Asian Journal of Electrical Sciences
ISSN: 2249-6297 (P) Vol.9 No.2, 2020, pp.17-24

DOI: https://doi.org/10.51983/ajes-2020.9.2.2552

Restaurant Recommendation System Using User Based Collaborative Filtering

Salu Khadka[1], Pragya Shrestha Chaise[2], Sujin Shrestha[3] and Satya Bahadur Maharjan[4]
Department of Computer Science and Information Technology, Trinity International College, Kathmandu, Nepal
E-mail: satya.maharjan@trinitycollege.edu.np
(Received 23 July 2020; Accepted 6 September 2020; Available online 15 September 2020)

***Abstract* - A recommendation system is an application that can identify entities of interest for a person and provide suggestions based on the past record of person's likes and preferences. The entity of interest can be anything, for example it can be a product, a movie or a news article. Recommender system is an effective way to help users to obtain the personalized and useful information. However, due to complexity and dynamic, the traditional recommender system cannot work well in mobile environment. Keeping such things into consideration, this recommendation system aims to recommend restaurants to users using their past preferences so they do not need to go through a list of choices. The recommender system adopts a user preference model by using the features of user's visited restaurants, and utilizes the location information of user via GPS(Global Positioning System) using LBS(Location Based System) and restaurants to dynamically generate the recommendation results using collaborative filtering technique. The suggestions will be based on the user preferences obtained from the past ratings and reviews given by the user, frequently visited cuisines of the user and the time preference of the user. Moreover, a brief analysis of reviews is also made to provide user a computed synopsis of the restaurant using text mining algorithm.**
Keywords: **recommendation, preference, suggestion, review, restaurant, location, GPS, LBS, text mining**

I. INTRODUCTION

With the explosive growth in the digital information and the number of internet users, a potential challenge has been created which hinders timely access to item of interest on the Internet. Information retrieval systems, such as Google, DevilFinder and Altavista have partially solved this problem but prioritization and personalization of information are absent. Thus, the demand of the recommendation engine has increased very much. Generally, Recommender systems are information filtering systems that deal with the problem of information overload by filtering vital information fragment out of large amount of dynamically generated information per user's preferences, interest, or observed behavior about item [1].

Recommender systems have become extremely common in recent years. They are applied in a variety of applications. The most popular ones are probably movies, books, news, music, research papers, social tags, and products in general. Most recommender systems typically produce a list of recommendations in one of the three ways- through collaborative-filtering technique, content-based technique, and hybrid algorithm [7]. Collaborative filtering system filters information by using the recommendations of similar interest-based people. It is based on the idea that people who agreed in their evaluation of certain items in the past are likely to agree in the future as well. Recommendation of friends in Facebook is an example of this approach. These systems assist users in overcoming the problem of information overload by suggesting or providing recommendations to the users based on their past ratings [6].

Restaurant recommendation system is a mobile application designed to provide user with all restaurant information that are either nearby them or matches their cuisine behavior. The system also has a web version for administration purpose whereby restaurant owner can register their business and exhibit their goodness. The mobile application is built in Android platform, while the web application is developed using Java Spring with Hibernate framework. Moreover, the system uses a web server architecture to communicate with its component and perform its functionality.

Restaurant recommender uses hybrid recommendation system to correctly create correlation among related users and provide a set of restaurants that the user might prefer. Beside a typical recommender, this application bridges between potential users and restaurant owner. On the other hand, the user can also get to know various cuisines that keep evolving in the food industry. For this a food dictionary is introduced in the system.

The other aspect of the system is sentiment analysis in the textual review of the restaurant. While the users might be in dilemma to choose a restaurant, the rating and review can be a judging element. Thus, to ease the process, analysis of the overall review is to make and a summary is provided that can sum up the reviews a restaurant gets. This can also be an important input for the restaurant owner to understand how their place is ruling over the users' mind.

