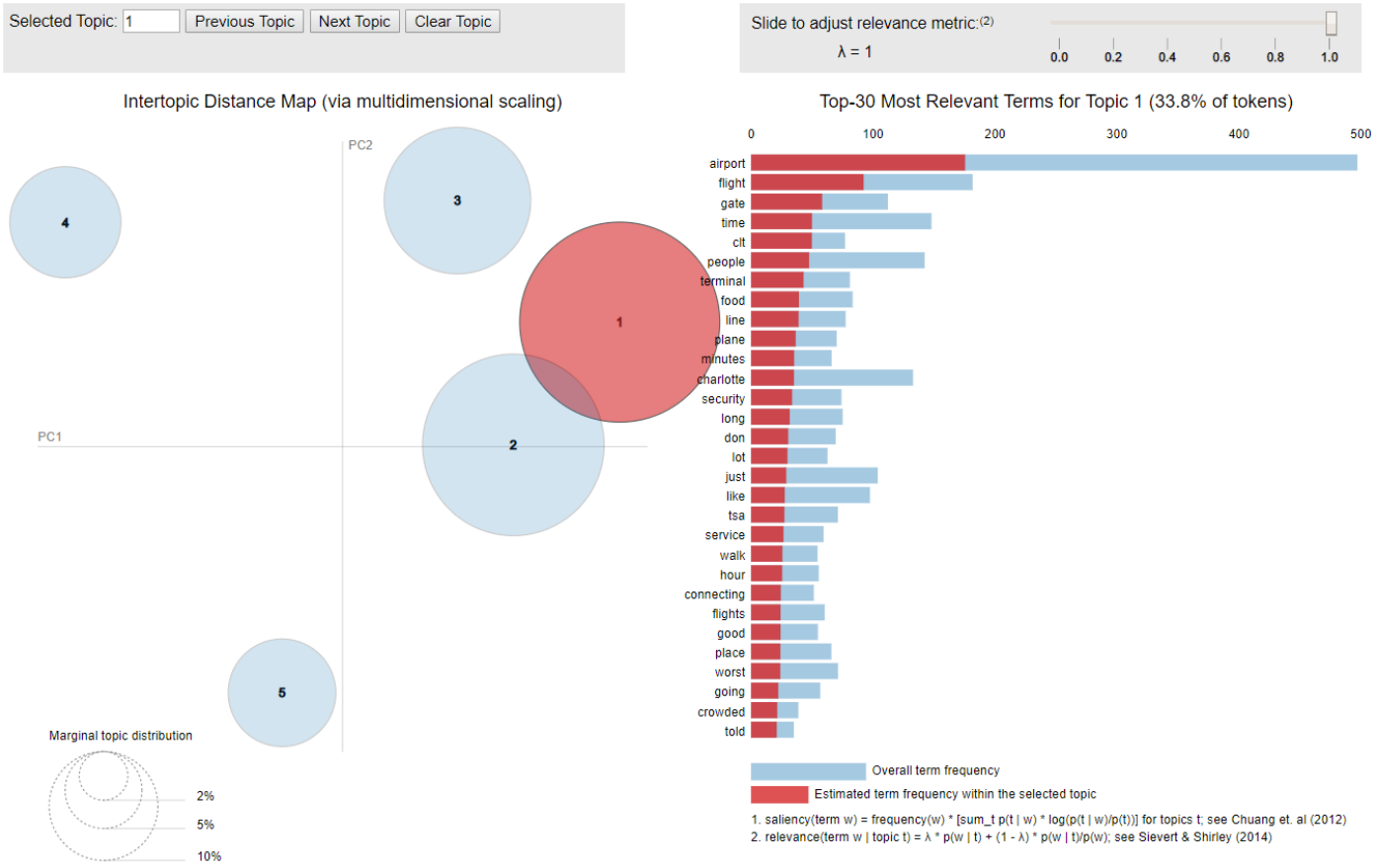



Fig 15. **Negative** Reviews for Business_id: yQab5dxZzgBLTEHCw9V7_w

### 6.5.4 Review Classification using CNN:

This is a binary classification problem. We built and trained  a neural network with LSTM (Long Short Term Memory) and a single embedding layer. The first step is to tokenize the text and convert them to sequences, then we trained the model and evaluated the model.

