Recommender Systems – Yelp Kaggle Dataset

By Team Name: Sigma

Team Members:

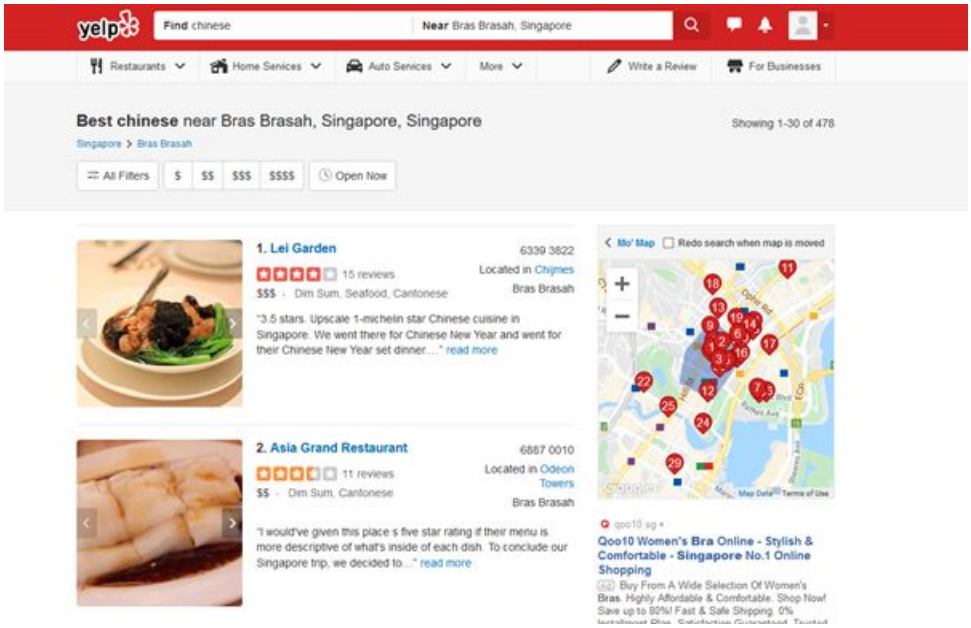
1. **Lim Wei Yang Jerome**
2. **Lee Kuo Ping**
3. **Terry Lin @ Thwin Htoo**
4. **Dinakar Mitte**

Business Problem

A recommender system is an algorithm aimed at suggesting relevant and accurate items to users by filtering through a huge pool of information base. Recommendation engines discover data patterns in the data set by learning consumer choices and produces the outcomes that co-relates to their needs and interests.

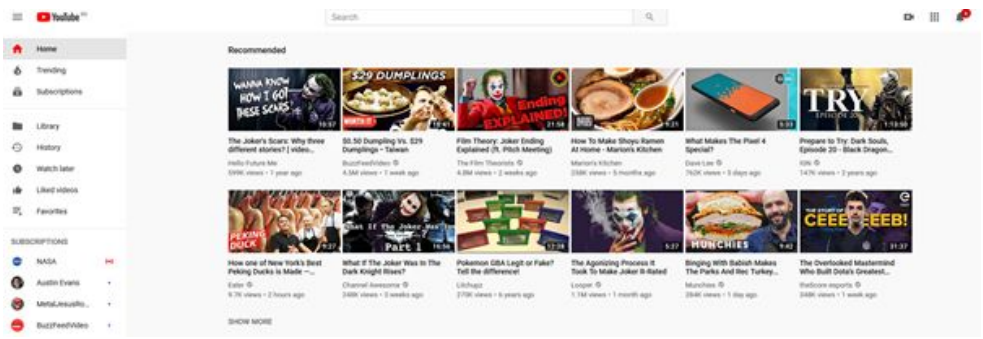
Nowadays, recommender systems are seen in a wide variety of business including web services and e-commerce. Examples include but are most definitely not limited to, Amazon, Netflix and YouTube. In this project, we have decided to tackle the Yelp recommendation engine as our business problem. Yelp is a popular online directory for discovering local businesses ranging from bars, restaurants, and cafes to hairdressers, spas, and gas stations. Users can sign up for a Yelp account on their website in order to post reviews for a particular place they visited to help others decide whether it is a good place to visit. Furthermore, users can also give ratings in the form of stars to represent how much they liked or disliked a place. The figure below shows a typical search result in Yelp.

Like all the other major online review services such as TripAdvisor, Yelp already has an inbuilt recommendation software it deploys whenever a user searches for a particular keyword. In the figure below, when the keyword “Chinese” is used, Yelp displays a list of recommended Chinese restaurants located nearby. Yelp even shows a map that can be zoomed in or out that shows where all these restaurants are located.



From this rudimentary exploration of the Yelp platform, Yelp seems to prioritize location over restaurant ratings when listing the recommended places. Scrolling down the list of recommended restaurants proves this as restaurants that have a higher number of stars are listed lower than restaurants with a lower number of stars even though they both serve “Chinese” food. While this method of recommending and listing restaurants could be a part of Yelp’s business strategy, we feel that there are ways that this business model could improve.

