Amazon Product Rating Prediction

Reya Sadhu)
rsadhu@ucsd.edu

Abstract

Reviews and ratings have a significant influence on consumer shopping behaviour. An online review typically consists of free-form text and a star rating out of 5. The problem of predicting a user's star rating for a product, given the user's text review for that product, is called Review Rating Prediction and has lately become a popular problem in machine learning. In this project, I treat this as a regression problem, where I use item metadata, reviews and also the user-item interaction to predict the rating of a particular review given by a user. I have explored contribution of these features both individually and combined and have shown how they differ.

1 Introduction

User reviews are integral part of e-commerce businesses like Amazon, where users can post their opinions about the business or particular product through textual reviews and numeric ratings. They have a significant impact on customer purchase decisions and thus on business revenue. On websites like Amazon, where there are hundreds of thousands of reviews and ratings, its not possible to go through every one of them to understand customer behaviour or preferences. Its possible to summarize the numeric ratings, but the relationship between reviews and ratings is not obvious. Different users may have different scale of reviews, so even if their ratings are same, the review texts can have very different polarity (Figure 1). Also, the review texts may sound similar but ratings may differ a lot. So it is not reasonable to determine the sentiment scale of a review purely based on its textual content since different reviewers or different products may have consistent biases when being associated the same terms. The process of predicting this relationship between user, reviews and particular rating is called review rating prediction.

Rating	Title	Body
3	Very pleasant tinting	I really liked the styling of these which isn't something I normally say. The tinting was also very pleasant. It may not be the glasses, it may just be me, so take what I say as you see fit! I tried these because I wanted to be able to check my phone or read a few lines when out and about or riding in the car. For that, they worked great and I don't even think I had a problem with the bifocal nature of them. I think my eyes are just spoiled from 30 years of wearing sunglasses with glass lenses, such that when I wear any other kind now, I get a little nauseous. They do come with a handy case you could clip to your belt/bag/lanyard or to keep them from getting crushed in the purse/bag or car If you're fine with plastic, then these might be just what you're looking for!
3	Not checked for quality	When I received them 2-3 of them did not open properly. Looks as though they will break easily

Figure 1: Bias in Ratings

This project deals with the analysis of customer reviews on Amazon, specifically for fashion items. The objective is to build an end-to-end Machine Learning model that predicts review ratings based on not only review contents or item metadata, but also using historical user-item interaction. The desired output is a "star" rating on a continuum from 1 to 5. Because we wish to produce a continuum of sentiment rather than just polarity, this project could be categorized as a regression variation of sentiment analysis.

I have experimented with different feature extraction models to represent the review body and title as high dimensional embeddings. I have used xgboost as a regression model and as final features for the model. I have tried different combination of features like, only review body textual features, review title and body features, user interactions, item metadata and different combination of them.

2 Your Dataset

I have used the Amazon Product Review dataset (Hou et al., 2024), with 8.19 million items, and 571.54 million reviews from 54.51 million users. There are different categories in the data like beauty, electronics, fashion etc. For this project I chose the amazon fashion category. This category has 2M users, 2.5M reviews and 826K items.
There are two datasets in the amazon fashion category, user review dataset and item metadata.

Features	**Details**
rating	rating(1-5)
title	title of review
text	body of review
images	images with review
asin	product id
parent asin	parent id of the product
user id	id of the reviewer
timestamp	unix time of review
verified purchase	boolean user purchase verification
helpful vote	number of helpful votes

Table 1: Reviews Dataset

The item metadata dataset have item related features like price, item images, item categories, average rating, number of ratings etc. I have chose a subset of features

for this project, which is mentioned in detail in experimental setup part.

