Predict Overall Ratings on Beer Reviews Dataset

Shuhao Chang, Wei He
Department of Computer Science and Engineering
University of California, San Diego
s2chang@ucsd.edu, whe@ucsd.edu

Abstract

With machine learning field becomes more and more popular in recent years, recommendation systems are widely used in various fields such as e-commerce, medical system and entertainment mobile app with huge benefit. Based on several information user provides previously, it is possible to learn from these data and provide an accurate prediction. In this paper, we are going to study a beer review data set, analyze useful features from dataset and use them to predict user's overall rating. In addition, we will use different models and analyze their rating prediction performance in this dataset with some error measurement metrics. In the end, we will describe our analysis results with these models and our interpretation of these results.

1 Introduction

In recent decades, Internet starts blooming. More and more people prefer purchasing items on e-commerce websites and watching videos online. In the meantime, some of these users have wrote their comments and reviews, which have been the most important data for companies to analyze and expand their business. Meanwhile, personalizing customer's main product page based on their previous purchase and reviews have large impact on customer decision and their time to stay on this product page. Therefore, a recommendation system, involving the prediction of user action and rating, will become a significant part of these e-commerce online shopping and video streaming websites.

In this paper, we will use a beer review dataset consisting 13 features and 1.5 millions of records. We will use some of these features to predict user's overall rating on each beer. In the beginning, we will perform exploration data analysis on these dataset to find out the relationship between each feature and overall rating. Moreover, we will split the dataset into three parts, training, validation and test sets, where the size of validation and test set are 100,000. Next, we will use average overall rating as the baseline model, and use linear regression, latent factor, random forest and neural network models with useful features we analyzed in EDA to build our model. In addition, we will tune our models with validation set and use Root Mean Square Error (RMSE) as error metrics. In the end, we will use the best model with best parameters to test on our test set and provide our result and analysis of it.

The report is divided into 6 sections. Section 3 describes the data and its basic statistics and properties alongside with the data analysis. Section 4 identifies the prediction task we are going to study with this dataset, the baseline mode we built for it for comparison, and the method that we use to evaluate the model performance. Section 5 introduces the algorithms and models, and the ways we try to optimize them. Section 2 describes the literature related to the problem we are studying. The results are given in Section 6. Lastly, the conclusion is given in Section 7.

2 Related Literature

In this project, we have used an existing dataset from data.world[3], as described in Exploration of Data Analysis. This dataset consisted the reviews Beeradvocate[1], which is an independent community of enthusiasts and professionals dedicated to supporting and promoting better beer. It has begun to record customers' reviews over 10 years, so that this dataset includes all 1.5 million reviews up to November 2011. It was used by other data scientists in e-commerce websites focusing on beer to learn from previous customers' reviews and better predict customer's decision. They are trying to find out the answer of following questions:

1. Which beer style should customer try if they prefer taste and appearance of a beer?
2. Which brewery produces weakest beer by ABV?
3. Which beer is the most popular that a new customer may like?
4. What is the overall rating a customer will give if they had previous purchasing experience on beers?

With these answers from questions above, companies are more likely to generate a customized beer page for user and attract them to make purchases, leading the increase of companies revenue.

In this project, we are focusing on the last question, which is using previous review data to predict users overall rating on a given beer. This problem has not been studied yet, so that we can try to solve this problem by using the methodologies we learned in CSE 258 and other novel regression models we have learned previously.

Another novel model builds a network of substitutable and complementary products for the recommender system[10]. This is a different approach from our Regression models. This is a network model that in built on the relationship between each products. After building the network, the model is able to identify the beers related to the one the users have purchased. As a result, related beers will be recommended to the users.

Similar datasets like Amazon Product Reviews and Google Local Business Reviews have been studies in a similar way. They have been contributed into separated categories such as classification, Regression and clustering. All these data we used in academic research and industrial data analytic in multiple disciplines are related to understand customer product experiences. Various machine learning and deep learning methods have been used in these studies. The study we presented in this paper is only answering a question among multiple questions that companies and researchers would like to know.

For classification problem, current industry and academic research use deep learning networks, which uses deep learning send the input (the data of images) through different layers of the network, with each network hierarchically defining specific features of images. In addition, current state-of-art methods studying type of regression review datas are the following: **Simple Linear Regression**, **Polynomial Regression**, **Support Vector Machine**, **Random Decision Tree**, **Random Forest** and **Artificial Neural Network**.

The conclusion we came up in this project are similar to existing works. Both works have found that by using the features of palate rating and taste in the reviews are most important in determining the overall rating of a beer. In addition, user tends to give high rating to products that are popular if they have purchased previously.

Table 1: Basic statistics of the Beer Review dataset

Metadata	Amount
Number of review	1,586,615
Number of user	33,388
Number of beer	66,055
Number of brewery	5,840
Number of beer style	104
Number of feature	13

3 Exploratory Data Analysis

3.1 Data Collection

This dataset we choose for this project come from data.world [3]. Originally, we tried to use the beer rating data from Prof. Julien's dataset [2]. But the dataset is no longer available on the links provided by Prof. Julien, because the dataset was removed as requested by RateBeer [7] and BeerAdvocate [8]. So we decide to search online and choose a similar beer rating dataset on data.word website.

This dataset contains 1,586,615 rating reviews with 13 features:

- brewery_id
- brewery_name
- review_time
- review_overall
- review_aroma
- review_appearance
- review_profilename
- beer_style
- review_palate
- review_taste
- beer_name
- beer_abv
- beer_beerid

The basic statistics are shown in table (Tbl. 1). In this beer review dataset, each review contains brewery id, brewery name, review time (which is in Unix epoch format), overall rating, aroma rating, appearance rating, palate rating, taste rating, profile name, beer style, beer name, beer abv, and beer id.

3.2 Data Cleaning

In order to use features in a better way, we have done data cleaning of this beer review dataset.

First, we convert the review time (in Unix Epoch Format) into four separate columns: year, month, day and hour. In this case, we can further convert month, date and hour with one-hot encoding and use them as features in predicting beer's overall rating.

Furthermore, we treat original review_profilename as username, and assign a random unique userId to each profilename. In addition, we drop columns **brewery_name** and **beer_name**, which are redundant information with **brewery_Id** and **beer_Id**

In the end, we reorder the columns of data as following: brewery_id, beer_id, user_id, review_overall, review_aroma, review_appearance, review_palate, review_taste, beer_abv, beer_style, review_year, review_month, review_day, review_hour. These 16 features are the ones that we are going to use in this study.

