Yelp data analysis

November 18, 2019

GROUP 8
Haifeng Liu, Jiawen Chen, Ruochen Yin

1 Introduction

Yelp is an extremely widely-used app in the United States. Reviews on Yelp can influence one's choice in literally every suspect in everyday life, which makes yelp reviews significantly related to business' profit. To help enterprises to improve their performance on Yelp, we decided to analyze the reviews and business attributes of business to help them improve their service and find out their strengths and weakness.

2 Data summary

We focused on restaurants in Wisconsin. There are 93488 reviews and 1703 businesses in total. We only used review.json and bussiness.json. For review, we used stars, text, business names to do further analysis. For business, we used stars, attributes, business names to do further investigation. Detailed procedures are in the review and business section.

3 Review

There are 93488 reviews of the restaurants in Wisconsin. To find out the strengths and weaknesses of each business, we used "business name", "text" and "stars" in the review.json to make models and do some statistical analysis.

3.1 Data preprocessing

There are several steps to process the text. Data preprocessing: First of all, we exacted the data with the state is Wisconsin and Restaurants in Categories from original data. To process reviews, we turn all upper case to lower case, and then we turn all Inflection word into period so that in the next step we can split a sentence with inflection word into two sentences. And there is some abbreviation, we turn it into is, so when we remove the stop word, we will remove that too. Before we let the sentence turn into words, we need to judge if this sentence is positive or negative, if this sentence is negative, we will add a suffix on every word in that sentence. Finally, we use sent_tokenize to turn a paragraph into the sentence and use word_tokenize to turn the sentence into words.

3.2 Feature selection

See code/review pre_visualization.ipynb

After the preprocessing of the reviews' text. We use two methods to reduce the number of variables and get some inspiration from the most important stems.

3.2.1 Method 1: Document Frequency

This method is based on the occurrence of a word in all documents We just calculate the number of times a given term appears in the whole reviews of each business. And then sort the words according to the times that they occur in all the reviews.

3.2.2 Method 2: TF-IDF

The full name of TF-IDF is "term frequency weighted by inverse document frequency". It is a relevance model that determines how relevant a particular document is for a given query. The method weights the number of times that query appears within the searchable text corpus (TF) by the number of times that query appears within the specific document (IDF).

3.3 Modeling and suggestion

See code/review_model&advice.ipynb

3.3.1 Step 1: Create a new dataset

From the two data files: review.json and bussiness.json, extract the variables that we want: 'business_id', 'business_name', 'business_star', 'review_text' and 'review_star' to create a new dataset for further use.

3.3.2 Step 2: Create a frequency matrix

For each business, we create a frequency matrix of the reviews' texts. In the code file "review_model&advice.ipynb", we use several functions to achieve the goal: 1. Use getWords() to get all the words with no repetition from each business' reviews. 2. Use getDataFrame() to make the frequency matrix for each business.

3.3.3 Step 3: Make models

Use the function makeModel() to do linear regression for each business according to their reviews. And output the rank of the important words according to their coefficients.

3.3.4 Step 4: Give advice

Use the function makeRawAdvice() to automatically create advice for each business according to the output from the makeModel() function.

4 Business

There are total 1703 restaurants in Wisconsin. To find out the strength and weakness of each business, we used attributes which means the Service provided in each business. There are some typical categories like “WIFI”, “Business Parking” which seem reasonable to affect the average review of one business.

4.1 Data preprocessing

There are sevral steps to process the attribute. Since the attribute is a multi-layer dictionary, so we have to extract its key and values for further investigation. Detailed code is in code/business.ipynb. ### STEP 1. Convert a multi-layer dictionary to a flat list. The first step is to convert a multi-layer dictionary to a flat list. We first transformed all the keys and values to lower case. For multiple-layer items, we use the key from the outer layer and key1 from the inner layer as the new key i.e. “key_key1” such as “bussinessparking_garage”. Then we try to combine the key and values, to unify the attribute, we convert all values like “yes”, “true” to be “yes”. All values like “no”,“none” to be “none”. For the keys with their unique values, we use the value to combine with the key like “wifi_free” and “wifi_paid”.

