Predicting Product Ratings From Review Text

Michael Tran
UCSD CSE
mat018@ucsd.edu

ABSTRACT
When users purchase items, they leave feedback in the form of ratings or text reviews. It would be helpful to be able to predict product ratings from features extracted from review text to discover potential relationships between the two. In this assignment, we extract features from text reviews and then use regression and topic models to predict users' product ratings.

Keywords
sentiment analysis, text mining, regression, recommendation

1. INTRODUCTION
Product reviews and ratings are usually the most popular forms of user feedback that recommender systems can use to learn what users like to purchase. It may be valuable to find relationships between user's product ratings and their reviews because some users may omit one or the other.

In this assignment, we tried Latent Dirichlet models and mostly regression models to try to model product ratings from review text. The best performance was found with using ridge regression models for text mining reviews for features to predict product ratings.

2. DATA SET
In this assignment, we use an Amazon product review data data set, specifically for the categories Grocery and Gourmet Food.[1] Each review consists of the user ID, item ID, helpfulness votes, review text, overall rating, summary, and the review times.

The data set consists of 1,304,394 reviews spanning May 1996 to July 2014. The 1,304,394 reviews are split into training, validation, and test sets consisting of 834,812; 208,703; and 260,879 reviews respectively.

The training set consists of 545,921 users having purchased 137,370 unique items and rating 834,812 times. The mean rating of the training set is 4.256. Figure 1 reveals that most users rate very highly with a rating of 5 out of 5. Over half of the rating distribution is contained in the rating value of 4-5. So, we should expect review texts to reflect this high frequency of positive ratings.

Most items are only purchased once, as revealed by figure 2. The mean item purchase count is 6.077 while the minimum item purchase count is 1 and the maximum is 4042.

Most reviews are not very lengthy as shown by figure 3. The mean review length is about 30.753 words after applying the transformations outlined in the later Features section to the reviews. The maximum review length is 1763 words.

Figure 4 compares review length to product rating, and there does not seem to be a clear correlation between review length and ratings given by users, as both positive and negative reviews may be lengthy. So, there may not be much progress made in adding review length as a feature.

Table 3 shows the topics generated by running the Latent Dirichlet Allocation model on the training corpus with 10 topics generated. Listed with each topic is the 10 words with the highest probability of being in the topic. It is interesting that the first column's topic clusters generally positive sentiment words. This motivated the potential use of LDA in a regression model.

3. PREDICTIVE TASK
Our predictive task will be a regression task of directly predicting the numerical value of a user's product rating given their review text. We used linear regression with least squares, ridge regression and Latent Dirichlet Allocation as models and least squares and LDA end up not performing as well as ridge regression.

The model will be evaluated by calculating the mean squared error:

$$MSE = \frac{1}{n}\sum_{i=1}^{\infty}(\hat{y}_i - y_i)^2$$

where $\hat{y}$ is the predicted rating and y is the actual rating label.

3.1 Relevant Baselines
The models will be compared to two relevant baselines.

One baseline will be using a global mean α of all the ratings in calculating the mean squared error:

$$MSE = \frac{1}{n}\sum_{i=1}^{\infty}(\alpha - y_i)^2$$

This baseline achieves a mean squared error of 1.5675 on the test set using the global mean of $\alpha = 4.2563$.

The second baseline is the latent factor model, which tries to calculate biases for both users and items. Although this

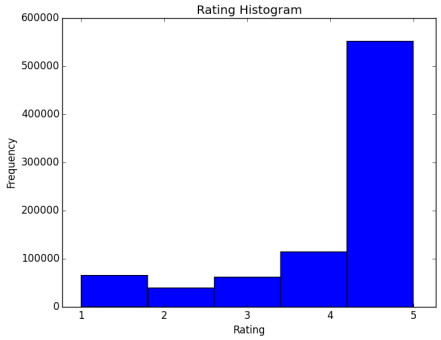

Figure 1: Distribution of user ratings

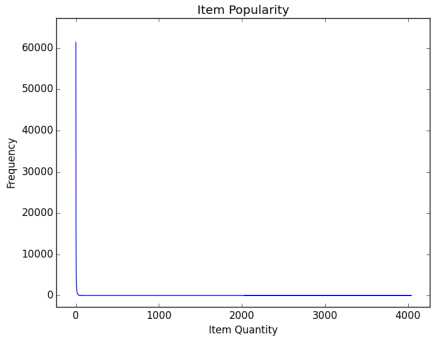

Figure 2: How popular each item is

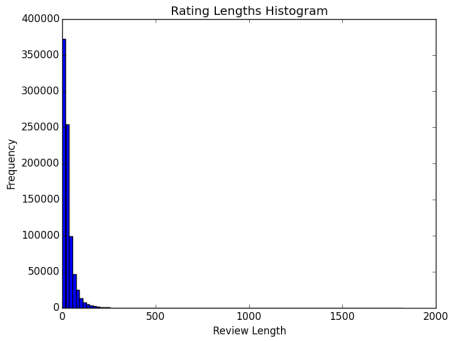

Figure 3: How long reviews are

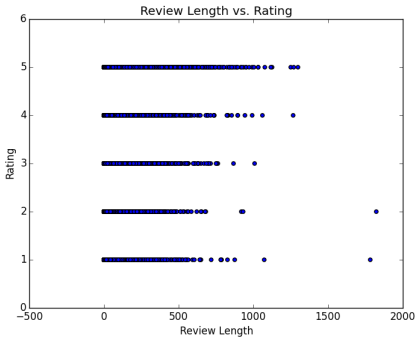

Figure 4: Length of review vs. rating

latent factor model does not take review length into consideration, this model's performance is used as sort of a worst-case baseline.

