Predict rating using Yelp review dataset

Jinhang Pan, Jialin Lou, Ryan Howard Keng

Abstract

Keywords: Sentiment analysis, bag-of-words, support vector machine, latent-factor model

1 DATASET

In this case, we used the dataset of Yelp reviews [1] provided by Kaggle. This dataset provides us with reviews from Yelp. Yelp, a local-online-reviews type of site, has developed rapidly in the recent years. The site has pages devoted to individual locations, such as restaurants or schools, where Yelp users can submit a review of their products or service using a one to five star rating system. As for a user looking for a new restaurant, he or she will pay much attention to the rating and reviews from other users and then make decision whether to try or not. As a business, the rating reveals the potential trend in the future. It becomes more and more popular to analyze the Yelp's dataset. Yelp itself devotes itself to exploring these data as well.

1.1 Data Exploring

There are 12,990 businesses, 47,823 users, and 289,907 reviews in the Yelp's dataset. This data set spanning from 2013-03-19 to 2013-06-12. We are going to explore this dataset and predict what rating the user will make for the business given a user and a business. We split the data set into three parts which are the training set (60%), the validation set (20%), and the test set (20%), respectively. The following analyses are all based on the training set. The data formats are described in Table1.

Table 1: Data Format

name	**description**
Business	'name','business_id', 'categories', 'city', 'full_address', 'Peoria','latitude', 'longitude', 'neighborhoods', 'open', 'review_count', 'stars', 'state', 'type'
User	'review_count', 'type', 'average_stars', 'name','user', 'user_id', 'votes'
Checkin	'business_id', 'checkin_info', 'type'
Review	'business_id', 'date', 'stars', 'text','review_id'

There are several sub-categories in this dataset which are associated with reviews and we decided to choose three which are strongly connected to ratings. The dataset we are choosing is as follows: fourteen fields in business category, seven in user and five in review. We will explore how will they influence with each other because with this information, we are able to select fewer but more useful features. In addition, we found out that the information in the User and Business is very helpful since it shows us the result of the whole review. For example, the 'average_stars' field in User is the average rating of the user for ALL the reviews he or she has rated. Same as the "review count" in Business.

1.2 Dataset analysis

We will first explore whether the total number of reviews a business has, and a user has reviewed have correlations to its rating or not. From Figure 1 and 3 we know that most of the business and user in our dataset only make no more than 300 reviews the size of which is too small when compared to our maximum of the review count in our data set. In case a tiny change in the counts will do matter, we make our count categories down to 30 per category as shown in Figure 2 and 4, , then we noticed that most of businesses/users only have no more than 30 reviews. Since most of the businesses and users don't have a large number of reviews, this might not influence a lot in our ratings.

