Amazon Product Recommendation System

Tyler Rowe, Francisco Guzman, William McWhorter, Yash Pandey

Abstract
The goal of this project...

[1] *Luddy School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA*

Contents

1. Problem and Data Description

1.1 Problem Statement

With the rise of online shopping, product recommendations have become an essential feature for online shopping websites, as they assist users in discovering products that they may be interested in, leading to increased sales and customer satisfaction. As a result, data mining techniques have shown to be powerful tools for analyzing user behavior and creating personalized recommendations. However, analyzing Amazon user data in order to create an effective recommendation system comes with its own challenges such as data volume, complexity, and scalability and efficiency of the model.

Therefore, the objective of this project is to use data mining to create an effective model that uses Amazon product reviews to suggest other products to users based on previous reviews and ratings. This system will use collaborative filtering as well as clustering techniques to analyze user behavior patterns within the data to generate recommendations. We will be using metrics such as sum squared error, silhouette score, and other techniques for accuracy and cluster validity in order to evaluate our model and the effectiveness of the recommendations. People who will benefit from this project include both Amazon users who will receive personalized recommendations for products, as well as Amazon, who will see more engaged users and an increase in product sales.

The data source we will be using is Amazon product review data[1][2], which contains information about the product and review such as product category, rank, the written review, rating given by the user, similar items, and more. This data includes many valuable attributes that will be crucial in determining recommendations for other products for each user. However, the project may also face challenges within the data that will need to be addressed before modeling. This includes effective data preprocessing and exploration. To face these challenges, we will employ techniques such as feature engineering, dimensionality reduction, and data transformation. This will ensure a multitude of things such as verifying that we are using only the most highly correlated and important features to make recommendations, as well as making sure the data is as small and concise as possible to speed up the efficiency of the model and reduce noise from unimportant attributes.

In summary, the project aims to develop an efficient and accurate recommendation model for Amazon based on their product and user data. Our model will use data mining algorithms and techniques to provide personalized recommendations to customers to increase user engagement and product sales.

1.2 Data Description

The data[1] that will be used to test and train our recommendation system is Amazon review data[1]. This dataset[1] contains more than 233.1 million product reviews from Amazon, ranging from May 1996 to October 2018. The data[1] is partitioned into product categories: Amazon Fashion, All Beauty, Appliances, Arts, Crafts, and Sewing, Automotive, Books, CDs and Vinyl, Cell Phones and Accessories, Clothing Shoes and Jewelry, Digital Music, Electronics, Gift Cards, Grocery and Gourmet Food, Home and Kitchen, Industrial and Scientific, Kindle Store, Luxury Beauty, Magazine Subscriptions, Movies and TV, Musical Instruments, Office Products, Patio, Lawn and Garden, Prime Pantry, Software, Sports and Outdoors, Tools and Home Improvement, Toys and Games, and Video Games.

For our model creation and data experimentation, we will be using a subset of the massive dataset[1] that resolves some data sparsity issues by only including the data in which all users and items have at least 5 reviews, which includes 75.26 million product reviews from Amazon, about a third as much as the original dataset[1].

Each review contains twelve attributes: overall: numerical, ordinal (rating out of 5), verified: categorical, nominal (whether or not the review was a verified purchase), reviewTime: categorical, nominal (date and time of the review), reviewerID: categorical, nominal (unique ID for the reviewer), asin: categorical, nominal (unique ID for the product), reviewerName: categorical, nominal (name of the reviewer), reviewText: categorical, nominal (text of the review), summary: categorical, nominal (summary of the review text), unixReviewTime: numerical, interval (Unix timestamp of the review), vote: numerical, ratio (number of helpful votes for the review), style: categorical, nominal (dictionary of product metadata), image: categorical, nominal (image posted by the user).

The attributes that are not vital to the model and will not be used in our experiments include image, unixReviewTime, reviewerName, and vote. These attributes are not important to predict products that the customer may purchase because they either cannot be used in the model (image and unixReviewTime) or they have no information about the customers opinion on the product (reviewerName and vote). Also, we will only be using datapoints in which the user is verified, i.e. confirmed to have purchased the product. This reduces the size of the data[1] to around 56 million datapoints with seven attributes.

