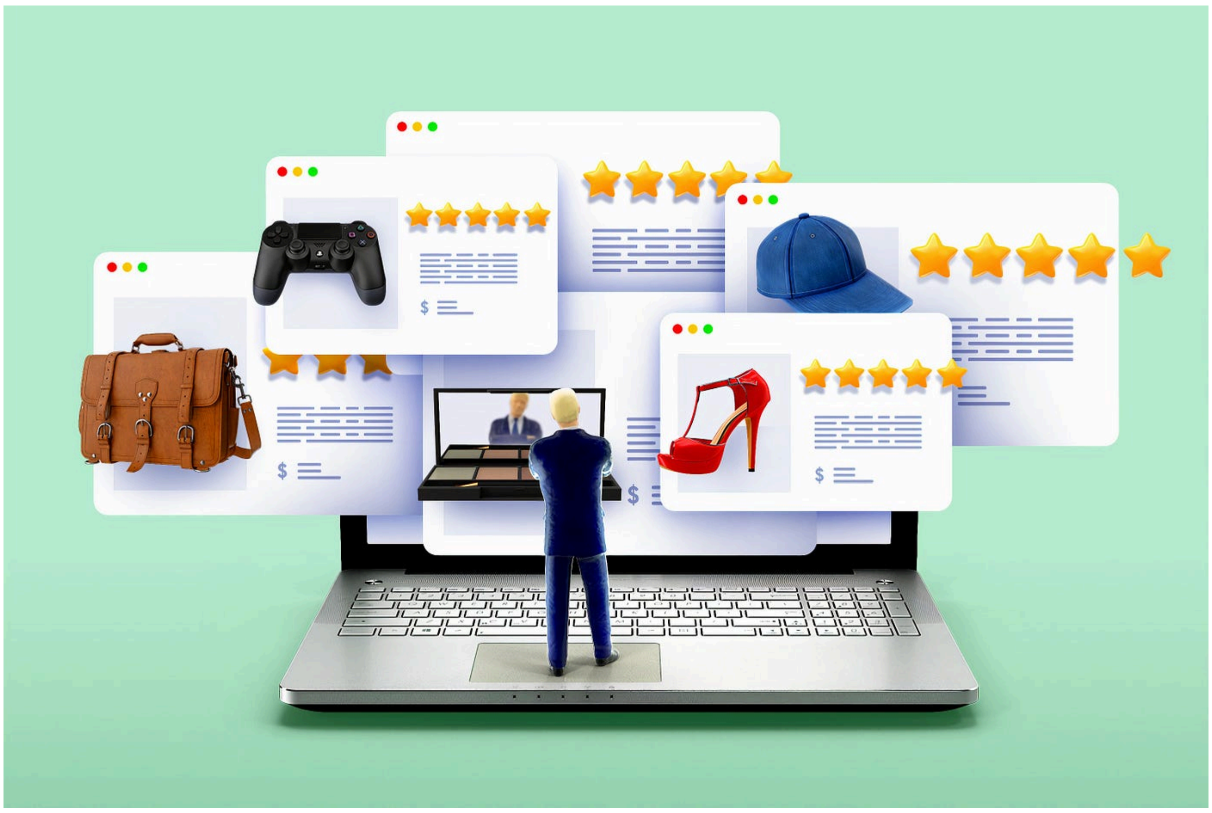
Task - 1

Recommendation System Using Product Rating

A recommendation engine is a system or algorithm that analyses user data and provides personalized suggestions or recommendations for items or content the user may be interested in. These recommendations are based on various factors such as the user's preferences, historical behavior, demographic information, and similarities to other users.

Recommendation engines are commonly used in e-commerce, OTT platforms, social media, and other online services to enhance user experience and engagement. They help users discover new products, movies, music, articles, or any other items that align with their interests.



The recommender system creates a similarity between the user and items and exploits the similarity between the user/item to make recommendations.