Our team believes that Yelp lacks a personal touch when it comes to recommending places to its users. We feel that there could be a better way to add a personal touch to the restaurants that Yelp recommends which would make Yelp more engaging and dynamic to its users. One of the ways this could be incorporated is through a personalized list of recommended restaurants which Yelp could show whenever a user logs onto Yelp. One company that does this in a very elegant manner is YouTube. As seen from the figure below, whenever a user logs on to YouTube, a recommended list of videos is presented at the top based on the user’s watching history as well as preferences. In contrast to this, in the next figure, Yelp presents a somewhat barren page of Recent Collections that does not seem to recommend any restaurants and fails to engage the user.



Our team believes that by leveraging on a user's past history of restaurants visited as well as his or her preferences in food, Yelp can build a recommendation system similar to that of Youtube's. Not only would this help to promote restaurants that are well reviewed, it would also encourage users to visit the site often after a positive experience, thereby increasing website traffic.

In addition, the group also envisions the recommender system built to be applicable to both existing users and new users. However, recommending restaurants to new users is slightly more complicated since the 'cold-start' problem makes it impossible for the recommender system to make use of information of the new user that it does not already have. Nonetheless, the team worked around this problem by mapping the new user to an existing user based on both the cuisine as well as the keywords that is input by the user. This will be further discussed later on.

Dataset

The dataset used can be found from https://www.kaggle.com/yelp-dataset/yelp-dataset. It was originally put together for the Yelp Dataset Challenge for students to conduct research analysis on Yelp's data and share their discoveries.

There are 5 different files that include business information, user profiles, reviews, check in and tip data. In total, there are 5.2 million reviews provided on 174,000 businesses. After studying the data in each file carefully, the group found that only the reviews file and the business information file was relevant in terms of building the recommender system.

The business information file is grouped by unique business entities and consists of the following information. The bolded column names denote the subset of columns that were used to build the final recommender system:

- ***Business_id*****,** which denotes the unique business identity number of an institution
- ***Name***, which denotes the name of the business institution (there can be multiple names of the same business institution since the same restaurant may have multiple branches across the country)
- *Neighborhood,* which denotes the neighborhood of the business that it is situated in
- ***Address****,* which denotes the address of the business that it is situated in
- *City,* which denotes the city of the business that it is situated in
- ***State****,* which denotes the state of the business that it is situated in
- *Postal code,* which denotes the postal code of the business entity
- *Longitude & Latitude,* which denotes longitude and latitude of the business entity
- ***Stars****,* which denotes the simple average rating by users of that business entity
- *Reviews Count,* which denote the number of reviews on this business entity
- *Is_open,* which denotes if this business entity is open
- ***Categories****,* which denotes the categories that this business entity belongs to

The reviews file is grouped by unique reviews of a user of the respective business entity that he or she has visited. It consists of the following information. Again, the bolded column names denote the subset of columns that were used to build the final recommender system:

- *Review_id,* which denotes the unique review identity number

- ***User_id**,* which denotes the user identity number that wrote this review
- ***Business_id**,* which denotes the unique business identity number that the review is referring to (since there can be multiple restaurant with similar names as described above)
- ***Stars**,* which denotes the number of stars that was given by this user of this business entity for this review
- *Date,* which denotes the date of which this review was written and submitted
- ***Text**,* which is the actual textual information of this review
- *Userful, Cool, Funny,* which denotes the number of votes by other users on whether this review was useful, cool or funny