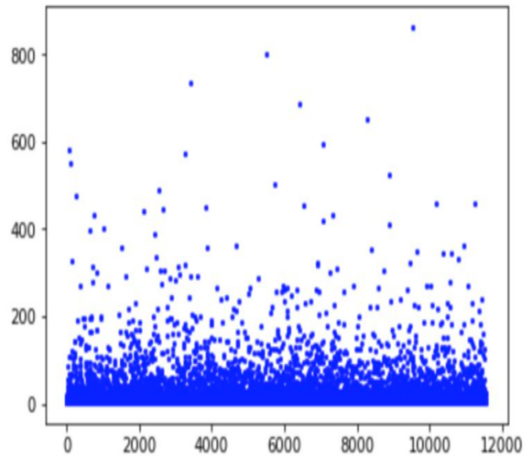

Figure 1: Review count for all businesses

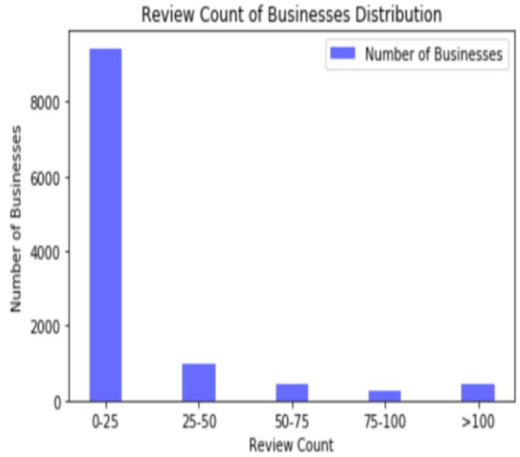

Figure 2: Review count for all businesses

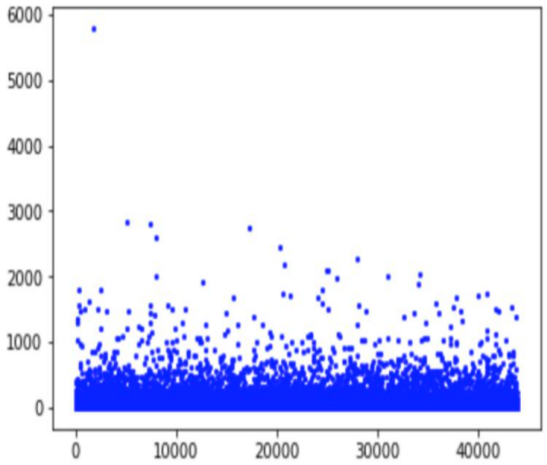

Figure 3: Review count from all users

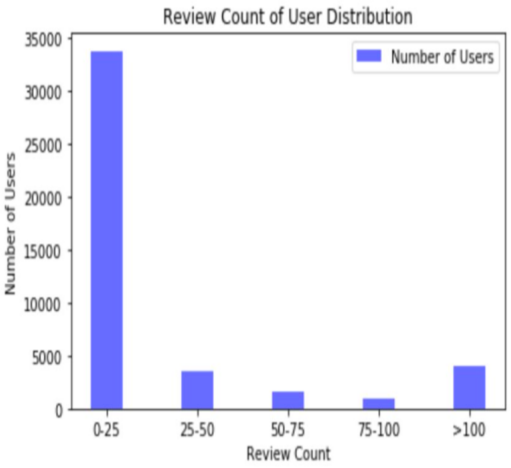

Figure 4: Review count from all users

1.2.1 Common Ratings

From 5 and 6, we also noticed that the most popular ratings business received are from three to four stars. This implies that when we do a prediction with a value from three to four, it is very possible that we get a decent baseline. Therefore, we make the average of rating for the businesses as one of our feature.

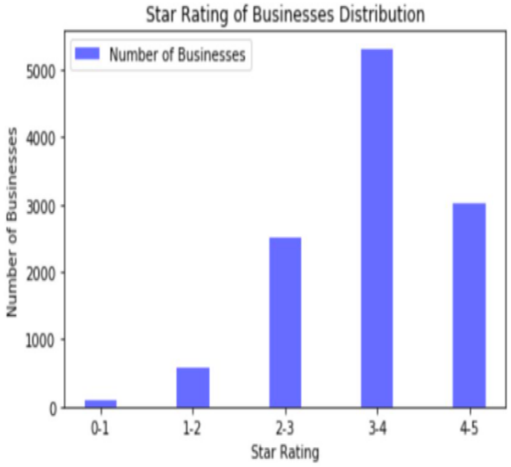

Figure 5: Rating Distribution from Business

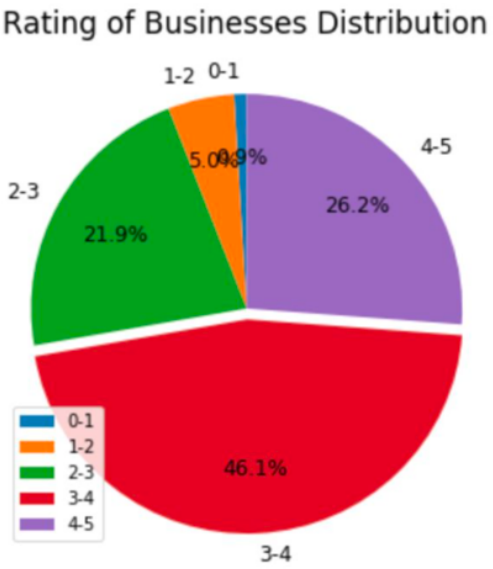

Figure 6: Rating Distribution from Business

In addition, we noticed that the most popular rating rated from users are from three to four and four to five. This also implies that when we do a prediction with a value from three to five, it is very possible that we get a decent baseline. Therefore, we make the average of rating from the users another feature.