- brewery_id
- beer_id
- user_id
- review_overall
- review_aroma
- review_appearance
- review_palate
- review_taste
- beer_name
- beer_styles
- beer_abv
- beer_style
- review_year
- review_month
- review_day
- review_hour

3.3 Data Analysis

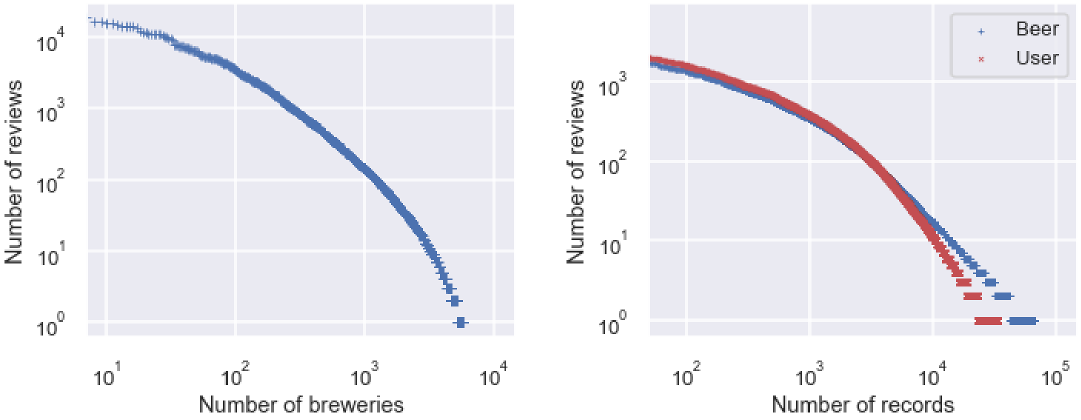
In this section, we take an exploratory data analysis on the whole dataset from two perspectives: 1) the distribution of reviews of overall ratings, among breweries, beers and users, time scales, etc. 2) the correlation and distribution between features. We hope to explicitly demonstrate the property of dataset and further provide inspirations for modeling.

3.3.1 Distribution of reviews

Table 2: Distribution of overall ratings from 1 to 5

Rating (1-5)	Percentage
Rating of 5	26.20%
Rating of 4	55.75%
Rating of 3	14.13%
Rating of 2	3.23%
Rating of 1	0.70%

In the beginning, we analyzed the distribution of overall ratings in the training dataset (Tbl.2). The statistics shows that more than 80% of the reviews have an overall rating of 4 or 5, and only around 4% of reviews have overall rating less than 2. It turns out that most of the customers are satisfied with the beer they purchased.

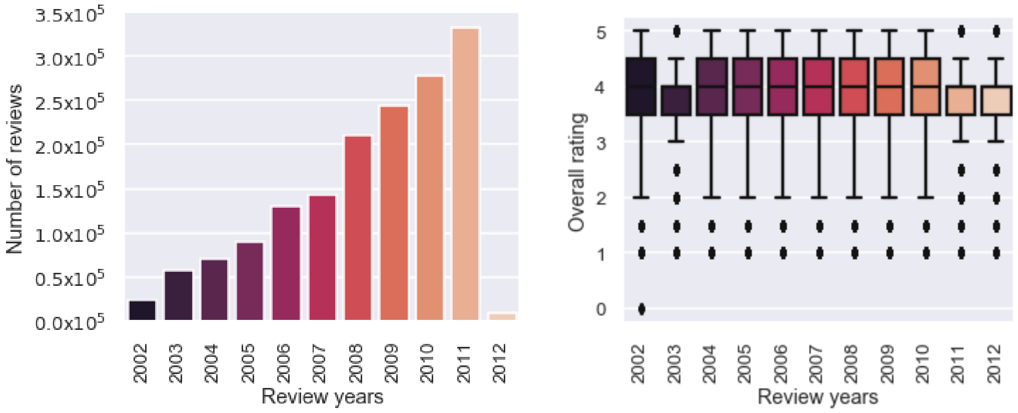


(a) Reviews of breweries

(b) Reviews of beers and users

Figure 1: Power law in beer reviews.

Figure 1 shows the power law phenomenon in beer reviews among different breweries, beers and users. According to the log scale plot, the distributions of review number among brewers, beers and users are quasi linear. This is a similar distribution as in social network. It describes that a small amount of breweries, beers and people correspond to a large amount of reviews, and vice versa. Some popular breweries get more than 10^4 reviews and some beers and people get or give more than 10^3 reviews.

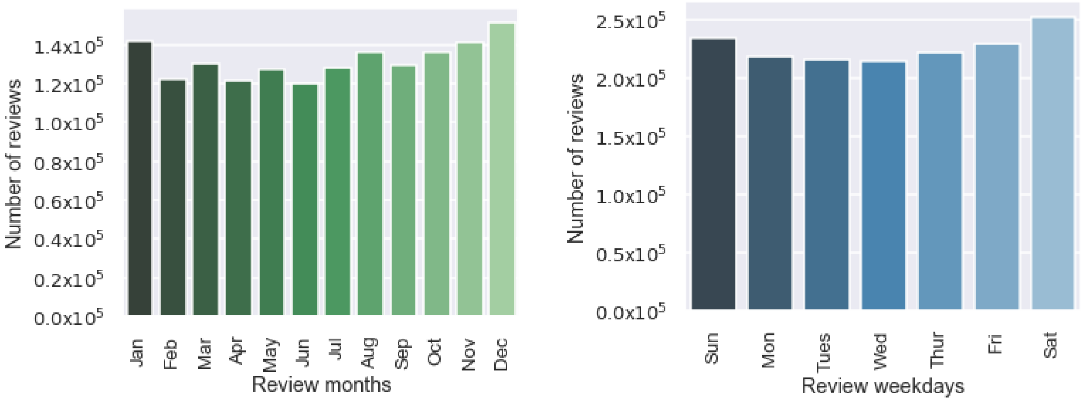


(a) Review numbers of different year

(b) Average ratings of different years

Figure 2: Distribution of reviews among years.

Figure 2 shows the distribution of review numbers and average ratings among years. According to a basic analysis, the majority of data are collected from years between 2002 and 2012. We calculate the number of reviews of each year and shows in Figure 2(a). It demonstrates that the reviews are monotone increasing since the development of online shopping and information collection techniques. The dataset is published early in 2012, which causes incomplete records. Figure 2(b) shows the distribution of overall ratings in each year. For most of the years, the median of overall ratings is 4.0 and the majorities are located between 3.5 and 4.5. This demonstrates that the ratings given to beers are highly centralized and skewed, which makes it more difficult to train a robust rating prediction model. Meanwhile, this observation enlightens us to compare our models with trivial mean predictor since the latter may have a not-bad performance as well.



