Midterm Report - Predict Restaurants Rating

Caroline Troude: cct65 - Diego Carrasco: dbc86 - Juan Felipe Gonzalez: jg2265

***Abstract*—What makes a restaurant successful? With this project we want to predict the rates of restaurants based on their characteristics, and comments gathered from the web to help restaurant owners to understand their industry, extract insights to improve the current rating, and even predict the future rates of a restaurant before opening.**

I. Data Analysis

A. Data collection

In recent years, several rating web services have emerged. For instance, when we started to do some research, we found that Yelp released a subset of their reviews to encourage students to analyze and obtain interesting insights about the restaurant market. Not surprisingly, Yelp dataset has already been used in various research papers. Consequently, gathering information from TripAdvisor will allow us to compare our results with some existing work and take advantage of the infinite amount of data that TripAdvisor represents. Moreover, we decided to focus on the New York area due to the size of the restaurant market.

To extract TripAdvisor information, we started by studying the website source code (cf. Fig. 1). Following this, we coded a Python scraper (cf. Annex 1) to collect the HTML tags contents by automatically crawling the website pages. During this phase, it was crucial to randomly select the target restaurants as we wanted to have an overall view of the New Yorker restaurant market. Indeed, if we had decided to only collect the comments of the first restaurants presented in the TripAdvisor website, we would only have had comments related to the best places. Moreover, to select target information from our extracted text we used data treatment techniques such as regular expressions (cf. Fig. 1). A regular expression is a sequence of characters defining a pattern that it is used to search information in a text.

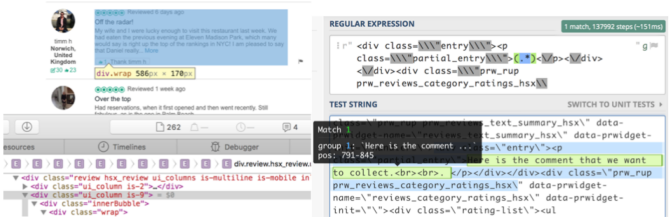

Fig. 1. TripAdvisor source code and extraction of a comment using a regular expression

B. Data Characteristics

We created a dataset composed of 71,800 rows (user reviews) corresponding to 1436 restaurants (50 comments per restaurant) and 11 features. During our collection process, we forced our scraper to collect the same amount of comments for each restaurant in order to prevent overfitting for some specific restaurants.

The features set is composed of 7 categorical variables and 4 nominal variables. The categorical variables describe restaurant features such as types of food or special occasions. The target variable is the restaurant rate. However, in order to predict this variable, we first need to analyze our data.

Firstly, we analyzed missing values. We erased the twenty-two rows among our data set that did not contain any comment. Then, we realized that some data were missing for three of our features: food, value and service ratings. Indeed for some comments the customers only put an overall rate and not categorical rates:

Fig. 2. Missing values for the categorical rates

To deal with these missing values, we thought about four methods: dropping the rows, replacing the missing rates by 0, replacing them by the overall grade of the comment or finally computing the average categorical rate of each restaurant and replacing the missing rates by the average. In order to make a choice between the different methods, we decided to perform a k-fold (k=2) cross validation with our linear model that will be explained later. The results led us to choose the last method, i.e. replacing the missing rates by the average categorical rate of the corresponding restaurant.

C. Data visualization

First of all, to get a sense of our data distribution, we outputted the summary statistics. However, we needed to look deeply into it to better understand it. For instance, it was crucial to visualize the distribution of our restaurants according to our target variable, i.e. rates. Figure 4 allowed us to observe that the distribution of the collected restaurants according to rates could be approximated by a normal distribution with a mean of 4.1. Moreover, it helped us to visualize one of the major limits of our dataset. Users reviews with lower rates (i.e 1 and 2) are rare among TripAdvisor comments. Moreover, we used several histograms and plots in order to visualize the feature variables that could explained more distinctly a restaurant rate. For instance, we found that the characteristic Number of food types proposed by the restaurants may have an impact on the customer's experience and thus on the rates that they give. As it can be observed in Figure 4, there is an overall trend that suggests that the average rate of a restaurant is going to increase with the number of food types proposed by the restaurant. Therefore, this feature may be a good predictor of a restaurant rate and should be taken into account.

