Title
NLP Analysis and Recommendation System for Yelp

Author
Sun, Jiancong

Publication Date
2020

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

NLP Analysis and Recommendation System for Yelp

A thesis submitted in partial satisfaction
of the requirements for the degree Master of Science
in Applied Statistics

by

Jiancong Sun

2020

ABSTRACT OF THE THESIS

NLP Analysis and

Recommendation System for Yelp

By

Jiancong Sun

Master of Applied Statistics

University of California, Los Angeles, 2020

Professor Yingnian Wu, Chair

Yelp provides a valuable platform to share massive restaurant information, but it is difficult for its users to distinguish a relevant one among others. People are overwhelmed by multifarious information and unable to efficaciously glean germane information. Thus, it's necessary to build a recommendation model which can filter and prioritize information, efficiently recommending appropriate restaurants to Yelp's users, so the users can make correct decisions. Meanwhile, it can also help businesses to target their potential customers more accurately by sending similar-preference recommendations. This research explores different preferences and topics from Yelp restaurant reviews to understand characteristics of each user and restaurant, and then applies four practical algorithms to provide the most precise and personalized restaurant recommendations for the users.

The thesis of Jiancong Sun is approved.

Vivian Lew

Frederic R. Paik Schoenberg

Yingnian Wu, Committee Chair

University of California, Los Angeles

2020

To my family and friends,

for their love, support and understanding.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

"Yelp connects people with great local businesses by bringing "word of mouth" online and providing a platform for businesses and consumers to engage and transact," Yelp annual report. [1]

Yelp is an online business review platform operating worldwide and brings convenience to consumers by finding local businesses such as restaurants, local services, travel, auto, beauty, fitness and more. The customers leave ratings and reviews on Yelp, and at the same time, other users can sift and winnow apropos local businesses based on those comments as references. It is hard to select an optimal option surrounding by mass information at various platforms online. Therefore, Yelp is considered as a trustworthy platform for users because Yelp is able to offer accurate recommendations to users. That makes Yelp much more valuable and outstanding than other similar platforms.

Yelp has grown at a breakneck pace in the past 10 years and owned a major market share of the online business review market. According to the 2019 fourth-quarter earnings report of Yelp', 96 million users contributed 205 million reviews on 4.9 million active claimed local businesses. Of these 4.9 million businesses, 565,000 of them are paying for ads on this platform. In 2019, Yelp earned $1.014 billion in total net revenue, an 8% increase from the year 2018. Advertising is Yelp's main income source, which contributes 90% of the total revenue. [2]

In the past 10 years, the advertising industry has changed completely since the advertisers are shifting their budget to digital advertising from traditional advertising - TV, radio, mailers, etc.

In most cases, digital advertising revenue is calculated by the following formula:

$$\text{revenue} = \text{impressions (number of ads views)} * CPM\ (unit\ price\ for\ every\ 1000\ ads\ views)$$

In order to increase revenue, ad-tech companies focus on expanding the user base and increasing the unit price of ads viewed. Yelp could make their web application easier to use and provide more personalized information so it can attract more users. More importantly, companies would prefer to pay more for advertising if the advertiser could pinpoint a right group of customers. Yelp should analyze customers' behaviors and preferences from their ratings and reviews so that it can provide more accurate recommendations to users as well as a higher efficiency advertising product to its advertisers.

Recommendation system is an algorithm that suggests related items to a specific user given the user's preference. Since the recommendation system has to satisfy different needs of users, a good recommendation system usually combines different machine learning models. Location-based model, content-based model and collaborative filtering are the three of the most common recommendation models. Location-based recommendation model generates search results that are closest to the users' location; content-based model analyzes the user's preference to find the best option; collaborative filtering model provides suggestions based on the behaviors of similar users.

In this research, we are going to use the restaurant reviews from the Yelp dataset to discover valuable information for users and local businesses. To clean up the dataset, we

normalize the text, remove stop words and apply models such as Unigram, Bigram Trigram models and Latent Dirichlet Allocation (LDA). After data preparation, I would like to see what makes a five-star restaurant, what are the top 30 topics about restaurant reviews and how word vectors and algebra perform in restaurant review data. Finally, I will train the three recommendation algorithms with the cleaned reviews and check if they can return precise restaurant recommendations. If everything works well, we can construct the final recommendation system so that users will get the best restaurant recommendations.

CHAPTER 2

Methodology

2.1 Text Preprocessing

Text Processing is one of the most common tasks in Natural Language Processing (NLP) applications, which helps transform the raw text into something that machines can understand. This is the initial approach to sort out the Yelp review data.

2.1.1 Text Tokenization, Normalization and stopwords

The very first step in text preprocessing is text tokenization. Text tokenization is an algorithm that breaks down a long text string into individual words and punctuations by whitespace characters. In addition, it also splits complex abbreviations. It also contains a dictionary package to identify each token's lemma, shape, part-of-speech tag, log_probability and stopword.

TEXT	LEMMA	POS	TAG	DEP	SHAPE	ALPHA	STOP
I	-PRON-	PRON	PRP	nsubj	X	TRUE	TRUE
am	be	AUX	VBP	ROOT	xx	TRUE	TRUE
a	a	DET	DT	det	x	TRUE	TRUE
data	data	NOUN	NN	compound	xxxx	TRUE	FALSE
scientist	scientist	NOUN	NN	attr	xxxx	TRUE	FALSE
.	.	PUNCT	.	punct	.	FALSE	FALSE

Table 2.1.1: Example of text tokenization

Text normalization is then used to map raw text strings to canonical form. A single English word, such as “connect”, can have multiple forms: “Connect”, “connects”, “connected”, “connecting”, “connection”, “connectivity” and etc. Text normalization converts all these forms back to the original word “connect” without changing the meaning. It allows the NLP model to recognize these words with similar meanings.

2.1.2 Unigram, Bigram and Trigram models

Our languages convey messages not only word by word, but also combinations of phrases. Unigram, Bigram and Trigram models can connect words to present more accurate messages, and the algorithm bridges the communication between human and computer. They are statistical language models to predict the next word in such a sequence in the form of a (n-1) -order Markov model. [3] The formula to identify next word is finding the maximum of the probability:

$$\mathrm{P}([\text{the } next \text{ word}] \mid [previous\ n-1\ word])$$

Denote the Nth word be W_n, the probability can be written as:

$$\mathrm{P}(W_n \mid W_{n-1}) = \frac{C(W_{n-1}W_n)}{\sum_w C(W_{n-1}W)}$$

Simplify this equation, we have:

$$\mathrm{P}(W_n \mid W_{n-1}) = \frac{C(W_{n-1}W_n)}{C(W_{n-1})}$$

The computation will present all combinations of W_{n-1} and W_n as a phrase if $\mathrm{P}(W_n \mid W_{n-1})$ is more than a given default threshold. [4]

The N-gram models are commonly used to capture phrases and link them together as a single word by an underscore, such as New_York, prime_rib, happy_hour, etc.

