Yelp User Rating Prediction

Yifei Feng
yife@stanford.edu

Zhengli Sun
zsun4@stanford.edu

I. Introduction

Yelp is a popular crowd-sourced local business reviews and social networking platform. For each individual business, users can submit a review on their products or services using a one to five star rating system while interacting with each other through following users and voting on reviews. Having accumulated an enormous amount of data on information, reviews, and ratings of businesses, Yelp has become an important reference for making consumer decision. However, going through a large quantity of raw data can be time consuming. In order to overcome information overload, we would like to create a more efficient option that gives users suggestions that are tailored to their personal preferences.

Recommender systems have become a key tool for providing users with personalized recommendations on items such as movies, music, books, news, and web pages.

We aim to create a recommender system for yelp users by predicting the users' future ratings of businesses. We will use the data set provided by Yelp Dataset Challenge.

We are building a few models using three different approaches, namely content-based filtering, collaborative filtering and hybrid approach.

Content-based filtering methods are based on a business's profile and the user's preference. We can predict the rating score from a user to a business by matching user's interests with description and attributes of the business. Standard machine learning techniques such as Logistic Regression can be applied to implement this approach.

Collaborative filtering methods are based on analyzing the information of users' reviews, ratings or preferences and predicting what users will like based on their similarity to other users. We will build a k-nearest neighbor model for this approach.

The hybrid approach is to combine collaborative filtering and content-based filtering. Recent research has demonstrated this approach can have better performance in some cases. We will explore a few ways of combination and implement them.

II. Dataset

Our dataset is the Yelp Dataset Challenge data (http://www.yelp.com/dataset_challenge). This dataset contains five tables for business, review, user, check-in and tips. We are using three (business, user and reviews) of them.

For review table, there are 1125458 reviews; each contains information such as business id, user id, star ratings etc. The business and user tables provide further information about each business/user such as their review counts, average reviews, etc. There reviews come from 252,898 users for 42,153 business. This yields a sparsity of 99.99%, which is relatively high (Netflix data have a sparsity of 98.82%[1]).To see the review count distribution of the users, we plot the histogram as shown in figure 1. We only used restaurant reviews by users with over 20 reviews, which boosted the sparsity to 99.63%.

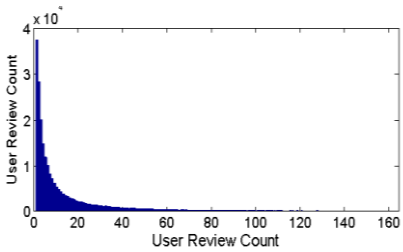


Figure 1 Histogram of user review count

III. Evaluation Metric

According to [3], the most widely used evaluation metrics for recommender system are root mean squared error (RMSE) and mean absolute error (MAE). We used both systems to check the performance of various methods implemented. RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{\sum\left(\hat{r}_{u,i} - r_{u,i}\right)^2}{n}}$$

, and MAE is defined as:

$$MAE = \sqrt{\frac{\sum\left|\hat{r}_{u,i} - r_{u,i}\right|}{n}}$$

where $\hat{r}_{u,i}$ is the predicted rating from user u on item i, $r_{u,i}$ is the actual rating and n is the size of test set. We used k-fold cross validation where k =10 to calculate the evaluation metrics.

IV. BASELINE

Our baseline model is similar to the model used in [2]. A baseline estimate rating is denoted by $b_{u,i}$ and accounts for the user and item effects:

$$b_{u,i} = \mu + b_u + b_i$$

where μ is the mean rating of all reviews in the system. The parameter b_u indicates the difference between the average rating of user u and μ. The parameter b_i indicates the difference between the average rating of business i and μ.

For each review in the dataset, we predicted the rating from the user to the business using the formula above and compared the prediction to the actual rating to calculate our baseline RMSE and MAE.

V. CONTENT-BASED FILTERING

In order to create user and business profiles, we used raw numerical data from selected features in the user and business tables, as well as derived feature based on restaurant categories. For raw data input, we used 3 features (user average review, review count and fan count) for users and 4 features (business star rating, review count, longitude and latitude) for businesses.

Content-based filtering predicts the rating score from a user to a business by matching user's interests with description and attributes of the business. In order to create more comprehensive profiles for users and businesses, we also derived a feature that reflect user's preference to restaurants of a certain category. In order to get this data, we wanted to create a binary table of what categories each restaurant belong to, as well as a table for what are the average review of each category for every user. However, there are 230 total available categories but each user only have >20 reviews. To solve this issue, we used k-means to cluster all the categories into five groups and created the above mentioned two tables based on the new categorization.