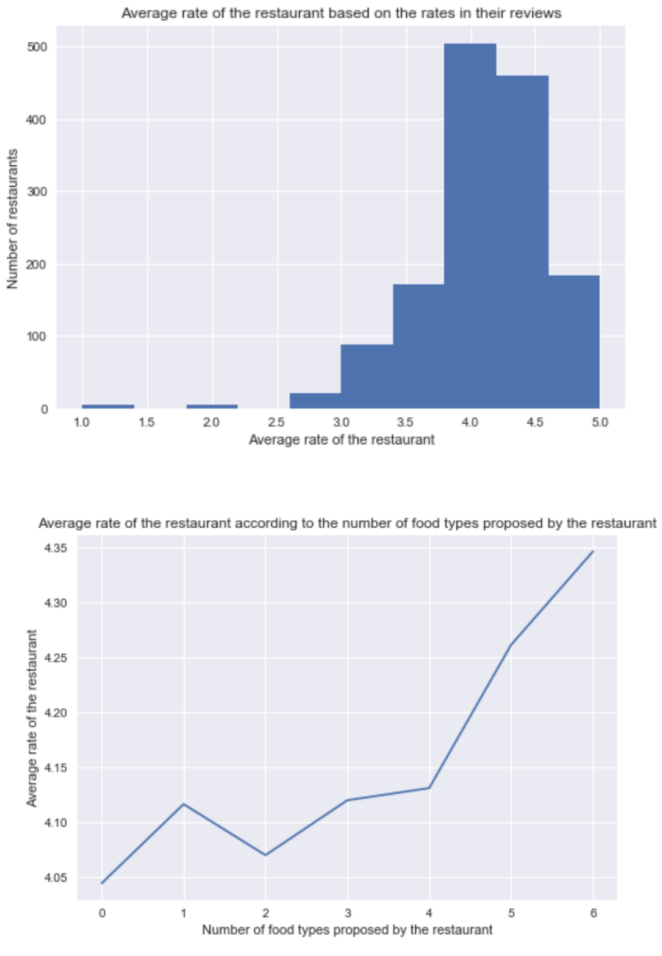

Fig. 3. Data visualizations

Besides, to include the users comments in our model, we wanted to visualize the most frequent words used in comments and observe if they could be good predictors. To do so, we visualized the wordcloud associated to low/high rates and we selected the most frequent words. In Figures 5, we can observe that, according to the rates, the most frequent words are different. For instance, between the most representative words of bad rates we found terms such as bad whereas in the comments related to good rates we found other words.

Fig. 4. Wordclouds for low/medium-rate (3/5) and good-rate (5/5) restaurants

Once we visualized our data, we applied feature engineering to transform some of our categorical and nominal variables into binary values. Besides, we decided also to use the most representative words per rate and add a new feature corresponding to the frequency of these words in comments. After the cleaning process and feature transformations we got 37 features.

II. Model Selection

A. Linear regression

For our first family of models, our focus was on testing different combinations of features by using a linear regression to fit our data. Then, in order to select the best model, we evaluated them using k-fold cross-validation (k=2). In this part, we tested 4 different models, ranging from 4 to 37 different features and we used least squares as our loss function.

The MSE in Cross Validation (CV) for the 4 different models are shown in Table I.

As we can see, model 4 with the following features (Service, Food, Value, Number of Foods, and six different key-words) had a better performance in cross validation, even when its training error was higher. This means that our more complex models are overfitting the data while our model 4 is generalizing better.

TABLE I
MSE results for the CV of the linear regression models

Models	M1	M2	M3	M4
Training Error 1	0.093	0.077	0.077	0.082
Training Error 2	0.083	0.069	0.069	0.077
Cross-Validation error 1	0.100	0.100	0.100	0.098
Cross-Validation error 2	0.109	0.118	0.110	0.102
Mean CV Test Error	0.105	0.109	0.105	0.100

B. Decision trees

The second model that we decided to implement was a regression decision tree. In order to do so, we used the DecisionTreeRegressor package from the Scikit-learn library in Python. We know that if the maximum depth of the tree is set too high, the decision tree learns too fine details of the training data and learns from the noise, i.e. it overfits. That is the reason why, to prevent overfitting, we chose a reasonable number of nodes for our decision tree. In order to develop it, we decided to perform a k-fold cross validation (with k=2). To do so, we split our dataset into 2 sets (T1 and T2) and we tested 3 different decisions trees (with respectively depth of 2, 3 and 4). By computing the MSE of each model when we tested them in the subset that was not used for the training (cf. Table II) we identified the best decision tree, which is the model M2 (depth of 3).

TABLE II
MSE results for the CV of the decision tree models

Models	M1	M2	M3
Training Error 1	0.115	0.090	0.107
Training Error 2	0.138	0.102	0.121
Cross validation error 1	0.144	0.122	0.140
Cross validation error 2	0.158	0.133	0.148
Mean CV Test Error	0.151	0.1275	0.144

On our best decision tree, we can observe that for example the feature Good, which is a common word used in the comments, has a significant impact. This observation encouraged us to include comments in our future model in order to be able to predict the overall rate without the need to have the three categorical rates.

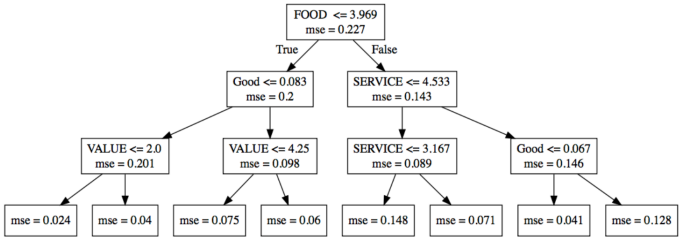

Fig. 5. Best decision tree model

C. Results

To test the effectiveness of our best model from each family, we used the 20% of the data that we reserved as final test set.

