Using Regression Methods to Predict Alcohol Beer Rating

Junwen Huang
juh059@ucsd.edu

Shiyun Chen
shc176@ucsd.edu

Zhiyan Gu
zhg013@ucsd.edu

Kuangyi Yang
kuy006@ucsd.edu

ABSTRACT
In this paper, we analyze the dataset of alcohol beer reviews, and propose the models to make prediction of brewer's rating on the beer based on the information of the review.

Keywords
rating, beer, tf-idf, Ridge Regression, Gradient Boosted Regression Trees, Random Forest

1. INTRODUCTION
Our goal is to predict the overall ratings from beer reviews based on all the other information. Our baseline model is a linear regression model, which is one of the simplest supervised learning approaches to analyze relationships between features and response. Then we discuss some models more complicated than the ordinary linear model that could fit the non-normally distributed dataset better. Our final goal is to build a model with higher accuracy, beating the baseline model, using linear and non-linear regressions. Since our prediction task involves text, we are especially interested in generating features from review texts, such as tf-idf and some meta features about counting.

2. DATASET
2.1 Dataset Description
We use RateBeer reviews dataset from SNAP [5] [6] [7]. This dataset consists of beer reviews from rate beer. The data span a period of more than 10 years, including all 3 million reviews from April 2000 to November 2011.

This RateBeer reviews dataset consist of 2,924,127 reviews from 40,213 users. The number of beers is 110,419, each review includes ratings in terms of five aspects: appearance, aroma, palate, taste and overall impression. Reviews also include beerID, brewerID (ID of reviewer), ABV (alcohol by volume), the review text, and the time of the review.

To simplify the problem, we only investigate the beer reviews with alcohol beer. In datails, we only consider reviews with ABV existing and not equal to zero. Then, there are 82,614 reviews we taking into consider, and there are 2,992 different beers and 277 brewers in these reviews.

2.2 Basic Dataset Characteristics
The figure 1 is the distribution of overall ratings, according to this histogram graph, the range of overall ratings is from 0 to 20, all of which are integers, and we find that the ratings are accumulated around 14 and 15.

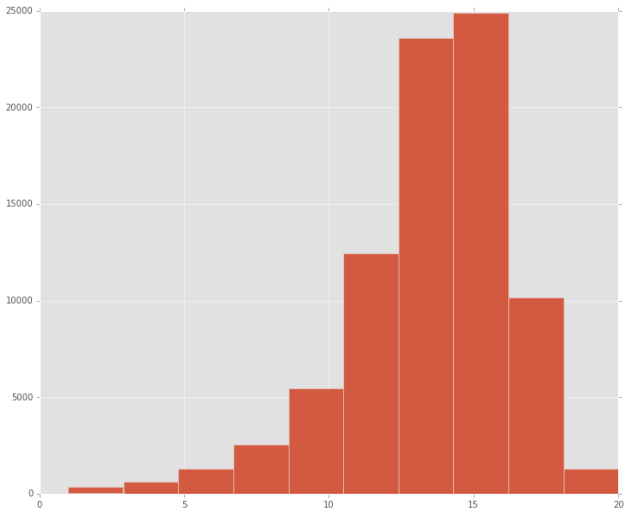

Figure 1: Overall Ratings

We are also interested in distributions of ABV, according to the figure 2, the range of ABV is from 0 to 25, all of which are integers, and in fact ABV are clustered around 5 to 6.

