Restaurants Review Star Prediction for Yelp Dataset

Mengqi Yu
UC San Diego
A53077101
mey004@eng.ucsd.edu

Meng Xue
UC San Diego
A53070417
m6xue@eng.ucsd.edu

Wenjia Ouyang
UC San Diego
A11069530
weouyang@eng.ucsd.edu

ABSTRACT

Yelp connects people to great local businesses. In this paper, we focus on the reviews for restaurants. We aim to predict the rating for a restaurant from previous information, such as the review text, the user's review histories, as well as the restaurant's statistic. We investigate the data set provided by Yelp Dataset Challenge round 5. In this project, we will predict the star(rating) of a review.Three machine learning algorithms are used, linear regression, random forest tree and latent factor model, combining with the sentiment analysis. After analyzed the performance of each models, the best model for predicting the ratings from reviews is the random forest tree algorithm. Also, we found sentiment features are very useful for rating prediction.

Keywords

Yelp rating prediction; data mining; sentiment analysis

1. INTRODUCTION

The rating stars of businesses influence people much more on making a choice among all available businesses. They will choose the one with higher review rating stars in order to ensure the quality of service they will get from the business. In a Yelp search, a star rating is arguably the first influence on a user's judgment. Located directly beneath business' names, the 5 star meter is a critical determinant of whether the user will click to find out more, or scroll on. In fact, economic research has shown that star ratings are so central to the Yelp experience that an extra half star allows restaurants to sell out 19% more frequently[1].

Currently, a Yelp star rating for a particular business is the mean of all star ratings given to that business. However, it is necessary to consider the implications of representing an entire business by a single star rating. What if one user cares about only food, but a particular restaurant's page has a 1-star rating with reviewers complaining about poor service that ruined their delicious meal?[3] The user may likely continue to search for other restaurants, when the 1-star restaurant may have been ideal.

In a data mining project, different feature selections will change the accuracy significantly in learning models. It's quite vital to find out the important features influencing the review rating star.

Although there are many data sets that could be used to study and learn models for businesses or similar ones for items, such as Google communities and Netflix dataset. The reason why we selected the Yelp data set is that it has more information among the users, reviews and businesses that could be investigated for feature selections and models building to help much more accurate predictions. In this project, we will predict the star(rating) of a review. Three machine learning algorithms are used, linear regression, random forest tree and latent factor model, combining with the sentiment analysis. We analyze the performance of each of these models to come up with the best model for predicting the ratings from reviews. We use the dataset provided by Yelp for training, validation, and testing the models.

Table 1: Yelp Dataset attributes

Name	Attributes
Business	Business Name, Id, Category, Location, etc.
user	Name, Review Count, Friends, Votes, etc.
Review	Date, Business, Stars, Text, etc.

2. DATASET CHARACTERISTICS

2.1 Properties

In our project, we used a deep dataset of Yelp Dataset Challenge round 5, which is available at yelp website. The Challenge Dataset has 1.6M reviews and 500K tips by 366K users for 61K businesses; 481K business attributes, e.g., hours, parking availability, ambience; Social network of 366K users for a total of 2.9M social edges aggregated check-ins over time for each of the 61K businesses. More specifically, it includes the data of 61184 businesses, 1569264 reviews and 366715 users, which contains following information shown in Table 1. All the data are in json format. For reviews, the data looks like:

```
{
  'type': 'review',
  'business_id': (encrypted business id),
  'user_id': (encrypted user id),
  'stars': (star rating, rounded to half-stars),
  'text': (review text),
  'date': (date, formatted like '2012-03-14'),
  'votes': (vote type): (count)
}
```

