Final Report - Predict Restaurant Ratings

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

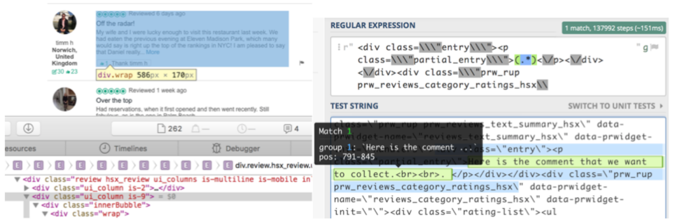
***Abstract*—What makes a restaurant successful? With this project we want to predict the ratings of restaurants based on their characteristics, and comments gathered from the TripAdvisor website to help restaurant owners to understand their industry, extract insights to improve the current rating, and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


Reviewed 6 days ago
Off the radar!
My wife and I were lucky enough to visit this restaurant last week. We had eaten the previous evening at Eleven Madison Park, which many would say is right up the top of the rankings in NYC! I am pleased to say that Daniel really... More
Thank timm h
timm h
Norwich, United Kingdom
30 23
div.wrap 586px × 170px
Reviewed 1 week ago
Over the top
Had reservations, when it first opened and then went recently. Still
262
4
esources
Timelines
Debugger
div.review.hsx_review.
REGULAR EXPRESSION
1 match, 137992 steps (~151ms)
TEST STRING
SWITCH TO UNIT TESTS
Match 1
group 1: `Here is the comment ...`
pos: 791-845

```
<div class="review hsx_review ui_columns is-multiline is-mobile in
  <div class="ui_column is-2">…</div>
  <div class="ui_column is-9"> = $0
    <div class="innerBubble">
      <div class="wrap">
```

```
r" <div class=\\\"entry\\\"><p
class=\\\"partial_entry\\\">(.*)<\/p><\/div>
<\/div><div class=\\\"prw_rup
prw_reviews_category_ratings_hsx\\
```

```
class=\"prw_rup prw_reviews_text_summary_hsx\" data-
prwidget-name=\"reviews_text_summary_hsx\" data-prwidget-
class=\"entry\"><p
class=\"partial_entry\">Here is the comment that we want
to collect.<br><br>. </p></div></div><div class=\"prw_rup
prw_reviews_category_ratings_hsx\" data-prwidget-
name=\"reviews_category_ratings_hsx\" data-prwidget-
init=\"\"><div class=\"rating-list\"><ul
```

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

B. Data Characteristics

We created a dataset composed of 405,950 rows (user reviews) corresponding to 8,119 restaurants (50 comments per restaurant) and 12 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.

On the one hand, the features set is composed of 8 categorical variables and 4 nominal variables.

The categorical variables describe features such as:

- Restaurant characteristics: Neighborhood, number of types of food, dinning styles (e.g. *Breakfast, brunch, lunch, dinner*), the *good for* section (i.e. Occasions for which the restaurant is good for. For instance, a restaurant can be good for families or for business meetings) and other restaurants features (e.g. *Wi-Fi, Wheelchair Accessible, Reservations*).
- User reviews ratings related to specific fields such as the price-quality ratio, the service and the quality of the food.

The nominal variables describe features such as the restaurant address, the name of the restaurant, the review title and the users reviews.

On the other hand, the target variable is the restaurant rating. For a specific restaurant, this rating is a 1-5 star continuous rating, which corresponds to the average of the 50 reviews ratings that we collected for that restaurant. In the reality, this variable indicates the average customer satisfaction level for that restaurant.

However, in order to predict this variable, we first needed to analyze our data.

Firstly, we analyzed the missing values. We erased the 185 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 rating and not categorical ratings:

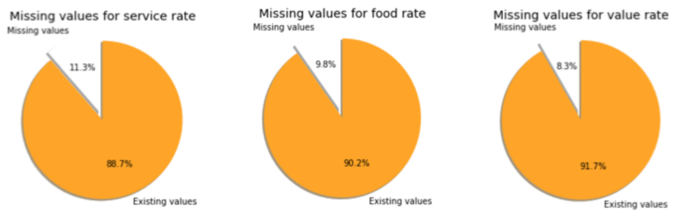


Fig. 2. Missing values for the categorical ratings

To deal with these missing values, we thought about four methods: dropping the rows, replacing the missing ratings by 0, replacing them by the overall rating of the comment or finally computing the average categorical rating of each restaurant and replacing the missing ratings 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 ratings by the average categorical rating 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. their average ratings. Figure 3 allowed us to observe that the distribution of the collected restaurants according to their average ratings is heavily skewed to the right. Indeed, when we look at our dataset, we can notice that it contains around 70% of restaurants with an average rating between 4-5, whereas the remaining 30% of restaurants from our dataset have an average rating that is in the ranges 1-2, 2-3 and 3-4 (cf. Figure 4). This observation can be confirmed by other studies such as the analysis made by Max Woolf[1] on Yelp reviews through which it was shown that lately Yelp reviews have started to appear more optimistically biased (skewed to the 4-5 ratings). Moreover, it helped us to visualize one of the major limits of our dataset. Users reviews with lower ratings (i.e 1, 2 and 3) are rare among TripAdvisor comments.

