Amazon Product Recommendation System

Shreya Saxena

8/25/2024

Contents / Agenda

- Business Problem and Data Overview
- Exploratory Data Analysis
- Rank Based Model
- User-User Similarity-based Model
- Item-Item Similarity-based Model
- Matrix Factorization based Model
- Conclusion and Recommendations

Business Problem & Data Overview

Business Problem

Background

- With vast amounts of information growing globally, it has led to **information overload**
- There are way too many choices for a consumer to choose from
- Recommender system help consumers by providing personalized recommendations which is relevant to the specific user
- We are focused on Amazon, which is known to provide their users with a wide selection of recommendations by analyzing and predicting the shoppers preferences
    - One of the recommendation system Amazon uses is item to item collaborative filtering

Problem

- Build a recommendation system that recommends products to customers based on their previous ratings for other products

Data Overview

The Amazon dataset contains the following attributes:

- **userId:** Every user identified with a unique id
- **productId:** Every product identified with a unique id
- **Rating:** The rating of the corresponding product by the corresponding user
- **timestamp:** Time of the rating

Exploratory Data Analysis

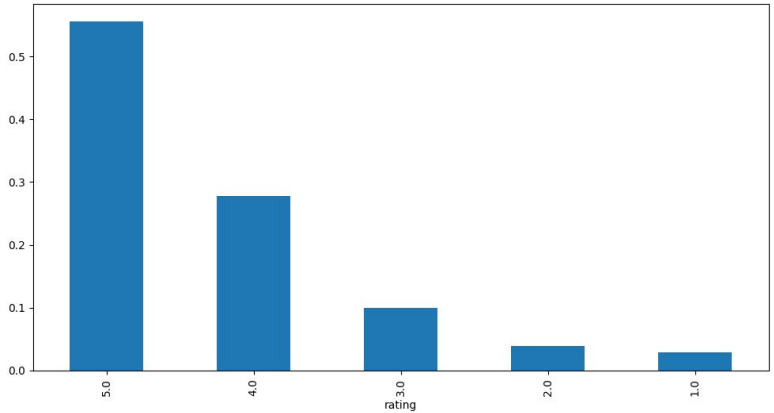
Exploratory Data Analysis

- The dataset contains 65290 rows
  - Number of observations has been reduced from 7,824,482 to 65,290 due to filtering out users who gave fewer than 50 ratings and products that have at least 5 ratings
  - This will help us focus on more reliable data
- The dataset contains 3 columns
  - userID, prodID, and the rating
- Data types
  - userID: object
  - prodID: object
  - Rating: float
- There are no missing values in any of the columns indicating the dataset is clean to proceed

Exploratory Data Analysis - Summary Statistics & Rating Distribution

- 65,290 ratings in the dataset (ranges from 1-5)
  - Average rating is 4.29
  - Standard deviation of 0.99 shows that some ratings are spread out around the mean and most are within 1 point of the mean (3.29 and 5.29)
  - Lowest rating given is a 1
  - 25% of the ratings are 4 or below
  - Median rating is 5, which means at least half of the ratings are a 5
  - 75% of the ratings are 5 or below
  - Highest rating given is 5
- Ratings seem to be skewed to the left with a low SD, meaning majority of the time, there is positive bias and users tend to like the products overall

- Plot shows ratings of 4 and 5 are most frequent, proving that most users like their products
- Ratings of 1, 2, 3 are less frequent (in order), indicating users rarely do not like the product



Exploratory Data Analysis - unique USERS and PRODUCTS

- The number of observations in the final data = 65290
- Number of unique USERS in Raw data = 1540
- Number of unique ITEMS in Raw data = 5689
- Even after filtering, each product is not rated by every user
- The highest number of ratings by a user is 295 which is far from the actual number of products present in the data. We can build a recommendation system to recommend products to users which they have not interacted with

Rank Based Model

Rank Based Model - Approach Taken

1. **Data Preparation:**
- Built using a dataset that contains user ratings for various products
- Calculate two key metrics for each product:
    - **Average Rating:** provides the mean of all ratings received by a product (satisfaction)
    - **Count of Ratings:** counts the number of ratings each product has received (popularity)
2. **Data Aggregation:**
- Group the dataset by product ID and compute the average rating and the count of ratings for each product
- The values are then combined into a new DataFrame
3. **Sorting & Filtering:**
- Sorted by their average rating in descending order to identify the top-rated products
- Filtered products that have received a minimum number of interactions
4. **Product Recommendation:**
- A function (top_n_products) was created to recommend the top N products based on their average rating and a minimum interaction threshold
- Balanced the popularity and quality

Recommendations Based on Popularity:

- The model was used to recommend the top 5 products with at least 50 and 100 interactions
- This approach helps to compare the impact of different interaction thresholds on the recommendations

Rank Based Model - Observations

Influence of Rating Count:

- This model was able to identify top rated products
- When the minimum interaction threshold is applied, it recommends the well rated and popular products

Impact of Minimum Interactions:

- When threshold is set lower (ex 50 interactions), the recommended products are more likely to be the less popular ones with higher ratings compared to when the threshold is high
- This might be due to the filtering of products with less ratings

Quality vs. Popularity:

- Recommends high quality products that are popular enough
- However, sometimes, it might not show the products that are still really popular with have less interactions

Capabilities:

- This model does not account for individual preferences, making it a less wanted model

User-User Similarity-based Model

User-User Similarity-based Model - Observations

Baseline Model

- **Root Mean Squared Error (RMSE):** 1.0012
  - Indicates that on average the predictions deviate from the actual ratings by approximately 1.0012 points on the rating scale
- **Precision:** 0.855
  - About 85.5% of the items recommended by the model are relevant to the users
- **Recall:** 0.858
  - The model recommends 85.8% of all relevant items
- **F1-Score:** 0.856
  - Indicates a fairly well-performing model, although it may still misidentify some items as relevant or irrelevant.
- **Prediction Behavior:**
  - For a user-product pair with a true rating (r_ui) of 5, the model estimated a rating of 3.40, showing a significant deviation
  - In some cases, the model was unable to make predictions due to insufficient neighbors, which suggests limitations in the default parameter settings

User-User Similarity-based Model

Optimized Model (Post Hyperparameter Tuning)

- **Root Mean Squared Error (RMSE):** 0.9510
  - The optimized model's predictions are more accurate, with a lower deviation of 0.9510 points from the actual ratings, indicating improved performance
- **Precision:** 0.849
  - The precision slightly decreased, meaning a small percentage of recommended items are less relevant compared to the baseline model. This could be due to a focus on increasing recall.
- **Recall:** 0.893
  - The recall improved significantly, with the model capturing 89.3% of all relevant items. This suggests that the tuning process made the model more effective at identifying relevant items.
- **F1-Score:** 0.87
  - Improved to 0.87, indicating that the model achieved a better balance between precision and recall after tuning.
- **Prediction Behavior:**
  - The estimated rating for the same user-product pair remained at 3.40, but the overall prediction accuracy improved
  - The model also maintained the same issue with insufficient neighbors for certain predictions, indicating that even post-tuning, some data sparsity issues persisted

User-User Similarity-based Model

Comparison of Performance:

- The optimized model has a lower RMSE compared to the baseline model → shows that the tuned model is more accurate in predicting ratings
- Post-tuning, the model's recall increased, which means it became better at finding all wanted items
- The overall F1-Score improved, meaning there is a better balance between identifying wanted items and avoiding the not wanted ones

User-User Similarity-based Model

Explanation of Differences in Predictions:

- The optimized model is more accurate because it has the lower RMSE
- It better estimates ratings closer to the true ratings
- The optimized model's higher recall means it is better at identifying relevant items that the user might like
- Although the precision slightly decreased, the trade-off is that the optimized model is more confident in suggesting a broader range of potentially relevant items, which might lead to slightly more false positives but ensures fewer relevant items are missed
- By optimizing hyperparameters such as k (number of neighbors) and using cosine similarity, the model could more effectively leverage user-user similarities, leading to more accurate predictions

Item-Item Similarity-based Model

Item-Item Similarity-based Model

Observations on Model Performance:

Default Parameters:

- **RMSE:** 0.9950
  - Indicates the average error in the model's predictions compared to the actual ratings, and a lower value signifies better performance
- **Precision:** 0.838
  - Indicates that 83.8% of the recommended items were relevant to the user.
- **Recall:** 0.845
  - Indicates that 84.5% of the relevant items were recommended to the users.
- **F1 Score:** 0.841
  - Suggests a well-balanced model in terms of both precision and recall.

Post Hyperparameter Tuning:

- **RMSE:** 0.9578
  - Shows that the tuned model is better at predicting ratings compared to the default model
- **Precision:** 0.839
  - Marginal increase in the accuracy of relevant recommendations.
- **Recall:** 0.88,
  - Indicating that the model now recommends a higher percentage of relevant items
- **F1 Score:** 0.859
  - Overall better balance between precision and recall in the tuned model.