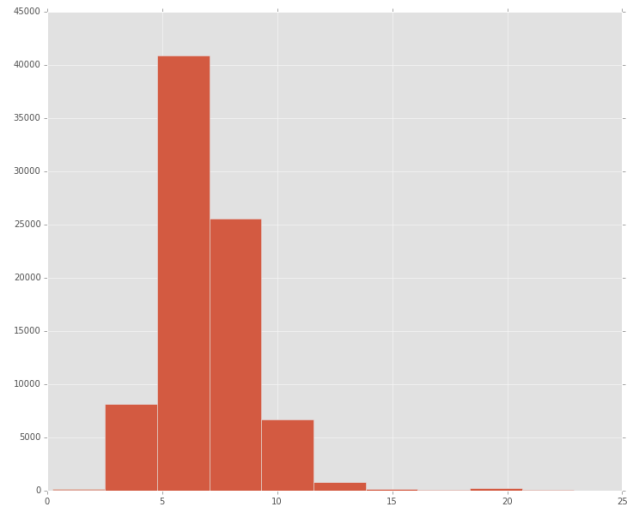

Figure 2: beer/ABV

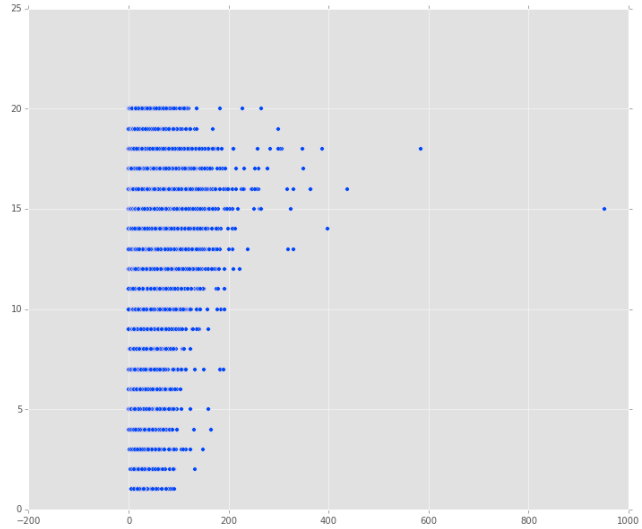

Figure 4: length of reviews versus overall ratings

Then we analyzed some properties of reviews and its relationship with overall ratings. Figure 3 shows that there is correlation between ABV and overall ratings; Similarly, figure 4 shows the correlation between the length of reviews after removing stop words and overall ratings. These figures give us some insights that it may be useful to add these as features in our predictive models.

Next we want to detect if there is any correlation between time and overall ratings, figure 5, figure 6 and figure 7 are box plots of ratings versus years, months and weekdays respectively. These three figures show that there are some differences in ratings between different years, months or weekdays.

