Data Analytics Project : Recommendation system
Group - 8

Done by : Rahul Shah, Metehan Dündar

Aim : To do the pre-analysis of data and describe the dataset based on the dataset given and compare their computation as well as build the recommendation system given by any content-based or collaborative-based

---

1 Description of the dataset :

- The first dataset under the name of ratings has columns that are
  1. **user ID :** Defines the users by their ID.
  2. **Item ID or asin :** Defines the instruments or related items by their ID.
  3. **ratings :** Defines the ratings by the users to respective items.
  4. **Timestamp :** Defines the date and time during which the rating was recorded.
- The second dataset under the name of the meta-musical instrument is all about item URL (as in images), salesRank, categories, related, brand and description.

2 Detailed expansion of rating dataset :

2.1 Ratings and its frequency (number of users) :

- Here we have described the relation of ratings to the number of users or say frequency as shown in the graph.

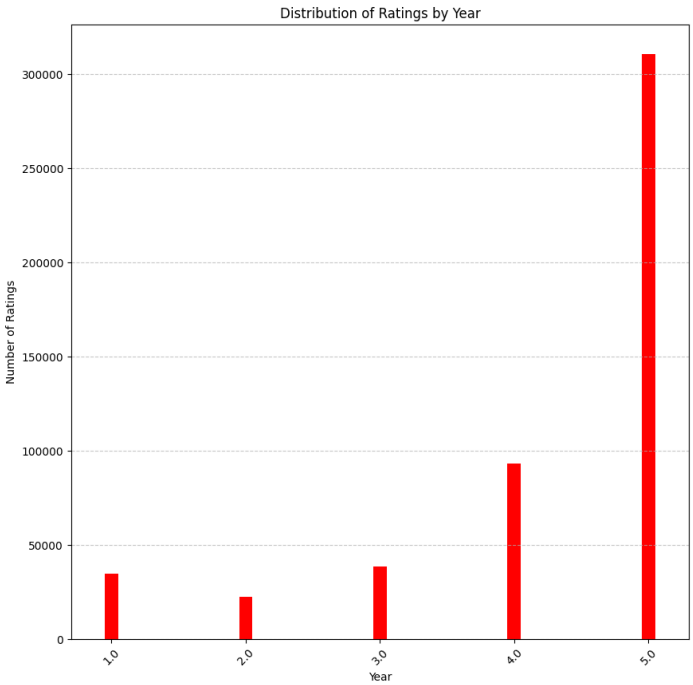

Figure 1: Ratings vs Total number of ratings

2.2 Number of users and their total ratings :

- In this graph we have created the graph having the total number of users rated items for more than some threshold (above or equal to certain no of ratings) as defined in the code.

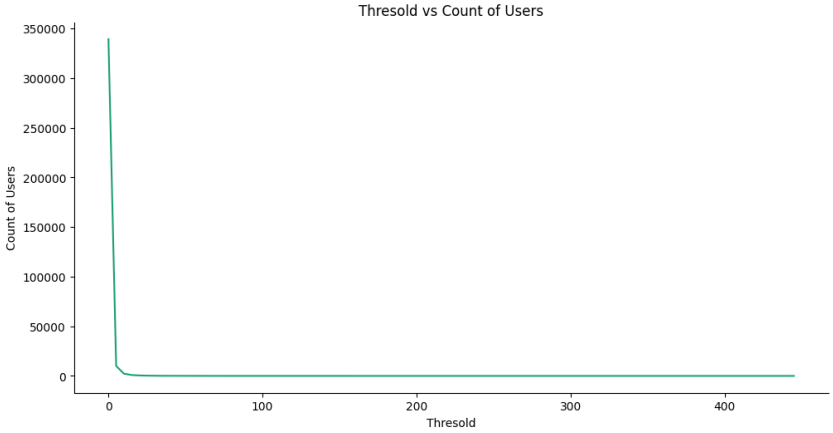

Figure 2: Total no of users vs number of ratings by user

2.3 Number of ratings done during each year :

- Here we have created a graph that sketches the relation between the number of ratings done each year in the form of frequency