(a) Review numbers of different months

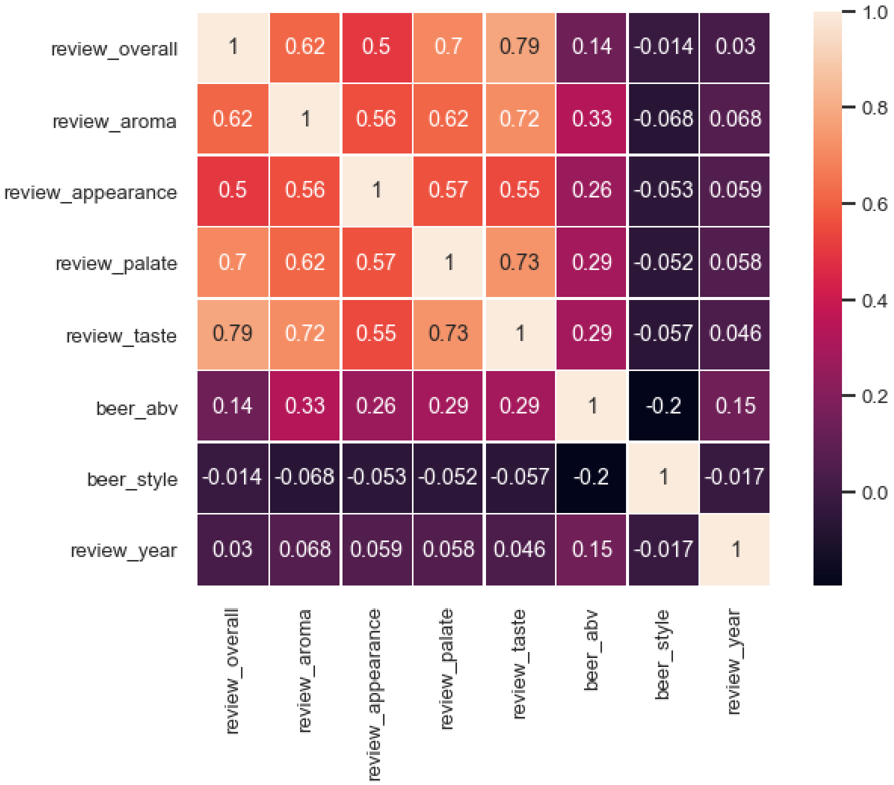
(b) Average ratings of different weekdays

Figure 3: Distribution of reviews among months and weekdays

In Figure 3, we calculate the number of reviews in different months and days in a week similar to Figure 2(a). Both Figure 3(a) and 3(b) shows a U shape changing tendency, which means a high review number at the beginning and end of a year and a week. This phenomenon is easy to explain. There are more holidays and festivals (e.g. Thanksgiving, Christmas, New Year's Day, etc.) during November, December and January, when people are likely to buy beers, drink and give reviews afterwards. It is the similar case in a week since people are more likely to drink on weekends. These observations enlighten us that the seasons in a year and whether the date is weekday or weekend may have impact on the reviews. In the following modeling process, we may consider the time stamp as an important feature.

3.3.2 Correlation between features

It is important to formulate the pattern of data and extract features. But it is more important to know the relationship between features, which is a guide to help us select useful features. Through calculating the correlation coefficients among features, we are able to know which one is highly correlated to the overall ratings. Figure 4 is the correlation matrix among the overall ratings and the ratings of aroma, appearance, palate, the alcohol by volume of beer, style of beer as well as the year of review. In this heat map, a brighter background color represents for a higher correlation coefficient and vice versa. It is obvious that the partial reviews towards the aroma, appearance, palate and taste is the most important factors that influence overall ratings. Among these four partial ratings, the review of taste has the largest impact (i.e. 0.79 positive correlation) on overall ratings. And people do not care so much about the appearance of a beer (i.e. only 0.5 positive correlation). Since the problem we want to solve is to predict the overall rating a person will give to a beer, we will not use the partial ratings in the same record. Instead, we calculate the historical average ratings of the same user and the same beer as the features. Therefore, it is guaranteed that the test data is used only once at the evaluation part.



	review_overall	review_aroma	review_appearance	review_palate	review_taste	beer_abv	beer_style	review_year
review_overall	1	0.62	0.5	0.7	0.79	0.14	-0.014	0.03
review_aroma	0.62	1	0.56	0.62	0.72	0.33	-0.068	0.068
review_appearance	0.5	0.56	1	0.57	0.55	0.26	-0.053	0.059
review_palate	0.7	0.62	0.57	1	0.73	0.29	-0.052	0.058
review_taste	0.79	0.72	0.55	0.73	1	0.29	-0.057	0.046
beer_abv	0.14	0.33	0.26	0.29	0.29	1	-0.2	0.15
beer_style	-0.014	-0.068	-0.053	-0.052	-0.057	-0.2	1	-0.017
review_year	0.03	0.068	0.059	0.058	0.046	0.15	-0.017	1

Figure 4: Correlation matrix of features

Besides partial ratings, we also consider beer alcohol, style and review time in this analysis. Since beer alcohol is not partial ratings, it seems to be a good feature with a 0.14 positive correlation