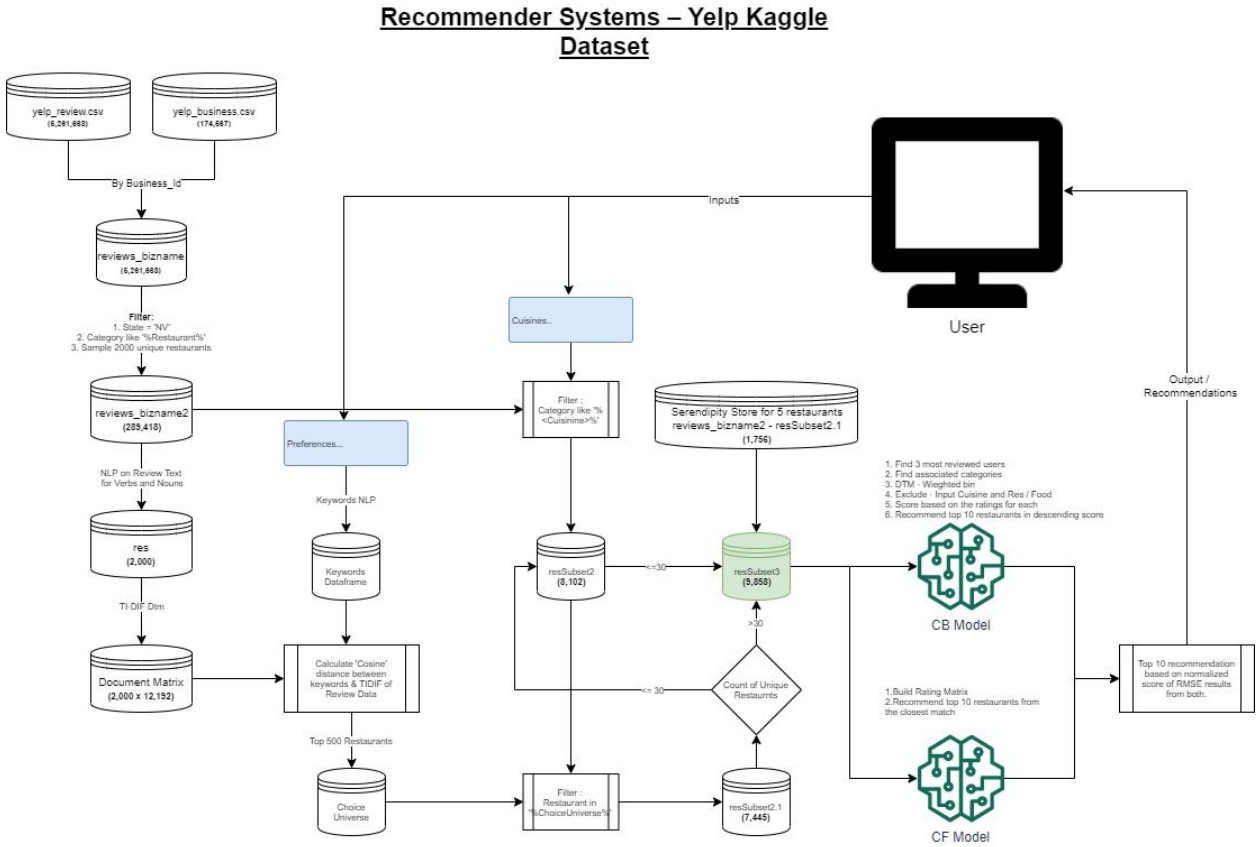
Due to the size of the dataset as well as the variety of different businesses that Yelp allows ratings for, our group decided that we would only focus on restaurants and food establishments as these places are what most users would use Yelp for.

End Product

The group planned to allow the user to interact with the recommender system in the following manner:

- The user will key in the cuisine that he or she is interested in
- The user will key in keywords about the food or restaurant that he or she wants to eat. This will give more detail to the recommender system with regard to which restaurant to recommend
- Two sets of recommendations (hybrid) will be generated behind the scenes (One using Content-Based method and the other using the Collaborative-Filtering method)
- Both sets of recommendations will be aggregated based on the respective weights (calculated using RMSEs through model evaluation) to produce a final set of top 10 recommendations for the user.