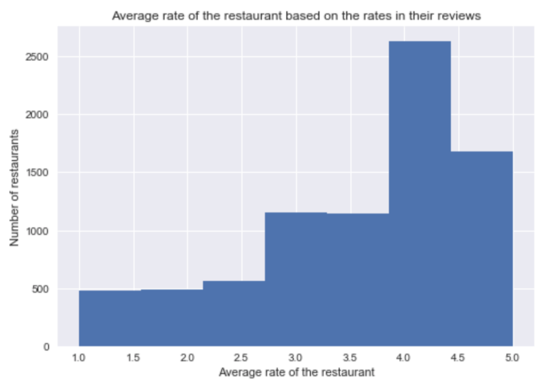


Fig. 3. Average rating of the restaurant based on the ratings in the related reviews

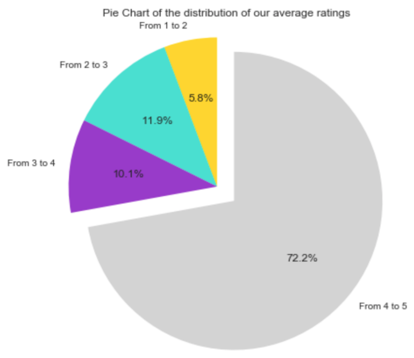


Fig. 4. Pie Chart of the distribution of our average ratings

In addition to this, we also used data visualization to understand our feature variables. However, to do so, we had to apply feature engineering to transform some of our categorical variables into binary values (cf. End of the section).

Among these visualizations, we plotted the correlogram of all our features and the target variable in order to highlight the most correlated variables, but also, the feature variables that could explain more distinctly a restaurant rating (cf. Fig. 5.).

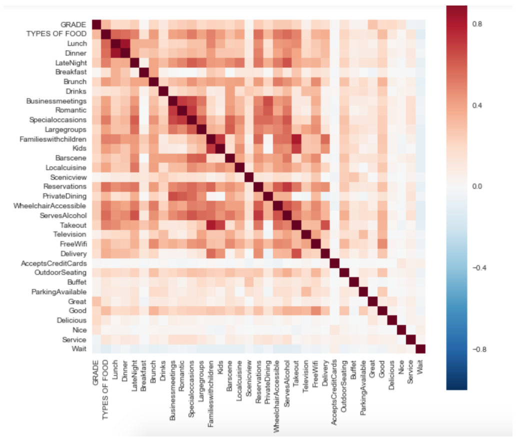


Fig. 5. Correlogram of all our features and the target variable

For instance, we observed that characteristics such as the number of types of food proposed by the restaurants may have an impact in the ratings. Following this, we plotted the average ratings according to the ”number of types of food” proposed by the restaurants (cf. Figure 6). As it can be observed, there is an overall trend that suggests that the average rating of a restaurant increases with the number of types of food. Therefore, this feature may be a good predictor of a restaurant rating and should be taken into account.

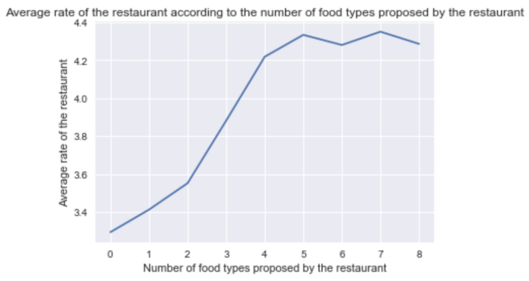


Fig. 6. Average rating of the restaurant according to the number of types of food proposed by the restaurant

Furthermore, we also visualized the impact of our nominal values. For example, to understand the impact of the neighborhood, we visualized the ratings distribution on the New York map.

As it can be observed in Figure 7, we can notice some clusters of good rated (represented in green) and bad rated (represented in red and orange) restaurants according to the neighborhoods. Thus, the restaurant location may be also a good predictor of a restaurant rating.

Besides, we wanted to include the users comments in our model. Thus, we decided 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 ratings and we selected the most frequent words. In Figure 8, we can observe that, according to the ratings, the most frequent words are different. For instance, between the most representative words of bad ratings we found terms such as awful whereas in the comments related to good ratings we found other words.

