Content-based Recommender System

Dataset: BigBasket Product List

Mahan Madani

1. Overview

This dataset is comprised of over 27000 records of products, containing information in 10 different columns. The majority of the columns are text-based and include information on each individual product's title, sale price, and type.

The goal of this project is to design and implement a recommender system based on the given dataset. I built a **Content-based** Recommendation system using the **K-Nearest Neighbors** algorithm.

2. Dataset Exploration & Analysis

The dataset stores 10 different attributes. Each record in the dataset belongs to one specific product.

Below you can find a list of all the attributes:

- **index:** The index of the record, starting from 1.
- **product:** Title of the product as it is listed on the website.
- **category:** Category into which the product has been classified.
- **sub_category:** Subcategory into which the product is kept.
- **brand:** Brand of the product.
- **sale_price:** Price at which the product is being sold on the website.
- **market_price:** Market price of the product.
- **type:** Type into which product falls.
- **rating:** The average rating the product has gotten from its consumers.
- **description:** Description of the product.

3. Data Preprocessing

3.1 Check for Duplicate Records:

To ensure the dataset contains no duplicate records, a combination of all features (except 'index') can be used. If two or more records share the same value for these columns, that means they are potentially duplicates.

Even if the records aren't completely identical, they may still contradict each other and need to be handled. **After testing the dataset, it seems that it contains 707 duplicate records**. All instances of a duplicate record except the first one must be dropped, as we don't want to recommend the exact same item to a customer who has already purchased an item.

3.2 Handle Null Values:

As seen in this table, out of the 10 features in the dataset, 4 contain null values.

For the 'product' and 'brand' columns, I decided to drop the two records that were missing these features.

For the 'rating' attribute, I simply imputed all missing values with 0, meaning that said products haven't received any ratings yet.

For the description attribute, I imputed the missing values with an empty string.

	Null Count
index	0
product	1
category	0
sub_category	0
brand	1
sale_price	0
market_price	0
type	0
rating	8463
description	113

3.3 Detect Outlier Values:

Using the z-score method, outlier data can be identified. Outliers can be detected from the 'sale_price' column, and over 500 records are detected based on this feature. However, due to the fact that the goal is to build a recommendation system, it's not logical to drop the outliers as we won't be able to recommend anything to the customers who have bought the outlier products.

3.4 Feature Generation:

Based on the available data, the following features can be added to further improve the dataset:

- **discount:** The discount value for each product (can be 0). Extracted from the difference between 'sale_price' and 'market_price'.
- **discount percent:** The percentage of the discount. Extracted from the 'discount' and 'market_price' attributes.

Natural Language processing tools can be used to extract useful data from the 'Synopsis' column. I used Sentiment Analysis (from the TextBlob package) to extract numerical values relating to each product's 'description', these values are **Polarity** and **Subjectiveness**, and they were added as new features to the dataset.

3.5 Feature Transformation:

Training and testing a model require numerical data, and considering how the majority of this dataset is text-based, some features must be transformed so that they can be used in machine-learning tasks.

Vectorization is the technique used here. All categorical features like 'category', 'sub_category', and 'type' can be vectorized using **one-hot encoding**, creating a new column for each unique category in them.

The 'brand' attribute has too many unique values so one-hot encoding isn't the best approach here. Instead, **TF-IDF** vectorization can be used to transform this attribute to a collection of numerical features. **SVD** is then used for dimension reduction where necessary, especially for the TD-IDF vectors.

Feature Scaling: All of the numerical attributes must be scaled similarly. Otherwise, having features with varying degrees of magnitude and range will cause different step sizes for each feature. I used a **Standard Scalar** to scale features such as 'sale_price' and 'rating'.

4. Visualization

Figure 4.1: Wordcloud of ‘description’

This plot displays the most frequently repeated word in the ‘description’ column.

Figure 4.2: Wordcloud of ‘brand’

This plot displays the most frequently repeated word in the ‘brand’ column, showcasing the most popular brands on the website.