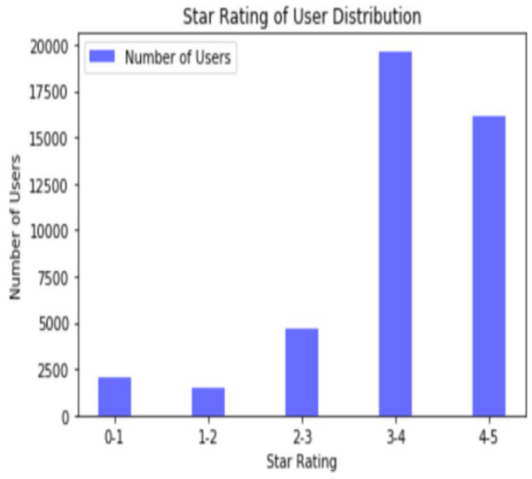

Figure 7: Rating Distribution from User

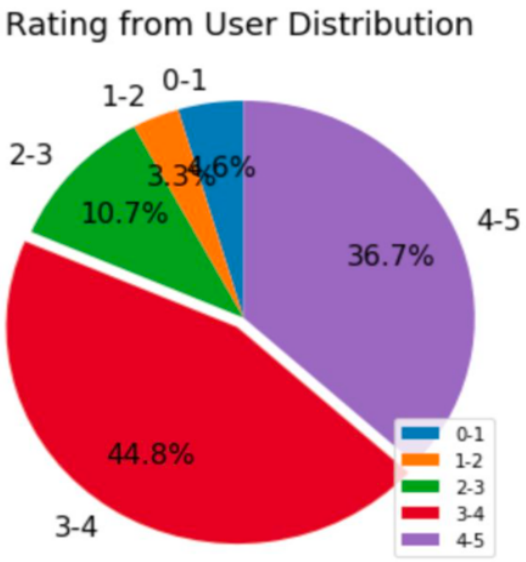

Figure 8: Rating Distribution from User

1.2.2 Three additional features of reviews: 'Cool', 'Useful', 'Fun'

There are also three counts for votes, counts for funny, cool, useful, respectively. We noticed that these three counts do not correspond to a specific range of rating. However, they give me some useful information when the number grows large. When the number of votes for any one type grows over twenty, the rating begins to decline, especially for 'funny' and 'useful'. However, after the number exceeds thirty, the rating begins to increase when the number of votes increases which implies that there is a correlation between these votes and the rating . 9. And We can also get an idea of the distribution of each type in Figure 8.

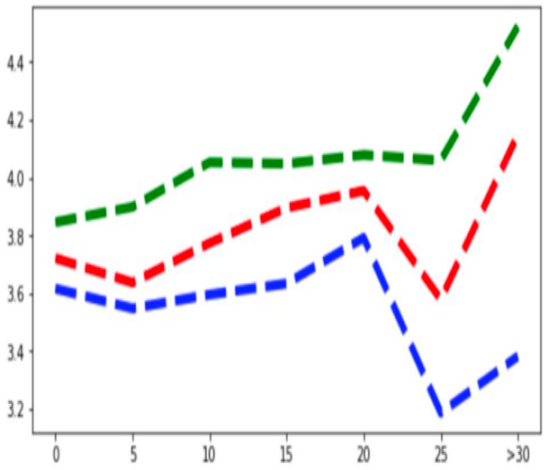

Figure 9: Number of Votes with Ratings, Green is 'cool', Red is 'useful', Blue is 'funny'