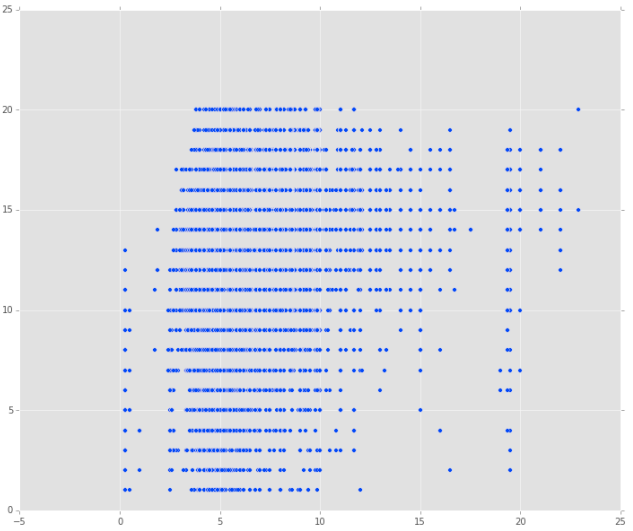

Figure 3: beer/ABV versus overall ratings

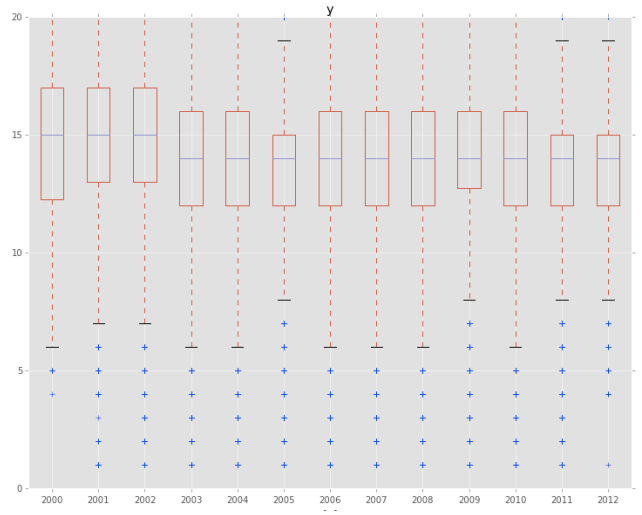

Figure 5: boxplot of years

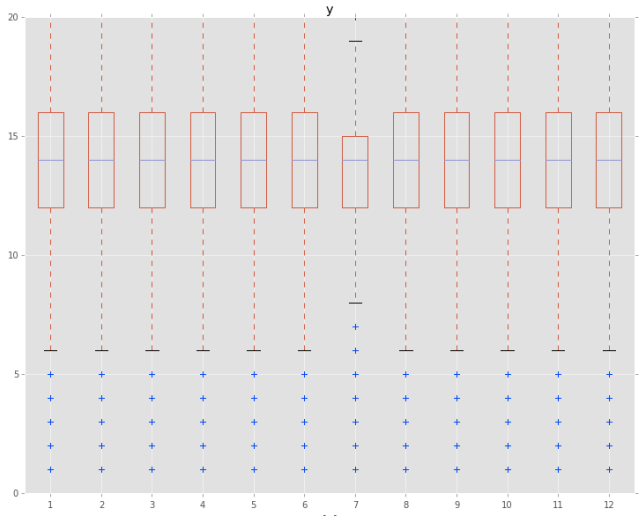

Figure 6: boxplot of months

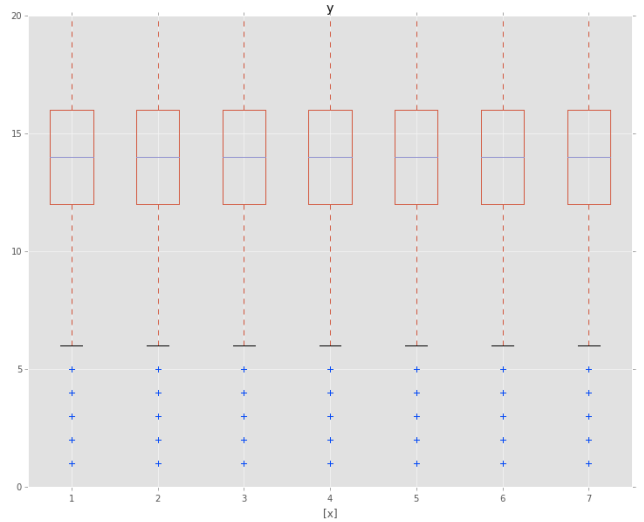

Figure 7: boxplot of weekdays

To verify our assumption, we also plot the scatter figure of time versus ratings in figure 8, which shows that there is a little correlation between timestamp and overall ratings.

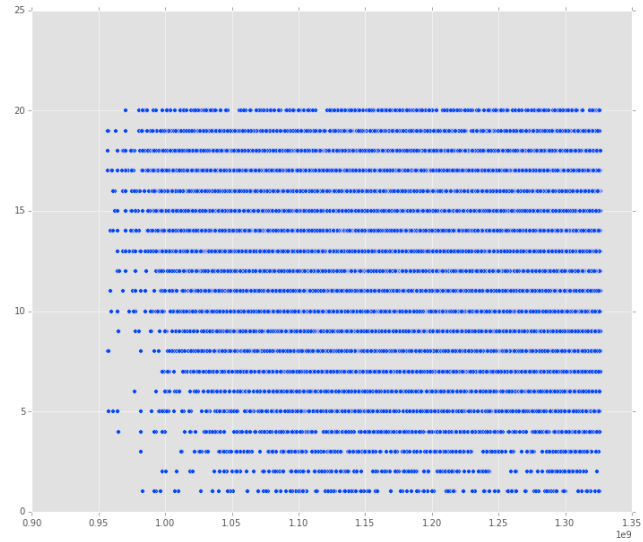

Figure 8: time versus overall ratings

Finally we generate an interesting plot about the review texts and overall ratings. We are especially interested in the tf-idf matrix (described in later section), and we want to visualize its relationship with overall ratings. Figure 9 is the result of our attempt: firstly we generate the tf-idf matrix then use SVD to reduce it to $n * 2$, then the x, y axises of the figure are the columns of this $n * 2$ matrix, and the z axis of the figure is the overall ratings.