2.1.3 Latent Dirichlet Allocation (LDA)

We use Latent Dirichlet Allocation (LDA), a generative probabilistic model for collections of discrete data such as text corpora. LDA is a three-level hierarchical Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of topics. Each topic is, in turn, modeled as an infinite mixture over an underlying set of topic probabilities. In the context of text modeling, the topic probabilities provide an explicit representation of a document.

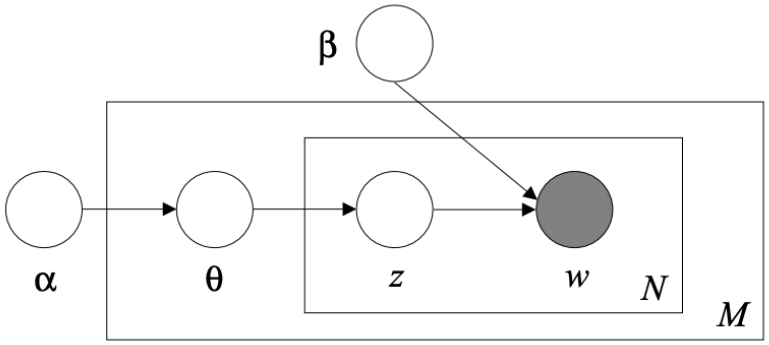


Figure 2.1.3: Graphical model representation of LDA.

The boxes are "plates" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.

A k-dimensional Dirichlet random variable θ can take values in the (k −1)-simplex (a k-vector θ lies in the (k−1)-simplex if $\theta_i \geq 0$, $\sum_{i=1}^{k} \theta_i = 1$), and has the following probability density on this simplex:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \theta_1^{\alpha_{1-1}} \dots \theta_k^{\alpha_{k-1}}$$

Where $p(z_n|\theta)$ is simply θ_i for the unique i such that $z_n^i = 1$. Integrating over θ and summing over z, we obtain the marginal distribution of a document:

$$p(w|\alpha, \beta) = \int p(\theta|\alpha)(\prod_{n=1}^{N} \sum_{Z_n} p(Z_n|\theta)p(w_n|Z_n, \beta))d\theta$$

Finally, taking the product of the marginal probabilities of single documents, we obtain the probability of a corpus:

$$p(D|\alpha, \beta) = \prod_{d=1}^{M} \int p(\theta_d|\alpha)(\prod_{n=1}^{N_d} \sum_{Z_{dn}} p(Z_{dn}|\theta)p(w_{dn}|Z_{dn}, \beta))d\theta_d$$

The LDA model is represented as a probabilistic graphical model in Figure 2.1.3. As the figure makes clear, there are three levels to the LDA representation. The parameters α and β are corpus level parameters, assumed to be sampled once in the process of generating a corpus. The variables θ_d are document-level variables, sampled once per document. Finally, the variables z_n^d and wdn are word-level variables and are sampled once for each word in each document. [5]

2.1.4 Word vectoring

A word vector is a row of real valued numbers where each point captures a dimension of the word's meaning and where semantically similar words have similar vectors. With word embedding, vectors will learn about the meaning of the terms and the relationships between terms in the vocabulary. By learning the meanings and relationships independently from any previous background knowledge, a word vector model is an unsupervised model.

Word2vec helps to measure the quality of the resulting vector representations. This works with similar words that tend to close with words that can have multiple degrees of similarity. The algorithm uses Feed Forward Neural Net Language Model (NNLM) and Recurrent Neural Net Language Model (RNNLM) to maximize the accuracy and minimize the computation complexity. [6]

Word2vec has the following three user-defined hyperparameters:

1. The dimensionality of the vectors. Typical choices include a few dozen to several hundred.
2. The width of the sliding window, in tokens. Five is a common default choice, but narrower and wider windows are possible.
3. The number of training epochs.

The Word2Vec model can potentially be implemented in a recommendation model when we want to find the meaning and relationship between some words.

2.2 Recommendation Models

After the completion of text processing, recommendation models are applied to provide the most accurate results to the given users by analyzing useful information such as location, preference and relation. The models discover data patterns by studying users' choices and show the suggestions that match with their needs and interests. We will discuss further in the following section.

2.2.1 Location-based Recommendation: K-means clustering

In the location-based recommendation model, the algorithm only relies on one variable - the location (longitude and latitude) of the Yelp users. The K-means clustering model is a good fit for the case. K-means clustering is an unsupervised algorithm that can group data points together into K different clusters.

The first step is applying the “elbow” method to find the optimal number of clusters (K). The concept of elbow method is to run the K-means clustering model for a range of values of K and calculate the sum of squared errors (SSE) for each value of K. We can determine the optimal number of cluster K by finding a K that is small and its SSE is relatively small enough.

The main idea of K-means algorithm is to find the best location of center points that can minimize the “distance within the same cluster” while it maximizes the “distance between different clusters”. The logic behind the model is as following:

1. Choose number of clusters K by elbow method
2. Randomly assign K centroids in the data
3. Assign all points to the nearest cluster centroid
4. Recompute centroids of newly formed clusters

5. Repeat step 3,4 until centroids do not change

When the model is well trained, it will return the cluster and corresponding data points with weighted and filtering logics. [7]

2.2.2 Content Based Recommendation

By analyzing behaviors, the content-based recommendation model links users' preference with the end results of great similarities.

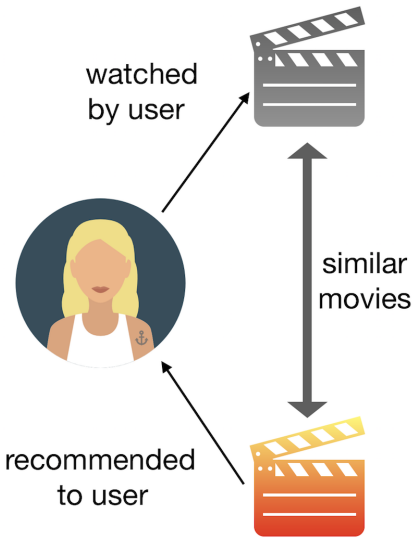


Figure 2.2.2: basic idea of content-based recommendation

Given the characteristics of the users, we can adopt the Cosine Similarity method to find the most relevant subjects. After converting the information of both users and subjects into vectors, we will apply the formula below to compute the similarity score:

$$\text{similarity} = \cos(\theta) = \frac{u * v}{\|u\|\|v\|} = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2}\sqrt{\sum_{i=1}^{n} v_i^2}}$$