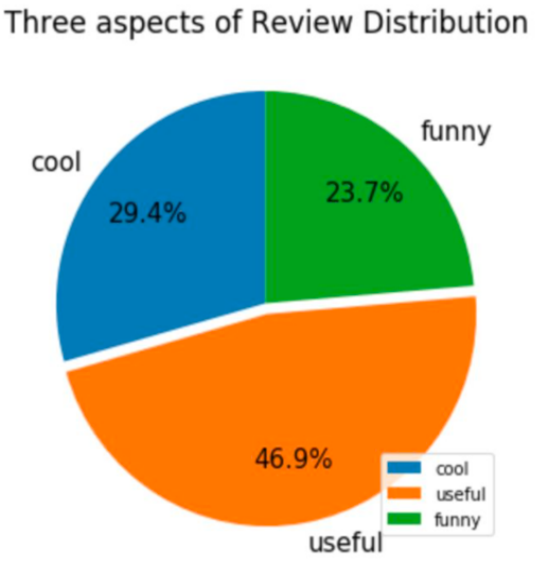

Figure 10: Number of Votes distribution

1.2.3 Ratings respective to review text length

Review text is also an important element we need to consider making a prediction for rating, so we analyzed the relationship between the length of the review and the rating it made. We noticed that as the trend for the length of review and the rating is opposite which means that longer-length review, lower rating 11. We assume that when user makes longer review, it is highly possibility that they are making complaints towards that business. However, after we process the validation set with our models, the length of review is actually redundant towards with the top 100.

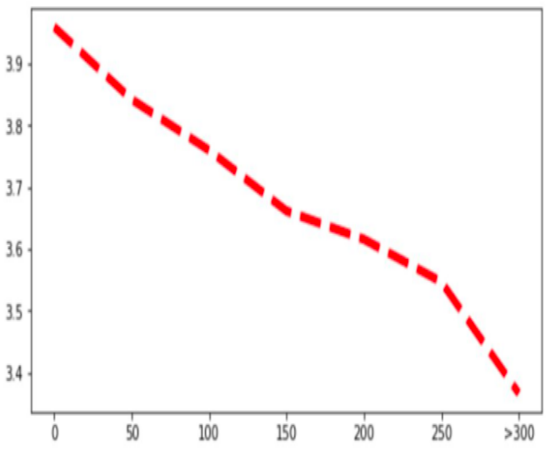

Figure 11: Rating with respect to review text length

1.2.4 Extra findings

We also noticed some other interesting facts we are not using in this task. The reason is either unbalanced distribution of data, or the feature not associated with our predictive task.

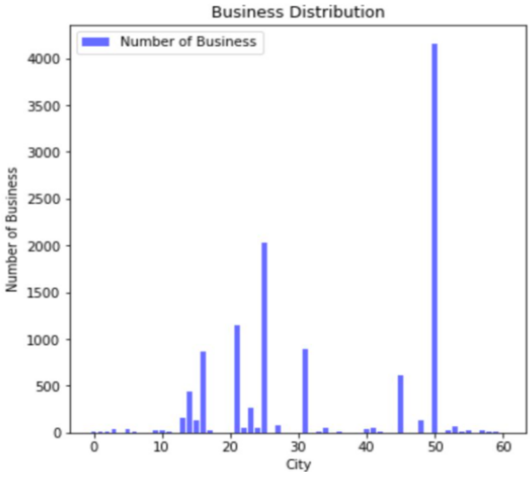

Figure 12: City of Business Distribution

The businesses from our dataset are mostly from Arizona (11,534), each one from California, Colorado, and South Carolina. The top ten cities are Phoenix, Scottsdale, Tempe, Mesa, Chandler, Glendale, Gilbert, Peoria, Surprise, Avondale. 13

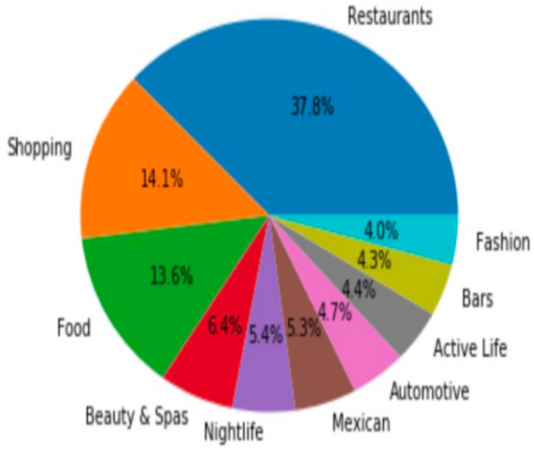

Figure 13: Top 10 Business Category

We also analyzed the top 10 categories that people tend to rate.13 And we found out people tend to give restaurant high rating. However, this finding maybe because Yelp have most of their businesses as restaurants.