The data[1] has little issues with data quality and missing values, as the information is pulled from Amazon website reviews, which include certain information for each review. The attributes that do include missing values are image, vote, and style, since these are not required in an Amazon review. While image and vote will not be used in our model, the missing values in style will be dealt with per product, determining if the style is a important aspect of the product and the review. Also, there are no duplicates in the dataset[1].

There are no ethical concerns regarding the dataset[1] as the it does not contain any personal information other than the reviewer's name, which will not be used in any experiments or the product recommendation model.

2. Data Preprocessing & Exploratory Data Analysis

By reading through the entire dataset[1], we see that we have a total of around 80 million reviews of products. However, for this product, we will only consider the reviews in which the user is verified, i.e. the user is confirmed to have bought the product they reviewed. By doing this, we reduce the chance of having faulty reviews that might be generated by bots or others. In doing this, we reduce the overall dataset[1] from around 80 million reviews to roughly 58 million reviews. For our use case, some of the attributes such as "reviewerName", "reviewTime" are not needed, so we can remove those. Also by downcasting some of our attributes, we reduce the overall memory needed to hold the dataframe from around 55 GB to only 22 GB.

2.1 Handling Duplicates and Missing Values

For this project, we are using the 5-core dataset[1] for the Amazon data. This means that each reviewer of a given product has at least 5 products in which they have reviewed. This makes sure that we do not have any missing values in our dataset[1] which makes pre-processing much simpler. However, as expected, there are duplicate product ID's that we have to take care of. For example, two vendors could be selling the same product, but since they are different vendors, they do not get assigned the same "asin" product ID. Thankfully, the creator of this amazon dataset also provides a duplicates text file in which we can see which product ID's point towards the same product. By applying a map from these duplicate products, we can reduce the number of unique product ID's which helps later on when creating our collaborative filtering matrix.

2.2 Exploratory Data Analysis

To get an overall feel for the dataset[1], we graphed the Top-7 products bought for each category.

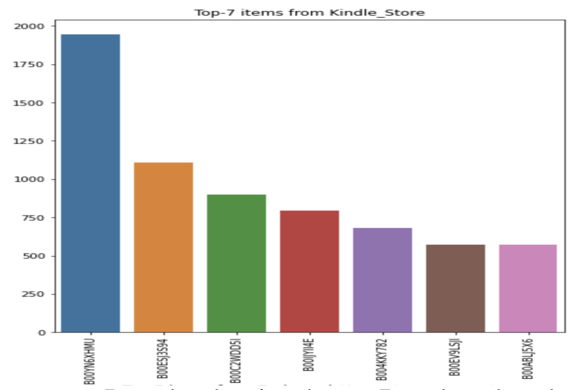

Figure 1. Box plot showing the top 7 products bought in the Kindle Store category

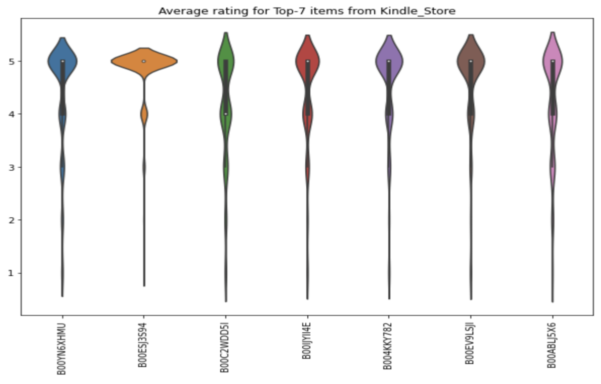

Figure 2. Violin plot showing rating distribution of the top-7 products bought in the Kindle Store category

As we can see from Figures (1) and (2), the most common bought product from the Kindle Store category had a product ID of "B00YN6XHMU", which if we go on amazon, we can see that this ID references the “Fifty Shades of Grey” book which was quite a famous book at the time it was published. From the violin plot, we can also see that most of the ratings for this book was around a 5/5 star rating.

In addition to the top items of each category, distributions of the overall ratings in each category were also examined. These distributions can be found in Figures(3)-(6).

