Recommender systems

Introduction

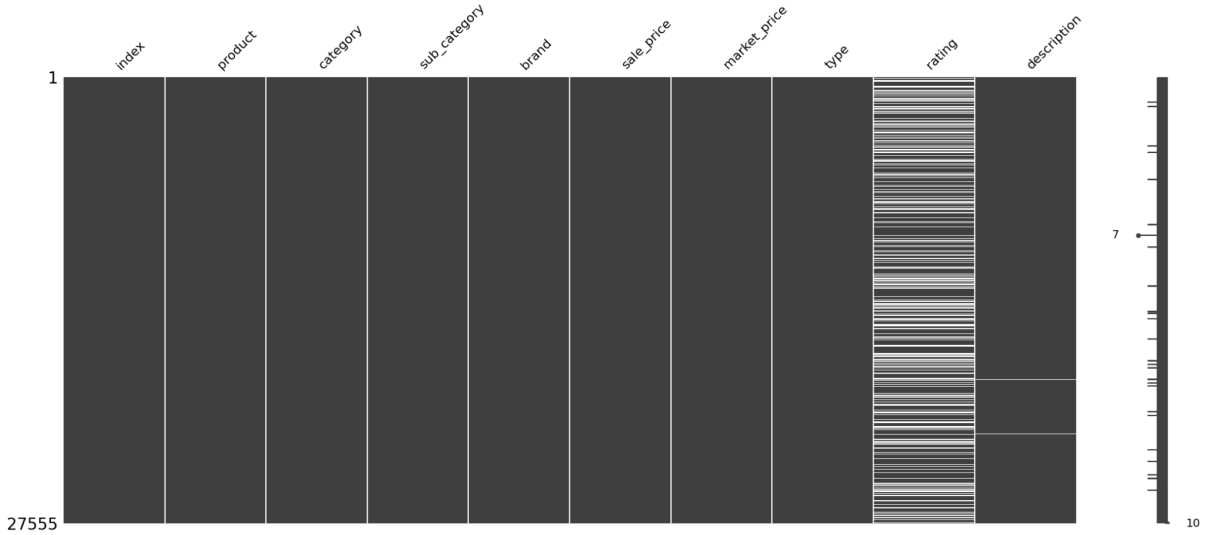
In the vast landscape of data science, recommender systems play a pivotal role in delivering personalized and targeted content to users, enhancing their overall experience in various domains such as e-commerce, streaming services, and social media. These systems are designed to predict and suggest items that users might find interesting or relevant, thereby addressing the challenge of information overload. Two prominent approaches to building recommender systems are collaborative filtering and content-based methods. Collaborative filtering leverages the wisdom of the crowd, making predictions based on the preferences and behaviors of similar users. On the other hand, content-based methods focus on the intrinsic characteristics of items and users, recommending items that are similar to those the user has interacted with before. This data science project delves into the realm of recommender systems, exploring both collaborative filtering and content-based methods to create a robust and effective recommendation engine. Specifically, Singular Value Decomposition (SVD) will be employed as a powerful technique within the collaborative filtering framework. SVD enables the extraction of latent factors from user-item interaction matrices, providing insights into underlying patterns and relationships that contribute to accurate recommendations.

Through the exploration of these methodologies and the application of SVD, this project aims to enhance our understanding of recommendation systems, ultimately contributing to the development of more efficient and precise algorithms for personalized content delivery. As we navigate the intricacies of collaborative filtering and content-based approaches, we embark on a journey to unravel the complexities of user preferences and interactions in the ever-evolving landscape of data science.

Methodology

Content-based

Missing values



The `rating` and `description` columns have missing values, as we dont need rating we can fully drop the column and other two rows in description.

Preprocessing

TF-IDF, which stands for Term Frequency-Inverse Document Frequency, is a widely used technique in natural language processing and information retrieval to represent the importance of words in a document relative to a collection of documents. It is particularly valuable when dealing with text data, such as product descriptions or reviews, and aims to capture the significance of individual terms within a document while considering their prevalence across the entire dataset.

The TF-IDF score for a term in a document is calculated by multiplying two components: Term Frequency (TF) and Inverse Document Frequency (IDF).

