BT5153 Group 10 Project
Review Quality Based Recommendation for Yelp

Bryce Lee Zheng Hui (A0124294H) [1] **Liu Yue (A0218913A)** [2] **Mai Peihua (A0218951Y)** [3]
Tang Yin (A0218854W) [4] **Wu Lingyun (A0218846U)** [5]

Github Repository:

https://github.com/Whalelingyun/BT5153-Group10-yelp

1 Introduction

With the convenience brought by Internet and the massive amount of available information, people tend to find a review of a business before they decide to visit it. As the most widely used online review site in the North America, Yelp enables customers to read and write reviews to offer their insights based on their experience with the business. In addition to sharing their own opinions, visitors can also search for a business by name, by category, or by location and can view the reviews posted by others online.

Containing lots of information of businesses, users, reviews and interactions among them, Yelp attempts to solve common user problems like "What should I eat today?" by presenting available options to users. However, Yelp only provides visitors with a holistic view about a business. On the business's main page, an average rating score is shown under the name of the business, and only 5 reviews among thousands of reviews are selected and presented on the page. Since the quality level of user-created-contents is not uniform, it is tedious and error-prone for Yelp to filter and rank useful information. Moreover, since every customer has his/her own personal preference, it is likely that the businesses shown on the default main page are unsuitable. This means that finding useful reviews and satisfactory businesses might be time-consuming for Yelp visitors, which can lead to decreased user willingness to use Yelp. Hence, our project aims to bring business benefits and commercial usefulness to both Yelp and its users.

2 Project Objectives

Our primary objective is to develop a personalized recommendation system based on review quality. The quality of reviews might be associated with the rating's authenticity, and thus can be put into the recommendation framework to improve the performance of rating prediction. (Raghavan et al., 2012) To build such recommendation system, we proposed a two-stage model. In the first stage, we predicted the helpfulness of the user's review through classification models. In the second stage, the estimated probability was used to develop our recommendation system.

The first stage focused on evaluating new reviews to determine if it is of high quality, based on its text and non-text features. Though the helpfulness of the reviews can be evaluated by the votes they received, for more recent reviews without sufficient feedback, the number of votes might not reflect the real quality of the reviews. Therefore, a classification model is entailed to predict the helpfulness of the latest reviews. Besides being pushed into the pipeline as user metadata and used to build our recommendation system, the helpfulness prediction will help platform prioritise these reviews to be shown to users.

In the second stage, we focused on building a personalized recommendation system which will show users businesses and reviews that are more aligned with their taste and preferences. This would aid users in making quicker and better decisions. We proposed a pipeline which incorporates contextual and text sentiments, along with other non-textual features, into decision-making. With an improvement in performance and tailor-fitted journey, we aim to help reinforce brand loyalty of users and further boost Yelp's revenue from the perspective of higher user usage.

3 Datasets

3.1 Data

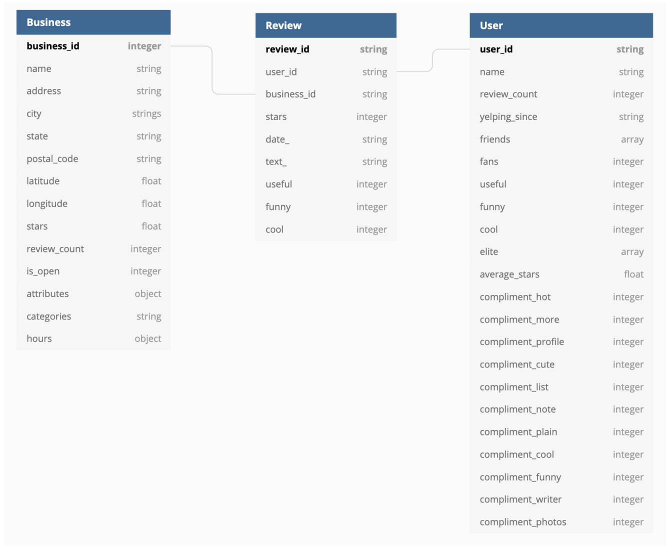
We used the Yelp Open Dataset (source: https://www.yelp.com/dataset) from October 2004 to February 2020 downloaded from yelp website, which is composed of 5 json files —- Business, Review, User, Checkin and Tip. We chose 3 sub datasets covering information of businesses, reviews and users. Checkin table was eliminated since it is not essential for our project, and Tip table was not used because it's a subset of Review table.

Figure 1 illustrates the relationship across three Yelp tables. Feature linkages among tables are indicated by grey lines.

Source Table	Variable Name	Data Type	Variable Description
Business	Stars	Float	Star rating, rounded to half-stars
Business	is_ Open	Integer	0 or 1 for closed or open
Business	Attributes	Object	Business Attributes to values
Business	Categories	String	An array of strings of business categories
Review, User	Useful(Funny/Cool)	Integer	Number of useful (funny/cool) votes received
User	Friends	Array of strings	An array of the user's friend as user_ids
User	Fans	Integer	Number of fans the user has
User	Elite	Array of integers	The years the user was elite
User	Compliment_hot (more/profile/cute,etc)	Integer	Number of hot(more/profile/cute) compliments received by the user

Table 1. Yelp Dataset Description

Business table contains 14 variables with a total of 209,393 observations, including location data, attributes and categories, etc. Review table contains 9 variables with 8,021,122 review instances including review text data, the user_id that wrote the review and the business_id the review was written for. User table contains 22 variables with 1,968,703 users' data, which includes the user's friend mapping and all the metadata associated with the user. Table 1 explains some non-common features in Yelp dataset. By joining 3 tables, there are a total of 43 variables and 8,021,122 records.



