**INFOSYS SPRINGBOARD INTERNSHIP**

**FINAL REPORT**

**DOMAIN:** ARTIFICIAL INTELLIGENCE

**PROJECT TITLE:** AI STYLIST

**MENTOR NAME:** ANIL KUMAR SHAW

**TEAM MEMBERS:**

1. Monish
2. Varsini
3. Sakshi
4. Bharath
5. Eldho Elias

# TABLE OF CONTENTS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CHAPTER NO.** | | **TITLE** | **Page**  **Number** | |
| **1** |  | **INTRODUCTION** | |  |
|  | 1.1 | Abstract | | 4 |
|  | 1.2 | Overview of the project | | 4 |
| **2** |  | **DATASET PREPROCESSING** | |  |
|  | 2.1 | Data Loading and Preview | | 5 |
|  | 2.2 | Descriptive Statistics | | 6 |
| **3** |  | **VISUALIZATION** | |  |
|  | 3.1 | Top Brands by Product Count: | | 7 |
|  | 3.2 | Sales Price Distribution | | 8 |
| **4** |  | **TEXT PREPROCESSING** | |  |
|  | 4.1 | Stop Words | | 9 |
|  | 4.2 | Remove Special Characters | | 10 |
|  | 4.3 | Removing Non-English Words | | 11 |
|  | 4.4 | Use Contractions | | 13 |
|  | 4.5 | Lemmatization | | 15 |
| **5** |  | **CONTENT BASED FILTERING** | |  |
|  | 5.1 | Bag of Words (BoW): | | 17 |
|  | 5.2 | TF-IDF | | 19 |
|  | 5.3 | Word2Vec | | 21 |
|  | 5.4 | Hybrid Recommendation | | 23 |
|  | 5.5 | Brand-Based Recommendation | | 25 |
|  | 5.6 | Image-Based Recommendation | | 27 |
|  | 5.7 | Combination of Product Name and Brand-Based Recommendation | | 29 |
| **6** |  | **COLLABRATIVE FILTERING** | |  |
|  | 6.1 | Types of Collaborative Filtering | | 31 |
|  | 6.2 | Recommendations Using VGG16 | | 32 |
|  | 6.3 | Recommendations Using MobileNet | | 34 |
|  | 6.4 | Implementation Steps | | 39 |
|  | 6.5 | Evaluation and Advantages | | 41 |
| **7** |  | **CONCLUSION** | |  |

1. **Introduction**

**Abstract:**

The project "AI Stylist" explores the intersection of fashion and technology through data analysis and visualization. Utilizing Python and various data manipulation libraries, the project aims to provide insights into the most prominent brands and their product distributions. This document outlines the key components of the analysis, including data preprocessing, statistical summaries, and visualization techniques, to empower data-driven decision-making in the fashion industry.

**Overview of the project**

The fashion industry has evolved into a dynamic space where creativity meets technology, enabling brands to deliver personalized experiences to consumers. With the rise of artificial intelligence (AI), applications like virtual stylists and personalized recommendation systems are transforming how people shop for fashion. This project explores how data analysis and visualization can uncover trends and patterns that inform the development of an AI Stylist system.

An AI Stylist leverages data to suggest products tailored to individual preferences, making the shopping experience more engaging. The effectiveness of such systems relies on analyzing product datasets to identify brand popularity, category trends, and other key insights. This study focuses on using a JSON dataset containing fashion product information to derive meaningful patterns that can enhance AI-driven recommendations.

Python libraries such as Pandas and Matplotlib are used for data manipulation and visualization. The project objectives are:

1. Exploring the dataset’s structure and understanding its attributes.
2. Identifying key trends, including brand popularity and distribution.
3. Presenting findings visually to support intuitive understanding.

Through structured data analysis and visualization, this project demonstrates how insights from fashion product data can lay the groundwork for developing intelligent, customer-focused AI solutions.

1. **Data Preprocessing**

**1.Data Loading and Preview**:

The first step in the analysis involves loading the dataset using the Pandas library. The dataset, provided in JSON Lines format, is read into a DataFrame to facilitate data manipulation.

Code:



This code snippet reads the dataset and displays the first few rows to provide an overview of its structure and contents. The data.head() function is particularly useful for verifying the successful import of the data.