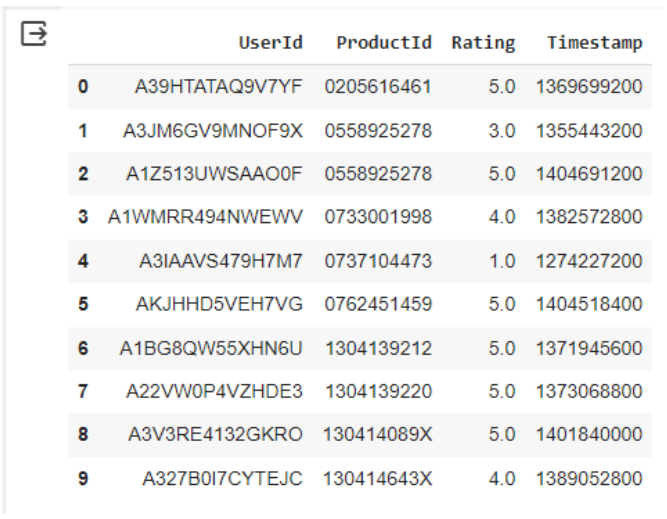
Problem Statement

Amazon relies heavily on a Recommendation engine that reviews customer ratings and purchase history to recommend items and improve sales. This dataset is related to over 2 Million customer reviews and ratings of beauty-related products sold on their website.

These are the Steps that I perform for this model.

Step 1: Download the dataset.

- In this, download the dataset called “amazon-rating” from the Kaggle website in which the dataset contains 4 attributes:



	UserId	ProductId	Rating	Timestamp
0	A39HTATAQ9V7YF	0205616461	5.0	1369699200
1	A3JM6GV9MNOF9X	0558925278	3.0	1355443200
2	A1Z513UWSAAO0F	0558925278	5.0	1404691200
3	A1WMRR494NWEWV	0733001998	4.0	1382572800
4	A3IAAVS479H7M7	0737104473	1.0	1274227200
5	AKJHHD5VEH7VG	0762451459	5.0	1404518400
6	A1BG8QW55XHN6U	1304139212	5.0	1371945600
7	A22VW0P4VZHDE3	1304139220	5.0	1373068800
8	A3V3RE4132GKRO	130414089X	5.0	1401840000
9	A327B0I7CYTEJC	130414643X	4.0	1389052800

UserId: Every user is identified with a unique ID.

ProductId: Every product is identified with a unique ID.

Rating: Rating of the corresponding product by the corresponding user.

Timestamp: Time of the rating.

Step 2: Import Required Library.

- Import the required libraries like:

Pandas: Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data.

Seaborn: Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas' data structures.

NumPy: NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays.

Matplotlib: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

Sklearn: Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms.

Decomposition.Truncated SVD: Truncated SVD is a class in sklearn.decomposition module of Scikit-learn, a popular machine-learning library in Python. It's used for performing dimensionality reduction using Singular Value Decomposition (SVD). This method is particularly useful for reducing the dimensionality of sparse data.

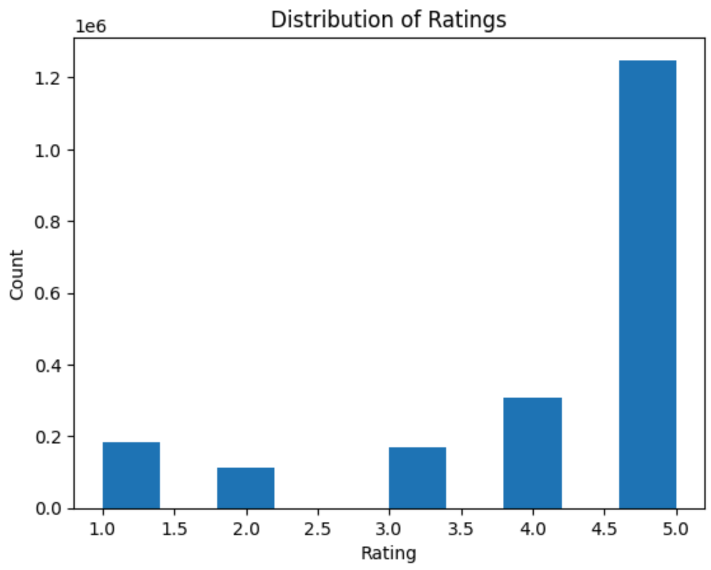
Step 3: Load the dataset.

- Load the dataset using “pd.read_csv” used to read the comma-separated values (CSV) into the data frame.