We created our review data matrix by replacing the business ID column with business features, replacing the user ID column with user features, and the derived user's preference of this category. Ordinal multinomial logistic regression and binary decision tree was applied to implement this approach.

A. Ordinal Multinomial Logistic Regression

The review ratings we are trying to predict can take on a value $y \in \{1, 2, 3, 4, 5\}$. Since these categories are ordered, we used a proportional-odds cumulative logit model[5] to fit the data. Assume the associated probabilities of each value y can take are $\{\pi_1, \pi_2, \pi_3, \pi_4, \pi_5\}$ and the accumulative probability of y being less or equal to j is,

$$P(y \le j) = \pi_1 + \cdots + \pi_j$$

The proportional odd is,

$$L_j = \ln\left(\frac{P(y \le j)}{P(y > j)}\right) = \ln\left(\frac{\pi_1 + \cdots + \pi_j}{\pi_{j+1} + \cdots + \pi_5}\right)$$

If we use covariates into the model and make the coefficient of each feature identical across the 4 logit equations, we have that

$$L_1 = \alpha_1 + \beta_1 X_1 + .. + \beta_7 X_7$$
$$L_2 = \alpha_2 + \beta_1 X_1 + .. + \beta_7 X_7$$
$$\vdots$$
$$L_4 = \alpha_4 + \beta_1 X_1 + .. + \beta_7 X_7$$

MATLAB function *mnrfit* was applied to the training set to calculate the intercept terms α and coefficients β. Then α and β were used to calculate the proportional odds of testing data.

B. Binary Decision Tree Regression

Because the reviews are ordered, we used classification and regression trees (CART) to split the dataset on each feature to maximize the standard deviation reduction. We swept a range of minimum numbers of observation per leaf node. As shown in Figure 2, a minimum number of around 10^3 yields the lowest error rate.

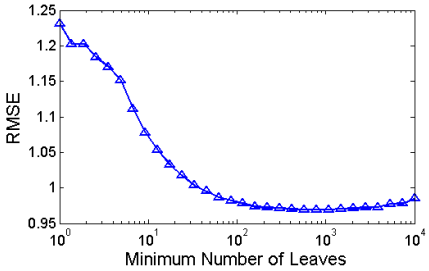


Figure 2. RMSE vs the minimum number of observation per leaf

VI. COLLABORATIVE FILTERING

A. Item-based K Nearest Neighbors (KNN)

In the collaborative filtering method, in order to predict the rating of user u on item i, we look at the top k items that are similar to item i, and produce a prediction by calculating the weighted average of ratings from user u on these items.

It's crucial to calculate the similarity between each two items. We use the cosine similarity of item X and Y as the similarity measure:

$$siml(x, y) = \text{cost}(\vec{x}, \vec{y}) = \frac{\sum_{u \in U_{xy}} r_{u,x} r_{u,y}}{\sqrt{\sum_{u \in U_x} r_{u,x}^2} \sqrt{\sum_{u \in U_y} r_{u,y}^2}}$$

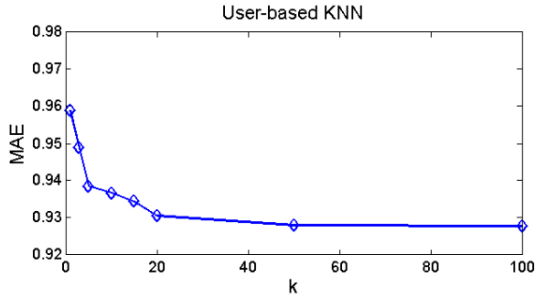
where U_{xy} is the set of users who rated both x and y, and $r_{u,x}$ and $r_{u,y}$ are the ratings from user u on item x and y.

We predict the value of ratings user u gives to item i as a weighted average of similar items user u has rated:

$$r_{u,i} = \frac{\sum_{n \in N_u^k(i)} siml(i,n) * r_{u,n}}{\sum_{n \in N_u^k(i)} siml(i,n)}$$

where $N_u^k(i)$ is the top k neighbors of item i that user u has rated.

We used cross-validation to train the model with different k.