2 PREDICTIVE TASK

2.1 Task Description

We will identify a predictive task to predict ratings of each business on Yelp when given each review text, reviewer id and user ids.

$$f(review\_text) = ratings \tag{1}$$

$$g(user, business, review\_text) = ratings \tag{2}$$

We will explore using the review text to predict ratings of the businesses. And then explore using the users and businesses to predict ratings.

2.2 Processing data

We will split the dataset into three portions: 60% for training set, 20% for test set, and 20% for validation set.
We loop through the entire training set to collect all the ratings and to get their average value. Since we already have the individual business and user rating and review count in our business and user set, we take them out and make a dictionary for them. And we count the number of different votes from each review in the training set and put them into a dictionary. Also, we preprocess the words using the method we have been used in text mining section. We remove the stopwords, lower the words, count and sort them. Then we choose the top n words, which is a parameter we need to tune. We also get all the text length from the training set and we store them into a list.

2.2.1 Special processing for classifcation models

Originally, the stars of the businesses are one decimal point accuracy. To enabling predict using different classification models, we round it to integers number from 1-5 for classification models.

Table 2: Process 1-decimal point precision to integer labels

Rounded stars	Real stars
1	1-1.4
2	1.5-2.4
3	2.5-3.4
4	3.5-4.4
5	4.5-5

2.3 Feature Selection

TF-IDF/Sentiment Analysis:
For these two models, we will use the review text to do the classification text to predict stars in whole numbers. We will analyze the common words and also add sentiment to each words based on the business stars overall.

Latent factor model:
We will use users and businesses and explore user bias and business bias to predict the ratings.

Ridge/Lasso Linear Regression:
From the exploratory analysis part, we decided to use some features with various combinations of them. We first try the review count of user and business and try to prove our analysis is correct, which indeed is correct. And based on the rating distribution part, we believe the average rating of user and average rating of business will be good features. Same with the text length, we found the longer length is, the lower the rating. Also, based on the interesting trends and the equal distribution of the three aspects of the review, we decide to use these votes to be one of our features.

2.4 Possible models

For baseline models, we are going to use user rating average and global rating average to predict the ratings.
For advanced models, we can predict ratings using the TF-IDF features and sentiment analysis if we transform the stars into integer numbers.
However, if we don't want to lose precision that much, and would like to keep the stars as they originally are, then Latent factor model and ridge regression are reasonable and ideal because they can predict on continuous values.

2.5 Evaluating models

We first categorize our models into two sub categories: concrete values and continuous value. Then, for each sub categories, we will evaluate our models through comparing the MSE of our models on validation sets.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{3}$$

3 Models

3.1 Baseline model using average

For the baseline solution, we are going to use the global average and user average rating to predict

the star of each business. The global average is calculated as the following way:

$$globalAverage = \frac{\sum_{reviews} rating}{total\_number\_of\_reviews} \tag{4}$$

The user average is calculated as the following way:

$$userAverage = \frac{\sum_{reviews\_user}^{all\_users} rating}{total\_number\_of\_reviews\_user\_did} \tag{5}$$

For each review, if we find that the user has been seen in our data, we will use the user average rating as our final rating for this business, or if our user has not been seen before, we will use the global average rating as our final rating for this business.

Table 3: MSE on different dataset

dataset	MSE
Train set	0.9805480025099351
Test set	1.0231760374832664
Validation set	1.2126291018433781

3.2 Advanced models using classification ideas

3.2.1 Sentiment analysis with SVM and logistic regressor

Sentiment analysis aims to determine the attitude of a speaker, writer, or other subject with respect to some topic or the overall contextual polarity or emotional reaction to a document, interaction, or event [4].

Because reviews really relates to ratings a lot, if we add sentiment to the words based on the stars of the review, we will have a better performance than solely just counting the common words. First to start the sentiment analysis, we will get 4000 word counts. 4000 most common words will be large enough to give us enough precision. Then, we need to have a threshold for our sentiment to be negative or positive. If it is negative, then we will not going to add word count to the words dictionary. It it is positive, then we will add one to the word count. To decide our sentiment threshold, we refer to the data analysis we did in the previous section. From figure 5 we can see that most of the ratings are between 3-5. Therefore, setting a threshold between 3 will be good.

Then to do the classification using words with sentiment, we decided to test on two different classifiers. The first one is the logistic regression. The second on will be SVM. We will compare which one's MSE is better.