By the definition of similarity, it will be 1 if the two vectors are identical, and it will be 0 if the two are orthogonal. In other words, the similarity is a number bounded

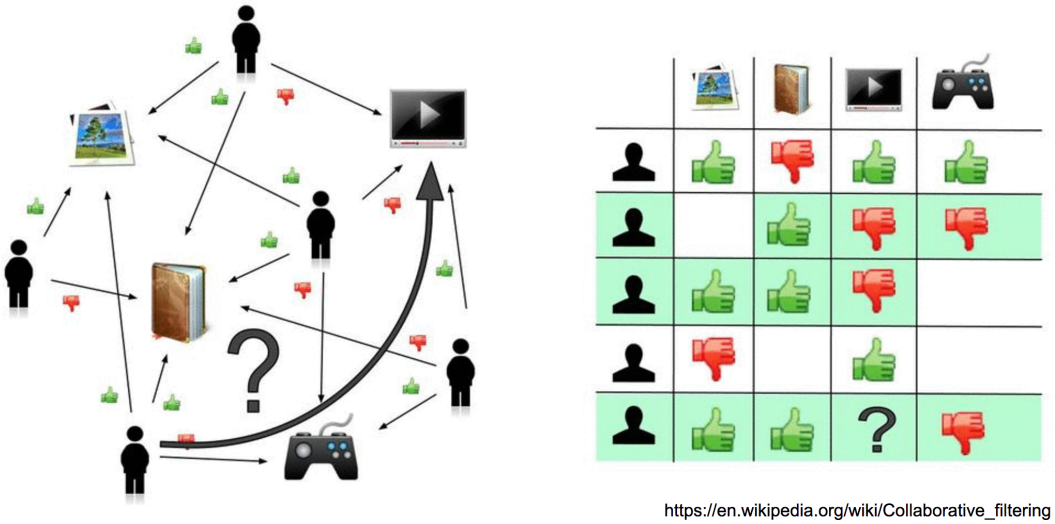
between 0 and 1 that tells us how much the two vectors are similar. The model will output the results by the order of similarity scores. [8] To ensure the accuracy of the recommendation results, we can review if the features of the subjects accommodate the preference of users. For example, if the input data includes an address, the model should return the outputs near the given location.

A shortcoming of this model is that its recommendations heavily rely on the historical behaviors and preferences. The results will be very similar to the users' previous items, less diverse options.

2.2.3 Collaborative Filtering Recommendation

Collaborative filtering is one of the recommendation systems to predict the interests of users by examining the selections of the similar users. The underlying assumption is that if person A agrees with person B on many subjects, A has a greater chance to share the same opinions with B on other subjects, compared to those unrelated users.

Figure 2.2.3 is a visual example. In order to predict the unknown rating on the fifth row', we observe the opinions from other users with similar rating histories (green rows). As a result, the answer is a negative rating with a thumbs-down.



https://en.wikipedia.org/wiki/Collaborative_filtering

Figure 2.2.3: basic idea of collaborative filtering recommendation

Mathematically, there are many different models that can be applied to find the similar users. Take Singular Value Decomposition (SVD) as an example, it has better performance in many different use cases. SVD is a matrix factorization technique that is usually used to reduce the number of features of a data set by reducing space dimensions from N to K where K < N. For the purpose of the recommendation systems however, we use SVD in the matrix factorization part keeping the same dimensionality. The matrix factorization is done on the user-item ratings matrix. From a high level, matrix factorization can be thought of as finding 2 matrices whose product is the original matrix.

Each item can be represented by a vector q_i. Similarly, each user can be represented by a vector $p_u$such that the dot product of those 2 vectors is the expected rating

$$\text{expected rating} = r_{ui} = q_i^T * p_u$$

$q_i$and $p_u$can be found in such a way that the square error difference between their dot product and the known rating in the user-item matrix is minimum. [9]

$$\text{minimum } (\mathrm{p}, \mathrm{q}) \sum_{(u,i)\in K} (r_{ui} - q_i^T * p_u)^2$$

In application, Surprise is a useful python library that can implement SVD, KNN and other algorithms for different recommendation models.

CHAPTER 3

Dataset Overview

The dataset sources from Yelp's businesses, reviews, users and checkin data. It was originally distributed for the Yelp Dataset Challenge which was an event for students to share their data analysis. This dataset contains information of businesses across 11 metropolitan areas of four countries.

The complete dataset includes 6.68 million reviews from1.6 million users evaluating 192,000 businesses from 2004 to 2018. Within these 192,000 businesses, the top 5 categories are restaurants, shopping centers, home services, Health & Medical and Beauty & Spas. In addition, the top 5 cities ranking by the number of local businesses are Las Vegas, Toronto, Phoenix, Charlotte and Scottsdale.

The goal of the research paper is building a recommendation model focusing on one business type in one location. We are going to analyze 1,253,154 reviews from 440,130 users about the 6,786 restaurants in Las Vegas.

The original dataset is a combination of multiple json files. In order to process the data easier and faster, I created a database and inserted all the data inside. The join keys are business_id, review_id and user_id. Business_id in business table and user_id in user table are unique, but review_id can be duplicated since a user can leave multiple reviews for the same business. In addition, the Parsed_review table has similar data with the review table, but the review texts have been cleaned by Unigram, Bigram, Trigram and spacy package.