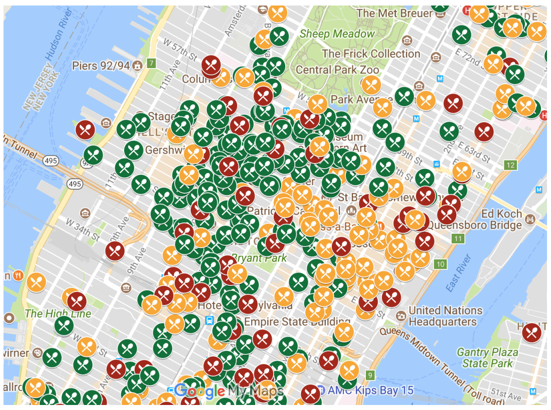


Fig. 7. Ratings distribution on the New York map - Manhattan view

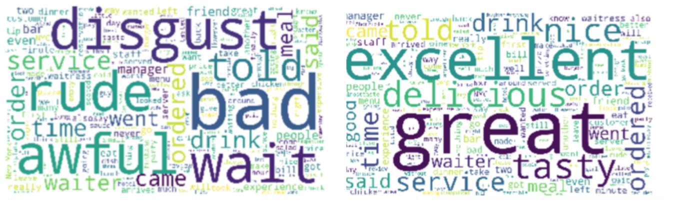


Fig. 8. Wordclouds for low/medium-rate (3/5) and good-rate (5/5) restaurants

As a result, we decided to use the most representative words in the comments (e.g *wait, great, good, service, nice, delicious, tasty, rude, awful, disgust, bad, excellent*) and add features corresponding to their frequencies in the comments. Thanks to this first analysis, we realized the importance of words in the ratings prediction. That is the reason why we decided to apply some text mining techniques to build new comment-related features that could improve our model accuracy.

II. COMMENT-RELATED FEATURES

A. Latent Dirichlet Allocation (LDA)

Our final goal is to be able to predict ratings without the categorical ratings, 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 follows: 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.

The first step before performing the LDA model is to pretreat the comments. In order to do so, we first removed the extremely common words that are not useful for our model (such as a, an, are, the...). The significance of a word is evaluated from its distribution in a collection of texts. In other words, a word which appears with a similar frequency in each of the texts is not discriminating and it does not add extra information to the text. These words are called stop words and there is a package in python called get_stop_words from the stop_words library that allowed us to get rid of them. After that, the comments were ready to build the LDA model.

To implement the model, we used the lda package in the gensim library in Python. The only parameter we needed to fix is the number of topics. To choose the best value of this parameter we performed a cross validation that will be explained later in details (cf. section *III.A Linear regression*. We combined all the comments of a single restaurant in a larger text. By doing so, we increased the length of the corpus allowing us to better classify our restaurants. Therefore, in our case, after applying our LDA model each restaurant is characterized by one or more topics (e.g. Restaurant 23 belongs for 60% to topic 1, 30% to topic 6 and 10% to topic 11)(cf. Figure 10).

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 restaurant we have some (X, Y) coordinates that are assigned.

Here is an example of a topic that we created and the associated heat map:

```
**** 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"
```

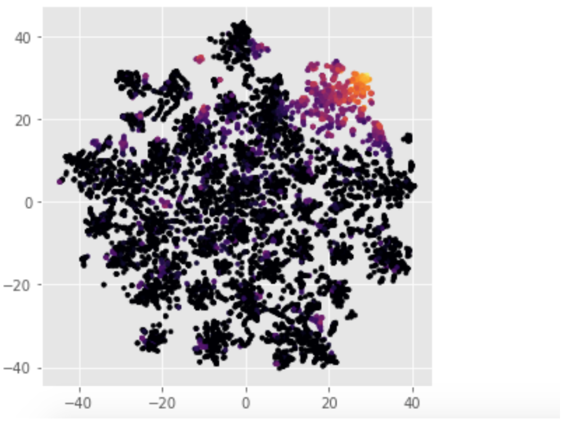
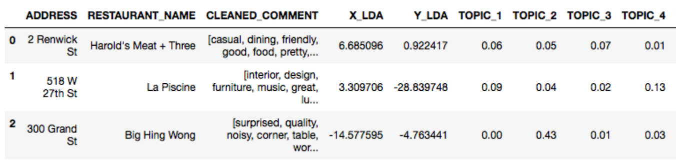


Fig. 9. Example of a topic that we created and the associated heat map

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 rating. 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 4.93, 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 rating of each restaurant and that topics are thus worth being taking into account.

That is the reason why we decided to add, for each restaurant, the belonging percentage for each topic as features. In order to determine the best number of topics for our model, we decided to perform a k-fold cross validation (with k=2) with our linear model that will be explained later using all the available features. To do so, we split our dataset into 2 sets (T1 and T2) and we tested 4 different values for the number of topics (25, 30, 35 and 40 topics). By computing the MSE of each model when we tested them in the subset that was not used for the training we identified the best value for the parameter, which is equal to 30 topics.