One observation from the result is that as the value of k increases, we can get a lower MAE, i.e. a better prediction. However, the improvement plateaus at $k \approx 50$.

Another observation is that comparing this result to our baseline, this model's performance is not good. The main reason is that Yelp review data is highly sparse due to the relatively high costs of writing review for Yelp users. Specifically, even we are aware of the k items most similar to the item i, the probability of user u having reviewed any of these items is relative small. So we are still not able to make good predictions.

B. *User-based K Nearest Neighbors (KNN)*

Another approach of Collaborative Filtering is User-based K Nearest Neighbors. Similar to Item Based KNN, this model first found the top K users who are most similar to the user u, and based on their ratings on item i to predict the rating from user u on item i.

C. *Matrix Factorization*

Matrix Factorization is a factor method that performs well on sparse datasets. It was proved to be efficient in several top submissions to the Netflix Prize contest.

The idea of this method is to map both users and items to a joint latent factor space. Each user u is associated with a vector p_u, and each item i is associated with a vector q_i. The rating can be predicted as

$$\hat{r}_{u,i} = q_i^T * p_u$$

We ran updates on each user and feature vectors by minimizing:

$$\min_{q,p} \sum_{(u,i) \in K} (r_{u,i} - q_i^T * p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$

where $r_{u,i}$ is the known rate from user u on item i, and λ is the regularization parameter.

We used a stochastic gradient descent optimization[4] by Simon Funk to minimize the equation above.

We also used the biased stochastic gradient descent algorithms [6] by Yehuda Koren to take bias of users and items into accounts:

$$\min_{q,p} \sum_{(u,i) \in K} \left(r_{u,i} - \mu - b_u - b_i - q_i^T * p_u\right)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2 + b_u^2 + b_i^2)$$

Where μ is the global average of ratings, and b_u and b_i are the bias of user u and item i respectfully.

VII. HYBRID

We implemented a hybrid model to combine both content-based filter and collaborative filter. The purpose is to take the advantages of users' and restaurants' profiles while also maintaining the advantages of a neighborhood model.

Specifically, we trained a KNN model first and obtained predictions for each users based on their neighbors. The prediction was then added to MNOLR as a feature to recalculate the proportional odds model. The prediction was also added to BDTR for comparison.

VIII. RESULTS

Table # below shows the RMSE and MAE of each applied algorithms.

Table 1 Results of various algorithms

Method	RMSE	MAE
Baseline	1.0289	0.8772
OMNLR	1.0241	0.8485
BDTR	0.9691	0.8726
KNN (Item-based)	1.1222	0.9223
KNN (User-based)	1.1066	0.8989
Matrix Factorization (Stochastic gradient descent)	1.0269	0.8567
Matrix Factorization (Stochastic gradient descent with bias)	0.9829	0.8674
OMNLR + KNN	0.8876	0.7479
BDTR + KNN	0.8736	0.7256

IX. DISCUSSION

For content-based filtering, with the limited information given in the user table, it was difficult to create user profiles that reflect user preference of business types. Therefore, we didn't see drastic improvements from the baseline. For collaborative filtering, KNN algorithm is not efficient because the sparseness of the dataset makes it difficult to find representative neighbors for users. While Matrix factorization solution has a slightly better performance on the sparse data. Our hybrid models produce a significant improvement since both users' and neighbors' preferences are taken into account.

X. Future work

We would like to analyze the review text and use this information to create more comprehensive user and business profiles. We would also want to explore more sophisticated hybrid techniques, such as cascade or switching methods.

References

1. Mendeley Data vs. Netflix Data. (2010, November 2)., from https://synthese.wordpress.com/2010/11/02/mendeley-data-vs-netflix-data/
2. Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '08). ACM, New York, NY, USA, 426-434. DOI=10.1145/1401890.1401944 http://doi.acm.org/10.1145/1401890.1401944
3. Shani, Guy, and Asela Gunawardana. "Evaluating recommendation systems." Recommender systems handbook, Springer US, 2011. 257-297.
4. Lops, Pasquale, Marco De Gemmis, and Giovanni Semeraro. "Content-based recommender systems: State of the art and trends." *Recommender systems handbook*. Springer US, 2011. 73-105.
5. STAT 504 - Analysis of Discrete Data. (n.d.). from https://onlinecourses.science.psu.edu/stat504/node/176
6. Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." *Computer* 42.8 (2009): 30-37.