GREAT LAKES
INSTITUTE OF MANAGEMENT, CHENNAI

Stuart School of Business
ILLINOIS INSTITUTE OF TECHNOLOGY

IMPROVING EFFICACY AND EFFICIENCY OF RESTAURANT RECOMMENDATION MECHANISM

PGP BABI Capstone Project Report

JUNE 17, 2018

Group 1, Section B, Bangalore August 2017 Batch
Group Mentor – Dr. Mallikarjuna Rao, Dr. Narayana Darapaneni

Abstract

The aim of the project is to develop a robust and dynamic recommendation system for restaurant selection mechanism. The project submitted towards partial fulfilment of the criteria for award of PGP BABI by Great Lakes Institute of Management.

It is observed that current recommendation system used within various popular mobile and web applications are static in nature and are not representative of users' preference. The project is focused towards understanding the current methodologies in the available options and analysing the gaps and challenges in the current scenario and mitigating the identified gaps to cater to customers' dynamic palettes.

We created a hybrid Recommendation Model, using R and Python, featuring three Recommendation types (Collaborative Filtering Recommender System, Content-Based Recommender System and Knowledge-Based Recommender System) to ensure the robustness of the recommendation system. User experience or preferences and Restaurant experience over time may vary, Static Recommendation Models fail to capture the degree of change and its impact, thus affecting the efficiency of the model. We captured the trend of the similarity among the Users and similarity among Restaurants over time and incorporated the same in the model, thus ensuring the dynamic nature of Recommendation model.

Keywords: *Recommendation System, Dynamic, Robust, Similarity, Reviews, Restaurant*

Acknowledgement

We would like to gratefully acknowledge the contribution of our mentors, **Dr. Mallikarjuna Rao** and **Dr. Narayana Darapaneni** from Indian Institute of Technology and Great Lakes Institute of Management respectively, who took active part and provided valuable support to us during the course of this project.

We also sincerely thank **Dr. P K Vishwanathan** and **Great Lakes Institute of Management**, for providing valuable feedback and being a source of inspiration in helping us to work on this project.

We certify that the work done by us for conceptualizing and completing this project is original and authentic.

Date: June 17, 2018
Place: Bangalore

Saurabh Arora
Jaya Garg
Chiranjeevi Uppu
Asra Jabeen

Certificate of Completion

This is to certify that the Capstone Project titled "Improving Efficacy and Efficiency of Restaurant Recommendation Mechanism" is a record of the Project work carried out by Saurabh Arora, Jaya Garg, Chiranjeevi Uppu and Asra Jabeen submitted towards partial fulfilment of the criteria for award of PGPBA by Great Lakes Institute of Management. This is their original work to the best of our knowledge.

Dr. Mallikarjuna Rao
Dr. Narayana Darapaneni
(Project Mentors)

Dr. P K Vishwanathan
(Program Director)

Date: June 17, 2018
Place: Bangalore, India

Table of Contents

Table of Figures

List of Tables

Abbreviations

S. No.	Abbreviations	Explanation
1	RMSE	**Root mean square error**. This is the standard deviation of the difference between the real and predicted ratings
2	MSE	**Mean squared error**. This is the mean of the squared difference between the real and predicted ratings. It's the square of RMSE, so it contains the same information.
3	MAE	**Mean absolute error**. This is the mean of the absolute difference between the real and predicted ratings.
4	TP	**True Positives**. These are recommended items that have been purchased.
5	FP	**False Positives**. These are recommended items that haven't been purchased.
6	FN	**False Negatives**. These are not recommended items that have been purchased.
7	TN	**True Negatives.** These are not recommended items that haven't been purchased. A perfect (or overfitted) model would have only TP and TN.
8	TPR	**True Positive Rate**. This is the percentage of purchased items that have been recommended. It's the number of TP divided by the number of purchased items (TP + FN).
9	FPR	**False Positive Rate**. This is the percentage of not purchased items that have been recommended. It's the number of FP divided by the number of not purchased items (FP + TN).
10	ROC	**Receiver Operating Characteristic**. The ROC-curve is a plot of sensitivity or true positive rate TPR which is equivalent to recall by the false positive rate FPR or 1−specificity, with regard to model parameters. A possible way to compare the efficiency of two systems is by comparing the size of the area under the ROC-curve, where a bigger area indicates better performance.
11	SVD	**Singular Value Decomposition SVD**
12	ALS	**Alternating Least Squares method**

Executive Summary

The purpose of the project is to understand consumer's approach or drive, in selecting restaurants via mobile or web applications for dine-in or take away, taking into account the various parameters such as Cuisine / Category, Location, Postal code, Users' ratings, Users' Review.

The initial focused towards understanding the current methodologies, limiting to the recommendation systems only, and identifying the gaps and challenges in the current scenario.

For the purpose of conducting the study, we used the data provided by Yelp for academic purposes. The dataset included Yelp's original data from year 2004 till 2017. For the project we have only included the data since 2013 till 2017.

Once the data was cleaned and usable parameters was identified mapped, we initiated the exploratory data analysis on the data and gather further business insights on the data. It was identified that people seem to be more likely to write a review for a positive experience than a negative one. Also, the relative review count overtime is decreasing.

Next phase was focused on building models, User Ratings were taken as the base for creating models. It was identified that the some of the other models created on Yelp dataset considers review as base for creating model but incorporation of the reviews in the recommendation models is excluded from the current scope of the project. As a continuation to the project post capstone submission, we look forward to incorporate these qualitative inputs into the recommendation model and evaluate its impact.

Hybrid recommendation models for Content were built and analyzed for its performance, in the final phase of the project.

Introduction

User's selection process for trying out a new restaurant contains three major components word of mouth, Restaurant's review and rating provided in Mobile applications and proximity to the restaurant.

The project is focused towards understanding the current methodologies, limiting to the recommendation systems only, and to understand the gaps and challenges in the current scenario. The objective was to build a dynamic recommendation system which mitigates the identified gaps and is robust enough to cater to customers' dynamic palettes.

Some of the gaps, in the current scenario, catered through our model:

- Static Recommendation & Rating system – the system does not take into account the preferences or parameters provided by the customers and the recency of the ratings provided. It is majorly the aggregation of the ratings provided by the user population.
- Unrelated Reviews – the customer reviews available for user's reference are a pool of random reviews provided to the Restaurants, it is not representative of the user's preferences and set filters or parameters. The reviews are fetched from the pool of reviews and not representative of the filters or preferences set by the customer.
- Virtually none of the apps offers its users proper recommendations based on his/her tastes or preferences.

Importance of Recommendation Systems

According to a Search Engine Land survey conducted in 2014, 88% of consumers read online reviews to determine the quality of a local business and 72% consumers said that positive reviews made them trust a local business more.

What is a local business? It can be a medical practice, a restaurant, or a gym. The result of this survey showed the importance of reviews in deciding to choose a business. A Search Engine Land survey in 2012 showed a positive shift in consumer trust and appreciation of online reviews.

Among the many websites which host reviews for local businesses, Yelp is the most popular choice and the most influential local review site. This claim is based on a survey conducted by Yelp and Nielson in 2014. Before the creation of Yelp, people went to a restaurant usually by hearing from friends, or occasionally seeing an interesting place. Now, since websites like Yelp have been built, people can check information before directly visiting a restaurant. The available information can be reviews, ratings, price ranges and hours of operation. After evaluating prices, times and reviews, people make decisions about whether to go or not.

Study claims that among all the various types of business available on Yelp, restaurant is the most searched category by users. However, oftentimes users just search a restaurant by using word "restaurant", while the word "restaurant" means differently to different individuals. For an Asian, it can mean a "Chinese restaurant" or "Thai restaurant". How to correctly interpret search requests based on people's preference is a challenge. Building a machine-learning model based on activity history of a registered user can solve this problem.

Project Overview

Problem Statement

Recommendation mechanism in the mobile applications, used in Restaurant recommending and rating application such as Zomato, Google Review and Food ordering mobile applications such as Swiggy, Foodpanda, forms an integral and important part of the user experience and service.

The recommender system used in the mobile applications are static in nature, they do not take into consideration the restaurant's current rating and mostly is a cumulative sum of inception (restaurant start date) to date ratings and reviews. Also, the recommendations are majorly based on the ratings and reviews and is not representative of the user's preference.

Project Scope

The scope of the project includes, Development of the Dynamic Rating & Recommendation System, based on customers' reviews and ratings for customer specific food preference, recency of the inputs and other parameters.

The objective of the project is to build a robust Dynamic Recommendation Model. There are two words to be focused here, '**Robust**' and '**Dynamic**'.

There are four types of recommendation systems – Collaborative Filtering Recommender System (User Based), Content-Based Recommender System (Restaurant Based), Knowledge-Based Recommender System (Based on User's Purchase History) and Constraint-Based Recommender System (Used in cases where user information is limited). We created a hybrid Recommendation Model featuring three Recommendation types, this will ensure the **Robustness of the model**.

User experience or preferences and Restaurant experience over time may vary, Static Recommendation Models fails to capture the degree of change and its impact, thus affecting the efficiency of the model. We captured the trend of the similarity among and Users and similarity among Restaurants over time and incorporating the same in the model, thus ensuring the **dynamic nature of Recommendation model**.

Project Objective

The purpose of the project is to understand consumer's approach or drive, in selecting restaurants via mobile or web applications for dine-in or take away, taking into account the various parameters such as Cuisine / Category, Location, Postal code, User ratings, Friend's Review.

The project is focused towards understanding the current methodologies, limiting to the recommendation systems only, and the gaps and challenges in the current scenario. We built a

dynamic recommendation system which mitigates the identified gaps and is robust enough to cater to customers' dynamic palettes.

Key Deliverables

- Business Insights from the Data available from various sources
- Dynamic & Robust Recommendation System

Approach

The analytical approach for the project was based on the following methodology:

- **Data Gathering Phase** – The first phase of the project was focused on identification & assessment of the various parameters affecting users' selection criteria. Based on the parameters, data collection was carried out from available sources. Secondary Research and assessment of the available techniques was carried out on the project was also included in this phase.
- **Data Cleaning & Preparation Phase** – It was observed that parameters captured from yelp focus on different parameters, crucial for building the dynamic nature of the algorithm, in different datasets which were mapped together to create two datasets - Master Database – Business and Master Database - User. Data Cleaning was also carried out in this Phase.
- **Exploratory Data Analysis** – Once the cleaned data was available, EDA Phase was initiated. Obtaining Business & Data Insights was a key deliverable of this phase. Other deliverables included understanding the Customers' Preferences, Relation and Correlation of the Variables and identifying the key Drivers.
- **Model Building Phase** – The next phase of the Project was focused on building the model.
- **Testing & Evalution Phase** – Once the model was developed the testing of the Model was carried out.

Limitations

- **Constraints while performing Data Sanctity Check**: As the dataset was downloaded from Yelp, treating outliers and data anomalies was a limitation. Unlike in case of business or company data, data sanctity check is possible with the operations team, was not possible. We had to either remove the outliers or incorporate the anomalies in the model.
- **Yelping Patten**: People use Yelp to find restaurants, but only a relatively small fraction of them leave a rating, and even less of them write a comment to explain the rating. The performance of a good recommendation system depends on a good user-based data.
- Unavailability of Sales Data: The reviews and restaurant ratings are not reflective of Sales information. Though it is true that the restaurant rated by the user is the one for which he/she has rendered services, its vice versa is not true.
- Limited Domain knowledge
- **Huge Data Size**: Handling huge dataset was one of the biggest limitation. Some of the matrix formed for computational purposes exceeded 80 GB memory size. R, because of the data limitations was unable to compute this level of dataset.
- **Limited Tools knowledge**: The dataset was available in SQL and JSON format, team had limited knowledge on SQL and Python. Python would have been a better choice for creating

model, but due to limited knowledge, primary reliance was on R, only Collaborative based model was written in Python.

Literature Review

With the vast amount of information available on the products and businesses to users nowadays, both online as well as offline, there is an increasing interest in developing recommender system that can provide users with recommendations based on their preferences and personalities. Usually the mobile applications, recommending restaurants, works by predicting numeric ratings users provides to restaurants or services, and in generally the they belong to one of two types: Content based or collaborative filtering.

Recommendation systems evolved as an independent research area in the mid-1970s in Duke University [1] and until today, researchers have developed many recommendation systems for almost every domain like entertainment, social-networking sites, content-based sites (e-learning, books or articles recommendation, e-filtering etc.), e-commerce, tourism, match-making and a lot more, all dealing with real-world. [2]

Whether consumers want to select a restaurant to dine-in or takeaway, increasingly the tool of choice is the mobile phone. The attempt of this project is not only to find what factors that contribute or stimulate customer's satisfaction but also to link these factors with customer intent and to find out which of these variables plays the largest role in solving the problem statement. Some of the factors considered are dining duration, table characteristics, food quality, the fairness of wait, pin code, user & restaurant ratings, cuisine and consumer reviews etc.

Machine learning algorithms would be used for this purpose. Selecting an appropriate algorithm to decide which properties of the application to focus on, is crucial. Indeed, recommendation systems have a variety of properties that may affect user experience, such as accuracy, robustness, scalability, and so forth. [4]

Above attributes are considered as a base for the recommendation systems. To understand both positive and negative reviews, text Mining would be leveraged. As mentioned, the objective of the project is to recommend customers with restaurants that they would like using following algorithms;

Various Approaches in Recommendation Systems

- **Collaborative Filtering Algorithm**
  As per Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho, Traditional recommendation systems use collaborative filtering to predict the rating of a product to a particular user. The general idea behind collaborative filtering is that similar users vote similarly on similar items. [5]. There are two major flavors of collaborative filtering. They are:

  Memory-based methods:
  Memory-based methods also referred as neighborhood based collaborative filtering algorithms. These were among the earliest collaborative filtering algorithms, in which the

ratings of user-item combinations are predicted on the basis of their neighborhoods. These neighborhoods can be defined in one of two ways: User-based Collaborative Filtering and Item-Based Collaborative Filtering. [6]

Model-based methods:
In model-based methods, machine learning and data mining methods are used in the context of predictive models. In cases where the model is parameterized, the parameters of this model are learned within the context of an optimization framework. Some examples of such model-based methods include decision trees, rule-based models, Bayesian methods and latent factor models. [6]

- **Content-based Filtering**
  Content-based systems are based on the concept "Show me more of what I have liked" [7]. The basic idea behind such systems is to recommend items or products to a particular user, which are similar to the ones that user has already liked in the past. The similarity between two or more items can be calculated based on their similar features. To understand better, continuing with the same example discussed above, whenever one watches a video on Facebook like the ones posted on Pet lover's page, after the viewer has finished watching it, he/she gets links of similar videos on the home page. This is Content-based filtering i.e. if the viewer/user like an item from some particular category, it is likely that he/she might like any other item similar to it (from the same category or some category similar to it). [8]

- **Knowledge-Based Systems:**
  Knowledge-based systems are based on the concept "Tell me what fits my needs". [7]. Such systems make recommendations to the user based on a specific domain knowledge i.e. the system gets the user requirements, matches it with its knowledge base about that particular domain and recommends those items which according to it are the most appropriate and useful for the user, besides keeping user preferences in consideration. Consider the online shopping sites, when user wants to buy something, say a laptop, one will be asked to provide the requirements, once the requirements are filled in the system recommends the most suitable product, which does not only meet the requirements but is also the options that the system considers, would be the most suitable ones. [8]

- **Hybrid Recommendation Systems:**
  Hybrid systems as the name suggests are the combinational systems, the idea behind which are to combine the features of two systems (recommendation approaches) in such a way that the shortcomings of one are overcome by the other or it gives the best of both worlds. Considering the example of Netflix, it is a hybrid of collaborative filtering and content-based filtering i.e. it recommends movies to the user on the basis of both his / her liking and the similarity with other the liking of other users with similar preferences. For instance, a user likes The Notebook, fated to Love you, PS I Love you etc. then the next time he visits the site, he would be recommended movies from the Romantic genre (content-based). Also, if a user x and a user y have plenty of movies they have liked in common, then each of them would be recommended the next movie either of them likes (collaborative-filtering). [8]

- **Demographic-based Recommendation Systems:**

Demographic systems as the name say clearly are based on the demography of the user or on the region the user belongs to. The basic idea is that the recommendations made are based on the demographic region of the user. For example, consider eBay (online shopping site), the user has to select the region (country in this case like eBay. in, eBay.uk etc.) he or she belongs to for better use. As a result, only the products available in that particular region selected would be recommended to the user and the cost too would be given as per the relevant currency. And hence only relevant recommendations would be made. [8]

- **Community-based Recommendation Systems:**
  Community-based systems follow, "Tell me who your friends are, and I will tell you who you are". [9]. Such systems make recommendations to the user based on the preferences of the friends of the user i.e. unlike the case with collaborative filtering where recommendations are based upon the similarity of users (random or friends), here the recommendations are purely based on the similarity with the user's friends. For instance, Facebook, it is quite likely for the user to add someone in their profile, suggested by their friends, whereas the chances of the user adding the ones shown in the people that the user knows are comparatively less. Community-based systems are all about, which clearly is based on the fact that users rely more on the recommendations taking their friends into consideration rather than the ones where random individuals are considered. [8]

Using available techniques, we built a hybrid model on the basis of the ratings provided by the users. We improved our model by incorporating the trend of similarity factor into the model. This equips model to predict and recommend best options based on User's current preferences and current ratings of the restaurant, which as observed from the data, can change overtime.

Data Gathering Phase

Data Source

For the project, we referred to Yelp Dataset. The dataset was available in the SQL and JSON format. For the project we referred to the SQL Dataset. The dataset is 8.1 gigabytes (uncompressed) in size.

Yelp Data for New York Restaurants – https://www.yelp.com/dataset/download

The dataset is provided by Yelp which is a subset of businesses, reviews, and user data for use in personal, educational, and academic purposes. It includes 5,200,000 reviews, 174,000 businesses, 200,000 pictures and 11 metropolitan areas. It comprises of 1,100,000 tips by 1,300,000 users, Over 1.2 million business attributes like hours, parking, availability, and ambience, Aggregated check-ins over time for each of the 174,000 businesses.

Structure of Data set: The dataset consists of 7 files such namely – Attribute, Business, Category, Friend, Review, Tip and User (refer Figure 1 for Dataset schema) and each file consists of specific information.