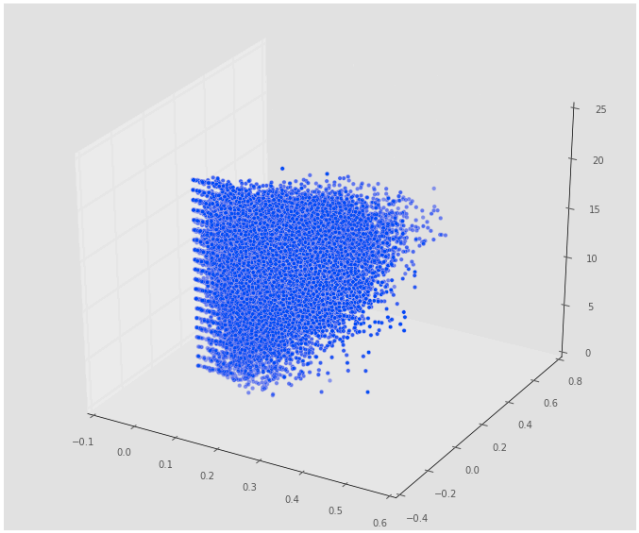

Figure 9: 2-dim projection of tf-idf versus overall ratings

3. PREDICTIVE TASK

Our task is to predict the overall rating of each beer based on its essential features and properties. That is to say, we will not take the review of aroma/ appearance/ palate/ taste into consideration, since they are opinions given by the reviewer and also highly corelated to the overall rating. Fitting a

regression model with these reviews as predictors will have good predictive accuracy on overall review, but it makes little sense. For other features in dataset, we will include some of them by model selection. And for the review text, we need to use some text mining technologies to extract important informations, like, the tf-idf matrix (may be truncated by SVD). Including all the features we select (described below) in the predictor, different kinds of regression model, such as Lasso, Ridge Regression and Random Forest, are applied to make comparisons.

3.1 Model Evaluation

After processing the data, we split the dataset randomly as 22% and 78%. We use the first part as test set, and then split the remaining part again as 22% for validation and 78% for train. To determine the hyperparameter of each model (model selection process), cross validation process is applied to compare and choose the best model when using each method.

The model performance is evaluated by Root Mean-Squared Error (RMSE) as prediction accuracy. All results returned by different models are judged and compared to baselines according to their respective RMSE.

3.2 Baseline Model

In order to evaluate our efforts, we would like to set up a simple and trivial baseline model. We could predict the new ratings based on the "history", then we could just use this trivial algorithm [1]:

1. For training set, compute each brewer's average rating, and the average rating over all the reviews.
2. In validation/test set, for each new review, check if the brewer occurs in the training set or not firstly. Then if the brewer occurs in the training set, we predict that the rating for this review is the average rating of this brewer. Otherwise, we predict the rating for this review is the average rating over all the reviews.

This is equivalent to applying the linear regression model regarding each brewer in training set as a category. We use Python package scikit − learn [8], and its functions sklearn .linear_model.LinearRegression and sklearn . feature_extraction . DictVectorizer here.

If we train the model on the training set and make prediction on validation set, this algorithm gives us the RMSE as 2.3964. And if we train the model on the training set and validation set ("real training set") and make prediction on test set, this algorithm gives us the RMSE as 2.3962. In fact, we also try to round the predictions since all true predictions are integers, but we get a worse result.

4. FEATURES AND MODELS

4.1 Features

We should make good use of the available data as much as variable. We find that the brewer ID and the review text are the most important features. We hope to extract the information from the review texts, to get the reviewers' opinion on the beers, and predict the ratings the brewers could give to the beers. After investigation, we figure out some useful features as follows.

Tf-idf, term frequency-inverse document frequency, is a common way of extracting the internal structure of the corpus, and we use it as a kind of feature. It's widely used in text mining. It is a statistics of vectoring the text in corpus as a numerical vector, each element of which represents the importance of the corresponding word (or n-gram) in the text compared with that in the complete corpus. To avoid extract computation process, we remove the English stop words in the beginning. Then we compute the tf-idf matrix of the unigrams.

The length of each review text, after removing the stop words, is another feature in our final model. Intuitively, the longer the review text is, the more information it would provide. As a result, we decide to include this feature into the regression model.

For each brewer, we compute the mean of the length of each review text, then plug in these numbers into the corresponding reviews. Because different brewers may have different customs about the lengths of the review texts. As a result, we should combine this feature with the length of each review text, in order to investigate more information.

For each brewer, we compute the standard deviation of the length of each review text, then plug in these numbers into the corresponding reviews. Besides the means mentioned above, we also want to have a look at the standard deviations. Thus, we extract this meta information and use it in the models, too.

The original dataset contains ABV, alcohol by volume, a measurment of beer characteristic. We believe that this information is useful for predicting the ratings the brewers would give to the beers. So we add this variable into our feature set.

The original dataset also contains time. It means the timestamp of each brewer give to the beer. From the month - overall and weekday - overall plotting, we think that there are at least some weak correlations between the time and overall rating. So we include this variable as a feature.

