Cross Category Recommendation Systems: Prediction of User Tendencies Based On Previous Consumption

Sue Deng, Lea Wang-Tomic
Stanford University
suedeng@stanford.edu, leawt@stanford.edu

December 6, 2024

1 Introduction

Our objective is to develop a machine learning model to predict customers' cross-category rating behaviors. Traditional recommendation systems focus on recommendations within single-category and primarily suggest products based on user purchase or browsing history. However, these systems often lack an understanding of potential user feedback, which can result in recommending irrelevant items. By predicting user review scores across categories, our approach seeks to uncover deeper insights into user behavior, enabling improved personalization and user engagement.

The input to our algorithm includes user historical reviews in one product category, consisting of features such as user average rating, average sentiment scores of review texts, etc. Additionally, it incorporates product features from product metadata, including attributes such as sentiment of product description, product title length, etc. Initially, we developed a two-layered model to first predict whether a user from one product category is likely to purchase products from another category, followed by review rating prediction. However, since Amazon reviews often include non-verified purchases due to purchases from other retailers or gifts, we refined the approach to predict review scores only. We employed techniques such as regression, KNN, and different filtering models to output a predicted review rating score (1–5), and then subsequently recommended to the user the top five items based on the predicted highest rated values.

2 Related work

Recent approaches in e-commerce recommendation systems include content-based, collaborative, and hybrid filtering systems. Content-based filtering provides recommendations based on a user's profile, which ensures user privacy and adaptability to changing preferences. Despite its benefits, it suffers scalability issues due to individualized profiles. Collaborative filtering leverages similarities between users or items to facilitate novel recommendations without requiring specific item attributes. However, it encounters challenges such as cold-start problems, privacy concerns, and computational inefficiencies, especially with sparse, large-scale datasets. Hybrid filtering systems merge the strengths of both approaches, enhancing accuracy and mitigating limitations like cold-start, although they are complex to implement and computationally demanding(Roy&Dutta, 2022).

Marie (2016) explored using NLP techniques to analyze product reviews and predict user ratings. This method further improves customers experiences, although there could be biases inherent in human feedback and the specificity of vocabulary related to different product categories. To tackle computational efficiency challenges, Amazon's item-to-item collaborative filtering, noted for its scalability and real-time recommendation capabilities, stands out(Linden, Smith & York, 2003). Additionally, to tackle data sparsity, Rendle introduces Factorization Machines (FMs) that combine the advantages of Support Vector Machines with factorization models, making them particularly effective in e-commerce environments where traditional methods falter due to sparse data(Rendle, 2010).

Further, Two-Tower Models like the Dual Augmented Two-tower Model (DAT) enhance user and item interactions through innovative mechanisms such as the Adaptive-Mimic Mechanism (AMM) and Category Alignment Loss (CAL), boosting performance in large-scale environments(Yu,Wang,Feng&Xue, 2021). Our research aims to broaden recommendation diversity and accuracy by developing models that predict cross-category ratings, integrating NLP analysis of user reviews with traditional machine learning models such as regression and innovative techniques like collaborative filtering and Two-tower models.

3 Dataset and Features

3.1 Data Overview, Category Selection and Data Filtering

We utilized 2023 Amazon Review dataset, which contains user review data and product metadata for over 30 product categories. Given the extensive dataset (over 500 million reviews across 50 million users), we focused on the Beauty and Fashion categories only for our analysis due to their notably high overlap in customer interactions, with **116,4283**, unique users making cross-category purchases—significantly higher than other category interactions. A preliminary analysis verified that there are no overlapping products between the two categories, and each user-product review pair is unique, which ensures data integrity for our cross-category predictions. In data-processing phase, we noticed a challenge - the user-product pair matrix for these categories remained sparse, exhibiting a median of only one review per user. To increase data density, we refined our dataset by including only users who posted more than the median number of reviews. This filtering process reduced our dataset by 50%. Furthermore, we narrowed the product scope by retaining only the top **1,000** most-reviewed products in each category. This selection led to a 99% reduction in data within each category, resulting in **3,431** unique user-product pairs in the Fashion category and **4,651** in the Beauty category with available rating labels. Subsequent analysis of the correlation matrices for each category revealed minimal variance (*less than* ± 0.03)4, indicating similar consumer behaviors across these two categories. This similarity provides a robust foundation for using Fashion category data to predict user ratings in the Beauty category.