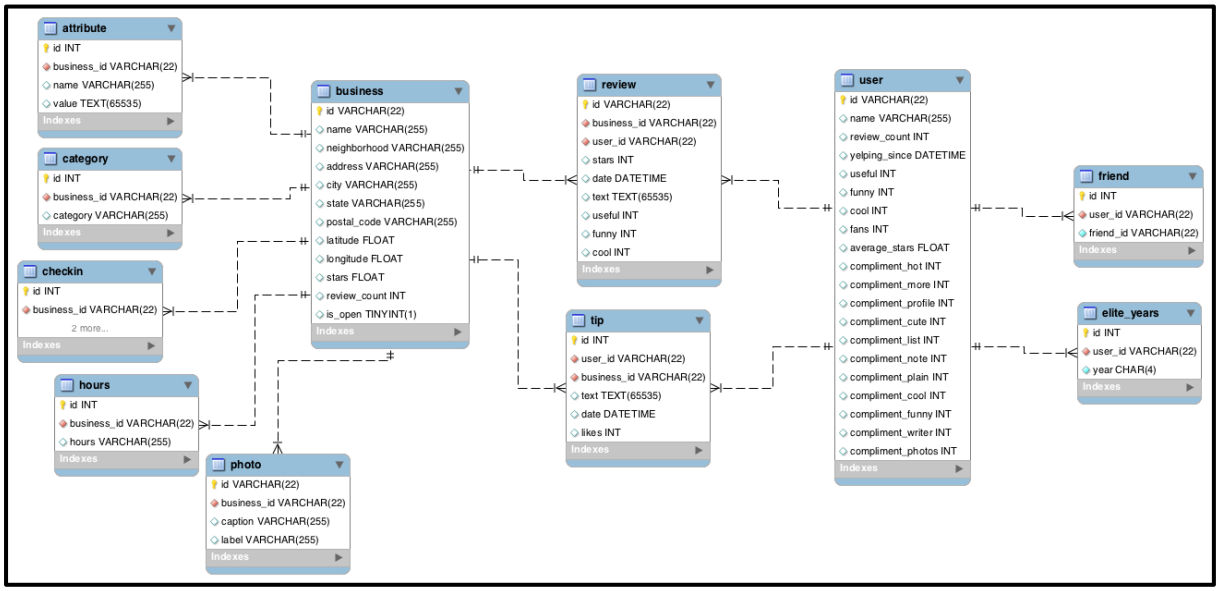

Figure 1: Yelp Dataset - Schema

Data Cleaning & Preparation Phase

The scope of the project involves building a recommendation model, following are the datasets that we have included in the study.

Table 1: Yelp Dataset - Size

S. No.	File	File Size	Row Count	Column Count
1	Attribute	111.6 MB	1,303,812	4
2	Business	23.6 MB	173,424	12
3	Category	41.6 MB	666,484	3
4	Friend	3.3 GB	46,778,288	3
5	Review	3.9 GB	5,047,073	9
6	Tip	160.7 MB	1,069,497	6
7	User	169.9 MB	1,280,597	20

Following is the list of attributes for each file.

Table 2: Final Dataset - Attributes

Attribute	Business	Category	Friend	Review	Tip	User
Business Id	Business Id	Business Id	User Id	Review Id	User Id	User Id
Name	Name	Category	Friend Id	Business Id	Business Id	Name
Value	Neighbourhood			User Id	Text	Review Count
	Address			Stars	Date	Yelping Since
	City			Date	Likes	Useful
	State			Text		Funny
	Postal Code			Useful		Cool
	Latitude			Funny		Fans
	Longitude			Cool		Average Stars
	Stars					Compliment Hot
	Review Count					Compliment More
	Is Open					Compliment Profile
						Compliment Cute
						Compliment List
						Compliment Note
						Compliment Plain
						Compliment Cool
						Compliment Funny
						Compliment Writer
						Compliment Photos

Step 1: Mapping Datasets together

- The dataset was further mapped together into three master files, namely 'Master Database – Business', 'Master Database – User' and 'Master Dataset – Review'. The business part of the file covers the restaurant information and User database comprises of the User information, their reviews and their ratings. Following are the Columns captured in each dataset:
- **Master Dataset – Business**: Business Id, Restaurant Category, Restaurant Name, Restaurant Neighborhood, Restaurant Address, Restaurant City, Restaurant State, Restaurant Postal Code, Restaurant Latitude, Restaurant Longitude, Restaurant Stars, Restaurant Review Count, Is Open
- **Master Dataset – User**: User Id, User Name, User's Review Count, Yelping Since, Useful, Funny, Cool, Fans, Average Stars, Compliment Hot, Compliment More, Compliment Profile, Compliment Cute, Compliment List, Compliment Note, Compliment Plain, Compliment Cool, Compliment Funny, Compliment Writer, Compliment Photos
- **Master Dataset – Review**: Review Id, Business Id, User Id, Stars, Date, Text, Useful, Funny, Cool.

Step 2: Treating Missing Values

It was observed that some of the ids had a syntax error and appeared as '#Name!' or '#Ref' when read in Excel and '#NA' when read in tableau & R. These business and User Ids were corrected in the Master Database using Excel.

Also, the Ids for which the relevant information was not available, were excluded from the master dataset.

Step 3: Setting up Master files

- **Master Dataset – Business**:
  - A total of 174,567 unique business IDs of all the businesses and shops available on Yelp.
  - Some of these businesses are not under the purview of this project, as they are not categorized as restaurants. Thus, these business ids were excluded from the scope of this project. So, 72,705 unique business ids of Restaurants were included in the master dataset.
  - In one of the columns, 'Is Open' in Master dataset, some of the Business Ids are as categorized as 0, depicting that these business Ids are not operational anymore, these were further filtered out, leaving 55,010 unique Business Ids.
  - Next, we filtered out Business Ids which were not based out of USA.
  - Finally, a total of **46,973 unique Business Ids** were considered in building the recommendation model.
- **Master Dataset – User**:
  - A total of 1,226,101 unique user IDs of the users of all the businesses on Yelp is available. Since some of Business Ids are not under the purview of the project, they were excluded from the master dataset. **846,067 are the remaining unique User IDs** considered in the Master dataset.
- **Master Dataset – Review:**
  - Raw dataset contains 5,047,073 reviews, this includes all the reviews for all the Business Ids and from all the User IDs from July 22, 2004 till December 11, 2017. The Reviews which were not related to unique Business IDs included in the final dataset were removed. Further, only the reviews for last 5 years starting from January 1, 2013 till December 11, 2017, were taken into consideration. This resulted in **2,426,487 reviews**.

Step 4: Setting up Assumptions

Following are some of the assumptions considered for building model:

- Though it is true that the restaurant rated by the user is the one for which he/she has rendered services, its vice versa is not true. So, for the sake of the model building we have assumed it's vice versa to be true.
- Some of the locations where there was an ambiguity and detailed information was not available we have considered as part of USA. One such example is Montreal, which is a city in Wisconsin, USA as well as in Canada, we have considered it as a part of USA, in cases where other information such as state was not mentioned.

Exploratory Data Analysis

Exploratory Data Analysis (EDA) was conducted using Tableau and R. The EDA was carried out to understand the Restaurant business and gather business insights.

Top Brands

- **Top 10 Restaurant Brands (in terms of number of unique Business Ids)**

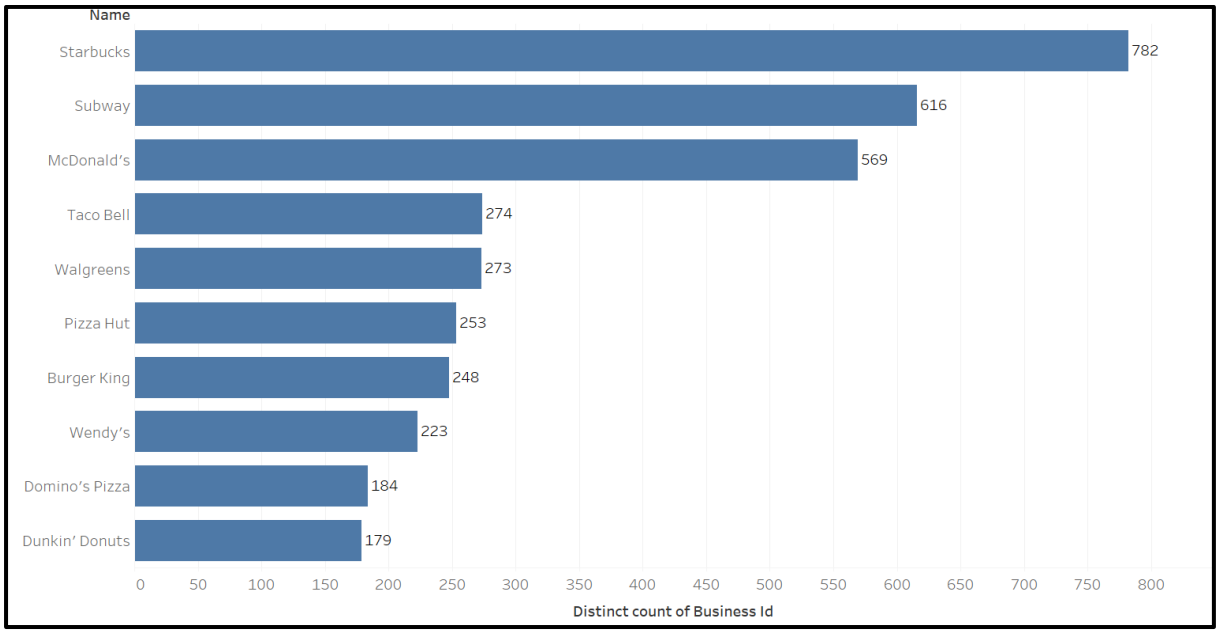

Figure 2: Top 10 Restaurant Brands

- Starbucks had the majority no. of outlets, based on the data. Followed by Subway and Mc Donald's.
- Taco Bell which has 48% of the outlets compared to the Mc Donald's is the fourth largest brand (in terms of Business Ids). Followed by Walgreens, Pizza Hut and Burger King.

- **Top reviewed Restaurant Brands (Top 50)**

Figure 3: Top reviewed Restaurant Brands (Top 50)

  - With over fifty thousand reviews **Yard House** is the top reviewed restaurant.
  - Followed by **Starbucks** with over thirty-eight thousand reviews and **Bacchanal Buffet** with over thirty-five thousand reviews.
  - **Bachi Burger** and **Earl of Sandwich** also register over thirty thousand reviews.

- **Top Restaurant Brands (Reviews V/s Average Stars)**

  - **Bacchanal Buffet** and **Mon Ami Gabi** are the highest rated and reviewed restaurants.
  - **Earl of the Sandwich** and **Bachi Burger** have an average rating of 3.8.
  - Yard House has an average rating of 3.78, reaffirming its popularity among customers.

Business Insight: It is to be noted that, except for Starbucks and Mc Donald's, none of the top ten brands (in terms of no. of unique business Ids) have been registered in top 10 top reviewed restaurants. Some of them are not even on the top 50 list.

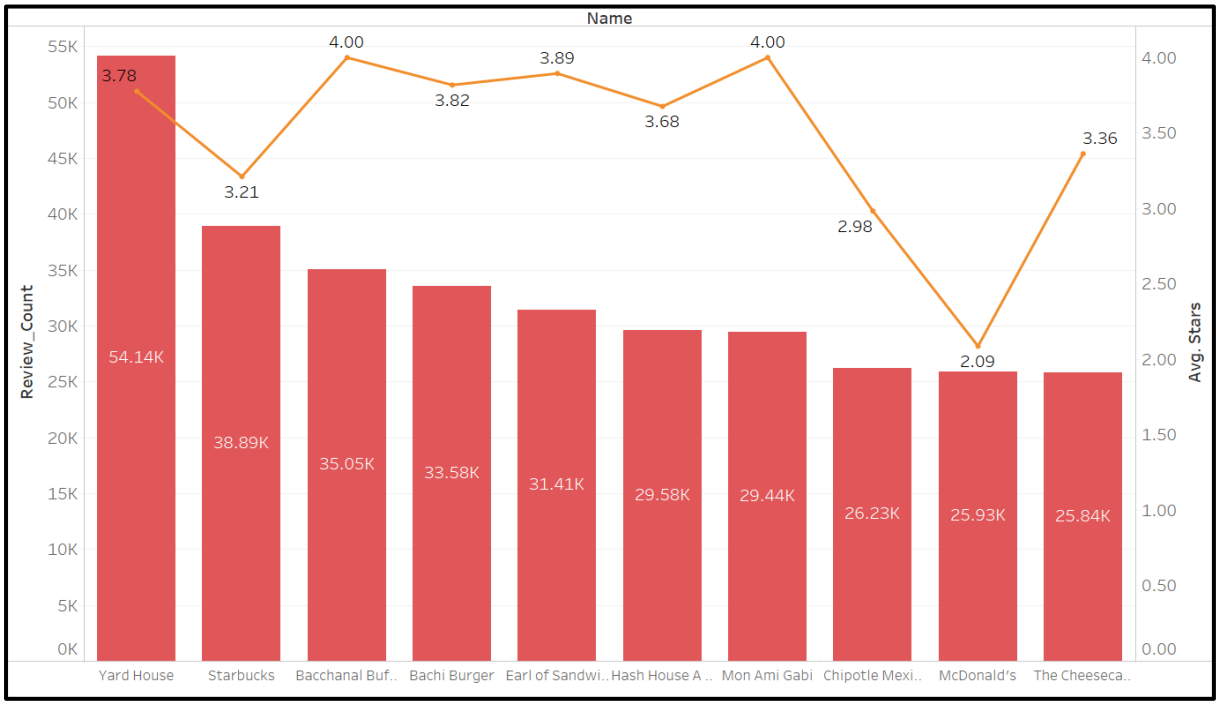

Figure 4: Top Restaurant Brands (Reviews V/s Average Stars)

Top Categories

- **Top 10 Restaurant Categories (in terms of Business Ids and Review counts)**

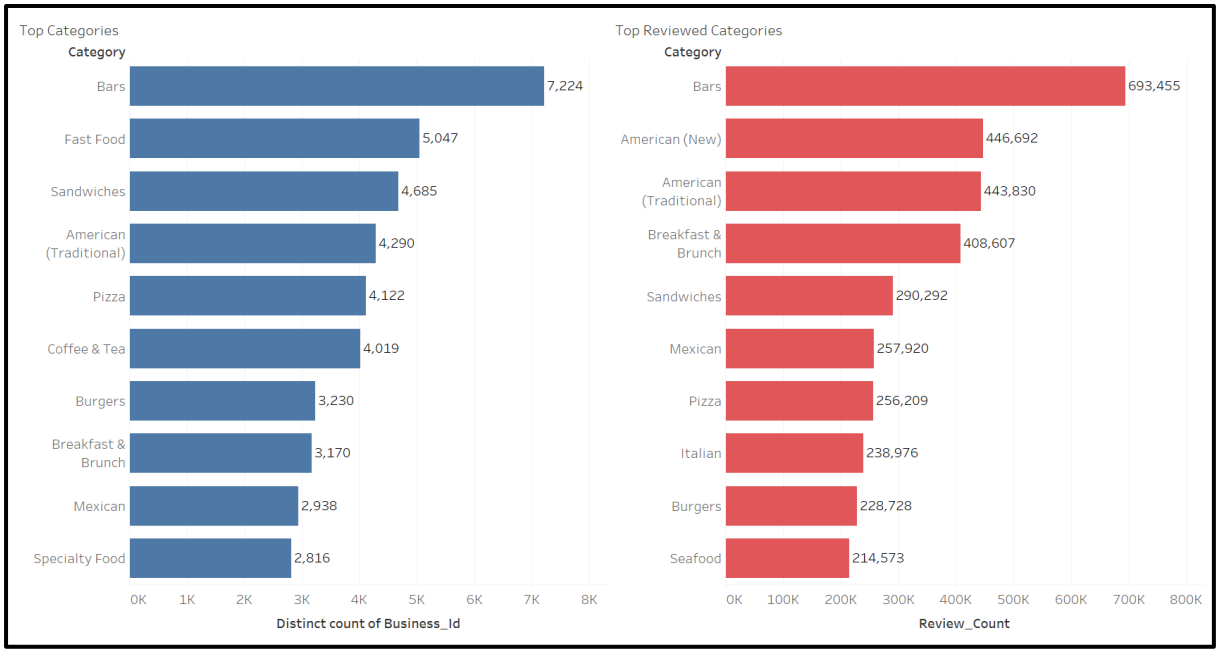

Figure 5: Top 10 Restaurant Categories

- Most of the unique Business Ids are categorized as bars and bars is also the most reviewed category. So, it is safe to assume that, Bar, is the most preferred category among Restaurant Owners as well as customers.
- American (Traditional) Restaurants also are categorized as one of the most preferred categories among customers and Business Owners alike.
- Fast Food is one of the top categories, in terms of no. of unique business ids, stating that it is one of the top food categories among restaurant owners. But, it is not a top food category among customers, in terms of reviews.
- Sandwiches, Pizza, American, Breakfast and Brunch, Italian and Mexican are other top categories which are equally popular among restaurant owners and customers.

- **Top 10 Restaurant Categories (Reviews V/s Average Stars)**
  - Average Star Rating for Seafood is the highest.
  - Breakfast and Brunch, Bars, American (New) and Italian are among the top 5 highest rated category.

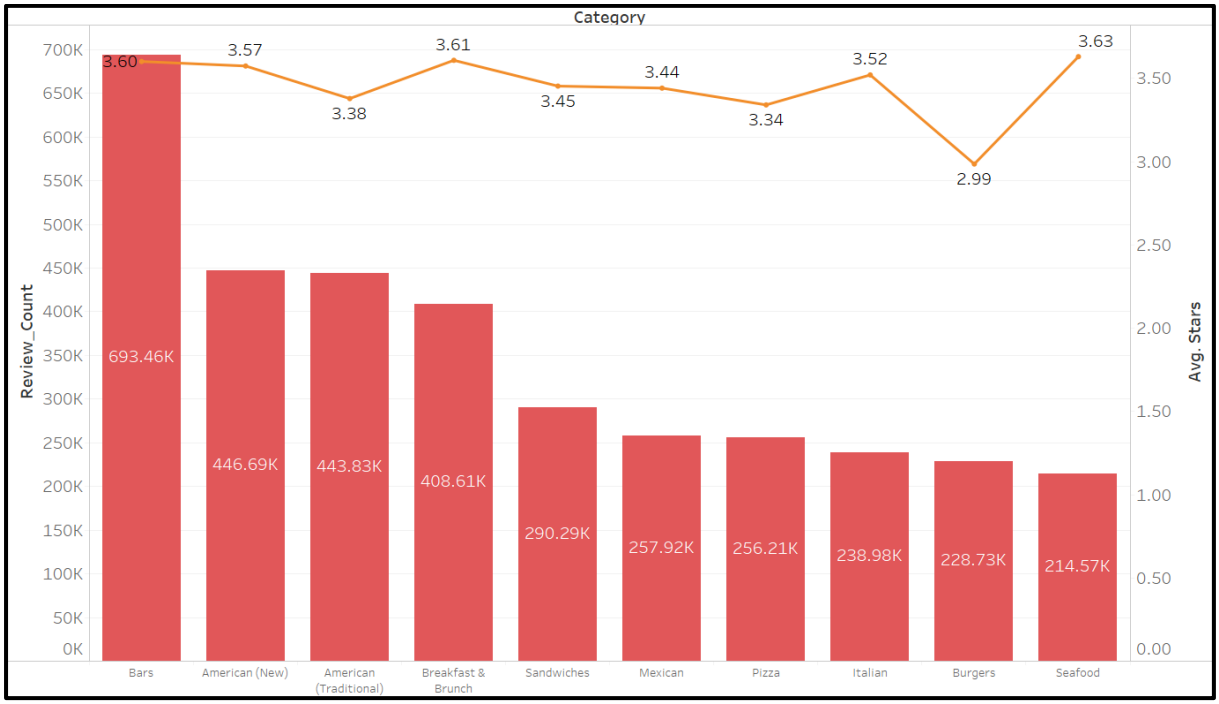

Figure 6: Top Categories (Reviews V/s Average Stars)

Business Insight: Fast Food is a favorite food category but not the category that customers rate quite often. There are lot of Coffee houses but reviewers don't review them quite often.