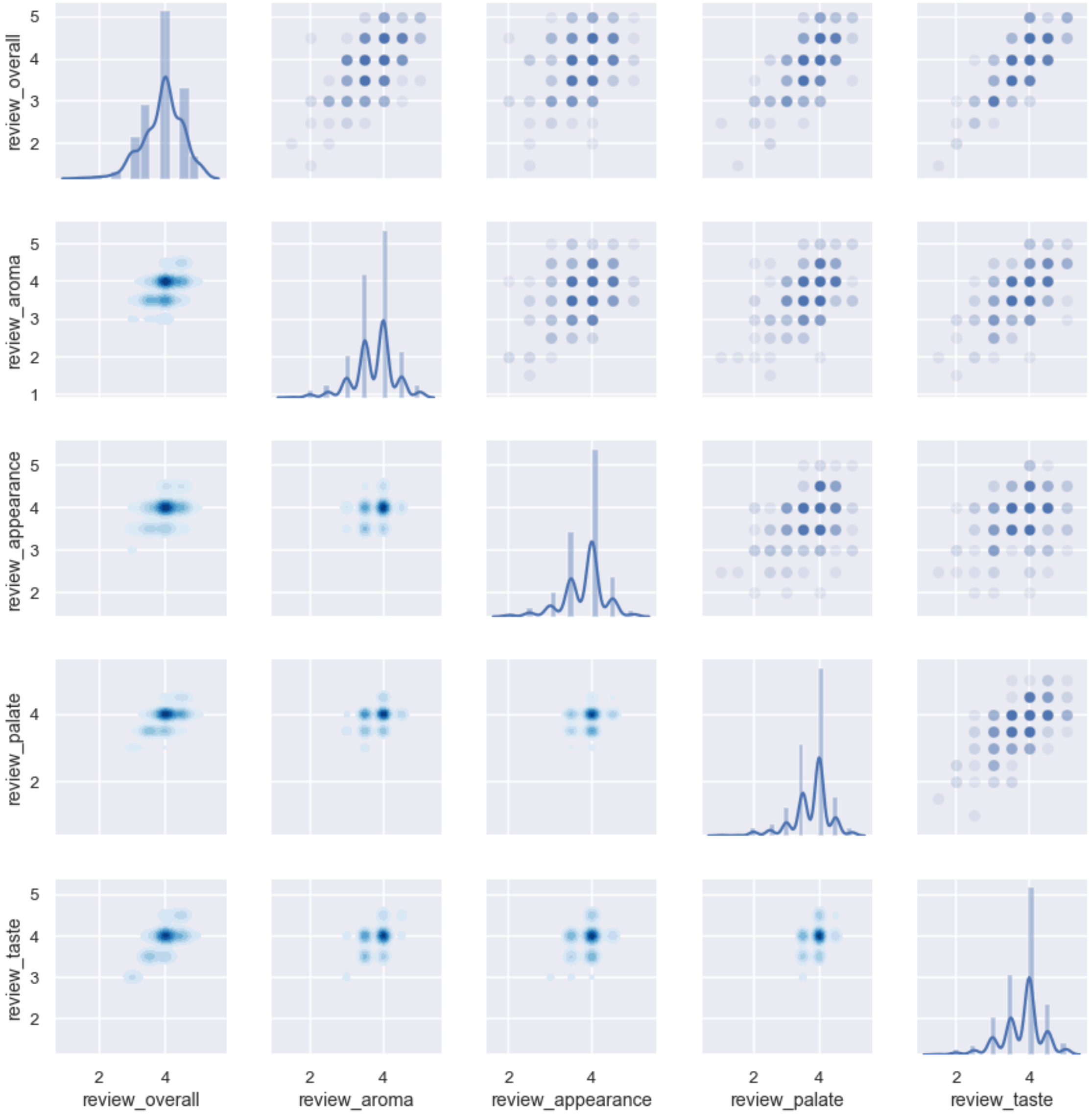


Figure 5: Joint distribution of reviews

with the overall ratings and much higher correlation with partial ratings. It is interesting that beer ABV influences the aroma a lot, which explains the fragrance of a beer somewhat comes from alcohol. The other two features seem not so correlated with ratings. This is because we use the style ID and the original value of time to calculate the linear correlation coefficient, which may not disclose the true influence of them. In the modeling part, we may use one-hot encoding to reformulate these features so that the non-linear impact is considered.

We further take an analysis on the joint distribution of overall reviews and partial reviews and give the results in Figure 5. The diagonal figures are probability distribution functions (PDF) of reviews. The upper triangle contains the scatter plots and lower triangle contains kernel density estimations (KDE). From this figure, we notice that both overall ratings and partial ratings are centralized at 4.0. The KDE plots of the column of overall reviews are more dispersive than others, which means the rating of one aspect of the beer may not be the decisive factor to the overall rating. On the contrary, a rating of one aspect as 4.0 means a same rating of 4.0 for another aspect with high probability. This observation tells us that it is not enough to only use partial ratings to predict overall ratings. In other words, other features like beer ABV, styles and review time may also be important factors.

4 Prediction and method of Evaluation

In this project, we will try to predict the overall beer rating given a userId and beerId based on previous reviews data in the given training set. If we come up an accurate prediction model, this model can be used in real e-commerce system to predict the overall beer rating this user may give, and further customize the product page for users.

We also introduce the Root Mean Square Error(RMSE) as our measuring metrics to differentiate performance between models where Y is the true label and $\hat{Y}$ is the prediction:

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

The baseline model we use is the global average overall rating. With the mean and variance formula, the RMSE will be the square root of variance, which can be treated as a baseline model.

$$\text{Mean}\,\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$$

$$\text{Variance} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y})^2$$

Firstly, we have done the data cleaning by converting timestamp into year, month, day and hour. Then we split split our dataset into three parts: training, validation and testing sets. The size of validation and testing datasets are 100,000, while the size of training dataset is 1,386,615. It may be the case that original dataset is sorted in some order. With random shuffling dataset, we can test our models in a more random way.

In **Linear regression** and **Ridge Regression**, we are going to use following features and their different combination as different models:

- the average aroma rating of this beer in training set
  - It is obtained by calculating the sum of aroma rating in training set and the total number of aroma rating in training set
- the average appearance rating of this beer in training set
  - It is obtained by calculating the sum of appearance rating in training set and the total number of appearance rating in training set
- the average palate rating of this beer in training set
  - It is obtained by calculating the sum of palate rating in training set and the total number of palate rating in training set
- the average taste rating of this beer in training set
  - It is obtained by calculating the sum of taste rating in training set and the total number of taste rating in training set
- the number of appearance of this userId in training set
  - We generate userId for each profile name in original dataset. Then we calculate the unique number of userId in the training set splitted from origin dataset

- the number of appearance of this beerId in training set
  - Since beerId is given in the original dataset as a part of the features, we only need to calculate the unique number of userId in the training set splitted from origin dataset
- the average overall rating this userId has given in training set
  - For each userId, We first generate a list of all overall rating that each userId. Then for each user, we can calculate the average overall rating that this userId has given
- is the average overall rating this beerId has received in training set
  - For each beerId, We first generate a list of all overall rating that each beerId. Then for each beerId, we can calculate the average overall rating that this beerId has given

In **Ridge Regression**, we will use the validation set to adjust the penalty term (lambda) which has regularizes the coefficients, so that we can find the best lambda that has the lowest RMSE of validation set.

In **Random Forest**, we select the following 14 features for the random forest model, where the *beer_review*, *user_review*, *style_review* is the average rating for each beer, user and beer style from the historical records.