With logistic regression: To optimize our solution, we are going to choose the λ with the least MSE.

Table 4: choosing lambda for logistic regression

λ value	MSE
0.001	0.6201472556894244
0.01	0.5953815261044176
0.1	0.5864290495314591
1	0.6116967871485943
10	0.6449966532797858
100	0.6719377510040161
1000	0.6835676037483266

Therefore, we are going to choose λ value as 0.1 to optimize our MSE.

With SVM: To optimize our solution, we are going to choose the λ with the least MSE.

Table 5: choosing lambda for SVM

λ value	MSE
0.001	0.6018239625167336
0.01	0.5850066934404283
0.1	0.6142904953145917
1	0.6452476572958501
10	0.6674196787148594
100	0.7453982597054887
1000	0.7555220883534136

Therefore, we are going to choose λ value as 0.01 to optimize our MSE.

Comparison: We could see that in comparison, there is no big difference on whether to use SVM or logistic regression in this case if we are choosing the best λ value for both of them. SVM is only slightly better than logistic regression.

3.2.2 TF-IDF feature

Tf–idf or TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general[2].

Term frequency tf(t,d), the simplest choice is to use the raw count of a term in a document[2]:

$$\mathrm{tf}(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}} \tag{6}$$

The inverse document frequency is a measure of how much information the word provides, i.e., if it's common or rare across all documents[2]:

$$\text{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|} \tag{7}$$

And tf-idf feature vector can be calculated:

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D) \tag{8}$$

We will processing the data first by removing all the punctuation and lower the cases, then we are going to stem all the words in the training dataset review texts. After processing the data, tfidf feature vectors can be extracted from the sklearn.feature_extraction.txt library[3]. In this case, we basically treat predicting star labels as doing a multi-class classification. Therefore, we are using a multi-class linear SVC to do the classification task. We optimize our solution by applying the λ value that gives us the least MSE. The chart for choose λ value as 10 are below:

Table 6: choosing λ for best MSE

λ **value**	**MSE**
0.01	0.6099397590361446
0.1	0.5922021419009371
1	0.5599062918340026
10	0.5559738955823293
100	0.5570615796519411

3.2.3 Overall strengths and weakness of classification models

From the comparison, we can see that both sentiment analysis and TF-IDF provides us with a better MSE than baseline model. However with choosing the best λ value for both models, TF-IDF performs a better MSE than sentiment analysis. This may result from the fact that sentiment analysis just add weight on the word count for common appear words, while TF-IDF will reflect both common words and rarely appear words. Therefore, TF-IDF ends to have a better performance than sentiment analysis.

Table 7: compare models

model	**best λ MSE**
sentiment analysis w/ LR	0.5864290495314591
sentiment analysis w/ SVM	0.5850066934404283
TF-IDF	0.5559738955823293

3.3 Advanced models predicting on continuous values

3.3.1 Latent factor model

$$f(u, i) = \alpha + \beta_u + \beta_i + \gamma u \cdot \gamma_i \tag{9}$$

Given one user and one business, we want to predict what rating the user will make for the business. This model projects user's preferences and business's properties into lower dimensional space. Therefore, we aims to minimize the objective function which is:

$$f(u,i) = argmin_{\alpha,\beta,\gamma} \sum_{u,i}(\alpha + \beta_u + \beta_i + \gamma u \cdot \gamma_i - R_{u,i})^2 + \lambda[\sum_u \beta_u^2 + \sum_i \beta_i^2 + \sum_i ||\gamma_i||_2^2 + \sum_u ||\gamma_u||_2^2] \tag{10}$$

And it is actually using the alternating least squares: [8]:

$$\alpha = \frac{\sum_{u,i \in \text{train}}(R_{u,i} - (\beta_u + \beta_i))}{N_{\text{train}}}$$

$$\beta_u = \frac{\sum_{i \in I_u} R_{u,i} - (\alpha + \beta_i)}{\lambda + |I_u|}$$

$$\beta_i = \frac{\sum_{u \in U_i} R_{u,i} - (\alpha + \beta_u)}{\lambda + |U_i|}$$

fix γ_i. Solve $\arg\min_{\alpha,\beta,\gamma_u} objective(\alpha, \beta, \gamma)$

fix γ_u. Solve $\arg\min_{\alpha,\beta,\gamma_i} objective(\alpha, \beta, \gamma)$

