Simply Predicting Wine Review Ratings

CSE 190 Assignment 2

A10702215

I. INTRODUCTION

CellarTracker, as self-described, is "the world's largest collection of wine reviews, tasting notes, and personal stories from people who love wine." With the CellarTracker dataset, consisting of over two million reviews written by over 44,000 wine connoisseurs, it should be possible to use well known and simple data mining techniques to accurately predict the rating that a reviewer gives.

II. THE DATASET

The format of the dataset was first converted into JSON objects, for easier querying and manipulation. HTML entities were unescaped, but actual HTML tags in the review text were preserved.

The CellarTracker dataset consists of online reviews of various wine products. Several general statistics about the dataset are listed below:

Property	Quantity
Number of Reviews	2,025,995
Number of Reviews with Rating	1,569,655
Number of Reviews with Rating and Year of Wine	1,521,552
Number of Wines (by Name)	479,776
Number of Wines (by ID)	485,179
Average Review Length (in words)	38.600
Number of Users (by ID)	44,268
Number of Users (by Name)	42,496
Average Review (Points) Given	88.821
Maximum Review Given	100
Minimum Review Given	50

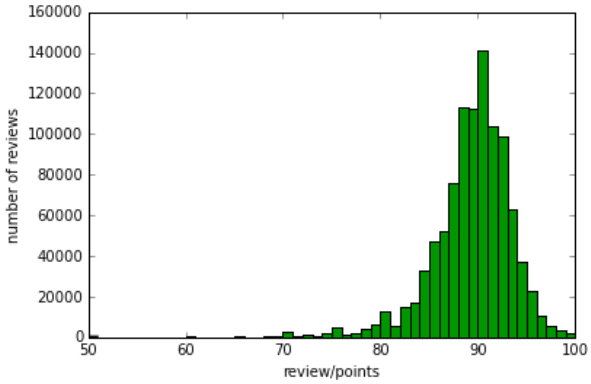
Each point in the dataset represents a review given by a user with nine values: wine/name, review/time, wine/variant, review/userId, review/username, review/points, wine/year, wine/wineId, and review/text.

Several of these fields are initially appear redundant. For example: review/userId and review/username should uniquely identify a user on the site; this is based under the assumption that websites do not tend to allow a user to register with a username that is already taken. Similarly, in this dataset wine/name and wine/wineId should identify a specific wine – the wine names are fairly unique and descriptive. Another area of redundancy that lies in the wine/name field is the year of the wine. If the wine has a year associated with it, then the year also exists in the name as well. This redundancy needs to be considered if the wine name is used as a textual feature to a model; the model must ensure not to double count influences by the wine year in both features.

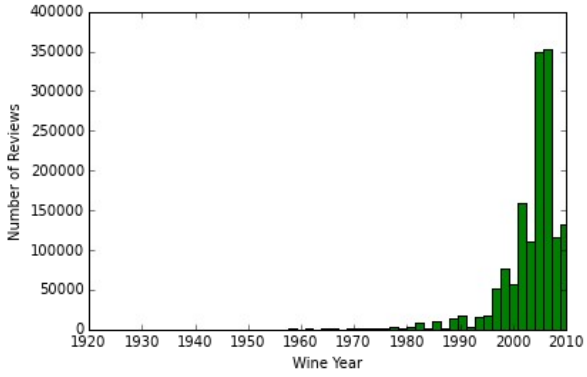
Several interesting aspects of the dataset can be observed by looking at the overview statistics. Unlike other traditional online review systems, CellarTracker allows a reviewer to not submit a score along with their review. Additionally, the number of wines by ID differs than the number of wines by name. Since the wine names are fairly descriptive and unique, the disparity between these two values should be considered when selecting features dependent on an item. Perhaps it is more wise to identify a wine by the name, rather than the ID, as this would merge related reviews together. Finally, our assumption that review/userId and review/userName is invalidated by

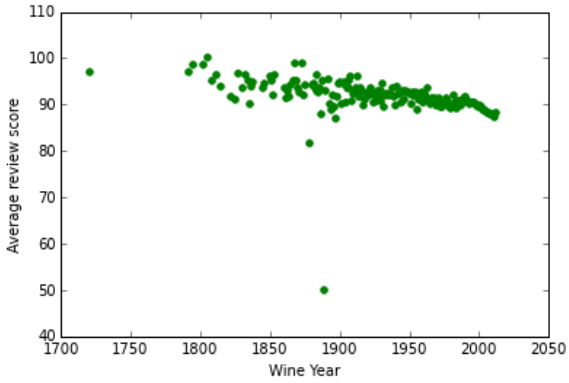
the difference the number of users by id and the number of users by name. This difference is due to the username "Anonymous" – CellarTracker allows users to hide their username from the public.