3 Related Work

Review-rating prediction has traditionally been considered as a feature engineering problem, where authors have exploited various features from the review text, such words, patterns, syntactic structure and semantic topic to improve the performance[(Qu et al., 2010)].
(Fan and Khademi, 2014) predict a restaurant's average star rating on Yelp from its reviews (this is business rating prediction, and is different from review rating prediction). To form the feature vector, they formed bag of words from text reviews of all business, and then picked the top K frequent words. They calculated the frequency of these K words in all, reviews for each business. They combine the unigrams model with feature engineering methods such as Parts-of-Speech tagging, and use linear regression, support vector regression and decision trees for prediction.
But in this project, I have shown that it also depends on review writers and products rather than just the review. (Li et al., 2011) performed rating prediction for reviews by extracting additional features of the reviewer and the product being reviewed.
(Ye et al., 2017) has used product images and review texts along with user item embeddings in a Neural Matrix Factorization model to predict the rating. I have used the similar idea here with a lit bit of modification. So, I have used the semantic analyses of the text, rather than sentiment analysis, along with reviewer and item features to train a modified NMF model.

4 Experimental Setup

In this project, I explored three distinct approaches for rating prediction. The first approach involves collaborative filtering and matrix factorization, focusing solely on user-item interactions. The second approach utilizes only the review text and title to predict ratings. Building on this, I extended the method by incorporating item-specific features alongside the review text to evaluate their impact on prediction accuracy. Finally, I implemented a neural matrix factorization technique to combine the strengths of the previously mentioned approaches into a unified model.

4.1 Pre-processing

Due to the large volume of data, training on all reviews is computationally expensive. To make the analysis manageable, I sampled a subset of users and items. One of the key challenges in recommendation systems is the cold start problem, where unseen users or items appear in the test set but were absent in the training set. However, since the primary goal here is to predict ratings based on reviews, along with item and user features, I have chosen to exclude the cold start problem from this analysis.

- **Reviews Data:** To ensure meaningful insights, I filtered users with more than four interactions in the review dataset and split their interactions across training and validation sets, avoiding unseen users in the validation set. Similarly, I selected products with more than four interactions to maximize the number of items with sufficient historical data. After this filtering and splitting process, the training dataset consists of 26k users, 40k items, and 72k reviews.

- **Item Metadata:** As I am using item features in predictive modeling, I only kept features with meaningful information and removed rest of them. Images and videos were excluded from the scope of this project. Most columns in the metadata contained very few non-null values. While columns like price, bought together, categories, and description could have provided valuable information, over 90% of their values were null, leading to their exclusion. The store column, although potentially useful as a categorical feature, was also excluded due to its high cardinality with over 12k unique values, which significantly increased feature dimensionality. Ultimately, for the metadata, only the columns asin, parent asin, average rating, and rating number were retained.

- **Text data:** For the textual features, like review body, title, item titles, I first normalize the dataset by converting all the characters to lowercase. I have also converted all whitespace and punctuation into a single space to get rid of any inconsistencies. Then I have used NLTK's stopwords package to remove any stopwords like I, are, there that do not contribute in any polarity. Then I have used lemmatization to transform any word into their base form like run from running and so on. This steps make the textual data uniform throughout the datasets.

4.2 Exploratory Data Analysis

The original review dataset spans from 2007 to 2023. After analyzing the distribution, I found that 96% of the reviews were written after 2015, while only 4% were from the period between 2007 and 2015. This distribution seems reasonable, as Amazon was not as widely popular during the earlier years. Consequently, there were fewer users and a lower tendency to review and rate products. For this project, I included only reviews

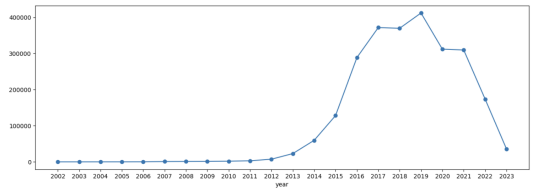

Figure 2: Reviews Count By year

posted after 2015. To proceed with feature selection, I examined the relationship between various features and the target variable—rating. Specifically, I derived several time-related features from the timestamp, such as year, month, day of the week, and hour of the day. However, upon plotting the correlation matrix, it became evident that ratings show no significant correlation with any of these time-related features.