Finally, we applied our LDA model with 30 topics giving us the following table for the first 4 topics and the associated percentages for the first 3 restaurants:



	ADDRESS	RESTAURANT_NAME	CLEANED_COMMENT	X_LDA	Y_LDA	TOPIC_1	TOPIC_2	TOPIC_3	TOPIC_4
0	2 Renwick St	Harold's Meat + Three	[casual, dining, friendly, good, food, pretty,...	6.685096	0.922417	0.06	0.05	0.07	0.01
1	518 W 27th St	La Piscine	[interior, design, furniture, music, great, lu...	3.309706	-28.839748	0.09	0.04	0.02	0.13
2	300 Grand St	Big Hing Wong	[surprised, quality, noisy, corner, table, wor...	-14.577595	-4.763441	0.00	0.43	0.01	0.03

Fig. 10. Table with features related to LDA

B. *Word2Vec + K-Means*

Another text mining technique that we wanted to try on our dataset was Word2Vec, a word embedding model. Word embeddings is a technique that expresses a word as a real number vector of low dimension from their distributional properties observed in a collection of documents. Words that have similar meaning can be made to correspond to close vector and obtain meaningful results (e.g. king − man + women = queen) by adding or subtracting vectors. As a result, the word vectors are positioned in the vector space so that the words that share common contexts are located close to each other in the space.

To perform our model, we also used the pretreated comments that we used for our LDA model. We built our new model using the Word2Vec package from the gensim library in Python. Once again, in order to visualize our results, we used the t-distributed Stochastic Neighbor Embedding (t-SNE). This technique gave us the following graph: On the

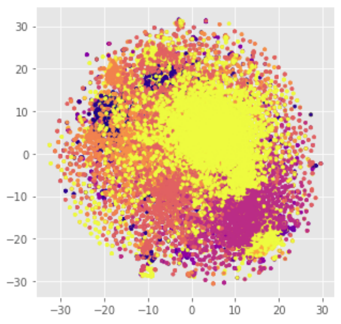


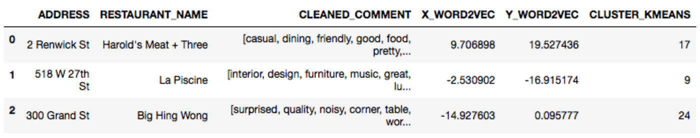
Fig. 11. Visualization of Word2Vec model with t-SNE coordinates

above graph, we can observe many clusters. An interesting observation that we have been able to make was that the clusters were different from the one we obtained using our LDA model. It reveals that Word2Vec allows to make another type of classification that may be worth being taking into account. In order to be able to use these results in our dataset, we applied a K-means clustering method on our Word2Vec model.

K-means is one of the most popular clustering algorithms. K-means stores k centroids (points which are the center of a cluster) that it uses to define clusters. A point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other centroid. In order to find the best centroids, K-means alternates between assigning data points to clusters based on the current centroids and between choosing centroids based on the current assignment of data points to clusters. To perform this technique, we used the KMeans package from the sklearn.cluster Python library on our Word2Vec model.

Both K-means and Latent Dirichlet Allocation (LDA) are unsupervised learning algorithms, where the user needs to decide a priori the parameter K, respectively the number of clusters and the number of topics. If both are applied to assign K topics to a set of documents, the most evident difference is that K-means is going to partition the documents in K disjoint clusters (i.e. topics in this case). That is the reason why, similarly to what we did for LDA, we decided to perform a k-fold cross validation in the same way as we did for LDA. By computing the MSE for each value of the parameter we identified the best value which is equal to 35 clusters in this case (compared to 30 topics for LDA).

Then, we applied our K-means clustering method with 35 clusters giving us the following table for the first 3 restaurants:



	ADDRESS	RESTAURANT_NAME	CLEANED_COMMENT	X_WORD2VEC	Y_WORD2VEC	CLUSTER_KMEANS
0	2 Renwick St	Harold's Meat + Three	[casual, dining, friendly, good, food, pretty,...	9.706898	19.527436	17
1	518 W 27th St	La Piscine	[interior, design, furniture, music, great, lu...	-2.530902	-16.915174	9
2	300 Grand St	Big Hing Wong	[surprised, quality, noisy, corner, table, wor...	-14.927603	0.095777	24

Fig. 12. Table with features related to Word2Vec & K-Means

We know that K-means clusters are not equally separated in space (i.e. you do not have the same distance apart between cluster 1 and cluster 2 for example), that is the reason why we performed a feature transformation on the CLUSTER_KMEANS feature to transform it into 35 binary features (one for each cluster). The difference compared to LDA, as the clusters are disjoint, is that for a specific restaurant we only have the value 1 for one of the 35 clusters and 0 for the remaining clusters.

