

Yelp Review Prediction

CUN XIAOFEI (A0186051J),

DING SHIEN (A0185995A),

LIANG XINRAN (A0186708R),

REN JIEWEN (A0186102N),

WANG XINRUI (A0186103M)

Section 1: Overview of Problem and Research Objectives

1.1. Introduction

Yelp is one of the most popular social networking sites that publish crowded-sources reviews about local businesses. Founded in 2004 in San Francisco, California, the website is a large online bulletin board featuring user-generated content, where members can share, review and communicate their opinions. Although the company is based out of San Francisco, its set up online communities in every major city in the United States and has recently expanded its reach to Canada, Ireland and the United Kingdom.¹

1.2 Business Insights

Yelp allows users post online reviews and using 1 to 5 stars to rate businesses. These reviews influence people's purchase behavior to a large extent. However, information overload is one of the common issues making it impossible for readers to read all of them, and potential customers tend to look at the star ratings of a business and ignore its textual reviews. Still, the relationship between businesses, reviews and ratings is unclear. How does review show a customer's attitude towards business and embody

this attitude in one's rating? What aspects of a business customers care the most? In these scenarios, Review Rating Prediction comes in handy. A Review Rating Prediction Model can help us decide whether it is enough to look at the star ratings of a product and ignore its textual reviews. For businesses, the model helps find out quantitative relationships between customer rating and its influence factors by predicting customer rating of restaurants using both restaurant-level features and customer reviews. For Yelp, Review Rating Prediction allows it to know better about its users, and optimize its recommendation system.²

1.3 Objective

The objective of this report is to Review Rating Prediction. In this paper, we treat Review Rating Prediction Problem as a regression problem in Machine Learning, where the dependent variable is individual review's star rating. The ultimate goal of this paper is to create a model to predict the rating a user would assign to a business.

Section 2: Data

2.1 Description and Basic Cleaning

The dataset was downloaded from the Kaggle competition - "RecSys2013: Yelp Business Rating Prediction"³. The original data consists of four Json files: business, user, review, checkin, each file consists of one json-object-per-line. They provided us with 11,537 business information including restaurants, bars and hotels, 229,907 customer reviews by 43,873 users. Except for 'checkin' dataset, there are no 'NA's in the other

¹Fuller, John. "How Yelp Works." *How Stuff Works*, InfoSpace Holdings LLC, computer.howstuffworks.com/internet/social-networking/networks/yelp.htm.

² Ricci et al. (2013). Ricci, F., Rokach, L., & Shapira, B. (2011). *Introduction to recommender systems handbook* (pp. 1-35). Springer US..

³"RecSys2013: Yelp Business Rating Prediction." *Kaggle*, www.kaggle.com/c/yelp-recsys-2013.

three datasets. The total size of data is about 170M.

Concretely, a business is represented in the ‘business.json’ file as a json object which specifies the business ID, its name, location, stars, review count, opening hours, etc. A text review is a json object in the ‘review.json’ file, which specifies the business ID, user ID, stars (integer values between and including 1 and 5), review text, date and votes.

For this analysis, we focused on reviews for restaurants and used the customer reviews and business attributes data. The necessary data is contained in the ‘business.json’, ‘review.json’ and ‘user.json’ files. After filtering restaurants out of all business, there were 177,273 customer reviews collected from 5,242 different restaurants.

State	AZ	CO	SC	Others
Number of restaurants	5,240	5	5	0

Table 2-1 Distribution of restaurants by state

We merge the three datasets based on review ID, customer ID and Business ID to obtain “yelp.csv”. We deleted the neighborhood variable because of the wrong format and we extract the vote variable to 3 different new variables from its sub-categories. The original vote variable was transformed to votes.useful, votes.cool and votes.funny.

2.2 Data Filtering

First, out of 177 thousand customer reviews, only 10 of them are for restaurants in CO or SC. Therefore, only reviews for restaurants in AZ(Arizona) are chosen. Second, The reviews in the Yelp dataset belong to several main categories, such as restaurants, hotels and shopping malls. The text reviews differ to a great extent if they are in different business categories.

For an instance, a typical hotel review may contain the following keywords: ‘concierge’, ‘room service’ and ‘bedding’, but these words are highly unlikely to appear in reviews for restaurants. Therefore, it is important to perform Review Rating model training and testing for each business category independently. For the purpose of this paper, we are going to **focus on reviews for restaurants only**. Restaurant is defined as a place where meals are prepared and served to customers by Cambridge Dictionary.⁴

Figure 2-1 shows the distribution of main business categories in the dataset. Around 75% of the 5996 businesses are restaurants. Moreover, customer reviews has a high percentage of 74.33% of the restaurant text reviews, as shown by Figure 2-2. After filtering, the current dataset contains 304576 reviews for 5242 restaurants.

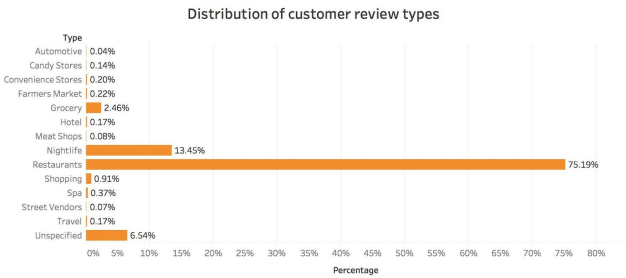


Figure 2-1 Distribution of customer review categories

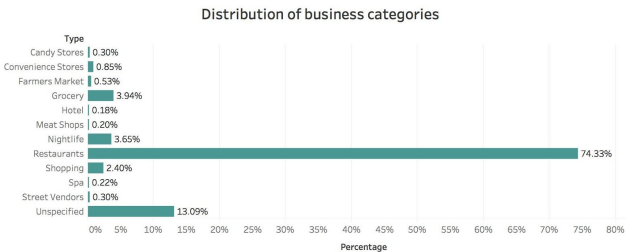


Figure 2-2 Distribution of business categories

2.3 Feature Construction

⁴ “Meaning of ‘Restaurant’ in the English Dictionary.” Cambridge Dictionary, <https://dictionary.cambridge.org/dictionary/english/restaurant>.