DFD

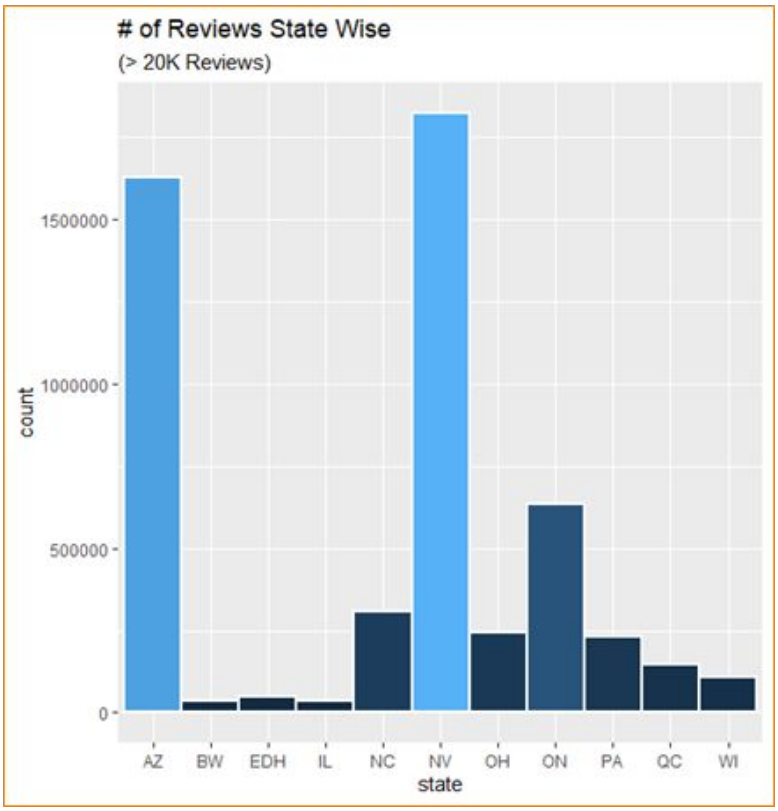


Dataset Preprocessing

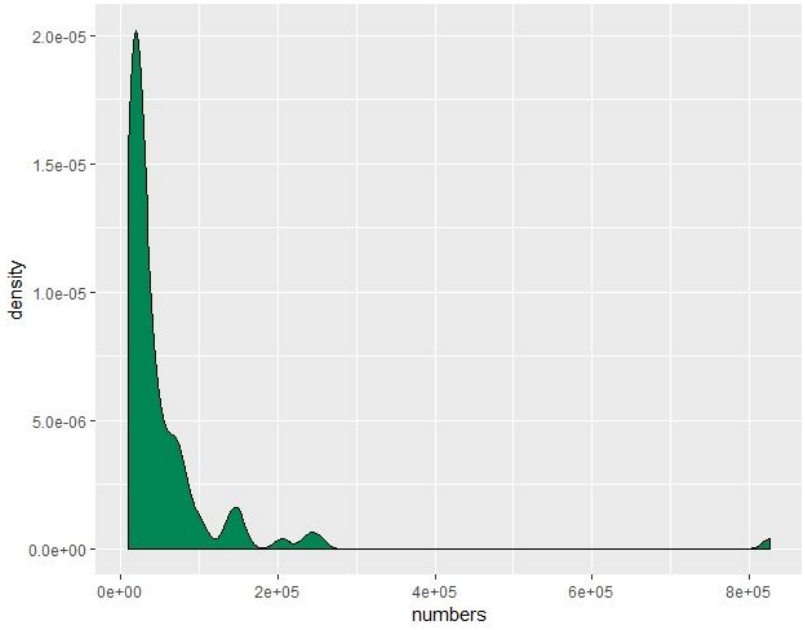
Due to the size of the full dataset and the machine capacity constraints that the group faces, it is not feasible to build a recommender system using all 5 million reviews of 174,000 businesses. The group filtered down the dataset based on the following criteria.

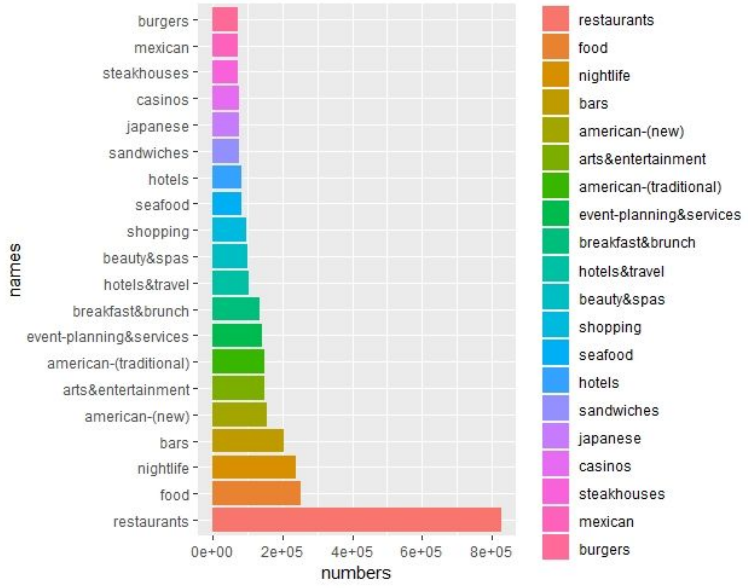
- The group has decided to focus on the state which businesses have the most reviews on, this ensures that there is sufficient data to build a robust recommender system. Exploration was therefore done to count the number of reviews by each state. It was found that the state of Nevada (state == “NV”) had the most number of reviews. The group based off this simple criteria to filter the original dataset and effectively reduced

the reviews dataset from 5 million observations to about 1.75 million observations. Nonetheless, this number was still large.



- Next, the group decided to study the categories of these business entities. Since a business entity can belong to different categories, the group did some data exploration to see which categories occurred the most out of these businesses that were rated. The categories column, which was delimited by a semicolon sign, was processed and cleaned using text-mining techniques and converted into document-term matrix. It was found that there are a total of 1161 categories. Using the density plot (of the top 100 categories), it was found that there is a large number of categories that only occur only on a few instances. Additionally, a barplot (of the first 20 categories) showed that the category ‘Restaurants’ occurred most frequently. The team decided to focus on businesses which categories consisted of ‘Restaurants’.





- After all these filtering, the group still had more than 800,000 reviews of more than 5000 restaurants. This was still too big. A final reduction was done by randomly selecting 2000 restaurants out of this subset of restaurants. The group has chosen to choose these final subset of restaurants randomly as opposed to other measures such as selecting the top 2000 restaurants that were most reviewed or rated; this is to ensure some degree of serendipity when

recommending products to users of the recommender system. Otherwise, only the most popular restaurants would be recommended.

- After this final filtering rule, there were 237 categories that were still present. At this stage, 2 matrices are constructed.
    - For exploration, the TF-IDF weighting function was used to look at which categories are popular among the restaurants. This weighting was used so that categories such as ‘Restaurants’ and ‘Food’ which were present in almost every single business (since the dataset was filtered by this rule) would be removed from the word-cloud. A word-cloud of these categories revealed the following:

    - The remaining 235 categories also serves as the features for each restaurant. Therefore, a second Document Term matrix (same dimensions of 2000 rows by 235 columns) was constructed using the binary-weighting function, where any restaurant that belonged to any one of these categories are labelled as 1. Otherwise they will be labelled as 0.