Location Based Services (LBS) are the type of services offered through mobile devices that takes into account the device's geographical location. Since LBS are largely dependent on the mobile user's location, its main objective

is to determine the exact location of the user. It uses real time geographic data to provide information, security and entertainment. Location-based services use a smartphone's GPS technology to track a person's location. Location of smartphone can be easily identified due to special internally equipped chip that support the Global Positioning System (GPS). Using GPS along with Google map can help the user to discover the nearby bookshops, restaurants, etc [3]. The popularity of LBS applications has led to development an application in this field. Most of free software and commonly used applications do not meet the needs of the user in term of interactivity with the user. These applications either be navigation systems, find nearby places or display locations on a map.

II. LITERATURE REVIEW AND METHODOLOGY

A. Literature Review

The Yelp Food Recommendation System has used various principles and techniques to recommend by developing a predictive model of users' review and rating about the restaurant. Using available dataset, the system extracted features of the user preferences and made use of collaborative and content based filtering algorithms. This system implemented clustering method as K-nearest neighbour, weighted bi-partite graph projection, and several other learning algorithms [9].

Similarly, Preference-based Restaurant Recommendation System is a restaurant recommendation system for individuals and group in accordance to their fondness. For a large number of users, a ranking SVM model with features encompassing users' food preferences and dietary restrictions, such as cuisine type, services offered, ambience, noise level, average rating, etc. is built for this system. This system also maximized minimum happiness across a group of users, as an alternative to other group recommendation systems where the most commonly recommended restaurant across individual users is selected [9].
Likewise, Feature Selection Methods for Text Classification is an unsupervised feature selection strategy to generalize textual data into useful information by generating classes for those texts. The system has significantly increased the accuracy of classification problem by combining the features than to without features. The major feature selection strategies used are subspace sampling, uniform sampling, document frequency and information gain [10].

B. Existing System in Nepal

The increasing trend of recommendation has also flourished in Nepal. Various web application and mobile application has implemented user and content based recommendation system. There also exists a number of android application offering restaurant details to their customer. Foodmandu, Yellow App, Real Mountain, Restaurant Guide are few of such. However, these systems are descriptive about the restaurant rather than recommending. The system provides static information that is pre-defined by the restaurant owner. The user can search a range of restaurants of their choice but the system cannot predict their preference. Overcoming this drawback, our system makes intelligent prediction about the user food choice on the basis of their past preference.

C. Collaborative Filtering

In the newer, narrower sense, collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the collaborative filtering approach is that if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue x than to have the opinion on x of a person chosen randomly[1].

The motivation for collaborative filtering comes from the idea that people often get the best recommendations from someone with similar tastes to themselves. Collaborative filtering explores techniques for matching people with similar interests and making recommendations on this basis [2].

Collaborative filtering algorithms often require:

1. Users' active participation,
2. An easy way to represent users' interests to the system, and
3. Algorithms that are able to match people with similar interests.

Typically, the workflow of a collaborative filtering system is:

1. A user expresses his or her preferences by rating items (e.g. books, movies or CDs) of the system. These ratings can be viewed as an approximate representation of the user's interest in the corresponding domain.
2. The system matches this user's ratings against other users' and finds the people with most "similar" tastes.
3. With similar users, the system recommends items that the similar users have rated highly but not yet being rated by this user (presumably the absence of rating is often considered as the unfamiliarity of an item)

A key problem of collaborative filtering is how to combine and weight the preferences of user' neighbors. Sometimes, users can immediately rate the recommended items. As a

result, the system gains an increasingly accurate representation of user preferences over time. Also, a collaborative filtering system does not necessarily succeed in automatically matching content to one's preferences. Unless the platform achieves unusually good diversity and independence of opinions, one point of view will always dominate another in a particular community. As in the personalized recommendation scenario, the introduction of new users or new items can cause the cold start problem, as there will be insufficient data on these new entries for the collaborative filtering to work accurately. In order to make appropriate recommendations for a new user, the system must first learn the user's preferences by analyzing past voting or rating activities. The collaborative filtering system requires a substantial number of users to rate a new item before that item can be recommended.

