Google Local Business Ratings Prediction

Chen, Cai
cac005@ucsd.edu

Zhu, Hao
h8zhu@ucsd.edu

***Abstract*— When we use Google Maps to find some restaurants or shopping malls or groceries, we always check for the ratings and comments of this site first to choose a favorite place. In this report, we want to find out the relationship between comments and ratings and predict users' ratings using their reviews. We use two different models and three different feature vectors and decreasing the mean squared error about 32% compare to the baseline.**

I. Introduction

From websites such as Amazon to online multimedia sites such as Netflix and YouTube, recommendation algorithms have become critical to the design and implementation of a successful online platform. Modeling and predicting the interactions between users and items, as well as the relationships amongst the items themselves are the main tasks of recommender systems. Local Reviews data which contains people's comments are another important feature to get a better prediction result. We are going to use the Google Local businesses data in the California Area which is a wide range of publicly available datasets varying significantly in domains, sizes, data sparsity, and variability/complexity. We want to dive into the relationship between the ratings and the reviews to find out what factors affect ratings and then we can make a model predict ratings based on reviews.

II. Dataset

This dataset introduced by [1] contains reviews about businesses from Google Local (Google Maps). The Data includes geographic information for each business as well as reviews and ratings. Since the data is sparse around the world, we will only pick those places within California and their reviews. So the data is preprocessed through a preprocessor which eliminates reviews outside California by their longitude and latitude. Besides, we drop the reviews whose fields are NaN or the GPS data does not have the right range of latitude and longitude. Moreover, a few reviews written in Spanish are translated into English for better analysis. The remaining reviews are around 220000, with about 44000 users and 99000 places. The amount should be sufficient to train and test our models.

III. Data Analysis

We analyze the rating in terms of categories, locations and review time. First of all, Fig 1 and Fig 2 show the overall distribution of rating and top 20 categories of the dataset. As we can see, half of those reviews are 5.0 rating and the distribution is in increasing tendency. The interesting thing is that the increment is fast. This distribution can be explained by purchasing bias [2]. In fact, people rate a business because they are interested in such places and they consume, which is purchasing bias. Customers who consume in a place are therefore more likely to write positive reviews. And if they really satisfy the place, they are eager to give high ratings.

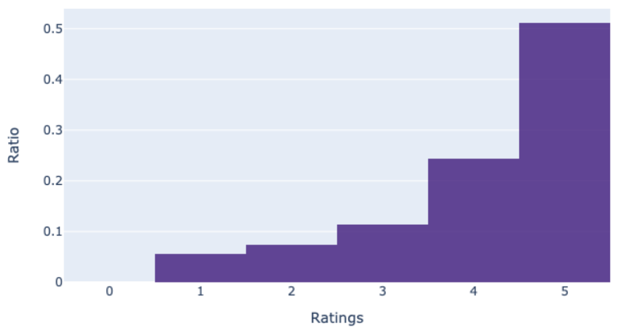

Fig. 1. The ratio of reviews ratings in California

The pie chart in Fig 2 shows the top 20 business categories rated in California. Most of the business is American Restaurant. But each category proportion does not vary too much. Also, we can see that the restaurant appears frequently in top 20 since there is more restaurant than other businesses. Therefore, more people will rate this kind of business.

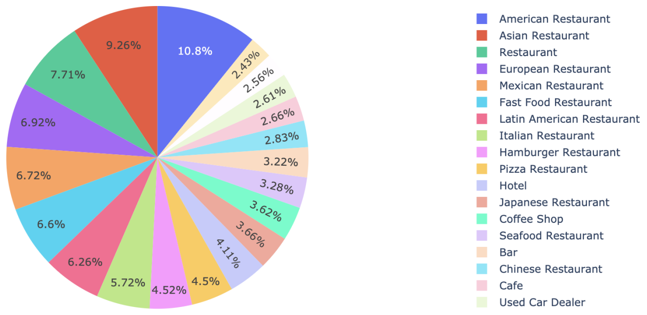

Fig. 2. Top 20 business categories rated in California

To find more information between rating and category, the distribution of ratings among the top 20 business category are revealed in Fig 3. The order of the y-axis from the bottom up is the top 20 order. The chart shows that the restaurant's distribution is more alike, while other kinds of business are more different.