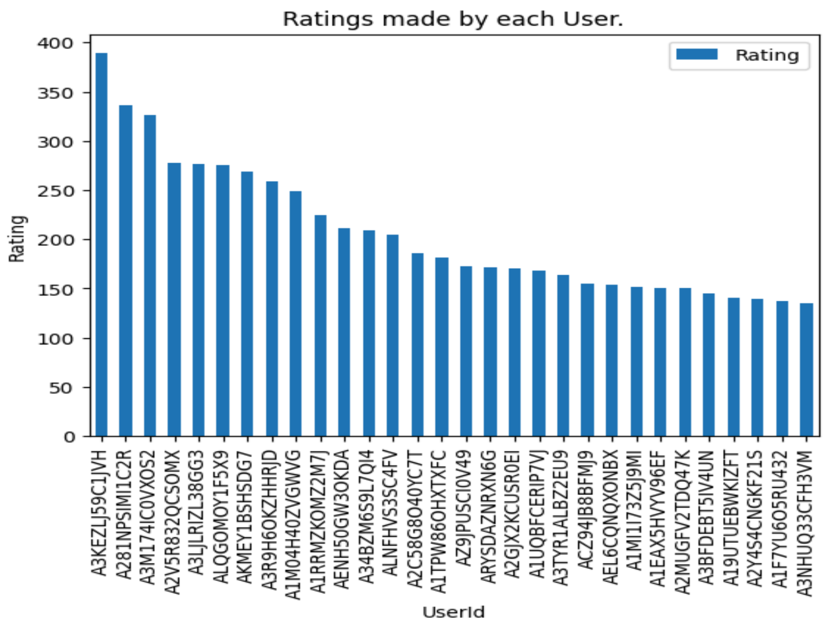
Step 4: Drop the irrelevant column.

- In the dataset, the Timestamp column is not useful for this model so drop the column using the “drop” keyword.

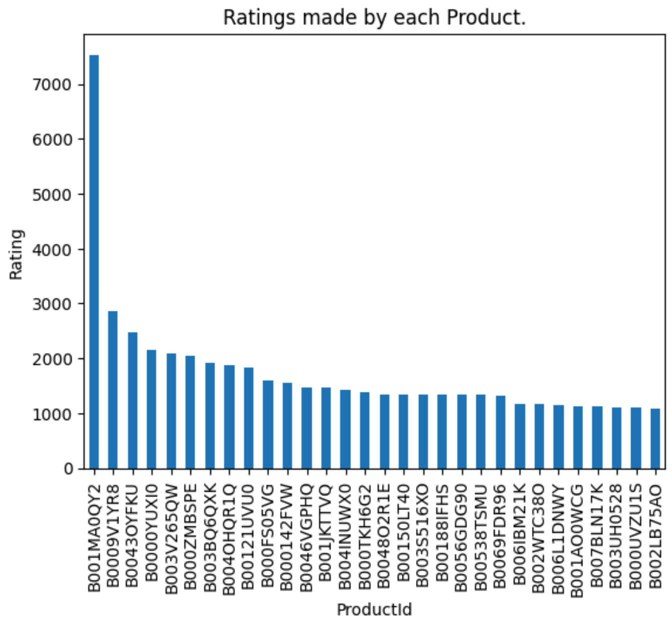
Step 5: Perform Data Visualization.



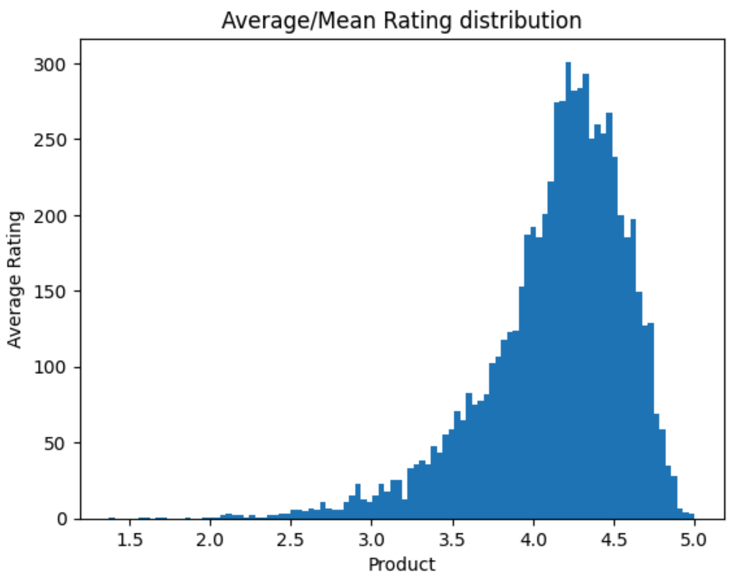
1. The resulting histogram visualizes the distribution of ratings within the dataset. Each bar in the histogram represents a range of ratings (based on the number of bins) on the x-axis, and the height of each bar represents the count or frequency of ratings falling within that range on the y-axis. It helps understand how ratings are distributed across the dataset and provides insights into the prevalence of certain rating values.