The value that varies the most within the dataset is review/points, which is a real number in the range 0 and 100. About 77.475% of the dataset had review/points; the remainder of the dataset did not have an associated review. The distribution of review/points is fairly normal, with the median review score of 89 and an average of 88.82077.

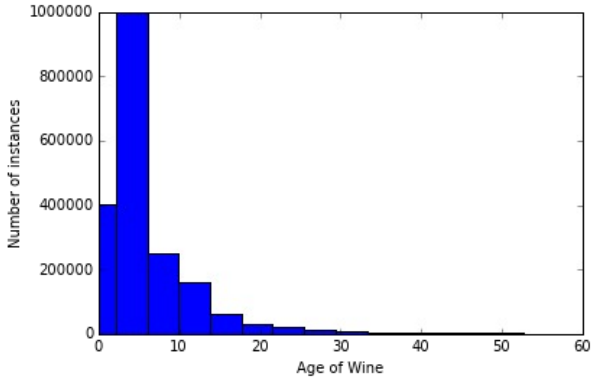


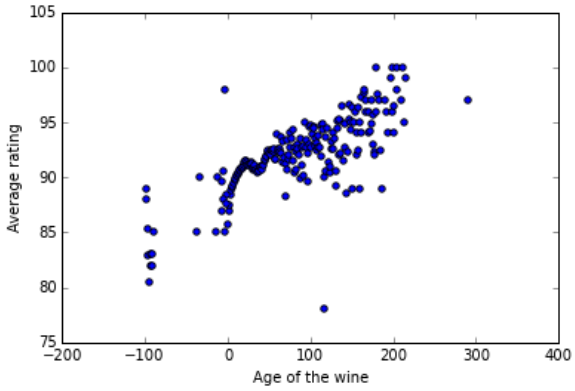
Only 75.101% of the dataset had wines with a valid year and a review score. The plots below show the relationship between the year of the wine and how it was reviewed.





However, because the reviews were made over a period of multiple years, the year of the wine may be less predictive than the age of the wine. The age of the wine is calculated as the difference between the time in which the wine is reviewed and the year of the wine. Common intuition says that aged wines generally tend to be more highly rated than their younger counterparts. Below, we examine how the age of the wine compares with its rating.