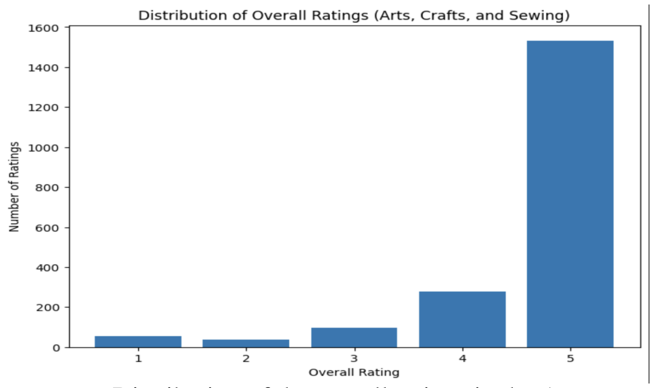

Figure 3. Distribution of the overall ratings in the Arts, Crafts, and Sewing product category

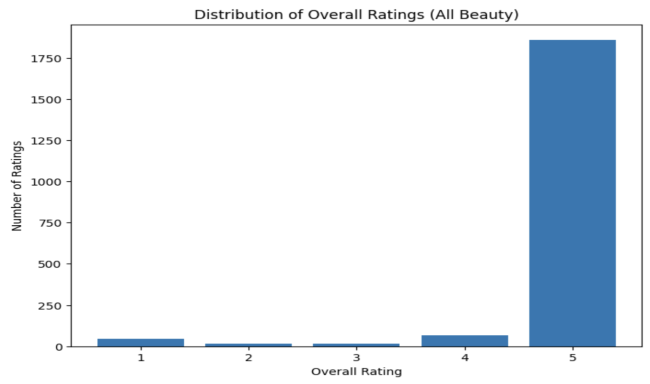

Figure 4. Distribution of the overall ratings in the Beauty product category

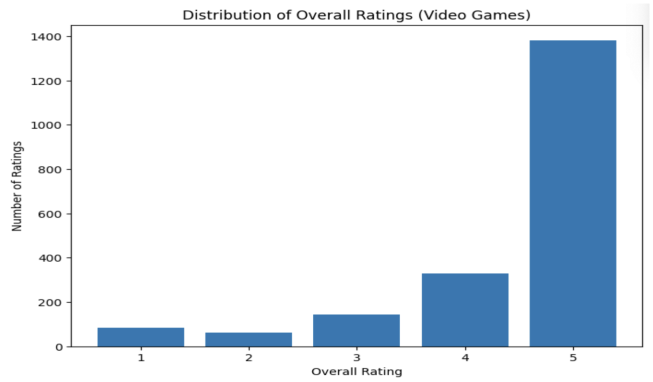

Figure 5. Distribution of the overall ratings in the Video Games product category

Another method that was employed to explore the data was applying various clustering techniques on the `reviewText` data. The comments were sampled such that one dataset contained equal numbers of each overall rating and a second dataset contained equal numbers of each overall rating for each category. The title, brand, and product description were then concatenated and they along with reviews were separately converted into TF-IDF values, as well as count frequencies. K-means clustering was applied to the TF-IDF data and Latent Dirichlet Allocation (LDA) to the word count data. As can be seen in Figure (7), using the elbow method of determining clusters does not have any significance, as there is no elbow in the data. Similar results were found using the further subdivided data including `category` and also using LDA over the counts instead. Nonetheless, word clouds showing the most common words after using equal counts of each overall rating for two total clusters can be found in Figures (8) and (9).

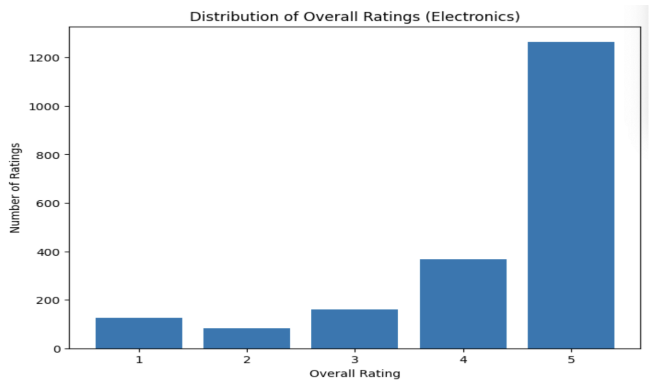

Figure 6. Distribution of the overall ratings in the Electronics product category