Business	
business_id	**integer**
name	string
address	string
city	strings
state	string
postal_code	string
latitude	float
longitude	float
stars	float
review_count	integer
is_open	integer
attributes	object
categories	string
hours	object

Review	
review_id	**string**
user_id	string
business_id	string
stars	integer
date_	string
text_	string
useful	integer
funny	integer
cool	integer

User	
user_id	**string**
name	string
review_count	integer
yelping_since	string
friends	array
fans	integer
useful	integer
funny	integer
cool	integer
elite	array
average_stars	float
compliment_hot	integer
compliment_more	integer
compliment_profile	integer
compliment_cute	integer
compliment_list	integer
compliment_note	integer
compliment_plain	integer
compliment_cool	integer
compliment_funny	integer
compliment_writer	integer
compliment_photos	integer

Figure 1. Overall Dataset Structure

However, 8 million records was too memory intensive and time consuming for our future modelling, we reduced the number by filtering. Firstly, we filtered out the businesses which are closed, leaving only those that are still in operation. Next, we extracted out businesses that are tagged as "Restaurant" under categories. Lastly, we focused on the state of Nevada since the restaurants in Nevada are widely distributed and are almost evenly located as shown in Figure 2. Nevada also has the most reviews among all the states in Yelp. Finally, this left us with a total of 1,410,254 records.

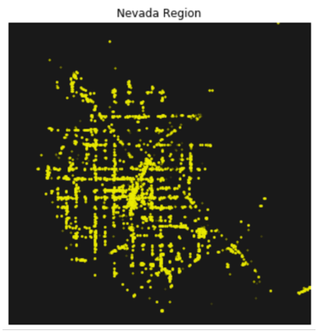


Figure 2. Lat-Long Plotting For Nevada

3.2 Exploratory Data Analysis

Before diving into model development, we conducted preliminary data exploration with our dataset. We explored the star rating distribution of both reviews and businesses in Figure 3. Most reviewers gave 5-star ratings to restaurants, while the star rating distribution of businesses is significantly left skewed, which reveals that there are a lot of high-rated restaurants in the dataset. This made our recommendation even more complicated since sorting the restaurant by star rating will be ineffective. Under this situation, business rating can serve as a guideline but is not entirely indicative of user's individual score for the business. By deep diving into reviews, Figure 4 shows the density distribution and cumulative distribution of the number of votes reviews received. As close to 80% of reviews received 2 votes, we defined reviews with votes greater than 2 as high-quality reviews later in our models.

4 Classify High-quality Reviews on Yelp

We grouped the filtered dataset by the review years, and found out that reviews posted in 2018 and 2019 occupied

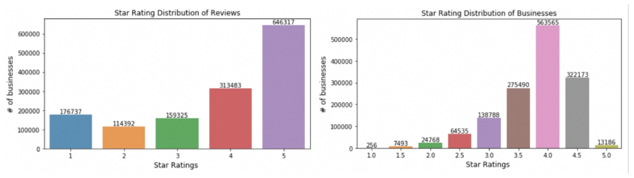


Figure 3. Star Rating Distributions

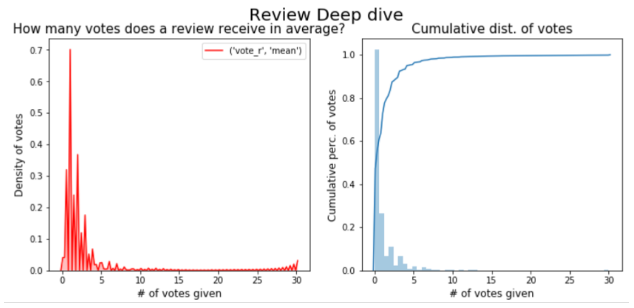


Figure 4. Review Deep Dive

the majority of the dataset, each with around 250,000 data points. Therefore, reviews and related information posted in 2018 and 2019 were selected as our training and testing sets, respectively.

4.1 Variables

4.1.1 Target Variable

Online review sites depend heavily upon the content users provide. However, it is hard to quantify and label the quality of a review. Therefore, we decided to utilise the user voting system that Yelp introduced, which allowed users to upvote each other's review if the review is "useful", "funny" or "cool". In this project, reviews are considered high-quality if total upvote count is greater than two and low-quality if upvote is fewer than or equal to two. By applying this classification approach, we obtained 216,034 low-quality reviews and 38,279 high-quality reviews in 2018, and 209,015 low-quality reviews and 37,983 high-quality reviews in 2019.

We understand that these labels are subjective and may contain bias where high-upvote reviews get more and more upvotes and low-upvote/new reviews have little attention. This issue could potentially be rectified by our classification model where reviews are recalled based on both non-text and text features.