- beer_review_aroma
- beer_review_appear
- beer_review_palate
- beer_review_taste
- user_review_aroma
- user_review_appear
- user_review_palate
- user_review_taste
- style_review_aroma
- style_review_appear
- style_review_palate
- style_review_taste
- beer_abv
- review_year

For each beerId, we will have lists of aroma rating, appearance rating, palate rating, taste rating. So for each beerId, we can calculate the average aroma rating, average appearance rating, average palate rating, average taste rating.

For each userId, we will have lists of aroma rating, appearance rating, palate rating, taste rating. So for each userId, we can calculate the average aroma rating, average appearance rating, average palate rating, average taste rating.

For each style, we will have lists of aroma rating, appearance rating, palate rating, taste rating. So for each style, we can calculate the average aroma rating, average appearance rating, average palate rating, average taste rating.

beer abv and review year have been given in the dataset. For review year, we have converted it into one-hot encoding in our pre-processing section.

In **Latent Factor**, we will only use over rating from dataset into our singular value decomposition for each user and each beer. This feature is given in the dataset, which we don't need to process.

All these four models are appropriate because they were designed for regression problem, the same as the predict problem that we have: to predict the overall rating of a given userId and beerId. We will use all these models and their best combination of features to test against our test set, and provide the RMSE result in the result section. With such result, we can know which model has the best performance in test set.

Since our goal is to predict user's overall rating of an unseen beer, the model will only have access to the user Id, beer Id and brewery id (if necessary) of the test set. No other review ratings will be given to the model during the prediction. In this case, we are able to best simulate the real use case of future unseen item rating prediction.

5 Methodology

5.1 Linear Regression

After evaluation of data analysis, we found some relationship between every feature and the overall rating, so linear regression and ridge regression model will be the first two models we try to use. Linear Regression model is a process to learn the relationships between features and parameters to predict read valued outputs, while Ridge regression is a linear regression with an l2 regularizer.

5.1.1 Linear Regression:

$\text{rating}(u, b) = \theta_0 + \theta_1 \times F_1 + ... + \theta_n \times F_n$

where u is the user Id, b is the beer Id, and F represents each feature we use in this linear regression. In this model, we have combined using all important features that we found in Exploration Data Analysis previously. The cost function is

$$\sum_{i=1}^{n}(y_i - \hat{y_i})^2 = \sum_{i=1}^{n}(y_i - X_i \times \theta_i)$$

In Exploration Data Analysis, we found that aroma rating, appearance rating, palate rating and taste rating are positively related to overall rating. Therefore, we decided to use the average of these four rating categories as features in our Linear Regression Model.

1. **First linear regression model:**

   $$\text{rating}(u, b) = \theta_0 + \theta_1 \times R_1 + \theta_2 \times R_2 + \theta_3 \times R_3 + \theta_4 \times R_4 + \theta_5 \times F_5 + \theta_5 \times F_6$$

   where R_1 is the average aroma rating of this beer in training set, R_2 is the average appearance rating of this beer in training set, R_3 is the average palate rating of this beer in training set, R_4 is the average taste rating of this beer in training set, F_5 is the number of appearance of this userId in training set, F_6 is the number of appearance of this beerId in training set.

   The RMSE of this model in validation set is 0.736, while its RMSE performance on training set is 0.615. In order to further optimize the performance, we try to change our model by removing some features.

2. **Second linear regression model:**

   $$\text{rating}(u, b) = \theta_0 + \theta_1 \times R_1 + \theta_2 \times R_2 + \theta_3 \times R_3 + \theta_4 \times R_4$$

   where R_1 is the average aroma rating of this beer in training set, R_2 is the average appearance rating of this beer in training set, R_3 is the average palate rating of this beer in training set, R_4 is the average taste rating of this beer in training set.
   The RMSE of this model in validation set is 0.735, while its RMSE performance on training set is 0.615.

3. **Third linear regression model:**

   $$\text{rating}(u, b) = \theta_0 + \theta_1 \times R_1 + \theta_2 \times R_2 + \theta_3 \times R_3 + \theta_4 \times R_4 + \theta_5 \times F_5 + \theta_5 \times F_6$$

   where R_1 is the average aroma rating of this beer in training set, R_2 is the average appearance rating of this beer in training set, R_3 is the average palate rating of this beer in training set, R_4 is the average taste rating of this beer in training set, F_5 is average overall rating this userId has given in training set, F_6 is the average overall rating this beerId has received in training set.

The RMSE of this model in validation set is 0.794, while its RMSE performance on training set is 0.577.

In the end, we choose the second linear regression model to validate on test set, where it has RMSE performance of 0.728, which is worse than baseline.

5.1.2 Ridge Regression:

Ridge Regression is a regular linear regression with an l2 regularizer. Its cost function is

$$\hat{\beta}_{\text{ridge}} = \sum_{i=1}^{n}(y_i - \hat{y_i})^2 = \sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

The penalty term (lambda) regularizes the coefficients such that if the coefficients take large values the optimization function is penalized. So, ridge regression shrinks the coefficients and it helps to reduce the model complexity and multi-collinearity.

In Ridge Regression, we also used three models we used in linear regression above to perform our testing. In our experience, we tried to increase or decrease the model complexity by adjusting penalization term λ with used of validation set to find the best λ with lowest RMSE. In the end, we will use the model with best λ on validation set to test on our test set.

1. **First Ridge regression model:**

$$\text{rating}(u, b) = (\theta_0 + \theta_1 \times R_1 + \theta_2 \times R_2 + \theta_3 \times R_3 + \theta_4 \times R_4 + \theta_5 \times F_5 + \theta_5 \times F_6) + \lambda \sum_{j=1}^{p} \beta_j^2$$

   where R_1 is the average aroma rating of this beer in training set, R_2 is the average appearance rating of this beer in training set, R_3 is the average palate rating of this beer in training set, R_4 is the average taste rating of this beer in training set, F_5 is the number of appearance of this userId in training set, F_6 is the number of appearance of this beerId in training set.

   The best λ we found is 0.01 on validation set, where its RMSE is 0.736. In the meantime, its RMSE on training set is 0.615 with $\lambda = 0.01$.

2. **Second Ridge regression model:**

$$\text{rating}(u, b) = (\theta_0 + \theta_1 \times R_1 + \theta_2 \times R_2 + \theta_3 \times R_3 + \theta_4 \times R_4) + \lambda \sum_{j=1}^{p} \beta_j^2$$

   where R_1 is the average aroma rating of this beer in training set, R_2 is the average appearance rating of this beer in training set, R_3 is the average palate rating of this beer in training set, R_4 is the average taste rating of this beer in training set.

   The best λ we found is 0.1 on validation set, where its RMSE is 0.735. In the meantime, its RMSE on training set is 0.615 with $\lambda = 0.1$.