Network Architecture:

The network starts with an embedding layer. The layer lets the system expand each token to a more massive vector, allowing the network to represent a word in a meaningful way. It follows by a LSTM unit, which is shown as below. Finally, a fully connected dense layer is applied with 'sigmoid' as activation function.

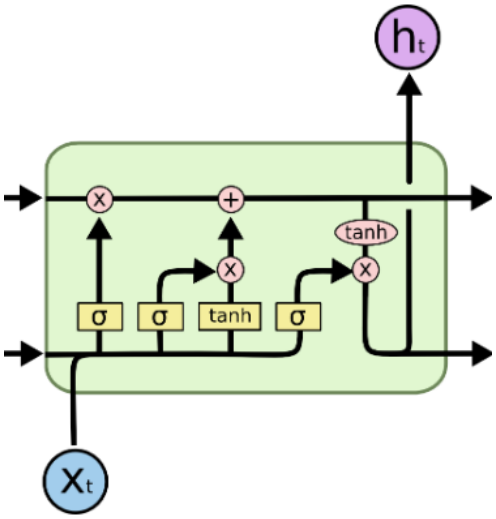



Fig 16. A LSTM unit (Source : http://colah.github.io/posts/2015-08-Understanding-LSTMs)

Here we got the results after 10 epochs training : val_loss: 0.1686 - val_accuracy: 0.9376. More details are shown as the below graphs.

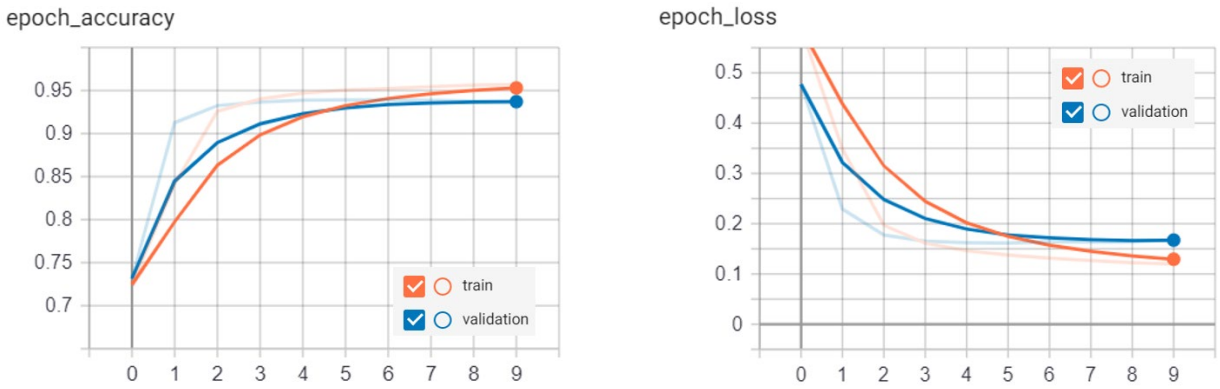



Fig 17. Training accuracy and loss trends

We could see that the model we built has highly performance to predict the sentiment of reviews. We noticed that the accuracy tends to decrease a bit or smooth when the accuracy of training dataset achieves a certain high level, that is caused by overfitting problem.

### 6.6 Algorithm Experiment

The setup for the recommender system implementation was Jupyter notebook, Python==3.5, PySpark.MLlib and Scikit-learn. Two datasets were used, the dataset that consists of all restaurants in the North Carolina area and a joint dataset between all such restaurants and Review table.

For testing, a random user was selected. 70% of user's rated restaurants and his/her rating would be used as the training set. The rest 30% of user's rated restaurants would be served as the validation set. We wanted to validate whether our recommender system would recommend restaurants within the high-rated restaurants in the validation set. A restaurant is considered high-rated if the user rated 4 or 5. The user, "Ry10_KXZHGRI8g5zBR3IcQ", with No.3 amount of user reviews were selected in this experiment.