4.1.2 Predictor Variables

Non-text Features

Figure 5 illustrates correlation between all potential predictor variables and target variable. As shown in the correlation map, the activeness of a Yelp user is highly correlated to his/her review quality. Therefore, when extracting features to classify high-quality reviews, Yelp user details and business information were taken into consideration. There are 4

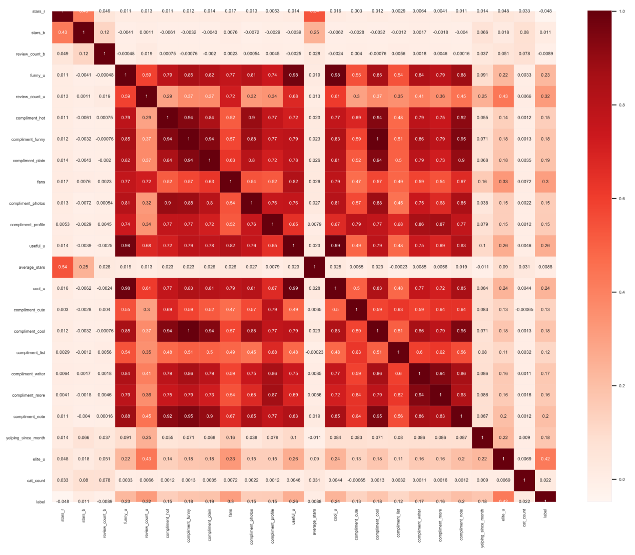


Figure 5. Correlation Map

engineered features. *Active duration* of user was calculated as number of months between review date and the first date user joined Yelp. *Elite status* of user was updated to a binary variable where the status is one if a user was rewarded an elite title at least once. *Elite count* records total number of years that a user got elite. Top 184 frequently appeared *categories* were kept in business category as we found that some businesses associate irrelevant categories with them, maybe intend to improve public exposure. Standardization was applied to all continuous variables. Some additional examples of non-text features are:

- Cumulative upvotes that author received
- Number of funs that author has
- Average rating that business received
- Total review count of business
- Review star

Text Features

Review content is unstructured text data that requires pre-processing. Multiple text cleaning techniques such as special symbol removing, lowering, stopwords removing and punctuation removing were explored. Text pre-processing tools provided by NLTK package were heavily used in text pre-processing and text feature extraction.

After the initial pre-processing, we explored the content preliminary by using the word cloud tool to see if any specific words are frequent in each category. Word clouds of the two categories of reviews in 2018 are shown in Figure

6, suggesting that both categories have similar topics and frequent words. We observed an almost identical result for reviews in 2019 as well.

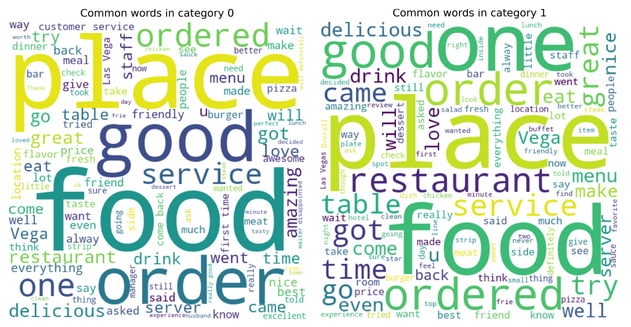


Figure 6. Word Cloud of 2018 Reviews

Beside the content, we would also like to find out if high-quality reviews and low-quality reviews have difference in sentiment. In addition, one hypothesis is that high-quality reviews have more adjective words than low-quality reviews. As a result, we extracted text features in the following categories:

- Structural features (eg. Length of review, wordcount)
- Lexical features (term-document matrix. eg. TF-IDF, Bag-of-Words)
- Syntactic features (eg. Percentage of verbs, nouns, top frequent words)
- Sentiment scores

We obtained the length and word count for each review and compared the average results of both categories. For both reviews posted in 2018 and 2019, the lengths of reviews are around 875 for high-quality reviews and 427 for low-quality reviews; the text count of low-quality reviews is around 79, while the count of high-quality reviews is about 2 times of it. Therefore, we can declare that the length and text count have obvious difference for two categories.

Sklearn's feature_extraction package was used to generate term-document matrix such as Bag of Words and TF-IDF. We applied this technique to the cleaned review content, as well as to only frequently appeared words. We also examined the POS tagging results obtained for the two categories. In both years, the percentage of each tag is almost identical (Figure 7).

In addition, to investigate if polarity and intensity of sentiment exist in each review, sentiment analysis using VADER (Valence Aware Dictionary for Sentiment Reasoning) model was deployed for reviews in both 2018 and 2019. VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media and suitable for both polarity and intensity of emotion.