3.2 Feature Engineering

We engineered a range of features based on each customer's behavior and relevant product attributes. These features include:

- **user average rating**: the mean rating given by each user, indicating overall satisfaction.
- **user std rating**: standard deviation of user ratings, with higher values suggesting varied opinions.
- **avg user sentiment title/text**: we used TextBlob to calculate sentiment scores from review titles and text by each user.
- **user review count**: total reviews by each user, reflecting engagement level.
- **helpfulness ratio**: ratio of helpful votes to total reviews, measuring perceived value.
- **product title length**: number of words in each product title.
- **rating number**: number of reviews by product.
- **average rating**: average review rating by product.
- **description sentiment score**: used TextBlob to calculate sentiment scores from product description text.

3.3 Data Splitting and Normalization

In our study, rather than adopting the traditional approach of randomly splitting data into training and testing sets, we specifically designated the Fashion category data, which includes user profiles and product attributes, as the training set5 (3431 training examples). Conversely, we employed data from the Beauty category, similarly characterized by user profiles and product attributes, as the testing set (4651 testing examples). This way aligns with the project's emphasis on cross-category prediction. Additionally, we have also normalized the features for standardization, ensuring that each feature contributes proportionately to the final prediction, thereby enhancing the model's ability to generalize across different data scales.

4 Methods

We established two baseline models: Linear Regression and K-Nearest Neighbors (KNN), followed by advanced recommendation system techniques, including Collaborative Filtering, Content-Based Filtering, and a Hybrid Model.

Linear Regression predicts continuous outcomes by estimating coefficients for predictor variables in a linear framework. Mathematically, it is expressed as:

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

where $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ represent the model parameters, and $x_1, x_2, \ldots, x_n$ are the predictor variables. The parameters β are optimized by minimizing the residual sum of squares between observed targets and predicted values using gradient descent.

The **K-Nearest Neighbors (KNN)** algorithm predicts a user's rating based on the ratings of the k most similar data points, determined by a distance metric such as Euclidean distance:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

KNN identifies similar users or products and infers user preferences by averaging the ratings of these nearest neighbors. Hyperparameter tuning was performed to optimize the number of neighbors (k) and the weighting scheme (uniform vs. distance-based). The optimal configuration was found to use $k = 12$ neighbors with uniform weighting.

We then applied **Collaborative Filtering**, which leverages historical interactions between users and items to predict user ratings. The similarity between entities is quantified using cosine similarity:

$$\text{sim}(x, y) = \frac{\sum_{i \in I} r_{xi} \cdot r_{yi}}{\sqrt{\sum_{i \in I} r_{xi}^2} \cdot \sqrt{\sum_{i \in I} r_{yi}^2}}$$

where r_{xi} and r_{yi} are the ratings of entity x and y for item i, and I is the set of items rated by both entities. The predicted rating is a weighted average of the ratings from similar entities:

$$r_{ui} = \frac{\sum_{v \in N(x)} \text{sim}(x, v) \cdot r_{vi}}{\sum_{v \in N(x)} |\text{sim}(x, v)|}$$

where $N(x)$ represents the set of nearest neighbors for entity x.

Next, we implemented **Content-Based Filtering**, which uses features of users and products to predict ratings. User-specific features include attributes such as average ratings, sentiment scores from review texts, and review counts, while product-specific features include attributes such as title length, average rating, and description sentiment. The model predicts a user's rating for an item by calculating the dot product between the feature vectors of the user and the item:

$$r_{ui} = \mathbf{w}_u \cdot \mathbf{f}_i$$

where $\mathbf{w}_u$ represents the weight vector of user u, capturing their preferences, and $\mathbf{f}_i$ is the feature vector of item i.