2.2 Businesses Statistic

There are multiple types of businesses, such as Doctors,

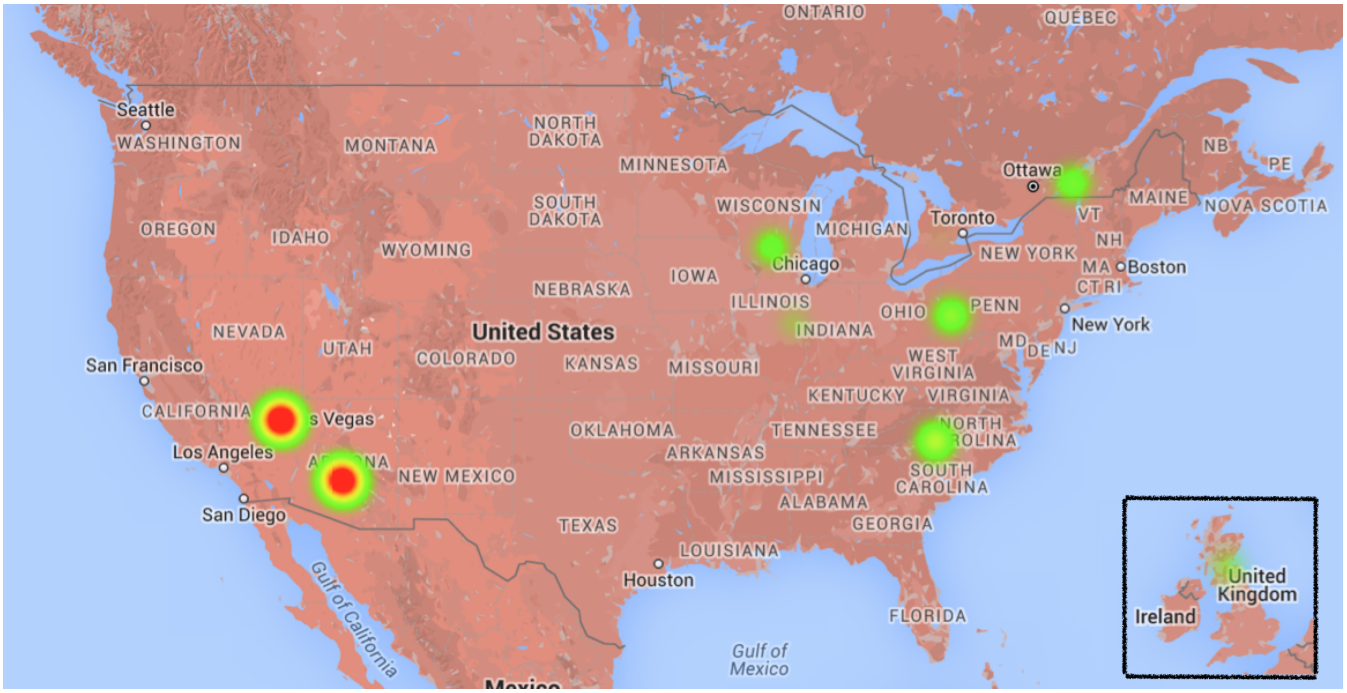

Figure 1: Restauant locations distributed

Health&Medical, Active Life, Cocktail Bars and Nightlife, but we have focused only on restaurant reviews as they account for major part of the dataset, which hold 35.78% of total businesses.

Figure 1 shows all restaurant's locations globally. It gives us an initial idea about where these restaurants are: the most restaurants are located in US. Then, Figure 2 tells us restaurants distribution by state. It surprised us that all restaurants from given dataset are located only in seven states, PA, NC, NV, WI, IL, AZ and SC, and Nevada has most number of restaurants.

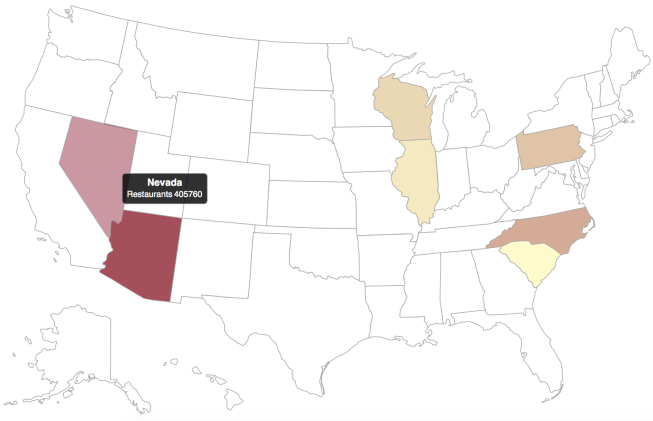

Figure 2: Restauant locations distributed

2.3 User Statistic

In the user dataset, we have 366715 users, and it is interested to find that 88% users are given less then 60 reviews. Figure 3 shows the distribution of how many reviews a user