Item-Item Similarity-based Model

Comparison of Predictions:

Default Model Predictions:

- For the user **A3LDPF5FMB782Z** who had already interacted with the product **1400501466**, the estimated rating was **4.27**, which is close to the actual rating of **5.00** → Indicates that the model was fairly accurate in predicting the rating for an already interacted product
- For the user **A34BZM6S9L7QI4** who had not interacted with the product **1400501466**, the model estimated a rating of **4.29** but flagged that it was "impossible" due to "not enough neighbors," meaning there wasn't sufficient data to make a reliable prediction.

Tuned Model Predictions:

- For the user **A3LDPF5FMB782Z** with the same product, the tuned model estimated the rating at **4.71**, which is even closer to the actual rating of **5.00**. This suggests that the tuned model offers more accurate predictions for interacted items.
- For the user **A34BZM6S9L7QI4** with the non-interacted product, the tuned model still estimated a rating of **4.29** and flagged it as "impossible" for the same reason as the default model, indicating that even with tuning, the lack of neighbor data remains a challenge.

Item-Item Similarity-based Model

- The hyperparameter tuning led to a noticeable improvement in model performance, particularly in terms of RMSE, recall, and F1 score, suggesting that the tuned model is better at both predicting ratings and identifying relevant items to recommend
- The predictions for interacted products improved in accuracy, while the challenge of predicting for non-interacted products remains, probably due to insufficient data or similar items.
- Overall, hyperparameter tuning refined the model's performance, making it more reliable and effective, especially for users with prior interactions with products.

Matrix Factorization Based Model

Matrix Factorization based Model Observation

Model Performance with Default Parameters:

- **RMSE:** 0.8882
- **Precision:** 0.853
- **Recall:** 0.88
- **F1-Score:** 0.866
- **Prediction for User "A3LDPF5FMB782Z" on Product "1400501466":** The estimated rating was 4.08 against an actual rating (r_ui) of 5.
- **Prediction for User "A34BZM6S9L7QI4" on Product "1400501466":** The estimated rating was 4.40, and there was no actual rating provided (r_ui=None).

Observations:

- The model with default parameters provided a decently good RMSE of 0.8882, indicating that the model's predictions are fairly accurate
- The predictions for individual users are close to what might be expected, but the estimated ratings are slightly lower than the actual rating for user "A3LDPF5FMB782Z."

Matrix Factorization based Model Observation

Model Performance Post Hyperparameter Tuning:

- **Best RMSE after tuning:** 0.8808
- **Precision:** 0.854
- **Recall:** 0.878
- **F1-Score:** 0.866
- **Optimized Parameters:** {'n_epochs': 20, 'lr_all': 0.01, 'reg_all': 0.2}
- **Prediction for User "A3LDPF5FMB782Z" on Product "1400501466":** The estimated rating was 4.13 against an actual rating (r_ui) of 5.
- **Prediction for User "A34BZM6S9L7QI4" on Product "1400501466":** The estimated rating was 4.22, with no actual rating provided (r_ui=None).

Observations:

- After tuning, the RMSE slightly decreased to 0.8808, showing a minor improvement in the prediction accuracy
- The precision and recall values remained stable, suggesting that the tuning did not significantly affect these metrics
- The estimated ratings for individual users were slightly higher in the optimized model compared to the default model
- The optimized model predicted 4.13 for user "A3LDPF5FMB782Z" compared to 4.08 with the default model, showing an improvement in the prediction's proximity to the actual rating

Matrix Factorization based Model Comparison

Default Model:

- User "A3LDPF5FMB782Z" was predicted to rate product "1400501466" as 4.08, slightly underestimating the actual rating of 5.
- User "A34BZM6S9L7QI4" had an estimated rating of 4.40 for the same product.

Optimized Model:

- The prediction for user "A3LDPF5FMB782Z" improved slightly to 4.13, getting closer to the actual rating of 5.
- The prediction for user "A34BZM6S9L7QI4" decreased to 4.22, which could indicate the regularization effect reducing the potential overfitting.

Conclusion & Recommendation

Conclusion

Summary:

- Rank based model is best for when customers care more about product popularity
- Collaborative FIltering Model is best for individual users that want personalized recommendations
- Matrix Factorization Model is best for larger crowd of users that want personalized recommendations

Model Selection:

- I would chose the Matrix Factorization Model because it can analyze large amounts of datasets to create personalized recommendations with little error, making it suitable for improving a big business like Amazon

Recommendations

- Use the SVD Model to personalize recommendations to increase customer engagement and ratings
- Use a combination of models (hybrid) that can balance popularity (rank based) and personalization (SVD)
- Implement ways to filter and provide recommendations for new products or customers
- Make sure to monitor the performance of the chosen model and do hyperparameter tuning to make sure it keeps adapting to customer behaviors

Great Learning

Happy Learning !