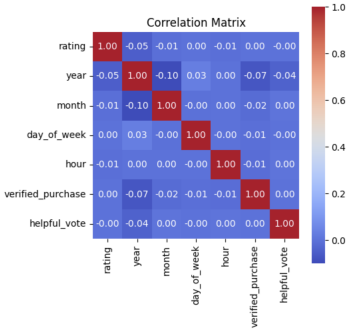

Figure 3: Ratings correlation

However, since correlation measures only the linear relationship, it is better to examine the individual distributions by plotting them to gain a deeper understanding of their dependency.

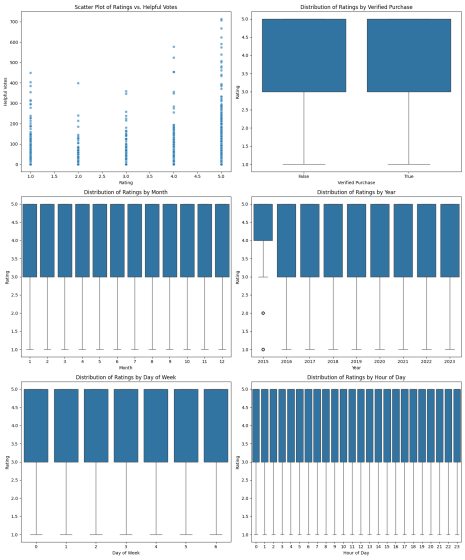

Figure 4: Distribution of features with ratings

Since none of these features show a clear dependency on the ratings, I have excluded them from the modeling. If we examine item-specific features like average rating and rating count, we can see that average rating has a strong positive correlation with review ratings, as expected. Although the correlation between rating and rating count is weaker, there is still a positive relationship between average rating and rating count. This is also evident from the bar plot, which suggests that products with higher ratings tend to receive more reviews. Therefore, I have included both average rating and rating count as item features in the model.

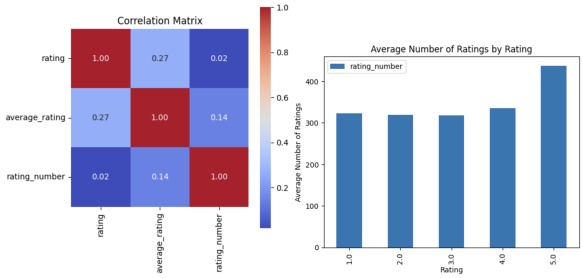

Figure 5: Rating by item features

4.3 Feature Extraction

The key focus is on the review texts. I used four different methods to extract meaningful features from the reviews and create a feature vector for each one. To explore which method might work best, I started by creating word clouds for the review texts—one for reviews with a rating of 1 and another for those with a rating of 5.

(a) Words for rating 1

(b) Words for rating 5

Figure 6: Word cloud for different ratings

The words in the reviews clearly show whether the review is positive or negative. For example, reviews with a rating of 1 often contain negative words like “cheap”, “disappointed” and “broken” while reviews with a rating of 5 tend to have positive words like “love”, “nice”, and “perfect”. This is why I started by experimenting with simple models based on word frequency. Additionally, when analyzing the meaning of text, bigrams (two-word phrases) work better than unigrams (single words), as phrases like “not nice” or “too small” can change the meaning of the words. Therefore, I experimented with both unigram and bigram

feature extraction methods. Below are the techniques I used in the modeling process:

1. **Count Vectorizer:** Its a bag of words model, where we represent each word by its corresponding frequency in corpus. To keep our feature dimension in bound, I have used 2000 most frequent words as features for unigram and 4000 for bigram.
2. **TF-IDF Vectorizer:** Penalize the most frequent words in a document by its count in all documents. So that if a word is very common for all the files, it doesn't hold much importance.
3. **Word2Vec:** We fit a small neural network to predict the context of a word from the word. The weight matrix is used as word embeddings. For our model, I have used a window size of 5 and embedding length of 5.
4. **BERT :** I have used pretrained BERT tokenizer and model to get embeddings from the sentence. I have experimented with max pooling, mean pooling and CLS token embedding to get the final feature vectors.

