Yelp Recommendations

Gavriel Adler
Carnegie Mellon University
gya@andrew.cmu.edu

Spencer Barton
Carnegie Mellon University
sebarton@andrew.cmu.edu

Fridtjof Melle
Carnegie Mellon University
fmelle@andrew.cmu.edu

Abstract

When looking for somewhere to eat, people often read online reviews of restaurants they have not visited themselves. One of the most common websites that aggregates user reviews is `http://www.yelp.com`*. Using machine learning techniques, we aimed make this process simpler by giving users specific restaurant recommendations, replacing their time spent looking through many possible places to eat, turning down many or most of them, with a small subset of restaurants the user is likely to enjoy based on the reviews of similar users.*

1. Introduction

1.1. The Problem

`http://www.yelp.com` is a popular destination for people looking to find out what other people think of restaurants, coffee shops, and other food establishments in their area. Often people make decisions on where to eat based on the reviews other yelp members give to restaurants they're considering. While this is more useful than having no information about the restaurant at all, someone looking for a recommendation is bound to the opinions of strangers, who perhaps have very different tastes in food.

1.2. The Solution

Our solution is to fix this problem by giving a person personalized recommendations. Based on the person's past reviews, we aim to find a subset of *Yelp* users with similar taste and guess the scores that the given person would give to restaurants he or she has not yet visited. This allows the person to try a new restaurant with a greater sense of security that he or she will enjoy the experience and not waste money. [1]

2. Algorithm Overivew

2.1. The Pipeline

Our pipeline takes in a person, and outputs a restaurant recommendation. In order to do this we first create a feature vector for the person, and find a list of similar users based on these features. We then see how these similar users rated restaurants the person has not been to, and therefore guess how the person would rate those restaurants. Finally, the restaurants with the highest expected ratings are returned to the user.

2.2. Similar Users and Restaurant Rating Prediction

Our solution creates a feature vector of each user and uses the Kth Nearest Neighbor algorithm to cluster the feature vectors and find K users closest to the person based on the feature vector. We then take a weighted average of the reviews from those users for every restaurant the person has not visited to guess how that person would rate those restaurants.

3. What We Have Done

3.1. Parsing the Data

The *Yelp* dataset is very large, and so we have parsed it down and removed a small subset to work with. We took the data for the top 50 users, and all the restaurants they have visited, as well as their reviews of them, and the personal data *Yelp* stores for those users. We converted all the data from JSON objects to easy to load and work with .csv files. The scripts which do this can easily be changed to load in data for a different set of users and save the same .csv files.

3.2. Making User Vectors and Finding Similar Users

Right now our user vectors are based on the user's *Yelp* profiles. Yelp user reviews can be rated by other users as ”funny”, ”useful”, and ”cool”. Users can also compliment other users in many ways. We vectorize all of a users ratings and complements, as well as append the user *Yelp* statistics (fans, average review, and number of reviews). This feature vector gives us an idea of how other users view this user in the *Yelp* community. This feature vector was a quick way to create a feature vector that we could use and test in our pipeline, in the future we plan to add food and restaurant-specific features. To find similar usres, we simply run KNN for a single user feature vector on the entire set of user feature vectors. Modifying the feature vector is one of the key steps we plan to work on in the next half of the project.

3.3. Guessing User Reviews

Right now, to guess a user review, we take two things into account: the ratings a user's similar users rated a restaurant and the average rating of the restaurant overall We weight each by a half to guess how a user would rate a restaurant.

3.4. Implementation

Our code is implemented in Python's PANDAS module. Coding in Python allows us to rapidly prototype and get things working in the short amount of time we have to do the project. PANDAS, being strictly typed and implemented in C, speeds of computation time considerably and allows us to run our code on a much larger dataset. The code is visible at `https://github.com/sbarton272/PatRec`.

References

[1] A. Alpher. Frobnication. *Journal of Foo*, 12(1):234–778, 2002.