2.3.1 Feature Construction ‘Category’

Typically, categorical data attribute represents discrete values which belong to a specific finite set of classes. These are also often known as labels in the context of variables which are to be predicted by a model. In the dataset, the category of a user review typically looks like this: “*Food, American (New), Breweries, Restaurants*”. There are 2061 different categories in the dataset which provide repetitive information but in slightly different wording. For example, “*Bridal, Fashion, Shopping*” and “*Bridal, Shopping*”. Such a feature might induce the following 2 problems: For categorical feature of m distinct labels, we would get m separate features. This can easily **increase the size of the feature set** which leads to model training problems with regard to time, memory. We might also fall into the ‘**curse of dimensionality**’. There might be massive number of features and not enough representative samples, which deteriorates model performance and often provokes overfitting problem.⁵ For example, categories like “*Food, Desserts, Specialty Food, Chocolatiers & Shops*” and “*Bridal, Fashion, Shopping*” have 3 entries under these 2 categories. Instead of using the actual label values for model training, we **created a new feature “cuisine” with 46 labels** representing and summarizing the information provided in the original 2061 levels in “category” using dictionary.

We built an initial dictionary of 86 cuisines based on the most popular categories, as well as the prevailing cuisine types on “OpenTable.com” (OpenTable is an online restaurant-reservation service website that allows users to review and

rate restaurants). We matched the key and value according to the following rule:

“if *category* contains *key*, then we match *value* to the *key*.”

By matching, we labelled that entry with the dictionary key. The issue relating to this approach is that single review might belong to more than 1 key in the dictionary in this approach, therefore causing duplicate entries with only difference in the “cuisine” category. We decided to only take the first cuisine that entry has been labelled for the purpose of reducing dimensions of modelling. We then created another dictionary with more generality and repeated the mentioned process again. In the end, we transformed the information “categories” contains with 2061 labels to “cuisine” with 46 labels.

2.3.2 Feature Construction from ‘Reviews’

2.3.2.1 Word Count

The total number of words in a review maybe a good predictor to the star of the review, since the more a customer would like to write, the stronger one’s preference or aversion of this restaurant. So we added this variable “**wcount**” in our model.

2.3.2.2 Variables from Punctuation

When people write review, the punctuation also plays a important role in representing one’s emotional state. For instance, a review with a great number of question marks may suggest a strong confusion. Thus, we also included the total number of question marks and total number of exclamation point in a sentence to our model by creating variable “**qmark**” and “**emark**”.

2.4 Feature Visualization

Feature visualization is a good way to better understand the dataset. In this session, we tried to understand the constitution of the dataset to figure

⁵“Understanding Feature Engineering — Categorical Data.” *Towards Data Science*, towardsdatascience.com/understanding-feature-engineering-part-2-categorical-data-f54324193e63.

out relationships between different features, so that we could identify key features for the model.

2.4.1 Review Stars Distribution

Figure 2-3 shows that the star ratings (out of 5) for the restaurant reviews are not uniformly distributed. 60% of the reviews rate the corresponding restaurants highly (4 stars or more); the other classes are smaller.

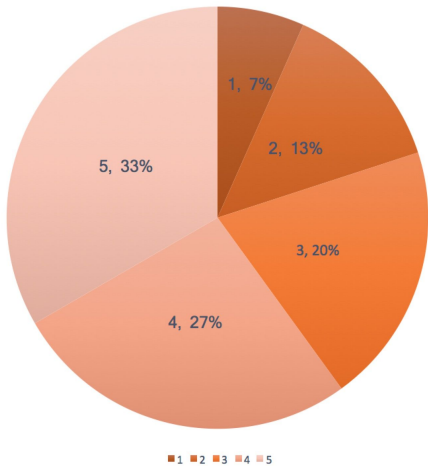


Figure 2-3 Distribution of review stars

2.4.2 Geography and Cuisine

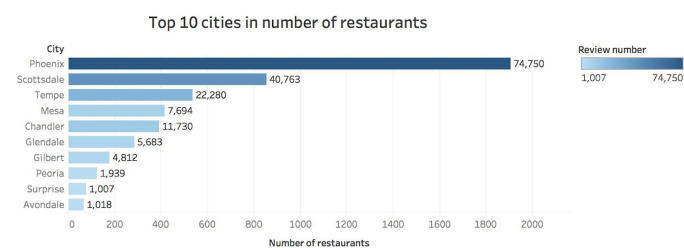


Figure 2-4 Distribution of restaurants among cities

Figure 2-4 shows the relationship between cities and number of restaurants. Colors and labels represent different review number of each city. Phoenix clearly dominate in both number of restaurants and review number among all the cities. This result is quite reasonable because Phoenix is the capital of Arizona. Phoenix, together with Scottsdale, contains more than 65% restaurants of the dataset.

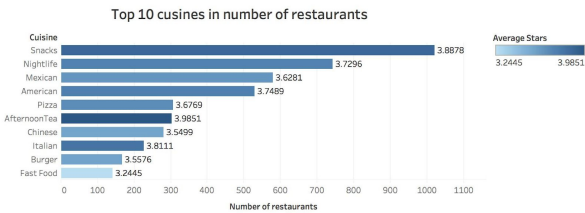


Figure 2-5 Distribution of restaurants among cuisines

As shown in figure 2-5, according to the cuisine which is built based on category information for each restaurant, number of restaurants and their average stars in each cuisine is shown in this histogram. Snacks, nightlife and Mexican are three top cuisines in the dataset. An interesting finding is the average stars of restaurants varies from different cuisines, showing customers tend to score higher or lower for certain cuisines, which provides evidence for the correlation between cuisine and stars. For example, afternoon tea and snakes have scores around 3.9, while fast food only has 3.24 of average score.