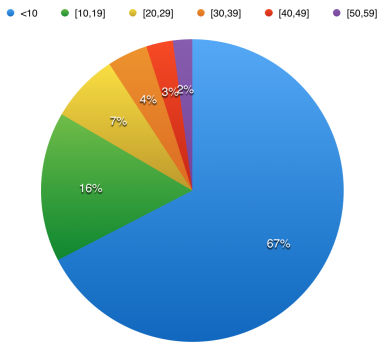

Figure 3: Distribution of how many reviews a user given

given. It clearly shows that the most user only given a few reviews(less than 10 reviews), and a few user provided huge number of reviews(more than 50 reviews), which reveals that different group of users has different habits and we could separate them into groups to get more appropriate features.

Moreover, by analysis the time of new users joined in and the time of a review wrote in Figure 4, we knew that the most users joined in Yelp around 2010, and they continue contributing increasing number of reviews year by year.

2.4 Review Statistic

Review data is the most interested dataset part and included more information. Therefore, we are planning to predict the star of a review will give. The restaurant review dataset contains 990627 reviews (63% of total reviews), which amounts to around 1GB of data.

Firstly, we try to figure out distribution of review rat-

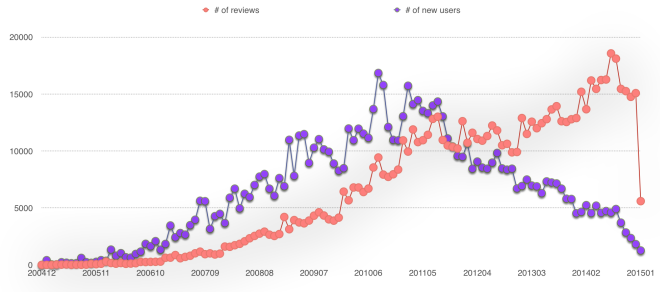

Figure 4: Number of new user joined in with number of reviews

ing. From Figure 5, we found the review distributions in our dataset were skewed to the 4 and 5 star categories heavily. They consist of around 70% of the distribution, whereas the 1, 2, and 3 star categories are each only around 10-15% at most. This is confirmed by a separate analysis by Max Woolf[8] on 1 and 5 star reviews which showed, excellent visualization aside, that Yelp reviews have started to appear more optimistically biased as time passes. This skewed distribution is reflected by our sample dataset, and uneven class distribution will become noteworthy further on in our predictive task.

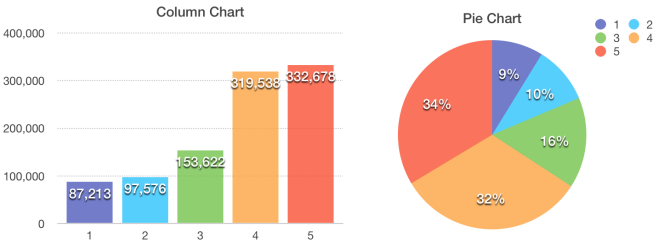

Figure 5: Rating Stars distribution

We also plotted relationship of how many reviews a restaurant received by US state, as shown in Figure 6. It follows the common sense that in general, most restaurants will not receive too many reviews(less than 50), and only some popular restaurants received more than 100 reviews.