Top Neighborhood

- **Top Neighborhood (in terms of number of Restaurants and Reviews)**

- Ville-Marie has the maximum no. of Restaurants but is not very popular from Consumers' perspective, as in terms of number of reviews it falls at the eighth spot. The review count for the business ids in Ville-Marie is comparatively quite low.
- The Strip is one of the most preferred Restaurant spots, as it has most number of restaurants and the most reviewed restaurants. This shows that the restaurants in this area are preferred a lot by its users.
- The difference between the total reviews received by all Business Ids based out of 'The Strip' v/s that of 'Downtown', which is the second most reviewed Neighborhood, is more than 70%.
- Westside, Southeast and Spring Valley are among the other top 5 reviewed neighborhood.
- Westside and Spring Valley which is at the eighth and tenth spot respectively, in terms of number of restaurants, are among the top five most reviewed neighborhood, depicting users' high frequency to the restaurants based out of these neighborhoods.

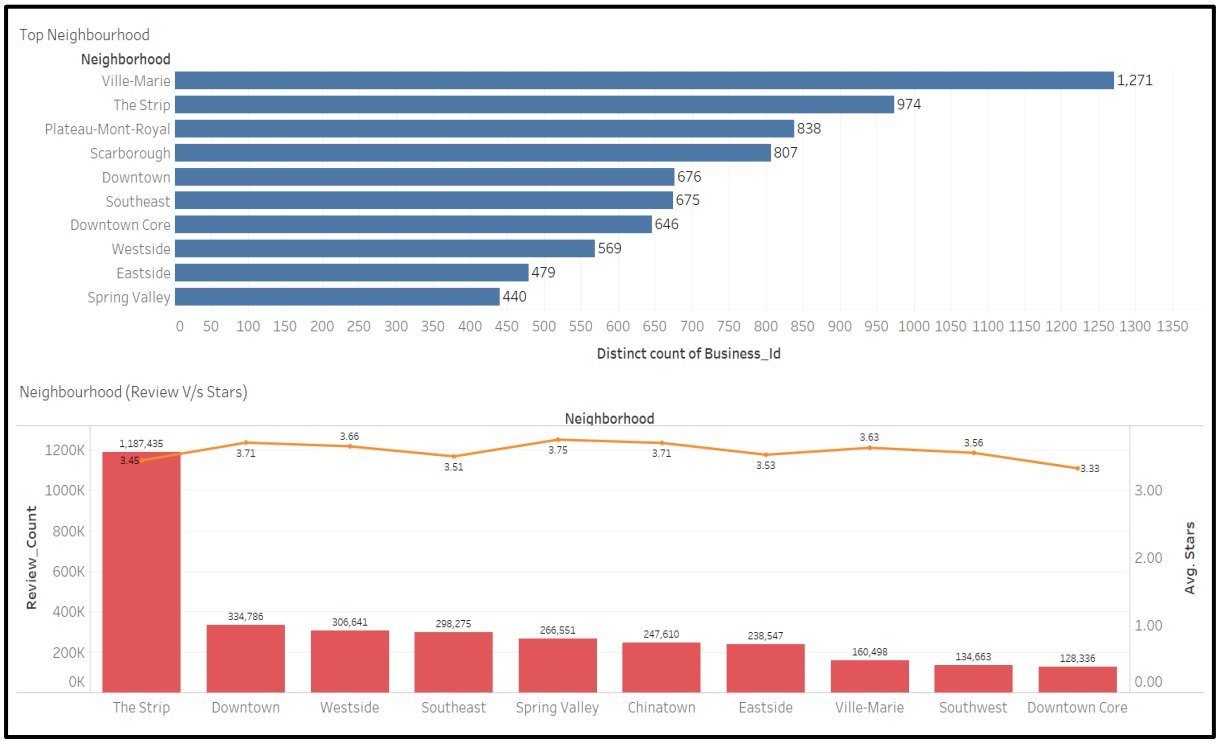

Figure 7: Top Neighbourhood

To have a deeper understanding of the User preference, let us now look at the average ratings provided to these neighborhoods.

- On an average, the restaurant ratings at The Strip is 3.45 Stars.
- Business Ids based out of Spring Valley, Downtown and Chinatown are reviewed better, with an average rating of 3.7 Stars.
- Followed by, Westside and Ville-Marie with an average rating of 3.6 Stars.

Business Insight: In case a User Id is new or visiting the city and he/she searches for best dining options in his mobile application, the mobile application should suggest top restaurants, as per his/her taste, based out of The Strip. Similar recommendations be made in the case of

Spring Valley, Downtown, Chinatown, Westside and Ville-Marie based on User's proximity and preference. Ville-Marie and Strip can be considered as one of the top restaurant spots, with lot of good options for the users.

Top City

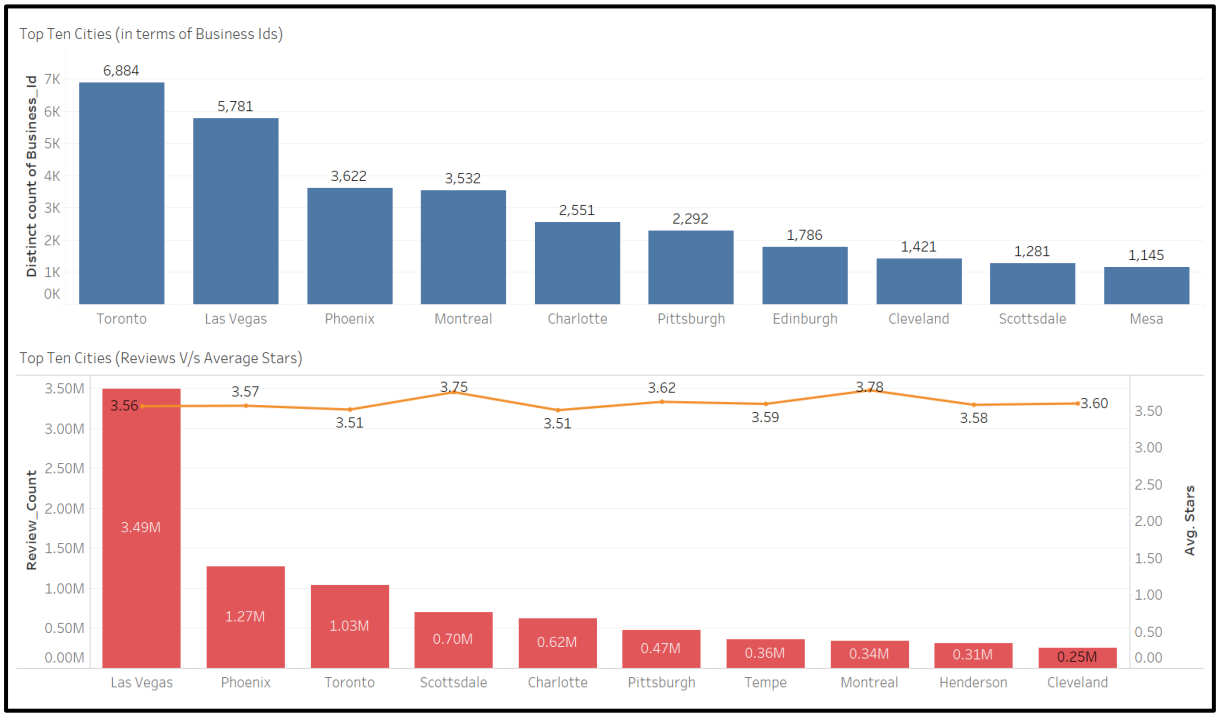

Figure 8: Top 10 Cities

- Las Vegas houses the more than 5.7 thousand Restaurants and also is the top reviewed restaurant, with average Review Rating of 3.56 Stars. The data reaffirms the fact that Las Vegas is one of the top tourist destination.
- Toronto, which houses the highest number of restaurants, falls to third spot, in terms of reviews.
- Next in the list is Phoenix, with over 3.6 thousand restaurants, the city manages to secure second spot in terms of reviews, with 3.5 7Average Star Rating. The number of reviews is 36% of the reviews received by Las Vegas.
- Montreal and Scottsdale are the top-rated Cities with average Star Rating of above 3.7.

Top States

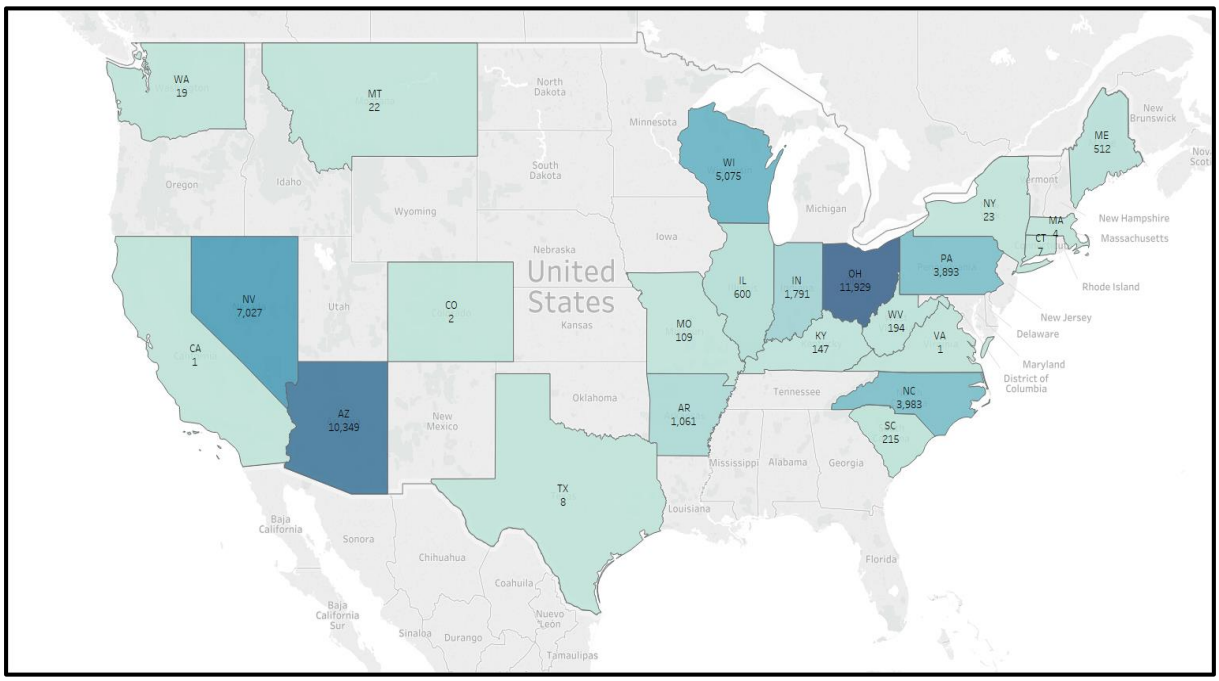

Figure 9: Top Cities (in terms of Business Ids)

- Ohio, Arizona and Nevada are among the top three states, in terms of number of Restaurants.

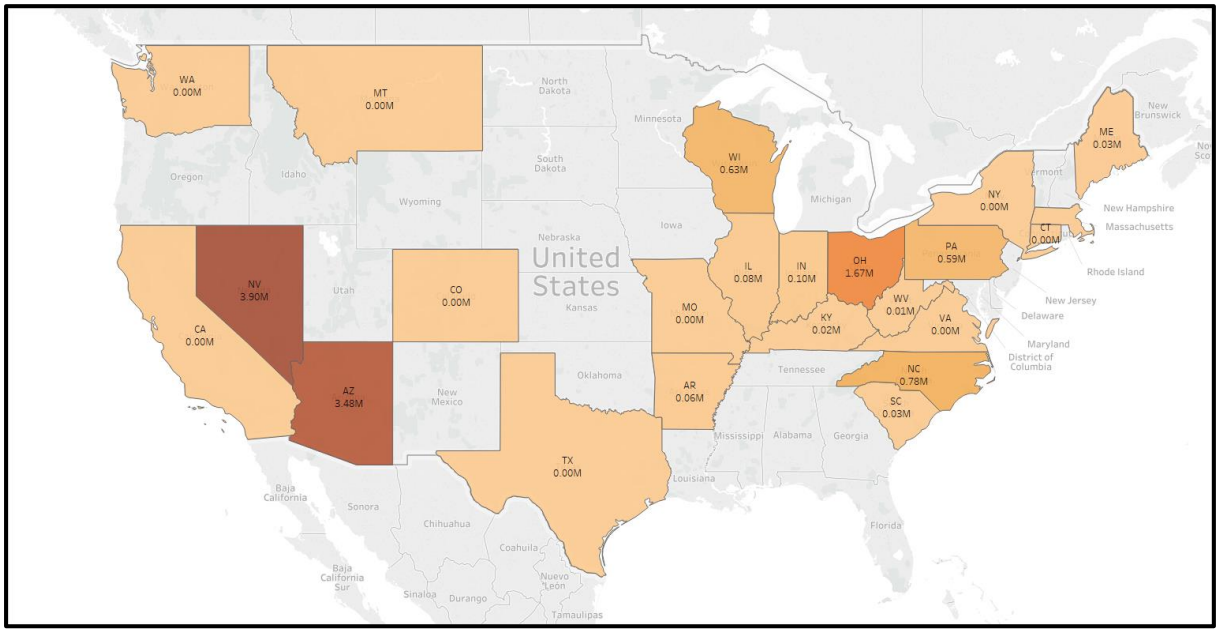

Figure 10: Top Cities (in terms of Reviews)

- Nevada is the top state, in terms of review, followed by Arizona.
- Ohio, which houses highest number of Restaurants has comparatively lesser reviews, less than 50% of the total reviews.

Review trend

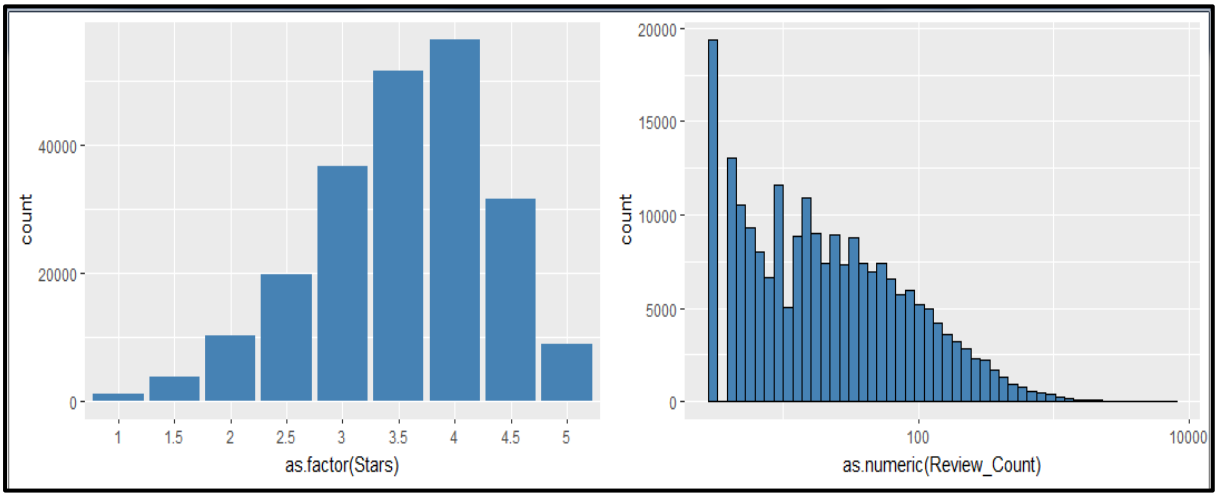

Figure 11: Review trend

Distribution of rating is left skewed with the most frequent rating being 4 and 3.5, whose frequency is higher than other ratings.

It is possible that people who rate and review are the ones who will view review/ratings online before deciding to try a new restaurant. So, there are more chances that these people like what they chose. Also, in general, people seem to be more likely to write a review for a positive experience than a negative one. This is evident from Figure 12, as maximum no. of reviews is within 3.5 to 4.5 range of rating. Also, most business ids are reviewed between 3 to 4.5, refer figure 13

This tells us that recommendation system is playing a vital role for people to decide on a restaurant and an improved personalized recommendation system could further increase the popularity of a restaurant.