C. *Final features*

Finally, once we visualized and analyzed our data, we applied feature engineering to transform some of our other features in order to incorporate them in the future steps. From the 12 starting features, we defined a total of 127 features that can be divided in three different subsets:

- 3 features associated with the user reviews ratings of specific fields: the price-quality ratio, the service and the quality of the food.
- 47 features associated to the restaurants characteristics: we applied feature engineering to transform the starting categorical variables (e.g. neighborhoods, ”good for” section, dinning styles) into binary values (e.g.*dinner, lunch, brunch*).
- 77 features related to the users comments: firstly, we decided to use the 12 most representative and significant words in the comments (e.g *wait, great, good, service, nice, delicious, tasty, rude, awful, disgust, bad, excellent*) and add features corresponding to their frequencies in the comments. Finally, we added the percentage for every LDA topic and the binary K-means cluster features for each restaurant in our dataset.

III. Model Selection

In this project we implemented linear regression and decision trees. We decided to compare them to a baseline scenario to give perspective on the results. We computed a straightforward baseline which is the average rating of all the New York restaurants.

A. Linear regression

As we are trying to predict a variable that has a lower and upper bound (1 and 5), we had to make sure that our predicted variable was contained in this interval. In order to do so, we generated the prediction and after we applied the truncation accordingly ($= 1$ if $y < 1$ and $= 5$ if $y > 5$).

i) Loss Functions

Quadratic Loss function: As our initial predictive model, we implemented a quadratic loss function without regularization. The quadratic loss has the nice property of being differentiable and symmetric, meaning that it penalizes the same an error above or below the target. This model will be useful as reference for more complex models. The quadratic loss is expressed as follows:

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i)^2$$

l_1 ***Loss function:*** After testing a quadratic regression, we decided to use also an l_1 loss. The motivation for doing so is that the l_1 loss penalizes more the errors smaller than one. The l_1 Loss function is defined as follows:

$$\frac{1}{n}\sum_{i=1}^{n}|y_i - w^T x_i|$$

ii) Regularizers

l_1 ***regularizer:***

As we mentioned at the beginning, one of our objectives is to provide information for restaurant owners of how to improve their ratings. That is the reason why we implemented the lasso regularizer in order to achieve a sparse solution and identify the main factors contributing to a good rating. The l_1 regularizer is defined as follows:

$$r(w) = \lambda\sum_{i=1}^{n}|w_i|$$

l_2 ***regularizer:*** We also decided to include a quadratic regularizer into our model. The intuition behind is to achieve smaller coefficients that generalize better. Even if it does not produce sparse solutions, we believe that it will produce more robust predictions. The l_2 regularizer is defined as follows:

$$r(w) = \lambda\sum_{i=1}^{n}w_i^2$$

Elastic Net regularizer: The Elastic Net Regularizer linearly combines the l_1 and l_2 penalties of the lasso and ridge methods giving us the following formula:

$$r(w) = \lambda_1\|w\|_1 + \lambda_2\|w\|_2^2$$

The elastic net enforces sparsity, it has no limitation on the number of selected variables and it encourages grouping effect in the presence of highly correlated predictions. One major con is that naive elastic net suffers from double shrinkage which leads to increased bias and poor predictions. One way to correct it is to multiply the estimated coefficients by (1 + λ_2).

Using the defined loss functions and regularizers, we decided to test 8 different objective functions:

- Model M1: l_1 loss function without regularizers
- Model M2: Quadractic loss function without regularizers
- Model M3: l_1 loss function with l_1 regularizer
- Model M4: l_1 loss function with l_2 regularizer
- Model M5: Quadractic loss function with l_1 regularizer
- Model M6: Quadractic loss function with l_2 regularizer
- Model M7: l_1 loss function with the elastic net regularizer
- Model M8: Quadractic loss function with the elastic net regularizer

To optimize our models, we used different algorithms. On the one hand, for the smooth optimization models (M2 and M6), we set the gradient to 0 and we solved the resulting system of equations. On the other hand, for the models containing a differentiable part and a part with an easy prox function (all the other models except models M2 and M6), we used the proximal gradient algorithm.

Moreover, in order to test the significance of our features, we decided to use 4 different feature subsets from our dataset. First, we only used the 3 categorical rates (S1) and then we added the restaurants characteristics (S2) to see if the accuracy of our models could be improved. Then we also added the comment-related features (S3) to see how our MSE would evolve. Finally, we erased the categorical ratings from our features in order to see how our models would perform with only the restaurants characteristics and the comment-related features (S4).

To study these linear regression models, we divided our dataset into a training set (80% of the dataset) and a test set (remaining 20%). Following this, we decided to perform a k-fold cross validation (with k=2). To do so, we split our training set into 2 sets (T1 and T2).