3. **Third Ridge regression model:**

$$\text{rating}(u, b) = \theta_0 + \theta_1 \times R_1 + \theta_2 \times R_2 + \theta_3 \times R_3 + \theta_4 \times R_4 + \theta_5 \times F_5 + \theta_5 \times F_6 + \lambda \sum_{j=1}^{p} \beta_j^2$$

   where R_1 is the average aroma rating of this beer in training set, R_2 is the average appearance rating of this beer in training set, R_3 is the average palate rating of this beer in training set, R_4 is the average taste rating of this beer in training set, F_5 is average overall rating this userId has given in training set, F_6 is the average overall rating this beerId has received in training set.

The best λ we found is 100 on validation set, where its RMSE is 0.792. In the meantime, its RMSE on training set is 0.577 with $\lambda = 100$.

In the end, we choose the second linear regression model to validate on test set, where it has RMSE performance of 0.729, which is worse than baseline.

5.2 Random Forest

Random forest is an ensemble model for classification and regression problems. It is first proposed by Ho et al.[6] and further extended by Breiman et al.[4]. It consists of a series of random decision trees and gives the classification or regression results through a *voting* procedure of trees. Random forest adopts a *bootstrap aggregating* idea, which means each decision tree in the forest will randomly select a set of features to train with. Through such process, random forest is able to overcome the over-fitting problem which is often the case in classification and regression problems. Compared to linear regression models, random forest is able to consider the relationship between features. Each node in a decision tree will select a feature for conditional judgment and the choice of a lower level node is influenced by the output of its parent node. Shi et al.[13] points out that random forest models can be viewed as a weighted neighborhood schemes like k-nearest neighborhood algorithm.

In this paper, we select the following 14 features for the random forest model, where the *beer_review*, *user_review*, *style_review* is the average rating for each beer, user and beer style from the historical records.

- beer_review_aroma
- beer_review_appear
- beer_review_palate
- beer_review_taste
- user_review_aroma
- user_review_appear
- user_review_palate
- user_review_taste
- style_review_aroma
- style_review_appear
- style_review_palate
- style_review_taste
- beer_abv
- review_year

The hyper-parameters we choose for this model is as follows. The max-depth of random forest is 5. The number of estimators (i.e. random decision trees) is 500. By increasing the max depth, we increase the size (i.e. number of nodes) of the forest by exponential speed. While increasing the number of trees means a linear augment of the forest. With a shallower but broader forest, we not only reduce the time consumption of training, but also improve the robustness for the model.

5.3 MLP Neural Network

Multilayer Perceptron (MLP) is one of the earliest developed backpropagation artificial neural networks. It consists an input layer, an output layer and multiple hidden layers. Each node in hidden layers calculates the weighted sum of the output of previous nodes. A non-linear activate function is then applied to this sum so that the neural network is able to consider non-linear relationship between features. Figure 6 is a theoretical model of MLP neural network. In the input layer, there is a bias term involving into the following computation with other features. ω is the weights of previous nodes, which is what we are going to train. σ represents for the activate functions. Normal choices for activate function could be *sigmoid*, *tanh*, *ReLU* and other transformations of *ReLU*.

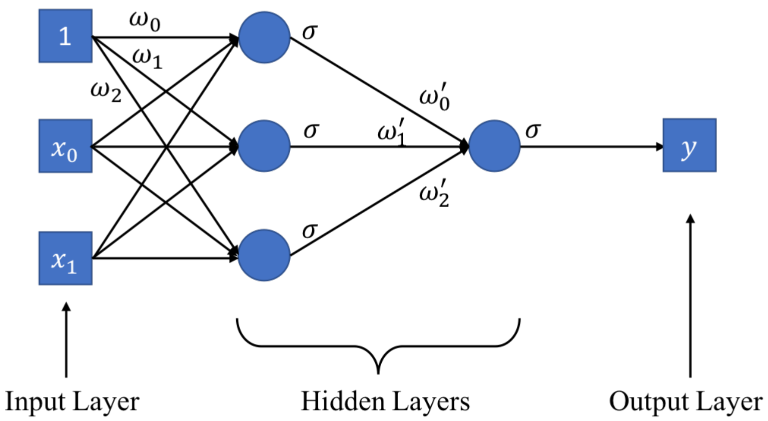


Figure 6: Theoretical model of MLP neural network

The actual model we implement for this problem has the following parameters. There are five hidden layers with two dense layer with 64 kernels, one dropout layer in the middle with a 10% drop probability and another two dense layers with 32 kernels. The dropout layer helps to prevent the neural network from overfitting. The activate function we choose is *ReLU*. We adopt an Adam optimizer as the gradient descent algorithm and set the learning rate as a fixed value 0.001. The batch size is 128. The model is trained in 50 epochs with an early stop mechanism. The features we use are the same with random forest model as show above.

5.4 Latent Factor

Latent factor model is a kind of collaborative filtering method which is able to learn the latent variable vector of users and items from a low dimension space. It has been widely used in recommendation systems and performs as the state-of-the-art methods. In this section, we describe the theoretical functions of a naive version and a full version of latent factor model. We also give the empirical value of hyper parameters in the models.

5.4.1 Naive Latent Factor

We first adopt a naive version of latent factor model which only considers the independent biases of users and beers. This model can be denoted as follows:

$$\text{rating}(u, b) = \alpha + \beta_u + \beta_b$$

where α represents for the average estimation of all predictions, β_u and β_b represents the biases of a user and a beer. β_u and β_b describes how a user tends to give a higher or lower rating and a beer tends to receive a higher or lower rating. The optimization problem we are going to solve can be denoted as:

$$arg\,min_{\alpha,\beta} \Sigma_{u,b}(\alpha + \beta_u + \beta_b - R_{u,b})^2 + \lambda(\Sigma_u \beta_u^2 + \Sigma_b \beta_b^2)$$

We take the mean square error as the optimization function and add a regularization part with an adjustable hyper parameter λ. We choose $\lambda = 10^{-4}$ in this case.