D. User Based Collaborative Filtering

User–user collaborative filtering, also known as KNN collaborative filtering, was the first of the automated CF methods. It was first introduced in the Group Lens Usenet article recommender [2]. The Ringo music recommender and the Bell Core video recommender also used user-user CF or variants thereof. User–user CF is straightforward algorithmic interpretation of the core premise of collaborative filtering: find other users whose past rating behavior is similar to that of the current user and use their ratings on other items to predict what the current user will like. To predict Mary's preference for an item she has not rated, user–user CF looks for other users who have high agreement with Mary on the items they have both rated. These users' ratings for the item in question are then weighted by their level of agreement with Mary's ratings to predict Mary's preference. Besides the rating matrix R, a user–user CF system requires a similarity function s: U×U → R computing the similarity between two users and a method for using similarities and ratings to generate predictions.

E. Pearson's Correlation Coefficient

Pearson's correlation coefficient is the covariance of the two variables divided by the product of their standard deviations. The form of the definition involves a "product moment", that is, the mean (the first moment about the origin) of the product of the mean-adjusted random variables; hence the modifier product-moment in the name[6].

Pearson's correlation coefficient when applied to a sample is commonly represented by the letter *r* and may be referred to as the sample correlation coefficient or the sample Pearson correlation coefficient [3]. A formula can be obtained for *r* by substituting estimates of the covariance and variances based on a sample into the formula. So if one have one dataset $\{x_1,...,x_n\}$ containing *n* values and another dataset $\{y_1,...,y_n\}$ containing *n* values then that formula for *r* is:

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

Equation 1 Pearson's correlation coefficient

where:

n, x_i, y_i are defined as above (the sample mean); and analogously for $\bar{y}$

$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ Rearranging gives us this formula for r:

$$r = r_{xy} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\,\sqrt{n\sum y_i^2 - (\sum y_i)^2}}.$$

Equation 2: Rearranged Correlation formula

where: $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ are defined as above

This formula suggests a convenient single-pass algorithm for calculating sample correlations, but, depending on the numbers involved, it can sometimes be numerically unstable.

F. Euclidian's Distance

In mathematics, the Euclidean distance or Euclidean metric is the "ordinary" (i.e. straight-line) distance between two points in Euclidean space. With this distance, Euclidean space becomes a metric space. The associated norm is called the Euclidean norm[4].

The Euclidean distance between points **p** and **q** is the length of the line segment connecting them ($\overline{\mathbf{pq}}$)[3].

In Cartesian coordinates, if $\mathbf{p} = (p_1, p_2,..., p_n)$ and $\mathbf{q} = (q_1, q_2,..., q_n)$ are two points in Euclidean *n*-space, then the distance (d) from **p** to **q**, or from **q** to **p** is given by the Pythagorean formula:

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

$$= \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}.$$

Equation 3:Euclidean's distance formula

G. Haversine formula

The haversine formula is an equation important in navigation, giving great-circle distances between two points on a sphere from their longitudes and latitudes. It is a special case of a more general formula in spherical

trigonometry, the law of haversines, relating the sides and angles of spherical "triangles"[5].