Using cross-validation allowed us to:

- Calculate the best λ for the models in which we applied regularizers (cf. Figure 13)
- Compare the performance of the 8 linear regression models (cf. Table I)

On figure 13, we can observe that the smallest MSE that we obtained doing our cross validation for the quadratic loss with a l_1 regularizer is reached for $\lambda = 0.97$. We followed the same method in order to determine all the others λ.

On Table I, we can observe the MSE of the best models with the optimal parameters that we found performing cross validation explained earlier.

We can see on Table I that for each model, when we are adding the characteristics, the MSE decreases. This is even more noticeable when we are adding the comment-related features in our dataset. These results show that the characteristics and comments are helpful for our prediction problem.

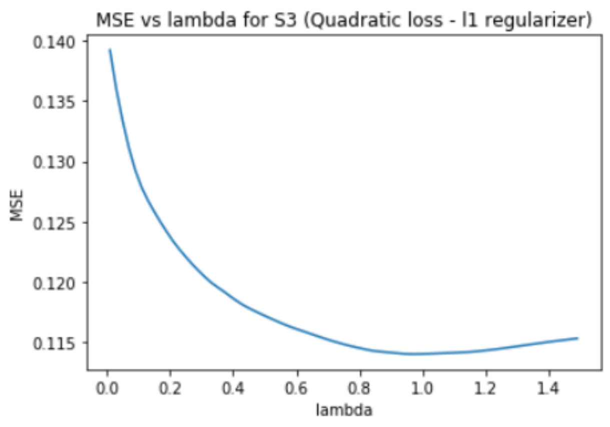


Fig. 13. MSE vs lambda for the quadratic loss with l1 regularizer

TABLE I
CROSS-VALIDATION MSE

Models	S1	S2	S3	S4
M1	0.1265	0.1121	0.1098	0.1747
M2	0.1326	0.1217	0.1145	0.1634
M3	0.1213	0.1076	0.1056	0.1521
M4	0.1298	0.1186	0.1106	0.1717
M5	0.1313	0.1209	0.1136	0.1626
M6	0.1318	0.1211	0.1141	0.1629
M7	0.1225	0.1082	0.1063	0.1526
M8	0.1317	0.1210	0.1139	0.1537

Furthermore, we can notice that when we are removing the categorical ratings from our features, even if the MSE is obviously higher than the one with all the features, it remains reasonably small. This result confirms that we are confident about building a good prediction model based on restaurants characteristics and comments.

As we can see on Table I, our model with the l_1 loss and lasso regularization performs consistently better than the other models. This could be explained by the fact that by using the l_1 loss function we are penalizing more the small errors that with the other tested loss function. Knowing that our prediction is a continuous variable contained in the [1, 5] interval, it makes sense because small errors can be interpreted as large ones in our prediction problem.

Concerning the regularizer, we have been able to visualize on the correlogram that some features variables may be inter-correlated. In addition to this, there is also some features that appeared not to be correlated with our prediction variable. However, we cannot erase them because uncorrelated variables can actually be very useful in the model. Erasing features only based on correlation can lead to a loss of accuracy in our model. That is the reason why by using lasso regularizer we are identifying an optimal subset selection that improves our predictions. Therefore, we selected this model for our test phase.

For the other models that we decided to implement, we are going to only consider the restaurants characteristics and comment-related features because one of our objectives is to be able to predict restaurants ratings without the use of categorical ratings.

B. *Decision trees*

Decision trees

The third 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. Moreover, to implement it, we divided our dataset into a training set (80% of the dataset) and a test set (remaining 20%). For this model, we only used the restaurants characteristics and the comment-related features as we mentioned previously.

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 depth for our decision tree.

In order to find the best decision tree, we decided to perform a k-fold cross validation (with k=2) in the same way that we did it in section *III. A Linear regression* (cf. III.A for explanation). We tested different values for the max_depth parameter (maximum depth of the tree) and at the end we identified the best decision tree, which was the M3 model (depth of 4). Indeed, this tree has the lowest cross-validation MSE (cf.Table II). The left branch of this tree is represented in Figure 13.

