Music to Your Ears: Song Recommendation Using Rating Prediction on Amazon Digital Music

Casey Graff *A11419976*
Sirinya Phakoom *A10680320*
Shaina Wan *A98112057*

Abstract

In this project, we worked with Amazon product reviews from the "Digital Music" category to perform rating prediction. We studied the unique features of the dataset and how they apply to rating prediction in recommender systems. Then went go on to utilize the information contained in the ratings that users assigned to the products they bought, and attempted to extrapolate their future purchases from that. In particular, our main focus was the use of a Latent Factor Model (LFM) to discover underlying, low-dimensional representations in the rating data.

1 Dataset

We used a collection of over 800,000 Amazon product reviews for the "Digital Music" category [1]. This data reflects customer purchase history, product rating and user sentiment. The information was captured through two files.

- **meta.json:** a list of each product categorized under "Digital Music".
- **reviews.json:** a list of reviews by users who purchased items in the "Digital Music" category.

The data was incredibly sparse, as demonstrated by **Figures 1** and **2**, which depict the number of users and products (respectively) that had a specific number of reviews. The vast majority of users had only reviewed one product, and most products had been purchased only once. This sparsity was a challenge to our predictive task, as it meant that our models could not generate reliable representations of the users or products.

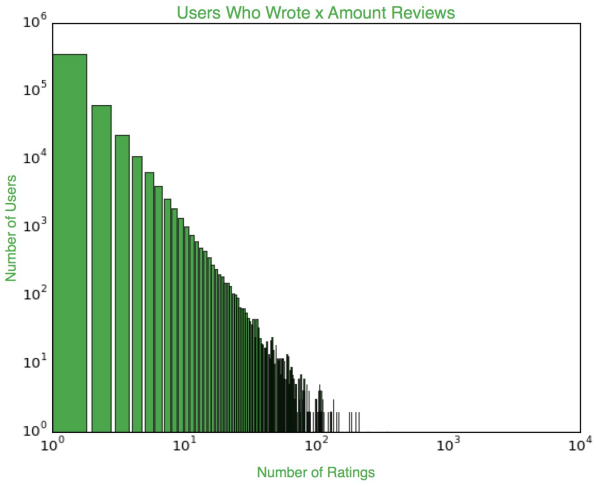

Figure 1: *Number of users with x amount of reviews*

Dataset	Users	Products
1	478,243	836,015
2	108,028	283,152
3	22,941	141,497

Table 1: Size of data sets

To attempt to address the issue posed by sparse data we generated three separate data sets by filtering out users who had reviewed only a small number of products (**Table 1**). This technique has been demonstrated to be effective for other rating prediction tasks [3]. We discovered, however, that this was not the case for our data and models.

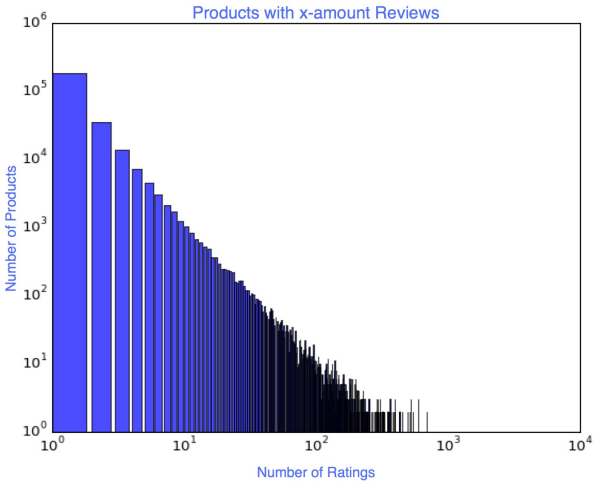

Figure 2: *Number of products with x amount of reviews*

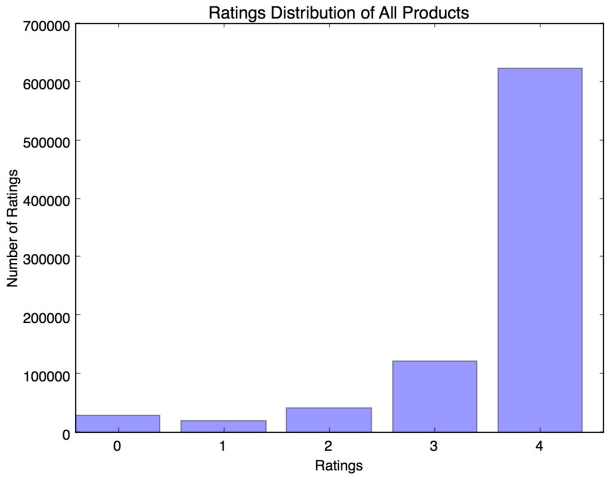

Figure 3: *Distribution of Ratings*

- **Dataset 1:** All users and products from the raw data are included.
- **Dataset 2:** Only users with at least 2 reviews are included.
- **Dataset 3:** Only users with at least 5 reviews are included.