For any two points on a sphere, the haversine of the central angle between them is given by

$$hav\left(\frac{d}{r}\right) = hav(x_2 - x_1) + \cos(x_1)\cos(x_2)\, hav(y_2 - y_1)$$

or,

$$hav(\theta) = sin^2\left(\frac{\theta}{2}\right) = \frac{1 - \cos(\theta)}{2}$$

Equation 4 Haversine function

where, hav is the haversine function,

d is the distance between the two points (along a great circle of the sphere; see spherical distance),
r is the radius of the sphere,
x1, x2: latitude of point 1 and latitude of point 2, in radians
y1, y2: longitude of point 1 and longitude of point 2, in radians
On the left side of the equals sign (d/r) is the central angle, assuming angles are measured in radians. To solve for d, one can apply the inverse haversine (if available) or by using the arcsine (inverse sine) function:

$$d = r\, hav^{-1}(h) = 2r \arcsin\left(\sqrt{h}\right)$$

Equation 5 Arcsine(inverse sine) function

H. Text Mining

Text mining refers to finding of certain patterns and understanding from a piece of text. In this research, text mining is concerned with a multilevel classification problem that involves reviews of a restaurant as input and five distinct labels as output. Often, a review describes various dimensions about a business and the experience of user with respect to those dimensions. In this paper, a classifier is build that automatically classifies restaurant business reviews into those dimensions. A few hundred reviews are manually inspected for restaurant businesses and found 5 important dimensions and these include “Food”,“Service”, “Ambience”, “Deals/Discounts”, and “Worthiness”.

I. Location Based Services

Location Based Services (LBS) are the type of services offered through mobile devices that considers the device’s geographical location for computational purpose. Thus, its main objective is to determine the exact location of the user. It uses real time geographic data to provide information, security and entertainment. Location-based services use a smart phones’(Global Positioning System) GPS technology to track a persons’ location. Location of Smartphone can be easily identified due to special internally equipped chip that support the GPS. Using GPS along with Google map can help the user to discover the nearby bookshops, restaurants, etc. [1].

III. DATA COLLECTION

The research uses a scaled down version of Yelp Dataset available from the website of Kaggle .Since, the yelp dataset consists of reviews and business information of businesses other than restaurants, the dataset is further cleaned up by removing irrelevant data.

A.Summary Statistics of Data before Data Clean-up

TABLE I SUMMARY STATISTICS OF DATA

Businesses	11,537
Check-in Sets	8,282
Users	43,873
Reviews	229,907

The reviews related to restaurants are only kept, as our research is focused towards generating restaurant recommendation. Other restaurant information and check-in sets are deleted.

B.Summary Statistics of Data after Data Clean-up

TABLE II SUMMARY STATISTICS OF DATA AFTER CLEAN UP

Restaurants	4503
Users	34789
Reviews	149319

[Source: https://www.kaggle.com/c/yelp-recsys-2013/data]

C. Data Format

All the data (restaurants, users, reviews) are represented as list of dictionaries.
For example,

1. Sample representation of a restaurant:

{'city': 'Glendale Az',
'full_address': '6520 W Happy Valley Rd\nSte 101\nGlendale
Az, AZ 85310',
'latitude': 33.712797,
'longitude': -112.200264,
'new_id': 1,
'rating': 3.5,
'restaurant_id': 'PzOqRohWw7F7YEPBz6AubA',
'restaurant_name': 'Hot Bagels & Deli',
'review_count': 14,
'state': 'AZ'}

2. Sample representation of a review:

{'cool': 2,
'date': '2011-01-26',
'funny': 0,
'rating': 5,
'restaurant_id': 3010,

'review': 'My wife took me here on my birthday for breakfas t and it was excellent. The weather was perfect which made sitting outside overlooking their grounds an absolute pleasu re. Our waitress was excellent and our food arrived quickly on the semi-busy Saturday morning. It looked like the plac e fills up pretty quickly so the earlier you get here the be tter. Do yourself a favor and get their Bloody Mary. It wa s phenomenal and simply the best I\'ve ever had. I\'m prett y sure they only use ingredients from their garden and blend them fresh when you order it. It was amazing. While EVERY THING on the menu looks excellent, I had the white truffle s crumbled eggs vegetable skillet and it was tasty and delicio us. It came with 2 pieces of their griddled bread with was amazing and it absolutely made the meal complete. It was th e best "toast" I\'ve ever had. Anyway, I can\'t wait to go back!',