In summary, there are three steps. 1. Key & value to lower case 2. Multiple layer: use key_key1 as new item => businessparking_garage 3. “yes” or “true” values => “yes”.“no” or “none” => “none”.

4.1.1 STEP 2. Deal with missing data.

Since the attribute of every bussiness is not same and havs different categories compared to others. So adding NA value is necessary. However, add a NA value for each existing attribute is too distractive and with too many varaibles, it will decrease the statistical power of our final model. So we decided to add NA for each categories. Like “wifi”, there are wifi_yes, wifi_none, wifi_free and wifi_paid. Adding NA to each of them seems dumb, so for those bussiness who does not have any one of these four, we will add a wifi_NA for them. At last, we can convert these five words to be “free”, “paid”, “none”, “yes” and NA under “wifi” category.

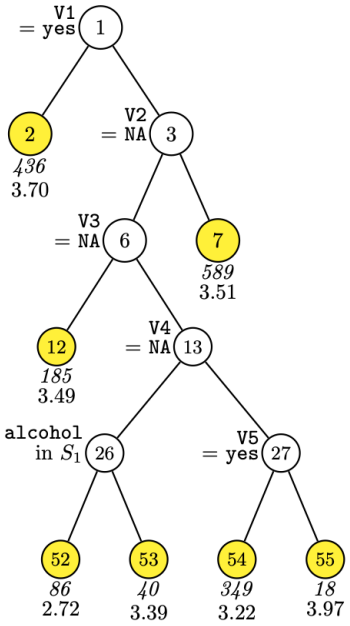
After data processing, we got a list with business stars and 74 attibutes. Detailed attributes can be found in

4.2 Regression tree

Since the missing rate is extremely high in the missing data, we decided to use GUIDE[1], a statistical software to deal with missing data and fit tree model for it. GUIDE is developed and maintained by Weiyin, Loh in statistical aprtment in UW-madison. Since dependent variable: stars is a continuous variable, we fitter regression tree using the processed data. GUIDE input and output file are in data/guide folder.

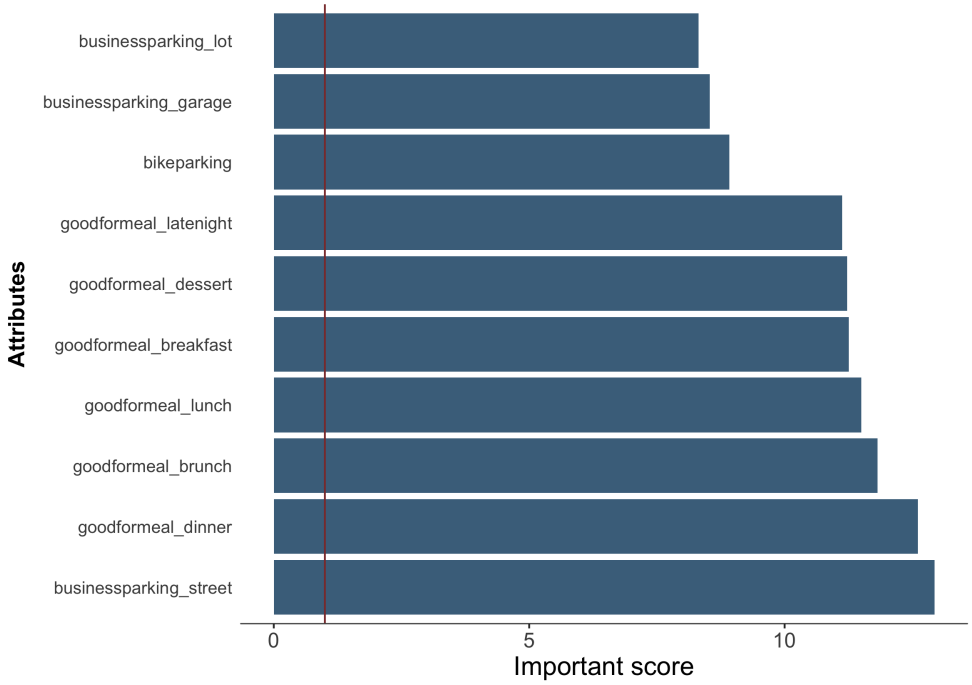
4.2.1 Regression tree result

Number of terminal nodes of final tree is 7 and total number of nodes of final tree is 13. Best split variable is businessparking_street and second best split variable (based on curvature test) at root node is goodformeal_dinner. The tree diagrame is shown as following.