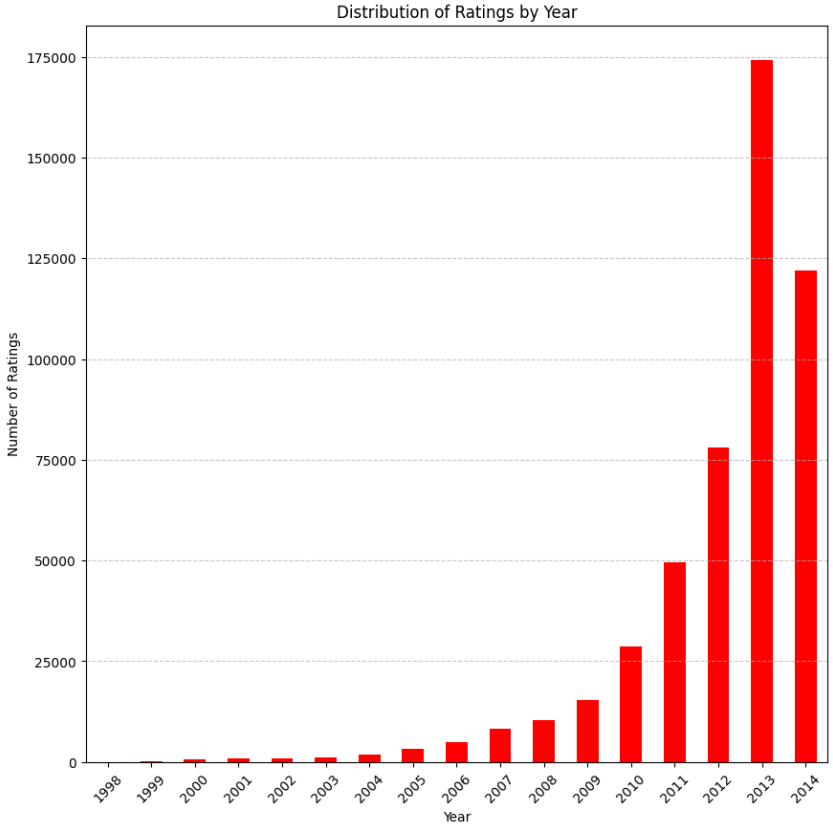

Figure 3: No of ratings vs years

3 Detailed expansion of meta-musical instrument dataset :

3.1 Brand names and its counts :

- Since, in this data, there are many NaN values (i.e. not integer or any string). So, excluding those NaN values created a graph that denotes the relationship between brands and the number of brands in the dataset.

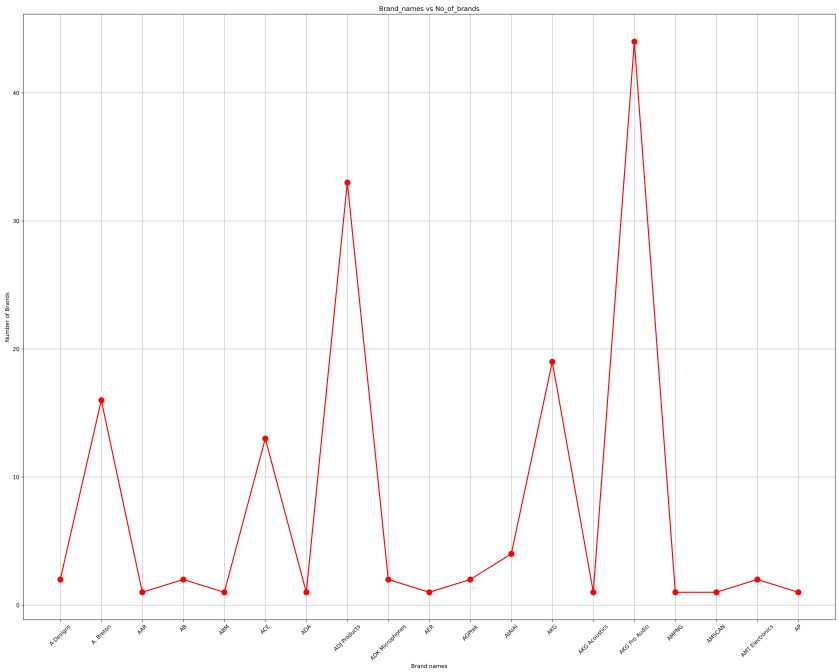

Figure 4: Brands VS Total number of the brands

3.2 Understanding related options to Item IDs :

- Since, in this data, there are many NaN values (i.e. not integer or any string). So, excluding those NaN values created a graph that denotes the relationship between Item ID and lengths of related options.
- Here what related options are the options may user has also bought or bought after watch or bought together or also viewed and represented those list of each of relation as in length.