These features will be used to build the Content-Based Recommender System later on.

The final reviews dataset consisted of about 290,000 reviews over 2000 restaurants, and has approximately 156,000 unique reviewers. The names of these businesses were extracted by merging the reviews dataset with the business dataset using the unique *business_id* index. This final reviews dataset (`reviews_bizname2` is used during the choice universe filtering process as well as modelling process later on.

Natural Language Processing

An additional data-table is created using the reviews column of `reviews_bizname2`. Natural Language Processing was then done using the CoreNLP package on the 290,000 reviews in the *text* column of the reviews dataset. In this step, the nouns and verbs of the reviews are extracted, which gives us more insight into the more specific elements that made the restaurant experience for the user to be good or bad. This was done as follows:

- The unique list of 2000 restaurants are chosen
- For each restaurant, we filter out the reviews that pertains to that restaurant and extract lemmatized the nouns and verbs of that review.
- We convert all text into lower case
- We remove any predefined stop-words in that bag of words
- We output the list of unique words in this bag of words and assign it to this restaurant
- After the entire process, we get a 2-column data-table with 2000 rows. The first column denotes the name of this restaurant, and the second column denotes the bag of words that is related to this restaurant

Any text that does not belong to the “ACSII” encoding in the second column is coerced to a blank space. After which, this data-table is then converted into a Document Term Matrix with a binary-weighting function. The binary-weighting function were used instead of other weight functions because this set of words was going to be used for a keyword string matching against the words that the user inputs.

The group found that even after processing the words during every iteration of verb and noun extraction, there are still more than 70,000 unique words in this document term matrix. Sparse

terms are removed up to a threshold of 0.995 sparsity level. By applying this rule, the number of unique terms and columns is reduced to approximately 12,500. This final NLP data-table (`dt`) is used during the second part of the choice-universe filtering process.

Choice Universe Filtering

The first part of filtering the choice of the restaurants to recommend to the user is based on the category or cuisine input that the user has given. The group filters out the restaurants that were not relevant to `reviews_bizname2` by searching for the cuisine in the categories column in the data-table. This subset is then saved as `resSubset2`.

Independently, the group also created a subset of restaurants using the keywords input by the user. These keywords are converted into a one-hot vector of the same dimensions as `dt.` The similarity of these words with the 2000 restaurants in `dt` is determined using *cosine distance*. Based on these keywords, a subset of the closest n (this may be defined by the user) restaurants out of 2000 restaurants is returned. Unlike the first part of this filtering process, there is no limit of words that the user can input here. If a keyword is not found in the bag of words universe consisting of the ~12,500 words defined in the NLP process above, that word will simply be discarded. Also, unlike the first part where the cuisine has to be a sample from the categories column, there is no such restriction here. The purpose of this step is simple - to select the closest n-number of restaurants simply based on the free-text that is input by the user at the beginning.

The final part of the Choice Universe Filtering process is to find the subset of restaurants that are occuring in both subsets defined above. The final subset of these 2 subsets is named `resSubset2.1.` To ensure that there are sufficient number of restaurants to recommend from, the group incorporated an additional control, whereby if the number of restaurants in `resSubset2.1` is less than 30, then the restaurants generated from the keywords would not be considered. Otherwise, if both keyword-restaurants and cuisine-restaurants are included, this final subset of restaurants essentially take into account the unique food elements that the user is looking for as well as the cuisine of the restaurant that the user wants. This final subset, either `resSubset2` or `resSubset2.1` is finally renamed to `resSubset3` and used in the subsequent modelling processes.

Content-Based Recommendation

Data

The Content based recommendation model is built using the features of the restaurants in the choice universe stored in `resSubset3.` The features of each restaurant are used to determine the underlying things that the user really likes. Since the universe of restaurants share the same dimension of features, the model will then be able to recommend restaurants that has features which the user ranks highly of.

Methodology

To work around the cold start problem, the new user is mapped to 3 existing users so that the content based model may be applied. This is done looking for the top 3 existing users with the most number of ratings in `resSubset3`. These existing users are hence deemed as 'experts' in the type of food that the new user wants. This is feasible because `resSubset3` was essentially derived using the inputs of the new user. As such, the type of food that is recommended to these existing users based on the features that he or she enjoys, may be extended to that of the new user, whom we do not have any information about otherwise.

This specific 'experts' user-feature matrix that is a subset of `resSubset3` will have dimensions (n, k) where n is the total number of restaurants that has been rated, and k is the number of features (after reducing common features such as 'food', 'restaurant' and the respective cuisine that the dataset was already filtered by). In this case, the category tags assigned to restaurants are used as feature representation of the restaurants thanks to the domain expertise of the Yelp community that produced the category tagging. Indeed, a set of community-curated categories offer a good representation of each restaurant's feature since they continuously being revised and maintained, and in turn offers us a representation of the restaurants' feature.

