Wine Ratings

Forrest Delavega fdelaveg@ucsd.edu A10096902
Ryan Almodovar ralmodov@ucsd.edu A11173777

Introduction

This paper describes how to construct a model based on linear regression that will attempt to predict the ratings a particular reviewer would give to a certain wine. Although we recently completed a similar task for assignment 1, we felt that with proper analysis of our dataset, we could design meaningful features for our model to use. Based on limited information of reviewers and wine, it is possible to train a model to learn some of the reviewers' traits and predict a numerical value for a rating. The rating system for wine is the same general idea that we have seen in class for beer data provided by BeerAdvocate and RateBeer, but rather than a 5 point rating scale from the beer data, it is a 100 point rating scale on wine. Surprisingly, much of the range is blank and scores are usually not as far separated as one might expect.

For this assignment, we wanted to see what factors are best to use when predicting a rating. Anyone who has heard anything about wine has heard that wine only gets better with age. We wanted to find out if there is indeed some correlation so we may use it in our model. Some other ideas that we fleshed out in our exploratory analysis included user experience, wine popularity, and review text length.

I. Dataset

The dataset we decided to use is from CellarTracker and can be found at http: // snap.stanford.edu/data/cellartracker.txt.gz. This dataset consists of wine reviews that span a period of more than 10 years, including over 2 million reviews up to October 2012. Reviews include product and user information, ratings, and a plaintext review. More specifically, each review has nine different fields: the name of the wine, rating, variant, unique user-id, username, time of review, age of wine, unique wine-id, and the review in text. Here is a review for reference:

wine/name: 2007 Domaine Berthelemot Bourgogne
review/points: 89
wine/variant: Pinot Noir
review/userId: 152917
review/userName: MFC
review/time: 1321660800
wine/year: 2007
wine/wineId: 712760
review/text: Great Village, see my previous notes

After scanning the dataset, we found that some of the nine fields were left blank, or had an N/A entries. We thus discarded any reviews which did not match the filter, and were left with a new filtered dataset consisting of 1,521,552 reviews - plenty enough to do an analysis and train a predictor. Now that we have a filtered version of our data, we can dive deeper and attempt to understand the underlying patterns.

A. Basic statistics

Looking at a histogram (Figure 1) of all ratings in our data, we can see that most of the 100 point scale is not used. Surprisingly, the average rating is 88.87 on such a large scale. In fact, a majority of the reviews are from 80-100. Since most of the ratings are close together, we feel that this fact will make finding certain features for our linear regression model difficult. As we will see later in the report, it's not a bad idea if we just predict the average of all ratings to use as a baseline.

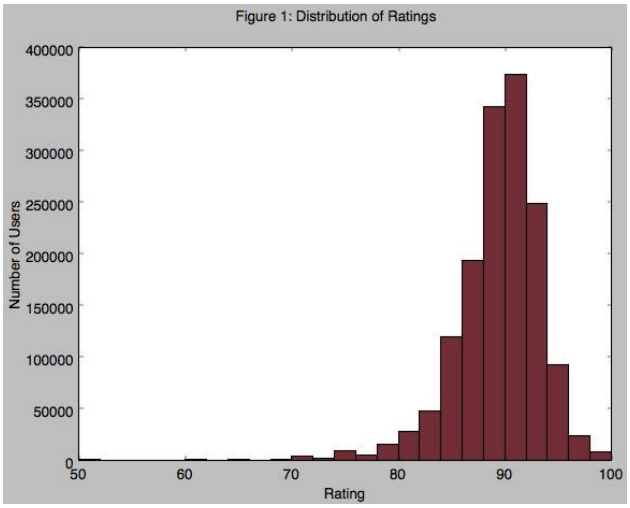
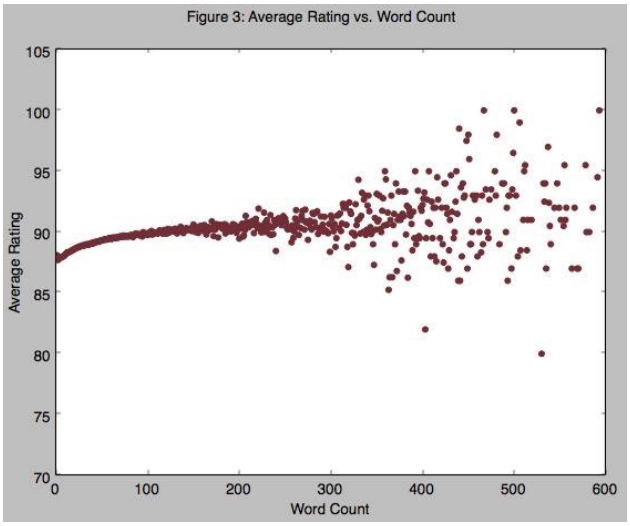