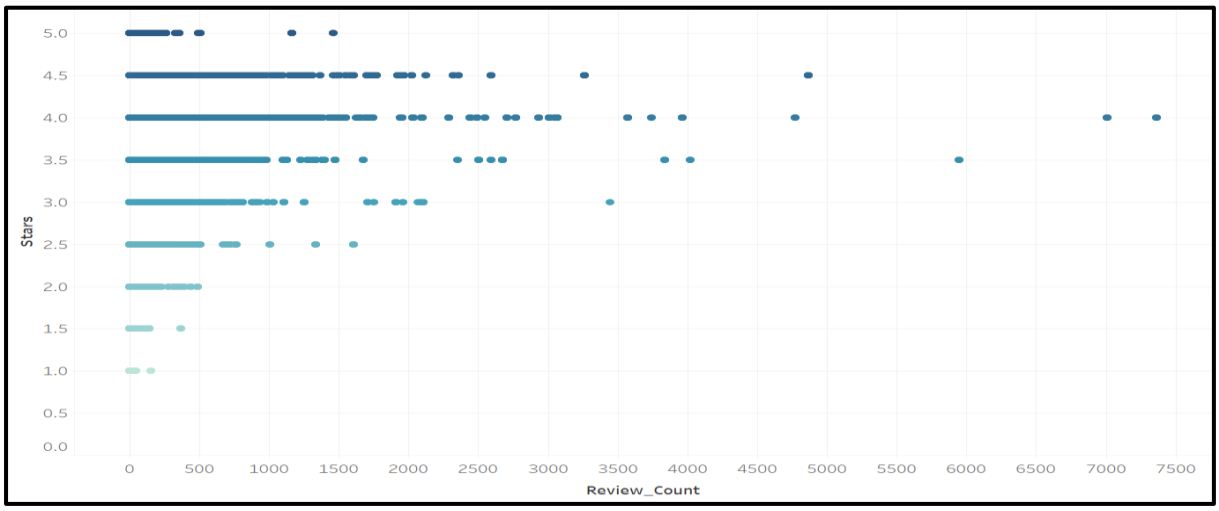

Figure 12: Review V/s Rating

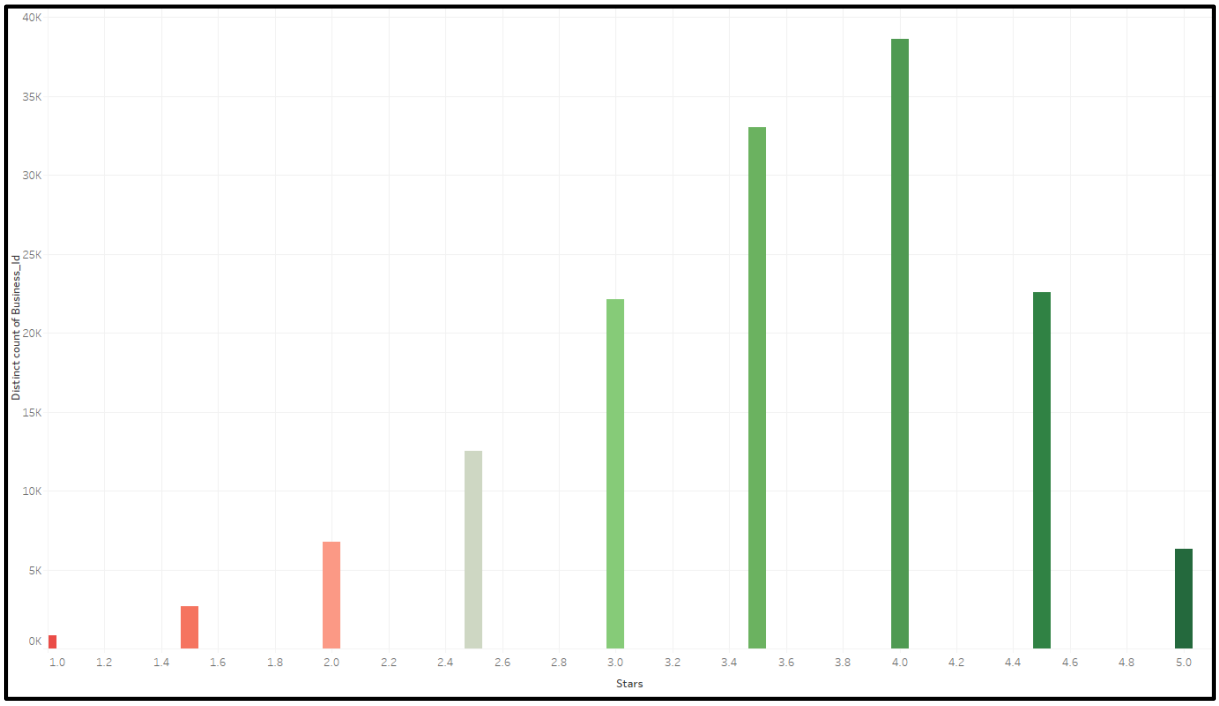

Figure 13: Review V/s Business Ids

- **Users have a wide range of number of reviews**
  - 13.42% of users reviewed only once.
  - 77.65% of users having less than 20 reviews and hence 22.35% of users having more than 20 reviews
  - Only 0.12%(1007) of users (846067) have more than 1K reviews.

- **Review Count over time**

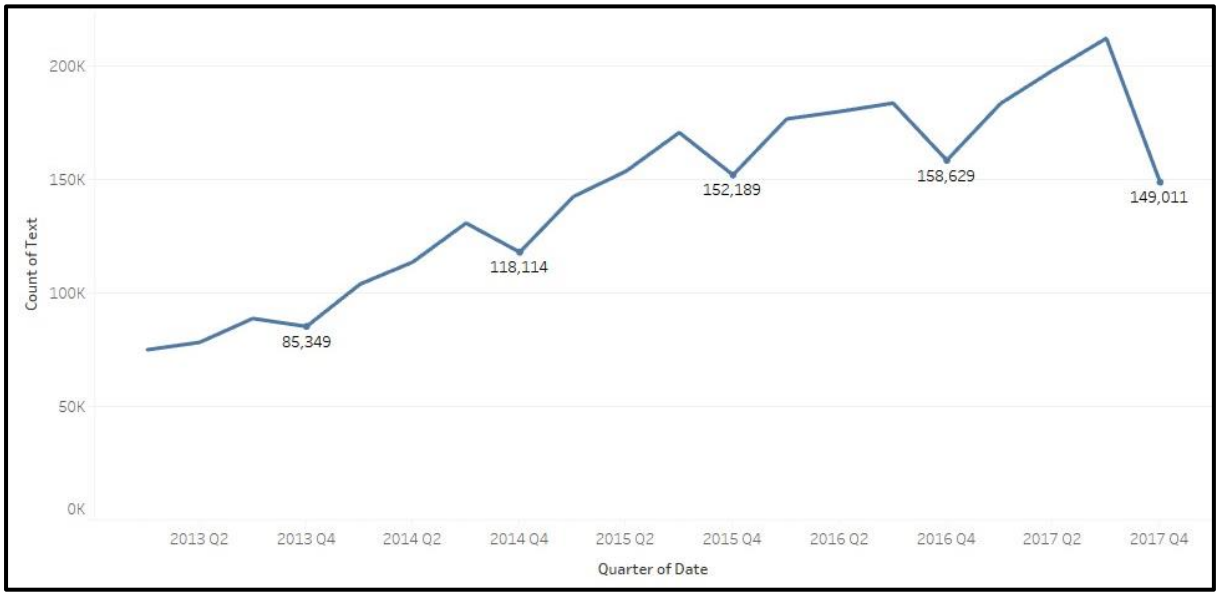

Figure 14: Review count Over time

- Though the review count is steadily increasing each year but it dips towards the quarter 4 for every year.
- Q4 is generally the holiday season in United States
- Dip in the reviews for Q4 of 2017 is higher than its predecessors. As the data is only available till December 11, 2017, this could be one of the reasons for the dip.

- **User's Relationship with the App**

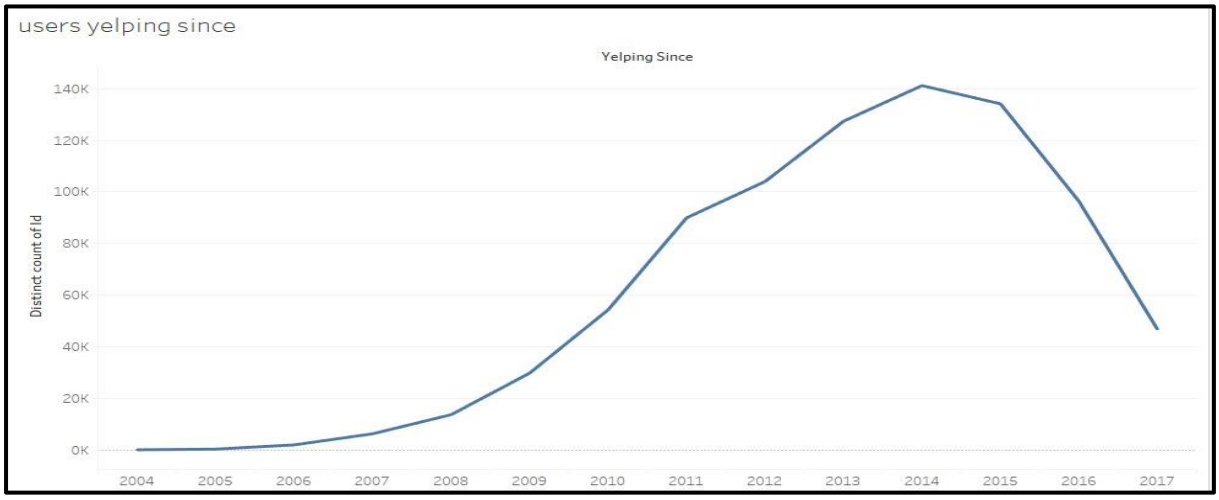

Figure 15: User Trend

- No. of users were max in 2014, around 140K, and have been decreasing since then. This is evident as majority of the tech-savvy population might already be enrolled in on Yelp. Also, competition in the market has increased since last 5 years. Today lot of apps are available in the market.

Model Building Phase

Analytical Approach

The objective of the project is to build a robust Dynamic Recommendation Model. There are two words to be focused here, 'Robust' and 'Dynamic'.

Following are the list of techniques, to be used, in the final model and their applicability:

- **Similarity Measures:** Assessing and identifying the similarity among Users and Restaurants, effectively, is one of the prime factors in success of the Model, thus making identification of similarity one of the most critical part of the project. Euclidean Distance, Cosine Distance, Pearson Correlation and Jaccard techniques were used for measuring the similarities. The results from each of these techniques was measured to understand the effectiveness of the applicable techniques.
- **Data Mining Techniques:** Once the Similarity among the users and the restaurants was identified, techniques such as k-means clustering were used for grouping the objects together. To solve the classification problem, techniques such as Support Vector Machines

and Decision trees were used. As multiple machine learning models were used to obtain better predictive results, using ensemble techniques such as bagging, boosting and random forests, to improve the stability and accuracy of machine learning algorithms were incorporated.

- **Evaluation Techniques:** Evaluation of the results in recommendation models was a tricky part, techniques such as Cross-Validation, Regularization, Confusion Matrix, RMSE, Model Comparison to assess the efficiency and efficacy of different techniques and recommendations were incorporated in the model.

Algorithm

To cater to the main objectives, 'Robustness' and 'Dynamic Nature' of the Recommendation model, the first part of the model development was focused towards building the Static Recommendation System, Robustness of the model was addressed during this stage. The Second Stage of the model building model was focused towards building a dynamic model.

Algorithms used for the model:

- **Static Recommendation System**
  - Collaborative Filtering Recommender System (User Based)

Algorithm:

If (U1 == RA)
    If ((U2 == RA & U2 == RB)
        If Similarity (U1, U2) >= 0.4
            Then U1 ε RRB

Notations:

User 1	:	U1
User 2	:	U2
Restaurant A	:	RA
Restaurant B	:	RB
User 1 likes Restaurant A	:	U1 == RA
User 2 likes Restaurant A	:	U2 == RA
User 2 likes Restaurant B	:	U2 == RB
Recommend Restaurant B	:	RRB
Similarity between User 1 & User 2	:	Similarity (U1, U2)
Recommend Restaurant B to User 1	:	U1 ε RRB

Theorem:

If two users share the same interests in the past, i.e. If U1 & U2 has strong similarity, their restaurant selection or preference will also be similar in future.

Definition:

We have considered that User 1 is similar to User 2 only if the similarity between User 1 & User 2 is greater or equal to 0.4. Similarity of a User is calculated based on the

features or parameters associated with the compared Users and is matched with the user's historical preferences

Remark:
For convenience, we have taken 0.4 as benchmark. Going forward we look forward to optimize this number.

- Content Based Recommender System (Restaurant Based)

Algorithm:
If (U1 == RA)
    If Similarity (RA, RB) >= 0.4
        Then U1 ε RRB

Notations:

User 1	:	U1
Restaurant A	:	RA
Restaurant B	:	RB
User 1 likes Restaurant A	:	U1 == RA
Recommend Restaurant B	:	RRB
Similarity between Restaurant 1 & 2	:	Similarity (RA, RB)
Recommend Restaurant B to User 1	:	U1 ε RRB

Theorem:
This system recommends Restaurants to users by taking the similarity among the compared restaurants and user profiles (historic preference) into consideration. The system will recommend Restaurants similar to those that the user has liked in the past.

Definition:
We have considered that Restaurant A is similar to Restaurant B only if the similarity between Restaurant A & B is greater or equal to 0.4. Similarity of a Restaurant is calculated based on the features or parameters associated with the compared Restaurants and is matched with the user's historical preferences

Remark:
For convenience, we have taken 0.4 as benchmark. Going forward we look forward to optimize this number.

- Knowledge Based Recommender System (Based on User's Purchase History)

Algorithm:
If (U1 == CRA)
    If Similarity (RA, RB) >= 0.4
        If Similarity (CRA, CRB) >= 0.4
Then U1 ε RCRB

Notations:
User 1 : U1

Restaurant A	: RA
Restaurant B	: RB
Category 1 from Restaurant A	: CRA
Category 1 from Restaurant B	: CRB
User 1 likes Category 1 from Restaurant A	: U1 == CRA
Category 1 from Restaurant A is similar to that of Restaurant B	: Similarity (CRA, CRB)
Recommend Category 1 from Restaurant B	: RCRB
Similarity between Restaurant 1 & 2	: Similarity (RA, RB)
Recommend Category 1 from Restaurant B to User 1	: U1 ε RCRB

Theorem:
This system recommends Restaurants to users based on user's past preferences (Category of Food offered by Restaurant) and recommends Restaurants serving similar category of food, provided the service offering of the recommended restaurants is similar to that of the historic purchases, when the historic data is smaller.

Definition:
We have considered that Category of Food offered by Restaurant A is similar to the Category of food offered by Restaurant B only if the similarity between Restaurant A & B is greater or equal to 0.4 and the similarity of the category of food offered by Restaurant A & B is greater or equal to 0.4. Similarity of the recommended Restaurants & Categories is calculated based on the features or parameters associated with the user's historical preferences.

Remark:
For convenience, we have taken 0.4 as benchmark. Going forward we look forward to optimize this number.

- **Dynamic Recommendation System**

Algorithm:
If (U1 == RA)
    If Similarity (RA, RB) >= 0.4
        If Similarity Trend (RA, RB) >= 0
            Then U1 ε RRB

Notations:

User 1	: U1
Restaurant A	: RA
Restaurant B	: RB
User 1 likes Restaurant A	: U1 == RA
Recommend Restaurant B	: RRB
Similarity between Restaurant 1 & 2	: Similarity (RA, RB)
Trend of similarity b/w Restaurant 1 & 2 for last 1 year	: Similarity Trend (RA, RB)
Recommend Restaurant B to User 1	: U1 ε RRB

Theorem:
This system recommends Restaurants to users by taking the similarity among the compared restaurants and user profiles (historic preference) into consideration. The system will recommend Restaurants similar to those that the user has liked in the past.

Definition:
We have considered that Restaurant A is similar to Restaurant B only if the similarity between Restaurant A & B is greater or equal to 0.4. Similarity of a Restaurant is calculated based on the features or parameters associated with the compared Restaurants and is matched with the user's historical preferences. Also, the trend of the similarity for previous year is calculated and checked, if it is greater than or equal to zero. Recommendation for the Restaurant B will be made in case it satisfies all the conditions.

Remark:
For convenience, we have taken 0.4 as a benchmark for similarity and previous year as a benchmark for trend. Going forward we look forward to optimize this number.

Model Creation

- **Content Based Recommendation System**

Content-based systems rely on a typical description of items over feature vectors and then recommend novel items to users by computing some similarity metric between them and the items that the user has already rated.

The principal advantage of the content-based filtering approach is in its nature: it can start to recommend as soon as there is information about items available. The latter means that a recommender system does not require any user input to recommend.

Step 1: Loading & Understanding Dataset
Primary Dataset used for building the model was Master Dataset – Business. Business Id is the unique Id assigned to each Restaurant outlet. This is the column which differentiates Mc Donald's outlet based out of New York and Mc Donald's outlet based out of New Jersey

In the dataset available, there is parameter called 'Category', this parameter states the category of the restaurant based on the cuisine or restaurant type, such as American, Bar, Asian. This forms the distinctive factors among the restaurants. In the content-based filtering, Category of the restaurants forms the 'content' of the Restaurant. There are 186 such unique categories.

The other factors categorized as the feature of the restaurants was its location.

Table 3: Master Dataset - Business (Parameters)

S. No.	Master Dataset – Business
1	Business Id
2	Restaurant Category
3	Restaurant Name
4	Restaurant Neighborhood
5	Restaurant Address
6	Restaurant City
7	Restaurant State
8	Restaurant Postal Code
9	Restaurant Latitude
10	Restaurant Longitude
11	Restaurant Stars
12	Restaurant Review Count
13	Is Open

Next, the Master Dataset – Review is loaded in the model, this is done to understand user's preferences.

Table 4: Master Dataset - Review (Parameters)

S. No.	Master Dataset – Review
1	Review Id
2	Business Id
3	User Id
4	Stars
5	Date
6	Review Text
7	Useful
8	Funny
9	Cool

In the Master Dataset – Review, User Id is the unique id provided to the User and Review id is the unique id provided to the review. Star contain the information about how many Stars have been provided to the user to that mentioned business id and at the particular instance.

Review Text contains the review provided by the customer. Useful, funny and Cool categorizes if the Review was useful, funny or cool, this is the characteristic of the Review and is captured using a binary value, 0 or 1.