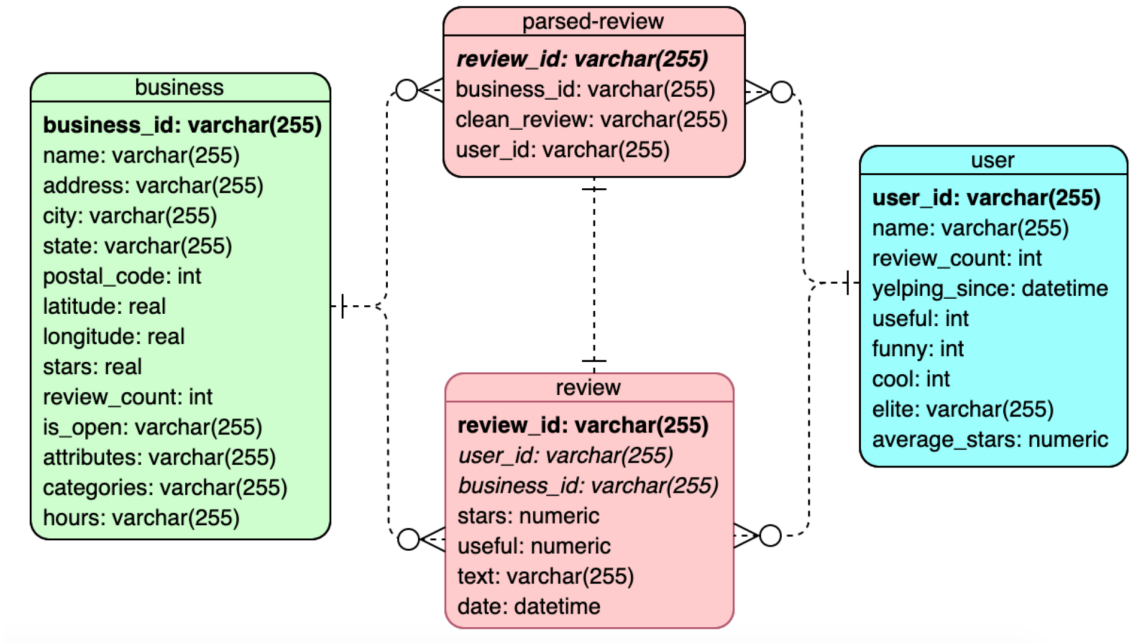


Figure 3: Entity Relationship Diagram (ERD) design of Yelp database

CHAPTER 4

Experiment

4.1 EDA and Visualization

In this section, we conduct preliminary exploratory data analysis ("EDA") of the dataset to identify if any significant factors could impact the recommendation models. We notice that the Yelp ratings and the number of reviews would play an important role as variables. Also, we utilize visualization tools to present the most frequently used texts of the reviews for good and bad restaurants, which may set up the topics of the models.

There are 30,995 restaurants in Las Vegas with an average 3.68 rating in Yelp. 25,176 of them remain in business while 5,819 restaurants have been closed due to various reasons. What are the differences between open and closed restaurants? From Figure 4.1(a), we can see a higher portion of open restaurants have ratings 3.5 and above, averaging at 3.71. Meanwhile closed restaurants have lower average Yelp rating score at 3.58. This is expected since customers would choose the restaurants with higher ratings.

However, from Figure 4.1(b), it is interesting to see that most of those highly rated restaurants don't have as many reviews as we expected. It is also worth noting that some businesses with 5-star ratings are closed too. This phenomenon indicates that the quality of the restaurants may not completely match with the Yelp ratings. The count of reviews could be another consideration of identifying a good restaurant. If the volume of the reviews is not sufficient, it is hard to make decisions solely based on ratings.

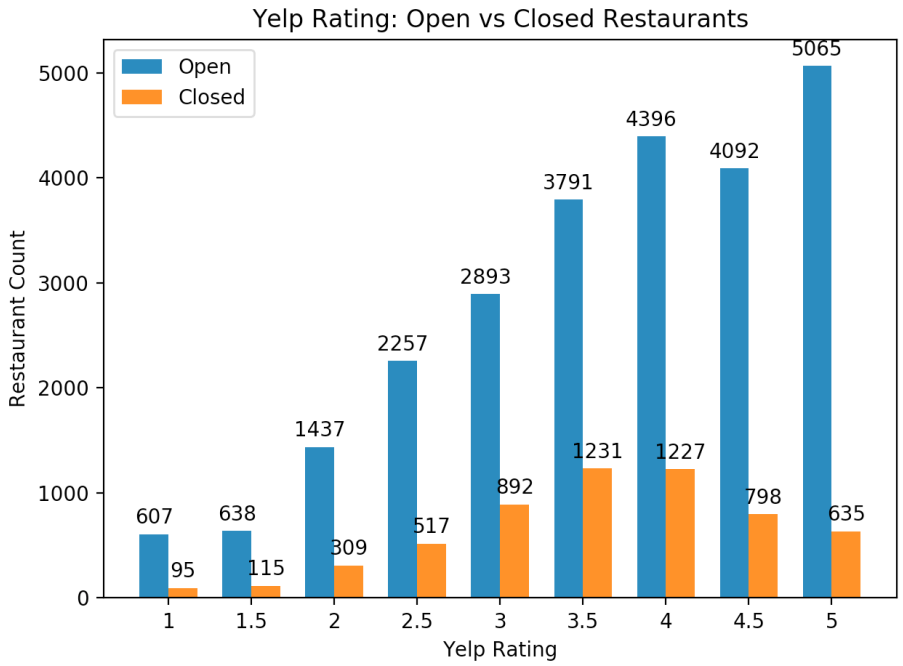


Figure 4.1(a): Number of restaurants in each Yelp rating group

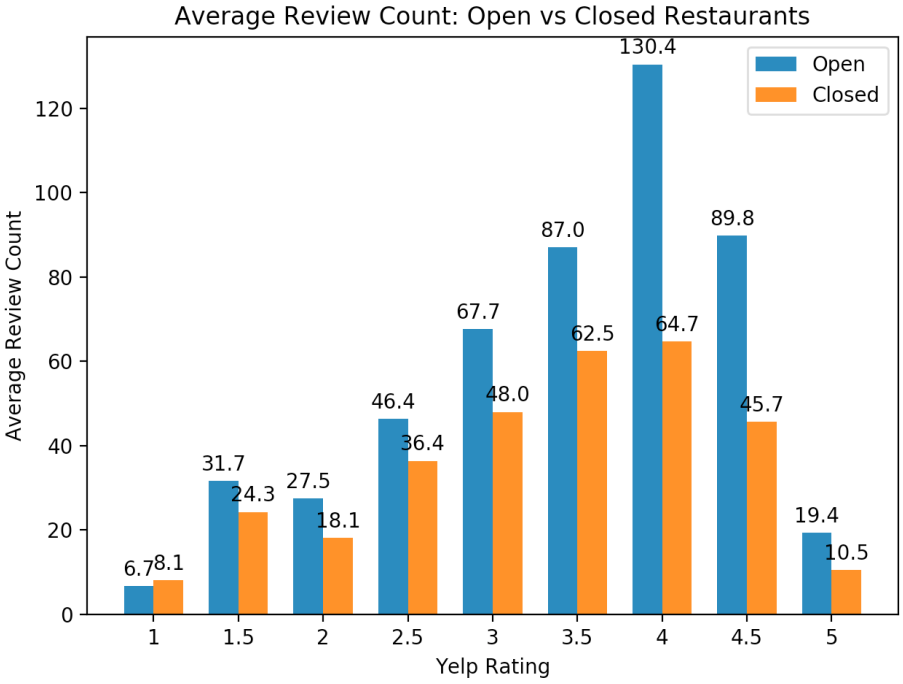


Figure 4.1(b): Average review count in each Yelp rating group

Furthermore, it is important to see the keywords or topics from the reviews to understand more about the preference of the users. I filter all reviews that have 4.5+ stars for good restaurants and 2-stars for bad restaurants. The first word-cloud plot Figure 4.1(c) is the most frequent words in the good restaurant reviews, and larger font means higher word frequency. We can see many people care about the great food quality, friendly customer service and enjoyable experience. On the other side, Figure 4.1(d) shows that many customers rate 1-2 stars when they comment on little meals and rudeness. People also tend to give lower ratings for fast food restaurants.