5.4.2 Latent Factor

Based on the naive latent factor, we build the full latent factor model with the addition of a cross term as follows:

$$\text{rating}(u, b) = \alpha + \beta_u + \beta_b + \gamma_u \cdot \gamma_b$$

where γ_u and γ_b represents how a user will rate for this exact beer. An accurate estimation is to take the singular value decomposition (SVD) on the rating matrix. However, due to the huge size of this matrix, we are not able to do so with limited computation resources. Gradient descent can be an available solution. The optimization function can be denoted as :

$$arg\,min_{\alpha,\beta}\Sigma_{u,b}(\alpha + \beta_u + \beta_b + \gamma_u \cdot \gamma_b - R_{u,b})^2 + \lambda(\Sigma_u\beta_u^2 + \Sigma_b\beta_b^2 + \Sigma_u||\gamma_u||_2^2 + \Sigma_b||\gamma_b||_2^2)$$

Note that the regularization coefficient λ can be different for β and γ. However, in the experiment below we notice that a different λ may not affect the performance a lot. Therefore, we set these two parameters as the same. We choose $\lambda = 5 \times 10^{-5}$ in this case. The dimension of vector γ_u and γ_b is set as 5.

6 Results

Table 3: RMSE of different models on test set

Model	RMSE
Baseline Mean Model	0.718
Linear Regression	0.728
Ridge Regression	0.729
Random Forest	0.641
MLP Neural Network	0.664
Naive Latent Factor	0.635
Latent Factor	0.628

6.1 Linear Regression

In linear regression, we have tried three different models with various combined features, such as average aroma rating of given beerId in training set, average appearance rating of given beerId in training set, average palate rating of given beerId in training set, average taste rating of given beerId in training set, the number of appearance of this userId in training set, the number of appearance of this beerId in training set, the average overall rating of this userId gave in training set, the average overall rating of this beerId was given in training set.

It turns out that the model with all four average ratings as features has the best performance on both validation set and test set. It is matching what we have done in Exploration Data Analysis in previous section, where we found that these four partial ratings have close relationship with overall rating. Features ”number of appearance of this userId in training set” and ”number of appearance of this beerId in training set” had no or very little impact on overall rating. In addition, when we added features ”average overall rating of this userId in training set” and ”average overall rating of this beerId in training set”, the performance of this model is much worse than previous model on validation set. However, we found that this model has very good performance on training set, compared with other linear regression model. It may be the case that these two features have made the model overfitting on training set, causing it have worse performance on validation set. Therefore, we gave up using these two features.

However, all these three linear regression models have performance worse than baseline model, which is using global average overall rating as prediction. They had such performance because of the skew of original dataset, where over 80% of reviews have overall ratings over 4 (rating from 1 to 5).

6.2 Ridge Regression

In addition to simple linear regression models, we would like to have a penalty term lambda to penalize the model complexity and optimize its cost function. Same combination of features of linear regression have been used in Ridge Regression. We used validation set to tune the best penalty term lambda in ridge regression model.

We had the same found in Ridge Regression as in Linear Regression that four partial ratings (aroma, appearance, palate and taste) have close relationship with overall rating. With using these four features in Ridge Regression model, it has the lowest RMSE compared with other two models. In addition, features "number of appearance of this userId in training set" and "number of appearance of this beerId in training set" had no or very little impact on overall rating, while features "average overall rating of this userId in training set" and "average overall rating of this beerId in training set" will make the model overfitting on training.

The same as linear regression, all these three models have performance worse than baseline model, because of the skew of original dataset.

6.3 Random Forest

The lowest RMSE we achieved with the random forest model we defined in Section 5.2 is 0.641 as shown in Table 3. It has a 10.7% improvement compared to the mean-rating model.

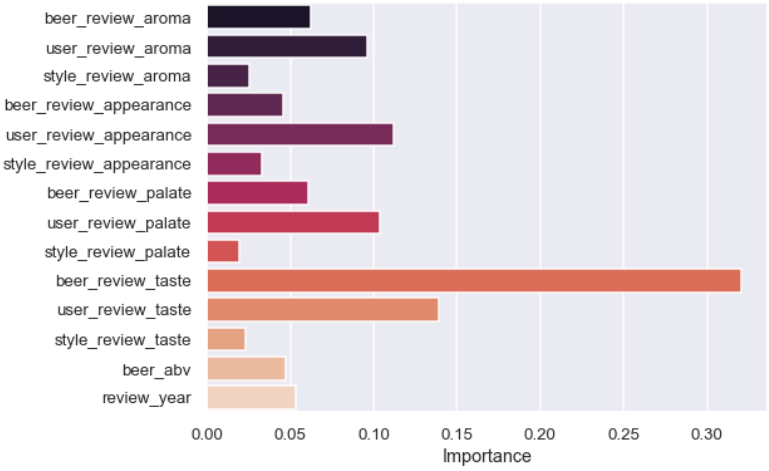


Figure 7: Importance of features in random forest

Here in Figure 7 shows the importance of features of our random forest model. According to this figure, three observations can be concluded: 1) The average rating for taste of a beer is the most important feature that helps to make prediction. This observation enlightens us that to further extract features of beer taste may improve the performance of our model. For instance, we can calculate the maximum, minimum, median of beer taste or calculate the percentage of each rating level. 2) Among all partial ratings, the average ratings for users are more importance than the average for beers and for styles, except for partial reviews of taste. This shows that in most cases, the user biases are more importance than beers'. Due to the style of beers is a rough granularity representation for beers, the average ratings for styles are somewhat positive correlated with those for beers. Therefore, the features of styles seem to be not so important. 3) Beer ABV and review years are not good features in this model. Compared to partial ratings, the importance of these two features are much lower.

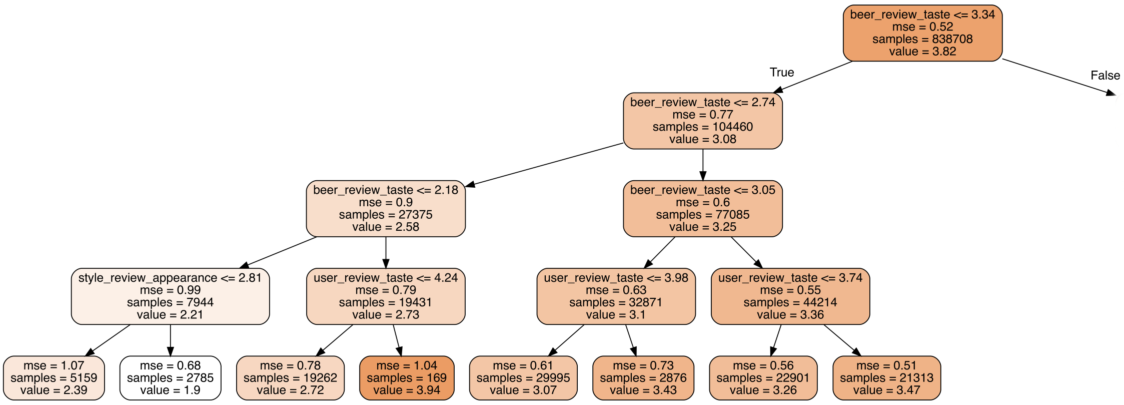


Figure 8: Part of one random decision tree from the random forest

Figure 8 is the left subtree of a random decision tree in the forest. The height of this tree is four and it mainly considers three features: beer_review_taste, user_review_taste and style_review_appearance. There are two observations in this figure: 1) It verifies that the average taste rating for beers is the most important feature again, since the majority of judgment is correlated with this feature. 2) It shows the output of parent node will influence the judgement condition of child node which is mentioned in Section 5.2. Due to such properties of random forest, it is able to consider non-linear relationship between features, which is helpful for prediction.