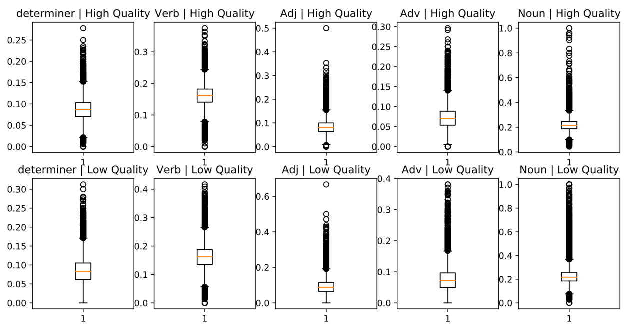


Figure 7. POS Tagging for high and low quality reviews

Since Yelp is a social media platform and reviews posted by users may contain non-standard expressions that only applicable on the Internet, we deem VADER is an appropriate tool to use in our context. The resulted sentiment scores range from -1 to 1, indicating not only the positivity and negativity of a text, but also how positive or negative the sentiment is. As we can see from Figure 8, the distributions of sentiment scores for high-quality and low-quality reviews are similar in 2018 and 2019, both having massive reviews with highly positive sentiment. As the sentiments of both categories are similar, we deem sentiment differences may not be significant for our scenario and hence excluded sentiment as one of the features for our models.

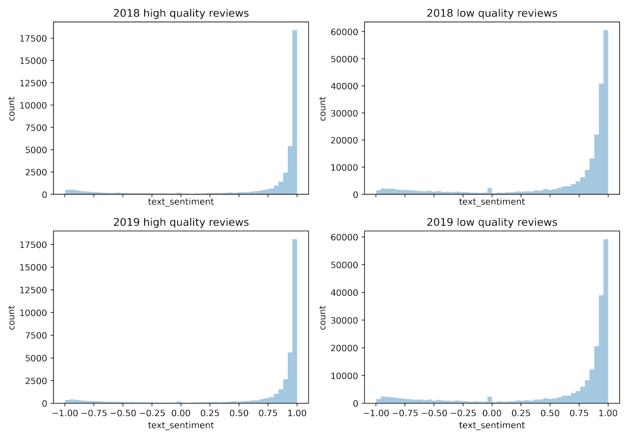


Figure 8. Sentiment Scores of Reviews in 2018 and 2019

4.2 Models

We identify this task as a supervised binary classification problem, with reviews being labelled as high-quality or low-quality depending on a predefined upvote count threshold. To effectively classify high-quality reviews, we first utilized NLP techniques to extract features from review text, before applying proper classification model on text-based features and user/business metadata features.

Common classification techniques such as Logistic Regression, KNN and Random Forest were implemented as baseline models. Hyperparameter tuning using grid search cross validation were conducted to enhance model performance. We also consider utilizing Naïve Bayes Classifier. Naïve Bayes Classifier is a probabilistic machine learning model, and it has been proven to be efficient and effective when dealing with large text content and big term-document matrix. Therefore, Naïve Bayes model was also applied to text only features for evaluation comparison.

Three boosting models — XGBoost, LightGBM and CatBoost models were used. Boosting method is essentially an ensemble method. It is an extension to the classical Decision Tree Classifier which trains models in a gradual, additive and sequential manner. By combining with previous models, the best possible next model minimizes the overall prediction error through setting target outcomes. The target outcome of each iteration is based on how much the prediction impacts on the overall prediction error. Boosting method generally has high accuracy but low explainability. XGBoost, LightGBM and CatBoost are all advanced boosting algorithm. They differentiate themselves by splits, leaf growth, missing value handling and categorical feature handling. These differences leads to slight performance delta. For example, LightGBM has the fastest training speed because it offers gradient-based one-side sampling (GOSS). Moreover, in some scenarios, CatBoost outperforms the rest because of its unique encoding for categorical features.

4.3 Classification Evaluation

As shown in Section 4.1.2, text features have similar distribution for both high-quality and low-quality reviews. Hence, we first trained all models using non-text features in training set and evaluated the testing set based on accuracy, AUC, precision, recall and F1 score. Grid search cross validation was conducted on the training set for parameter tuning. Table 2 shows the evaluation result. We observed that three boosting models – XGBoost, Light GBM and CatBoost outperform the rest. XGBoost model performed the best in terms of accuracy (0.893), AUC (0.899), recall (0.411) and F1-score (0.542). The classification model worked very well on non-text features. Activeness of user is the main contributor.

Text extracted features were then added in predictive variables. Multinomial Naïve Bayes classifier was applied to term-document matrix and serves as a baseline model. We chose the top-performing model – XGBoost and compared its performance on different combination of non-text and text features. Evaluation result is shown in Table 3. We noticed that even though there were some improvement after adding text extracted features, the improvement is very minimal, which also proves that there's no significant text difference between high quality & low quality review content. Resampling technique is explored and applied to

	Accuracy	ROC AUC	Precision	Recall	F1-score
Multinomial NB	0.872	0.688	0.630	0.401	0.490
Gaussian NB	0.871	0.771	**0.901**	0.179	0.298
Logistic Regression	0.878	0.771	0.809	0.268	0.403
KNN	0.873	0.715	0.700	0.306	0.426
Random Forest	0.884	0.844	0.827	0.310	0.451
XGBoost	**0.893**	**0.899**	0.800	**0.411**	**0.542**
Light GBM	0.891	0.893	0.841	0.363	0.507
CatBoost	0.891	0.893	0.839	0.361	0.505

Table 2. Evaluation: Non-text features

non-text & text stats features only. This is because resampling method typically works bad in high dimensional data as the linear relationship of clusters does not maintain. After applying random oversampling, recall improves to 0.74 and F1-score increases to 0.59. However, precision drops to 0.496. Precision and recall is always a tradeoff and it is up to business consideration to select the best acceptable result. In our project, since we will apply predicted high-quality review in recommendation system, we chose XGBoost with top 20 features (with highest accuracy) as our final model.