Figure 14: Update process

Table 8: MSE of latent Factor Model with different λs

λ_1 \ λ_2	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$	$1e^{-6}$
$1e^{-2}$	1.1320	1.14367	1.27326	1.16714	1.1520
$1e^{-3}$	1.1319	1.14267	1.26232	1.23122	1.142346
$1e^{-4}$	1.1318	1.16627	1.24535	1.14524	1.12535
$1e^{-5}$	1.1318	1.15352	1.32240	1.14459	1.14455
$1e^{-6}$	1.1318	1.15502	1.27817	1.12956	1.14502

Throughout the comparison, we found when $\lambda_1 = 1e^{-6}$ and $\lambda_2 = 1e^{-2}$, the MSE is the lowest. Then, we adjust the λ to make the number of dimensions of the user and business interaction matrix 8.

Table 9: Latent Factor Model with different Number of Dimensions

Number of dimensions	Validation set
12	1.142148527
24	1.142065245
36	1.141932761
48	1.141921342
60	1.1418846238
72	1.141675484

We noticed that the more dimension we have, the lower the MSE. The lowest MSE is of 1.141675484, however, it is still higher than the MSE of the Ridge Linear Regression model. We guess that the amount of the data we are using is not suffice to do the latent factor, in addition, since some features are strongly correlated with the rating, it is a good property for linear regression.

3.3.2 Ridge classifier model

From the exploratory analysis part, we decided to make some combinations of the features we choose. We first use data of the review count of user and business with the model and success to prove our analysis is correct. And based on the distribution of the data, it is also a good property for the field to be a good feature. As for the text length, we also found the longer length of the review is, the lower the rating. In addition, since three aspects (cool, interesting, useful) of the review are equally distributed, we will make these votes as one of our features.
After the analysis of the data features which are associated with the prediction, we noticed that most of the data we choose is either numerical or linearizable. Therefore, we decided to use linear regression. It is calculateed by:

$$y = X \cdot \theta \tag{11}$$

(X represents the features of data, and y represents the label we want to predict. θ is the coefficient that how features influence the label in a linearly way).

Feature	MSE (Validation Set)
[user average rating, business average rating]	0.955182915964
[user_review_count, business_review_count]	1.46527847939
[text length]	1.45119273238
[useful, cool, funny]	1.31613344182
[top_words]	1.10202541473
[vote, text length, average user rating, average business rating]	0.918189316572
[text length, top_words]	1.11936788518
[text length, (user_average rating, business_average rating) / 2]	0.94591371397
[vote, top_words, average user rating, average business rating]	0.783451232371

Figure 15: Features of Ridge Linear Regression with $\lambda = 1.0$

Firstly, we have calculated the MSE of prediction with every single feature we choose in the data exploring section (eg. top words, text length,). Next, we optimized linear regression model by reselecting features and combining some features based on the finding in the data exploring section. As you can see in table 1, we have chosen the combination of counts number for votes, top 100 words, average rating for the user and business as our final result of ridge linear regression and we got a lowest MSE 0.783451232371.

Table 10: λ Selection of Ridge Linear Regression

λ	MSE
0.1	1.1042133131
0.3	1.0135079073
0.5	0.9038975972
0.7	0.8488622982
1.0	0.7834512716
1.1	0.8257951064
1.4	0.9241515545

We then adjusted the parameter ([vote, top_words, average user rating, average business rating] feature) by updating λ value from 0.1 to 1.4 and then we realized that $\lambda = 1.0$ is the best one for the final result (see Table) since the MSE is lowest. Next step, we used PCA in order to reduce the dimension, but the MSE is pretty high, which means this method is not proper for this task. We guessed PCA may not be a good choice for this task.

3.3.3 Overall strengths and weakness of continuous value models

Table 11: Model Comparison

Model	MSE
Baseline Model	1.2126291018433781
Ridge Linear Regression	0.783451233716
Latent Factor Model	1.141675484

As can be seen from Table 11,gave the lowest MSE and it will be our model for this task. **Ridge Linear Regression:** The time complexity of Ridge Linear Regression is smallest. However, we need to pre-investigate which features are more associated with this prediction task. And we have to have the risk for the overfitting problem.