### 6.6.1 Content Filtering Method

The textual information of the restaurant were converted to a matrix of token counts using CountVectorizer in sklearn. Cosine_Similarity function in sklearn package was then used to compute similarity as the normalized dot product between matrices. All restaurant scores were then sorted for the visited restaurants and the restaurant recommended based on the scores.

In the experiment, the top rated restaurants were selected based on the user rating history. The top five restaurants were chosen and used to generate similar restaurants as recommendation. Two similar restaurants were generated for each which yield ten recommendations in total. We wanted to observe whether the recommended restaurants fall into the validation set. The user, Ry10_KXZHGRI8g5zBR3IcQ was selected. The recommendations were shown below. All ten recommendations were not found in the validation set.

Table 8

| | |
|---|---|
| 'Primavera Pizza' | 'Somi Sushi' |
| 'Mario's Italian Restaurant' | 'Ace of Spuds Truck' |
| 'Kabob Hot' | 'Vitos Pizza Truck' |
| 'Zeitouni Grill' | 'Greco Fresh Grille' |
| 'Osaka Japanese Cuisine' | 'Carolina Beer Company' |

### 6.6.2 Collaborative Filtering - model based Method

Alternating Least Square(ALS) in MLlib is a model-based collaborative recommendation algorithm. It receives a sparse user-item rating matrix as input and then output fitting result for all absent ratings. As other learning models, it contains a hyperparameter tuning process in order to get an optimal model for each different dataset. The main hyperparameter is "rank", which is the number of latent factors to predict in this dataset. Usually with more latent factors, the model would predict more accurately, but with a higher computational cost. We split the whole user-item rating dataset into 9:1 and performed validation to choose the suitable rank for the whole dataset. Finally, we set "rank" = 20, "iterations" = 15 (number of max iterations to converge), "lambda" = 0.1 (regularization parameter).

```
For rank 12 the RMSE is 1.4267500675153593
For rank 16 the RMSE is 1.4194837666095224
For rank 20 the RMSE is 1.4114615232038865
The best model was trained with rank 20
```

To evaluate the performance of the chosen model, we still chose user Ry10_KXZHGRI8g5zBR3IcQ as our testing data. The method is to randomly remove part of the records of this user, set this part as testing data and compare model predicted scored with the hidden score using RMSE metric. For this model, the final prediction RMSE for this testing user is 0.54. Following is part of the validation score pairs. We can see, for a large number of record, our learning-based model can predict fairly well.

```
#(restaurant_id, (user_rating, pred_rating))
```

```
[(40, (4.0, 3.905885696074257)),
 (200, (4.0, 3.929040539361)),
 (390, (3.0, 3.731138805852912)),
 (1630, (4.0, 3.395819429722844)),
 (2390, (4.0, 3.752716469882226)),
 (2630, (4.0, 2.814938789572305)),
 (3310, (4.0, 4.042636309304268)),
 (3530, (4.0, 3.438731145547899)),
 (3660, (4.0, 3.8704340975668727)),
 (3720, (4.0, 3.832137587689461))]
```

After tuning the model, we could give final top recommendations to this user Ry10_KXZHGRI8g5zBR3IcQ from all the unrated restaurants according to his records. Following are part of his highly rated restaurants.

```
# (resraurant_id, (user_rating, restaurant_name))
```

```
[(100, (5.0, 'Not Just Coffee- Packard Place')),
 (148, (5.0, "Sir Edmond Halley's")),
 (284, (5.0, "Bojangles' Famous Chicken 'n Biscuits")),
 (396, (5.0, 'Yummi Banh Mi Truck')),
 (396, (5.0, 'Yummi Banh Mi Truck')),
 (560, (5.0, 'Bakersfield')),
 (576, (5.0, 'Siam Garden')),
 (600, (5.0, 'Rhino Market & Deli')),
 (628, (5.0, 'The Common Market')),
 (680, (5.0, 'Tanaka Grill & Sushi')),
 (848, (5.0, "Bad Daddy's Burger Bar")),
 (1128, (5.0, 'Pop the Top Craft Beer Shop')),
 (1248, (5.0, 'Seoul Food Meat Company')),
 (1828, (5.0, 'Haberdish')),
 (1856, (5.0, 'Two Scoops Creamery')),
 (2028, (5.0, 'Toyama Express')),
```