5 Customized Recommendation

The predicted review quality in the previous section was used to develop our customized recommendation model. Figure 9. is an overview of the pipeline and data flow within the system. We use a combination of content-based and collaborative filtering methods, performed sequentially, to build a hybrid recommendation system. Firstly, the content-based method is applied to generate candidate restaurants for users. Then based on the candidate pool, we predicted the rating the user will give to each restaurant through collaborative filtering method. The top N business with highest predicted rating will be shortlisted as our final recommendation.

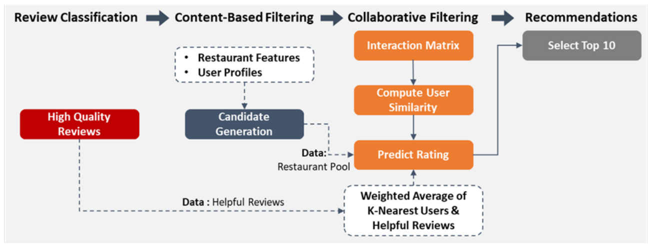


Figure 9. Recommendation System Overview

5.1 Candidate Generation

The recommendation model begun by generating a pool of restaurant candidates under content-based filtering. In this stage, we analysed text features in restaurant reviews and

	Accuracy	ROC AUC	Precision	Recall	F1-score
Multinomial NB: Text only	0.805	0.671	0.371	0.387	0.379
XGB: Non-text	0.893	0.899	**0.800**	0.411	0.542
XGB: Non-text + TextStats	0.894	**0.903**	0.786	0.432	0.557
XGB: Non-text + TextStats + **Smote**	0.894	0.900	0.777	0.438	0.560
XGB: Non-text + TextStats + **RandomOver**	0.844	0.901	0.496	**0.737**	**0.593**
XGB: Non-text +TextStats +Text	0.894	**0.903**	0.789	0.426	0.553
XGB: Top 20 features	**0.895**	**0.903**	0.790	0.430	0.557

Table 3. Final Evaluation

the categories that restaurants were being tagged with. Prior to creating Bag-of-Words (BOW) for the model, certain measures were taken to ensure that contextual meaning and sentiments would be retained as much as possible after words have been vectorized i.e., expanding contractions, retaining negative stop words. Trigrams were also captured during the CountVectorizer process.

The model ingested inputs from two different data-set; one comprising only of restaurant data selected by/for the user and the other for the rest of the historical restaurant data. Similarity matching between these two data-sets were computed using correlation similarity for two features: reviews and categories. Thus, for the user selected restaurant, two correlation similarity values (review and categories) were generated separately with the other individual restaurants. By taking the mean of these correlation similarity, we were able to utilise the averaged score to rank restaurants' similarity to the user selected restaurant, keeping in mind that a large correlation value represents higher similarity.

$$Corr.\ Sim_{mean} = \frac{Corr.Sim_c \times Corr.Sim_r}{2} \quad (1)$$

With this ranking, we select a pool of restaurant candidates according to a set of standards, passing this pool down the pipeline for collaborative filtering and rating prediction i.e., select top 50 restaurants with highest correlation similarity, which may be interpreted as restaurants being sufficiently similar, and let it pass to the next stage of processing.

5.2 Rating Prediction

We develop a collaborative filtering framework based on review quality to predict the users' unknown preference, which is denoted by the rating they will give to a specific item. The rating prediction is composed of 3 phases: (1) construct the user-restaurant interaction matrix; (2) compute pairwise user similarity; (3) predict user's rating to the shortlisted restaurant.

5.2.1 Interaction Matrix Construction

Restaurant Clustering

To address data sparsity and high dimension problem, we apply clustering algorithms to partition the restaurants. Table 4 gives the features to conduct the classification. The follow techniques are used in feature generation and engineering.

Features	Type	Description
Latitude	Coordinate	Latitude of the business.
Longitude	Coordinate	Longitude of the business.
LDA i	LAD Score	LDA score of topic i. The LDA score for business is the average of that for each review.
Cat i	Category	Category i, 1 when the restaurant belong to the category, otherwise 0.
Avg. Rating	Average Rating	Average rating of all the reviews in the training set for the business

Table 4. Features for Business Clustering

- Latent Dirichlet Allocation (LDA)

We leverage LDA topic modelling to discover the hidden topics across reviews. After removing non-alphabetic characters and deleting stop words including common words like "restaurant", we trained the LDA model on all the restaurant reviews.
One important hyperparameter for the model training is the number of topics, which is determined by Kullback-Leibler (KL) divergence and the reasonability of the word distribution. KL divergence measures the difference between words distribution in two topics with the following formula:

$$D\left(P\,||Q\right) = \sum_i P\left(i\right) log\frac{P\left(i\right)}{Q\left(i\right)} \quad (2)$$

From the formula, we know that the higher the divergence value, the more separated the two topics. We choose the topic number that would maximize the average KL divergence. From Figure 10, we noticed that LDA model with around 12 topics gives better performance as demonstrated by the larger proportion of light cells. Figure 11 shows the word cloud for two selected topics when the number of topic is 12. We clearly observe that the two topics focus on different aspects of the restaurants. The result looks sensible and we finally chose 12 as our topic number.