Figure 4.1(c): Most frequent words in good reviews with 4.5 stars or above

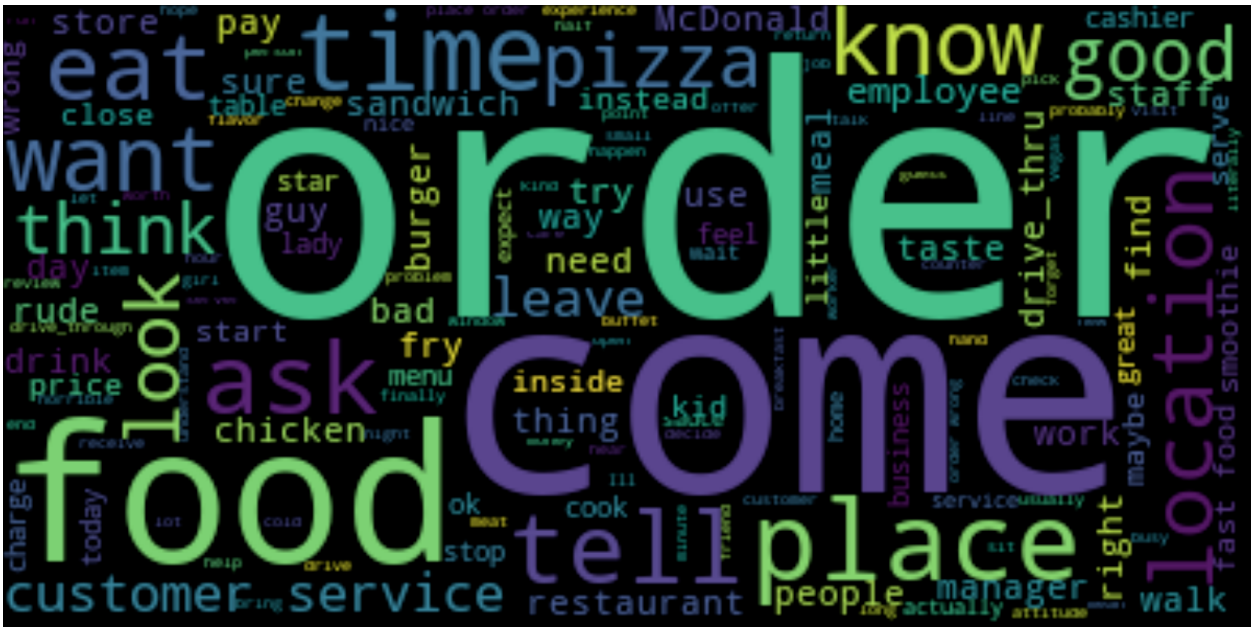


Figure 4.1(d): Most frequent words in bad reviews with 2 stars or less

4.2 Data Preparation for Modeling

4.2.1 Text Processing

We use spaCy and gensim packages to break down the paragraphs into words and perform changes such as text normalization and removing stop words. Given a full text data, the first step is to create unigram, bigram and trigram models. Unigram normalizes all text data, such as removing all stop words, and both bigram and trigram models connect words as phrases.

Take one of the reviews as an example:

Figure 4.2.1: Example workflow of text normalization, N-gram model and cleaning

From the example above, we can see that Unigram converts all verbs to present tense, all plural nouns to singular and all kinds of pronouns to the same word '-PRON-'. Then our Bigram model identifies all the two word phrases and connects them together with underscore, such as 'Saturday_morning', 'bowling_league' and 'pin_ball'. Trigram model links more relevant words than Bigram to better convey the messages, such as 'pin_ball_machine'. The final step is to remove all stop words, which are commonly used but meaningless, such as "be" and "to". With the text processing, the algorithm can transform the original text to shorter and unified word strings/vectors so that the data can be consumed by the LDA model.

4.2.2 Topic Modeling

After text pre-processing, we apply the Latent Dirichlet allocation (LDA) model to the cleaned restaurant review texts and try to group the words into topics. Python library 'gensim' is a very useful tool in topic modeling. We use the gensim Dictionary to generate a bag-of-words representation for each review, then save the bag-of-words reviews as a matrix.

With the bag-of-words corpus, we need to pass the bag-of-words matrix and Dictionary from our previous steps to LdaMulticore as inputs, along with the number of topics. I generated 10, 30 and 50 topics for the restaurant review texts. After examination, I found that 30 topics would have better accuracy. These 30 topics are related to different types of food, environment, services, etc. as shown below at Figure 4.2.2. If a given text string is input in the model, it can be broken down to a list of topics with probability.

```
topic_names_30 = {
    0: u'experience', 1: u'sandwich', 2: u'customer service', 3: u'asian',
    4: u'breakfast', 5: u'discount', 6: u'value', 7: u'pizza', 8: u'burger', 9: u'menu',
    10: u'chinese', 11: u'food quality', 12: u'thai', 13: u'buffet', 14: u'hotel', 15: u'steak',
    16: u'sushi', 17: u'location', 18: u'bar', 19: u'feeling', 20: u'customer service',
    21: u'italian', 22: u'fine dinner', 23: u'dessert', 24: u'wing', 25: u'kid', 26: u'BBQ',
    27: u'mexican', 28: u'price', 29: u'environment'
}
```

Figure 4.2.2: 30 topic names for LDA generated topics

For example, if the input is

"I ordered chicken curry today, the food is tasty. and the restaurant has free parking."

The model will generate:

"[('discount', 0.168443), ('thai', 0.431476), ('location', 0.249965)]".

The result shows that the sample text is 16.8% related to discount, 43% related to Thai food, and 25%related to location. Since all these models are unsupervised, they can learn everything by itself. We will use these outputs for further analysis in text vectorization and content-based recommendation models.