In the plot, V1 = businessparking_street. V2 = goodformeal_dinner. V3 = restaurantsgoodforgroups. V4 = businessparking_street. V5 = businessacceptscreditcards. Set S1 = {NA, full_bar}. We can see the split nodes are based on business parking and good for meal. We can see that the estimate of business with street parking is much higher than a business without it. Also, not providing alcohol and providing full bar alcohol both decreases the stars, so the business have to seek a balance between them. Also, providing credit card payment somehow decreases the stars of review, which means the tax maybe influence the stars.

Importance score The importance score is one of the functions of GUIDE, GUIDE calculates the importance score for each variable when fitting a regression tree based on their p-value. Importance score over one is considered as "important". To find out which variables affect the result most, we calculated the importance score for every variable in attributes. Here are the top 10 variables.



We can see that the most important variables are "Business parking" and "Good For meal" which is reasonable. To improve their average review stars, the business has to focus on their service about business parking and Good for the meal, which means their food.

Linear Regression To get the weight and p-value of each variable to claim the importance and improvement of each attribute, we select variables with important score larger than 1 (according to the GUIDE manuscript: importance score larger than one is considered as essential variables). Since the large amount of NAs in the data, we replaced NA with a new level that is not in the attributes. We used those variables to fit linear regression and delete all the variables with NA in their coefficients since, in this way, there are multi-collinearity in those variables. And we fit the model again. At last, we got 19 significant variables. The top 5 variables are in the below table. Full table, please see data/bussiness_linear.csv.

Attribute:Level	Estimate	P value
businessparking_street: yes	0.4329297	5.215079e-09
restaurantsattire: new_level	0.4182954	5.524229e-08
noiselevel: very_loud	-0.6002776	9.299125e-06
drivethru: yes	-0.6382427	1.202254e-05
goodformeal_dinner: yes	0.2194556	1.770675e-04

New level refers to NA in the table. We can see that the top five variables are "bussiness parking : street", "Resuaurant attire: NA", "noise level: very loud", "drive thru: yes", "Good for meal: dinner". All those five are intuitive, and we can give business advice based on the result of linear regression.

4.2.2 Advice

We gave advice based on the estimated slope of variables and decide whether the variable is the strength or weakness of the business by the estimated slope is positive or negative. We can see from the above table that has street parking will improve the star by 0.4, very loud noise level decreases the star by 0.6, good for a meal in dinner can raise the star by 0.2. We use such information to create advice. For example, the noise level, for the restaurant with "very_loud" or "loud" noise level, we give them advice: The noise level in your business is too loud, and it decreases your business review about 0.2-0.6 stars. We do the same to all the significant variables in the linear regression output and generate a paragraph of advice for each business. Detail code is in: code/business_advice.R. Advice result is in data/bussiness_advice.txt.

5 Website

We used React and Javascript to build a restaurant website, which contains all the restaurants in Wisconsin. If you want to run these codes in your local computer, please install react and nodejs first, and then set the necessary environment configuration.

Here is our website link: https://vast-harbor-08232.herokuapp.com

6 Contributions

Haifeng Liu: Analysis of reviews' words. Previsualization of reviews. Modeling and suggestions according to the reviews. The part of review analysis in PPT and summary.jpynb

Jiawen Chen: Business file pre-procession. Regression tree on attributes. Linear model of attributes. Advice based on attributes.

Ruochen Yin: Preprocessed the data and built website.

7 Reference

1. Wei-Yin Loh Nunta Vanichsetakul (1988) Tree-Structured Classification via Generalized Discriminant Analysis, Journal of the American Statistical Association, 83:403, 715-725, DOI: 10.1080/01621459.1988.10478652