2. The resulting bar plot illustrates the count of ratings made by each user. Each bar represents a user (identified by 'UserId') on the x-axis, and the height of the bar corresponds to the number of ratings made by that user on the y-axis. This visualization helps in understanding the distribution of ratings contributed by individual users, highlighting users who have provided a higher number of ratings in the dataset.



3. The resulting bar plot visualizes the count of ratings received by each product. Each bar represents a product (identified by 'ProductId') on the x-axis, and the height of the bar corresponds to the number of ratings received by that product on the y-axis. This visualization offers insights into the popularity or engagement level of different products based on the number of ratings they've received.



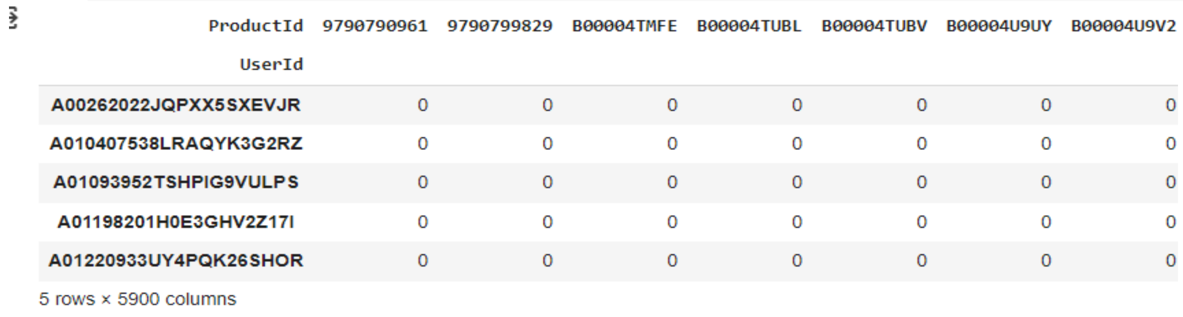
4. The resulting histogram visualizes the distribution of mean ratings for the products. Each bar represents a range of average ratings, and the height of the bar indicates the frequency of

products falling within that range of average ratings. This helps in understanding the distribution pattern of average ratings across the products in the dataset.

Step 6: Building Utility Matrix

In this section, we will call a function named pivot table. This function is aimed to cross tabulate each user against each place and output a matrix. It rearranges the sampled data to create a matrix where rows represent users ('UserId'), columns represent products ('ProductId'), and the cell values represent ratings given by users to products. The fill_value=0 parameter fills any missing values (if present) with zeros. This type of matrix representation is common in recommendation systems and collaborative filtering techniques.

```
data2=data.sample(20000)
ratings_matrix = data2.pivot_table(values='Rating',
index='UserId', columns='ProductId', fill_value=0)
ratings_matrix.head()
```



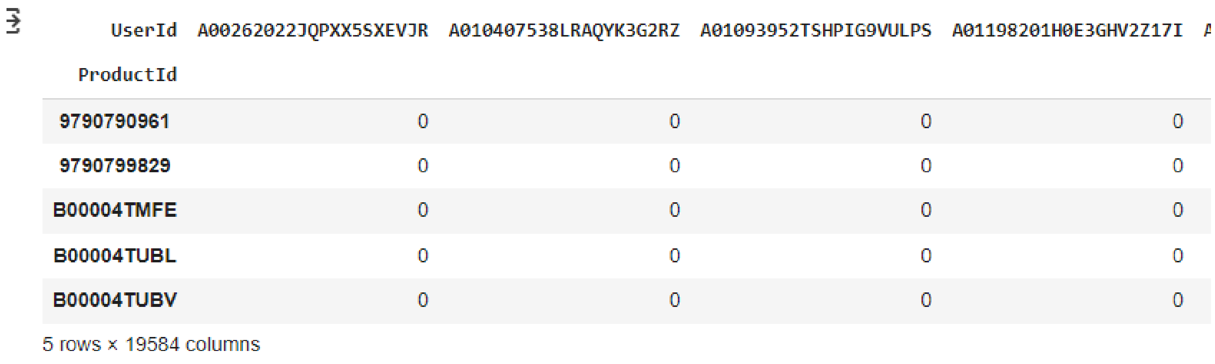
ProductId / UserId	9790790961	9790799829	B00004TMFE	B00004TUBL	B00004TUBV	B00004U9UY	B00004U9V2
A00262022JQPXX5SXEVJR	0	0	0	0	0	0	0
A010407538LRAQYK3G2RZ	0	0	0	0	0	0	0
A01093952TSHPIG9VULPS	0	0	0	0	0	0	0
A01198201H0E3GHV2Z17I	0	0	0	0	0	0	0
A01220933UY4PQK26SHOR	0	0	0	0	0	0	0