4.2.3 Word vectoring by Word2Vec

After getting the topics of each review, we also want to know how well the computer can understand words in a restaurant view. Among all the words embedding models, word2vec is a very popular one. Word2vec is able to interpret meanings of the target words through the words before or after the target words.

```
In [36]: get_related_terms('steak')
sirloin              0.777
filet_mignon         0.777
filet                0.773
porterhouse          0.76
ribeye               0.749
skirt_steak          0.748
rib_eye              0.748
stake                0.721
top_sirloin          0.714
prime_rib            0.711
```

```
In [45]: get_related_terms('happy_hour')
Happy_Hour           0.93
hh                   0.863
HH                   0.859
reverse_happy_hour   0.852
during_happy_hour    0.827
happy_hr             0.818
3pm-6pm              0.813
4_7pm                0.789
during_Happy_Hour    0.781
3_6pm                0.78
```

Figure 4.2.3(a)(b): word2vec model to find similar terms for 'steak' and 'happy_hour'

We trained the Word2Vec model on all text cleaned by N-gram models, using 100-dimensional vectors and setting up our training process to run for twelve epochs. The trained Word2Vec model contains the matrix of 100 dimensions of each word appearing in the cleaned dataset. With that, we can get the related terms of a given word. For example, finding related terms for "steak" Word2Vec will return different kinds of steak

names such as sirloin, filet_mignon, ribeye, etc. In another example, Word2Vec notices alternative spellings for happy and more importantly, so it has discovered that the concept of happy hour is related to the block of time around 3-7pm.

```
In [47]: word_algebra(add=['beef','vegetable'], subtract=['meat'], topn=1)
tofu
```

Figure 4.2.3(c): word2vec example: 'beef' - 'meat' + 'vegetable' = 'tofu'

Word2Vec can also understand the relationship between some words by vector addition and subtraction. What's the result of "Beef" - "meat" + "vegetable"? Word2Vec's answer is "Tofu". Both tofu and beef have similar textures, and Tofu was made of soy which is vegetable. The subtraction remains the taste, protein of the beef, and addition limit the answer should be made from some kind of vegetable.

```
In [49]: word_algebra(add=['filet_mignon','seafood'], subtract=['beef'], topn=1)
lobster_tail
```

Figure 4.2.3(d): word2vec example: filet_mignon - 'beef' + 'seafood' = 'lobster_tail'

In another case, what would the model recommend when a filet mignon lover would like to eat seafood instead of meat? The Word2Vec model will return "lobster tail". The model learned the concept of delicacy. Subtracted by "beef", "Filet mignon" left with a vector that corresponds to delicacy and high end. When we add "seafood" to it, it contains the meaning of delicious and high-class seafood, that is, lobster tail.

The Word2Vec model will be a powerful tool to enhance our recommendation models because it really understands meanings and relations of all words.

4.3 Training Recommendation Models

4.3.1 Location Based Model

The location-based model uses the K-means model in Python sklearn library to cluster the restaurants in Las Vegas by their longitudes and latitudes. It then ranks the restaurants in each cluster by star-rating. To make a recommendation, the model needs to identify the input address and assign a location cluster based on longitudes and latitudes. The results provide the users with the top-ranked restaurants nearby.

To determine the number of clusters, an "elbow plot" Figure 4.3.1(a) was used by fitting the silhouette scores for a series of k, ranging from 2 to 25. The higher the scores, the more distortion from the data. The reasonable number of clusters k should have relatively low scores and sufficient clusters to represent the locations. According to the plot, if k is greater than 10, the distortion will not be significantly lower. Hence, we use K=10 as an example.

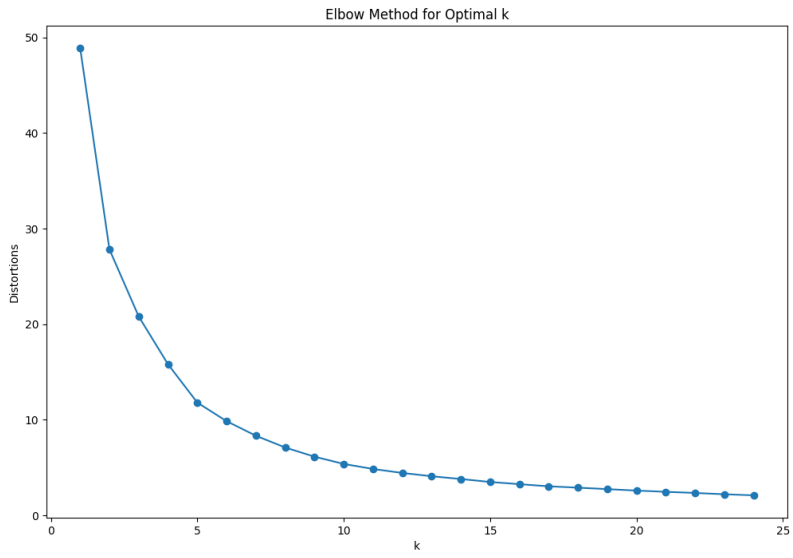


Figure 4.3.1(a): Elbow plot to determine the number of clusters

	cluster	name	stars
2605	0	Fish In the Spotlight	5.0
5717	0	La Herradero 2	5.0
3401	1	Lisa Maries Catering Services	5.0
5724	1	Soul Food Cafe Express	5.0
181	2	Those Guys Pies	5.0
193	2	Queen Tacos	5.0
4423	3	Murdock Meals	5.0
1845	3	Yummy Chinese restaurant	5.0
3505	4	Snow Ono Shave Ice	5.0
2758	4	El Camaron Jarocho	5.0

Figure 4.3.1(b): Top 2 ranking restaurants in first five location clusters

After training the model with K=10, it groups all restaurants into 10 location clusters. When a user requests recommendation, the model runs the algorithms with the given input of location (either address or coordinate) and returns the most popular and highly-rated restaurants within the same cluster. This model can be effectively used for new users without historical information but with location.

4.3.2 Content-based Model

The content-based recommendation focuses on finding the similarities among the test set, the restaurants not reviewed by the user, and the training set, the restaurants that the user has reviewed. Based on how a user ranks the previous restaurants, the model

predicts whether the user will like a new restaurant then makes some appropriate recommendations.

Users' reviews are imported to the model which generates a matrix of correlations between review texts and LDA topics. The matrix can be grouped by user_id and busines_id, so the attributes of restaurants and users can be separated as profiles. All correlations are quantified between 0 and 1 for each topic from the reviews, and cosine similarity can analyze those correlations to reveal the relationship between restaurants and the users' profiles. Eventually, the content-based model can match an user's profile with restaurants that share a similar profile, so decent and correct recommendations of restaurants will be delivered to the user.

For example, one of the users posted the reviews as shown below. The content-based model will recommend the restaurants in the following graph:

"...fast service seat fact different kind french_fry chicken strip wander place fairly_inexpensive burger great evening Penn_Teller..."

"...think good fettuccine_Alfredo life sauce buttery creamy excellent day..."

"...wow margarita nachos steak excellent..."

