Implementing a Recommendation System with a collaborative filtering method

Sarah Hosseini 400222026

Shahid Beheshti University

1. Exploratory data analysis

1.1. Data Exploration

The dataset that we are going to work on is available at ‘https://www.kaggle.com/datasets/skillsmuggler/amazon-ratings/data’. We are going to present this report on the development and evaluation of a recommender system for this beauty product dataset. In today's digitally-driven age, personalized recommendations have become integral to enhancing user experiences and driving business growth. This report aims to provide a comprehensive analysis of the dataset, propose a recommender system architecture, and evaluate its performance using various metrics. By leveraging the vast amount of user and product information available, our recommender system aims to deliver tailored product recommendations that cater to individual preferences, ultimately fostering customer satisfaction and boosting sales. Through this report, we delve into the key aspects of our recommender system, including data preprocessing, algorithm selection, evaluation methodologies, and future recommendations for system improvement.

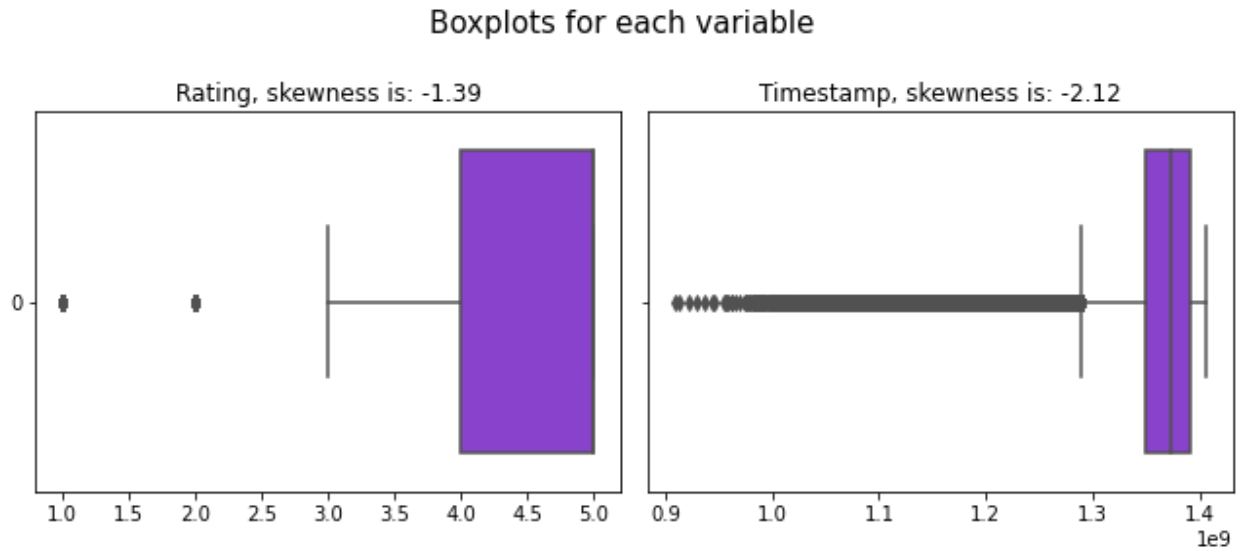
The dataset has a shape of (2023070, 4). The columns are 'UserId', 'ProductId', 'Rating', and 'Timestamp'. It has no duplicate or null values. The columns have a unique value count of:

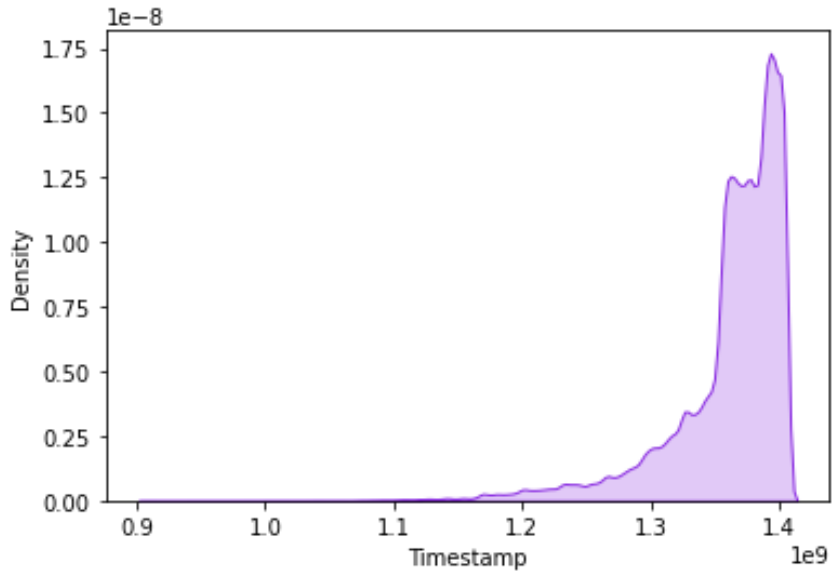
- ‘UserId’ = 1210271
- ‘ProductId’ = 249274
- ‘Rating’ = 5 (ranging from 1-5 based on customer satisfaction)
- ‘Timestamp’ = 4231.

‘Timestamp’ is the time that the rating was done in Unix time.

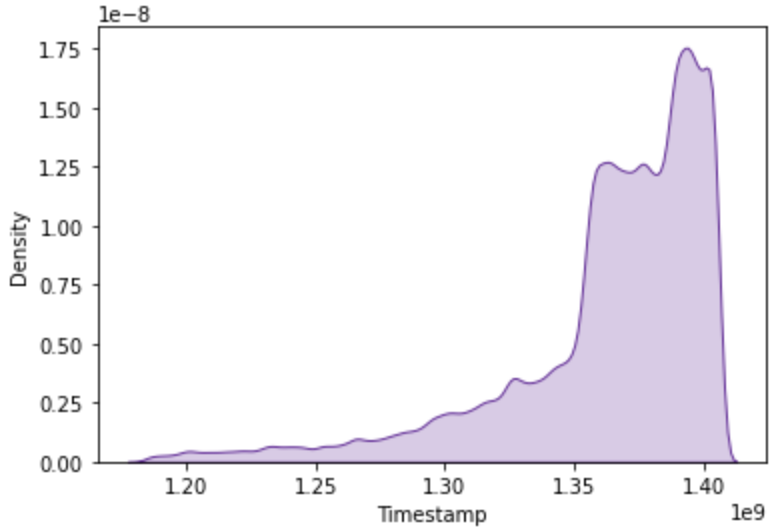
1.2. Univariate data analysis

The box plots below show the distribution of the two numerical variables. the mean of rating is 4.14 and the mean of the timestamp is 1.360389e+09, both variables being highly skewed.





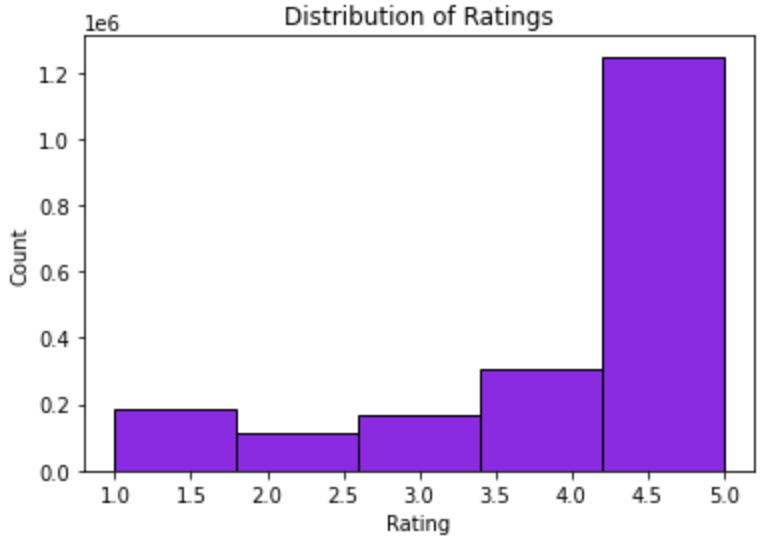
Then, we plotted another plot for timestamp:



According to that plot, most data has a timestamp of greater than 1.2. So, we plot again using the portion of data greater than its 1st percentile:

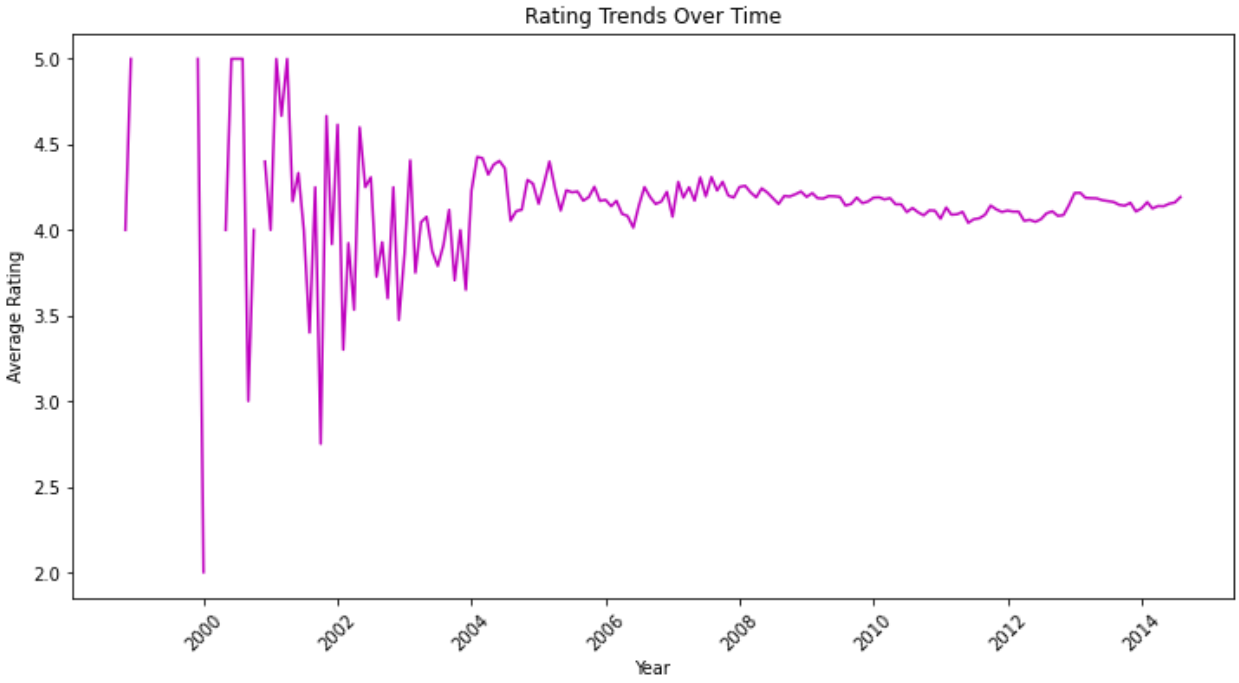
This shows that 99 percent of data has a timestamp of greater than 1.2e+09, which makes sense because this timestamp is for the years 2008 and later.

This plot demonstrates the distribution of ratings. It ca be seen that most ratings are 5 out of 5:

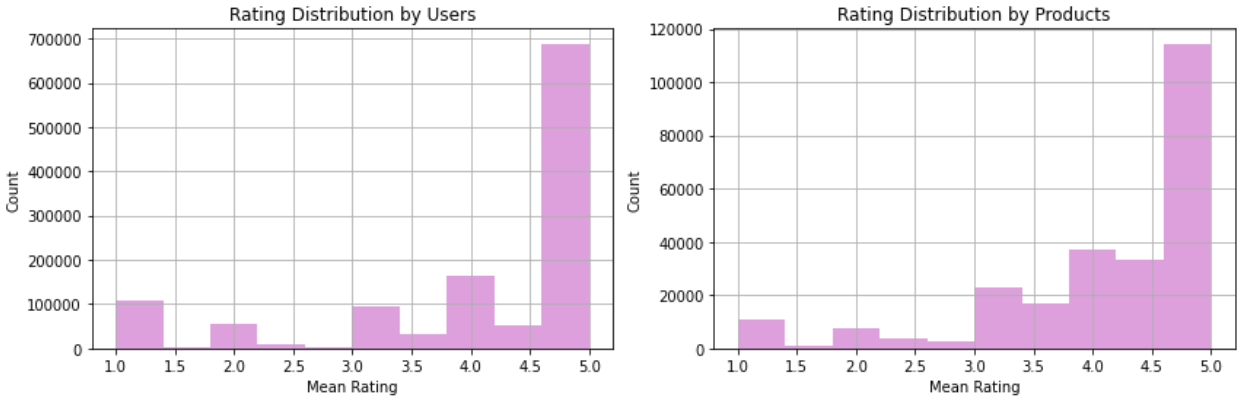


1.3. Multivariate data analysis

Here is another plot to visualize the rating trends over time:



These figures have first grouped the data based on users and products, then calculated the mean rating for them and plotted the frequency of each rating mean value:

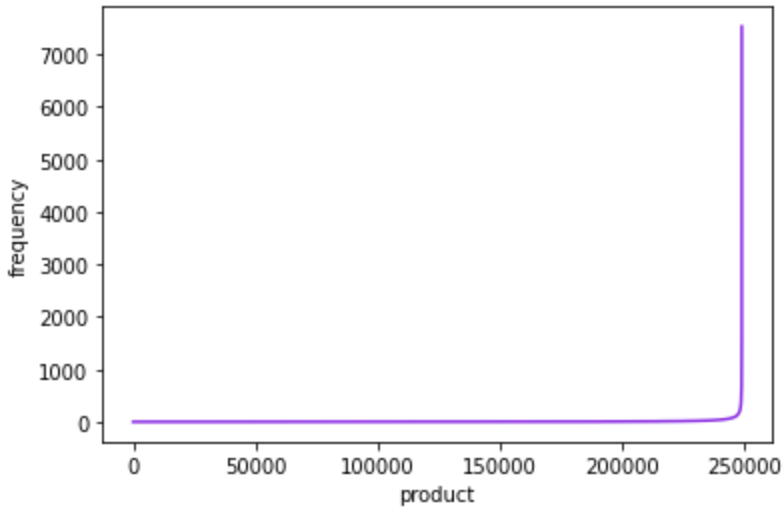


They show that most users generally give high ratings.

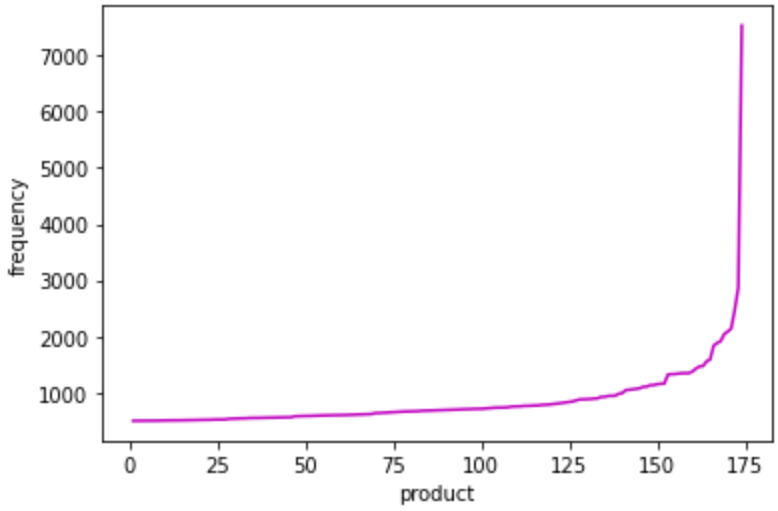
This is how many users gave each rating which proves that point:

```
1.0     183784
2.0     113034
3.0     169791
4.0     307740
5.0    1248721
```

We then group the products and plot their frequency in our data. In these plots, the products are sorted based on their frequency and put on the x-label but the IDs aren't there, instead we used numbers to show them:

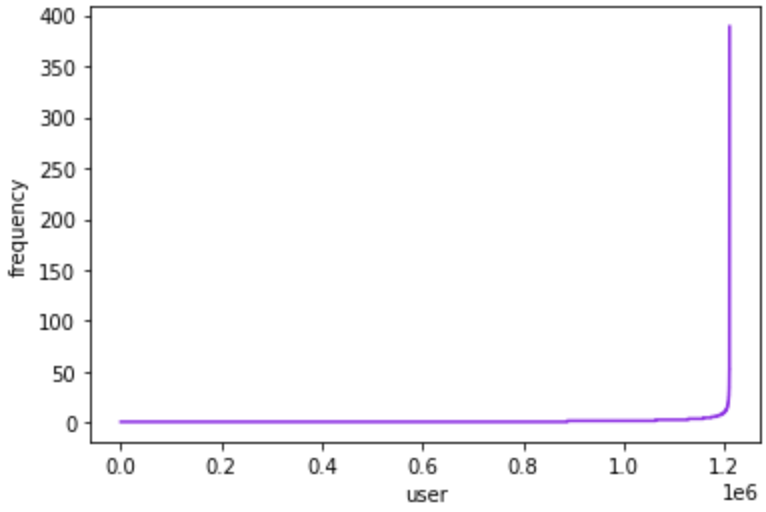


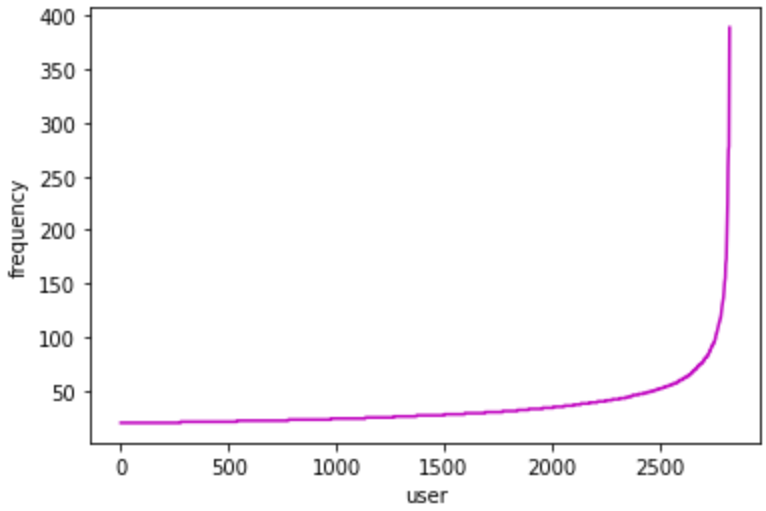
This plot shows that only a small portion of products have a high frequency. We plot again only those that are rated in the dataset more than 500 times:



Only 174 products are each rated more than 500 times.

The same goes for users:





Only 2826 users have each rated more than 20 times.

If we consider the shape of the rating matrix, we notice that this matrix has a large number of 301689093254 elements. Meanwhile, only 0.00067 percent of this matrix is filled, which means it is extremely sparse.

When working with recommendation systems, dealing with incredibly sparse matrices poses significant challenges. As the number of users and products grows, the matrix size increases

exponentially, making it memory-intensive to store the full matrix. However, the advantage of using matrix factorization is that it can implicitly represent the rating matrix by decomposing it into lower-dimensional factor matrices, reducing memory requirements. This approach allows the model to capture and represent user-item interactions implicitly, even with missing entries, enabling it to approximate the missing ratings based on the learned factor matrices. By leveraging matrix factorization, recommendation systems can efficiently handle sparse matrices and provide accurate recommendations on a global scale while mitigating memory constraints.

In the next section, we are going to train a neural network for this task. To prepare the data, we at last convert the non-numerical values to numerics using label encoding.

2. Model

We present an implementation of matrix factorization using PyTorch, a powerful deep learning framework. By leveraging user and item embeddings, the model can capture latent factors and make accurate predictions even in the presence of sparse data. The MatrixFactorization class offers a straightforward interface for training the model and generating recommendations based on the learned embeddings.

This implementation serves as a foundation for further exploration and customization in recommendation systems.

We prepared the data for training with a dataloader, because efficient data loading and transformation are crucial for our task.

In conclusion, the training process of our recommendation system using matrix factorization has yielded promising results. Over the course of 50 epochs, the model underwent iterative updates, gradually refining its latent factors for users and items. The training loss, starting at 17, steadily decreased over the epochs, reaching an impressive value of 0.362 by the 50th epoch. This significant reduction in loss indicates that our model has successfully learned and captured the underlying patterns and relationships within the data. Moving forward, these tuned latent factors can be utilized to generate accurate recommendations, enhancing user experiences and enabling effective decision-making in various domains. The success of this training process serves as a testament to the effectiveness of matrix factorization and its application in building robust recommendation systems.