From table results, we group city by review number and we decide to put all cities which total review count less than 1000 together as “others”. Doing so not only elevate the results performance but also make model part more economic. For the same purpose, we also group cuisines which total review count less than 300 together.

2.4.3 Review Date and Stars

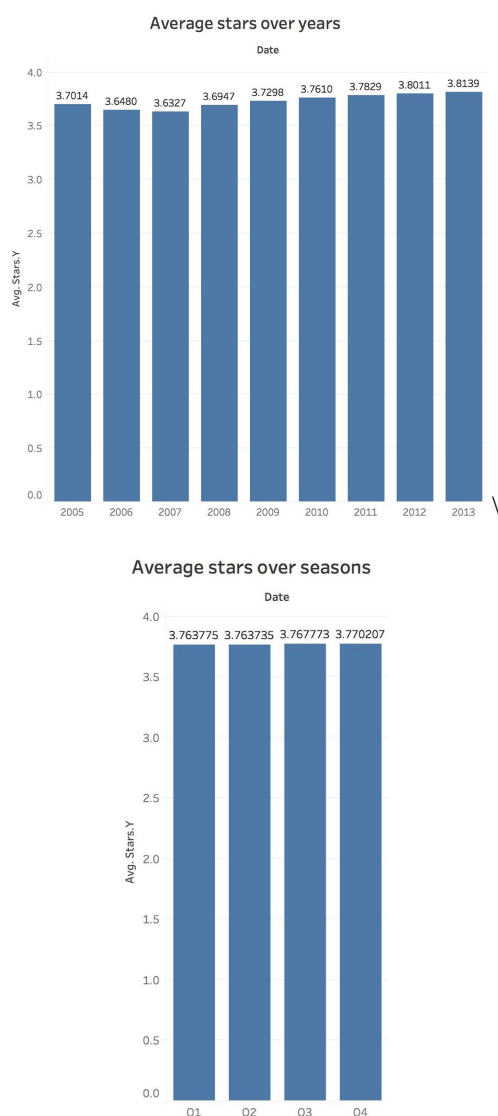


Figure 2-6 Distribution of average stars over years and seasons

Figure 2-6 show average stars of all reviews in the dataset over years, and seasons. We can find that average stars, which stay almost constant over seasons, vary obviously only over years. Therefore, we decide to use review year instead of review date in the original dataset as a feature of our model.

2.4.4 Word Cloud of Reviews



Figure 2-7 Word Cloud on reviews

Figure 2-7 shows the most frequent words customers use in reviews. “Good”, “food” and “place” are three biggest focuses in reviews, and “great”, “place”, “service” and “like” also have a large frequency. This result embodies that food, place and service are three features customers discuss most frequently in their reviews, which shows the great influence of these three features to a customer’s judgement to a restaurant. Moreover, reviewers usually use words such as good, great and like to express their emotion of their restaurant experience. One more insight for this word cloud is that people tend to write more positive reviews than negative ones since few negative words appear in this word cloud.

Section 3: Features from Text Mining

3.1 Scores from Sentiment Analysis

The emotion contained in a customer's review matters much on predicting his rating. In theory, one is greatly likely to rate high if he writes a positive review. As a result, in this session, we use sentiment analysis, a most common text classification tool which helps data analysts to get a rough understanding of people opinions, to analyze the review in yelp dataset, trying to find some meaningful features to build the prediction model.

3.1.1 Sentiment Dictionary

Sentiment dictionary is one important tool in sentiment analysis. Several dictionaries already exist for evaluating the emotion in text. In our project, we use the sentiment lexicons contained in “tidytext” package. The “bing” lexicon category words into positive and negative type and it is the most common used lexicon in text mining. “AFINN” assigns each word a score based on its sentiment level, which ranges from -5 to 5, with 5 presenting the most positive word and -5 showing the most negative emotion. Both lexicons are based on single word. After considering the context of our project, we decide to get sentiment on the whole review through both lexicons firstly, and choose the one that perform best after comparing the results.

3.1.2 The “bing” Lexicon

Firstly, we used the “bing” dictionary to assess the sentiments contained in review. After using the “get_sentiment” function in “tidytext” packages and doing “inner_join” with word table of all review to get the sentiment contribution based on the appearance time of positive and negative word, we got the top 20 most common positive word and negative word separately. A bar chart and is shown as Figure 3-1.

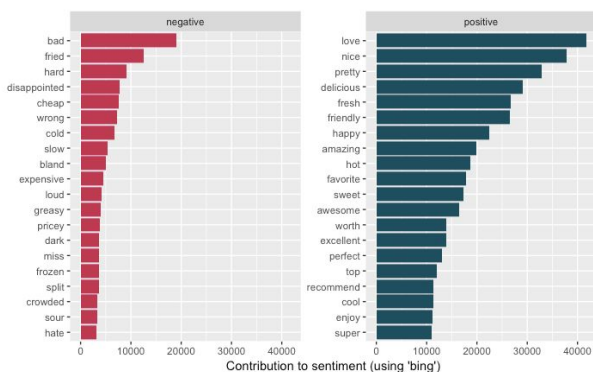


Figure 3-1 top 20 positive and negative words in reviews using ‘bing’

From the results shown in figure 3-1, it is obvious customers are much more likely to use words like “love” “nice” “pretty” to present their desired opinion, while among negative word, “bad” “fried” “hard” appear more frequently.