These features are one-hot encoded where if this particular category exists in the *categories* column of the restaurant, it will be denoted as '1', otherwise it will be denoted as '0'. The respective ratings of these restaurants that the 'expert' has given are also retrieved. An example of a matrix is shown as follows:

	name	restaurants	thai	chinese	food	vietnamese	bars	desserts	nightlife	wine-bars	salad	soup	ratings
1:	Lao Thai Kitchen	1	1	0	0	0	0	0	0	0	0	0	3
2:	Gata Thai Cuisine	1	1	1	1	0	0	0	0	0	0	0	4
3:	Sun's Thai Food & Jerky	1	1	0	1	0	0	0	0	0	0	0	4
4:	Marnee Thai Restaurant	1	1	0	0	0	0	0	0	0	0	0	4
5:	MK Thai Cuisine	1	1	0	0	0	0	0	0	0	0	0	4
6:	Pho So 1	1	1	0	0	1	0	0	0	0	0	0	4
7:	Chada Street	1	1	0	1	0	1	1	1	1	0	0	4
8:	Surang's Thai Kitchen	1	1	0	0	0	0	0	0	0	1	1	3
9:	Mix Zone Cafe	1	1	0	0	0	0	0	0	0	0	0	3
10:	Penn's Thai House	1	1	0	1	0	0	0	0	0	0	0	4

These scores that are multiplied by the binary weighted features of these restaurants. The resulting matrix is shows the implicit scores given to each feature based on the overall rating that the user has given to the restaurant.

	name	chinese	vietnamese	bars	desserts	nightlife	wine-bars	salad	soup	ratings
1:	Lao Thai Kitchen	0	0	0	0	0	0	0	0	3
2:	Gata Thai Cuisine	4	0	0	0	0	0	0	0	4
3:	Sun's Thai Food & Jerky	0	0	0	0	0	0	0	0	4
4:	Marnee Thai Restaurant	0	0	0	0	0	0	0	0	4
5:	MK Thai Cuisine	0	0	0	0	0	0	0	0	4
6:	Pho So 1	0	4	0	0	0	0	0	0	4
7:	Chada Street	0	0	4	4	4	4	0	0	4
8:	Surang's Thai Kitchen	0	0	0	0	0	0	3	3	3
9:	Mix Zone Cafe	0	0	0	0	0	0	0	0	3
10:	Penn's Thai House	0	0	0	0	0	0	0	0	4

These feature scores of the n-restaurants rated are summed up column-wise and normalized. The resulting vector represents the aggregated normalized score that the ‘expert’ gives to these features of the restaurants that he has visited and rated.

chinese	vietnamese	bars	desserts	nightlife	wine-bars	salad	soup
0.1333333	0.1333333	0.1333333	0.1333333	0.1333333	0.1333333	0.1000000	0.1000000

Output

Finally, from the choice-universe in `resSubset3`, the values of these vectors are multiplied against the binary-weighted features in the choice universe of restaurants. Each restaurant’s feature scores are then summed up and sorted to give the top restaurants that the existing users (and therefore new user) would like. The scores and Restaurants of this model are then saved to be further used with that of the Collaborative Filtering Model to return the final set of restaurants to be recommended to the user.

	Restaurant	PredScore	City	Address
1	Chada Street	0.5333333	Las Vegas	3839 Spring Mountain Rd
2	Siamese Bistro	0.2333333	Las Vegas	7835 S Rainbow Blvd, Ste 3
3	Surang's Thai Kitchen	0.2000000	Las Vegas	5455 S Fort Apache Rd, Ste 105
4	Jacky Chan	0.1333333	Las Vegas	8544 Blue Diamond Rd, Ste 100
5	Pho So 1	0.1333333	Las Vegas	4745 Spring Mountain Rd
6	5 Dollar Cafe	0.1333333	Las Vegas	2200 Las Vegas Blvd S
7	Gata Thai Cuisine	0.1333333	Henderson	35 S Gibson Rd
8	SEA: The Thai Experience	0.1333333	Las Vegas	3645 Las Vegas Blvd S
9	Asian Wok	0.1333333	Las Vegas	6515 N Buffalo Dr
10	Kunchorn Thai Restaurant	0.0000000	Las Vegas	3430 E Trpicana Ave, Ste 1

Evaluation

The accuracy and performance of the Content Based Recommender Model is evaluated as follows. Firstly, the top 100 users based on number of restaurants reviewed were identified using `Reviews_bizname2`, which is our filtered dataset containing all the reviews from the random 2000 restaurants that were sampled. From these top 100 users, we then obtained all the reviews that they had written. These 100 users contributed to 8118 reviews in total in `Reviews_bizname2`.