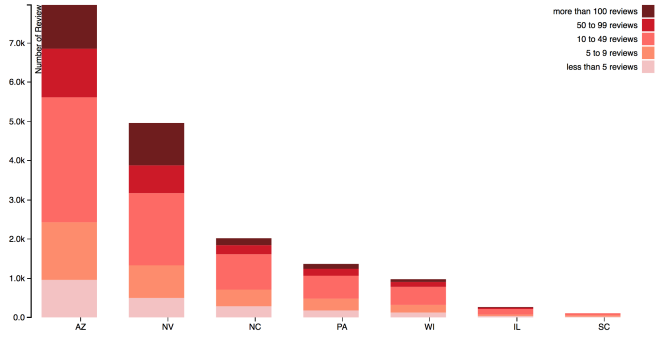

Figure 6: Number of review a restaurant received by US state

Furthermore, since text is a important part of reviews, we

Figure 7: WorldCloud of 1 star reviews

Figure 8: WorldCloud of 5 star reviews

generated the world cloud[5] of all 1 star review text and all 5 star review text, shown in Figure 7 and Figure 8 respectively. The world cloud return the most frequently single world with using stop worlds. These two figures should support to be much more different, however, we found they have some same worlds, such as "food", "place", "good" and "restaurant". This is caused by only a single world(unigram) frequency is counted, which can not tell the different between "good" and "not good". Bigram and trigram may be a good choice to represent content of reviews, as well as sentiment analysis of review text. We believe they could reveal more accurate features of the review text.

3. RATING PREDICTION PROBLEM SETUP

In this project, we planned to train a model to predict users' rating of a business. The motivation includes that if we can predict how a user is going to rate a business, then we can recommend the business that the user is more likely to rate higher than the others. The Netflix Prize dataset just has the user id, item id and the corresponding rating. The difference between Yelp dataset and Netflix dataset is that Yelp dataset has more information than the Netflix dataset. Therefore, we can not only try the latent factor model but also some feature-based models.

In the traditional rating prediction problem, we try to make our prediction $\hat{R}_{u,i}$ for user u and item i to be as close as $R_{u,i}$, the rating user gives. Usually, Mean Squared Error (MSE) is employed to compare the performance of the different model.

$$MSE = \frac{1}{|T|}\sum(R_{u,i} - \hat{R}_{u,i})^2 \quad (1)$$

4. MODEL SELECTION

According to the data analysis, the rating of new reviews should be predicted. There are several features we found useful in comparing with different models and help to choose the best model for prediction. The features are: the average rating stars for each user based on the review stars of the training dataset (uRate); the average rating stars for each business based on the review stars of the training dataset (bRate); the count of reviews this user had made (rCount); the lengh of the review text (tlen); the polarity of the sentiment of the review text (tpol); the subjectivity of the sentiment of the review text (tsub); the average rating stars for each user which can be get from dataset directly (uAvg); the average rating stars for each business which can be get from dataset directly (bAvg).

The data was separated into three parts: training dataset (80%), validation dataset (10%) and test dataset (10%).

4.1 Baseline

The baseline model is straightforward, using user's previous average given star to predict their future reviews' star. That is, based on the training data, a average star for each user is computed by computing the average of all their previous review stars. Then, we used these averages to predict test dataset. When a user in test dataset has not been seen before, we will use the global average star instead of it.

4.2 Linear Regression

Linear Regression is our first model to do the prediction. It calculates the weights of each feature and multiply them with features to do the calculation. The equation can be shown as:

$$y = X \cdot \theta + \epsilon \tag{2}$$

where y, θ, ϵ are 1-D array and X is 2-D array. y is the labels; θ is the weights, ϵ is the noise and X is the features for samples.

With the features mentioned above in this part, we applied the linear regression to the training dataset. The features are rCount, tlen, tpol, tsub. The label is set to (review_star - (uAvg+bAvg)/2.0) where $y[i]$ is review_star$[i]$ and $\epsilon[i]$ is (uAvg+bAvg)/2.0. With the features and labels, the linear regression is applied by using numpy.linalg.lstsq.

Then, the (features$\cdot\theta$+(uAvg+bAvg)/2.0) is applied, which is the equation mentioned above, to do the prediction for those users and businesses seen in training dataset. Otherwise, use (feature$\cdot\theta$+averageRate) for prediction where averageRate is the average rating stars for all training reviews.

Furthermore, in this model, we also add a correction to set the prediction value equal to 5.0 if it's calculated value is larger than 5.0, since the rating stars will never be larger than 5.0.