3.1.3 The “AFINN” Lexicon

Next, we used “AFINN” lexicon to do overall sentiment analysis as before. However, different from counting the frequency of each sentiment in previous step, we multiplied the appearance time with the emotion score to get each word’s sentiment contribution, of which the negative word is always below zero. Bar chart of different sentiment words with their corresponding scores is presented in figure 3-2.

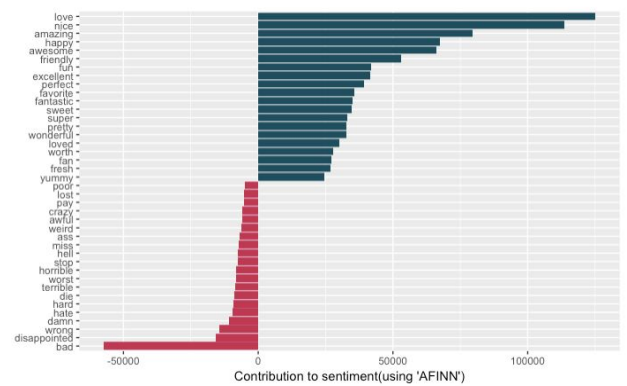


Figure 3-2 Distribution of sentiment words' scores using ‘AFINN’

3.1.4 Comparison

By comparing the common sentiment word of two analysis results, which is shown in figure 3-3, several findings are listed below.



Figure 3-3 Sentiment word clouds using 'bing'(left) and 'AFINN'(right)

- For positive emotion, the words appeared are very similar.
- However, for negative emotion, "bing" and "AFINN" give different results. In the former method, "fried" ranks second, and some adjective words that express food flavor or environment also listed in top 20. These words cannot be purely classed as negative under the background of our project, because people may just describe the fried food without any distinct emotion. What's more, "cheap" ranks high in the negative list, which is actually unreasonable cause cheap is generally

defined as an advantage of a good restaurant. In AFINN method, most common negative words all show strong and pointed negative emotion. For instance, disappointed and damn express reviewer's disgust distinctly. And considering AFINN assign each word with a score to present its emotion level, which is more valuable and meaningful in review sentiment analysis, we finally decide to use the "AFINN" lexicon to do each review's sentiment analysis.

3.1.5 "AFINN" Review Analysis

Based on the outcome of previous step, we can assign a score to each review by summing up all the words' emotion scores contained in this review at first. Positive number means the review shows customer's desire while negative number presents negative emotion. Then, categorize all reviews into five groups according to each review's star and plot each group's sample sentiment distribution, trying to roughly prove that the sentiment related to review does help in predicting the star. Figure 3-4 shows the result.

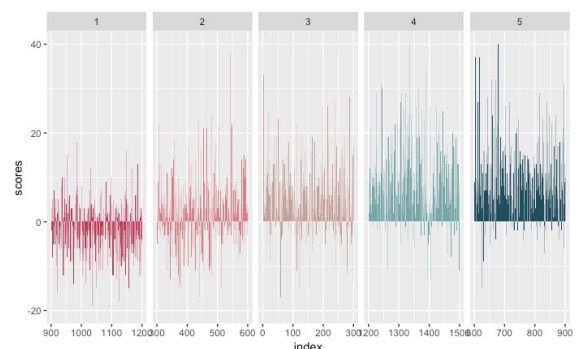


Figure 3-4 Distribution of sentiment scores of reviews in 5 groups

According to the result above, it is obvious that 1 star group's reviews have more negative scores than that of 5 star group. And the proportion of positive scores increases with the quantity of star, which proves that it is reasonable enough to

consider sentiment embedded in reviews as a variable in review's star prediction model.

As a result, we added the variable named “scores” into our prediction model, value of which presents the sentiment level embedded in reviews. Positive number means positive emotion, while negative number indicates downside emotion. The greater the absolute value, the stronger the emotion contained. Below shows several rows of yelp dataset with this newly added variable.

3.2 Topic Modeling

Topic modeling is a method for unsupervised classification of documents, similar to clustering on numeric data, which finds natural groups of items even when we're not sure what we're looking for. Latent Dirichlet allocation (LDA) is a particularly popular method for fitting a topic model. It treats each document as a mixture of topics, and each topic as a mixture of words. This allows documents to “overlap” each other in terms of content, rather than being separated into discrete groups, in a way that mirrors typical use of natural language. By applying LDA and topic modelling, the purpose of topic searching, exploring and recommending system can be achieved.

3.2.1 DTM construction

The next step is text-cleaning process. The purpose of text cleaning is to simplify the text data, eliminating as much as possible language dependent factors. Reviews are written in natural language for humans to understand. But in text mining, these data are not always easy for computers to process. In this case, there are two steps in text cleaning:

1. **Tokenization:** a document is treated as a string, removing all the punctuations and then partitioned into a list of tokens.

2. **Removing stop words:** stop words such as "the", "if", "and" which are frequently occurring but no significant meanings which need to be removed.

3.2.2 LDA Model

The training process requires R package “topicmodels” with its package dependencies (tm and others) to be loaded. An LDA model of simplified English Reviews on a sample of 166,931 reviews, returned after 50 iterations of Gibbs sampling, with $K=3$ topics, and Dirichlet hyper-parameters $\beta=0.1$ and $\alpha=50/K$. Meanwhile, topic terms in each topic are generated and we inspected the top 20 ones. This distribution represents how much a term is related to each topic.