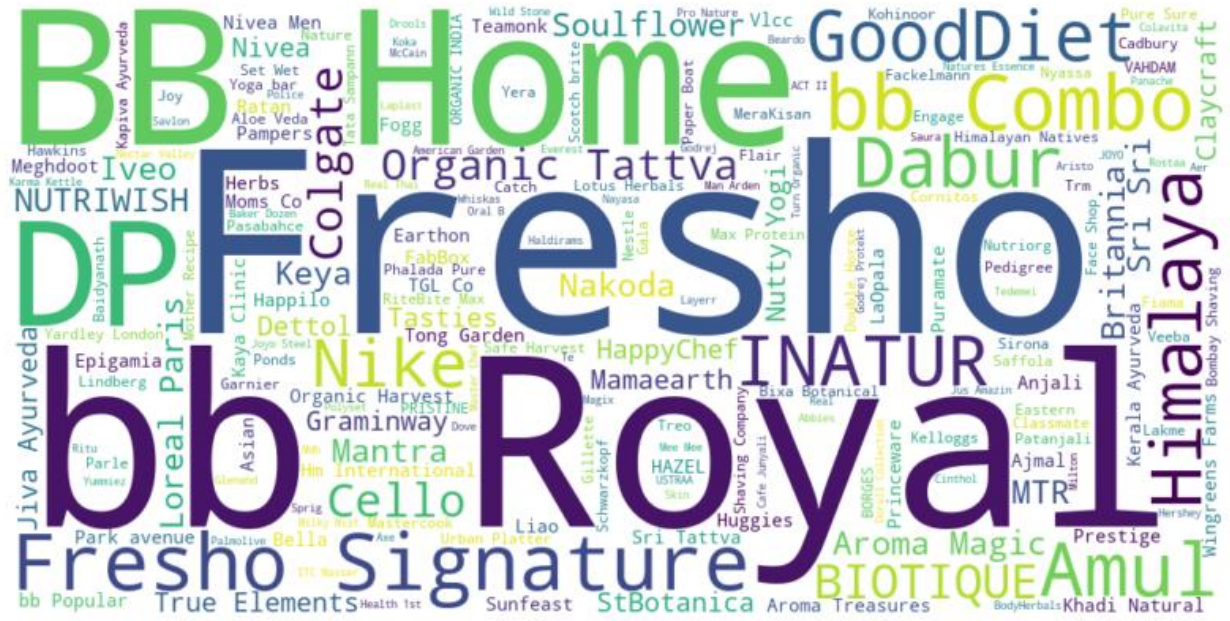


Figure 4.3: Bar plot of the most popular brands

An obvious connection is present between this plot and the wordcloud.

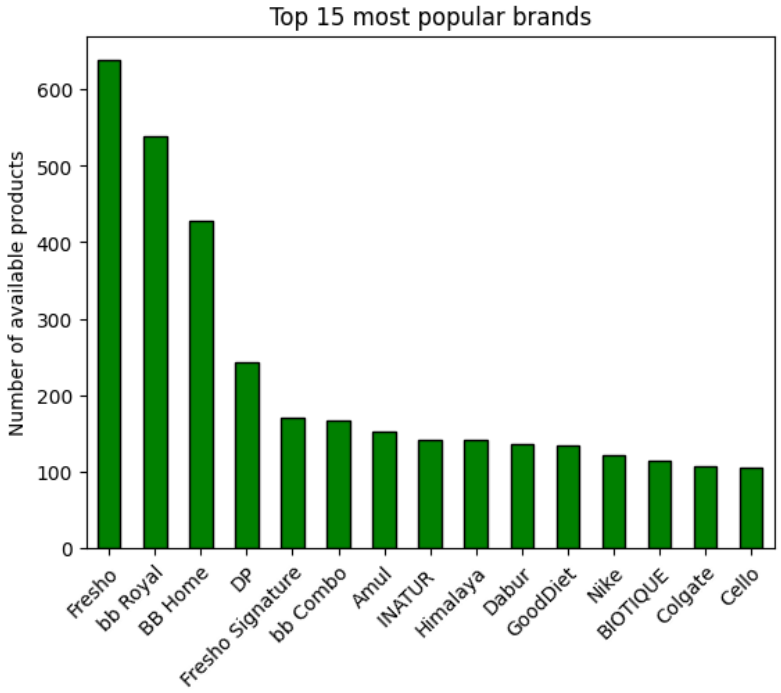


Figure 4.4: Average product rating, grouped by category

This plot displays the average rating of each product category.