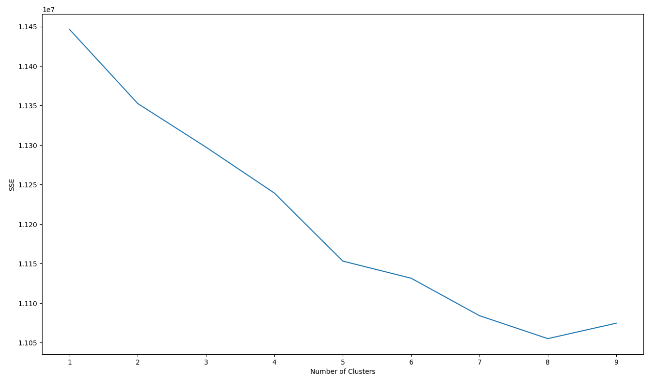

Figure 7. Distribution of the SSE as the number of clusters in K-Means was varied.

As can be seen in the word clouds, the clusters did not yield any significant information, as the first cluster primarily included words based around books, movies, and good quality and price, whereas the second cluster was more centered on good quality books and stories. Similar results were obtained with the other subdivisions of the data and increased number of clusters, as well as with LDA. The reason words focused on books appeared so frequently was likely due to the books

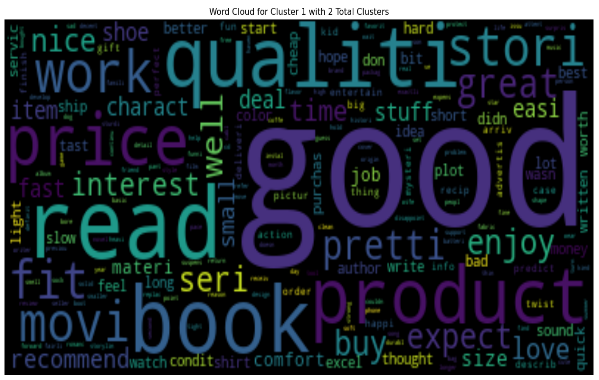

Figure 8. Word cloud showing the most common words for the first cluster of two when using K-Means on data with equal numbers of each overall rating.

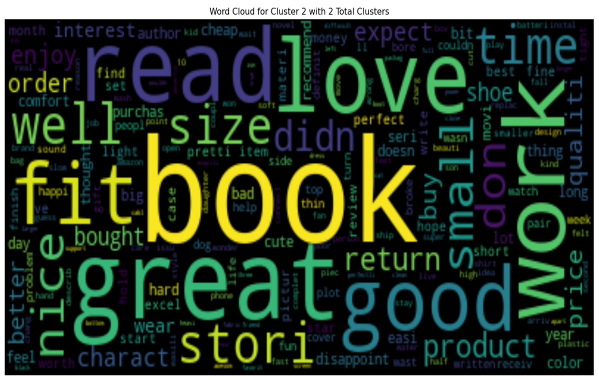

Figure 9. Word cloud showing the most common words for the second cluster of two when using K-Means on data with equal numbers of each overall rating.

dataset containing nearly six times as many entries as the next largest dataset, nearing slightly less than half the total entries.

3. Algorithm and Methodology

The 5-core datasets from the Amazon review dataset were used to ensure each entry had five reviews and each user had five reviews to create a collaborative filter and a content-based filter for product recommendations. Let U, P, and V be the set of users, the set of products, and the encountered vocabulary, respectively. The ratings for each product in the 5-core datasets were converted into a sparse $|U| \times |P|$ representation, and the product titles were converted into a sparse $|P| \times |V|$ TF-IDF representation.

To create the collaborative filter, the rating matrix was transformed into a $|P| \times |P|$ similarity matrix, using the cosine similarity as the similarity measure. In order to generate recommendations for a specified, predefined user U_i, the dot product between the product ratings of U_i with the cosine similarity values for all products. These values were then scaled down by the sum of similarity scores to obtain an estimated rating for all products for user U_i. The list of all products was then sorted, descending by predicted rating, and the top n values, n chosen by a user, are outputted as recommendations. If, instead, predictions are being made by a specified list of product + rating pairs and not a user in the database, the same calculations are performed, with the user rating lookup being replaced by the construction of a product review vector. The accuracy of predictions was measured by calculating the root mean-squared error (RMSE) of product ratings with their predicted values.

