CS229 Project report:
Yelp Personalized Reviews

Thomas Palomares, Alexis Weill, Arnaud Guille

December 12, 2014

1 Introduction

Using data from Yelp Dataset Challenge, we wanted to provide personalized restaurant recommendations to users based on the reviews they previously left on Yelp. To do so, we used one user's past reviews and ratings, as well as other users' reviews and ratings of the establishment we are predicting the rating of. The training is therefore made for a specific user and the predictions based only on his or her specific preferences.

One of the main applications of this project is to be able to recommend restaurants that a user is likely to appreciate. This could lead to a new business opportunity for Yelp: they could partner with restaurants willing to offer discounts to users likely to enjoy the restaurant and make a commission for every positive recommendations. This business model is currently used by TheFork (LaFourchette), a european company launched in 2007 linking users and restaurants offering discounts through their app with TheFork taking a commission on every transaction. Offering this service without personalized recommendations, TheFork managed to expand to 4 countries in 7 years and was acquired by TripAdvisor in 2014 for $140M. Leveraging Yelp's data, such a service could easily be implemented and become a solid source of income for Yelp before other competitors enter this niche.

2 Dataset

Yelp has made a subset of its data publicly available in the context of the Yelp Dataset Challenge [1]. Data is provided for users, tips, reviews check-ins and businesses. There are 1.1 million reviews and 250,000 users in the dataset provided, from which we removed all the reviews that were not about restaurants.Then we only kept users that had given at least 20 reviews.

The data provided was exclusively in JSON format so we used JSONlab [2] to convert the data in MATLAB format. The conversion time was non-linear so we developed a code to break the data in smaller chunks, convert that data in MATLAB format, and recombine those pieces together.

While studying the data obtained, we quickly realized that most users had only posted a handful of reviews, offering only a very small training set given the way we set up our algorithms. As explained at greater length in the discussion section, a way to compensate for this challenge would be to use unsupervised algorithms on the every reviews to separate them into different categories. A user with few reviews can be associated with one of these bigger data set which the machine learning algorithms could be trained with.

3 Features and Preprocessing

For our main features, we considered all the features available for each restaurant: from the average rating of the restaurant, its price and the type of restaurant we were considering, to more subjective features such as the ambiance or whether the place is good for romantic dates. We then used feature selections algorithms to keep only the most relevant features as presented on figure 1 for the linear regression. The label considered is the rating attributed by the user to a restaurant.

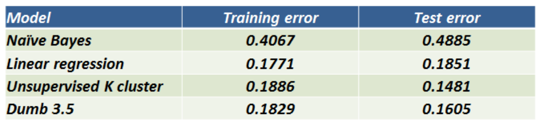
As discussed in details in the Future section, one of the difficulties of this approach is that many restaurants are missing some features in the Yelp dataset, making the implementation of the algorithms more challenging. This reduces the amount of data available and might skew our results.

4 Models

We have tested various models on the data: unsupervised algorithms (K-means with different number of clusters) and supervised algorithms (learning regression, Naive Bayes...). We then used feature selection algorithms to ensure that used relevant features.

5 First Results

To get representative results for our models, we decided to only consider users with the highest number of reviews (more than 400 reviews per user). We then split this training set for cross validation (70% - 30%) and computed the resulting errors according to the number of features considered. In the figures below we used 350 training examples and 81 testing examples. We considered that the algorithm was making an error if, when applied to a restaurant, the rating predicted was different by more than 0.8 than the actual rating given by the user. For instance, a predicted rating of 3.5 would be considered as good for an actual rating of either 3 or 4. We decided to compare our performance to results obtained by only using the median value of the ratings (3.5) as the prediction and called this model "Dumb 3.5". The results are presented on Figure 1 and 2.



Model	Training error	Test error
Naïve Bayes	0.4067	0.4885
Linear regression	0.1771	0.1851
Unsupervised K cluster	0.1886	0.1481
Dumb 3.5	0.1829	0.1605

Figure 1: results for the different algorithms

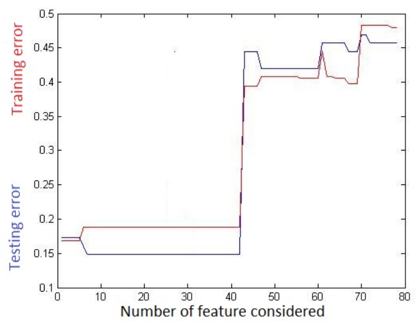


Figure 2: results for the different algorithms

We implemented different algorithms, either supervised or unsupervised learning and then applied feature selection to find the best set of feature for a specific user. While, at first glance, the results seemed correct for the linear regression and unsupervised algorithms, when comparing them to the "dumb" algorithm we realized that the errors were comparable.