"...personally opt combo plate taco tostada good chance..."

	business_id	name	address	city
0	--9e1ONYQuAa-CB_Rrw7Tw	Delmonico Steakhouse	3355 Las Vegas Blvd S	Las Vegas
1	eDn45jTzYgCXhG4a_1wykQ	Bottles & Burgers By Double Helix	450 S Rampart Blvd, Ste 120	Las Vegas
2	eDK7ns2bB8pmQCoZMy3Idg	San Salvador Restaurant	2211 S Maryland Pkwy	Las Vegas
3	eBtEx6IQsQDoIDJXTDKdXA	Arbys Roast Beef Sandwich Restaurants	4830 S Fort Apache Rd	Las Vegas
4	eBj_YyJU5jVu6tbZCkdtDA	Brio Tuscan Grille	420 S Rampart Blvd, Ste 180	Las Vegas
5	eAc9Vd6loOgRQolMXQt6FA	Mandalay Bay Resort & Casino	3950 S Las Vegas Blvd	Las Vegas
6	e9q0RPoKoja0C4Q_a6cnZA	Smokes Poutinerie	725 Las Vegas Blvd S	Las Vegas
7	e9julbRI_7QEI5WcFqBvYw	Cafe Mitz	4550 S Maryland Pkwy	Las Vegas
8	e9gaoUQEws5tmQROZodZMg	Manhattan Pizza II	4955 E Craig Rd	Las Vegas
9	e8aRJbv2EMH5DqMzbDK8wQ	Taco Bell	6010 W. Tropicana	Las Vegas

Figure 4.3.2: Top 10 recommended restaurants from content-based model

From these posted reviews, it is obvious to see that the user prefers Mexican and Italian food. The model successfully provides related suggestions like San Salvador, a Mexican restaurant, and Brio Tuscan Grille, an Italian restaurant. Also, the user has mentioned “inexpensive” before, and as a result the model proposes Taco Bell which is a cheap fast food chain. Clearly, the model has the great ability to offer good recommendations.

To make the content-based model more sophisticated, we also included an option to add location constraint to this recommendation model, i.e. highly recommended restaurants within a certain radius to the location of the users.

4.3.3 Collaborative Filtering

Collaborative filtering recommendation model learns user ratings of different restaurants and compares them with the ratings from the other users. The model finds similar users and uses the ratings from these users to predict the preference of the given user for a specific restaurant.

In Python, we use library Surprise to find the best clustering model based on RMSE, MAE and fitted time. According to Table 4.3.3 below, we see that SVD and SVD++ models perform better since they have lower RMSE and MAE. Even though both models have similar performance, the fitted time for SVD++ is way more than the SVD model, which means longer processing time. Therefore, SVD is the best model for collaborative filtering.

	test_rmse	test_mae	fit_time	test_time
KNN Basic	1.0306	0.7968	0.27	7.35
KNN Baseline	0.9834	0.7587	0.86	8.89
KNN WIth Means	0.9913	0.7655	0.40	9.69
SVD	**0.9633**	**0.7510**	**23.65**	1.90
SVDpp	**0.9646**	**0.7499**	**748.89**	31.52
SlopeOne	0.9998	0.7701	7.60	21.41
NMF	1.0340	0.8014	22.66	1.40

Table 4.3.3: Performance for different models

Incorporated with SVD model, GridSearchCV is used to find the best parameter for this data:
(n_epochs=25, lr_all=0.01, reg_all=0.4). It is the ideal combination for the well trained SVD model.

Similar to the content-based model, we add an optional location filtering for our collaborative filtering model. Yelp users can enter an address and set the distance limit as advanced search, so the model will return the best matched restaurants nearby.

4.4 Recommendation System Design

Nowadays, the number of mobile users is increasing rapidly and much more than the number of web application users. People open their mobile app anytime and anywhere to search for their needs. When I feel hungry, it's convenient to pull up my cell phone to

look up any suggested restaurants from the apps. From the business perspective, they prefer their advertising to be seen by more potential consumers. Recommendation models help target more users that would have higher probabilities to try the restaurants. Thus, it is a win-win situation for both users and businesses.

When a user opens Yelp's website, he could be a guest, new user or a registered user. Also, he may or may not provide the location information. Given all different scenarios, this recommendation system is designed to provide the most applicable and accurate results to our users. The content-based and collaborative filtering model considers up to two factors: user_id and address (optional). And for a user_id, it can be divided into a new user (less than 10 reviews) or a frequent user (more than 10 reviews). In summary, we have four different cases as presented below:

	without address	**with address**
new users	top restaurants in the City	location_based
frequent users	collaborative filtering content_based	collaborative filtering w/ distance content_based w/ distance

Table 4.4: Recommendation model selection based on use cases

If the new user has no location information, the system can only provide a list of best restaurants around the city. However, if a frequent user is looking for restaurant recommendations, the system will use both collaborative filtering and content based-models to recommend restaurants.

This design can maximize the utilization of available users' information to make precise restaurant recommendations. In addition to the ability of recommending potential restaurants with different models, we also add logics to increase the diversity of results.

A user may go through the first 10 recommendations but don't see anything he likes. In order to give our user more options, the recommendation system will return different results when the user refreshes the page or calls the API again. All the new results will come from the top 100 best matched restaurants from the model.

Another option of diversification is to include 75% results from recommendation models and 25% from various options in general. It helps users to explore their preferences and businesses to attract different consumers.

4.5 Examples for Restaurant Recommendations

After we design all our models and combine them as a recommendation system, we would like to test a few examples and check the performance of our final system.

Example:

user_id = 'tL2pS5UOmN6aAOi3Z-qFGg'

address = '3655 S Las Vegas Blvd, Las Vegas, NV 89109'

Dist_limit = 1 (mile)

Result for Content Based with Location:

	business_id	name	address	city	stars	distance
59	9avLLw9uke50m8qO9doyMQ	Kings Sausage	917 Fremont St	Las Vegas	5.0	0.498489
46	oKATEDDmUGDDr-tIbDX5Pg	Rachels Kitchen	888 W Bonneville Ave	Las Vegas	5.0	0.636674
24	eRKVkDsHSFjaej3j7H0HyA	Taqueria La Herradura	1436 E Charleston Blvd	Las Vegas	5.0	0.661485
30	UZsxZdWt0quvtN0cAmiVUw	Pos Paque Takos	1468 E Charleston Blvd	Las Vegas	5.0	0.718978
86	Fmij544FE1iOruoxI41kew	Pepito Shack	1516 S Las Vegas Blvd	Las Vegas	5.0	0.925476
49	G4hjhtA_wQ-tSOGpgGlDjw	Bajamar Seafood & Tacos	1615 S Las Vegas Blvd	Las Vegas	5.0	0.966631
45	07UMzd3i-Zk8dMeyY9ZwoA	Art of Flavors	1616 S Las Vegas Blvd, Ste 130	Las Vegas	5.0	0.999110