4.3 Random Forest Regression

A better but more complex way for the prediction is using Random Forest Regression. For each review in the training data, the features are chosen as uAvg, bAvg, rCount, tlen, tpol, tsub. The label is set to (review_star). With the features and labels, we applied random forest regression by using sklearn.ensemble.RandomForestRegressor. For those user or business not seen in training data, we use averageRate instead of uAvg and bAvg. Then apply the predict(feature) to do the prediction.

In this model, we compared the validation MSE with different parameters (n_estimators and max_depth). The evaluation for the model will be discussed in **Part 5**.

Furthermore, the same as what we did for the Linear Regression, we add a correction to set the prediction value equal to 5.0 if it's calculated value is larger than 5.0.

4.4 Latent Factor Model

This model has been discussed in the lectures. And it is very popular in recommender system. The basic idea behind this model is projecting the user's preferences and item's properties into low dimension space. The model can be formulated as:

$$\hat{R}_{u,i} = \alpha + \beta_u + \beta_i + \gamma_u \cdot \gamma_i \tag{3}$$

where α is the global average rating, β_u and β_i are bias term for user and item respectively and γ_u and γ_i are the interactive term for user and item respectively.
The optimization can be formulated as:

$$\begin{aligned} &\arg\min_{\alpha,\beta,\gamma} obj(\alpha,\beta,\gamma) \\ =&\arg\min_{\alpha,\beta,\gamma} \sum_{u,i}(\alpha+\beta_u+\beta_i+\gamma_u\cdot\gamma_i - R_{u,i})^2 \\ &+\lambda_1[\sum_u \beta_u^2 + \sum_i \beta_i^2] \\ &+\lambda_2[\sum_u \|\gamma_u\|_2^2 + \sum_i \|\gamma_i\|_2^2] \end{aligned} \tag{4}$$

This objective function is not convex, obviously. But we can find a approximate solution. The update procedure is described below:

1. Fix γ_i and solve $\arg\min_{\alpha,\beta,\gamma_u} obj(\alpha,\beta,\gamma)$
2. Fix γ_u and solve $\arg\min_{\alpha,\beta,\gamma_i} obj(\alpha,\beta,\gamma)$
3. Repeat 1) 2) until convergence.

The update rule for solving $\arg\min_{\alpha,\beta,\gamma_u} obj(\alpha,\beta,\gamma)$ is described below:

$$\alpha^{(t+1)} \leftarrow \frac{\sum_{i,j} R_{u,i} - \beta_u^{(t)} - \beta_i^{(t)} - \gamma_u^{(t)}\cdot\gamma_i}{N_{train}} \tag{5}$$

$$\beta_u^{(t+2)} \leftarrow \frac{\sum_{i\in I_u} R_{u,i} - \alpha^{(t+1)} - \beta_i^{(t+1)} - \gamma_u^{(t+1)}\cdot\gamma_i}{\lambda_1 + \|I_u\|} \tag{6}$$

$$\beta_i^{(t+3)} \leftarrow \frac{\sum_{i\in U_i} R_{u,i} - \alpha^{(t+2)} - \beta_u^{(t+2)} - \gamma_u^{(t+2)}\cdot\gamma_i}{\lambda_1 + \|U_i\|} \tag{7}$$

$$\begin{aligned} &\forall k \in [0...m], \gamma_{uk}^{(t+4)} \leftarrow \frac{1}{\lambda_2 + \sum_{i\in I_u}\gamma_{ik}^2}\cdot \\ &\sum_{i\in I_u}\gamma_{ik}(R_{u,i} - \alpha^{(t+3)} - \beta_u^{(t+3)} - \beta_i^{(t+3)} - \sum_{j\neq k}\gamma_{uj}^{(t+3)}\gamma_{ij}) \end{aligned} \tag{8}$$

where m is the number of dimensions of γ.