In spite of many efforts to diversify the algorithms, considering labels ("0" or "1" for good and bad restaurants instead of actual ratings) or feature selections, the results of the algorithms were still very close to the ones of the dumb algorithm. We also observed that our algorithms, when optimised to reduce the errors, tended towards the value of the dumb algorithm. In the linear regression, most of the parameters were close to 0 while the intercept term was close to 3.3. As far as the unsupervised algorithm is concerned, the parameter "rating from the user" was generally similar among the clusters and close to 3.5. We then wondered if better results with the current data and features were really reachable.

6 New models

Because our results were disappointing, we decided to see if generalizing the problem to all users would generate better results. We tried to predict restaurant ratings based on its features. We decided to use k-nearest neighbors to get a sense of whether there was a way to group restaurants based on features to estimate their rating (see [3] for k-nearest neighbors). For features, we chose the categories the restaurants belonged to (such as italian or gastropub), the number of reviews, its price range and a handful of characteristics (parking, kids-friendly). We chose to round our real-numbered estimate and considered valid any guess that was 0.5 away from its actual average given by the users.

7 New results

We can see that the testing error of our k-nearest neighbors algorithm converges to around 0.25, as detailed on figure 3. Given that the distribution of ratings for all restaurants can be modeled by a normal distribution with an average around 3.5, this result is not much better than guessing the median, which could indicate that the features we had at our disposal were not good indicators of the value of a restaurant.

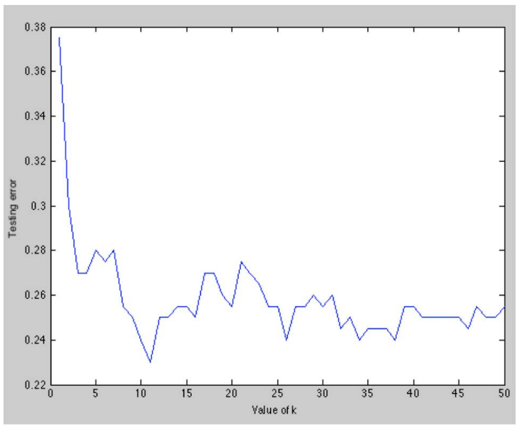


Figure 3: Testing error of k-nearest neighbors algorithm with varying k-values

8 Discussion

Based on our results for individualized ratings, we decided to look at generalizing review predictions based on similar features. We realized that we were getting similar results, which seems to indicate that the features we were considering might not be representative of the ratings left by users.

Despite using feature selection methods, we had trouble having good features to use for our algorithms. Part of this issue is due to the fact that there were only very few features that had data for all restaurants. Most of their data points are optional which makes our algorithms weaker when using those features.

9 Future

We believe that the features we used had low correlation to the reviews. To get better results, we have to either find better features or use a different approach.

One method would be to process the text of a user's reviews to get rid of all common words (such as "the" or "restaurant") and take the most common relevant words (such as "tortilla" or "patio") in the reviews to create new features that could capture information about a user's preference missed by our current features. We could extend this to all the restaurants reviews in the dataset in order to identify new relevant features and therefore increase the number of features we can use for each restaurant (a restaurant may have a new feature for its amazing tacos or french bread).

Using those new features, a new approach would be to apply unsupervised learning algorithms to form restaurant clusters. We will then suggest new restaurants to users based on the clusters of restaurants they liked before, similar to the process of product recommendation on Amazon.

From Yelp's perspective, it would be very useful to urge restaurants to fill the missing information on their page. Indeed, much information is currently missing about the restaurants characteristics we used as features for our algorithms, which impacts the effectiveness of our machine learning algorithms.

References

[1] Public data: *http : //www.yelp.com/dataset_challenge*

[2] JSON Lab: http://iso2mesh.sourceforge.net/cgi-bin/index.cgi?jsonlab

[3] Larose, D. T. (2005). kNearest Neighbor Algorithm. Discovering Knowledge in Data: An Introduction to Data Mining, 90-106.