- Word2Vec & K-means Cluster

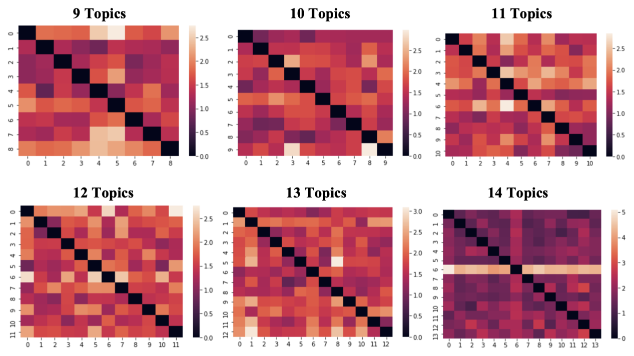


Figure 10. KL Divergence Score for 6 Different Topic Models

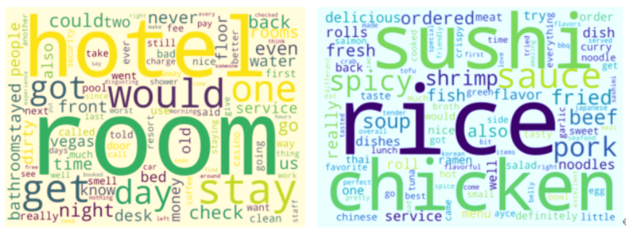


Figure 11. Word Clouds of Top 100 Keywords for Model with 12 Topics (2 Examples)

Each business belongs to one or more categories, and there are 546 categories in total. To reduce feature dimensionality, we applied Word2Vec and K-means algorithm to grouping similar categories. Specifically, for every category we vectorized the words through Google's pretrained Word2Vec and average the context vectors. Then we implemented K-means clustering on the vectorized data with manual fine-tune.
By examining the clustering result, we observed that cluster number of 250 gives reasonable output since the categories within each cluster are almost similar to others, as opposed to a lower value of cluster number that groups too many unrelated categories together. Therefore, we determined our final number of clusters as 246 and adjusted some misclassified categories manually after K-means clustering.

Our processed features are normalized and fed into three cluster models: K-means, hierarchical clustering and GMM. We determined the number of clusters for each model respectively through silhouette score. Table 5 shows our hyperparameters for three clustering models.
Based on the clustering results, we constructed the user-business preference matrix. The user's preference towards a particular business cluster is the average of their score on all businesses within that cluster.

Sentiment Analysis on Review

To evaluate user's preference on each item, we leverage both actual rating and sentiment scores computed from the reviews. Rating given by the user is a coarse evaluation of user's attitude on the business and may not reflect their

Model	Number of Cluster	Silhouette Score
K-means	250	0.193
hierarchical clustering	1300	0.244
GMM	280	0.208

Table 5. Number of Clusters and Corresponding Silhouette Score for Different Model

fine-grained opinions.(Pero and Horváth, 2013) Table 6 indicates that users might overrate or underrate the restaurants compared to their reviews.

To make use of the opinions expressed in the reviews, we integrate sentiment analysis to construct the preference matrix. We retrieved the polarity score from TextBlob for each cleaned review, and rescaled the polarity score to 1-5 based on the distribution of actual ratings.
Through opinion mining, we were able to construct two user-business preference matrices, rating matrix and opinion score matrix.

Auto Encoding

To fill in the missing values in the interaction matrix, we apply autoencoder to infer user's preference over unrated business. The hyperparameter for the model is given by figure 7.

5.2.2 User Similarity and Rating Prediction

User Similarity

We compute the pairwise user similarity based on the two matrices respectively. The cosine similarity is chosen as our metric with the following formula:

$$s_{i,j} = \frac{\vec{i} \times \vec{j}}{\left|\left|\vec{i}\right|\right| \left|\left|\vec{j}\right|\right|} \quad (3)$$

Based on the cosine similarity, we chose the 50 nearest neighbours for each user to estimate their preference score over particular items.

Review Quality

The prediction in the review quality classification were used to enhance our recommendation model. The quality of the review is measured by its predicted probability to be helpful. We have tried the following two methods to incorporate review quality into our model:

- use quality scores as weights when taking the average of the neighbour's rating;
- filter out the low quality reviews.

The first method resulted in inferior performance to the baseline model, while the second approach leads to higher

User	Business	Rating	Opinion Score	Review
U1	B1	4	5	Quick and delicious. I had the three cheese crepe which was savory and flavorful. Decent prices by Vegas standards.
U2	B2	4	3	My lobster was a little overcooked, prices are a little high, and the side of coleslaw was pathetically small. But the food was good (I had a beltway), and it was a nice treat to have a fat lobster roll, so far from maine.