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

Here, β_u is a user bias and β_i is an item bias. This baseline achieves a mean squared error of 0.4658 on the training set, 1.4461 on the validation set, and 1.45922 on the test set.

Assessing the validity of our models will be achieved by comparing to the mean squared errors of the baselines.

3.2 Features

In constructing features for user reviews, review length was not included as the exploratory analysis of the training data did not appear to reveal any correlation between review length and product rating. Instead, features about the words in the text reviews were used.

First, all the reviews were stripped of punctuation and converted to all lowercase letters. Including punctuation and capitalization as features only seems to be intuitively useful for predicting helpfulness votes, not product ratings.

Then, reviews had all stopwords removed for words such as "the" or "and" do not seem to add any value in unigram sentiment analysis. In addition to removing stopwords, any words that were numbers such as 12 or 1 were removed.

Afterwards, the most popular 1000 words of the training corpus were found after cleaning all of the text reviews using the method above. The feature vector was then a 1000-dimension vector consisting of word counts for a text review for those 1000 most popular words. This feature vector was then used for the LDA model. However, for the regression models, tf-idf counts were used rather than just word counts.

$$tfidf(t,d,D) = tf(t,d) \times idf(t,D)$$

where tf is the term frequency of a term in a document and idf is the inverse document frequency of a term across all the documents in the training corpus

$tf(t,d) =$ number of times term t appears in document d

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

where D is the set of all documents in the training corpus, and N is the total number of documents.

4. PREDICTION MODELS

4.1 Linear Regression

The first model tried was the linear regression model:

$$X\theta = Y$$

of which X is a sparse array of tf-idf counts of all the training documents, theta consists of the weights assigned to each of the words in the feature vector, and Y is the array of product ratings associated with each review.

Results of using this simple linear regression using least-squares was poor. The average mean squared error across

both validation and test sets was around 60. This is likely due to most feature vectors being very sparse because the vast majority of reviews are not very long, which may have led to severe overfitting to training data.

4.2 Ridge Regression

Ridge regression minimizes:

$$\min_{\theta} ||X\theta - Y||_2^2 + \alpha||\theta||_2^2$$

and imposes a penalty on the weighted values in theta.[2]

Doing this helps avoid the shortcomings of the previously mentioned linear regression model using least squares. When trying different values of α to use as a penalty, the mean squared error was nearly uniform for all values between 0.05 and 1.0, so $\alpha = 1.0$ was used.

Ridge regression resulted in a mean squared error of 0.93148 on the validation set and 0.92915 on the test set. These results improved greatly on the mean baseline as well as the previous least-squares linear regression model.

4.3 Latent Dirichlet Allocation

Latent Dirichlet Allocation generates probabilities of a document belonging to a certain topic:

$p(d|\theta, \phi, z) = \prod_{j=1}^{n} \theta_{z_d,j} \phi_{z_{d,j}, w_{d,j}}$

where d is a document, n is the length of document d, θ is the probability of the word's topic and ϕ is the probability of observing the word in that topic.

LDA generates a specified number of topics and then assigns probabilities of each document being in a certain topic as well as words being in certain topics. So, each document can be assigned some weights on the probabilities of the topics they are assigned to. Using those weights as a feature vector, ridge regression was attempted using these LDA probabilities, but the model did not scale well and reducing the set to train the model did not yield great performance.

5. RESULTS

Using the simple linear regression model with least squares performed poorly, and the ridge regression model performed much more effectively. The positively weighted theta values correspond to positive sentiment words while negatively weighted theta values correspond to negative sentiment words. The most highly weighted words are listed in tables 2 and 3.

Feature representations including length of review or considering amount of punctuation or capitalization would not work well because there is not a clear correlation to product rating. For example, both negative and positive reviews can be lengthy.

The simple linear regression model likely overfitted, of which the ridge regression remedied with the penalty. The LDA model was not able to scale well with the large data set and reducing the size of the training data set to train this model did not yield greater performance.