The update rule for solving $\arg\min_{\alpha,\beta,\gamma_i} obj(\alpha,\beta,\gamma)$ is described below:

$$\alpha^{(t+1)} \leftarrow \frac{\sum_{i,j} R_{u,i} - \beta_u^{(t)} - \beta_i^{(t)} - \gamma_u \cdot \gamma_i^{(t)}}{N_{train}} \quad (9)$$

$$\beta_u^{(t+2)} \leftarrow \frac{\sum_{i \in I_u} R_{u,i} - \alpha^{(t+1)} - \beta_i^{(t+1)} - \gamma_u \cdot \gamma_i^{(t+1)}}{\lambda_1 + \|I_u\|} \quad (10)$$

$$\beta_i^{(t+3)} \leftarrow \frac{\sum_{i \in U_i} R_{u,i} - \alpha^{(t+2)} - \beta_u^{(t+2)} - \gamma_u \cdot \gamma_i^{(t+2)}}{\lambda_1 + \|U_i\|} \quad (11)$$

$$\forall k \in [0...m], \gamma_{ik}^{(t+4)} \leftarrow \frac{1}{\lambda_2 + \sum_{u \in U_i} \gamma_{uk}^2} \cdot \sum_{u \in U_i} \gamma_{uk}(R_{u,i} - \alpha^{(t+3)} - \beta_u^{(t+3)} - \beta_i^{(t+3)} - \sum_{j \neq k} \gamma_{uj}\gamma_{ij}^{(t+3)}) \quad (12)$$

where m is the number of dimensions of γ.

5. EVALUATION

5.1 Baseline Model

We use the baseline model as we described in Section 4.1. The MSE of baseline model on validation dataset and test dataset are 1.36854701572 and 1.40981598434, respectively.

5.2 Linear Regression

Table 2: Feature Selection

No.	Features	MSE (validation)
(1)	(uAvg+bAvg)/2,tlen	1.12497365579
(2)	(1)+rCount	1.12314746791
(3)	(1)+tpol,tsub	0.83652044455
(4)	(uAvg+bAvg)/2,rCount	1.13349795590
(5)	(4)+tpol,tsub	0.84183016090
(6)	(uAvg+bAvg)/2,tpol,tsub	0.84183092194
(7)	(6)+rCount,tlen	0.83621779642
(8)	(7)+business review count	0.83637389167

Feature selection is a key step in Linear Regression model. We investigate different features and did calculated the MSE for the validation dataset. The different features and their results can be seen in table 2, and it shows these feature representations worked well or not.

In this table, the *'(uAvg+bAvg)/2'* is used in the label part which detailed explained in Section 4.2. According to the results, the features uAvg, bAvg, tlen, rCount, tpol, tsub improve the accuracy and business review count The minimum of MSE for the validation dataset occurs when the features are rCount,tlen,tpol,tsub with rating calculated with (uAvg+bAvg)/2. Then we choose this model for the test dataset. The MSE for the test dataset is 0.792026711082.

5.3 Random Forest Regression

In the Random Forest Regression, we changed the parameters n_estimators and max_depth to train the training dataset. With the model, we can get the MSE for the validation dataset. As the result in table 3, the larger the n_estimators and the the max_depth, the better the performance. In the experiments we did, the minimum MSE for the validation dataset is 0.694501549823 with n_estimators as 200 and max_depth as 12. Then we choose this model for the test dataset. The MSE for the test dataset is 0.639872918805.

Table 3: Parameter Selection

n_estimators	max_depth	MSE (validation)
100	4	0.840209944925
100	8	0.721826119193
200	8	0.721655426556
300	8	0.721448861872
200	12	0.694501549823

5.4 Latent Factor Model

The latent factor is effective and widely used in prediction task. In this model, we have three parameters to tune. λ_1 and λ_2 are the regularization parameters in equation 3. m is the dimension of the latent factor.

First, we fix the dimension of the latent factor to 8, and then tune λ_1 and λ_2. Our observation is the result is more sensitive to the value of λ_2. The performance with different λ_1 and λ_2 is shown in table 4.

Table 4: Regularization Parameter

$\lambda_2 \backslash \lambda_1$	1e-3	1e-5	1e-7	1e-9	1e-11
1e-3	1.27970	1.27199	1.27199	1.27181	1.27182
1e-4	1.27970	1.26571	1.26899	1.26910	1.26905
1e-5	1.27971	1.26773	1.31254	1.31479	1.31446
1e-6	1.27970	1.27162	1.32447	1.33089	1.33198
1e-7	1.27970	1.27184	1.32157	1.32914	1.32797

We fix λ_1 and λ_2 to the pair of λ_1 and λ_2 that perform best, and then we tune parameter m. The results with different m are shown in table 5.