5 My Implementation

For the baseline, I have used filtering techniques by only considering user item interaction.

- **Collaborative Filtering:** The rating is used to form the interaction matrix. As the matrix is very sparse, I have used the sklearn csr matrix for the computation. For calculating similarity I have used cosine similarity.
  - **User-user:** Similarities between users are measured and then following formula is used to estimate the rating.

$$r(u,i) = \bar{R_u} + \frac{\sum_{v \in U_i \setminus \{u\}} (R_{v,i} - \bar{R_v}) \cdot \text{Sim}(u,v)}{\sum_{v \in U_i \setminus \{u\}} \text{Sim}(u,v)}$$

  - **Item-Item:** Similarities between items are measured and then following formula is used to estimate the rating.

$$r(u,i) = \bar{R_i} + \frac{\sum_{j \in I_u \setminus \{i\}} (R_{u,j} - \bar{R_j}) \cdot \text{Sim}(i,j)}{\sum_{j \in I_u \setminus \{i\}} \text{Sim}(i,j)}$$

- **Matrix Factorization:** I have used SVD plus from the surprise library to achieve the matrix factorization.

Now, for the review rating prediction, I have used 3 different feature combination. Each combination is used with 4 feature extraction techniques mentioned before to generate word vectors. The created vectors are then used by a xgboost model for training. I have experimented with different ML techniques like linear regression, random forest, LightGBM, but xgboost works best for almost all techniques. So, only that result is mentioned here.

- **Only review body:** Only consider the body of the review as feature.
- **Review body and title:** Use both of them as features. They could have transformed by separate vectorizers, but they both are just text features, and they are indistinguishable. So, they were concatenated and delimited by a space to further reduce feature complexity.
- **Review body, title, item features:** With the textual information, numeric item features like average rating and rating number are used. They are normalized by standard scaler and concatenated with the word embeddings before passing to the model.

- **Neural Matrix Factorization:** Neural Matrix Factorization (NMF) is an extension of traditional matrix factorization methods that incorporates neural networks to capture more complex patterns and relationships in the data. The following diagram (Li et al., 2020) shows how it's generally structured.

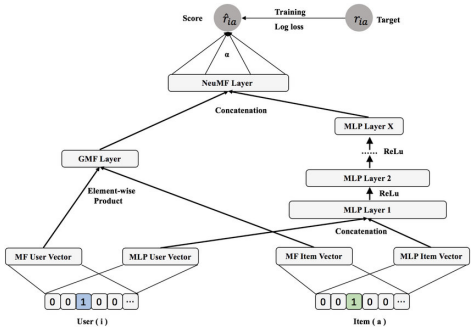

Figure 7: Neural Matrix Factorization

To integrate review and item features, I made a slight modification to this structure. The updated framework captures both interaction and review features, enhancing its ability to estimate rankings more effectively. Also, I have used MSE loss instead of log loss as we are considering this a regression problem.

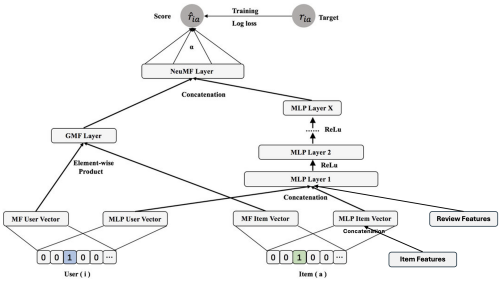

Figure 8: Modified Neural Matrix Factorization

6 Results

I have used Mean Squared Error as our metrics for regression analysis. The below tables show the MSE on validation set.