4.2 Models

4.2.1 Tf-idf and Ridge Regression

While selecting the features, we conclude that (unigram) tf-idf matrix generated from the review texts is very useful. Usually, tf-idf matrix is kind of a big sparse matrix. We have two strategies: one is directly treat it as the X in the regression and make prediction; the other one is using technologies such as SVD (preferred, because of high dimensions and large sample size) or PCA to reduce its dimensions, so that we could combine it with other features to do regression and make predictions. Previously in this report we use SVD to reduce the dimensions of tf-idf matrix and try to visualize its relationship with overall ratings. But while building the

[1]In fact, here we follow the similar thinking as shown in Assignment1 by professor.

model, we discover that using the whole matrix as an X is more powerful.

Because tf-idf matrix is very sparse, the ordinary least squares may be not efficient. We usually apply some regularization. Ridge regression applies L^2-norm constraint on the estimated coefficients:

$$\hat{\beta} = \underset{\beta}{argmin} \, ||X\beta - y||_2{}^2 + \alpha||\beta||_2{}^2$$

Here α is the complexity set by us. We use validation set to adjust the value of α. [2]

We use functions sklearn . feature_extraction . text . TfidfVectorizer and sklearn . linear_model . Ridge [8] here.

4.2.2 Features Except Tf-idf, Gradient Boosted Regression Trees and Random Forest

We build other models based on the features except tf-idf. Here, because the data tend to be non-linear, we want to apply some non-linear methods. In the end, we find that Gradient Boosted Regression Trees (GBRT) [4] and Random Forest (RF) [2] perform best.

Both GBRT and Random Forest usually use decision trees as their underlying base estimator. However, GBRT tries to fit a base learner in each iteration in order to get a more optimized residual after the iteration, while RF tries to generate lots of decision trees in the beginning and using bagging methods to choose best decision trees and ensemble them. Thus they use different non-linear approaches to optimize the goal, suitable on the non-normally distributed features we generate and use.

Furthermore, during fitting, we realize that both methods tend to be overfitting. That's why we split the whole training set into a smaller training set and a validation set. We use validation set to verify and set up the more appropriate parameters for both models. Moreover, the dataset is kind of large, so we build the model on the multi-cores laptops and use packages supporting building models in parallel to get the results. We end up with using Python library xgboost [3] for GBRT and and scikit −learn [8] function sklearn . ensemble. RandomForestRegressor for Random Forest here. By the way, we get an extra benefit that both algorithms are insensitive to outliers.

4.2.3 Ensemble

After building up the models in Ridge Regression, GBRT and Random Forest, we use the technology called ensemble to improve the accuracy and reduce the RMSE of the prediction. Ensemble, in this context, means "mixing-up" the based models in proper ways. Because different models may use different approaches to obtain the optimized goals, if we ensemble different models in proper ways, hopefully we could get a better result by taking advantages of different models and avoiding the incorrect predictions or outliers.

[2]In fact, here we follow the similar thinking (but not exactly the same) as shown in Homework4 by professor, and we add the step to choose better α to get some improvements.

We use the simplest ensemble way:

$$prediction_{ensemble} = \alpha_1 prediction_{Ridge} + \alpha_2 prediction_{GBRT} + \alpha_3 prediction_{RF}$$

$$s.t. \, \alpha_1 + \alpha_2 + \alpha_3 = 1$$

There are no standard way to determine the coefficients in the ensemble model. Here, we generate different combinations of α_1, α_2 and α_3, and use validation set to determine the optimized combination. Overfitting may be an issue here, but actually the final result on the test set shows our αs do not overfit in the end.

5. LITERATURES

Dataset RateBeer has been used in Text Based Rating Predictions from Beer and Wine Reviews [1]. The goal of this paper was to predict wine and beer ratings from text reviews (only), which is similar to our goal. However, the authors introduced another way to process this dataset. In general, they tried a number of different models, and eventually settled on linear methods and naive Bayes classification method, treating ratings as categorical variable instead of numerical variable, thus our models use different approaches compared with theirs.

The authors found that most reviews included four aspects: aroma, palate, taste, and appearance. They looked at the distribution of overall ratings, and some of the language properties of this dataset. And they discovered that there was a slight correlation between the length of reviews and overall ratings, with a similar but slightly smaller correlation between number of adjectives and overall ratings. Thus these relationships allowed them to make predictions about the ratings from text reviews.

The authors started by splitting the scores up into categories 1-20 inclusive. For prediction, they translated the continuous prediction to a classification based on these 20 categories. The baseline for their predictive task was a model that always predicted the most common label, i.e. the mean value of the data, which lead to a R^2 of zero.