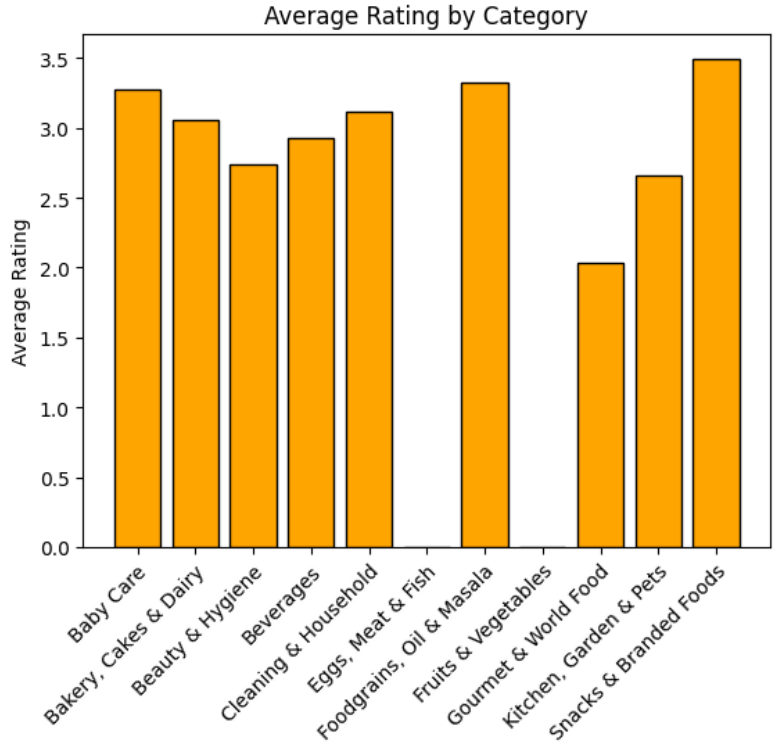
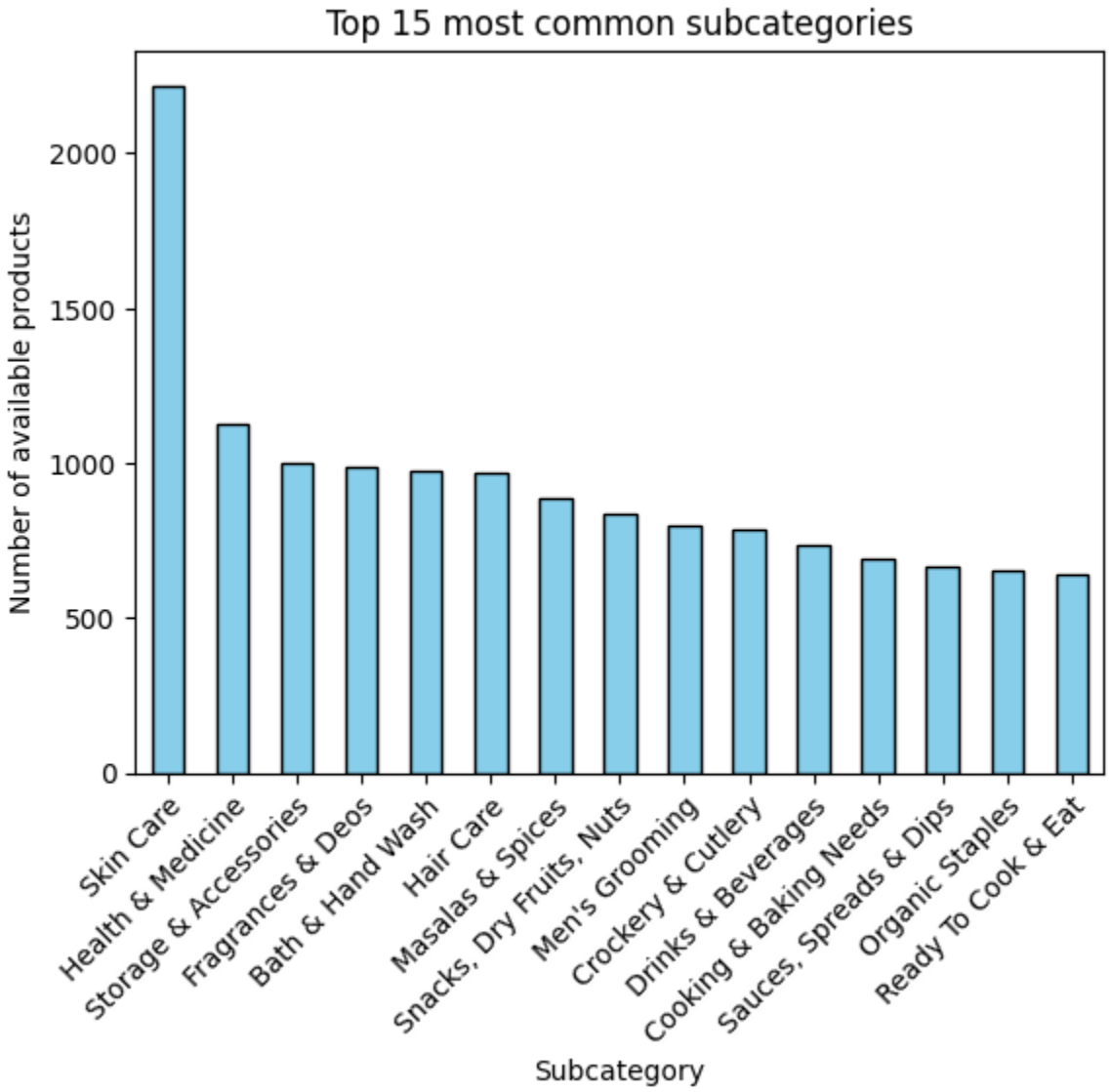