Finally, we developed a **Hybrid Model** that combines Collaborative Filtering and Content-Based Filtering. By integrating the strengths of both approaches, the Hybrid Model improves prediction accuracy and mitigates the limitations of individual methods, such as cold-start problems in collaborative filtering and over-specialization in content-based filtering.

5 Experiments/Results/Discussion

The key evaluation metric used for our models was Root Mean Squared Error (RMSE = $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$) to get a measure of how far off our predicted ratings are from the actual ratings. Focusing on minimizing RMSE ensured we prioritized accuracy and thus prioritized user preference in our recommendation system.

Our experiments had three core approaches: linear regression, KNN, and filtering models (collaborative filtering, content based filtering, and a hybrid model). We also followed up our feature engineering by conducting experiments to see the primary influence of the features on the accuracy of the model's predictions. Each method's

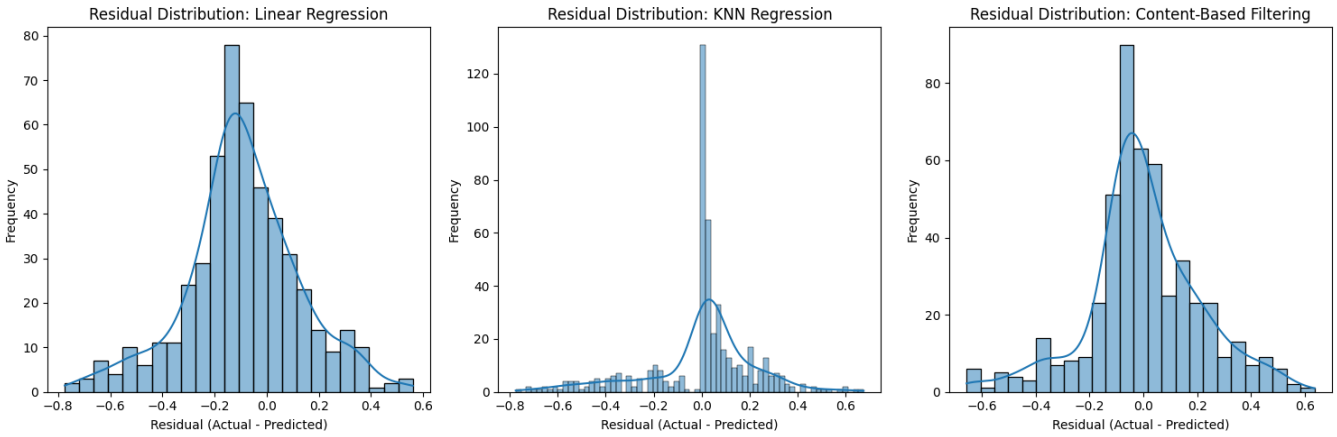

Figure 1: Histogram of Residuals for Best Performing Models

results are summarized below in Table 1. Linear regression and KNN (with predictions just based on user's, the items they purchased, and the ratings they left) served as a simple baseline model to predict review scores. Linear regression is the simplest model to select for the predictions of the ratings left by users, and KNN is the next (slightly more complex but still relatively simple) model used to incorporate user-specific insights. We then followed up these experiments by doing the same models but with the datasets we developed through feature engineering, including both user and product we tuned the hyper-parameters and were able to find the standard metrics for weighting and distance (uniform and euclidian) were optimal as well as discover 12-neighbors determines the best categories. Overall improving the performance of the KNN model.