Step 2: Processing the Dataset

While processing the dataset following factors were taken into consideration:

- Merging of Dataset: The two datasets, Master Dataset – Business & Reviews, needs to be merged together. This data frame is termed as 'Ratings'.
- Splitting of Dataset: The models were built on data from 2014 and testing was done on the data available for following year. As the model is built on 2014, business Ids which were in-business in 2014 were taken into consideration. Since, the business id's inception date is not covered in database, we have assumed that the Business ids which were reviewed in 2014 were the only ones active, this leaves us with **29,053 unique Business ids**.

```
str(rating)

'data.frame':    1398164 obs. of  12 variables:
 $ user_id    : Factor w/ 155382 levels "---1lKK3aKOuomHnwAkAow",..: 1 1 1 1 1
2 2 2 2 2 ...
 $ business_id: Factor w/ 29053 levels "--7zmmkVg-IMGaXbuVd0SQ",..: 9227 9227
9227 9227 9227 8638 8638 8638 8638 8638 ...
 $ review_id  : Factor w/ 369462 levels "----X0BIDP9tA49U3RvdSQ",..: 148975
148975 148975 148975 148975 322193 322193 322193 322193 322193 ...
 $ stars      : int  5 5 5 5 5 5 5 5 5 5 ...
 $ date       : Factor w/ 365 levels "2014-01-01 00:00:00",..: 176 176 176 176
176 231 231 231 231 231 ...
 $ text       : Factor w/ 369347 levels "'Absolutely amazing' is the only way
that I can describe the vegetarian combo at Blue Nile. A vegan delight, th"|
__truncated__,..: 253434 253434 253434 253434 253434 59955 59955 59955 59955
59955 ...
 $ useful     : int  1 1 1 1 1 0 0 0 0 0 ...
 $ funny      : int  1 1 1 1 1 0 0 0 0 0 ...
 $ cool       : int  1 1 1 1 1 0 0 0 0 0 ...
 $ restaurant : Factor w/ 31109 levels "'ONO PokÃ© Bar",..: 18084 18084 18084
18084 18084 4261 4261 4261 4261 4261 ...
 $ Category   : Factor w/ 186 levels "Afghan","African",..: 3 116 168 27 139
79 35 139 52 116 ...
 $ user       : Factor w/ 22372 levels "","'Nette","-CHris.",..: 14156 14156
14156 14156 14156 15694 15694 15694 15694 15694 ...
```

- Removal of duplicates: The duplicate entries were excluded from the model.
- Removal of unwanted columns, such as Postal Code, Latitude, Longitude, Address, Is Open, Review Text, Useful, funny and Cool.
- Creation of Matrix: Creation of matrix for building recommendation model is one of the most crucial steps. For the purpose of Content Based filtering User-Item (User V/s Business Id) was constructed.
- Creation of Affiliation Matrix: Data frame listing the restaurants and their corresponding category affiliation.

```
str(ratings)

'data.frame':    369462 obs. of  8 variables:
 $ restaurant   : Factor w/ 31109 levels "'ONO PokÃ© Bar",..: 21259 7211 7211
7211 7211 7211 7211 7211 7211 7211 ...
 $ business_id  : Factor w/ 29053 levels "--7zmmkVg-IMGaXbuVd0SQ",..: 1 2 2 2
2 2 2 2 2 2 ...
 $ restaurant_No: num  28982 2550 2550 2550 2550 ...
 $ Category     : Factor w/ 180 levels "Afghan","African",..: 69 36 160 36 160
36 139 139 160 139 ...
 $ stars        : int  4 3 5 5 5 4 5 5 1 5 ...
 $ user         : Factor w/ 22372 levels "","'Nette","-CHris.",..: 9178 9694
4930 4942 9943 12900 17444 846 9516 9328 ...
 $ user_id      : Factor w/ 155382 levels "---1lKK3aKOuomHnwAkAow",..: 153722
86903 84267 99508 12647 104344 140530 90864 41082 102141 ...
 $ user_No      : num  153722 86903 84267 99508 12647 ...
```

Step 3: Clustering

Next, we initiate the process of grouping the objects together. The objective of this step is to find the similarities among the objects and group them together.

The Clustering of the different combinations are done using Scree Plot. A standard scree plot has the number of clusters on the x-axis, and the sum of the within-cluster sum of squares on the y-axis.

The within-cluster sum of squares for a cluster is the sum, across all points in the cluster, of the squared distance between each point and the centroid of the cluster. Ideally, very small within-cluster sum of squares is required, since this means that the points are all very close to their centroid.

```
SumWithinss = sapply(2:20, function(x) sum(kmeans(cat_matrix2[,c(-1,-2)],
centers=x, iter.max=1000)$withinss))
NumClusters = seq(2,20,1)
#SCREE plot
plot(NumClusters, SumWithinss, type="b")
```

To determine the best number of clusters using this plot (Figure 16), we want to look for a bend, or elbow, in the plot. For this particular dataset, it looks like 4 clusters is a good choice. A total of 6 clusters were selected to provide the users more specific recommendation as the number of combinations available are high in number and also the Sum within are comparable to the ideal choice.

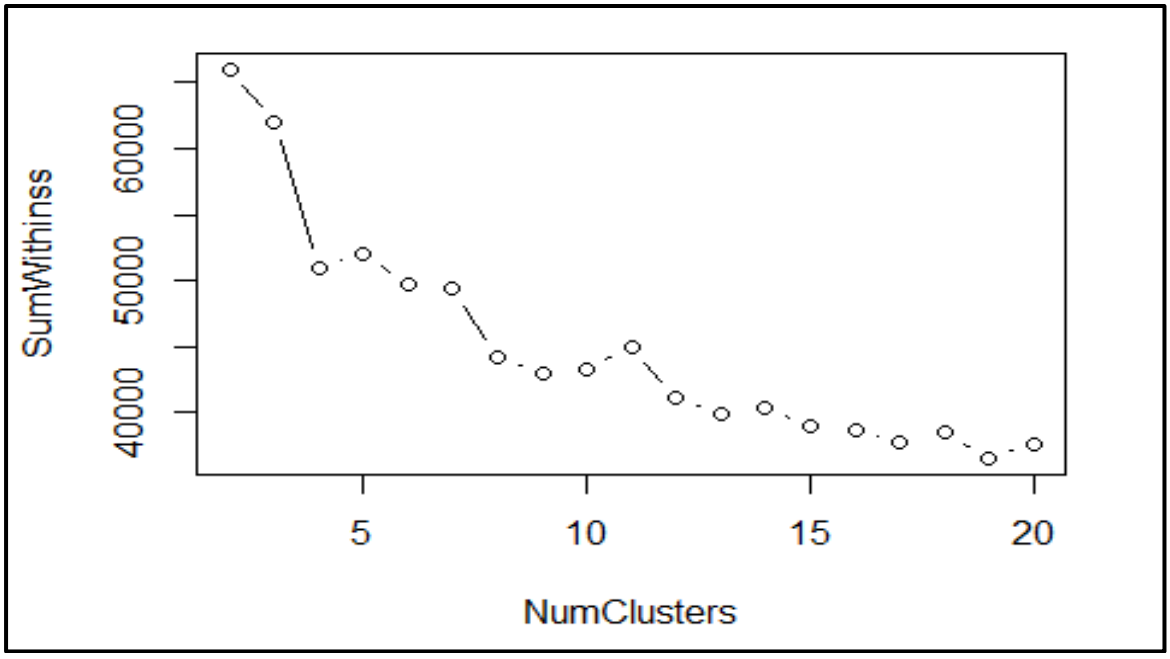

Figure 16: Content Based Filtering - Scree Plot

Next, to each restaurant a corresponding cluster number, was assigned.

```
setUserResCluster<-function(resCluster, activeUser){
  df1<- data.frame(cbind(cat_matrix2$business_id, clusterNum =
resCluster$cluster))
  names(df1)<-c("res_id", "cluster")
  activeUser$cluster<-df1[match(activeUser$restaurant_No, df1$res_id),2]
  return(activeUser)
}
```

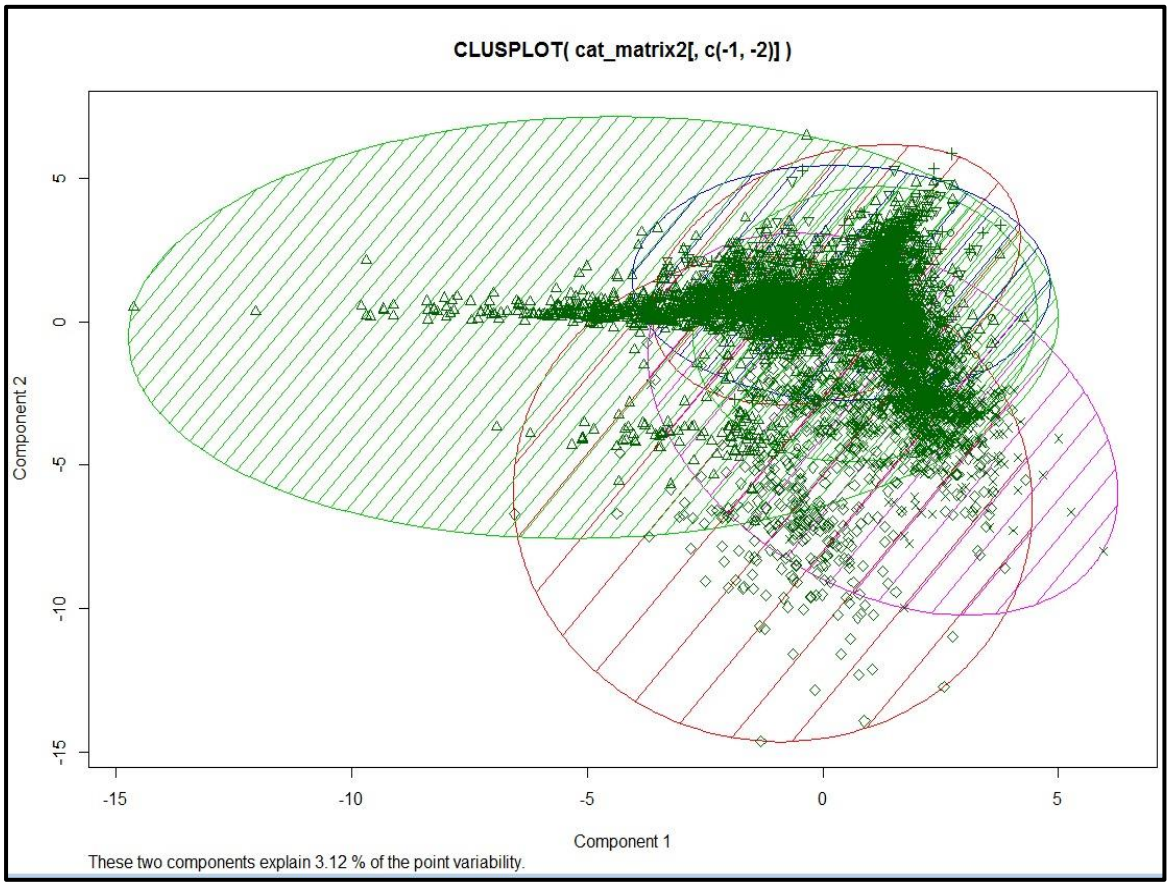

Figure 17: Clustering - Content Based Recommendation Model

Further, for each cluster the average of the restaurant ratings was calculated. In case the average rating of the restaurant and the category of the restaurant matches the category preferred by this cluster, the return value is "like".

An integer vector, in which all clusters whose average rating is greater or equal to 3 are included. If the clusters with >= 3 rating, are not found, the model will give a dummy value of 'zero'.

Next, Aggregate function is used along with the cluster memberships to determine variable means for each cluster in the original metric. If the max mean rating is below 3 it gives out the dummy value of 'zero', else it gives the cluster number of the max mean value.

A data frame was created to get a list of all restaurants and their associated clusters. If the like has a 'zero; value it randomly selects 100 restaurants, else it selects all restaurants from the winning max mean cluster.

Step 4: Implementation of the Clusters

The metrics created in the last step are now brought to use, to perform the following calculations:

- Creation of restaurant cluster and find the relevant information about the selected user.
- Calculation of the average rating per cluster and finding out the restaurants to the cluster, which the user likes the most.
- Selection of all the Business Ids the user has not yet seen.

The return value contains the restaurant name in addition to the restaurant ID.

Model Logic:
Once the User Id is fed in the model, the restaurant cluster will be identified, as per user's past preferences. Next, the model will select the restaurants that the customer has not rated yet, assuming that these are the restaurants that the user has not tried yet, these will be zeroed upon and Restaurant Name will be fetched from the Database. Next based on the average rating provided to these restaurants, top restaurants will be selected, based on the user's proximity to the restaurants, and suggestion will be made accordingly. Also, this can be executed at a city level, for the Users which are new to the city can view top 50 restaurants in the city, **which are in line with their taste palates.**

```
 # according to plan we call all functions in order of our logic
  resCluster<-clusterRestaurants(cat_matrix2,no_cluster)
  activeUser<-getUserInfo(ratings_bkup, user_No)
  activeUser<-setUserResCluster(resCluster, activeUser)
  like<-getMeanClusterRating(resCluster, activeUser)
  recommend<-getGoodRes(like, resCluster, cat_matrix2)
  # only select not yet rated restaurants
  recommend<-recommend[-activeUser$restaurant_No]
  # add restaurant name
  recommend <- data.frame(recommend)
  colnames(recommend) <- c("restaurant_No")
  #cat_matrix2$business_id <- as.integer(cat_matrix2$business_id)
  recommend <- left_join(recommend, ratings_bkup, by="restaurant_No")
  recommend <- recommend[,c(1,2)]
  recommend <- recommend[!duplicated(recommend),]
  return(recommend)
}
#7.
suggestRes<-function(cat_matrix2, ratings_bkup, user_No, no_res,no_cluster){
  #get suggestions
  suggestions = getRecommendedRes(cat_matrix2, ratings_bkup,
user_No,no_cluster)
  #select stated number of selections
  suggestions = suggestions[1:no_res,]
  colnames(suggestions)[colnames(suggestions)=="restaurant"] <- "Name"
  suggestions <- left_join(suggestions, df_itemList_group, by="Name")
```

```
  top50 = as.data.frame(suggestions)
  return(top50)
}
#8.
#set number of suggestions to view
no_res<-50
#user number
userid<-1
activeUser<-getUserInfo(ratings_bkup, userid)
activeUser
##                   restaurant            business_id restaurant_No
## 115583 Nacho Daddy Downtown eJKnymd0BywNPrJw1IuXVw             1
##              Category stars cluster
## 115583 American (New)     5       0
top50 <- suggestRes(cat_matrix2, ratings_bkup, userid, no_res,no_cluster)
writeLines("The chosen user liked: ")
## The chosen user liked:
kable(activeUser[,c(-2,-3,-6)],format = "rst",row.names = FALSE)
writeLines("You may also like these Restaurants:")
## You may also like these Restaurants:
kable(top50[,c(2,3)][1:10,],format = "rst",row.names = FALSE)

city <- master_business2[,c('Business.Id', 'Name','City','State')]
city <- city %>% group_by(Business.Id) %>% dplyr::summarise(Name =
first(Name),City = first(City),State = first(State))
colnames(city) <- c('business_id','restaurant','city','state')
idf_city <- merge(idf,city, by='business_id')

idf_city$city <- as.character(idf_city$city)
idf_city$state <- as.character(idf_city$state)
idf_city <- idf_city[order(idf_city$restaurant_No),]

#get 50 restaurants for User 1 from recommender system
colnames(top50)[colnames(top50)=="business_id"] <- "restaurant_No"
top50 <- merge(idf_city,top50, by="restaurant_No")
top50 <- top50[,c(-6)]

# For example, if user 1 want to get recommendation for restaurants in Las
Vegas, we can find out from the top50 list
Lasvegas <- filter(top50,city == "Las Vegas")
writeLines("The chosen user liked: ")
```

```
============================  ================================================================================
Name                          Category
============================  ================================================================================
Cafe Melissa                  American (Traditional),Food,Mexican,Gluten-Free,Restaurants,Cafes,Desserts
Bad Daddy's Burger Bar        Fast Food,Food,Restaurants,Burgers,Ice Cream & Frozen Yogurt
Euro Bakery                   Food,Bakeries
Brewster's                    Bars,Burgers,American (New),Restaurants,Pubs
E Noodles                     Restaurants,Asian Fusion,Noodles
Kaizen Fusion Roll & Sushi    Restaurants,Japanese,Sushi Bars,Asian Fusion
Settebello Pizzeria Napoletana Restaurants,Pizza
Sammy's Restaurant & Bar      Restaurants,Gluten-Free,Breakfast & Brunch,Pizza
Cucina by Wolfgang Puck       Italian,Restaurants,Pizza
Loblaws                       Food
============================  ================================================================================
```

Figure 18: Recommendation made to the Selected User Id (Content Based Recommender System)

restaurant	city	state	Category
Cucina by Wolfgang Puck	Las Vegas	NV	Italian,Restaurants,Pizza
Stacks & Yolks	Las Vegas	NV	Sandwiches,Breakfast & Brunch,Burgers,American (Traditional), urants,Food,Pretzels
Fat Choy	Las Vegas	NV	American (New),Asian Fusion,Chinese,Restaurants
KoMex Fusion	Las Vegas	NV	Korean,Mexican,Asian Fusion,Restaurants
Bayside Buffet at Mandalay Bay	Las Vegas	NV	Restaurants,Buffets,American (Traditional)
Chuck E. Cheese's	Las Vegas	NV	Restaurants,American (Traditional),Pizza
Oak and Ivy	Las Vegas	NV	Cocktail Bars,Food,Bars,Beer,Wine & Spirits
Gold Spike	Las Vegas	NV	Bars,Restaurants,Cafes,Lounges

Figure 19: Top Restaurants recommended in Las Vegas to the selected User Id (Content Based Recommender System)

Step 5: Creation of Dynamic Model

Till Step 4, the model is equipped enough to cater to static palate of the Users, but that is not an ideal phenomenon, as already established, user's might preference change overtime, especially in case of food, actually it is worse in case of food as the food User may like to have during weekdays might not be the food that he/she is wiling to have on weekends. Also, there might be a change in the rating provided to a certain Business Id, it may improve or deteriorate over time.

The objective of this step is to exclude such business ids which have deteriorated over time and also exclude categories which have fallen from the User's preferred list.

For the sake of model creation, we have assumed that there is a change in user preferences, year on year.