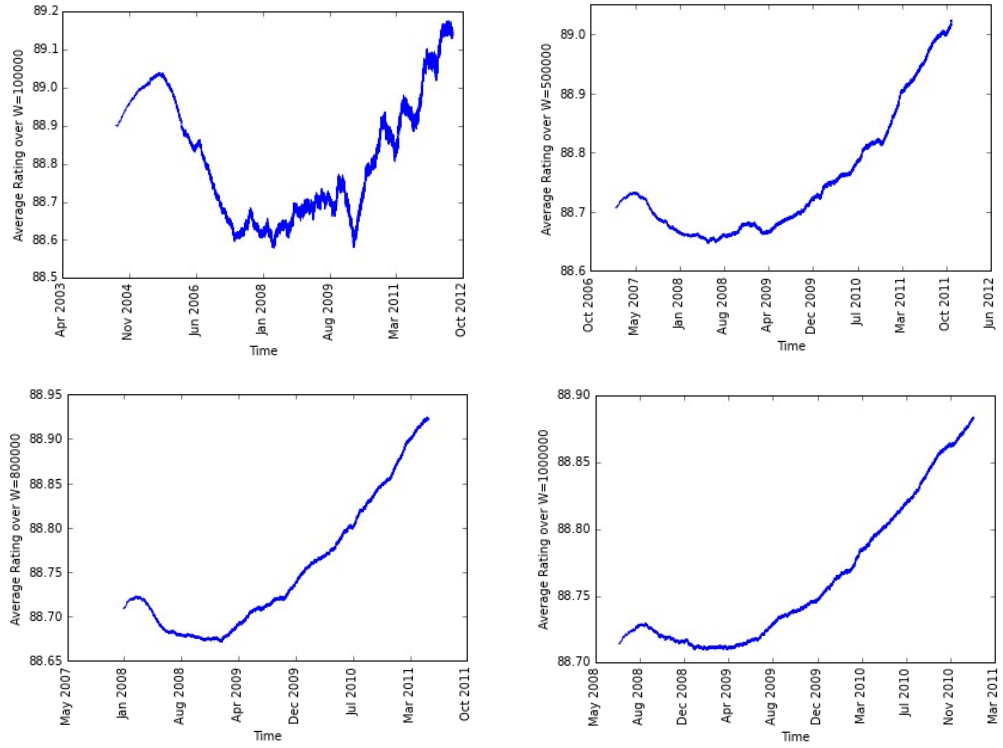




For some reason, there were entries in the dataset where the age of the wine was negative. That is, the time the review was submitted for the wine occurred before the wine actually was available for consumption. A similar error also is present the 'review/time' field; there were 42 data points that had a negative time. Upon further examination, these anomalies are due to incorrect data reported by CellarTracker itself.

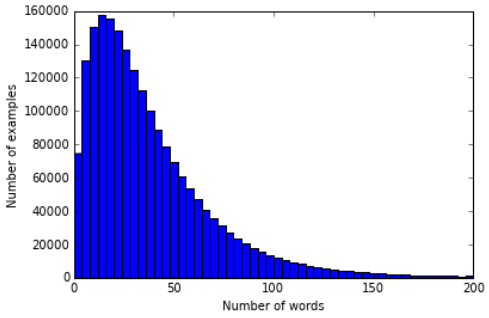
The median age of wine that was reviewed was 4 years. As expected, there does exist a correlation between the age of a wine and its average rating, although it is not perfectly linear.

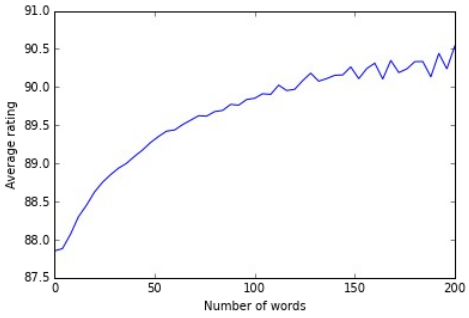
To see how average review ratings changed over time, we averaged the review points over a variable window size. Reviews with invalid times or negative times were excluded from this analysis.



In general, average review rating increased *slightly* over time (Within 1 point). This may be indicative of a “higher review rating” trend; however, the extremely small difference over time could be attributed to merely more users joining the website and rating wines with a score closer to the median.

Wine reviews given on CellarTracker are generally short and to the point. The average review length is around 38 words (including common English stopwords). Below, we analyze the relationship between the length of the review and the average rating given; our intuition is that longer reviews indicate a stronger opinion, and that not many reviewers would be inclined to write a long-winded, negative review.

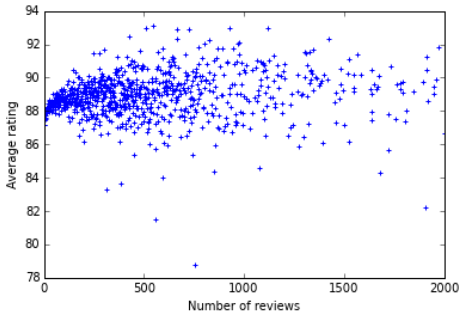


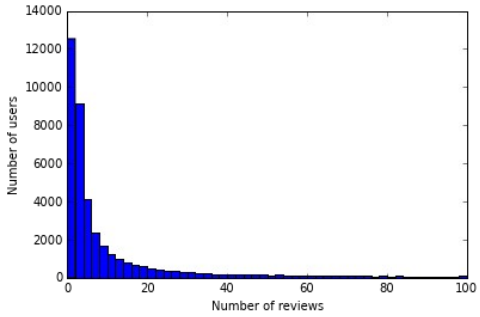


Our intuition has merit – the score of a wine was generally higher when the review length was longer.

Noting that 3.5% of our dataset are from "Anonymous" users, we looked at the average rating given by anonymous users compared to normal users. Perhaps the fact that the user chose to remain anonymous could influence the review that he gives? This turns out to be irrelevant – the average review of anonymous users (88.74027) and normal users (88.82466) could merely be due to the large difference in the amount of users in each set.