Table 6. Examples of User Rating and Reviews

Parameter	Size of Encoder	Size of Latent Space	Size of Decoder
No Cluster	512	256	512
Hierarchical Cluster	512	256	512
K-means	150	75	150
GMM	150	75	150

Table 7. Hyperparameters of Autoencoder for Different Clustering Models

accuracy compared to the case when we make no use of the review quality. It can be inferred that the reviews with extremely low quality might not be genuine, so removing them could make an impact on the model performance. But the normal and high quality reviews make little difference since both of them reflect authors' honest evaluation on the business.
Therefore, we chose the second method to build our helpfulness-enhanced model. After cross validation, we set 0.01 as the threshold for the quality score.

Rating Estimation

Based on the similar users' trustful ratings and reviews, we estimate the user's rating and opinion score towards a particular restaurant. The formula is given as follows:

$$P_{u,i} = \frac{\sum_{j \in U} r_{j,i} \times s_{j,u}}{\sum_{j \in U} s_{j,u}} \tag{4}$$

Where $P_{u,i}$ denotes user u's preference to business i, U denotes all the neighbour users, $r_{j,i}$ denotes user j's rating or opinion score on business i, $s_{j,u}$ denotes the similarity between user u and j.

The final predicted rating of user u for business i is the average of preference scores estimated through the rating and opinion score matrix.

5.2.3 Evaluation

To evaluation the collaborative filtering methods, we conducted train-test split by defining the reviews after 2019-06-30 as the testing set. The prediction accuracy is evaluated by R-square and RMSE. Table 8 gives the performance under different methods. We observed that removing unhelpful reviews and incorporating sentiment analysis could improve the accuracy. We finally chose the method based on K-means cluster to build our recommendation system.

5.3 Final Recommendation

We finally recommend top 10 restaurants based on predicted rating score from the candidate pool. The overall recommendation system is evaluated in the following sections.

5.3.1 Evaluation Metrics

We use four different metrics to evaluate our recommendation system.

Satisfaction measures users' attitudes towards the recommended restaurants. We define satisfaction as the average rating the user give to the recommended business that they visited after our recommendation. This metric focus on the businesses appearing in both the recommendation list and testing set as we are not sure about users' preference over the restaurants not visited by them. Along with satisfaction, we present the number of intercept business to show the size of our evaluation base.

Diversity measures how dissimilar the products in recommendation lists are. For each recommendation list, the diversity of recommendation for user u is defined as the average of pairwise dissimilarity between items with the following formula:

$$D_u = \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (1 - similarity(i,j))}{N(N+1)} \tag{5}$$

Method	No Filter		Remove Unhelpful Reviews	
	R2	RMSE	R2	RMSE
No Cluster (Rating Matrix Only)	0.103	1.395	0.115	1.386
No Cluster	0.125	1.379	0.137	1.369
K-means Cluster	0.125	1.379	0.139	1.368
GMM Cluster	0.125	1.379	0.138	1.368
Hierarchical Cluster	0.119	1.383	0.120	1.382

Table 8. Performance for Rating Prediction Under Different Collaborative Filtering Models. Noted that "No cluster (rating matrix only)" merely uses rating matrix, other methods apply both rating and opinion matrix.

We take the average of all the user's diversity to obtain the overall diversity.(Garcia Esparza et al., 2011)

Personalization measures the similarity between user's lists of recommendations. It is given by the average of the pairwise cosine similarity between each vectorized recommendation list.A lower value of the score indicates a higher extent of personalization.

Coverage measures the ability of our recommendation system to recommend as many products as possible.(Garcia Esparza et al., 2011) It is defined as the number of unique items in the recommendation list divided by the number of unique items in the training set. An ideal customized recommendation system should be able to cover a wide range of products instead of just recommending popular items.

Our primary objective is to retain an acceptable satisfaction score. At the same time, we aim to balance the diversity, personalization and coverage of our recommendation model.

5.3.2 Results and Analysis

We first formed a general recommendation for comparison. For the baseline model, we shortlist the restaurants with average rating equal to 5 and total review counts larger than 50, and randomly selected 10 businesses from the pool for recommendation. Then we simulated the recommendation using the current customized model with different number of candidate restaurants, and compared them with the general one.

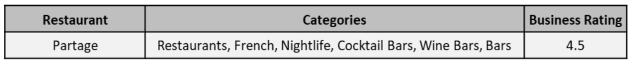
Table 9 shows the results of our evaluation. In terms of satisfaction, our model has lower score compared with the general recommendation, which is expected since the general one only focus on the popular business. On the other hand, the average rating of all the reviews in the testing set is 3.809, indicating that customers are more satisfied with the business recommended by our model.

In terms of personalization and coverage, our recommendation system outperforms the baseline model. By tailoring to the customer's preference, our model could reach a wider range of products rather than only focusing on the well-known and high-rated items. It can be observed in the table that narrowing down the candidate pool could improve these two scores, implying that the content-based stage plays an importance role to make personalized recommendation.

Finally, we noticed that our recommendation system has poor diversity compared with the baseline model. It can be inferred that the overspecialization problem of the content-based approach results in similar items that seldom jump outside the user's content profile.

6 Business Application

The proposed Yelp recommendation system aims to provide personalized recommendations to users, based on their past activities and reviews on the application. We considered two situation, recommendation for existing users and new users, to illustrate how this works.