TABLE II
PARAMETER SELECTION

Models	Cross validation MSE
M1 (max_depth = 2)	0.1712
M2 (max_depth = 3)	0.1556
M3 (max_depth = 4)	0.1723
M4 (max_depth = 5)	0.1843

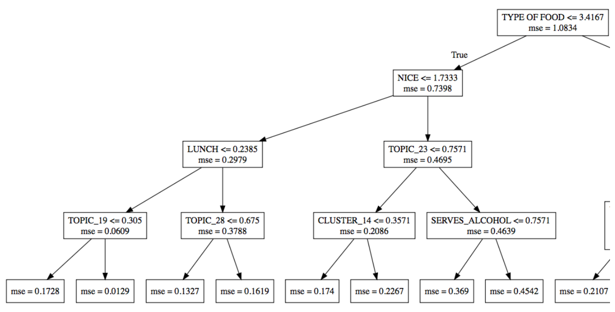


Fig. 14. Left branch of the Decision Tree with a depth of 4

On the above tree, we can observe that some restaurants characteristics and comment-related features are of a significance importance. For example, we can see, that the type of food, the lunch and the alcohol features are appearing on our decision tree. Concerning the comment-related features, we can observe that some topics (e.g. Topic 19, 23 and 28), some clusters (e.g. Cluster 14) and some words (e.g. nice) have a high impact on our prediction model.

Random forest

Another model that we took into account to solve this regression problem was the random forest. To apply this model, we divided our dataset into a training set (80% of the dataset) and a test set (remaining 20%). The way this ensemble learning method works in our application is that we fit a certain number of decisions trees (n_estimator parameter) on various bootstrap sub-samples of same size coming from our training set and output the average prediction of those individual trees. Each of the individual trees used a certain number of features at random (max_features parameter) to fix the samples.

We decided to apply this method to our problem in order to improve the model accuracy but also to control underfitting and overfitting. On the one hand, random forest prevents underfitting since decision trees tend to split the data until reduce the training error. On the other hand, by setting a minimum number of samples required to be at leaf node (min_samples_leaf parameter) we prevent each individual tree from overfitting. Moreover, by outputting the average prediction of several individual trees, we prevent to use trees that heavily overfit our data.

For this model, we only used the restaurants characteristics and comment-related features as we mentioned earlier. To apply it, we used the RandomForestRegressor package from the Scikit-learn library in Python. However, in order to use this package, we needed to set the different parameters of the model. Concerning the min_samples_leaf parameter we decided to set it to the default value of 5 (cf. Reference [2] Data Mining and Business Analytics with R). However, in order to choose the n_estimator and the max_features parameters, we compared the out-of-bag (OOB) errors for different values of the max_features parameter according to different values for the n_estimator parameter. As it can be observed in the Figure 15, the lowest OOB error was obtained by setting the max_features parameter to p/3 (where p=124 is the total number of possible features). It is important to notice that this result is consistent with the recommended value of the best max_features parameter from other regression studies (cf. Reference [2]). Moreover, this figure allowed us to approximate an accurate value of the n_estimator parameter. This value for which the OOB error stabilizes, i.e, n_estimator equal to 12.

Therefore, we are going to consider that the best model given by the random forest technique has a n_estimator parameter equal to 12 and a max_features parameter equal to p/3 (where p=124 is the total number of possible features). In the next section we are going to test it in our test set.

C. Results

Let's recall our best models:

- MF1: Best linear regression model - l_1 loss function with lasso regularizer
- MF2: Best decision tree - Decision tree with depth of 4
- MF3: Best random forest - Random forest with n_estimator parameter equal to 12 and a max_features parameter equal to p/3 (where p=124 is the total number of possible features)

To test the effectiveness of our best models we used the remaining 20% of the data that we reserved as final test set.

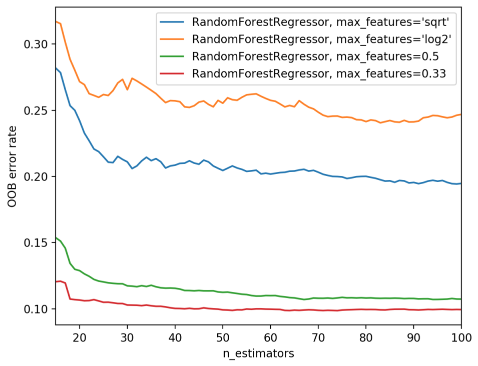


Fig. 15. Out-of-bag error according to different values for the n_estimator and max_features parameters

TABLE III
MSE FINAL RESULTS

Models	Baseline	MF1	MF2	MF3
Test error	1.0186	0.2103	0.2419	0.2237

On the table III, we tested our best models in the test set and we computed the MSE's. 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 comparing to the MSE of our straightforward baseline and considering that we are trying to predict ratings ranging from 1 to 5.

At the end, our best model was the linear regression model for which the objective function was the loss function l_1 with the l_1 regularizer. As it was already explained in section *III.A Linear regression*, the fact that this objective function has a lower test MSE for our problem makes sense. Indeed, knowing that our prediction is a continuous variable contained in the small [1, 5] window, small errors can be interpreted as large ones and thus the l_1 loss function allowed us to take this into account. Besides, by using lasso regularizer we performed an optimal subset selection that enabled us to improve our predictions.