Figure 2

Number of reviews	2,025,995
Number of users	44,268
Number of wines	485,179
users with > 50 reviews	5,957
Max rating	100
Min rating	50
Average rating	88.87

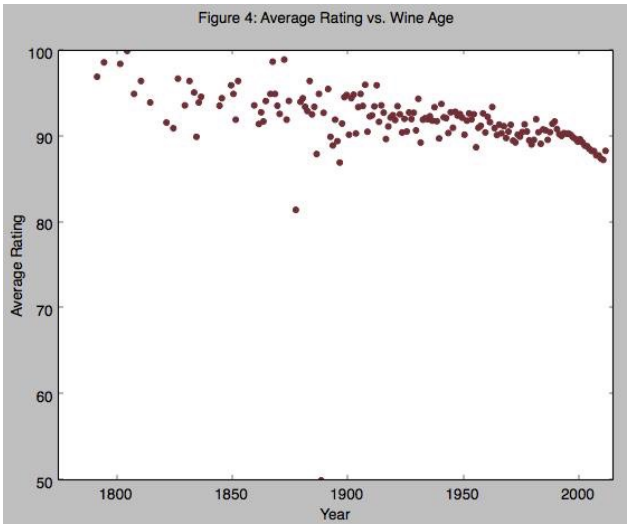
B. average rating vs. word count

In assignment 1, task 2, we used a linear regression model that incorporated word count as a feature, along with other features. This turned out to be a decent idea because we scored in the top 10 on the leaderboard. If the model worked for amazon products, why not try to see if there is some correlation with wine reviews? The y-axis represents average ratings while the x-axis is the word count of reviews. After inspecting the graph it appears that there is a linear function between average rating and word count up until a threshold of ~300 words. This divergence from what appears to be a gradual increase of rating makes sense. The lengthier a review, the more opinionated people tend to be. Not very many people will write a long review and say it was overall mediocre. People are more likely to either really enjoy or hate a wine the longer the review is. This could make a good feature in our model up until a threshold.



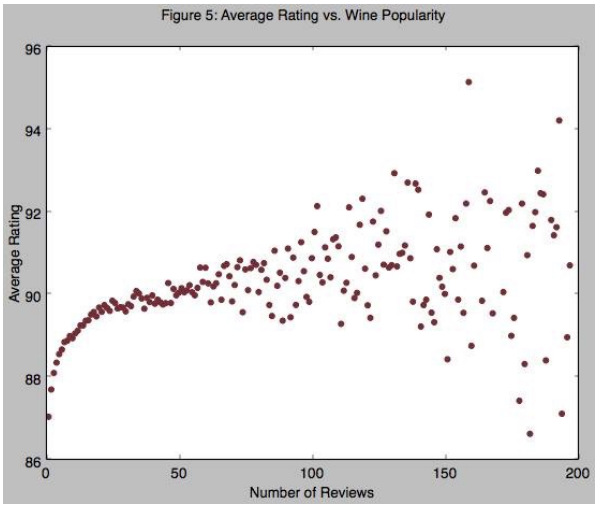
C. average rating vs. wine age

It is time to answer the age old question, does wine actually get better with age? According to the graph, older wines tend to have a higher rating than newly produced wine. One hypothesis could be the fact that because older wines tend to cost more money on average, the reviewer would be inclined to rate higher. After all, who would want to buy an expensively aged bottle of wine, only to confess that they wasted their money because it tasted below their standards? While not shown in the graph, it must be noted that there are many more reviews for new wines than old. This also might clear up why the graph becomes more scattered as the wine gets older. These findings sound reasonable to be used as a feature.