Model	RMSE
Linear Regression (baseline)	0.9220
KNN (baseline)	0.9219
Linear Regression (w/ feature engineering)	0.1949
KNN (w/ feature engineering)	0.1859
Content-Based Filtering	0.1912
Collaborative Filtering	0.7589
Hybrid Filtering	0.2113

Table 1: Best RMSE Results for Each Experiment

Next we wanted to see if we could dig deeper into exploring user and item interactions (both between users, between items, and between user-items). We applied collaborative filtering to analyze patterns in user-item interactions. However the filtering model was not effective and significantly raised the RMSE. This is likely due to sparsity in the data. Collaborative filtering has a hard time dealing with cold-start problems (working with unseen data), and the interactions between users and items in our data are sparse. From our feature analysis (see below), we were able to determine user specific information had the strongest influence on the performance of the model, and user-item affinities were actually not great at helping predictions. Continuing with filtering methods we also explored a content-based filtering approach, which explored the user generated features and the item specific attributes but used the engineered features to understand the user-item interactions. This method doesn't depend solely on user-user similarities or item-item similarities the way collaborative filtering does, so it had a better performance on this dataset. Finally we experimented with building a hybrid model to see if we could combine different aspects of content and collaborative filtering.

Seeing as feature engineering was the most significant factor in minimizing the RMSE, we evaluated the influence of each feature on the model performance. We were able to determine features exploring user behavior (such as a user's average rating or their standard rating) have more importance on the accuracy of the model. This indicates that user behavior should be prioritized over item specific information when creating recommendation systems. Product

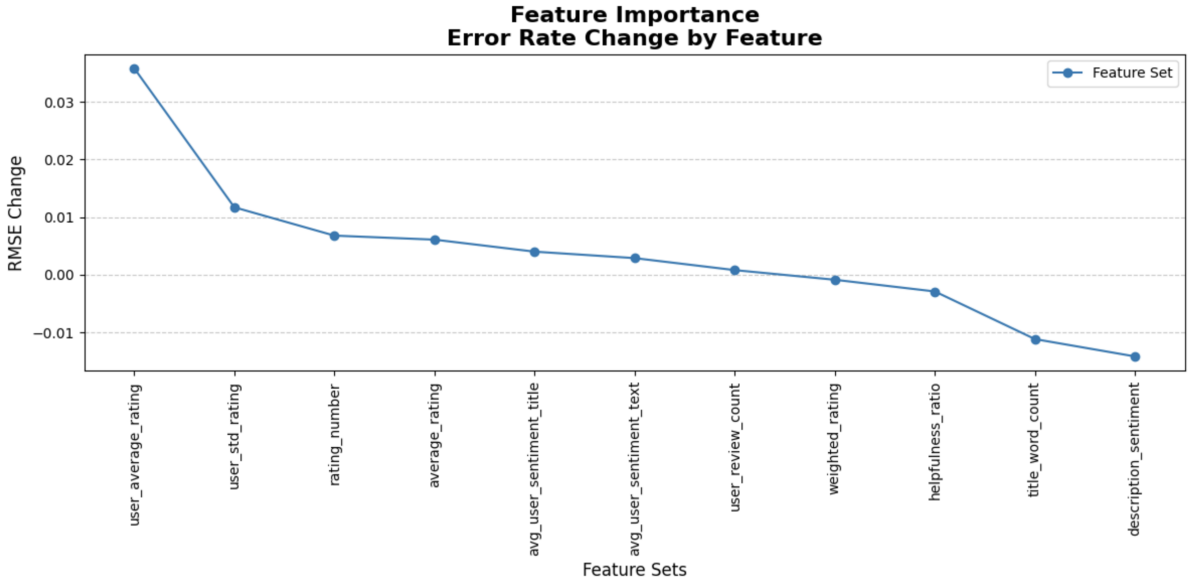

Figure 2: Feature Importance

specific attributes helped with contextualizing user ratings, but were second to the user specific information in their predicting power.

Overall the findings emphasize the significant role of feature engineering in recommendation systems. For real-world applications, this means recommendation systems should prioritize incorporating personalized features and context-driven insights over developing complex models to maximize relevance and user satisfaction.