Table 5: Dimension Selection

m	MSE (validation)
2	1.26571358485
4	1.26569328019
8	1.26574972949
16	1.26561341969
32	1.26560926510
64	1.26560277263

The result shows that the higher the dimension is, the better the performance will be. But the performance won't change much. Finally, we choose the parameters that have the best performance, i.e. $\lambda_1 = 1e-5$, $\lambda_2 = 1e-4$ and $m = 64$. We test the model on the test data and the MSE is 1.26561688673.

5.5 Comparison Between Models

In this project, we evaluated the Baseline, Linear Regression, Random Forest Regression and Latent Factor Model.

The Linear Regression costs the least and needs features to do the prediction. But it may have the overfit on training data.

The training cost of Random Forest Regression Model mainly depends on two parameters 1) the depth of trees 2) the number of estimators. It accepts large set of features

and can also optimize the performance by randomly selecting features from the set.

The Latent Factor Model costs a lot especially when the dimension of latent factor is very high, since it needs to iterate on multiple parameters until the objective function converges. But it doesn't need any features to do the prediction. One disadvantage of latent factor model is that when the dataset is not large enough for estimate all the parameters, the performance will become worse.

For each model, after comparing between different parameters or features, we choose the one with minimum MSE on the validation dataset to do the prediction on the test dataset to prevent the overfit on training dataset. The MSEs for the test dataset is shown in table 6.

Table 6: Comparison between MSEs for test data

Model	MSE (test data)
Baseline	1.40981598434
Linear Regression	0.79202671108
Random Forest Regression	0.63987291881
Latent Factor Model	1.26561688673

According to the statistic, the best model in this project based on the MSE comparison for the test data is the Random Forest Regression Model. The reason why it has the best performance may include that in this Yelp dataset, there are a lot of useful features could be used, as well as random forest tree could prevent overfitting. Besides, the sentiment parameters and the average rating stars of user and business may influence a lot on the review star. Since the Latent Factor Model doesn't use features, it may be the reason why Random Forest Regression has a better performance.

6. RELATED WORK

6.1 SVD++

Koren[4] proposed an algorithm that called SVD++ during the competition of Netflix Prize competition and they won the Netflix Prize by outperform Netflix's own recommending algorithm by 10%. The model can be formulated as:

$$\hat{R}_{u,i} = \mu + b_u + b_i + q_i^T \left(p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j \right) \quad (13)$$

where $\hat{R}_{u,i}$ is the estimated user's rating for item i, b_u and b_i are the bias term for a user and an item respectively, q_i and p_u are the latent factor for an item and a user respectively, $N(u)$ is the set of implicit information(the set of items user u rated), y_i is the impact factor associated with the the j-th item that user has rated.

Koren exploited both explicit and implicit feedback from user. The explicit feed back is the same as what we uses in the latent factor model, using users' rating as explicit feedback. Implicit feedback means whether users rate a restaurant and how many time they have rated. In other words, users can tell us their preference from whether they rate a restaurant and how times they have rated.

The cost function that we want to minimize is

$$\begin{aligned} \min \sum_{u,i} (R_{u,i} - \hat{R}_{u,i})^2 + \lambda_1 (\|p_u\|^2 + \|q_i\|^2) \\ + \lambda_2 (b_u^2 + b_i^2) + \lambda_3 \sum_{j \in N(u)} \|y_j\|^2 \end{aligned} \quad (14)$$