Topic 1		Topic 2		Topic 3	
just	time	place	great	good	food
food	one	friendly	good	chicken	menu
back	get	can	food	like	try
service	like	love	always	cheese	also
even	pizza	bar	get	delicious	lunch
order	don	service	like	salad	ordered
know	didn	friendly	nice	fresh	sauce
got	went	staff	happy	really	little
first	never	really	pretty	restaurant	meal
will	table	night	hour	burger	sandwich

Table 3-1 topics generated from yelp reviews

Table 3-1 shows the final topic terms after the model is built, where top ten terms listing for each topic. With LDA, terms in the same topic tend to be similar. Formally speaking, they should be highly correlated. This topic distribution provides a way to search topic and explore among topics in order to find the text the user is looking for.

3.2.3 Word-topic Probabilities

The tidytext package provides this method for extracting the per-topic-per-word probabilities, called β (“beta”), from the model.

topic	term	beta
1	food	0.01106222
2	food	0.01701003
3	food	0.00679313
1	complaint	0.00012266
2	complaint	0.00026843
3	complaint	0.00022761
1	good	5.5266E-07
2	good	0.01734204
3	good	0.01884122

Table 3-2 per-topic-per-word probabilities

Notice that this has turned the model into a one-topic-per-term-per-row format. For each combination, the model computes the probability of that term being generated from that topic. For example, the term “food” has 0.011062218 probability of being generated from topic 1, but a 0.006793133 probability of being generated from topic 3. We used `top_n()` in dplyr package to find the 10 terms that are most common within each topic.

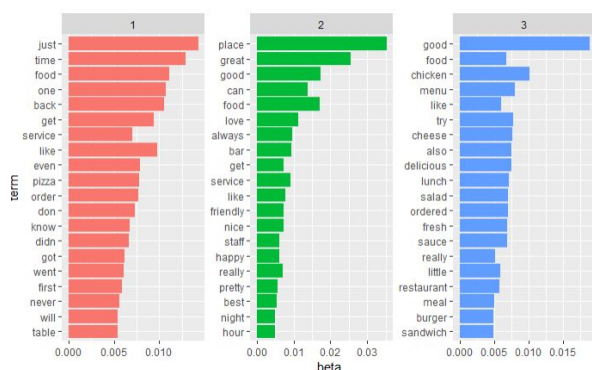


Figure 3-5 Most frequent terms within each topic

Figure 3-5 lets us understand the three topics that are extracted from the reviews. The most common words in topic 1 include “table”, “order”, “pizza”. Those most common in topic 2 include “staff”, “service”, and “friendly”, suggesting that this topic underlines the quality of service and restaurant location. And also topic 3 has most frequent terms like “chicken”, “cheese”, “burger”. Those are related to specific dishes. One important observation about the words in each topic is that some words, such as “service” and

“staff”, are common within both topics. This is an advantage of topic modeling as opposed to “hard clustering” methods: topics used in natural language could have some overlap in terms of words.

As an alternative, we could consider the terms that had the *greatest difference* in β between topic 1 and topic 2. This can be estimated based on the log ratio of the two: $\log_2(\frac{\beta_2}{\beta_1})$ (a log ratio is useful because it makes the difference symmetrical: β_2 being twice as large leads to a log ratio of 1, while β_1 being twice as large results in -1). We can also filter for relatively common words, such as those that have a β greater than 1/1000 in at least one topic, to compute a set of highly relevant words.

term	topic1	topic2	topic3	log_ratio
pizza	0.0077855	0.0000000	0.0000003	-18.0320022
order	0.0076626	0.0000000	0.0011790	-18.0090461
know	0.0067341	0.0000000	0.0000000	-17.8227020
didn	0.0066643	0.0000000	0.0000112	-17.8076830
got	0.0062233	0.0000000	0.0024775	-17.7088922
place	0.0000000	0.0351645	0.0000011	20.3496797
love	0.0000000	0.0113768	0.0000780	18.7216575
always	0.0000000	0.0096174	0.0000000	18.4792798
bar	0.0000000	0.0092950	0.0000000	18.4300833
staff	0.0000000	0.0062052	0.0000000	17.8471055

Table 3-3 Words with the greatest difference in β between topic 2 and topic 1

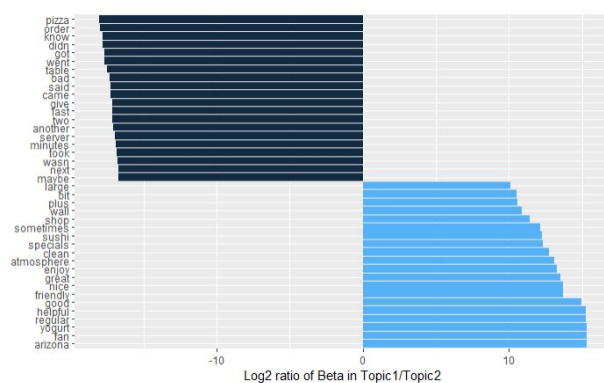


Figure 3-6 Words with the greatest difference in β between topic 2 and topic 1

We can see that the words more common in topic 2 include such as “excellent” and “favorite”, as well as service quality such as “friendly”. Topic 1

is more characterized by time and sceneries like “night” and “location”, as well as particular cuisine terms such as “salad”. In addition, Topic 2 has more adjectives which are related to food and menu.

3.2.4 Document-topic Probabilities

Besides estimating each topic as a mixture of words, LDA also models each document as a mixture of topics. We can examine the *per-document-per-topic probabilities*, called γ (“gamma”), with the matrix=“gamma” argument to *tidy()*.

The results are shown below. Each value is an estimated proportion of words from that document that are generated from the corresponding topic. In other words, it stands for the *probability* of a document belonging to a topic. For example, the model estimates that there are about **23.9%** of the words in document 120 are generated from topic 1.