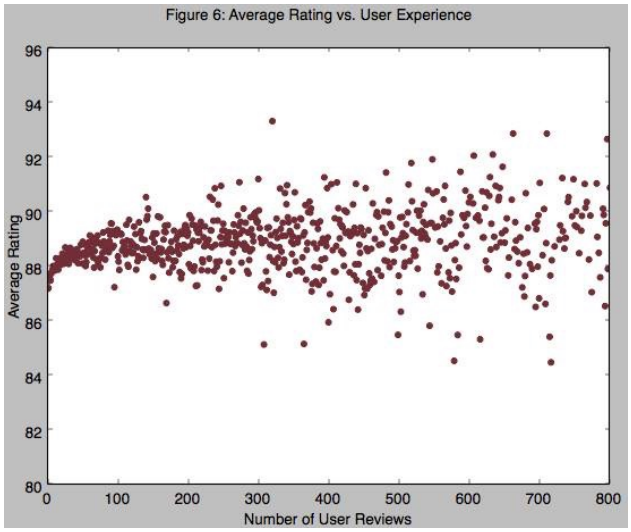
D. average rating vs. wine popularity

Wines with less reviews usually fall right near the average and gradually increase as more people review it. After a certain point, the points become scattered but still remain close to the overall average rating. This may or may not be helpful.



E. average rating vs. user experience

The only thing certain about this graph is that people who are just starting to review tend to always rate below the average. After a reviewer reaches a certain experience level, opinions deviate slightly and people develop unique tastes. Note how most of the dots are still around the global average of ~89 points.



II. Predictive task

After analyzing our filtered dataset, it became clear that most users of the website are wine enthusiasts with multiple reviews under their name. There are 485,179 different types of wines, yet only 44,268 reviewers. These people are more likely to be experienced when it comes to wine, and thus don't buy poor tasting wine. This hypothesis could explain Figure 1 where we can see most of the reviews clustered around very high ratings. For this reason we decided to use linear regression with a multitude of different features to find the best predictor of ratings. The task itself is not original at all, but it does make sense for the given data. Some of the equations differ depending on what features we want to use:

(1) $rating(reviewerId, wineId) = \alpha + \beta_1(wine/year)$

(2) $rating(reviewerId, wineId) = \alpha + \beta_1(wine/year) + \beta_2(wine\ popularity)$

(3) $rating(reviewerId, wineId) = \alpha + \beta_1(wine/year) + \beta_2(word\ count)$

(4) $rating(reviewerId, wineId) = \alpha + \beta_1(wine/year) + \beta_2(user\ experience)$

Linear regression is an appropriate model for this task because we are trying to predict a numerical value, and that is precisely what linear regression can help us achieve. Also, according to the exploratory analysis of our data and the assumption that the average rating may be a good predictor, it makes sense to have some baseline α that is usually around ~80 and can only increase depending on what features we have. The nature of distribution in the dataset justifies our use of linear regression.

We evaluated our linear models using mean square error,

$$MSE(f) = \frac{1}{N}\sum_{i=1}^{n}(X_i\Theta - y_i)^2$$

.

Our successful models we created minimized the MSE. We also needed a baseline to compare our features too. We used the overall average rating of all the reviews as our predicted rating for all the users. This baseline doesn't use any features of the user or product and just uses the global average rating. Surprisingly, the baseline had an MSE of 19.254. Almost all of the feature representations are equally good according to MSE. For a detailed version of the features we used, please refer to the end of the report (included code snippets).

For equation (1) and (2) there was no pre processing other than the filtering of our data. In equation (2) and (4), we had to do a run through of the entire dataset to count all the reviews for each wine-id and reviewer-id, respectively.