Figure 4.5: Bar plot of the most common subcategories

This plot visualizes the top 15 most common subcategories of products available on the website.



Skin Care products seem to have to highest share among all subcategories. In general, health and care-related products are prevalently found and food-related products start showing up as well.

Figure 4.6: Scatter Plot of sale price vs market price

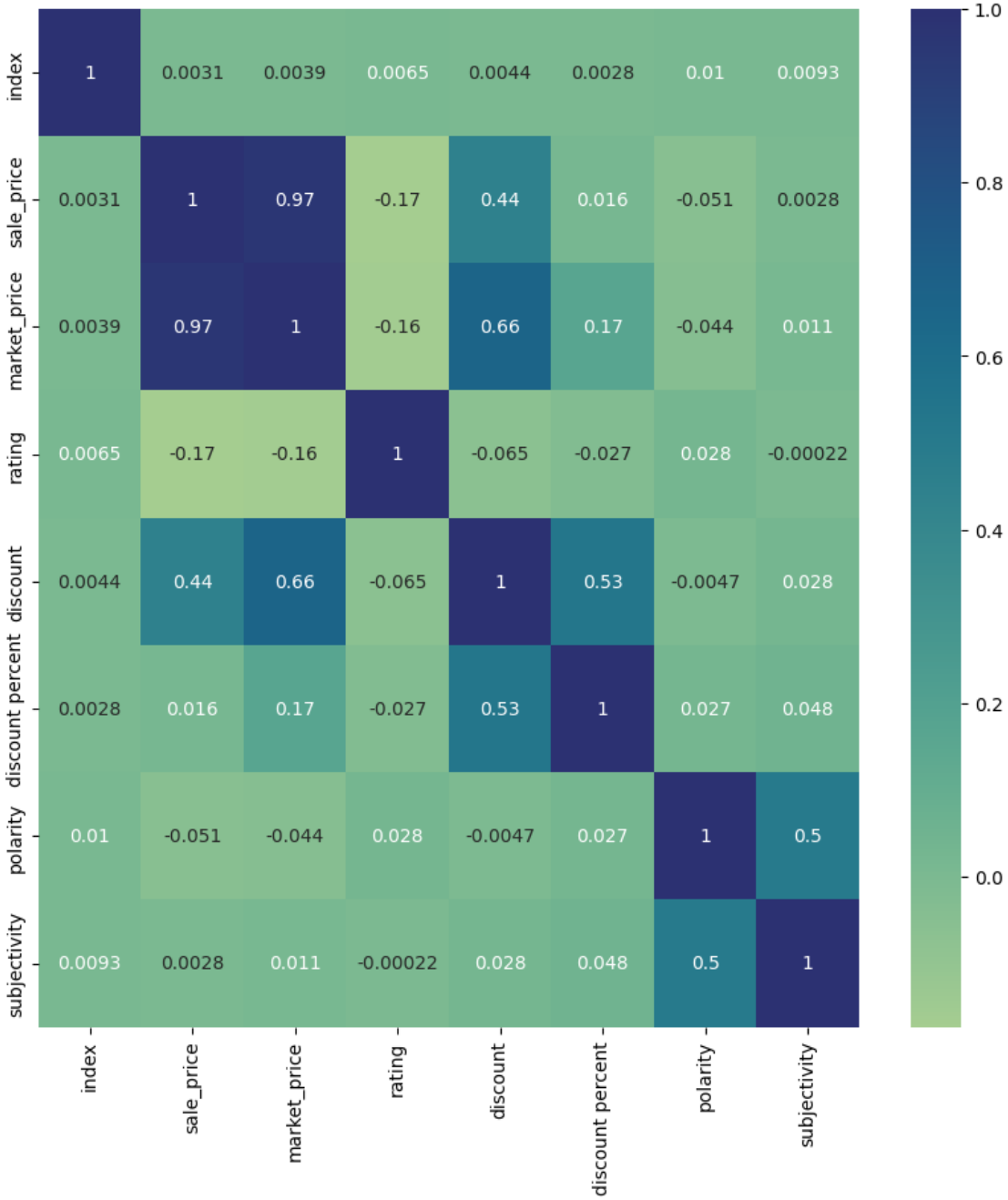
This scatter plot displays the relationship between the sale price and market price values of different products. It is clear that all available products are sold at or below the market price, with some having heavy discounts.