Restaurant	Categories	Business Rating
Partage	Restaurants, French, Nightlife, Cocktail Bars, Wine Bars, Bars	4.5

Figure 12. User Selection: Restaurant Details

6.1 Existing Users

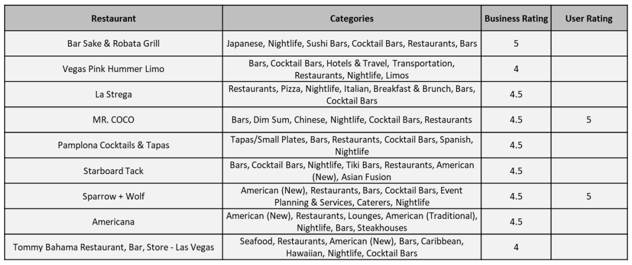
Based on the user's past reviews, a restaurant '*Partage*' has been chosen with its details as shown in Figure 12. The recommendation system takes this input and returns the top results as shown in Figure 13. During the process of the modelling, the data-set used for training was backdated and at the point of time, the user has yet to visit the following recommended restaurants (thus ignore "User Rating" Column for now). Subsequently, the user visited two of the following recommended restaurant and left a 5-star ratings, as shown in the "User Rating" column. As such, it is a testament that the proposed system would be able to recommend restaurants that are desirable by the user.

6.2 New Users

Nonetheless, as for every data science problem, there is always an issue of cold start and the model is unable to draw

Metrics	General	30	60	90
Satisfaction	4.895	4.196	4.328	4.233
Number of Actual Visit	38	150	174	146
Diversity	0.00033	0.000136	0.000154	0.000166
Personalization	0.244	0.016	0.024	0.031
Coverage	0.008	0.574	0.419	0.341

Table 9. Multi-metrics for Different Size of Candidates Pool



Restaurant	Categories	Business Rating	User Rating
Bar Sake & Robata Grill	Japanese, Nightlife, Sushi Bars, Cocktail Bars, Restaurants, Bars	5	
Vegas Pink Hummer Limo	Bars, Cocktail Bars, Hotels & Travel, Transportation, Restaurants, Nightlife, Limos	4	
La Strega	Restaurants, Pizza, Nightlife, Italian, Breakfast & Brunch, Bars, Cocktail Bars	4.5	
MR. COCO	Bars, Dim Sum, Chinese, Nightlife, Cocktail Bars, Restaurants	4.5	5
Pamplona Cocktails & Tapas	Tapas/Small Plates, Bars, Restaurants, Cocktail Bars, Spanish, Nightlife	4.5	
Starboard Tack	Bars, Cocktail Bars, Nightlife, Tiki Bars, Restaurants, American (New), Asian Fusion	4.5	
Sparrow + Wolf	American (New), Restaurants, Bars, Cocktail Bars, Event Planning & Services, Caterers, Nightlife	4.5	5
Americana	American (New), Restaurants, Lounges, American (Traditional), Nightlife, Bars, Steakhouses	4.5	
Tommy Bahama Restaurant, Bar, Store - Las Vegas	Seafood, Restaurants, American (New), Bars, Caribbean, Hawaiian, Nightlife, Cocktail Bars	4	

Figure 13. Recommendation System Results

any information from the user. In this case, new users would be problematic to the system as there is a lack of data. For such scenarios, the team has experimented on leveraging new user's friends/follows to predict recommendation, riding on the assumption that both parties would have similar taste and preferences. As the new users gradually built up their footprint in Yelp, the team believes that the proposed recommendation system would be able to serve them well.

7 Conclusion

In this project, we proposed a review quality based recommendation system for Yelp to bring business benefits and commercial usefulness to both Yelp and its users. We firstly used machine learning algorithms to predict the quality of a newly posed review and help Yelp prioritise high-quality reviews when presenting reviews to users. Among the algorithms we employed, XGBoost with top 20 features outperformed the others. Then, we built a personalized recommendation system by deploying those high-quality reviews and user metadata.

To address limitations we encountered during the whole process, we plan to explore and include more features, like images uploaded with the reviews for the classification of review quality, since review content itself does not play an important role in determining the quality of the review. As for the recommendation system, we plan to balance the number of recommended restaurants in each category for diversity in final recommendation list, to consider fine-grained opinions about specific aspects of a restaurant to capture users' focus on various features(Chen et al., 2015), and to advice Yelp to include initial profiling (i.e., what do you like?) during sign-up to get to know users better for cold start problems.

References

Li Chen, Guanliang Chen, and Feng Wang. 2015. Recommender systems based on user reviews: the state of the art. *User Modeling and User-Adapted Interaction* 25, 2 (2015), 99–154.

Sandra Garcia Esparza, Michael P O'Mahony, and Barry Smyth. 2011. A multi-criteria evaluation of a user generated content based recommender system. In *Presented at the 3rd Workshop on Recommender Systems and the Social Web (RSWEB-11), 5th ACM Conference on Recommender Systems, Chicago, IL, USA, 23-27 October 2011*.

Štefan Pero and Tomáš Horváth. 2013. Opinion-driven matrix factorization for rating prediction. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 1–13.

Sindhu Raghavan, Suriya Gunasekar, and Joydeep Ghosh. 2012. Review quality aware collaborative filtering. In *Proceedings of the sixth ACM conference on Recommender systems*. 123–130.