The content-based filter was constructed by transforming the TF-IDF matrix into a $|P| \times |P|$ similarity matrix. To determine the optimal similarity measure, the Euclidean distance, the Manhattan distance, and the cosine similarity were all utilized and compared using the RMSE. The measure that minimized the RMSE was chosen, and was found to be the cosine similarity measure. Then, for any specified product, the products with the highest cosine similarity were returned.

4. Experiments and Results

For this project, we used two widely known recommendation algorithms: A Item-based Collaborative Filtering algorithm, and a Content-Based Filtering algorithm. In order to evaluate the performance of our models, we calculated the Root Mean Square Error from the predicted ratings to the actual ratings for a given user.

4.0.1 Item-Based Collaborative Filtering

Since our data very large, calculating the RMSE for each and every user was not feasible so we opted towards randomly sampling around 5,000 users from our dataset. During testing, we saw that the best distance metric for our given task was the cosine similarity metric, and after predicting the ratings for these 5,000 users, we ended up with a RMSE value of around 0.1166. This here indicates that our model is able to correctly predict the wants and preferences for our users.

4.0.2 Content-based Filtering

We also implemented Content-based Filtering by using the metadata of the items such as the product title, and description in order to generate recommendations as outlined in the previous section. While we were not able to compute an RMSE for the Content-based Filtering approach, we were able to qualitatively evaluate the recommendations generated by this model, and compared them with the item-based filtering model. Overall, what we noticed was that in general, Item-based Collaborative filtering produced recommendations that are more so aligned with a given product/purchase history of a given user than Content-Based Filtering.

With the content-based filtering algorithm we were able to create a tool in which you could be presented with a list of products and select any given product. Once selected, five product recommendations were curated according to the cosine similarity of the description, brand, and title of the product. These similar products would be checked to make sure they are different products as well as not the product that was selected. This tool can be used to find similar products to any product from any category in the Amazon dataset.

In order to introduce our model to new data than the data it was first trained with, a web scraper tool was made to scrape data from any Amazon product url. This tool could be then be used in the same way as the previous tool, by recommending five products that are similar to the product given.

Code: https://github.iu.edu/fjguzman/DataMining_Final
Web App: https://amazon-final-ribknpuqoq-uc.a.run.app/

5. Summary and Conclusions

In today's highly competitive online shopping world, providing personalized recommendations is crucial to attract customers and keep them engaged. By curating product recommendations based on customer shopping history, businesses can increase customer satisfaction and loyalty, ultimately leading to increased sales. Our recommendation system was designed to address this need by leveraging customer data, such as purchase and viewing history and ratings to generate accurate recommendations. With the growing demand for personalized recommendations for online shoppers, our system provides a competitive advantage to businesses that adopt it.

Our objective was to create a recommendation system that can accurately and effectively generate recommendations for customers based on their product purchase/viewing history as well as their ratings and reviews. We achieved this goal by utilizing multiple algorithms that were able to provide accurate product recommendations for customers. We were able to use two significantly effective algorithms that are able to produce recommendations for customers depending on their product interests. The collaborative filtering algorithm was able to utilize the ratings of the user and other users in order to determine the products that the user would most likely be interested in. On the other hand, the content based filtering algorithm was able to provide n recommendations for customers based on the similarity of other products title, brand, and description. The performance and predictions of both of these models is substantial and enough to warrant our project as a success.

To build this project into something even more effective, we could need to take in more data such as user interaction data while on the shopping website. With user interaction data, we could track the amount of time a user spent looking at certain items and what recommendations they clicked on, as well as if it converted into a purchase. These changes would be able to take our algorithm into the next stages as the recommendations could be more personalized and generated with more data to make more informed recommendations.

6. Data

`https://nijianmo.github.io/amazon/index.html#subsets`[1]

References

[1] Jianmo Ni. Amazon product data. `https://nijianmo.github.io/amazon/index.html`, 2018.

[2] Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197. Association for Computational Linguistics, 2019.