There is no sale data available in this dataset so we can't make any statements regarding the actual sale of discounted products, but we can predict an increase in sales for them.

Figure 4.7: Heatmap of the Correlation Matrix

This heatmap visualizes the correlation between the various attributes of the dataset. Some strong correlations can be found here but are mostly expected, such as the correlation between sale_price and market_price.

An Interesting observation here is the lack of any strong negative correlation.



5. Recommendation System

I used the **K-Nearest Neighbors** algorithm, which makes recommendations based on shared similar features. This is a fully content-based method and we can only customize the recommendations based on the similarities between products, almost similar to a clustering algorithm. The difference is that instead of grouping up the products, we can find **k** number of neighbor products that share similarities with our target product.

Once the model is trained on all of the chosen features, we can pass a product name to the `recommend_products` functions and ask for a number of products. Below you can find an example of a product recommendation:

```
Creme Soft Soap - For Hands & Body                          rating: 4.4    price: 162.0    type: Bathing Bars & Soaps
The recomended products are:

Creme Care Soap - For Hands & Body                          rating: 4.1    price: 158.0    type: Bathing Bars & Soaps
Creme Care Soap - For Hands & Body                          rating: 4.5    price: 85.0     type: Bathing Bars & Soaps
Soap - Creme Care                                           rating: 4.0    price: 100.0    type: Bathing Bars & Soaps
Creme Care Soap - For Hands & Body                          rating: 4.3    price: 54.0     type: Bathing Bars & Soaps
Creme Soft Soap - For Hands & Body                          rating: 4.2    price: 85.0     type: Bathing Bars & Soaps
Creme Care Soap - For Hands & Body                          rating: 4.4    price: 162.0    type: Bathing Bars & Soaps
Gel Bathing Bar - Bearberry & Blackcurrant                  rating: 4.3    price: 65.0     type: Bathing Bars & Soaps
Soap Bar - Water Lily & Cooling Mint, Fresh Splash, Save Rs. 6/-    rating: 4.3    price: 135.0    type: Bathing Bars & Soaps
Aeda Soap - Ramacham                                        rating: 4.5    price: 25.0     type: Bathing Bars & Soaps
Hexa Advanced Soap                                          rating: 4.2    price: 34.0     type: Bathing Bars & Soaps
```

This model is flexible because it can recommend items regardless of category and type (as shown in the example found in the code). It works for new and existing users as well, having the ability to recommend products independent of a user's shopping history.

An enhanced version of this recommender should also consider user preferences as well, but this is not possible in this dataset.