In order to get a better model, the authors first did some prepossessing to clean up the review text by removing HTML tags and punctuation, etc., and used the number of words in each review as a feature. Then they stemmed the words, and classified the words into 17 categories: NOUN, VERB, ADV, etc. From the word stems, they generated a feature vector using the bag-of-words model with a limited dictionary of size 200,000. They also generated bigrams from the tokenized review corpus and select the 500 most frequent bigrams as additional features.

To construct the naive Bayes model, the authors measured the prior probability of each class and estimated the posterior distribution of words over each class using Laplace smoothing. Then they represented documents using the previous bag-of-words model, and made predictions based on the estimated parameters. They also tried to fit a Ridge linear model.

Both models had advantages and disadvantages. The naive

Bayes model slightly outperformed the baseline model. And when tuning the model, reducing the vocabulary size seemed to be the most successful improvement. The Ridge model did not significantly outperform the naive Bayes model, while stemming and bigrams did lead to improvements. It was good enough for such simple models to be able to explain about 40% of the variance among data.

6. CONCLUSIONS

We summarize the results in figure 10 and table 1, by noting down the final parameters and the RMSE on test set here. From the summary, we know that our final (ensemble) model does a significantly better job compared with the baseline benchmark.

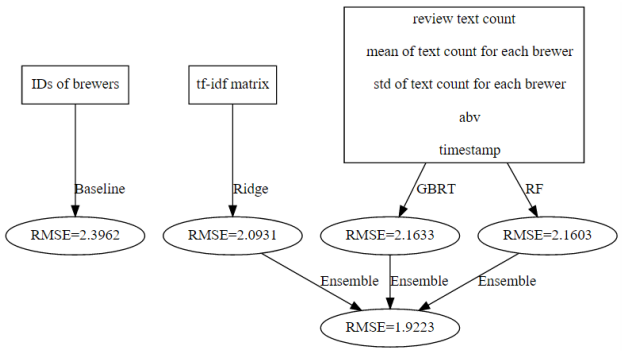

Figure 10: models results

By comparing models with each others, we find that the tf-idf matrix and Ridge Regression does slightly better than the models using GBRT and Random Forest. It shows that the structure inside the review texts used in Ridge Regression provides more information than the trivial meta features, ABV, and timestamp used in GBRT and Random Forest.

Moreover, GBRT and Random Forest perform similarly and better than baseline benchmark. This gives us insights that these two methods are useful for this dataset, and both indeed figure out some information based on the meta features, ABV and timestamp.

In the end, if we combine the results from Ridge, GBRT, and Random Forest models, we could get a much better result compared with any single simple model. We can conclude that three simple models investigate the problem using different approaches, and combining them indeed could reduce the errors effectively.

To sum up, for this dataset, we could use some easy-to-understand approaches to build a powerful model, to make the prediction of beer rating using the review text, review time, ABV, and the identity of brewer.

7. REFERENCES

[1] B. Braun and R. Timpe. Text based rating predictions from beer and wine reviews. 2015.

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

[3] T. Chen. Large-scale and distributed gradient boosting (gbdt, gbrt or gbm) library, on single node, hadoop yarn and more. https://github.com/dmlc/xgboost/, Feb. 2014.

[4] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[5] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[6] J. McAuley, J. Leskovec, and D. Jurafsky. Learning attitudes and attributes from multi-aspect reviews. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 1020–1025. IEEE, 2012.

[7] J. J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*, pages 897–908. International World Wide Web Conferences Steering Committee, 2013.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Table 1: results table

Feature	Model	Parameters	RMSE
IDs of brewers	OLS (Baseline)		2.3962
tf-idf matrix	Ridge	α=0.5	2.0931
review text count, mean of text count for each brewer, std of text count for each brewer, abv, timestamp	GRBT	{'num_round': 17, 'eta': 0.3, 'subsample':0.9, 'colsample_bytree':0.8, 'gamma':0.1}	2.1633
review text count, mean of text count for each brewer, std of text count for each brewer, abv, timestamp	RF	{'n_estimators':300, 'min_samples_leaf':20}	2.1603
tf-idf matrix, review text count, mean of text count for each brewer, std of text count for each brewer, abv, timestamp	Ridge, GRBT, RF, ensemble	$\alpha_{ensemble}$ = (0.6, 0.2, 0.2)	1.9223