In addition to filtering by the number of reviews a user had made, we pursued the idea of filtering products by the number of reviews they had received. Unfortunately, so few products in the data set are purchased more than once that the remaining reviews where both the user and the product had a sufficient number of reviews left too small of a set to be useful.

Another interesting characteristic of our data is the distribution of the ratings. **Figure 3** depicts the fact that the number of five-star ratings giving greatly exceeds the number of any other rating category. The average rating given was 4.539 and the variance of the rating data was 0.960.

2 Predictive Task

We chose to predict user ratings. For any given user-product pair we would have to generate a discrete numerical value between 1.0 and 5.0. To test the models we developed we split them into a training, validation and testing set. Our models used purely the numerical rating data - that is, it was not feature driven. This choice was driven from previous successful recommender engines, such as that used in the Netflix competition, that primarily relied on rating data to make predictions. We used the global average from the training set as a baseline model and attempted to improve from there using three other models: using user and product biases, using only latent factors, and using both biases and latent factors.

Global Average (GAVG): This model relies on taking the average rating from all reviews contained in the training set. This predictor is generally reliable, especially given the relatively low variance of the data. For each user-product pair in the test set we would predict:

$$r_{u,i} = \mu \tag{1}$$

Bias Model (BM): This model relies on the assumption that users are likely to rate similarly across different products and that products are likely to be rated similarly across multiple users. We calculated a global bias to shift all ratings by and then a user and product specific bias.

$$r_{u,i} = \alpha + \beta_u + \beta_i \tag{2}$$

This was solved iteratively, by fixing two parameters and then minimizing the following equation

with respect to the variable. This function is convex, so convergence to the global minimum will be achieved.

$$argmin \sum_{u,i} (r_{u,i} - \alpha + \beta_u + \beta_i)^2 - \lambda * (\beta_u^2 + \beta_i^2) \quad (3)$$

Latent Factor Model (LFM): This model relies on representing users and products using a low-dimensional vector and that compatibility between a particular user and product may be determined by the inner-product between the low-dimension representation of each.

$$r_{u,i} = \langle \gamma_u, \gamma_i \rangle \quad (4)$$

To generate the low-dimensional representation, we used stochastic gradient descent to minimize an objective function which included regularization.

$$argmin \sum_{u,i} (R_{u,i} - r_{u,i})^2 - \lambda * (|\gamma_u|^2 + |\gamma_u|^2) \quad (5)$$

Latent Factor Model with Bias (LFMB): This model is very similar, but is the combination of the previous two models. A rating prediction used the following equation and was also solved using stochastic gradient descent.

$$r_{u,i} = \alpha + \beta_u + \beta_i + \langle \gamma_u, \gamma_i \rangle \quad (6)$$

These values were calculated in a very similar way to LFM, using stochastic gradient descent to minimize the following objective function.

$$argmin \sum_{u,i} (R_{u,i} - r_{u,i})^2 - \lambda * (\beta_u^2 + \beta_i^2 + |\gamma_u|^2 + |\gamma_u|^2) \quad (7)$$

Of these models we expected the LFMB to perform the best due to the inclusion of the most information. As will be shown by our results, this was not the case.

3 Literature

Predicting ratings is one of the most prevalent problems in creating recommender systems, modeling the relationship between consumers and products. These problems are important, especially for companies such as Netflix, who hosted the famous Netflix Prize in an effort to improve their recommendation system. The winners, Koren and Bell, utilized a combination of Latent Factor Models, SVD-based models, and Collaborative Filtering [2], which were very effective since those techniques mapped users and the products into low-dimensional forms and gave recommendations based on similarities between users and similarities between products. Jin and Gu [3] also followed a similar approach to the Netflix prize winners, using latent factor models to predict the significance location and environment has on businesses, breaking down the latent factors into the business's inherent, internal factors and the external factors of their neighborhood and the surrounding, adjacent businesses. Seroussi, Bohnert, and Zukerman [4] addresses cold-start problems by integrating the user input and features from user text into a latent factor model. Seroussi successfully employed text analysis to yield information about the user interests without requiring the user to provide explicit feedback. Chen and his team [5] examined the predicted ratings distribution and concluded that the success of latent factor models are strongly affected by the distribution of reviews. Santos [6] proposed a hybrid recommendation system, which incorporates user demographics with the items' features and user feedback to build their latent factor model. Based on those factors, the recommender system would present the user with material that would correlate with their profile, thus illustrating the importance of including metadata in building a successful model. Kanagal [7] discusses how latent factor models in recommender systems often rely heavily on clustering the similar user and items together to determine predictions. Kanagal combines traditional latent factor model methods with taxonomy classification to produce a more efficient algorithm since this allows the model to make more accurate deductions and inferences, thus relying less heavily on user input.

4 Results

The results we received using the methods described above were unexpected. For one, filtering are data

into multiple data sets, through which we tried to make our data more dense, did not work. In fact, filtering worsened our results in most cases. For another, our LFMB performed worse than all other models in the two filtered datasets (#2, 3). As a note, we used the mean average error for our error calculations.