Once this once done, the categories column was cleaned up by removing unwanted symbols, spaces and repeating terms such as “Restaurant” and “Food”. We then built the corpus using the cleaned categories column and then applied a document term matrix to each of the category terms. This produced a matrix of 8118 rows with 218 different categories as features. A sample of the first 10 rows and the first 10 columns can be found below.

Docs	african	american-(new)	bars	cocktail-bars	nightlife	seafood	steakhouses	asian-fusion	chinese	japanese
1	1	1	1	1	1	1	1	0	0	0
2	0	0	0	0	0	0	0	1	1	1
3	0	0	0	0	0	0	0	1	1	1
4	0	0	0	0	0	0	0	1	1	1
5	0	0	0	0	0	0	0	1	1	1
6	0	0	0	0	0	0	0	1	1	1
7	0	0	0	0	0	0	0	1	1	1
8	0	0	0	0	0	0	0	1	1	1
9	0	0	0	0	0	0	0	1	1	1
10	0	0	0	0	0	0	0	0	1	0

From here, we split the document term matrix in a 90:10 ratio based on the 100 users in order to obtain our training and testing datasets. On our training dataset, we apply the same methodology used before when building our content based model by multiplying the terms by

the binary weighted features of these restaurants and obtaining the normalized features scores by column. We end up with a total of 218 normalized scores, one for each category.
With the normalized scores obtained from the test set, we can then apply a dot product to the training set in order to obtain the predicted score for each category for each review. Below is a sample of the scoring of each category for each review on the test set.

	african	american-(new)	bars	cocktail-bars	nightlife
11	0	4.202652e-05	0.0000000000	0.00000000	0.000000000
23	0	0.000000e+00	0.0000000000	0.00000000	0.000000000
53	0	0.000000e+00	0.0000000000	0.00000000	0.000000000
54	0	0.000000e+00	0.0000000000	0.00000000	0.000000000
58	0	1.418395e-03	0.0000000000	0.00000000	0.000000000
70	0	0.000000e+00	0.0002521591	0.02309357	0.000199626
74	0	0.000000e+00	0.0000000000	0.00000000	0.000000000
77	0	0.000000e+00	0.0000000000	0.00000000	0.000000000
85	0	6.619177e-04	0.0000000000	0.00000000	0.000000000
86	0	6.619177e-04	0.0000000000	0.00000000	0.000000000

Once this has been completed, we can then obtain the predicted score for each restaurant by summing up the scores across the columns. After which, we normalized the scores to fit within the accepted rating of 1 to 5 stars in order to compare the difference between the predicted rating and actual user ratings. Finally, we computed the RMSE and MAE of our test dataset to obtain values of 1.92 and 1.95 respectively.

Collaborative-Filtering Recommendations

Data

The collaborative filter model is build upon the choice universe stored in `resSubset3`, which was filtered by cuisine choice and associated keywords input by the user. This dataset is further reduced to retain only the necessary columns for this component of the recommender system. The reduced columns are:

```
"user"    "item"    "ratings" "city"    "address" "type"
```

The columns `user` and `item` are converted to factors to prepare for subsequent model training.

Library

The collaborative filtering model is build using recosystem library. A model-based approach using Matrix factorization was selected and tuned using the Alternating Least Square (ALS) algorithm.

Methodology

The methodology was chosen as there are multiple characteristics that can tie users and restaurants together. These characteristics are also known as latent factors, and the ALS algorithm adjusts in an alternating steps the users and restaurants latent factors to arrive at the rating matrix with the lowest error.

The `tune()` function is utilised to select the best tuning parameters with the minimum cross validated loss. The suggested candidate parameters with the minimum loss are:

mindim	10
mincostp_l1	0
mincostp_l2	0.1
mincostq_l1	0
mincostq_l2	0.1
minlrate	0.08

Evaluation

Before the collaborative filtering model was carried out on the new user, we first evaluated the model based on existing users. `reviews_bizname2` was first split into training and testing sets in 90:10 ratio. The training and testing sets are then converted into a data format using `data_memory()` that can be read by recosystem library.

Once this was done, the recosystem was initiated and the model was trained on the training dataset. The optimal parameters were obtained via `tune()` and the ideal number of latent variables were found to be 10. After the model has been trained, it is then applied onto the testing set in order to predict rating scores. Once the predicted rating scores have been obtained, they are then normalized to fit within the acceptable rating scores of 1 to 5. Finally, we can then proceed to calculate the RMSE score of 0.779 and MAE score of 1.51 respectively.

Results

The parameters are applied the model to predict the ratings by the expert user for all the unique restaurants in `resSubset3`, of which there are 50. The top 10 from this output is selected. This represents the restaurants that are predicted to be most highly rated by the expert user, whom we used as a proxy for our new user.