'review_id': 1,
'useful': 5,
'user_id': 24538}

3. Sample representation of a user:

{'new_id': 0,
'user_id': 'CR2y7yEm4X035ZMzrTtN9Q',
'user_name': "b'Jim'"}

D. Methods

Similarity Calculation

In this research, similarity between the users are calculated by using Pearson's correlation score between two users and finding the correlation between them. The correlation coefficient is a measure of how well two sets of data fit on a straight line. It returns a value between -1 and 1. A value of 1 means that the two people have exactly the same ratings for every item and value of -1 means that the two people have exactly opposite rating for every item.

```
>>> print sim_pearson(critics,'Lisa Rose','Gene Seymour')
0.396059017191
```

Fig. 1 Similarity between two users using Pearson's correlations

Building dictionary of critics

For this research, different users and their preferences are represented in a dictionary called critics in Python by using nested dictionary. This dictionary uses ranking from 1 to 5 as a way to express how much each of these restaurant critics (and the user) liked a given restaurant [11]. Dictionaries are used because they are convenient for experimenting with the algorithm and for illustrative purposes. It's also easy to search and modify the dictionary. Sample critics dictionary is given below:

```
critics={"Scott": {"Sandwich Club": 2.0}, "steven": {"El Molino Mexican Cafe":
3.0, "Oregano's Pizza Bistro": 5.0}, "Lawrence": {"SanTan Brewing Company": 2.0,
"Blu Burger Grille": 4.0}, "Chris": {"Yasu Sushi Bistro": 5.0, "Atlas Bistro":
4.0}, "Aaron": {"Zoes Kitchen": 3.0}, "Mike": {"Joe's Diner": 4.0, "Dick's
Hideaway": 4.0, "Boston Market": 3.0, "D'licious Dishes": 5.0, "Sushi Q": 3.0,
"SaBai Modern Thai": 4.0, "Rico's American Grill": 2.0, "Short Leash Dogs": 5.0,
"Kokopelli Mexican Grill": 3.0, "Elements": 5.0, "FEZ": 4.0, "Moto Sushi": 3.0,
"Chen Wok Express": 2.0, "Phoenix City Grille": 5.0, "Los Taquitos": 5.0, "Souper
Salad": 4.0, "Gyro's House": 3.0, "The Turf": 4.0, "Picazzo's Gourmet Pizza &
Salads": 5.0, "Traffic Jam": 3.0, "Flo's Chinese Restaurant": 3.0, "Golden
Valley": 3.0, "La Condesa Gourmet Taco Shop": 4.0, "Middle Eastern Bakery &
Deli": 5.0, "Veranda Bistro": 2.0, "Jerry's Restaurant": 3.0, "Hula's Modern
Tiki": 4.0, "The Parlor": 5.0, "Gourmet House of Hong Kong": 4.0, "CherryBlossom
Noodle Cafe": 4.0, "America's Taco Shop": 4.0, "Capriottis Sandwich Shop": 3.0,
"Taco Bell": 4.0, "Salvadore\u00f1o Restaurant #3": 3.0, "La Tolteca Mexican
Foods": 4.0}}
```

Fig. 2 Sample dictionary of critics in Python

Recommendation Model

The recommendation model is based on the similarity score which is calculated using Pearson's correlation coefficient. Here, Suppose A, and B be two users and R_i are the restaurants, where i = 1, 2, 3…. n. The value of n equals the total no. of restaurants. Assuming, following data from users:
Given,

TABLE III RATINGS OF USER A

User A	R_1	R_2	R_3	R_4
	3.0	4.0	3.5	4.5

TABLE IV RATINGS OF USER B

User B	R_1	R_2	R_3	R_5
	4.0	5.0	4.5	5.0

Now, at first a dictionary of critics is created, which is a nested dictionary of user's rating. Assuming, there are only two users now,

critics= {'A':{'R_1':3.0, 'R_2':4.0, 'R_3':3.5, 'R_4':4.5}, 'B':{'R_1':4.0, 'R_2':5.0, 'R_3':3.5, 'R_5':4.5}}

Now if the user B wants recommendations, then the recommendation model compares it with every other users' in the critics based on the similarity score calculated using Pearson's correlation coefficient.