Model Logic:

To capture the change in User preferences, the year-on-year trend of the ratings provided by the users is calculated. The average ratings, provided by the User Ids, to the specific Business Id, in the previous year, is captured and compared with that of the following year. In cases where the differential amount is zero or above are considered, these will either be excluded or recommended on a higher number in the list (list of recommendations made to customers). Once, the trend is captured and compared, all the steps in Step 3 & 4 will be executed and recommendations will be derived. Just like in previous step, this can be executed at a city level, for the Users which are new to the city can view top 50 restaurants in the city, **which are in line with their dynamic taste palates**.

```
compare_df13 <- master_review_2013[,c(3,5)]
compare_df13 <- compare_df13[order(compare_df13$business_id),]
compare_df13_group <- compare_df13 %>% group_by(business_id) %>%
dplyr::summarise(stars_13 = mean(stars))

compare_df14 <- master_review_2014[,c(3,5)]
compare_df14 <- compare_df14[order(compare_df14$business_id),]
```

```
compare_df14_group <- compare_df14 %>% group_by(business_id) %>%
dplyr::summarise(stars_14 = mean(stars))

merged_df <- merge(compare_df13_group, compare_df14_group, by = 'business_id')

merged_df <- mutate(merged_df, comparison = round((stars_14 - stars_13),2))

merged_df <- merged_df[order(merged_df$comparison),]

trend_negative <- split(merged_df,merged_df$comparison < 0.00)
trend_positive = split(merged_df,merged_df$comparison >= 0.00)[['TRUE']]

Recommendation based on dynamic Rating to the user that they might prefer within
the category

top50_trend <- merge(top50, trend_positive, by = 'business_id')
col1<- mapply(anyNA,top50_trend) # apply function anyNA() on all columns to
check if comparison column has any NA's

top50_trend <- top50_trend[order(-top50_trend$comparison),]
Lasvegas_dy <- filter(top50_trend,city == "Las Vegas")
writeLines("The chosen user liked: ")

writeLines("You may also like these Restaurants based on dynamic rating at
LV:")
```

restaurant	city	state	Category
Chuck E. Cheese's	Las Vegas	NV	Restaurants,American (Traditional),Pizza
Fat Choy	Las Vegas	NV	American (New),Asian Fusion,Chinese,Restaurants
Cucina by Wolfgang Puck	Las Vegas	NV	Italian,Restaurants,Pizza
Stacks & Yolks	Las Vegas	NV	Sandwiches,Breakfast & Brunch,Burgers,American (Traditional),Restaurants,Food,Pretzels
Gold Spike	Las Vegas	NV	Bars,Restaurants,Cafes,Lounges

Figure 20: Top Restaurants recommended in Las Vegas to the selected User Id, based on User's Dynamic taste palate

Table 5: Recommendation Comparison - Dynamic V/s Static Model (Content Based Filtering Model)

Recommendation Ranking	Static Recommendation	Dynamic Recommendation
1	Cucina by wolf gang Puck	Chuck E. Cheese's
2	Stacks & Yolks	Fat Choy
3	KoMex Fusion	Cucina by wolf gang Puck
4	Bayside Buffet at Mandalay Bay	Stacks & Yolks
5	Chuck E. Cheese's	Gold Spike

It is evident from the results that there is a difference in the top 5 restaurants recommended to the same user id. Chuck E. Cheese which was ranked at fifth spot moved to first and KoMex Fusion was removed from the top 5 list.

- **Collaborative Based Recommendation System**
  Collaborative filtering (CF) algorithms identifies the patterns in the user activity to produce user specific recommendations. It depends on user data, for example user ratings and looks for a pattern in the demand and preference. The idea behind Collaborative filtering is that the rating of a user for a new item is likely to be similar to that of another user, in future, if both users have rated other items in similar way, in past.

  The recommendation method is not dependent on having any additional information about the items (e.g. description, metadata, etc.) or the users (e.g. interests, demographic data, etc.) in order to generate recommendations.

  Collaborative filtering can be divided into two categories:
  - User-Based CF finds users who have similar appreciation for items as you and recommends new items based on what they like.
  - Item-based CF recommends items that are similar to the ones the user likes, where similarity is based on item co-occurrences (e.g. users who bought x, also bought y).

  Step 1: Loading & Understanding Dataset
  Primary Dataset used for building the model was Master Dataset – Business and Master Dataset – Reviews. This forms the distinctive factors among the users and their ratings.

  The primary factor is the category for which the user has provided the rating, it is assumed that the category or the that the customer has rated is the one that the customer interested in and will be open to the recommendations provided in the same category or similar restaurants. The other factors categorized as the feature of the restaurants was its location. As the User's address not available in the dataset, it is assumed that the rated restaurant's location can be considered approachable to the user and he/she will be open to the recommendations provided, of the restaurants, in the close proximity will be acceptable.

Table 6: Master Dataset - Business (Parameters)

S. No.	Master Dataset – Business
1	Business Id
2	Restaurant Category
3	Restaurant Name
4	Restaurant Neighborhood
5	Restaurant Address
6	Restaurant City
7	Restaurant State
8	Restaurant Postal Code
9	Restaurant Latitude
10	Restaurant Longitude
11	Restaurant Stars
12	Restaurant Review Count
13	Is Open

Table 7: Master Dataset - Review (Parameters)

S. No.	Master Dataset - Review
1	Review Id
2	Business Id
3	User Id
4	Stars
5	Date
6	Review Text
7	Useful
8	Funny
9	Cool

Next, the Master Dataset – User is loaded in the model, this is done to understand user's preferences.

Table 8: Master Dataset – User (Parameters)

S. No.	Master Dataset – User
1	User Id
2	Name
3	Review Count
4	Yelping Since
5	Useful
6	Funny
7	Cool
8	Fans
9	Average Stars
10	Compliment Hot
11	Compliment More
12	Compliment Profile
13	Compliment Cute
14	Compliment List
15	Compliment Note
16	Compliment Plain
17	Compliment Cool
18	Compliment Funny
19	Compliment Writer
20	Compliment Photos

Step 2: Processing the Dataset

While processing the dataset following factors were taken into consideration:

- Merging of Dataset: The three datasets, Master Dataset – Business, Users & Reviews, needs to be merged together. This data frame is termed as 'Ratings'.
- Splitting of Dataset: The models were built on data from 2014 and testing was done on the data available for following year. As the model is built on 2014, business Ids and User Ids which were in-business and were yelping respectively in 2014, were taken into consideration. Since, the business id's inception date is not covered in database, we have assumed that the Business ids which were reviewed in 2014 were the only ones active, this leaves us with **29,053 unique business ids, 369,462 reviews rated by 155,382 User Ids**

- Creation of Sparse Matrix: Recommender Lab is used for creating the Model for Collaborative filtering. As a requirement, the data frame needed to be converted into a sparse matrix and then into a 'Real Rating Matrix'.

```
rating <- merge(master_review_2014[,-
1],master_business[,c('Business.Id','Name','Category')],
by.x='business_id',by.y='Business.Id')
rating <- merge(rating,master_user_14[,c(2,3)], by.x='user_id',by.y='id')
dim(rating)
## [1] 1272555      12
```

- Removal of duplicates: The duplicate entries were excluded from the model.
- Building User-Item Matrix: Data frame listing the Business Ids and User Ids were created as matrix.

```
udf <- data.frame(user_No=
seq(1:length(unique(rating_t[,"user_id"]))),user_id=
unique(rating_t[,"user_id"]))
idf <- data.frame(restaurant_No=
seq(1:length(unique(rating_t[,"business_id"]))),business_id=unique(rating_t[,"
business_id"]))

ratings <- merge(rating_t,udf,by.x='user_id',by.y='user_id')
ratings <- merge(ratings,idf,by.x='business_id',by.y='business_id')
```

- Removal of unwanted columns: All parameters except Business Id, Business Name, Restaurant Category, Average Stars, User Id, User Name, User No. and Restaurant Number were considered for the purpose of model building. Restaurant Number and User Number are Unique no generated to map business id and user id

```
str(ratings)
## 'data.frame':    325604 obs. of  8 variables:
##  $ restaurant   : Factor w/ 11837 levels "'ONO PokÃ© Bar",..: 7974 2590
2590 2590 2590 2590 2590 2590 2590 2590 ...
##  $ business_id  : Factor w/ 29053 levels "--7zmmkVg-IMGaXbuVd0SQ",..: 1 2 2
2 2 2 2 2 2 2 ...
##  $ restaurant_No: num  12592 2213 2213 2213 2213 ...
##  $ Category     : Factor w/ 173 levels "Afghan","African",..: 65 150 130
150 150 130 35 130 150 35 ...
##  $ stars        : int  4 3 4 5 5 4 5 4 5 4 ...
##  $ user         : Factor w/ 22372 levels "","'Nette","-CHris.",..: 9178
6115 9675 12319 20533 13831 7635 14001 4750 13031 ...
##  $ user_id      : Factor w/ 155382 levels "---1lKK3aKOuomHnwAkAow",..:
153722 120391 76022 50252 115289 30419 132822 93440 9761 4475 ...
##  $ user_No      : num  141656 110979 70074 46306 106258 ...
```

Step 3: Understanding User V/s Item (Business Ids) Pattern

Once the User V/s Item Matrix is constructed we can understand the pattern of User Rating.

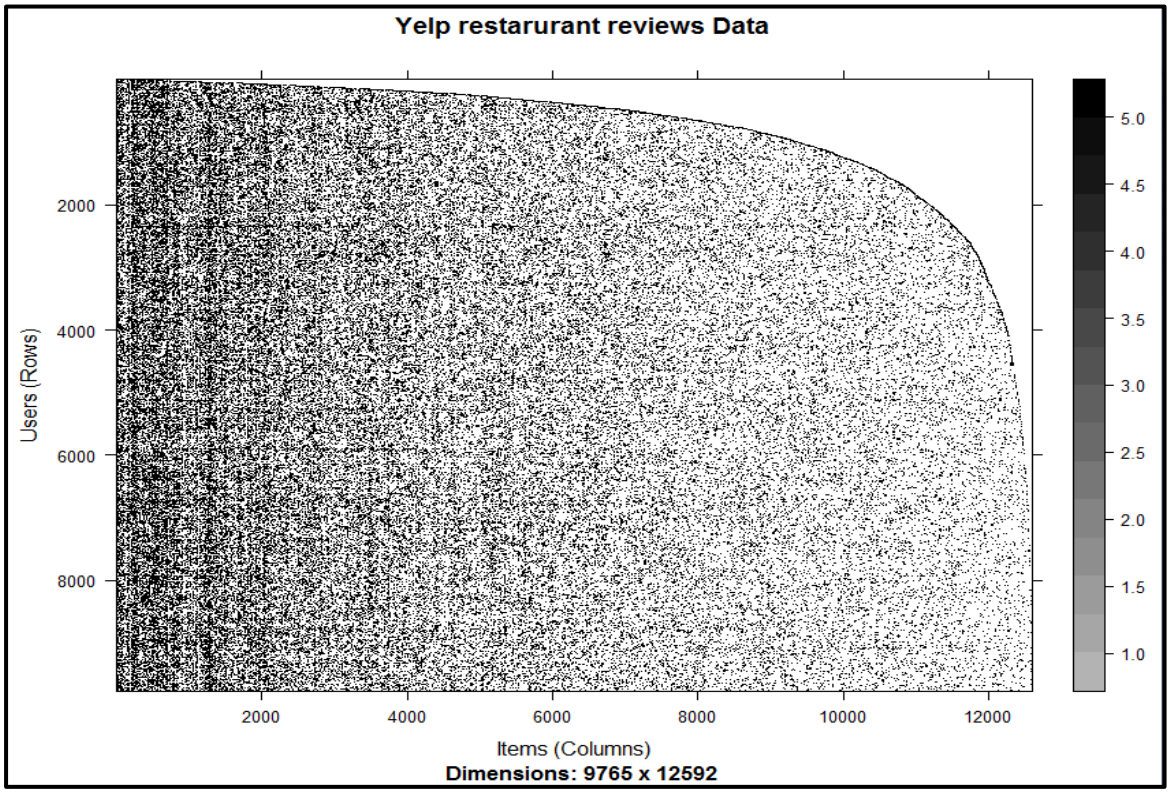

Figure 21: User Ratings - Heatmap

Most of the Businesses are rated between 3 &5

- Average Rating per business

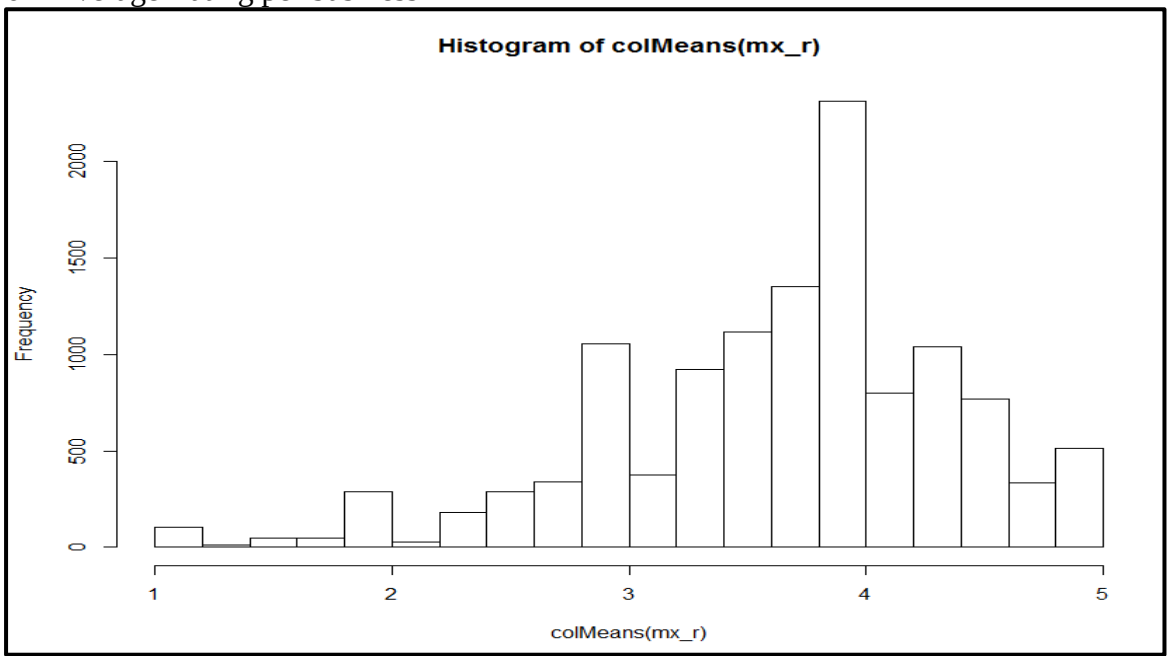

Figure 22: Average Rating per Business

- Normalized ratings histogram:
  From the normalized rating we can find that there are more zero rating

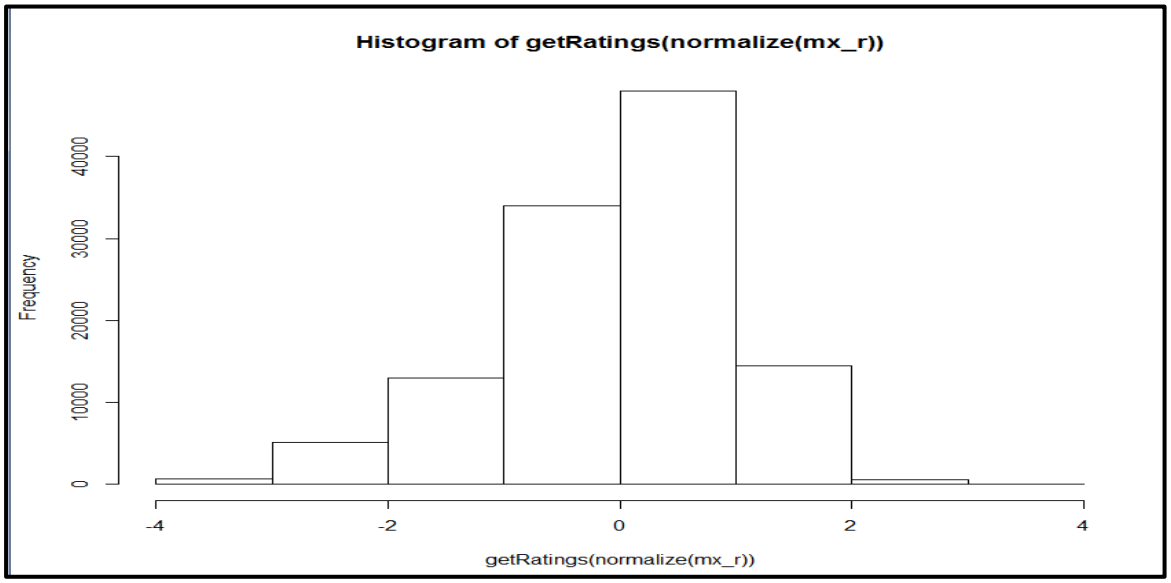

Figure 23: Normalized Rating Frequency

Step 4: Calculating Similarity Measures

Under User Based Collaborative Filtering, Similarity among the Users were captured and used in building Model. Cosine Distance, Euclidean and Jaccard were the techniques for calculating similarity distance among users. It was observed that the Jaccard was giving optimum results for all binary variables and for other variables Cosine Distance outperformed all others.