Following are the highest predicted rating scores and restaurants for this user. The interesting thing is ‘Not Just Coffee’ appears both in user’s high-rating records and predicted recommendation list.

```
# (restaurant_id,(pred_rating,resraurant_name))
```

```
[(5522, (5.336022369152394, 'Sospeso Coffee Roasters')),
 (3682, (5.135183741552405, 'Masa Casa')),
 (1398, (5.1311082252074005, "Waiter's Choice Catering")),
 (2706, (5.106847804319242, 'Java Joes')),
 (4203, (5.106561492133734, 'Chosen Roaster')),
 (1941, (5.098154800464003, 'Carolina Craft Butchery')),
 (3389, (5.061450340201568, 'Natural Blendz')),
 (557, (5.055129798022431, 'Food Truck Friday')),
 (2149, (5.050663935223758, 'Hot Rods Cafe')),
 (3024, (5.0491941078095195, 'BoCo Coffee Roasters')),
 (2554, (5.018870355372531, 'Cardais Gourmet')),
 (2711, (5.0104175532252695, "Le Cochon d'Or")),
 (4781, (5.004546136894303, 'Pour Olive')),
 (3899, (5.002446967377348, 'Cotswold Farmers Market')),
 (3987, (4.999620716364449, 'Winestore')),
 (2586, (4.993317672504332, 'Groundwork Common')),
 (2982, (4.981263067939209, 'Lupitas Carniceria & Tortilleria')),
 (1730, (4.966388717067602, 'Zippy Ice')),
 (3512, (4.960979820841706, 'Cloister Honey')),
 (2748, (4.959824798382823, 'Tacos El Regio')),
 (256, (4.956337425393287, 'Daily Mews Cat Cafe')),
 (4887, (4.950264965361071, 'Cookies N Creme')),
 (4109, (4.948715025544038, "Charlie's - The Coffeehouse LKN")),
 (3037, (4.943958558258881, 'Not Just Coffee - Providence')),
 (1784, (4.942784799047005, 'Las Lupitas')),
```

### 6.6.3 Collaborative Filtering - Item-Item Recommender

A user's neighborhood is a group of users who have similar rating behaviors. Rating is generated by the average rating of the neighborhood. The recommender gives recommendation by sorting the neighborhoods' average rating.

The user-item matrix was first created, which consists of rating of each user-item pair. Then, we built an item-item matrix which consists of similarity score for each pair of restaurant using CosineSimilarity from the sklearn package. The recommender calculate predicted rating for each restaurant and outputs top10 restaurants. For the user, "Ry1O_KXZHGRI8g5zBR3IcQ". The results are shown below. The item-item recommender works well for the active users since it is more likely to match the user with the neighborhood who have similar rating behavior. The prediction also improves when the overall sample size is large.

Table 9

| | |
|---|---|
| 'Fresh Off The Grill' | 'BW Sweets Bakery' |
| 'Burger King' | 'Domino's Pizza' |
| ' The Captain's Cap ' | 'Rushco Davidson Market' |
| 'Aly Baba' | 'Fox and Hound Smokehouse' |
| 'The Silver Grill' | 'Dim Sum China Express' |

**7. Observations**

The content based filtering is not based on the user features but rather the restaurant's textual description. The method is capable of generating restaurants that is similar to the restaurant that the user has rated, in this case the same cuisine categories. The method does not consider the user's rating or other's rating. If the user has a specific cuisine preference, for example Japanese, the method is able to provide useful recommendations. However, for uses who have lots of rating histories or does not show a certain cuisine category preference, the algorithm would not be ideal. In the experiment, we observe that none of the restaurant we recommended were in fact visited or rated by the user.