Our baseline using GAVG performed relatively well compared to our models. In all three data sets it performed similarly to the BM, and in Datasets 2 and 3 it outperformed the LFM and LFMB. This is surprisingly well for such a simple predictor, but is understandable given the low variance of the ratings. Fortunately, our LFM and LFMB for Dataset 1 greatly outperformed the baseline, indicating that they were reasonably successful. Our LFM was run given a k value of 10, for a fixed set of 15 iterations, with $\eta = 0.05$. The LFM and LFMB used λ values of 2 and 7, respectively, utilizing the aforementioned validation set.

As can seen in **Table 2**, Datasets 2 and 3 performed worse using almost every model, and only slightly better in one. Although our intent was for the filtering to improve results, the opposite effect occurred. In reflection, this is likely the result of two simultaneous facts. One, the data's sparsity did not decrease substantially from the filtering (e.g. most products still only had a single review). Two, filtering the data dramatically decreased the number of available training points, and as such reduced are ability to estimate the global bias. Furthermore, the larger data likely had a lower overall variance so it was easier for our predictors to do well.

The second area worth analyzing is the performance of the various models. It would seem reasonable to assume that using the combined LFMB would improve results over the LFM, but this was not the case. One possible explanation is that in the LFM model if either the product or the user was missing, the predictor had to default to using the global average. However, in the LFMB, if one of the two was found in the training data, the user or item bias could still be used. Perhaps, since each user and product had so few reviews that this bias was less accurate than the global average.

The most astonishing result was the LFM on Dataset 1 which was nearly half of the next lowest value. We believe that this was the result of a combination of the positively contributing factors mentioned in the two paragraphs above coupled with a well validated regularizing value and gradient descent step-size.

Model	Dataset 1	Dataset 2	Dataset 3
GAVG	0.460	0.461	0.494
BM	0.489	0.459	0.489
LFM	0.158	0.588	0.564
LFMB	0.319	0.693	0.693

Table 2: MAE of Methods Used

5 Conclusion

From our experiment we have made several interesting insights. Firstly, that although filtering the data to try and improve density may work on some data sets, it is not always effective, and may actually worsen predictions. Secondly, for data that is inherently sparse, as is the case in this data, using solely rating based methods may not yield the best results. The inclusion of models that incorporate feature data, such as the product categories, may help further improve accuracy, and further map out the user's interests. As referenced, considering the similarity of user and products would help to connect users to new products. Also, as referenced in [4], incorporating user review text would also enhance the accuracy of our user inferences since this could give us more data to parse. This may be especially true for products and users that have not been seen in the training data, often referred to as the cold-start problem.

There are several ways to improve upon our results. The first is to have access to more powerful computational devices. When performing the iterative procedure for gradient descent and when consistently retraining models to find optimal values for hyper-parameters, computational complexity was always a limiting factor. As a result the number of parameters (such as the dimension of the latent factors), the number of iterations, and the number of hyper-parameter configurations had to be reduced. With more significant computational power, these results can easily be extended. The second is to, as mentioned, utilize features to improve the

model's ability to generalize to users and products that have not be seen in the training set. Lastly, including text analysis would further increase the performance of our model since it would yield more implicit user data so that we could generate a more accurate user profile.

References

[1] J. McAuley, C. Targett, J. Shi, A. van den Hengel "Image-based recommendations on styles and substitutes" In SIGIR, 2015

[2] Robert M. Bell, Yehuda Koren *"Lessons from the Netflix prize challenge"* ACM SIGKDD Explorations Newsletter - Special issue on visual analytics, Volume 9 Issue 2, December 2007 Pages 75-79 (2007)

[3] Long Jin, Xinchi Gu *"Rating Prediction for Google Local Data"* In CSE 255, 2015.

[4] Yanir Seroussi, Fabian Bohnert, and Ingrid Zukerman. *"Personalised Rating Prediction for New Users Using Latent Factor Models."* Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia - HT '11 (2011)

[5] Cheng Chen, Lan Zheng, Alex Thomo, Kui Wu, and Venkatesh Srinivasan. *"Comparing the Staples in Latent Factor Models for Recommender Systems."* Proceedings of the 29th Annual ACM Symposium on Applied Computing - SAC '14 (2014)

[6] Edson B. Santos, Jr , Marcelo G. Manzato, and Rudinei Goularte. *"Hybrid Recommenders."* Proceedings of the 19th Brazilian Symposium on Multimedia and the Web - WebMedia '13 (2013)

[7] Bhargav Kanagal,, Amr Ahmed, Sandeep Pandey, Vanja Josifovski, Jeff Yuan, and Lluis Garcia-Pueyo. *"Supercharging Recommender Systems Using Taxonomies for Learning User Purchase Behavior."* Proceedings of the VLDB Endowment Proc. VLDB Endow. 5.10 (2012)