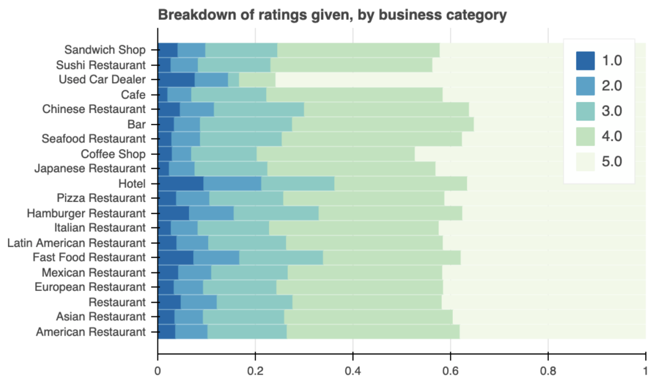

Fig. 3. Distribution of ratings by business category

Concerning location, we point out all the places on the map with different colors represent the ratings. As we zoom the map, we can clearly find that low rating usually appears in the denser area. In addition, if the area has less business, the ratings tend to be high. It makes sense because people will make a comparison if they have more choices in a certain area.

IV. Predictive Task

For the predicting rating task, we have given a review data without rating and want to predict the rating of this review.

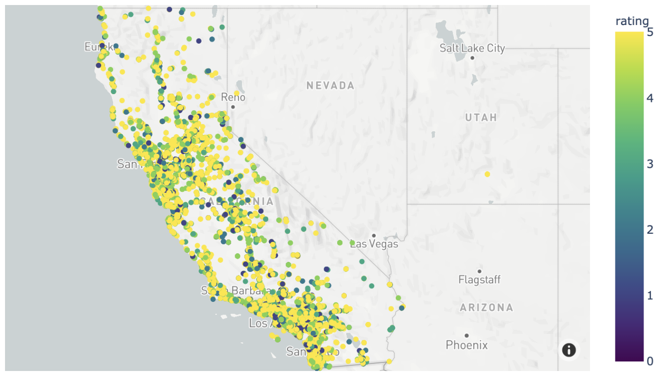

Fig. 4. Distribution of ratings by business location

A. Baseline

There is a straightforward way to do it which is to predict the rating as the user's average rating. This method is simple and can be used as our baseline.

$$rating_{u_i,b_j} = \frac{\sum_j rating_{u_i,j}}{number\ of\ u_i's\ ratings} \tag{1}$$

B. Gradient Boosting

1) Feature Extraction: Based on previous research and analysis, we find out that useful information for our predicting rating task can be the time of the review, the review text, and the business locations data. All those information shows in some way that they have some impact on the ratings the users gave. This can be our first kind of feature vector.

Another piece of information that is quite useful is the review text. In our approach, we use the term frequency-inverse document frequency information(tf-idf) feature [3]. The tf-idf is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. We think that tf-idf is a reasonable model because it shows the relation between ratings and different words. To build tf-idf model, bigrams will be better than unigrams for avoiding read 'don't like' as 'like'. We can calculate the tf-idf value for every word and choose the most popular 1000 words as our another feature vector. The formulas [4] are shown as below:

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{2}$$

$$idf_i = lg\frac{D}{\{j : t_i \in d_j\}} \tag{3}$$

$$tfidf_{i,j} = tf_{i,j} * idf_i \tag{4}$$

The $tf_{i,j}$ means the appearance of a word in a document divide by the total words in this document. The idf_i is the total number of documents that include this word divide by the total number of documents. Then, our tf-idf is the multiplication of these two numbers.

2) Algorithm: Based on those features, the methodology came into our mind is to use a Boosting model to do the regression [4]. As boosting model ensures that our predictions will not over-fit in the testing set and also provide a great result. In our approach, we choose to use a gradient boosting model. The gradient boosting method is that we assume there is a function $y = F(x)$ for our data, where x is the features, y is the ratings. And we want to find the approximation function $\hat{F}(x)$ that can minimizes the expected value of some specified loss function $L(y, F(x))$:

$$\hat{F} = \arg\min_F E_{x,y}[L(y, F(x))]. \tag{5}$$

The gradient boosting method combines a set of weak learner to get the final learner. So we can get the following formula:

$$\hat{F}(x) = \sum_{i=1}^{M} \gamma_i h_i(x) + \text{constant}. \tag{6}$$

To achieve the best approximation, the gradient boosting using decision trees as the base learners and it has the following iteration function:

$$F_m(x) = F_{m-1}(x) + \arg\min_{h_m \in \mathcal{H}} [G(x) + h_m(x_i)] \tag{7}$$

where $G(x)$ is:

$$\sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) \tag{8}$$

And at the begining of this iteration, we have a constant function $F_0(X)$, which is:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma) \tag{9}$$

C. Latent Factor Model

The latent factor model [5] is quite a suitable model for this rating problem [6]. In the previous data analysis, we found some attributes may be helpful for the predictive task but the rating distribution shows that higher ratings are the majority and that's the situation in every category, every different area. So this also means that the general features may not enough for this task. Here we can use the latent factor model, which will use the latent factors hiding behind some observable factors. We can take our task as an example to understand this model more precisely. In this task, we define two latent factors:

β_u user bias which means how much does this user tends to rate things above the mean.