TABLE III
MSE FINAL RESULTS

Models	Linear regression	Decision tree
Test error	0.199	0.223

As we can see, the MSE's incremented as expected (from around 0.1 to around 0.2). However it is still a reasonable value considering that we are trying to predict rates from 1-5. In Figure 7 we can appreciate a normed histogram of our best lineal model compared against the true distribution of the test set.

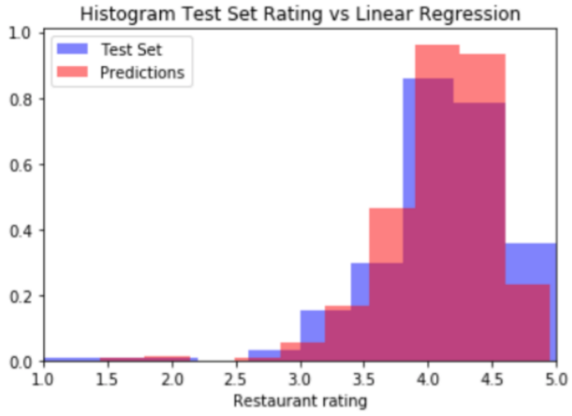

Fig. 6. Histogram of the test set rates vs. Predicted rates using the linear regression model

III. NEXT STEPS

A. Linear model improvements

As a first conclusion, we can say that our linear regression model is reasonably accurate. However, we still want to get better results. To start with, we already saw that low graded restaurants are underrepresented in TripAdvisor site, so we want to collect more data to improve our model and allow good low rate predictions. For this purpose, we will run our scraper in a Cornell server. Besides, we want to test adding a regularization parameter to our loss function. By implementing regularization we will be able to drop some of our irrelevant features and in general avoid overfitting.

B. Text mining model

Our final goal is to be able to predict rate without the categorical rates. That is the reason why we decided to include comments in our dataset. To exploit them, we chose to implement the Latent Dirichlet Allocation (LDA) model. LDA is a topic model and works as follow: if the observations are the words collected in comments, the LDA assumes that each comment is a mixture of a small number of topics, each topic is a mixture of words and each word has a probability associated to the topic. To implement this model, we used the lda package in the gensim library in Python.

In order to visualize our results, we used the t-distributed Stochastic Neighbor Embedding (t-SNE). This tool is clustering data in order to allow high-dimensional data visualization. Therefore for each comment we have some (X, Y) coordinates that are assigned.

```
**** Topic  28  *****

0.123*"friendly" + 0.113*"atmosphere" + 0.097*"charming" + 0.074*"pleasant" +
0.056*"nice" + 0.007*"kind" + 0.005*"warm" + 0.005*"convivial" + 0.004*"good"
+ 0.004*"sympathetic"
```

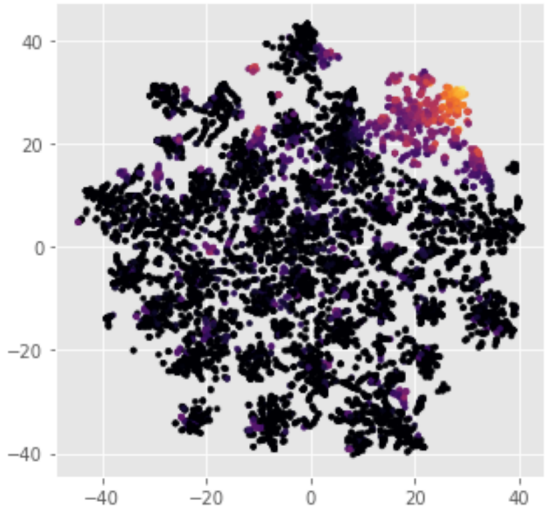

Fig. 7. Example of a topic that we created and the associated heat map

From the words contained in the above topic, we could assume that this is a topic related to a positive experience and therefore a good rate. In the visualization plot, we can observe that the restaurants that are significantly related to this topic are clustered in the upper left of the graph. Consequently, we decided to analyze restaurants of this cluster. In order to do so, we selected restaurants with constrained coordinates ($20 \leq X \leq 28$ and $22 \leq Y \leq 28$). Therefore we have been able to observe that the average rating of this cluster is equal to 4.73, which is a high value compared to the mean (= 4.1). We could therefore conclude that some topics may have a significant impact on the overall rate of each restaurant. That is why in the future we are aiming to add t-SNE coordinates in our features to improve our model and finally predict the overall rate without the categorical rates. We will also take a look at other text mining models and analyze them to choose the one that best fits our data without overfitting.