6.4 MLP Neural Network

The lowest RMSE we achieved with the MLP neural network is 0.664 as shown in Table 3. It has a 7.5% improvement compared to the mean-rating model. Figure 9 shows the training and validation losses during the gradient descent process. The results converge within 40 epochs and the training loss decreases monotonically. Although the validation loss fluctuates a lot, it still has a decreasing tendency revolving around the training loss. This shows the neural network is not overfitting.

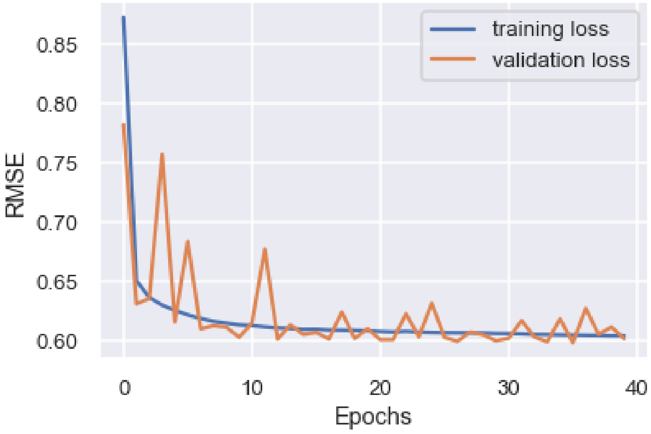
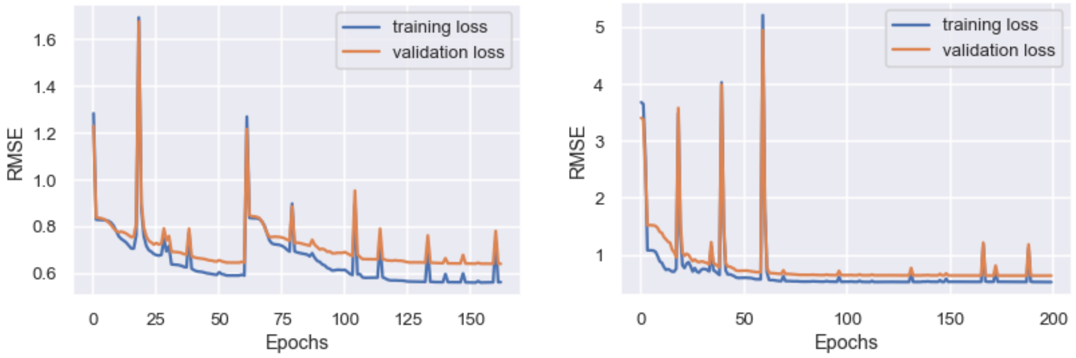


Figure 9: RMSE losses for MLP neural network

However, the performance of MLP neural network is a little bit worse than random forest model. There are three explanations for this: 1) The complexity of our MLP neural network is lower than the random forest model. 2) Neural networks are more adept at solving more complex problems, such as 2-dimensional image recognition. The beer rating prediction lacks rich features of different sources, which limits the performance of neural network. 3) We may miss important information which is important to MLP neural network during the feature extraction process. By extracting average partial ratings for beers, users and styles, we actually reduce the information with a rough granularity. This problem is solved in latent factor models, which have the best performance as shown below.

6.5 Latent Factor

The lowest RMSE we achieved with the naive latent factor model is 0.635 as shown in Table 3. It has a 11.6% improvement compared to the mean-rating model. With a better performance, the full latent factor model reaches an RMSE of 0.628, which is 12.5% lower than the baseline.



(a) Loss of naive latent factor model

(b) Loss of latent factor model

Figure 10: RMSE losses for latent factor models

Figure 10 shows the training and validation loss during the gradient descent procedure of two latent factor models. According to this figure, the convergence speed of the naive latent factor model is lower than the full model. The validation loss is higher than training loss which is normal.

Note that due to the large time consumption of iterative gradient descent, we only choose 200,000 records as the training data for this model. In spite of a lack of data, latent factor model still outperforms all other models. However, a weakness of it is the time complexity. Conclusively, we believe that this is a pretty good model for this problem and will reach even lower RMSE with the whole training dataset if given enough computation resources.

7 Conclusion

In this project we predict the overall ratings on the beer reviews dataset. We first take an detailed exploratory data analysis on the whole dataset and obtain several useful observations which enlighten us to build models afterwards. To solve this problem, we adopt several models including linear regression, ridge regression, random forest, MLP neural network and latent factor models. The results shows that the full latent factor model has the lowest RMSE of 0.628. For future work, we suggest to further improve the structure of latent factor models with the ideas in [5,9,11,12].

References

[1] Beer Advocate Website, https://www.beeradvocate.com/.

[2] Beer review product data, https://cseweb.ucsd.edu/~jmcauley/datasets.html#multi_aspect.

[3] Social Beer Advocate review product data, https://data.world/socialmediadata/beeradvocate.

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan. Personalized ranking metric embedding for next new poi recommendation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[6] T. K. Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[7] D. J. Julian McAuley, Jure Leskovec. Learning attitudes and attributes from multi-aspect reviews. *International Conference on Data Mining (ICDM)*, 2012.

[8] J. L. Julian McAuley. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. *WWW, 2013*, 2012.

[9] 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.

[10] J. McAuley, R. Pandey, and J. Leskovec. Inferring Networks of Substitutable and Complementary Products. *Knowledge Discovery and Data Mining*, 2015.

[11] A. Moreno, C. Ariza-Porras, P. Lago, C. L. Jiménez-Guarín, H. Castro, and M. Riveill. Hybrid model rating prediction with linked open data for recommender systems. In *Semantic Web Evaluation Challenge*, pages 193–198. Springer, 2014.

[12] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820. ACM, 2010.

[13] T. Shi and S. Horvath. Unsupervised learning with random forest predictors. *Journal of Computational and Graphical Statistics*, 15(1):118–138, 2006.