Latent Factor Model: We do not need do much pre-investigation about feature selection, because it won't use any features. Therefore, it will perform well on the data set which is large enough to make a more accuracy prediction. In our case, since we do not have a dataset which is big enough, and from the result, we can see that it is worse than our Ridge Linear Regression Model. In addition, it performs better as the dimension of the Latent Factor increases. However, it also costs much more runtime as dimension increases.

4 Literature related

4.1 About dataset

Our project is based on the Yelp Open Dataset where we predict star ratings using review text. Yelp, the crowd-sourced local business review forum, has been actively hosting dataset challenges in order to encourage people to share their discoveries in research in various topics involving the Yelp dataset.

4.2 Usage of the dataset

McAuley[5], one of the Round One Challenge Winners of the Yelp Dataset Challenge, present HFT, a model which combines latent rating dimensions and latent review topics. By using this model, they align hidden factors in ratings and hidden topics in reviews. Thus, they are able to recommend new products or identify useful reviews by using only a few reviews.
Feng[6] use a hybrid approach between collaborative filtering and content-based filtering. With a combination of methods, such as Binary Decision Tree Regression and K-Nearest Neighbors, they take advantages of users' and business' profiles, while maintaining the advantages of a neighborhood model.

4.3 State-of-art

One state-of-the-art method currently employed to predict star ratings from reviews is using Term Frequency Inverse Document Frequency (TF-IDF) vectors and fitting a classifier using a Linear SVM. TF-IDF helps identify more important words by normalizing the count of each word in each review text by the frequency of that word appearing in all review texts. Once fitting a classifier using the vectorized text, one can predict ratings.

4.4 Similar dataset studied

Liu[7], who predict ratings based on TripAdvisor Reviews, use TF-IDF vectors and MSE to conclude the accuracy of their model.

4.5 Compare between our work and theirs

We found that conclusions from existing work and the state-of-the-art method are similar to our findings. We have used techniques such as training an SVM and building a latent factor model while working toward results.

5 Results and conclusions

Between sentiment analysis and tf-idf vectors: TF-IDF vector performs better because in our case, we use the stars of reviews as sentiment level with SVM and logistic regressor, and we

only counted how many words are most frequent ones. However, using this method, we won't take rarely seen words that could really be characteristics of one label into predicting consideration. That's why tf-idf performs a better MSE than using the star of reviews as sentiment level. The MSE of tf-idf is about 0.03 lower than the sentiment analysis.

Between Ridge Linear Regression and Latent Factor Model: The reason that the Ridge Linear Regression performs well is that the sets of User and Business contain many useful features strongly associated with rating. The MSE is lower than others by around 0.2-0.3. For Latent Factor Model, since the size of dataset is not suffice, the result of the Latent Factor Model is worse than the good feature Linear Regression. For Linear Regression Model, we chose some features and make some adjustments such as combinations on them based on the result from our data exploring section. The final feature vector we have chosen is [vote, top_words, average user rating, average business rating], which generates the lowest MSE. Our thought is the vote (useful, cool, funny), user rating, and business rating are all equally distributed, and their trends are all strongly correlated with ratings.

References

[1] https://www.kaggle.com/yelp-dataset/yelp-dataset#yelp_academic_dataset_business.json

[2] https://en.wikipedia.org/wiki/Tf%E2%80%93idf

[3] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[4] https://en.wikipedia.org/wiki/Sentiment_analysis

[5] Julian McAuley and Jure Leskovec, Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text,
https://www.yelp.com/html/pdf/YelpDatasetChallengeWinner_HiddenFactors.pdf

[6] Yifei Feng and Zhengli Sun, Yelp User Rating Prediction,
http://cs229.stanford.edu/proj2014/Yifei%20Feng,%20Zhengli%20Sun,%20Yelp%20User%20Rating%20Prediction.pdf

[7] Dangyi Liu, Yu Chai, Chenxi Zheng, Yilun Zhang, Rating Prediction Based on TripAdvisor Reviews,
https://cseweb.ucsd.edu/classes/wi17/cse258-a/reports/a013.pdf

[8] http://cseweb.ucsd.edu/classes/fa18/cse158-a/slides/lecture7_annotated.pdf