We can exploit stochastic gradient descent (SGD) to train the model. The update rule is list below:

$$b_i \leftarrow b_i + \gamma(e_{u,i} - \lambda_2 b_i) \quad (15)$$

$$b_u \leftarrow b_u + \gamma(e_{u,i} - \lambda_2 b_u) \quad (16)$$

$$p_u \leftarrow p_u + \gamma(q_i e_{u,i} - \lambda_1 p_u) \quad (17)$$

$$q_i \leftarrow q_i + \gamma(e_{u,i}(p_u + |N(u)|^{-frac12} \sum_{j \in N(u)} y_i) - \lambda_1 p_u) \quad (18)$$

$$\forall j \in |N(u)|, y_j \leftarrow y_j + \gamma(e_{u,i} q_i |N(u)|^{-\frac{1}{2}} - \lambda_3 y_j) \quad (19)$$

6.2 Sentiment Analysis

Current sentiment analysis methods can be grouped into four main categories: keyword spotting, lexical affinity, statistical methods and concept-level techniques[2]. Keyword spotting classifies text by affect categories based on the presence of unambiguous affect words such as happy, sad, afraid, and bored.[6] Lexical affinity not only detects obvious affect words, it also assigns arbitrary words a probable "affinity" to particular emotions.[7] Statistical methods leverage on elements from machine learning. Concept-level approaches leverage on elements from knowledge representation and, hence, are also able to detect semantics that are expressed in a subtle manner.

The python library TextBlob is used in this project. It's a library help processing textual data and provides the API for natural language processing. For the sentiment analysis tasks, TextBlob returns a named tuple with float variables polarity and subjectivity.

The polarity score is a float number in range of [-1.0, 1.0]. The more positive the value is, the more positive the input text is. Vice versa the more negative input text leads to the more negative value of polarity score.

For example, the negative text may include the words like: *'hate'*, *'disappointed'*, *'dislike'*, *'annoyed'*, *'awful'*, etc. The positive text may include the words like: *'excellent'*, *'great'*, *'best'*, etc.

The subjectivity score is also a float number but in range of [0.0, 1.0]. The 0.0 indicates the most objective score and 1.0 indicates the most subjective score.

For example, the subjective text may have a high frequency of the word like: *'I'*, *'we'*, *'our'*, etc. The objective text may have a really low frequency of those words.

7. CONCLUSION

Yelp dataset have much more information than some similar rating-based dataset, such as Netflix dataset. It allows us to explore the correlation between each attributes and the rating and employ the relations to predict more accurately. In this paper, we have trained several models including linear regression, random forest and latent factor model. We have also exploited some text mining techniques such as sentiment analysis to build our features. We compare the

parameters setting for each model to tune the performance and to control the model complexity. Finally, we apply the models on the test dataset and compare their performance. The result shows that random forest model outperforms the others, since that it uses some reasonable features extracted from the rich information of the dataset.

8. FUTURE WORK

Due to the limited time, we have no chance to tried some models such as Support Vector Machine, k-nearest neighbors and Stochastic Gradient Descent. Also, in the future, we will try to develop more models about text mining, such as bi-gram, tri-gram and TF-IDF, and we may predict the review star by using the review text only.

Moreover, Yelp may release more dateset includes more businesses out of US. Then, we could analysis users' habits in different countries. At the same time, we could extent our review star prediction to all businesses, not only for restaurants.

9. REFERENCES

[1] M. Anderson and J. Magruder. Learning from the crowd. 2011.

[2] E. Cambria, B. Schuller, Y. Xia, and C. Havasi. New avenues in opinion mining and sentiment analysis. *IEEE Intelligent Systems*, (2):15–21, 2013.

[3] M. Fan and M. Khademi. Predicting a business star in yelp from its reviews text alone. *arXiv:1401.0864*, 2014.

[4] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[5] A. Mueller. A wordcloud in python. *Andy's Computer Vision and Machine Learning Blog*, June 2012.

[6] A. Ortony, G. L. Clore, and A. Collins. *The cognitive structure of emotions*. Cambridge university press, 1990.

[7] R. A. Stevenson, J. A. Mikels, and T. W. James. Characterization of the affective norms for english words by discrete emotional categories. *Behavior Research Methods*, 39(4):1020–1024, 2007.

[8] M. Woolf. The statistical difference between 1-star and 5-star reviews on yelp. *http://minimaxir.com/2014/09/one-star-five-stars/*, September 2014.