The number of reviews that a user makes can potentially give some indication to how likely he rates a product. The intuition here is that a new user might be hesitant to give his first reviews high ratings (since he has not experienced other types of wines). However, this turns out not to be too helpful, as the vast number of users have fewer than 20 reviews. Because there are fewer people who have reviewed more than 100 items, the data becomes very scattered.





III. Predictive Task and Model

We will attempt to predict review/points given the other properties of a review. To do this, we will be using linear regression, which models a predictor:

$$y = X\theta$$

where X is a matrix of features that we selected, and θ are their corresponding coefficients.

The primary task when utilizing a linear regression model is to determine a good feature set that is compatible with our linear assumption. Ideally, the features themselves should be independent from each other to prevent double counting; additionally, they should also ideally be explanatory of the value we would like to predict.

To evaluate our model, we will attempt to minimize the Mean Squared Error (MSE) with respect to a baseline which will be defined as one that predicts the average rating across the entire training set. Choosing MSE as the error function is convenient, as there exists a closed form solution to minimize:

$$\text{MSE} = \frac{1}{N}\|y - X\theta\|_2^2$$

However, to prevent overfitting, we will also introduce a regularization parameter λ to penalize complex models. Our objective function to minimize now becomes:

$$\text{argmin}_\theta \frac{1}{N}\|y - X\theta\|_2^2 + \lambda\|\theta\|_2^2$$

The MSE by itself is suitable for comparing the effectiveness of different models against the baseline, which admittedly is quite naive. Therefore, in order to evaluate the model holistically, we also measure the R^2 coefficient of determination, which is defined as:

$$R^2 = 1 - FVU(f) = 1 - \frac{MSE(f)}{Var(y)}$$

The methodology used in experimentation was as follows:
(1) Filter out all of the reviews without a valid "review/points" field. Predicting whether a review *would* have an associated review is a different classification task that we are not considering. Additionally, filter out the anomalies mentioned in Section II.
(2) Shuffle the data using an RNG initialized to a constant seed, for reproducibility.
(3) Select a random 20% of the data to act as the test set ("held out data"), which will only be used to evaluate the MSE and Training Error.
(4) We will use the remaining 80% of the data as our training set in a k=5 k-fold cross validation evaluation (80% training and 20% validation) and report the MSE and R^2 coefficient.

The features that were considered and used were:

Feature	**Description**
LRW	Length, in words, of the review.
LRW_BIN_1	Length, in words, binned by reviews less than 38 words and greater than 38 words. (e.g. a 20 word review would be encoded as the vector [20, 0]; a 90 word review would be encoded as [0, 90])
LRW_BIN_2	Length binned at < 38 words, < 100 words, and other.
AGE	Age of the wine in years. If the wine does not have an associated age, use the average wine age.
RTIME	Unix timestamp of review, if non-negative.
EXP	The number of a reviews that a given user makes.
EXP_BIN_1	Number of reviews that a user makes, binned into < 10, < 40, and other.
EXP_BIN_2	Number of reviews that a user makes, binned into < 5, < 10, < 15, < 40 , and other.
USER_AVG	The average number of points that the a reviewer gives. If the reviewer is not in the training set, then this feature is zeroed. If a user has fewer than 4 reviews, then he is not considered.
ITEM_AVG	The average rating that a particular item receives. Items are uniquely identified by their *name* instead of their *ID*. If the item is not in the training set, then this feature is zeroed. If an item has fewer than 10 reviews, then this feature is not considered.

Several variables, such as the number of reviews that a user makes, had a very skewed, long-tail distribution. Training examples became scarce as the number of reviews made by a user increased. In linear regression, because every training example affects the fit of the model, the binning technique was employed in order to prevent erratic outliers from sparse data affecting the overall fit of the variable.

IV. Results

Using linear regression, several features did help bring down the MSE, and increased the fraction of variance explained by 5% from the baseline model. The table below details the various results using the features (and a combination of several features) listed in Section III.