Algorithm for generating recommendation for user B,

1. Create a dictionary of critics of every user and their preference.
2. Compare user B with every other user in critics based on the similarity score.

If similarity_score> 0:

i) Score restaurants that user B has not visited but similar user has visited, score[i]= similarity_score * rating of restaurant from similar user

ii) Calculate average scores for scores calculated for each restaurant,

avg_score = sum(score)/ len(score)

iii) Rank the restaurants according to their avg_scores

iv) The recommendations for user B is the restaurants within the top20 rank.

recommendation = rankings[:20]

3. If len(recommendations) == 0:

i)return the top 20 most popular restaurants from the data recommendation = topRestaurants[:20]

4. Return the recommendations for user B

Tokening a review

In the process of text mining, the most preliminary step involves tokenization. Tokenization breaks a sentence to words called tokens. These token, further serves the system as a feature for the process classification. To tokenize a text, TextBlob library in Python is used to correct the sentence and also to lemmatize words to their root. This is useful for the next process of text mining which is a count vectorizer. In the research, only positive stop wordsare included as the negative could be a useful information to predict whether the text contains a positive or negative information about any fields.

```
In [9]: stemming_tokenizer("They serve a good food with great service. Though they did not have any happy hours."

Out[9]: ['serve',
         'good',
         'food',
         'great',
         'service',
         'though',
         'not',
         'happy',
         'hours']
```

Fig.3 Tokenization of Review

Building a classifier

Any user who visits a restaurant expresses their experience through reviews and rating. In this research, a classifier that automatically classifies restaurant business reviews into the dimensions a restaurant could be related be built. About a hundred reviews are manually inspected for restaurant businesses and 5 important dimensions are found which includes "Food", "Service", "Ambience", "Deals/Discounts", and "Worthiness". This research experimented with popular multi-label classification approaches using "unigrams", "bigrams", "trigrams", and "review ratings" as features.

The classifier can be addressed as a learning problem, where the task is to build alearner. The learner can classify a given review into respective categories. However, a review can be associated with multiple categories at the same time, it is not a binary but a multilabel classification problem.

This process of building classifier in our application is carried with the help of a python library Pipeline where three modules: CountVectorizer, TFIDF Transformer and KNeighbors Classifier works concurrently to form a stronger classifier. Later, this classifier is fitted with training data.

Formal Definition

Let H be the hypothesis of multi-label classification and C are the set of categories, X is the review text and Y is output, then:
C = {Food, Service, Ambience, Deals and Worthiness}
H: X -> Y, where Y ⊆ C

Review	**Categories**			
	Food	Service	Ambience	Deals
They have the best happy hours around, the food is good and their service is even better. When its winter, we become regulars.	1	1	N/A	1
Didn't like the food but the place has a great environment and friendly staffs.	0	1	1	N/A

Algorithm for building classifier

1. Create a tokenizer model that generates tokens by correcting sentences, eliminating stopwords and lemmatizing tokens.
2. Create a pipeline to perform a sequence of transformation, using countvectorizer, tfidftransformer and KNeighbor Classifier. The tokens from step 1 is also utilized.
3. Generate training and testing data using cross validation in the ratio of 3:1.
4. Train the classifier from step 2 with the data from step 3.
5. Return the classifier.

Algorithm for Text mining

1. Input restaurant_id
2. Fetch all the reviews from review table with restaurant id = restaurant_id
3. Form food classifier, service classifier, ambience classifier and deal classifier with the algorithm for building classifier.
4. Predict each review with classifier of step 3.
5. Compute percentage of positive predictions.
6. Return result.