Type	User CF	Item CF	SVD
Baseline	1.435	1.459	1.258

Features	Review Body	Review Title & Body	Review title, body, item feature
CV Unigram	0.753	0.537	0.527
CV Bigram	0.742	0.506	0.5
TF-IDF Unigram	0.709	0.516	0.514
TF-IDF Bigram	0.704	**0.489**	0.492
Word2Vec	0.697	0.53	0.52
Pretrained BERT	0.814	0.67	0.65
Neural MF			**0.48**

7 Conclusion

From the result, it can be seen that simple word vectorization techniques, such as bag-of-words and TF-IDF, outperform advanced methods like BERT for this task. This is primarily because individual words in reviews often have a strong correlation with ratings—for example, words like terrible are associated with low ratings (0), while words like excellent align with high ratings (5). As a result, deep semantic understanding of the text is often unnecessary to capture its polarity. But since bigrams can completely alter the meaning of unigram, we can see much better results when using bigrams.

Using the review title as a feature significantly improves the model's performance, increasing it by almost 30% compared to when only the review body is used. This shows that the title, which often contains clear and concise information, adds value beyond the detailed content in the review body. Adding item features also helps improve performance, suggesting that having more information about the items, like their attributes, makes the predictions more accurate. These results highlight the importance of combining different types of information—like text and item details—for better performance in recommendation systems.

Interestingly, matrix factorization also performs moderately well despite having no access to the textual data. By combining the strengths of both approaches in a neural matrix factorization framework, we achieve the best results in terms of mean squared error.

If we see the actual vs predicted ratings, we can see the skewed distribution is captured well. For further work, we can consider a non skewed balanced data to see if that improves performance. We can also try POS tagging with the review texts, so that we can create better feature vectors. We can also use sentiment score on each review using some pretrained and finetuned model , and use it as a feature.

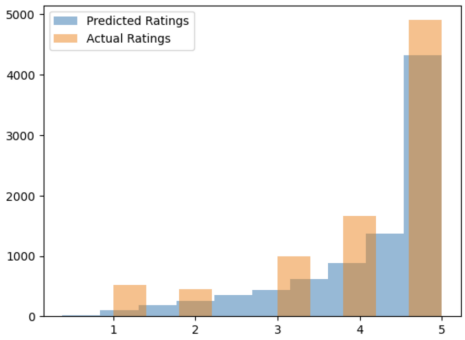

Figure 9: Actual Vs Predicted Ratings

The implementation for the project can be found here. Source Code

8 Acknowledgements

- I have used ChatGPT to restructure some of the sentences that I wrote.

References

Fan, M. and Khademi, M. (2014). Predicting a business star in yelp from its reviews text alone.

Hou, Y., Li, J., He, Z., Yan, A., Chen, X., and McAuley, J. (2024). Bridging language and items for retrieval and recommendation.

Li, F., Liu, N., Jin, H., Zhao, K., Yang, Q., and Zhu, X. (2011). Incorporating reviewer and product information for review rating prediction. pages 1820–1825.

Li, J., Wu, L., Hong, R., Zhang, K., Ge, Y., and Li, Y. (2020). A joint neural model for user behavior prediction on social networking platforms. *ACM Transactions on Intelligent Systems and Technology*.

Qu, L., Ifrim, G., and Weikum, G. (2010). The bag-of-opinions method for review rating prediction from sparse text patterns. volume 2, pages 913–921.

Ye, W., Zhang, Y., Zhao, W. X., Chen, X., and Qin, Z. (2017). A collaborative neural model for rating prediction by leveraging user reviews and product images. In Sung, W.-K., Jung, H., Xu, S., Chinnasarn, K., Sumiya, K., Lee, J., Dou, Z., Yang, G. H., Ha, Y.-G., and Lee, S., editors, *Information Retrieval Technology*, pages 99–111, Cham. Springer International Publishing.