Review	topic1	topic2	topic3
120	0.23929825	0.45403509	0.30666667
121	0.44888889	0.27555556	0.27555556
122	0.14595661	0.3530572	0.50098619
123	0.35930736	0.32034632	0.32034632
124	0.25591398	0.26236559	0.48172043
125	0.50617284	0.15432099	0.33950617
126	0.40784314	0.25490196	0.3372549
127	0.15151515	0.36174242	0.48674242
128	0.36769759	0.3161512	0.3161512
129	0.24702381	0.29166667	0.46130952
130	0.41666667	0.29166667	0.29166667

Table 3-4 Document-topic Probabilities

We can see that many of these documents are drawn from a mix of the two or three topics, but that document 122 is drawn from topic 2 with higher than 50% probability, having γ from the other two much smaller. To check this answer, we can tidy the document-term matrix and check the most common words in that document.

3.3 N-grams

So far, all the features we gain are based on one word unit. However, sometimes a single word can’t present the inherent information of the text adequately. The relationship between words is also of great importance. For instance, the single word “hour” can be interpreted as a type of time, but the word pair “happy hour” means totally different from “hour”. As a result, we analyze the review by tokenizing the text into n-grams, which is a continuous sequence of words collected from a text.

3.3.1 Bigram

Firstly, we want to find some meaningful pairs of consecutive words. The “unnest_tokens” function in “tidytext” package is very useful. By adding the augment “token= ‘ngrams’” and setting n to the number of words we wish to capture in each n-gram, we can do tokenizing. After obtaining the bigrams, we pop up the top 20 most common tokens (table 3-5 left). The result is not as good as we expected, because nearly all bigrams popped up are made up of stop-words, which have little value in text mining. Thus, we separate the bigrams into two columns, remove all the stop-words, recombine the columns into one and finally find the most common bigrams not containing stop-words (table 3-5 right).

	bigram	n		bigram	n
1	it was	78516	1	happy hour	13213
2	of the	77346	2	ice cream	6451
3	and the	71575	3	mexican food	4850
4	in the	63490	4	5 stars	3947
5	this place	60588	5	sweet potato	3245
6	on the	53276	6	fast food	3159
7	the food	49844	7	highly recommend	3039
8	and i	48685	8	customer service	2972
9	i was	44932	9	carne asada	2742
10	for the	39119	10	1 2	2613

Table 3-5 Top 10 tokens with and without stop-words of bigram

3.3.2 Trigram

Further, we set n to three, trying to find consecutive sequence of three words, and pop up the top 20 repeat most often trigrams. The result shows below (table 3-6). It is obvious that those meaningful trigrams are still related to the bigrams we obtain in previous step, such as “happy hour menu” and “sweet potato fries”. And the rest of trigrams give little useful information. As a result, we decide to make use of bigrams.

	bigram	n
1	sweet potato fries	2062
2	happy hour menu	746
3	love love love	693
4	http www.yelp.com biz_photos	567
5	happy hour specials	529
6	happy hour prices	513
7	chicken fried steak	472
8	pico de gallo	451
9	NA NA NA	440
10	pulled pork sandwich	430

Table 3-6 Top 10 tokens of trigram

3.3.3 Feature Creation

From table 3-5, we can find some important and insightful bigrams. “Happy hour” ranks number one, suggesting that happy hour is likely to play a role in influencing customer’s star. Besides, “ice cream” is listed, which shows a correlation exists between star and dessert. The same applies to “sweet potato”. The rest tokens either have been included in other features, like “fast food”, or have ambiguous meaning, like “5 star”. Consequently, we added three variables named “**happy**” “**ice**” and “**potato**”, presenting the tf-idf value of “happy hour”, “ice cream” and “sweet potato” across all reviews respectively.

Section 4: Baseline Model and Ensemble Learning

4.1 Variable Explanation

4.1.1 Dependent Variable and Independent Variables

“star.x” means the rating for a restaurant by a customer. In this analysis, that was to be predicted. Due to system limitation, customers could only choose the rating from five choices. But we treated it as numeric and continuous to improve the performance. Our predictions were in a continuous range from 1 to 5. As for independent variables, control variables about customers and restaurants were included, such as “open”, “city”, “average_stars” and so on. Also, we developed new features from text mining. One is the score of sentiment analysis, and the probabilities a review belonging to a topic. Paying attention to punctuations and emotional marks, we calculated the number of question marks and exclamation points.

4.1.2 OneHot Encoding

In order to dealing with categorical variables, we did OneHot encoding to convert them into dummy variables with two levels. Finally we had 25 key variables in the training dataset, which were “stars.x”, “open”, “city”, “review_count.x”, “state”, “stars.y”, “funny”, “useful”, “cool”, “average_stars”, “review_count.y”, “yelp.votes.funny”, “yelp.votes.useful”, “yelp.votes.cool”, “scores”, “happy”, “ice”, “potato”, “cuisine”, “t1”, “t2”, “t3”, “wcount”, “qmark”, “emark”. and there would be 116 variables after OneHot. See the appendix for variable dictionary.

4.2 Tuning Basic Models

4.2.1 KNN

In k-nearest neighbor regression, "closeness" is defined in terms of Euclidean distance. The unknown sample is assigned to the most common class among its k nearest neighbours. Using train function from "caret" package, we tuned `.k=seq(3,15,2)`. RMSE was used to select the optimal model using the smallest value and it chose `.k=3` automatically. In the optimal model, **RMSE=1.103993**.

4.2.2 GLM(Generalized Linear Model)

Multinomial logistic regression is used to treat classification problem with more than 2 types of outcomes. Similar to logistic regression. With library("glmnetUtils"), we used `cv.glmnet` function to get the optimal lambda and constructed the new one with **lambda=0.001057**. The RMSE from the final model is **0.845950**.