Figure 4.5(a): Top recommended restaurants from content-based model given user, address and radius

Result for Collaborative Filtering with Location:

Out[253]:

	business_id	name	address	city	stars	est_score	distance
68	1CaM8eIvl41l4f3V-V-cAw	Smooth Eats	124 S 6th St, Ste 160	Las Vegas	4.5	4.178216	0.326270
62	zpoZ6WyQUYff18-z4ZU1mA	The Goodwich Downtown	900 S Las Vegas Blvd, Ste 120	Las Vegas	4.5	4.186318	0.355151
66	HY1qcwlWLkH2_4dIWjCmQQ	D E Thai Kitchen	1108 S 3rd St	Las Vegas	4.5	4.182082	0.569910
42	eRKVkDsHSFjaej3j7H0HyA	Taqueria La Herradura	1436 E Charleston Blvd	Las Vegas	5.0	4.232009	0.661485
82	5WjFHat0vyx0OYtGbTpyag	Pacific Island Taste	1428 E Charleston Blvd	Las Vegas	4.5	4.155067	0.667365
25	G4hjhtA_wQ-tSOGpgGlDjw	Bajamar Seafood & Tacos	1615 S Las Vegas Blvd	Las Vegas	5.0	4.263323	0.966631

Figure 4.5(b): Top recommended restaurants from collaborative filtering model

Since the user is an old user and he provides an address, our recommendation system uses both content-based and collaborative filtering with location filtering models for prediction. From the results above we can see both models return similar types of suggestions, and surprisingly there is a common suggestion - Bajamar Seafood & Tacos. This proves that although the algorithm logic is different, they are both useful and accurate in the recommendation system.

CHAPTER 5

Conclusion

5.1 Conclusion

The objective of this paper is to analyze Yelp reviews for restaurants in Las Vegas and discover information that can benefit users and restaurant owners. During the process, we build various recommendation models based on the Yelp reviews from restaurants in Las Vegas.

From the initial observation of the Yelp review data, we realize that open restaurants have slightly higher averaging star ratings than closed ones, but the difference is not significant. Many new restaurants started with 5 stars but went down to 3.5 to 4 when the count of ratings increased. In addition, we learn that people care more about high food quality, good service and a good environment of a restaurant by comparing high frequent words in ratings among good and bad restaurants.

To analyze the full dataset, we build a SQLite database to store all Yelp data from json files. Then we read the data into python to process text information using natural language processing and word embedding algorithms. Thus, massive information of users and restaurants can be simplified and extracted to develop recommendation models. With emphasis on the restaurants in Las Vegas, we develop three recommendation models for different circumstances. The location-based model chooses the best restaurants near the given location. Content-based model learns the topics the users care about and suggests the corresponding restaurants that match all their needs.

The collaborative filtering model suggests the top restaurants rated by users that have similar interests as the given user.

My restaurant recommendation system combines all three models to predict the interests of the users. It is a comprehensive solution for consumers, restaurants and advertising companies like Yelp. Customers can pick their favorite restaurants by a few clicks on Yelp. It also increases restaurants' exposure rate to their potential customers via the platform. If a restaurant would like to attract more customers by advertising, our recommendation system could ensure that the right and accurate advertisement content would be delivered to their target customers. In other words, by increasing the advertising efficiency, the advertising company Yelp can also charge their ads at a higher rate. In short, the adoption of recommendation systems can expand the number of active users in Yelp and subsequently increase the income for the business owners and the revenue for advertising platforms.

5.2 Further Works

5.2.1 Extension of the recommendation system

Our current recommendation system is built only for restaurants in Las Vegas. As we discussed earlier, there are many other business categories such as beauty & spa, home services, local services and healthcare. The recommendation system should be able to work in each category, so users can search for diverse topics. After the model is extended and stabled, we can promote it to all other places that speak English. If the

recommendation system performs perfectly, it can also be extended to other popular languages such as Spanish and Chinese in foreign markets.

5.2.2 Sentiment analysis on user reviews

Another valuable potential is doing sentiment analysis on text data of reviews. Through the sentiment analysis, it is easier for us to understand what the users like or dislike about restaurants. Then we can improve our recommendation system and provide more precise results. In addition, business owners can also know the preference of their customers in detail. It will be a very useful guidance to show business owners their advantages and weaknesses.

5.2.3 Word vector recommendation model

As stated before, vector representations of words can help identify relations between different verbatim comments and reviews being analyzed. Word embeddings like Word2Vec also help figure out the specific context in which a particular comment was made. The Word2Vec model will use the similarity and vector algebra to recommend the best foods for the users.

Reference

[1] Finance.yahoo.com. 2020. *Yelp Inc. (YELP) Profile*. [online] Available at: <https://finance.yahoo.com/quote/YELP/profile/>.

[2] Yelp-ir.com. 2020. *Yelp Inc. - Investor Relations*. [online] Available at: <https://www.yelp-ir.com/overview/default.aspx>.

[3] En.wikipedia.org. 2020. *N-Gram*. [online] Available at: <https://en.wikipedia.org/wiki/N-gram>.

[4] Jurafsky, D. and Martin, J., 2020. *Speech And Language Processing*. [online] Web.stanford.edu. Available at: <https://web.stanford.edu/~jurafsky/slp3/3.pdf>.

[5] Blei, D., Ng, A. and Jordan, M., 2020. *Latent Dirichlet Allocation*. [online] Jmlr.org. Available at: <http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf>.

[6] M, M., 2020. *Introduction To Recommendation Systems And How To Design Recommendation System,That Resembling TheAmazon* [online] Medium. Available at: <https://medium.com/@madasamy/introduction-to-recommendation-systems-and-how-to-design-recommendation-system-that-resembling-the-9ac167e30e95>.

[7] Kumar, V., 2020. *Cluster Analysis: Basic Concepts And Algorithms*. [online] Www-users.cs.umn.edu. Available at: <https://www-users.cs.umn.edu/~kumar001/dmbook/ch7_clustering.pdf>.

[8] Medium. 2020. *How To Build A Content-Based Movie Recommender System With Natural Language Processing*. [online] Available at:

<https://towardsdatascience.com/how-to-build-from-scratch-a-content-based-movie-recommender-with-natural-language-processing-25ad400eb243>.

[9] Malaeb, M., 2020. *Singular Value Decomposition (SVD) In Recommender Systems For Non-Math-Statistics-Programming*…. [online] Medium. Available at: <https://medium.com/@m_n_malaeb/singular-value-decomposition-svd-in-recommender-systems-for-non-math-statistics-programming-4a622de653e9>.