Getting Geolocation of user

To find the location of the nearest restaurants, the geolocation is saved i.e. the latitude and longitude of the restaurants. Then, nearest places are classified based on the distance in meters. If the user has selected 100m as maximum distance, then the restaurants located within the 100m is only shown. The geolocation is implemented using Google Map API. This API is used to access and view different location all over the world. The location can be found using the latitude and longitude of that particular restaurant.

Finding Distance to selected Restaurant

This approach implements Euclidean distance formula which takes latitude and longitude as x and y parameter. Here two locations are compared at a time. Generally, the user current location remains constant i.e. (x1, y1) = (USER latitude, USER longitude). The next is the restaurants location i.e. (x2, y2) = (RESTAURANT latitude, RESTAURANT longitude).

Finding Nearby Restaurant

Here the nearby restaurant is calculated with the help of SQL query. The SQL statement will find the closest 20 locations that are within a radius of 25 km(input distance from user) to the user current location (33, -112) coordinate. It calculates the distance based on the latitude/longitude of that row and the target latitude/longitude, and then asks for only rows where the distance value is less than 25, orders the whole query by distance, and limits it to 20 results. To search by miles instead of miles, 6371is replaced with 3959.

```
SELECT restaurant_id, (6371 * acos( cos(
radians(33) ) * cos( radians( latitude ) ) *
cos(radians( longitude ) - radians(-112) ) + sin(
radians(33) ) * sin( radians( latitude) ) ) ) AS
distance FROM restaurant_us HAVING distance <
25 ORDER BY distance LIMIT 0 , 20
```

IV. RESULTS AND DISCUSSION

A. Recommendation and Classification

In this system, Pearson's correlation coefficient is used to calculate the similarity between users and generate recommendation based on the similarity. The recommendation model worked with decent accuracy of 80.71%. The following results are obtained after testing the data against the recommendation model.

TABLE VI PERFORMANCE OF RECOMMENDATION MODEL

	Precision	**Recall**	**f-score**	**Support**
Bad	0.31	0.21	0.25	87
Good	0.86	0.92	0.89	478
Avg/total	0.78	0.81	0.79	565

As, for the restaurant classifier, there are 4 separate classifiers for each label i.e. one for food, service, ambience and deals each. For food classification model, it distinguishes whether the reviewer states the food in the restaurant as good or bad and for service classification model, it distinguishes whether the reviewer states the service in the restaurant as good or bad. Similarly, the distinction for other labels is done accordingly. Each classifier model is tested separately against the respective trained classification model. The following results are obtained: For Food Classification model,

TABLE VII PERFORMANCE OF FOOD CLASSIFIER

	Precision	**Recall**	**f-score**	**Support**
Bad_Food	0.75	0.35	0.48	17
Good_Food	0.69	0.92	0.79	26
Avg / total	0.71	0.70	0.67	43

For Service Classification model,

TABLE VIII PERFORMANCE OF SERVICE CLASSIFIER

	Precision	**Recall**	**f-score**	**Support**
Bad_Service	0.76	0.83	0.79	30
Good_Service	0.50	0.38	0.43	13
Avg / total	0.68	0.70	0.69	43

Fig.4 Percentage of Features mentioned in reviews

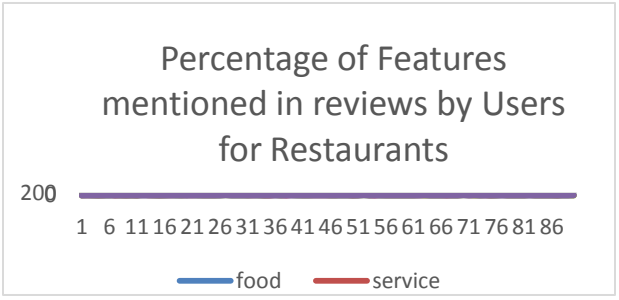

Fig.5 Percentage of Feature mentioned in reviews by user for restaurants

After the classification of reviews of restaurant, it is evident that the primary concern for restaurant customers is the quality of food while service, ambience and deals provided by the restaurant were secondary.