4.2.3 Rpart

In case of regression, the mean of the response variable in one node would be assigned to this node. The structure first split from the root. After each split, two new nodes are created (assuming we only make binary splits). Each node only contains a subset of the observations. The partitions of the data, which are not split anymore, are called terminal nodes or leafs. Tree models are easier to interpret and we can see the partial dependence of each variable. In the **post-pruned** tree, `.cp = 12`. The RMSE of this model is **1.1748**.

4.2.4 Neural Network

Artificial neural networks are statistical learning models, inspired by biological neural networks, that are used in machine learning. These networks are represented as systems of interconnected "neurons", which send messages to each other. The connections within the network can be systematically adjusted based on inputs and outputs, making them an efficient supervised

learning method. Since the dataset is huge, neural network worked very slow. We get the optimal parameters from **DataRobot**, `.layer = 1` and `.layer2 = 256`, RMSE from neural network is **0.8385**.

4.2.5 Support Vector Machine

Support Vector Machines (SVM) (Tsochantaridis et al., 2004) eliminate the non-uniqueness of solutions by optimizing the margin around the decision boundary, and handle non-separable data by allowing misclassifications. A parameter C controls overfitting. When C is small, the algorithm focuses on maximizing the margin, even if this means more misclassifications, and for large values of C, the margin is decreased if this helps to classify more examples correctly. With `.cost = 1` as the optimal parameter, we got the RMSE equaling to **0.8645**.

4.3 Ensemble Learning

4.3.1 Random Forest

Bagging is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting. Random Forest is one type of ensemble learning method which operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests effectively avoid the potential overfitting by single tree model. We tuned `.mtry=15:25` and `.mtry=16` is chosen for our optimal model. RMSE is **0.8215716**.

4.3.2 Xgboost

Xgboost is another type of ensemble learning method which is extremely efficient nowadays. XGBoost is a accurate implementation of gradient

boosting machines and it has proven to push the limits of computing power for boosted trees algorithms as it was built and developed for the sole purpose of model performance and computational speed. After tuning, we get `colsample_bytree = 0.8`, `subsample = 0.8`, `nrounds=50`, `max_depth = 4`, `eta = 0.2`, `gamma = 0.1`, `colsample_bytree = 0.8`, `min_child_weight = 4` and `subsample = 0.8`. The RMSE of this model is **0.8723659**.

4.3.3 Stacking

Stacking is an ensemble learning technique that combines multiple classification or regression models via a meta-classifier or a meta-regressor. The base level models are trained based on a complete training set, then the meta-model is trained on the outputs of the base level model as features to approximate the same target function.⁶ **Linear regression** is used here as the level_1 function, RMSE in the model is **0.8139**.

4.4 Results without Feature Engineering

Baseline model RMSE			
knn	rpart	random forest	xgbTree
1.1040	1.1748	0.8216	0.8724
SVM	neural network	glm	Stacking
0.8645	0.8385	0.8459	0.8139

Table 4-1 RMSE before feature engineering

Section 5: Features Engineering Process and Results

5.1 Feature Engineering

5.1.1 Score_emark

In previous step, we assumed the number of exclamation mark as a valuable indication for the emotion embedded in reviews. However, the emotion presented by exclamation mark can be ambiguous sometimes. For instance, exclamation

mark together with positive word strengthens the positive emotion, while exclamation mark in negative sentence actually shows more intense negative emotion. In order to avoid ambiguity, we assigned the number of exclamation mark with sentiment by multiplying the variable “**emark**” with “**scores**”, thus created the new variable names “**score_emark**”.

5.1.2 Score_qmark

The same ambiguous problem exists in question mark. Review like “how could this restaurant be such great?” has absolutely different impact on review’s given star when compared with sentence like “excuse me?”. Consequently, we also created another new feature named “**score_qmark**” by multiplying “**qmark**” with “**scores**”.

5.1.3 Senti_by_word

Both the sentiment and the length of a review can be good estimators for review star’s prediction. However, two reviews with same score but different word counts may have different influence on the star. For example, a short review with quite high positive score may include more intensive emotion word than one long review with the same score. Thus, we add a variable name “**senti_by_word**”, the value of which equals variable “**scores**” divided by “**wcount**”.

5.2 Results after Feature Engineering

Overall, RMSE reduces for all models. It means that the new features constructed from feature engineering make sense. Among all models the improvement of SVM is the highest, while other decreases in RMSE are not very obvious.

⁶ Smolyakov, Vadim. “Ensemble Learning to Improve Machine Learning Results.” *Stats and Bots*, blog.statsbot.co/ensemble-learning-d1dcd548e936.

Model RMSE				
	knn	rpart	random forest	xgbTree
Before	1.1040	1.1748	0.8216	0.8724
Feature Engineering	1.0829		0.8209	0.8711
Change	-0.0211		-0.0007	-0.0012
	SVM	neural network	glm	Stacking
Before	0.8645	0.8385	0.8459	0.8139
Feature Engineering	0.8427	0.8290	0.8413	0.8085
Change	-0.0218	-0.0095	-0.0046	-0.0054

Table 5-1 RMSE after feature engineering

5.3 Importance of Variable from Text Mining

We used **variable importance** to measure the value of variables created from text mining. After popping up the top 20 most important variables from models, we found that variables from text mining take up a large proportion, proving that text mining is quite useful in creating features for our prediction model. Below are variable importance results of XGBtree and SVM.