	Restaurant	PredScore	City	Address
1	Asian Wok	4.2266541	Las Vegas	6515 N Buffalo Dr
2	Gata Thai Cuisine	4.1555972	Henderson	35 S Gibson Rd
3	Pho So 1	3.6759093	Las Vegas	4745 Spring Mountain Rd
4	Chada Street	3.6311951	Las Vegas	3839 Spring Mountain Rd
5	Mix Zone Cafe	3.6057384	Las Vegas	2202 W Charleston Blvd, Ste 5
6	Archi's Thai Bistro	3.5332868	Las Vegas	6345 S Rainbow Blvd, Ste 100
7	Surang's Thai Kitchen	3.3082180	Las Vegas	5455 S Fort Apache Rd, Ste 105
8	5 Dollar Cafe	3.1076317	Las Vegas	2200 Las Vegas Blvd S
9	MK Thai Cuisine	3.0086646	Las Vegas	4420 E Charleston Blvd, Ste 5 & 6
10	Sister's Oriental Market & Video	3.0001948	Las Vegas	1732 Fremont St

Hybrid Model & Example

Multiple iterations of evaluations for various inputs suggest that the Collaborative Filter recommender model has better RMSE than the Content Based recommender.

	RMSE.CB	RSME.CF	MAE.CB	MAE.CF	Cuisine	keywords
1:	1.692528	0.8869093	1.775042	1.523215	thai	rice pineapple fishcake seafood green curry
2:	1.681367	0.7922279	1.764517	1.500385	pizza	kids friendly
3:	1.709376	1.0154736	1.784571	1.587495	indian	tandoor paneer family kids friendly

The errors of these different models are then used to calculate the weights for the final combined hybrid model. These weights are calculated by taking the complement of the proportion of each respective model's error over the total error of both models.

```
errorMat <- data.table(RMSE.CB = RMSE.CB, RMSE.CF = RMSE.CF, MAE.CB = MAE.CB, MAE.CF = MAE.CF)
cfWeight = 1-(errorMat$RMSE.CF/sum(c(errorMat$RMSE.CB, errorMat$RMSE.CF)))
cbWeight = 1-(errorMat$RMSE.CB/sum(c(errorMat$RMSE.CB, errorMat$RMSE.CF)))
```

Splitting the full dataset into training and test sets and conducting the relevant evaluation, it was found that the optimized weights for CF and CB recommender systems are 0.7115431 and 0.2884539, respectively. This reflects the better accuracy from the CF model as discussed above. It should be noted that the group has decided to use these static weights, instead of changing them every time a different cuisine or set of keywords are input because the lack of sample size for a specific input would mean the resulting weight is not statistically significant. As such, these static weights are applied on the choice universe that is defined dynamically.

Finally, the ranked results of the choice universe of both models are adjusted with its respective weights and presented them as final recommendation list using a simple weighted average calculation methodology. This error adjusted hybrid model will then sort the combined rankings and output the final set of restaurants to recommend to the user.

```
finalcombined$Final.Rank <- cfWeight*finalcombined$norm.x + cbWeight*finalcombined$norm.y
```

Example:

Inputs from user:

```
cuisine <- "chicken-wings"
keywords <- c("rice", "beancurd", "fishcake", "seafood", "waiter", "friendly") %>% tolower
```

Recommendations from the system (ranked in ascending order):

	Restaurant	Final.Rank	City	Address
1	Applebee's Neighborhood Grill & Bar	1.288451	Las Vegas	820 E Warm Springs Rd
2	Solaro	7.307616	Las Vegas	3325 Las Vegas Blvd S
3	Those Guys Pies	9.499795	Las Vegas	2916 Lake E Dr
4	Q Bistro	11.269238	Las Vegas	3400 S Jones Blvd, Ste 12
5	Original Chicken Tender	11.884512	Las Vegas	3900 Las Vegas Blvd S
6	PublicUs	11.942062	Las Vegas	1126 Fremont St
7	Rosati's Pizza	16.518958	Las Vegas	7380 Eastern Ave, Ste 110
8	Hooters Casino Hotel Las Vegas	17.230853	Las Vegas	115 E Tropicana Ave
9	Roma's Pizza	18.653587	Las Vegas	605 E Twain Ave, Ste 1
10	Raising Cane's	21.077022	Las Vegas	1120 E Flamingo Rd

Conclusion

In conclusion, our group has successfully implemented both collaborative filtering and content based models to build our recommender system. We were also able to overcome the “Cold Start” problem that plagues most collaborative filtering approaches by mapping the new user to three current users that already exist and provide a recommendation to the new user the moment he or she signs up for a Yelp account. We also addressed serendipity issues by randomly sampling 2000 restaurants instead of only using the top rated reviews. This would help to introduce more variety in the final recommended list.

Nonetheless, there are still improvements that can be made to our final model. For example, since only a random selection of 2000 restaurants were made in the end, it is possible that there are certain keywords or features that might not be within the 290,000 reviews. Hence, it is possible for a new user to enter a cuisine or keywords that could not be mapped to anything. This would end up just recommending to the user the top most rated restaurants within the dataset. Another compromise that we had to make for this project was to use user reviews for the content-based modelling approach. Traditionally, the best implementation of content-based modelling would be to make use of product descriptions to come up with features for the document term matrix. However, as we did not have access to this dataset, we had to settle for using user reviews instead.