- Calculating Cosine Distance:
  Cosine similarity provided a measure of similarity among users.

```
similarity_users <- similarity(mx_r[1:50, ], method =  "cosine", which =
"users")
image(as.matrix(similarity_users), main = "User similarity")
```

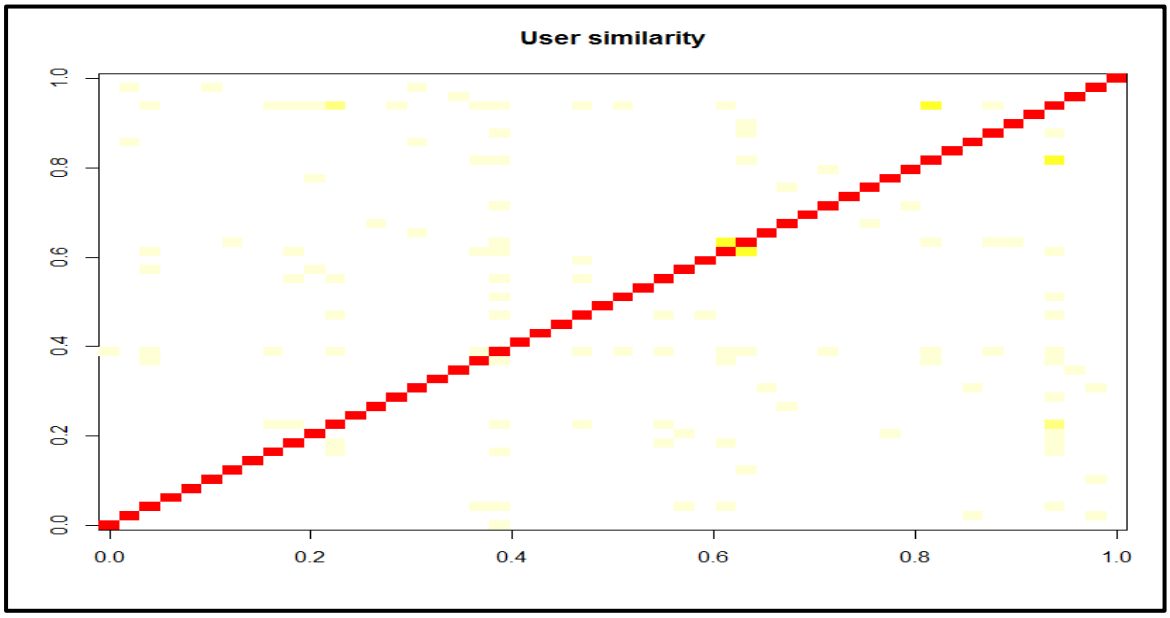

Figure 24: Cosine Distance (Users)

- Calculating Euclidean Distance:

```
similarity_users <- similarity(mx_r[1:50, ], method =  "euclidean", which =
"users")
image(as.matrix(similarity_users), main = "User similarity")
```

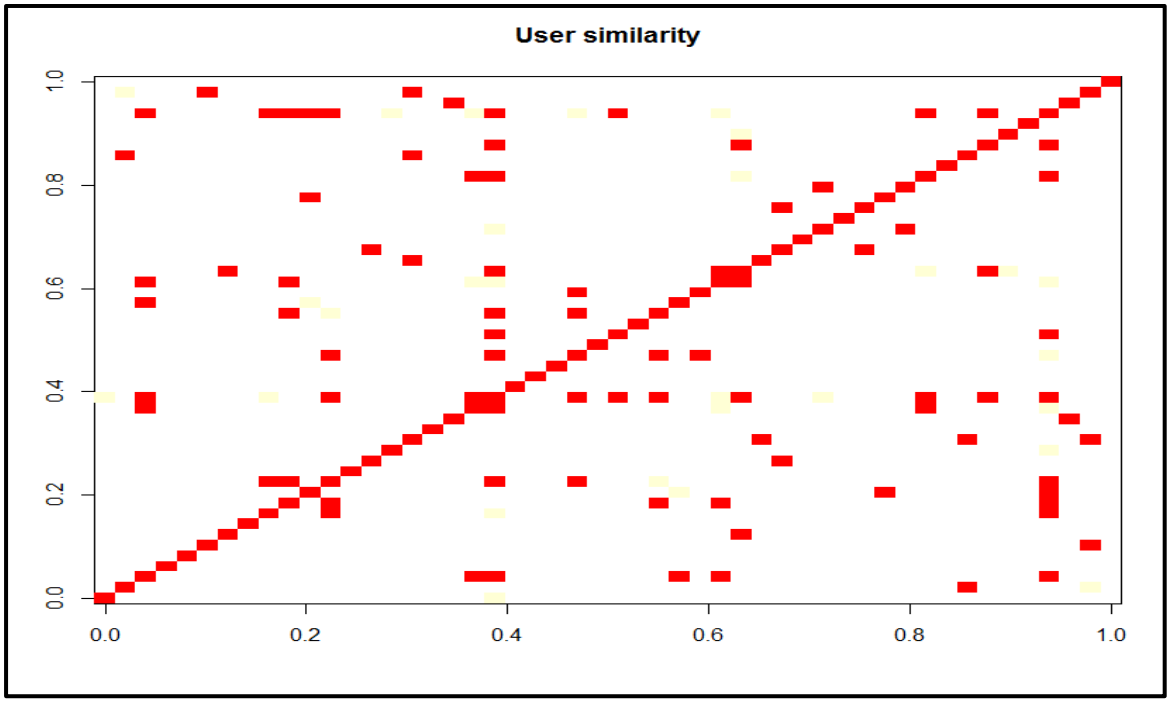

Figure 25: Euclidean Distance (Users)

Step 5: Building a User-based Collaborative Filtering Model

The model is build using Recommender lab package in R. Cosine Similarity is considered for calculating the similarity among the users.

```
rec_ubcf_cos=Recommender(mx_r,method="UBCF", param=list(normalize = "Z-
score",method="Cosine",nn=30))
print(rec_ubcf_cos)
```

Model Logic:
We start by selecting a User, in this case the first User Id in the dataset. Once the parameters are set to the model stated above, predictions can be made on the user's restaurant choices. Out of the list of predictions, top 10 predictions can be selected and mapped back to the Business Ids of the restaurants and User No to User Id, this is an important step as the prediction obtained are in form of Restaurant Number and User Number, as established earlier. This is later filtered by the city which has closest proximity to the User and top 10 restaurants, which are **in line with User's palate are recommended**. This can further be extended to other Users and other cities, in case the said User is travelling to the other city.

```
top10_pred_rate <- predict(rec_ubcf_cos,mx_r[1],type = "ratings")
top10_pred_rate <- as(top10_pred_rate, "list") #convert recommenderlab object
to readable list
top10_pred_rate

#Obtain top 10 recommendations for 1st user in dataset
top10_pred_list <- predict(rec_ubcf_cos,mx_r[1],n=10)

top10_pred_list <- as(top10_pred_list, "list") #convert recommenderlab object
to readable list
top10_pred_list

Mapping the Restaurant Numbers to Restaurant names
top_10_df=data.frame(top10_pred_list)
colnames(top_10_df)="BusinessId"
top_10_df <- gsub("R","",top_10_df$BusinessId)
top_10_df=data.frame(top_10_df)
colnames(top_10_df)="restaurant_No"
str(top_10_df)

top_10_df$restaurant_No <-
as.numeric(levels(top_10_df$restaurant_No))[top_10_df$restaurant_No]

Mapping the the restaurant ids with Restaurant names
names <- left_join(top_10_df, ratings_bkup, by="restaurant_No")
names <- names[,c(1,2)]

names_unique <- unique.data.frame(names)
colnames(names_unique)[colnames(names_unique) == "restaurant"]="Name"
names_unique

##      restaurant_No                              Name
## 1              501                 The Goodwich
## 181            547                Baguette Cafe
```

```
## 338            659                      Babystacks Cafe
## 461           2887                       Banger Brewing
## 582           4483                    Sin City Cupcakes
## 596           4572                  Greens and Proteins
## 742           7429           Lola's A Louisiana Kitchen
## 786            557                        Vintner Grill
## 881            775        Lucille's Smokehouse Bar-B-Que
## 1005           232 John Mull's Meats & Road Kill Grill

#Since user_No is of type integer in ratings data created, we typecast userNo
in our recommendations as well
user1$user_No <- as.numeric(levels(user1$user_No))[user1$user_No]
user1_det <- left_join(user1, ratings_bkup, by="user_No")
user1_det

Considering the location in the recommendation system
city <- master_business[,c('Business.Id', 'Name','City','State')]
city <- city[!duplicated(city$Name),]
colnames(city) <- c('business_id','restaurant','city','state')
idf_city <- left_join(idf,city, by='business_id')
idf_city$restaurant_No <- paste("R", 1:12592, sep = "")
idf_city$city <- as.character(idf_city$city)
idf_city$state <- as.character(idf_city$state)

#Predicting top 50 restaurants for User 1 from recommender system
(p_top50 <- predict(Rec.model.cos, mx_r[1],type="topNList",n=50))
## Recommendations as 'topNList' with n = 50 for 1 users.
## Recommendations as 'topNList' with n = 50 for 1 users.
# filter the restaurant for User 1 based on location

pred_restaurant <-  data.frame(as(p_top50, "list"))
pred_restaurant
colnames(pred_restaurant) <- "U18"
pred_restaurant[] <- lapply(pred_restaurant, as.character)
pred_restaurant$restaurant_No <- pred_restaurant$U18

pred_restaurant <- left_join(pred_restaurant,idf_city, by='restaurant_No' )
pred_restaurant$city <- as.character(pred_restaurant$city)
pred_restaurant$state <- as.character(pred_restaurant$state)

# For example, if user 1 want to get recommendation for restaurants in Las
vegas, we can find out from the top100 list
Lasvegas <- filter(pred_restaurant,city == "Las Vegas")
head(Lasvegas, n=10)
```

	U18	restaurant_No	restaurant	city	state
1	R501	R501	The Goodwich	Las Vegas	NV
2	R547	R547	Baguette Cafe	Las Vegas	NV
3	R2887	R2887	Banger Brewing	Las Vegas	NV
4	R4483	R4483	Sin City Cupcakes	Las Vegas	NV
5	R7429	R7429	Lola's A Louisiana Kitchen	Las Vegas	NV
6	R557	R557	Vintner Grill	Las Vegas	NV
7	R232	R232	John Mull's Meats & Road Kill Grill	Las Vegas	NV
8	R1262	R1262	Raku	Las Vegas	NV
9	R2338	R2338	Kabuto	Las Vegas	NV
10	R6026	R6026	Gilcrease Orchard	Las Vegas	NV

Figure 26: Top Restaurants recommended in Las Vegas to the selected User Id (UBCF – Static Model)

Top Restaurants recommended in Las Vegas to the selected User Id (UBCF)

Step 6: Creation of Dynamic Model

The objective of this step is to understand the change in the palate of the customer and exclude business ids which have fallen from the User's preferred list.

For the sake of model creation, we have assumed that there is a change in user preferences, year on year. For production purposes this can be reduced to month on month as well

Model Logic:

To capture the change in User preferences, the year-on-year trend of the ratings provided by the users is calculated and the Business Ids for which the customer is providing the rating is captured. The average ratings, provided by the User Ids, to the specific Business Id, in the previous year is captured along with the change in the similarity of the Restaurant type of the restaurant User is opting for, next these are compared with that of the following year. In cases where the differential amount is zero or above are considered, these will either be excluded or be recommended on a higher number (in the list of recommendations made to customers). Once, the trend is captured and compared, all the steps in Step 4 & 5 will be executed and recommendations will be derived. Just like in previous step, this can be executed at a city level, for the Users which are new to the city can view top 50 restaurants in the city, which are in **line with their dynamic taste palates**.

restaurant	city	state	stars_13
Sin City Cupcakes	Las Vegas	NV	4.177778
Greens and Proteins	Las Vegas	NV	3.791209
Carrabba's Italian Grill	Las Vegas	NV	3.416667
Mezzo Bistro and Wine	Las Vegas	NV	3.709677
Popped	Las Vegas	NV	4.263158
Cucina by Wolfgang Puck	Las Vegas	NV	3.828947
Kabuto	Las Vegas	NV	4.597701
John Mull's Meats & Road Kill Grill	Las Vegas	NV	4.243243

Figure 27: Top Restaurants recommended in Las Vegas to the selected User Id (UBCF – Dynamic Model)

Recommendation Ranking	Static Recommendation	Dynamic Recommendation
1	The Goodwich	Sin City Cupcakes
2	Baguette Café	Greens and Proteins
3	Banger Brewing	Carrabba's Italian Grill
4	Sin City Cupcakes	Mezzo Bistro and Wine
5	Lola's A Louisiana Kitchen	Popped

It is evident from the results that there is a difference in the top 5 restaurants recommended to the same user id. The Goodwich which was at top spot is moved out of top 5 recommendations, Sin City Cupcakes have taken the top spot and all other recommendations from the top five have moved down the list.

Other models, such as Item Based Collaborative Filtering, Alternating Least Squares method (ALS) and Singular Value Decomposition (SVD), were built under Collaborative filtering, using similar methodology.

Model Evaluation and Conclusion

In order to recommend items to new users, collaborative filtering predicts the ratings of items that are not yet purchased. This is followed by recommending the top-rated items. We evaluate the model by comparing the predicted ratings with the actual ones. For the evaluation purposes let us park the later part. Since the k-fold is one of the most accurate approaches, we have used it here

- **Step 1: Preparing the data to evaluate models**
  For the purpose of evaluation, we have to split the data into testing and training datasets.

- **Step 2: Using cross-validation to validate models**
  Once the dataset is split, we apply K-fold approach to evaluate models. For the computational simplicity, dataset was split into smaller data frames and accuracy was computed using these data frames individually. Once the accuracy of each data frame was obtained, weighted average of all the accuracy was computed and this would be the accuracy of the model.

  Two was taken as value for k for computing 2-fold cross validation.

```
eval_sets <- evaluationScheme(mx_r, method="cross-validation", given=5, k=2,
goodRating = 4)

eval_sets
## Evaluation scheme with 5 items given
## Method: 'cross-validation' with 2 run(s).
## Good ratings: >=4.000000
## Data set: 9765 x 12592 rating matrix of class 'realRatingMatrix' with
115839 ratings.
#We can count how many items we have in each set:
size_sets <- sapply(eval_sets@runsTrain, length)
size_sets
## [1] 4882 4882
Using 2-fold approach, we get four sets of the same size 4882.
#Create a UBCF recommender, using cosine similarity and 25 nearest neighbours.
RMSE.model.cos <- Recommender(getData(eval_sets, "train"), method = "UBCF",
                              param=list(normalize = "Z-score",
method="Cosine", nn=25))

print(RMSE.model.cos)
## Recommender of type 'UBCF' for 'realRatingMatrix'
## learned using 4882 users.
### create predictions for the test users using known ratings
RMSE.prediction <- predict(RMSE.model.cos, getData(eval_sets, "known"),n=10,
type="ratings")
class(RMSE.prediction)
## [1] "realRatingMatrix"
```

```
## attr(,"package")
## [1] "recommenderlab"

evaluate recommendations on "unknown" ratings
#rmse_ubcf <- calcPredictionAccuracy(prediction, getData(e, "unknown"))[1]
RMSE_ubcf_accuracy <- calcPredictionAccuracy(RMSE.prediction,
getData(eval_sets, "unknown"),byUser = TRUE)
head(RMSE_ubcf_accuracy)
##           RMSE       MSE       MAE
## U18  0.0000000 0.0000000 0.0000000
## U27  0.5181685 0.2684986 0.5073161
## U32  0.9130534 0.8336665 0.8003435
## U49  0.9831692 0.9666218 0.8720041
## U78  0.9141761 0.8357179 0.8509605
## U191 0.8586912 0.7373505 0.7598988
as(RMSE_ubcf_accuracy, "matrix")
##               RMSE          MSE         MAE
## U18     0.00000000 0.000000e+00 0.000000000
## U27     0.51816850 2.684986e-01 0.507316072
## U32     0.91305338 8.336665e-01 0.800343466
## U49     0.98316924 9.666218e-01 0.872004117
## U78     0.91417609 8.357179e-01 0.850960501
## U191    0.85869118 7.373505e-01 0.759898779
## U215    1.40000000 1.960000e+00 1.400000000
## U240    0.84720300 7.177529e-01 0.689119763
## U255    1.16619038 1.360000e+00 1.000000000
## U264           NaN          NaN         NaN
## U323    1.35804297 1.844281e+00 1.201777047
## U329    0.83066239 6.900000e-01 0.750000000
## U331    0.39632067 1.570701e-01 0.396303433
## U385    1.66836581 2.783444e+00 1.500444010
RMSE_ubcf_accuracy[[1]]
## [1] 0
#avg(RMSE_ubcf_accuracy)

qplot(RMSE_ubcf_accuracy[, "RMSE"]) + geom_histogram(binwidth = 0.1) +
  ggtitle("Distribution of the RMSE by user")

RMSE_ubcf_accuracyk <- calcPredictionAccuracy(RMSE.prediction,
getData(eval_sets, "unknown"),byUser = FALSE)
RMSE_ubcf_accuracyk
##      RMSE       MSE       MAE
## 1.1165478 1.2466789 0.8679567
```

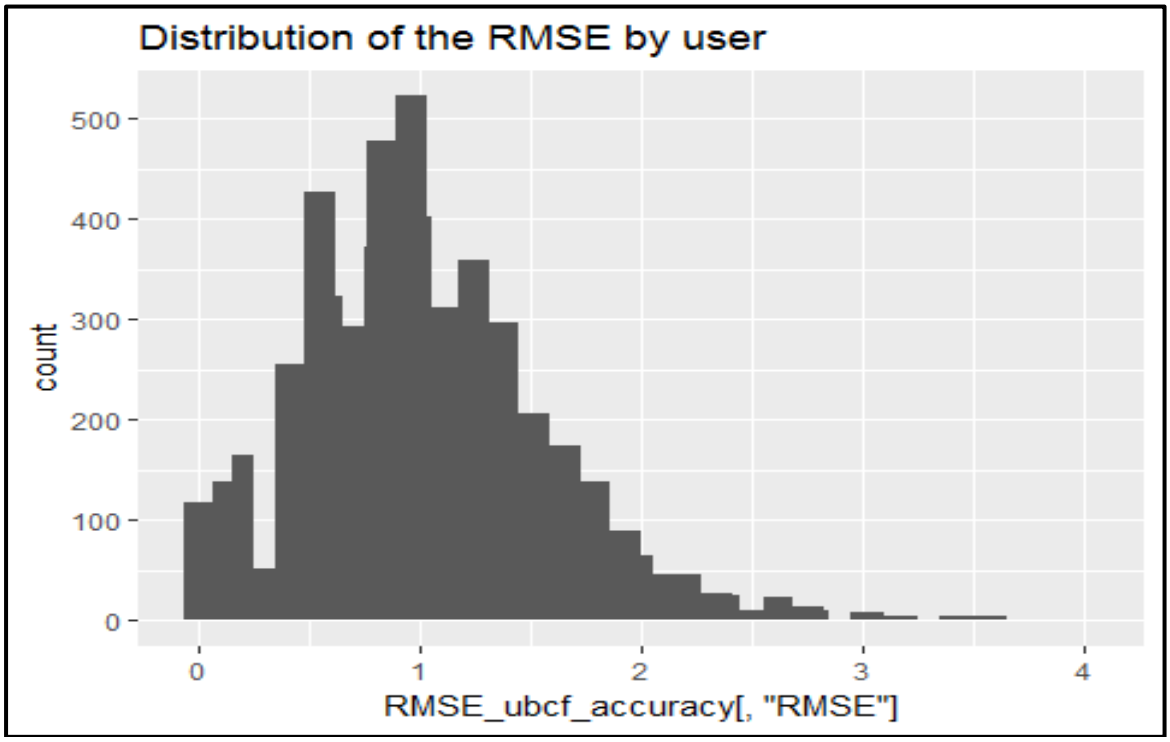

Figure 28: Distribution of the RMSE (UBCF)

We evaluated the model in cases where the 1, 3, 5, 10, 20, 100 recommendations are made to the users.

```
evaluation_results <- evaluate(eval_sets, method="UBCF", n=c(1,3,5,10,20,100))

class(evaluation_results)
## [1] "evaluationResults"
## attr(,"package")
## [1] "recommenderlab"
eval_results <- getConfusionMatrix(evaluation_results)[[1]]
eval_results
##             TP         FP       FN       TN   precision      recall
## 1   0.01904567  0.8109769 4.570346 12581.60 0.022945966 0.005137741
## 3   0.04382552  2.4462421 4.545566 12579.96 0.017600132 0.010399781
## 5   0.06389515  4.0862175 4.525497 12578.32 0.015396003 0.015367154
## 10  0.10546795  8.1947573 4.483924 12574.22 0.012706637 0.025170105
## 20  0.17489248 16.4255581 4.414499 12565.99 0.010535406 0.042506418
## 100 0.53778415 82.4644686 4.051608 12499.95 0.006479151 0.123720290
##             TPR          FPR
## 1   0.005137741 0.0000644526
## 3   0.010399781 0.0001944158
## 5   0.015367154 0.0003247535
## 10  0.025170105 0.0006512817
## 20  0.042506418 0.0013054301
## 100 0.123720290 0.0065539205
################################################################################
```

True Positives (TP): These are recommended restaurants that have been tried by the User

False Positives (FP): These are recommended restaurants that haven't been tried by the User
False Negatives(FN): These are not recommended restaurants that have been tried by the User
True Negatives (TN): These are not recommended restaurants that haven't been tried by the User
Precision: This is the percentage of recommended restaurants that have been tried. It's the number of FP divided by the total number of positives (TP + FP).
Recall: This is the percentage of purchased restaurants that have been recommended. It's the number of TP divided by the total number of restaurants tried (TP + FN). It's also equal to the True Positive Rate.