XGBtree Variable Importance(Top 20)	
scores	100
average_stars	73.1294384
stars.y	45.1892633
yelp.votes.cool	4.514897
yelp.votes.useful	4.380965
cool	3.0034772
yelp.votes.funny	2.9726029
funny	2.7735164
review_count.y	2.3061526
useful	2.2202559
review_count.x	1.6766964
happy	0.82276
ice	0.4115468
cuisine.Snacks	0.334949
openTRUE	0.262444
cuisine.Seafood	0.2366681
cuisine.Pizza	0.2318076
city.Phoenix	0.2154007
city.others	0.1764499
potato	0.1746601

Table 5-2 Variable importance of XGBtree

Support Vector Machine Variable Importance(Top 20)	
t2	100
senti_by_word	95.3950
average_stars	92.8630
stars.y	70.2217
scores	64.0822
t1	42.5082
t3	14.6358
score_emark	11.8641
qmark	10.0014
review_count.x	8.4525
wcount	7.4503
emark	7.1409
score_qmark	3.3538
cuisine.Snacks	1.7491
yelp.vote.funny	1.7272
openTRUE	1.5072
cuisine.Maxican	0.9000
yelp.vote.cool	0.7622
cuisine.Chinese	0.7365
yelp.vote.useful	0.5731

Table 5-3 Variable importance of SVM

Section 6: Conclusion and Future Developments

6.1 Conclusions

In this paper, we tackled the Review Rating Prediction problem for restaurant reviews on Yelp. We treated it as a numeric regression problem, and examined various feature extraction and supervised learning methods to construct prediction systems including seven basic models and two ensemble models. Experimentation and performance evaluation through 5-fold cross validation yields one system, random forest on the set of 63 features obtained from raw data and text mining, that exhibits better predictive powers than the others. Observing variable importance and significantly gives “scores”, “average_stars”, and “stars.y” as the most important variables. In general, features created from reviews make significant contribution to the prediction of consumer ratings. Other than “scores” which

represents sentiment score of each review, features created from topic modelling and most frequently appeared phrases also This result provides clear evidence that review is strongly related to customer rating, and it is practical to predict customer rating based on texture review. Our model can be used to generate star ratings on review websites where users can write free-form text reviews without giving a star rating.











#	Δ pub	Team Name	Kernel	Team Members	Score @	Entries	Last
1	—	BrickMover			1.21250	40	5y
2	—	vsu			1.21551	13	5y
3	▲1	Merlion			1.22723	29	5y
4	▼1	Sergey			1.22855	15	5y
5	▲8	liuyongqi			1.22979	13	5y
6	—	Gxav & Paul Duan			1.23020	37	5y
7	▼2	Bryan Gregory			1.23107	42	5y
8	▼1	Biro Biro & Dmitry			1.23118	43	5y
9	▲2	n_m			1.23687	30	5y
10	—	Li			1.23698	24	5y

Table 6-1 competition results

Also here we show the results from top ten teams in that kaggle competition. Obviously, with feature engineering and model tuning, we beat the champion.

6.2 Future Developments

We can try more sophisticated feature engineering methods, such as Parts-of-Speech (POS) tagging, to obtain more useful n-grams. For example, instead of considering all possible bigrams and trigrams, we can extract all the adjective-noun pairs or all the noun-noun pairs to get more meaningful 2-tuples or 3-tuples. Another avenue for future work is to test how our prediction models would perform on other business categories, such as shopping, hotels and so on. .Also, we could look more deeply at trend setting users - that is, those users who have accumulated significant reviews, friends and followers, to see if their individual reviews are more predictive than the remaining aggregate review data.

6.3 Limitations

Limitations that arose during the course of this study included processor and space limitations in computation, where the data set was too large to evaluate with our laptops alone. Future improvements could involve parallel processing and distributing data over multiple machines via Hadoop or Apache Spark technology.

References

1. Akaichi, Jalel, Zeineb Dhouioui, and Maria José López-Huertas Pérez. "Text mining facebook status updates for sentiment classification." *System Theory, Control and Computing (ICSTCC)*, 2013 17th International Conference. IEEE, 2013.
2. Asghar, Nabiha. "Yelp Dataset Challenge: Review Rating Prediction." *arXiv preprint arXiv:1605.05362* (2016).
3. Blei, David M. "Probabilistic topic models." *Communications of the ACM* 55.4 (2012): 77-84.
4. Fuller, John. "How Yelp Works." *How Stuff Works*, InfoSpace Holdings LLC, computer.howstuffworks.com/internet/social-networking/networks/yelp.htm.
5. Huang, James, Stephanie Rogers, and Eunkwang Joo. "Improving restaurants by extracting subtopics from yelp reviews." *iConference 2014 (Social Media Expo)* (2014).
6. "Meaning of 'Restaurant' in the English Dictionary." *Cambridge Dictionary*, <https://dictionary.cambridge.org/dictionary/english/restaurant>.
7. Pak, Alexander, and Patrick Paroubek. "Twitter as a corpus for sentiment analysis

- and opinion mining." *LREc*. Vol. 10. No. 2010. 2010.
8. "RecSys2013: Yelp Business Rating Prediction." *Kaggle*, www.kaggle.com/c/yelp-recsys-2013.
 9. Seide, Frank, et al. "Feature engineering in context-dependent deep neural networks for conversational speech transcription." *Automatic Speech Recognition and Understanding (ASRU)*, 2011 IEEE Workshop on. IEEE, 2011.
 10. Smolyakov, Vadim. "Ensemble Learning to Improve Machine Learning Results." *Stats and Bots*, blog.statsbot.co/ensemble-learning-d1dcd548e936.
 11. "Understanding Feature Engineering — Categorical Data." *Towards Data Science*, towardsdatascience.com/understanding-feature-engineering-part-2-categorical-data-f54324193e63.
 12. Wang, Chong, and David M. Blei. "Collaborative topic modeling for recommending scientific articles." *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011.
 13. Silge, Julia, and David Robinson. *Text mining with R: A tidy approach*. "O'Reilly Media, Inc.", 2017.
 14. Yu, Boya, et al. "Identifying Restaurant Features via Sentiment Analysis on Yelp Reviews." *arXiv preprint arXiv:1709.08698* (2017).