5 rows × 5900 columns

Step 7: Transposing the Matrix

Transposing a matrix swaps its rows and columns, effectively flipping it across \ its diagonal. In the context of recommendation systems or collaborative filtering, transposing the matrix might be done for various reasons, such as changing the orientation of user-item relationships for analysis of the algorithm Implementation.

```
        x_rating_matrix = ratings_matrix.T
        x_rating_matrix.head()
```



UserId / ProductId	A00262022JQPXX5SXEVJR	A010407538LRAQYK3G2RZ	A01093952TSHPIG9VULPS	A01198201H0E3GHV2Z17I
9790790961	0	0	0	0
9790799829	0	0	0	0
B00004TMFE	0	0	0	0
B00004TUBL	0	0	0	0
B00004TUBV	0	0	0	0

5 rows × 19584 columns

Step 8: Decomposing the Matrix

Decomposing a matrix, especially in the context of techniques like Singular Value Decomposition (SVD), refers to breaking down a matrix into constituent parts that reveal certain underlying structures or properties.

Singular Value Decomposition (SVD): SVD is a powerful matrix factorization method used in various fields, including data analysis, signal processing, and machine learning. It represents a matrix as the product of three matrices: $U \times \Sigma \times V^T$.

```
SVD = TruncatedSVD(n_components = 10)

decomposed_matrix =SVD.fit_transform(X)

decomposed_matrix.shape
```

The Truncated SVD technique reduces the dimensionality of the original matrix X by preserving only the most important information in a lower-dimensional space defined by the specified number of components (10 components in this case). The resulting decomposed_matrix is a reduced representation of the original matrix, containing fewer dimensions while retaining the essential information that best represents the original data. The shape of decomposed_matrix reflects the reduced dimensions after the decomposition.

Step 9: Generating a Correlation Matrix

The output of corr_matrix.shape will show the dimensions of the correlation matrix. If decomposed_matrix had a shape of (5900, 10), the resulting corr_matrix would likely have a shape of (5900, 5900) since corrcoef computes the correlation coefficients among the rows (or columns) of the input matrix, resulting in a square matrix where each row and column corresponds to a data point or component in the input matrix. The code calculates correlations between products based on a decomposed matrix identifies a specific product's correlations and counts how many other products have correlations higher than 0.80 with it.

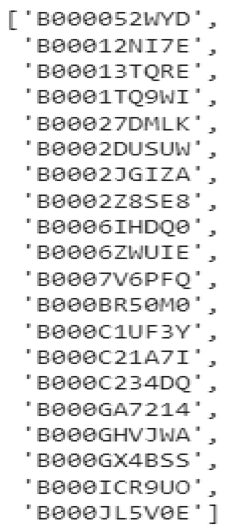
Step 10: Recommending the Highest Top Correlated Products.

```
#Recommending top 20 highly correlated products in sequence

recommend = list(X.index[corr_prod_id > 0.80])

recommend[:20]
```

In this, create a list of recommended products by selecting the top 20 products that have a high correlation (above 0.80) with a specific product based on the previously calculated correlations.



```
['B000052WYD',
 'B00012NI7E',
 'B00013TQRE',
 'B0001TQ9WI',
 'B00027DMLK',
 'B0002DUSUW',
 'B0002JGIZA',
 'B0002Z8SE8',
 'B0006IHDQ0',
 'B0006ZWUIE',
 'B0007V6PFQ',
 'B000BR50M0',
 'B000C1UF3Y',
 'B000C21A7I',
 'B000C234DQ',
 'B000GA7214',
 'B000GHVJWA',
 'B000GX4BSS',
 'B000ICR9UO',
 'B000JL5V0E']
```