III. Literature

The dataset we are using comes from http://snap.stanford.edu/data/cellartracker.txt.gz. and was used in [1] and [2]. The models for these papers focused on producing a recommender system for a particular user, and revealed that amateurs were difficult to determine, whereas connoisseurs were more inclined to think alike and give similar ratings on wines. Some of the similar datasets that include a similar rating system are BeerAdvocate and RateBeer. One of the models that was used for these tasks was linear regression, which is what we implemented to predict the user's rating.

Similar to [2], we used the linear regression model with features such as the year of the wine and review text length against the average rating. In contrast to Ruogu's logistic regression model to predict the regression between the features year and number of user reviews, we chose to use our linear regression model between these two features, and the resulting MSE was the best of any of our models, with a value of 15.1720999847 on our data set. Some of the state-of-the-art methods used for predicting the ratings of wine include linear regression, Bayesian linear regression, and nonparametric regression. Although other works that were performed using these methods produced highly accurate results, our findings were similar to these other results.

IV. Results and Conclusion

All of our different models and the baseline results are shown below in Figure 7:

	MSE
Baseline	19.2542
Eq(1)	15.8245
Eq(2)	16.0468
Eq(3)	15.6102
Eq(4)	15.1720

Overall, our linear models provided slight improvements from the baseline rating average. We found that the year of the wine in combination with the user experience resulted in the best performance as a

predictor. Due to the fact that our models were relatively simple, there was little overfitting encountered when using our predictor on the test data set. Although there was a slight improvement over the baseline, the linear regression model did not fit the data as accurately as some of the other methods described in the referenced articles.

The linear regression predictor allows for simple interpretation of the data. The model parameter α represents the average review of a particular wine. The model parameter beta1 of our first linear regression model denotes the year of the wine. In our other three models, beta1 denotes the year of the wine, and the beta2's denote the wine popularity, word count, and user experience, respectively.

One reason why our model did not perform as well as expected is because one cannot judge a wine based on simple features alone such as the year and popularity alone. There were other specific features that we did not consider which were used in other referenced articles, such as the sentiment of the text in the review, which might have allowed us to determine with a greater degree if lengthy reviews were weighted to be more positive or negative from the norm.

V. References

[1] J. McAuley and J. Leskovec, From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. WWW, 2013.
[2] R. Liu, Wine Recommendation for CellarTracker. WWW, 2015.

Condensed version of our features

```
def feature(datum):
  feat = [1]
  return feat

def lin_reg_baseline():
  X = [feature(d) for d in train]
  Y = [d['review/points'] for d in train]
  theta,residuals,rank,s = np.linalg.lstsq(X, Y)
  print "Baseline MSE:", MSE(theta, train)

def lin_reg_yr():
  X = [[1, (2012-int(d['wine/year']))] for d in train]
  Y = [d['review/points'] for d in train]
  alpha, beta = np.linalg.lstsq(X,Y)[0]
  print "Year MSE:", MSE_lin_reg_yr(alpha, beta, train)

def lin_reg_yr_tlen():
  X = [[1, 2012-int(d['wine/year']), len(d['review/text'].split())] for d in train]
  Y = [d['review/points'] for d in train]
  alpha, beta1, beta2 = np.linalg.lstsq(X,Y)[0]
  print "Year/Text Length MSE:", MSE_lin_reg_yr_tlen(alpha, beta1, beta2, train)

def lin_reg_yr_wpop():
  X = [[1, 2012-wpop[d['wine/wineId']][0], wpop[d['wine/wineId']][1]] for d in train]
  Y = [wpop[d['wine/wineId']][2] for d in train]
  alpha, beta1, beta2 = np.linalg.lstsq(X,Y)[0]
  print "Year/Wine Popularity MSE:", MSE_lin_reg_yr_wpop(alpha, beta1, beta2, train)

def lin_reg_yr_userpop():
  X = [[1, 2012-userpop[d['review/userId']][0], userpop[d['review/userId']][1]] for d in train]
  Y = [userpop[d['review/userId']][2] for d in train]
  alpha, beta1, beta2 = np.linalg.lstsq(X,Y)[0]
  print "Year/User Experience MSE:", MSE_lin_reg_yr_userpop(alpha, beta1, beta2, train)
```