β_b business bias which means does this item tend to receive higher ratings than others.

We can now clearly see that key features in the latent factor model are not features we can generate directly from our data. But those factors do have a relationship with the ratings. In our latent factor model, we use the following function to predict rating:

$$f(u, b) = \alpha + \beta_u + \beta_b \tag{10}$$

where α means the average rating and β_b and β_u are the bias. Then our optimization becomes:

$$\arg\min_{\alpha,\beta} \sum_{u,b} (\alpha + \beta_u + \beta_b - R_{u,b})^2 + \lambda[\sum_u \beta_u^2 + \sum_b \beta_i^2] \tag{11}$$

V. Results and Conclusion

A. Evaluation

In our project, we use the mean square error as the evaluation metrics.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2. \tag{12}$$

B. Experiments

In our predictive task, we use the baseline we have mentioned. It provides the baseline MSE of about 1.49.

For the Boosting model, we choose to use the XGBoost library which provides an effective and accurate gradient boosting model. We use the

XGBoost regressor for our regression problem. In this experiment, we use a simple feature vector and a feature vector with $tf\text{-}idf$ included to get two different models.

For the Latent Factor Model, we use userID and businessID as the attributes we want to find the latent factors and iterate the following update functions until convergence:

$$\alpha = \frac{\sum_{u,b \in train}(R_{u,b} - (\beta_u + \beta_b))}{N_{train}} \tag{13}$$

$$\beta_u = \frac{\sum_{b \in B_u}(R_{u,b} - (\alpha + \beta_b))}{\lambda + |I_u|} \tag{14}$$

$$\beta_b = \frac{\sum_{u \in U_b}(R_{u,b} - (\alpha + \beta_u))}{\lambda + |U_b|} \tag{15}$$

The α, β_u and β_b are the features we mentioned before. For the users or business that doesn't appear in the training set, we will not consider the bias of that one.

For both boosting and latent factor models, we use cross validation for selecting the best hyper-parameters and also ensures that models will not over-fitting.

After validation, we use the best parameters we found as the final models and get results on the testing set.

C. Results

The results below is the average MSE of 10-fold cross validation:

Model	**Validation MSE**	**Testing MSE**
Baseline	1.4886	1.4905
Simple Features Boosting	1.3186	1.3170
TF-IDF Feature Boosting	1.0151	1.0056
Latent Factor Model	1.1757	1.1887

TABLE I

MEAN SQUARED ERROR OF DIFFERENT MODELS

D. Conclusion

We can see that from the results table, the latent factor model and TF-IDF boosting model are much better than the simple features boosting model and they improve the baseline result quite a lot. So it does means that our method works for this rating prediction task. On the other hand, we notice that using different feature vectors in Boosting models make quite different results. This shows that the review text information impact ratings more than the categories, locations or time attributes.

In conclusion, our predictive task is succeeded because after carefully analyzing and preprocessing the raw dataset, the selected data are suitable for both this latent factor model and the tf-idf featured linear model. The latent factor model is appropriate for mining latent features behind information like IDs and tf-idf is quite suitable for text analysis. That means, thoroughly analyzing the original dataset leads us choosing suitable methods and features. Appropriate methods and features lead us to a successful model.

REFERENCES

[1] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2015, pp. 43–52.

[2] M. Martin *et al.*, "Predicting ratings of amazon reviews-techniques for imbalanced datasets," 2017.

[3] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Information Processing & Management*, vol. 39, no. 1, pp. 45–65, 2003.

[4] L. Breiman, "Arcing the edge," Technical Report 486, Statistics Department, University of California at …, Tech. Rep., 1997.

[5] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski, "A latent factor model for highly multi-relational data," in *Advances in Neural Information Processing Systems*, 2012, pp. 3167–3175.

[6] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, "Aspect-aware latent factor model: Rating prediction with ratings and reviews," in *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 2018, pp. 639–648.