Term Frequency (TF):

The Term Frequency of a term t in a document d is the number of times the term appears in the document. It is calculated using the formula:

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

The idea is to measure how often a term occurs within a specific document. The resulting value is normalized to ensure it is not biased towards longer documents.

Inverse Document Frequency (IDF):

The Inverse Document Frequency of a term t is a measure of how unique or rare the term is across all documents in the dataset. It is calculated using the formula:

$$IDF(t) = log\frac{N}{1+df}$$

Adding 1 in the denominator prevents division by zero when a term appears in all documents. The logarithmic scale helps give more weight to terms that are less common across the dataset.

TF-IDF Score:

The TF-IDF score for a term t in a document d is obtained by multiplying the TF and IDF scores:

$$TF - IDF(t,d) = TF(t,d) * IDF(t)$$

This score reflects the importance of the term within the specific document and across the entire dataset.

By applying the TF-IDF method to encode text features in a dataset, you create a numerical representation that emphasizes the importance of terms in each document relative to their occurrence in the entire collection. This approach is particularly useful for measuring the similarity between documents using cosine similarity, where the TF-IDF vectors of the documents serve as input for the calculation. Cosine similarity quantifies the cosine of the angle between two vectors and is frequently used to determine the similarity between documents in information retrieval and recommendation systems.

Content-based recommendation systems rely on the intrinsic characteristics of items and user preferences to generate personalized recommendations. One common approach is to represent items and users as vectors in a high-dimensional space, with each dimension corresponding to a feature. In this context, cosine similarity is often employed to measure the similarity between these vectors. Let's explore the content-based method using cosine similarity.

Representation of Items and Users:

Each item is represented as a feature vector, denoted as $(\mathbf{I}_i)$, where (i) represents the item.

Users preferences are captured by their feature vectors, denoted as $(\mathbf{U}_u)$, where (u) represents the user.

Cosine Similarity:

Cosine similarity measures the cosine of the angle between two vectors and is calculated as follows:

$$similarity(A, B) = cos(\theta) = \frac{A \cdot B}{||A||\,||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2 \sum_{i=1}^{n} B_i^2}}.$$

Where:

$(\mathbf{A} \cdot \mathbf{B})$ is the dot product of vectors $(\mathbf{A})$ and $(\mathbf{B})$.

$(\|\mathbf{A}\|)$ and $(\|\mathbf{B}\|)$ are the Euclidean norms of vectors $(\mathbf{A})$ and $(\mathbf{B})$, respectively.

Content-Based Recommendation using Cosine Similarity:

For a given user (u) and an item (i), the predicted preference score $(\hat{r}_{ui})$ can be calculated using the cosine similarity between the user's preferences $(\mathbf{U}_u)$ and the item's features $(\mathbf{I}_i)$:

$$[\hat{r}_{ui} = \text{Cosine Similarity}(\mathbf{U}_u, \mathbf{I}_i)]$$

This score represents the estimated preference of user s for item i.

Aggregate Recommendations:

To provide a list of recommendations for a user s items are ranked based on their predicted preference scores. The higher the score, the more likely the item is to be recommended.

Advantages of Content-Based Methods:

Content-based methods are capable of recommending items that are similar to those a user has interacted with before, making them suitable for providing personalized recommendations.

Limitations:

Content-based methods may face challenges in capturing user preferences for diverse or novel items as they rely on item features.

In summary, content-based recommendation systems leverage cosine similarity to quantify the similarity between user preferences and item features. By representing items and users as vectors in a feature space, these systems can provide personalized recommendations based on the intrinsic characteristics of items and the preferences of individual users.

The recommended products for 'Natural Peanut Butter Crunch - Unsweetened/Gluten Free/ Non-GMO/Vegan' was:

Product name	Cosine score
Peanut Butter - Crunchy	0.564834
Peanut Butter - Honey Roast Crunchy	0.563605
Cocoa Spread	0.510987
Spread - Cocoa with Almond	0.50073
Natural Honey - Multiflora	0.354396
Mango Jam	0.347216
100% Pure Honey, 1 kg + Oats, 1 kg	0.347198
Mayonnaise - Garlic	0.346637
Raw Organic Apple Cider Vinegar	0.345455
FunFoods Veg Mayonnaise Original	0.344311

Collaborative-filtering

Collaborative filtering is a powerful technique in recommender systems that leverages the collective wisdom of users to make predictions about their preferences. The underlying idea is to identify similarities between users or items based on their historical interactions and recommend items that similar users have liked or interacted with. Collaborative filtering methods can be broadly categorized into two types: user-based collaborative filtering and item-based collaborative filtering.