Table 1: Highest Positively Weighted Words

great	3.22910219521
best	2.51782695081
love	2.22746575795
delicious	1.94017899863
good	1.7211863438
perfect	1.72092422707
excellent	1.59042414934
wonderful	1.49705286709
amazing	1.49138111043
favorite	1.46160692651
loved	1.2491280511
awesome	1.24457851522
pleased	1.22772334139
exactly	1.22411807207
fantastic	1.21032448656
find	1.12865412375
nice	1.1143162813
loves	1.11216089251
highly	1.10836603646
hooked	1.08778548598

5.1 Related Literature

In related literature, the latent factor model for the second baseline was discussed by Yehuda Koren and Robert Bell and how it captures a user's interests in an item's characteristics, and this model was optimized using gradient descent:[4]

$$r_{u,i} = +b_i + b_u + q_i^T p_u$$

where b_i and b_u were the item and user biases and the last term captured the compatibility between a user and an item, noted by Koren.

This dataset came from an Amazon product data set compiled by Julian McAuley.[1] A later related study by Julian McAuley and Jure Leskovec resulted in the creation of the HFT Model which combined both the latent factor recommender system with the LDA model.[3] In their experiment, they evaluated their HFT Model on several different data sets including the very same Amazon product data set.[3] Their conclusions with their HFT Model were that review topics generated from LDA models could be used as a regularizer on top of the latent factor model from Koren.[3] They found that their model performed well on data sets that had few reviews for their products, of which the regular latent factor model does not perform well with products that have few ratings.[3] What they found is that these reviews were used to regularize or "justify" the ratings that users gave to certain products.[3]

Also, in McAuley's results, a mean squared error of 1.617 was found from evaluating the mean baseline on the gourmet food category alone.[3] The latent factors baseline yielded 1.515 mse and training a latent factor version of the LDA model achieved 1.492.[3] Their HFT Model achieved mean squared error of 1.434 and 1.431 on user and item latent factors respectively.[3]

Their baseline results were similar to those in this assignment and their LDA and HFT models slightly improved but it is not an applles to apples comparision to this assignment.

Table 2: Highest Negatively Weighted Words

awful	3.4669054645
terrible	3.23634823356
horrible	3.23524527187
return	3.09984400708
disappointed	2.75771270711
waste	2.56963695222
stale	2.24725078399
bad	1.96414167369
bland	1.90975890946
unfortunately	1.90015993994
description	1.79132288491
money	1.6997873314
weak	1.6058314557
hoping	1.59940179003
ended	1.55161903258
opened	1.53306958606
thought	1.53006183918
picture	1.48868761622
label	1.4782481187
maybe	1.47725831471

Our model predicted product ratings from review text directly, but their models were latent factor models trying to predict user tastes/biases so comparing mean squared errors is not a fair comparison. It is impressive that using review texts as a regularizer served to improve the mean squared error by 4-5% on the Amazon gourmet foods data set.[3]

There results are similar to findings in this assignment in seeing that user reviews are closely related to the ratings that people give to products. In their case, they used user reviews as a regularizer to their latent model.[3]

Reviews with positive sentiment tend to correlate to more positive user ratings of products while reviews with negative sentiment tend to correlate to more negative user ratings of products. The models used in this assignment are more for just evaluating user sentiment in text reviews while the HFT Model is more useful for learning user tastes and making predictions for future purchases.

6. CONCLUSIONS

By text mining user reviews of products, it is interesting to find a strong correlation between user sentiment in reviews and ratings of their corresponding products.

Looking at the word clouds in figures 5 and 6, highly-rated products generally correlate to generally positive sentiment words such as ”great” or ”best” or ”good”. Also, more specifically to this data set which consists of grocery and gourmet food reviews, it is interesting to see positive food-related words such as ”delicious” and negative food-related words such as ”stale” or ”bland”.

In future applications to improve this model, more work can be done to incorporate the Latent Dirichlet Allocation topics and probabilities into the model. Also, greater n-gram models can be evaluated to see if improvement in performance can be achieved.

Figure 5: Word Cloud of Positively-Weighted Words

Figure 6: Word Cloud of Negatively-Weighted Words

7. REFERENCES

[1] Amazon product data. http://jmcauley.ucsd.edu/data/amazon/. Accessed: 2015-05.

[2] Ridge regression. http://scikit-learn.org/stable/modules/linear$_m odel.html.Accessed$: $2015 - 05$.

[3] J. L. Julian McAuley. Hidden factors and hidden topics: Understanding rating dimensions with review text. 2013.

[4] Y. Koren and R. M. Bell. Advances in collaborative filtering. pages 145–186, 2011.

Table 3: Topics Generated By LDA and Top Weighted Words (Col = Topic)

great	find	great	like	tea	chocolate	coffee	oil	like	product
product	amazon	use	taste	flavor	like	cup	use	taste	box
good	price	make	good	like	good	like	sauce	sugar	one
love	store	mix	one	taste	taste	flavor	flavor	water	would
would	buy	milk	dont	green	snack	good	like	drink	bag
gift	local	free	would	good	great	taste	great	flavor	amazon
recommend	stores	gluten	really	love	love	great	salt	product	order
order	love	product	im	drink	flavor	one	good	good	time
loved	great	good	try	great	bars	strong	hot	sweet	ordered
candy	grocery	bread	flavor	teas	eat	use	taste	one	get