6 Conclusion/Future Work

In this project, we aimed to predict cross-category rating and purchasing behaviors in a recommendation system using machine learning techniques. Through a combination of basic models (Linear Regression and KNN), advanced filtering methods (Content-Based, Collaborative, and Hybrid Filtering), and robust feature engineering, we demonstrated how user-specific and product-specific features impact the accuracy of recommendations. Feature engineering proved to be the key factor in getting proper accuracy, significantly improving model performance, with KNN achieving the best results (RMSE: 0.1859) when incorporating features like user average ratings, sentiment scores, and engagement levels. This highlights the importance of understanding user behavior for accurate predictions, as user-specific features had a greater influence than item-specific attributes. Though filtering methods are often the recommended approach to use for recommendation systems, they didn't perform as well as anticipated. Collaborative Filtering, while effective in leveraging metadata highlighted by the feature engineering, struggled due to data sparsity and the unique challenge of cross-category predictions. Showing in this instance, simpler is better.

With more time, future work could focus on deeper feature engineering, such as using advanced NLP models for more nuanced sentiment analysis. Additionally, it would be interesting to try more complex recommendation algorithms, such as Neural Collaborative Filtering or Two-Tower Models to see if they do a better job at capturing complex user-item interactions. By exploring these avenues, we could develop even more effective and scalable recommendation systems tailored to diverse user needs.

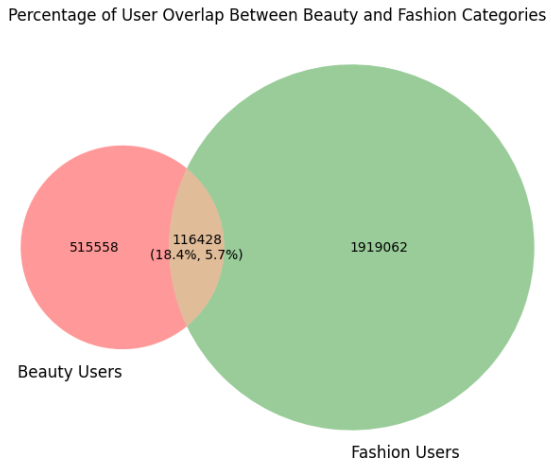

Figure 3: Percentage of User Overlap Between Beauty and Fashion Categories

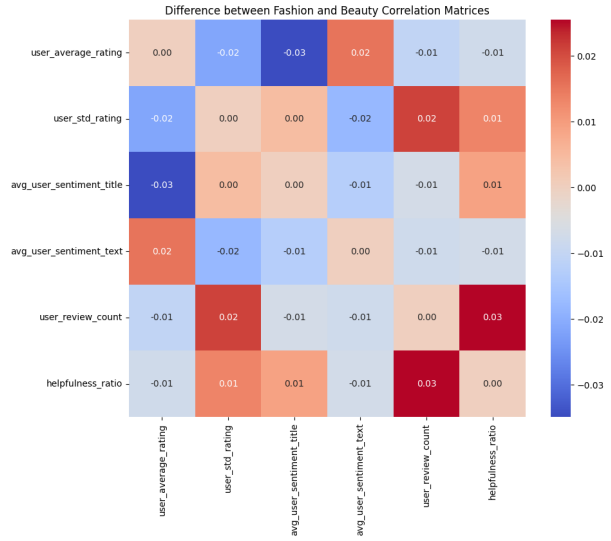

Figure 4: Difference between Fashion and Beauty Correlation Matrice

user_id	parent_asin	rating	user_average	user_std_rating	avg_user_sentiment_title	avg_user_sentiment_text	user_review_coun	helpfulness_ratio	average_rating	rating_number	title_word_count	description_sentimer
AEALTIXPOUZT	B003O6F9RQ	4	4	1.414213562	0.09	0.2052083333	4	3.75	4.1	2408	21	0.28
AG33HSQOUDE	B003O6F9RQ	1	3.75	1.892969449	0.4080357143	0.2496614583	4	0.5	4.1	2408	21	0.28
AGPD5YNSJVD	B003O6F9RQ	4	4.25	0.9574271078	0.08375	0.1405573593	4	0.75	4.1	2408	21	0.28