User-Based Collaborative Filtering:

1. User-Item Interaction Matrix:
Represent the user-item interactions in a matrix where rows correspond to users, columns to items, and the entries represent user ratings or interactions.

2.User Similarity Calculation:
Calculate the similarity between users based on their historical interactions. Common similarity metrics include cosine similarity or Pearson correlation.

$$[\text{Similarity}(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} \cdot r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2} \cdot \sqrt{\sum_{i \in I_{uv}} r_{vi}^2}}]$$

where (I_{uv}) is the set of items that both users (u) and (v) have interacted with, and (r_{ui}) is the rating given by user (u) to item (i).

3.Prediction of User Preferences:
Predict the rating or preference of a user \(u \) for an item \(i \) by considering the ratings of similar users for that item.

$$[\hat{r}_{ui} = \frac{\sum_{v \in N(u)} \text{Similarity}(u,v) \cdot r_{vi}}{\sum_{v \in N(u)} |\text{Similarity}(u,v)|}]$$

where $(N(u))$ is the set of users similar to user (u).

Item-Based Collaborative Filtering:

1.User-Item Interaction Matrix:
Similar to user-based collaborative filtering, represent the user-item interactions in a matrix.
2. Item Similarity Calculation:
Calculate the similarity between items based on user interactions. Common similarity metrics include cosine similarity or Pearson correlation.

$$[\text{Similarity}(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} \cdot r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2} \cdot \sqrt{\sum_{u \in U_{ij}} r_{uj}^2}}]$$

where (U_{ij}) is the set of users who have interacted with both items (i) and (j) , and (r_{ui}) is the rating given by user (u) to item (i).

3.Prediction of User Preferences:
Predict the rating or preference of a user (u) for an item (i) by considering the ratings of similar items that the user has interacted with.

$$[\hat{r}_{ui} = \frac{\sum_{j \in N(i)} \text{Similarity}(i,j) \cdot r_{uj}}{\sum_{j \in N(i)} |\text{Similarity}(i,j)|}]$$

where $(N(i))$ is the set of items similar to item (i).

Singular Value Decomposition (SVD) is a matrix factorization technique that is commonly used in collaborative filtering for recommender systems. SVD decomposes a user-item interaction matrix into three matrices, capturing latent factors that represent underlying patterns in the data. In the context of collaborative filtering, SVD helps in predicting missing values in the user-item matrix, enabling the recommendation of items to users.

1. *User-Item Interaction Matrix:*
Represent user-item interactions in a matrix (R), where rows correspond to users, columns correspond to items, and entries represent user ratings or interactions. The matrix may be sparse, as not all users interact with all items.

$$[R = \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,m} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n,1} & r_{n,2} & \cdots & r_{n,m} \end{bmatrix}]$$

2. Singular Value Decomposition (SVD):
Decompose the user-item interaction matrix (R) into three matrices (U), (Σ), and (V^T):

$$[R_{\text{approx}}\ U\Sigma V^T]$$

- (U) is a user matrix.
- (Σ) is a diagonal matrix containing singular values.
- (V^T) is an item matrix.

3. Matrix Reconstruction:
Reconstruct the original matrix (R) by multiplying the three matrices:

$$[\hat{R} = U\Sigma V^T]$$

The reconstructed matrix $(\hat{R})$ estimates the missing values in the original matrix.

4. Prediction:
To predict the rating $(\hat{r}_{ui})$ for a user (u) and item (i), use the corresponding elements in the reconstructed matrix $(\hat{R})$:

$$[\hat{r}_{ui} = (\hat{R})_{ui}]$$

5. Recommendations:

Generate recommendations for a user based on the predicted ratings. Items with the highest predicted ratings for a user are recommended.

The reported $RMSE$ for SVD model was 1.3192252722205793.