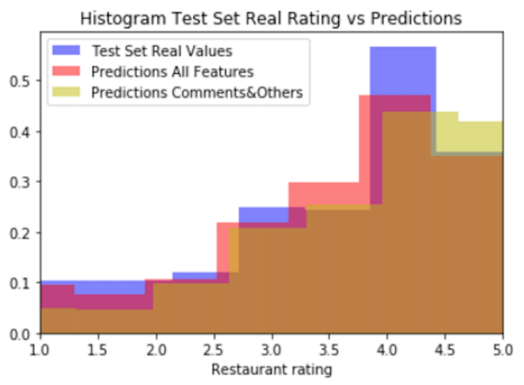


Fig. 16. Histogram of the test set rates vs. Predicted rates using the linear regression model with all the features (categorical ratings included) and with the restaurants characteristics + comment-related features

Furthermore, on the figure 16, we can appreciate a normed histogram of our best linear model predictions compared against the true distribution of the test set. When we look at the linear model that was trained on the dataset containing all the features, we can observe that it is performing well for low and high ratings. Nevertheless, the linear model that was trained on the dataset containing only the restaurant features and comments did not give so accurate predictions for extreme values. A possible explanation for this limitation is that the applied text mining techniques do not differentiate negative forms from positive ones. For example, if a comment contains the negative form *not great*, the applied text mining techniques are just going to identify the term *great* and it may be clustered in a positive topic. As we are going to detail in section *D. Accuracy and Limits*, this problem can be treated by applying sentiment analysis techniques that will take into consideration the negative and positive weights of words.

D. Accuracy and limits

As we previously saw in the report, the MSE that we obtained on the test set was not significantly higher than our training set error that we found using cross validation. This result shows that we are not facing an under or overfitting issue. In addition to this, as Yelp dataset has already been used in various research papers, we have been able to compare our results with some existing works (cf. References [3] Restaurants Review Star Prediction for Yelp Dataset). This analysis allowed us to conclude that our model is well performing and accurate in respect to others restaurant ratings prediction problems. We are therefore confident with the accuracy of our linear regression model.

Nevertheless, as we already mentioned it, we could try to continue to improve it by applying sentiment analysis techniques such as the N-grams method that would take the words polarity into account.

IV. CONCLUSION

With the results in hand, it is now time to return to our initial questions.

What makes a restaurant successful?

After performing several linear regression and choosing the best model, we took advantage of the properties of lasso regularization (subset selection) to identify the most relevant features contributing to a high restaurant rating. For our particular data set of New York restaurants, we could identify several features with a high impact in ratings.

The most relevant was the number of foods that a restaurant offers. This is remarkable and easily implementable; it would definitely be an interesting experiment for restaurants owners. The second most relevant feature that we found was free wifi, also an interesting thing to test. Some other impacting features were the outputs from lda and k-means analysis. This part is difficult to interpret for current restaurants but, as we are going to discuss later, it becomes very relevant for our next question. We also found that other features such as serving alcohol, or being a good place for a night out seem to have a positive impact on the final rating and are therefore worth investigating.

Can we predict the rating of a restaurant before opening?

The obvious complications of this question is that you dont have so much information of a restaurant that is not currently operating. However, we think its an interesting idea to explore because there are several methods to gather information of a prospective restaurant, for example focus groups or studying similar restaurants.

The interesting thing is that assuming that you are able to get comments of the food with a focus group as well as information of the characteristics of the restaurant (future location, services and amenities), our model still get useful predictions! With this analysis, we managed to get an MSE of 0.2103, which is considerable better than our baseline scenario and represent potential opportunities to restaurant owners to reduce risks on new inversion and tests different ideas before launching.

The most relevant features were similar to the last case, starting with number of foods and free wifi, but now the comments started to take more relevance. Using the lasso regularizer in our linear regression model helped us to notice some significant coefficients associated to specific topics.
For instance, we observe that the Topic number 23 from our LDA analysis has a high coefficient. Therefore, we decided to analyze this cluster and we found out that it was a music-related topic (containing words such as concert, music, song, atmosphere, guitar...). This result shows that a restaurant owner could focus on music to get a good rating.
On the other hand, we also observe some negative coefficients assigned to some topics. For example, Topic 19 has a negative coefficient in our lasso regression. Looking at this topic, we realized that it was related to overcrowded places (containing words such as noisy, crowded, cramped...).

As a conclusion, we hope that restaurants owners will find our prediction model helpful in order to improve customer's experience and get insights of their businesses.

V. REFERENCES

[1] *The Statistical Difference Between 1-Star and 5-Star Reviews on Yelp*, Max Woolf, 2014.

[2] *Data Mining and Business Analytics with R*, Johannes Ledolter, 2013.

[3] *Restaurants Review Star Prediction for Yelp Dataset*, Predicting Yelp Restaurant Reviews.