Figure 5: Sample of Training Data (user-product pair)

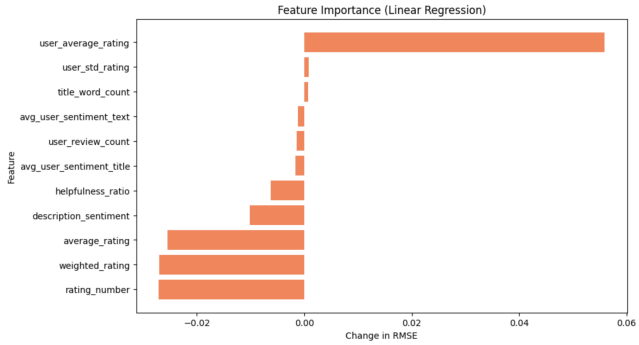

Figure 6: Influence of Features on Linear Regression Model

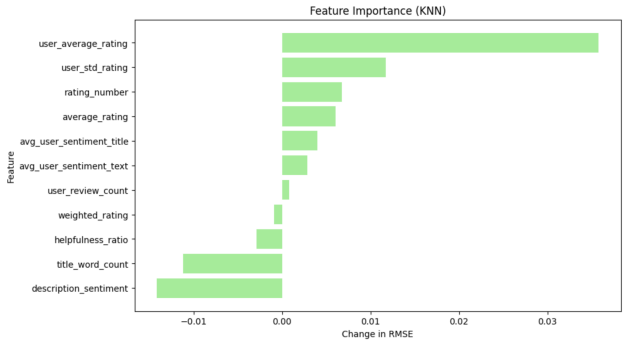

Figure 7: Influence of Features on KNN Model

7 Appendix

8 Contributions

Sue: Data Cleaning, Feature Engineering, Reformatting data to be ready for training, evaluated and provided advice on model performance optimization, wrote section 1-4 + created data visualizations in final paper.
Lea: Model training, testing, and evaluation. Wrote sections 4-6 + created data visualizations.

9 References/Bibliography

1. Linden, G., Smith, B., York, J. (2003). Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, 7(1), 76–80. https://doi.org/10.1109/MIC.2003.1167344

2. Martin, M. (2017). Predicting Ratings of Amazon Reviews: Techniques for Imbalanced Datasets. Master's Thesis, HEC-Ecole de gestion de l'Université de Liège. http://hdl.handle.net/2268.2/2707

3. Rendle, S. (2010). Factorization Machines. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)* (pp. 995–1000). https://doi.org/10.1109/ICDM.2010.127

4. Roy, D., Dutta, M. (2022). A Systematic Review and Research Perspective on Recommender Systems. *Journal of Big Data*, *9*, Article 59. https://doi.org/10.1186/s40537-022-00592-5

5. Yu, Y., Wang, W., Feng, Z., Xue, D. (2021). A Dual Augmented Two-Tower Model for Online Large-Scale Recommendation. In *Proceedings of DLP-KDD 2021*. https://doi.org/10.1145/1122445.1122456

6. McKinney, Wes. "Data Structures for Statistical Computing in Python." Proceedings of the 9th Python in Science Conference, edited by Stéfan van der Walt and Jarrod Millman, 2010, pp. 51-56.

7. Harris, Charles R., et al. "Array Programming with NumPy." Nature, vol. 585, no. 7825, 2020, pp. 357–362. Springer Science and Business Media LLC, https://doi.org/10.1038/s41586-020-2649-2.

8. Hug, Nicolas. "Surprise: A Python Library for Recommender Systems." Journal of Open Source Software, vol. 5, no. 52, 2020, p. 2174. The Open Journal, https://doi.org/10.21105/joss.02174.

9. Pedregosa, Fabian, et al. "Scikit-learn: Machine Learning in Python." Journal of Machine Learning Research, vol. 12, 2011, pp. 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html.