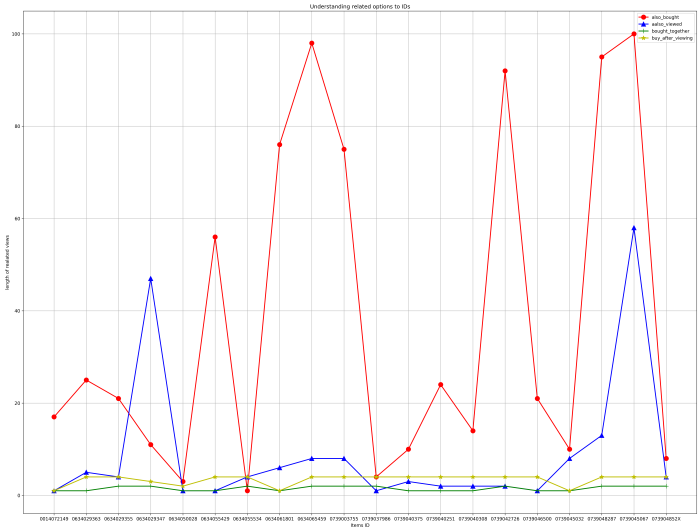

Figure 5: Item Ids vs length of each related view

3.3 Sales rank according to category :

- Since, in this data, there are many NaN values (i.e. not integer or any string). So, excluding those NaN values created a graph that denotes the relationship of sales rank according to categories for each of the items (IDs)

- Here what sales-rank represents is the rank of the particular item under the specific category
  [Note : A few of the categories are illustrated for better visualization of the graph]

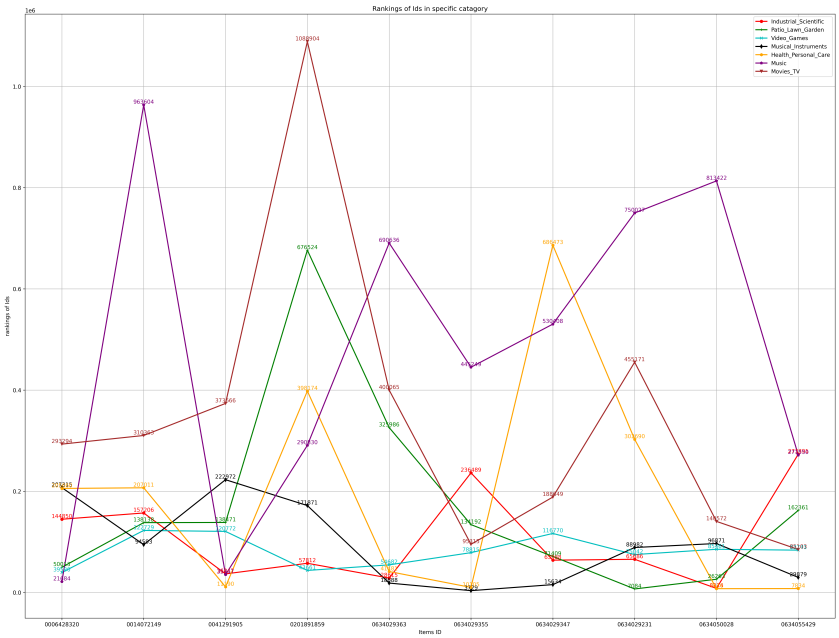

Figure 6: Item Ids vs rankings

3.4 Subcategories under particular category :

- Since, in this data, there are many NaN values (i.e. not integer or any string). So, excluding those NaN values created a graph that denotes the number of subcategories under particular categories.
  [Note : This is only for detailed analysis of the dataset but on working with systems we use it (i.e. categories – subcategories) as one.]