Name	**MSE**	**R^2**
Baseline	17.875605356182724	0
LRW	17.578135764555501	0.016636722787311653
LRW_BIN_1	17.577505184387526	0.016671998961579004
LRW_BIN_2	17.530450523095549	0.019304350113941493
AGE	17.196114984479106	0.03800788587826709
AGE + LRW_BIN_2	16.941937521674635	0.05222718570419094
RTIME	17.857910465709654	0.0009854517598743007
EXP_BIN_1	17.760086562053587	0.006457956681752197
AGE + LRW_BIN_2 + EXP_BIN_1 + RTIME	16.854983943486229	0.05709157842239587
AGE + LRW_BIN_2 + EXP_BIN_1 + RTIME	16.844491528643633	0.05767854939495454
USER_AVG	14.440621839548626	0.1921568130244361
USER_AVG + AGE + LRW_BIN_2 + EXP_BIN_1 + RTIME	14.015712837644241	0.2159272466032005
USER_AVG + ITEM_AVG + AGE + LRW_BIN_2 + EXP_BIN_1 + RTIME	13.666496156348353	0.23546326935192152

Our best selection of features resulted in an MSE of 13.667, compared to the baseline MSE of 17.876.

Since our model was simple and our features independent from each other, we avoided overfitting the data. Using the k-fold cross validation technique allowed us to determine the best regularization parameter for each feature set. There other hyperparameters, such as the threshold for discard in USER_AVG and ITEM_AVG, that could have been optimized more, but this would risk overfitting. Thresholds were instead chosen based on intuition and observance over its distribution.

An alternative model to consider would be the standard latent-factor model used in recommender systems (especially those dealing with collaborative filtering [3]):

$$rec(u, i) = \alpha + \beta_u + \beta_i$$

This model has the advantage of also being linear but also allowing usus to take into account the biases for individual users and items. This is different than the feature USER_AVG that we use in our regression model. In our model, the USER_AVG feature is weighted based on how users of the entire dataset (“the community”) behave. The latent-factor model used in recommender systems takes into account *individual* user biases while we take into account the how a *user in a community* tends to rate (the coefficient for USER_AVG remains the same across all

users). Similarly, ITEM_AVG measures how wines in CellarTracker generally affect the score, instead of taking individual item biases into account.

Despite not considering *individual* user and item biases, USER_AVG and ITEM_AVG were the two features that decreased the MSE significantly. A possible explanation for their efficacy is that users on the CellarTracker website generally stayed relatively consistent with their own reviews; item's that are reviewed favorably also are generally favored by the community. Because of the nature of the niche community of wine enthusiasts and the specificity of the product under review, USER_AVG and ITEM_AVG as simple linear features are effective. Contrast this with a more general, accessible review site such as Amazon product reviews – there are many more *types* of items. This means that a user that purchases many various types of items may not have a predictive USER_AVG (for example, the user might highly rate electronics, but tend to rate books lowly). CellarTracker deals primarily with wine, and its users are primarily wine enthusiasts. By discounting items and users by a threshold in ITEM_AVG and USER_AVG, we attempt to eliminate newbies or esoteric wine from muddling with the overall fit of ITEM_AVG and USER_AVG.

Other than USER_AVG and ITEM_AVG, the next helpful feature was AGE. This is compatible with our previous assumption that an aged wine is generally valued more, and thus rated highly. The other features were helpful in reducing the MSE, so they were left in the model.

V. Literature and Related Works

The CellarTracker dataset was used by J. McAuley and J. Leskovec [1] to examine how user experience and expertise can be effectively used to tailor recommendations to the user. Their latent factor model heavily utilized the temporal aspect of the review data to account for a reviewer's change in taste over time, as they acquire more experience. Our model tried to capture some notion of experience (in the form of the feature EXP), but without the temporal factor.

Narenda Gupta et. al [2] performed a similar regression type task in predicting star ratings given to restaurant reviews. While they used common textual features such unigrams and bigrams in a bag-of-words model, an interesting feature that they employed were *word chunks*, which are phrases conforming to a well-defined grammatical pattern. The only somewhat textual feature that we chose to use was the word length of a review. It is highly likely that incorporating more actual textual features into our model would increase its efficacy, and this would be the logical next step in improving our model.

VI. References

[1] J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. WWW, 2013.

[2] Gupta, Narendra, Giuseppe Di Fabbrizio, and Patrick Haffner. "Capturing the stars: predicting ratings for service and product reviews." *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search*. Association for Computational Linguistics, 2010.

[3] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." *Computer* 8 (2009): 30-37.