A perfect (or overfitted) model would have only TP and TN.

If we want to take account of all the splits at the same time, we can just sum up the indices:

```
columns_to_sum <- c("TP", "FP", "FN", "TN")
indices_summed <- Reduce("+", getConfusionMatrix(evaluation_results))[,
columns_to_sum]
head(indices_summed)
##             TP          FP       FN       TN
## 1   0.03440508   1.609052 9.161581 25163.19
## 3   0.08642228   4.843948 9.109564 25159.96
## 5   0.12656154   8.090723 9.069425 25156.71
## 10  0.20806881  16.226500 8.987917 25148.58
## 20  0.34896580  32.520172 8.847020 25132.28
## 100 1.06512390 163.280565 8.130862 25001.52

#avg(evaluation_results)
plot(evaluation_results, annotate = TRUE, main = "ROC curve")
```

Next, let us evaluate the ROC Curve

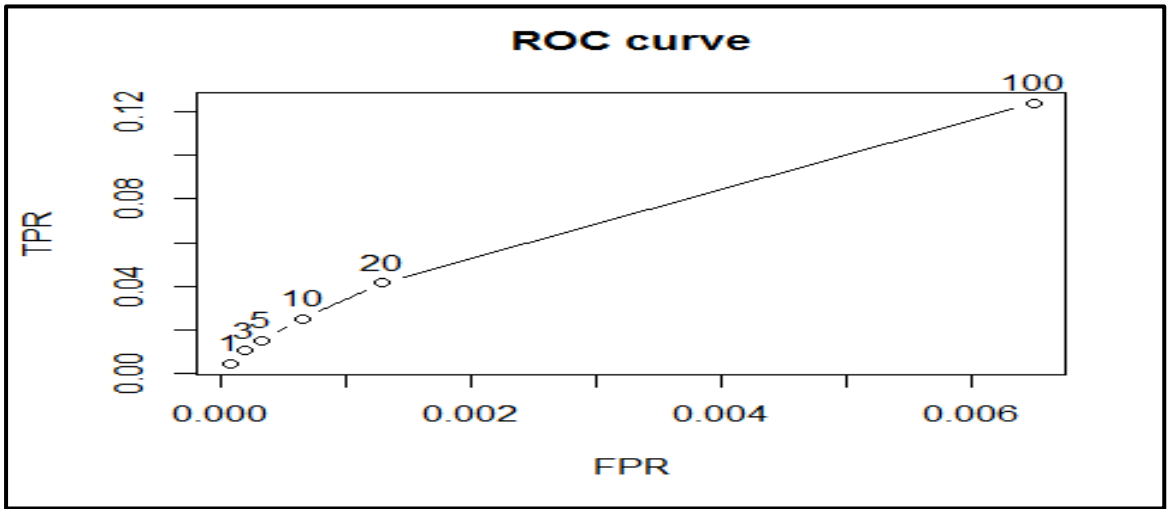

Figure 29: ROC Curve (UBCF)

```
plot(evaluation_results, "prec/rec", annotate = TRUE, main = "Precision-
recall")
```

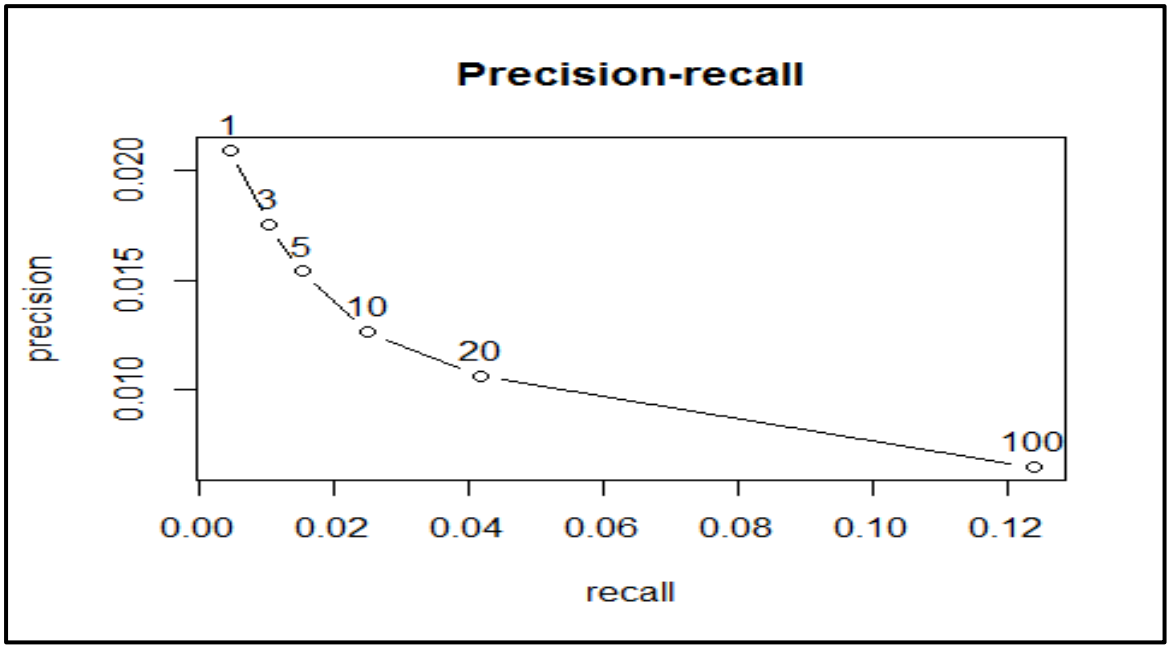

Figure 30: Precision - Recall (UBCF)

If a small percentage of rated restaurants are recommended, the precision decreases. On the other hand, the higher percentage of rated restaurants are recommended the higher is the recall.

- **Step 3: Comparison of other Models**

  Other models are compared such as ALS, SVD

```
algorithms <- list(
  RANDOM = list(name = "RANDOM", param = NULL),
  POPULAR = list(name = "POPULAR", param = NULL),
  IBCF = list(name = "IBCF", param = NULL),
  UBCF = list(name = "UBCF", param = NULL),
  ALS = list(name = "ALS", param = NULL),
  SVD = list(name = "SVD", param = NULL)
)

# Predict top-N recommendation lists
ev <- evaluate(eval_sets, algorithms, type="topNList", n=c(1, 5, seq(10, 100,
10)))
avg_matrices <- lapply(ev, avg)
avg_matrices
```

- **Step 4: Evaluating different Models**
  Let us evaluate the performance of each of the model using ROC Curve and Precision-Recall graphs.

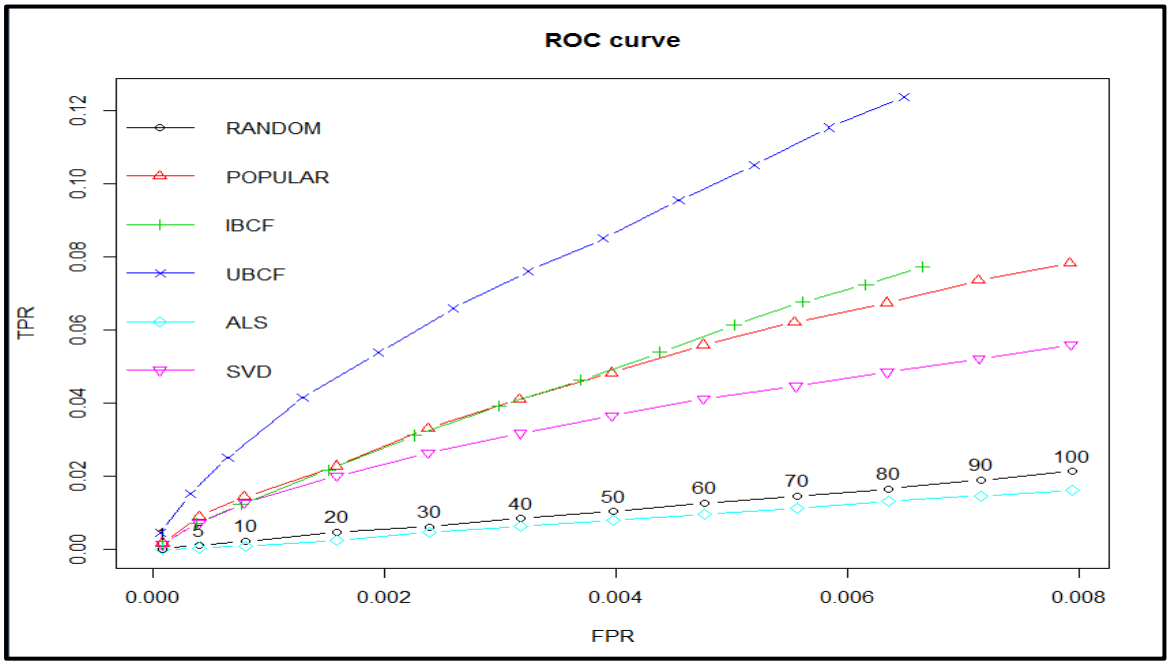

Figure 31: ROC Plot (Model Comparison)

Comparison of precision-recall curves for several recommender methods for the given-5 evaluation scheme.

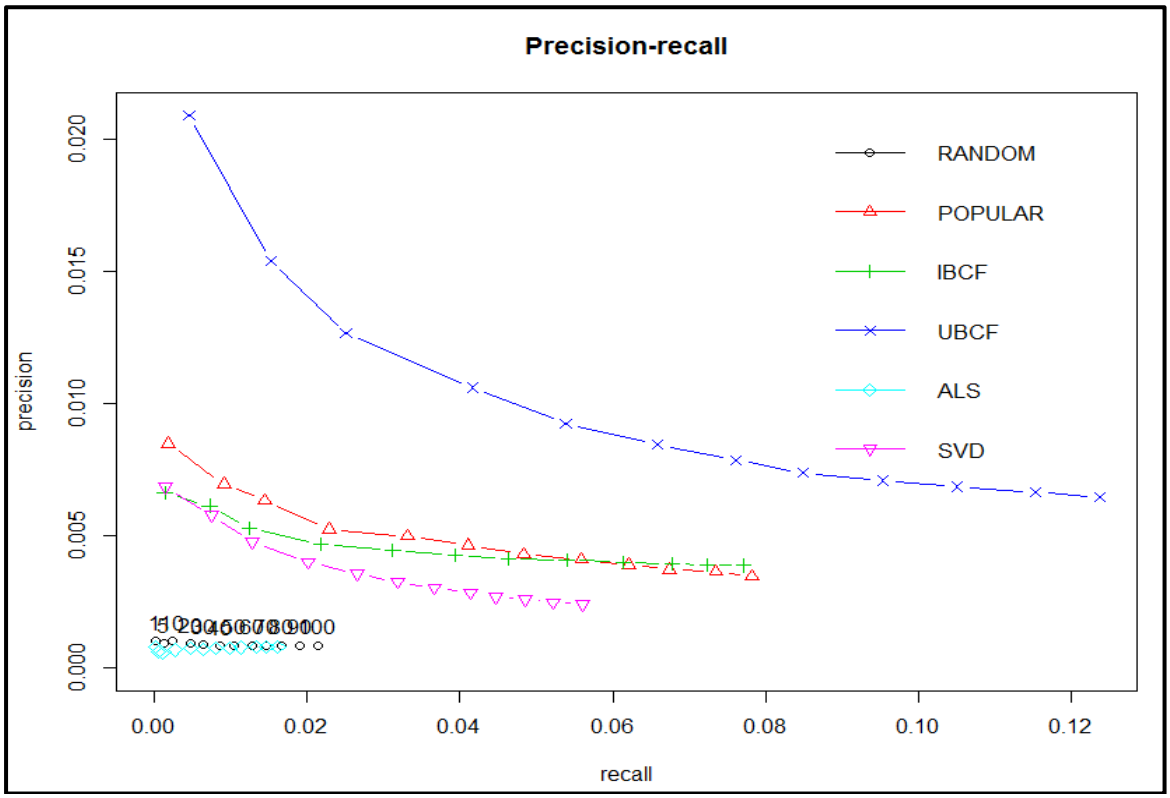

Figure 32: Precision-Recall (Model Comparison)

Comparison of RMSE, MSE, and MAE for recommender methods for the given-5 evaluation scheme.

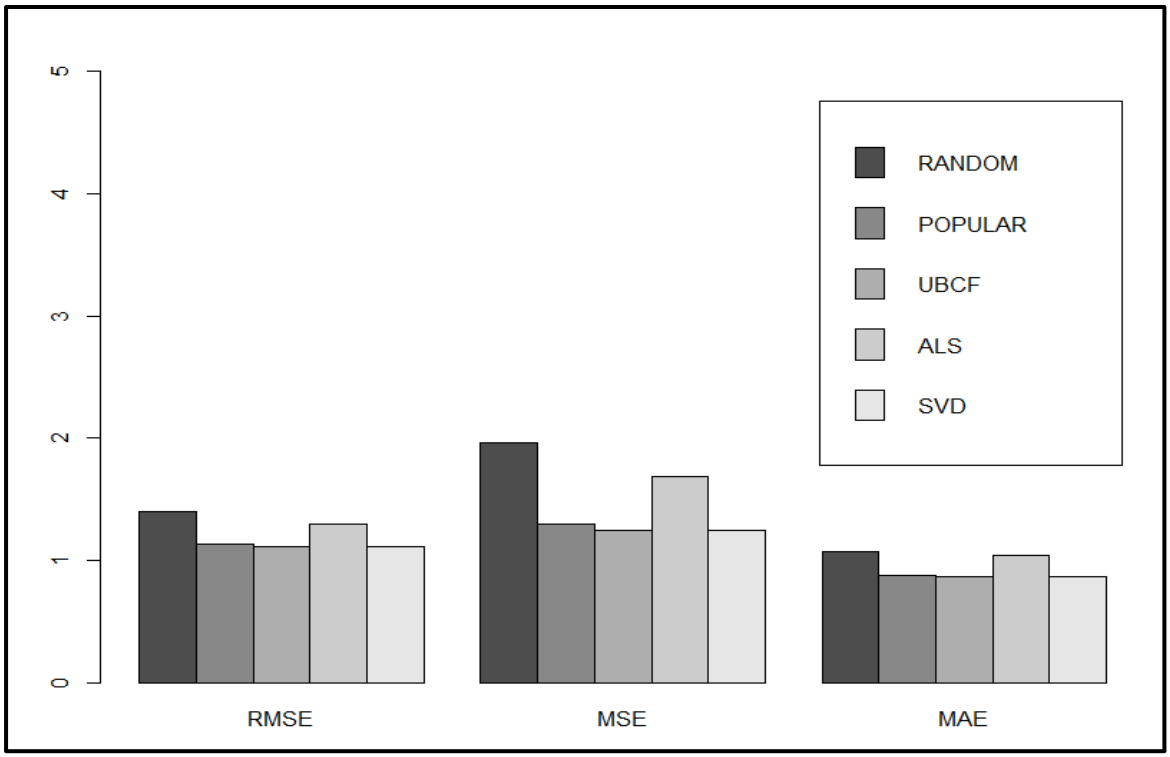

Figure 33: RMSE, MSE & MAE Comparison

Table 9: Model Performance Comparison

	RMSE	MSE	MAE
RANDOM	1.404816	1.973507	1.0736432
POPULAR	1.136857	1.292443	0.8809080
UBCF	1.116745	1.247120	0.8680726
ALS	1.301434	1.693732	1.0421037
SVD	1.116075	1.245624	0.8679257

A good performance index is the area under the curve (AUC), that is, the area under the ROC curve. Even without computing it, the chart shows that the highest is UBCF with cosine distance, so it's the best-performing technique. The UBCF with cosine distance is still the top performing model.

Bibliography

1. Evolution of Recommender Systems from Ancient Times to Modern Era: A Survey Richa Sharma* and Rahul Singh University Institute of Engineering, Chandigarh University, Gharuan, Mohali - 140413, Punjab, India;
2. Lu J, Wu D, Mao M, Wang W, Zhang G. Recommender system application developments: a survey. Decision Support Systems. 2015 Jun 30; 74:12–32.
3. Introduction to Recommender Systems Handbook by Francesco Ricci, Lior Rokach and Bracha Shapira
4. Content-Based Recommendation Systems Michael J. Pazzani1 and Daniel Billsus2
5. Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices by Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho (2007)
6. An Introduction to Recommender Systems by Francesco Ricci, Lior Rokach, Bracha Shapira (2010)
7. Ekstrand MD, Riedl JT, Konstan JA. Collaborative filtering recommender systems. Foundations and Trends in Human Computer Interaction. 2011 Feb 1;4(2):81–173
8. Evolution of Recommender Systems from Ancient Times to Modern Era: A Survey Richa Sharma and Rahul Singh
9. Ricci F, Rokach L, Shapira B. Introduction to recommender systems handbook. by Springer US; (2011). P.1–35.
10. Tell me who you are and I will tell you where to go: use of travel personalities in destination recommendation systems - Gretzel, Ulrike & Mitsche, Nicole & Hwang, Yeong-Hyeon & Fesenmaier, D.R.. (2004)
11. New Recommendation Techniques for Multi-Criteria Rating Systems Gediminas Adomavicius gedas@umn.edu youngok Kwon ykwon@csom.umn.edu
12. Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho.
13. A customized real time restaurant recommendation system by Rui Jiang
14. Https://pdfs.semanticscholar.org/5c7a/8f1f630428e7a471e831dde6514e91721492.pdf
15. Https://pdfs.semanticscholar.org/32fc/01833f22364ff6520cf10c2168da36f47beb.pdf
16. Building a Recommendation System with R by Suresh K. Gorakala and Michele Usuelli
17. Predicting Yelp Ratings Using User Friendship Network Information by Wenqing Yang, Yuan Yuan, Nan Zhang (2015)