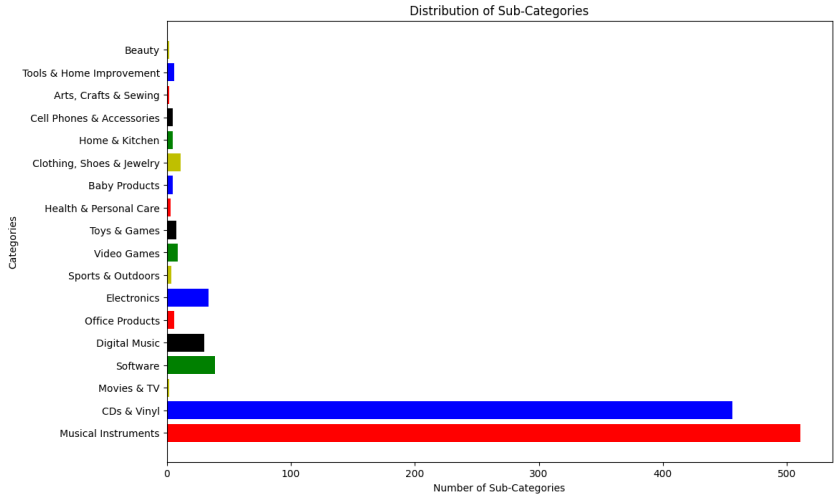

Figure 7: categories vs number of sub-categories

4 Preprocessing the Data:

- After reading and converting the JSON file to DataFrame, we figured out most of the data is missing as below;

```
asin            0.00
title          11.38
price          18.65
imUrl           0.08
salesRank       2.79
categories      0.00
related        19.50
brand          61.02
description    16.23
dtype: float64
```

Figure 8: Percentages of missing values in metadata of musical instruments

- Then we decided to drop the columns which are **price, imUrl, salesRank, related, brand, description**. We didn't remove the column "title" because that column would be useful when our recommender system gives recommendations. Of course, we remove the rows whose titles are missing, NaN. (Under name of Drop_df)

- The categories column is processed so that we can divide each category. Then we get 1020 unique categories. and with the help of unique categories evaluated categories column and created DataFrame (under name of New_dropped_dataframe for evaluation of heatmap.

- At last, we have created a heatmap that relates the mean of the ratings via grouping by month and categories.

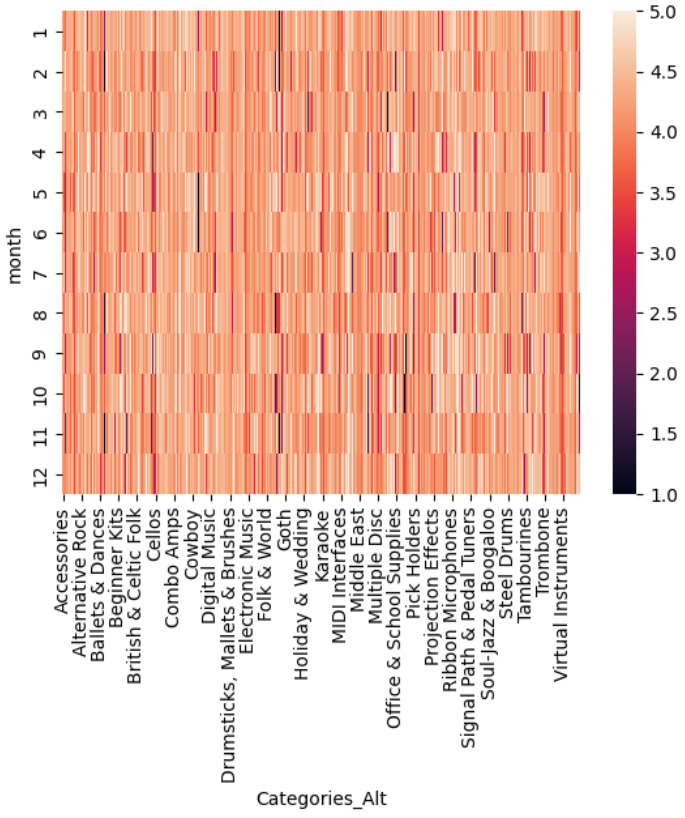

Figure 9: Heat-map for ratings

5 Content Based Recommender System:

- Since we are tackling the content-based filtering with the categories we have used sklearn library for betterment of the recommendation model and since we are dealing with words instead of integers for evaluation of cosine similarity we have used vectorizer for conversion from word to vector format and that's how we build recommendation system based on categories. Of course, using similarity scores.
  [Illustration can be seen in code]

6 Collaborative Recommender System:

- In the case of collaborative filtering, we are dealing with the item-item recommendation system. So, we have created a sparse matrix using the batch from the train set [Since the dataset is too large in train set] for item - item based system, and by evaluating the similarity of items we have created recommendation system.
  [Illustration can be seen in code]

7 Comparison & Evaluation of Models:

- We used SVD, Baseline, and KNN, they are collaborative filtering methods.
- All three models have comparable RMSE values, indicating that they perform similarly well in predicting user ratings.
- KNN on the shortened dataset shows a little lower RMSE, implying that it may be somewhat superior at predicting ratings in this particular scenario.
- The RMSE differences are minimal, thus while KNN has a tiny advantage, the SVD and BaselineOnly models are equally great performers.