The collaborative filtering models are based on rating records data by all users. It can provide reliable recommendations from an overall view of user rating behavior patterns, without considering any other information, like restaurant profile and user profile. The most intuitive collaborative method is neighborhood collaborative filtering. But it depends on the sparsity of rating data, which is a real problem in our dataset, where active users are a small part. It can be solved by model-based collaborative method, which achieves a good performance in our experiment. The reason may be that even for users with sparse rating records, the latent factor can be learned and expressed well by optimizing the whole matrix.

**8. Future work and conclusions**

The sentimental analysis offers useful analysis model to learn elements that are essential to customers feeling about the restaurant. CNN model is also a good tool to predict whether a text belongs to positive or negative. However, these tools have their limitations, because some data is hard to interpred, and we need more concise parameters to apply these tools in practice.

The recommender system consists of three major components, content-based filtering, collaborative filtering and neighborhood collaborative filtering. The content-based filtering only considers restaurants' textual description. Collaborative filtering and item-item recommendation leverage the user's rating history as well as other users' rating. Among all three methods that we have experimented, collaborative filtering achieved 0.5 MSE between user's actual rating and predicted one. Collaborative filtering method is the recommended method for this recommendation system. When the user is relatively new to the community and the user information is limited, the content-based filtering would be the ideal method for recommendation. When there is a large amount of user rating records in the database, collaborative filtering can provide a comprehensive recommendation based on the overall rating behavior pattern of all users.

Though collaborative filtering achieved the state-of-art performance, all three methods did not take the users' attributes or restaurants' features into consideration. In the experiment, we found it was difficult to develop a testing mechanism for all users. Developing a testing system could be implemented in the future. Future work can also be done to incorporate users and restaurants' features in addition to restaurant categories and user's ratings. A hybrid approach can also be developed to include all methods that we have experimented.

**9. Contributions of project members**

Maggie Liu: Review table data exploratory analysis and content-based filtering

Xiang Li: User&Tip table data exploratory analysis and collaborative filtering

Yuqin Shen: NLP Preliminary data visualization and sentiment analysis

Xi Li: Business dataset exploratory analysis and neighborhood based filtering

**References**

[1] Hu, M. and Liu, B., 2004, August. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 168-177). ACM.
[2] Salinca, A., 2015, September. Business reviews classification using sentiment analysis. In *2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)* (pp. 247-250). IEEE.

[3] Roy et al., 1999,  Content-Based Book Recommending Using Learning for Text Categorization. *In Computing Research Repository.*
[4] Herlocker et al., 2004, January. Evaluating Collaborative Filtering Recommender Systems. *In ACM TOIS*. 22(1):5-53, 2004
[5] Bell et al., 2009, August. Matrix Factorization Techniques for Recommender Systems. *In IEEE Computer Society,* pp. 30-37, vol. 42
[6] Gomez-Uribe  et al., 2015, December. "The Netflix Recommender System". *In ACM Transactions on Management Information Systems*. 6(4): 1–19.
[7] Bouneffouf et al., 2013, DRARS, A Dynamic Risk-Aware Recommender System (Ph.D.), Institut National des Télécommunications
[8] Yin et al., 2013, August. LARS: A Location-Aware Recommender System. *In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 221-229)
[9] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2018. Deep Learning based Recommender System: A Survey and New Perspectives. ACM Comput. Surv. 1, 1, Article 1 (July 2018), 35 pages
[10] Shi, Chao, et al. “A Hybrid Recommender with Yelp Challenge Data -- Part I.” *NYC Data Science Acedemy*, https://nycdatascience.com/blog/student-works/yelp-recommender-part-1/.

Graph and Equation references:

1. https://en.wikipedia.org/wiki/Collaborative_filtering
2. https://jhui.github.io/2017/01/15/Machine-learning-recommendation-and-ranking/