V. CONCLUSION

Recommendation systems help users discover items they might not have found by themselves and promote sales to potential customers, which provide an effective form of targeted marketing by creating a personalized shopping experience for each customer. Lots of companies have such kind of systems, especially for e-commerce companies like Amazon.com, an effective product recommendation system is very essential to their businesses. Historically, we all rely on recommendations given by our friends or relatives but this research focuses on expanding the set of people from whom we users can obtain recommendations. Also, this research aims on effectively narrowing down the choices of the users. Good recommendation creates more user-friendly interfaces. As a result, it also increases and create a number of users for business which can consequently increase the popularity of that restaurant. Significantly, it also helps the application to stand out amongst hundreds of other websites. Similarly, the opinion mining feature can make an application smart enough to extract the information itself and strengthens the system to produce better and efficient outcomes.

VI. FURTHER RECOMMENDATION

For future work, the recommendation system can be upgraded to a more advanced version by adding content based filtering or using algorithms that include neural network. In this research, since collaborative filtering is used the recommendation depends on user data. If user data is no available then the recommendation system is not able to give any recommendation. Also, the effectiveness of the recommendation depends on the diversity of all the user's data. Moreover, the recommendation system does not recommended restaurants that have not been rated by any user. On top of that, for new users that have not rated any restaurants. The recommendation model also faces cold start problem at the beginning of deployment as no ratings are available at first. For Text mining, the model does not handle ambiguous reviews, so the model can be enhanced to handle ambiguity. Moreover, the accuracy of the classification model can be increased in the future.

REFERENCES

[1] A.Dasgupta and P. Drinea, et al. "Feature Selection Methods for TextClassification".[ONLINE]Available:www.stat.berkeley.edu/~mmahoney/pubs/kdd07.pdf[Accessed 21 May 2017]

[2] C. Pan and W. Li. "Research paper recommendation with topic analysis," *In Computer Design and Applications.* IEEE. Vol. 4 (2010), pp. V4-264

[3] C. Zaiontz. "Real Statistics Using Excel", *Real Statistics Using Excel*, 2017. .

[4] E. Gabrielova and C. Lopes, et al. "The Yelp dataset challenge - Multilabel Classification of Yelp reviews into relevant categories", *Ics.uci.edu*, 2017. [Online]. Available: http://www.ics.uci.edu/~vpsaini/. [Accessed 27 April 2017]

[5] E. R. Hedrick, LoCajori and Florian *(1952) [1929]*. A History of Mathematical Notations. *2 (2 (3rd corrected printing of 1929 issue) ed.). Chicago, USA:* Open court publishing company. [Accessed 14 June 2017]

[6] F. Ricci, L. Rokach and B. Shapira. (2011). "Introduction to Recommender Systems Handbook. Springer." [Online]. pp. 1-35. [Accessed 15 may 2017]

[7] H. Jafarkarimi, A.T.H. Sim and R. Saadatdoost. (2012, June). "A Naïve Recommendation Model for Large Databases." *International Journal of Information and Education Technology*.

[8] J. A. Konstan and J. Riedl. "Recommender systems: from algorithms to user experience User Model User-Adapt Interact." Vol.22 (2012), pp. 101–123

[9] R. J. Mooney and L. Roy. "Content-Based book recommendation using learning for text categorization". In *Proc. Fifth ACM conference on digital libraries*, 2010, pp. 195-204

[10] S. Sawant, and G. Pai. "Yelp Food Recommendation System." [ONLINE] Available:http://cs229.stanford.edu/proj2013/SawantPaiYelpFoodRecommendationSystem.pdf[Accessed 15 April 2017]