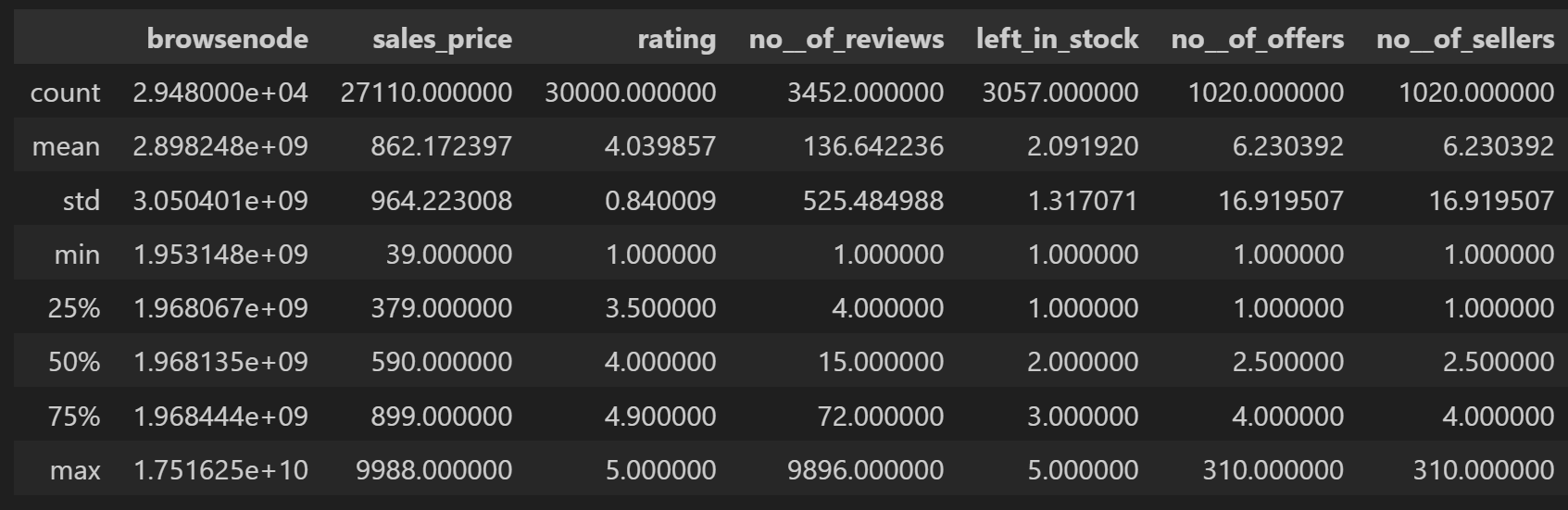
**2. Descriptive Statistics**:

To understand the dataset’s statistical properties, we use the describe and info methods. These functions summarize the data distribution and provide details about data types and missing values.

Code:



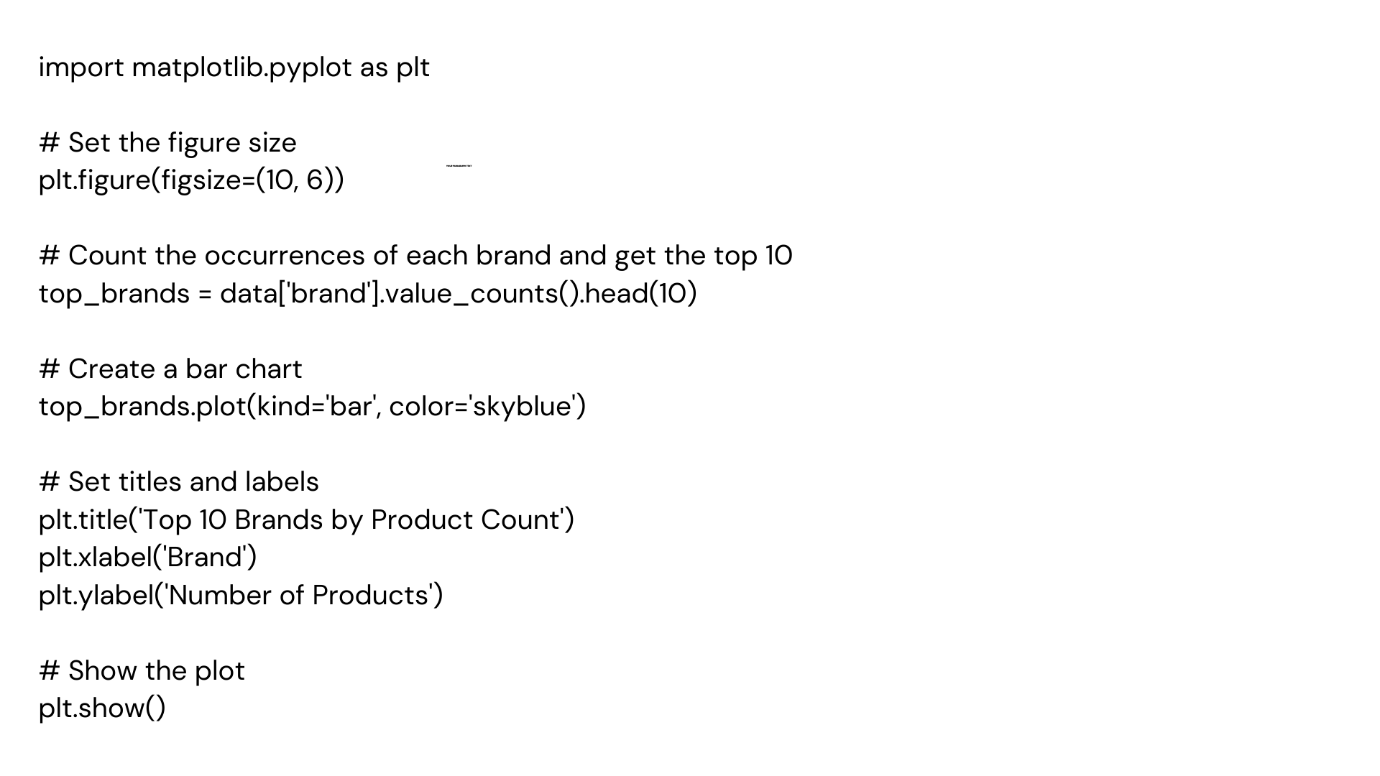
* data.describe(): Offers statistical insights such as mean, standard deviation, and quartiles for numeric columns.
* data.info(): Displays the structure of the dataset, including the number of entries, data types, and non-null counts.



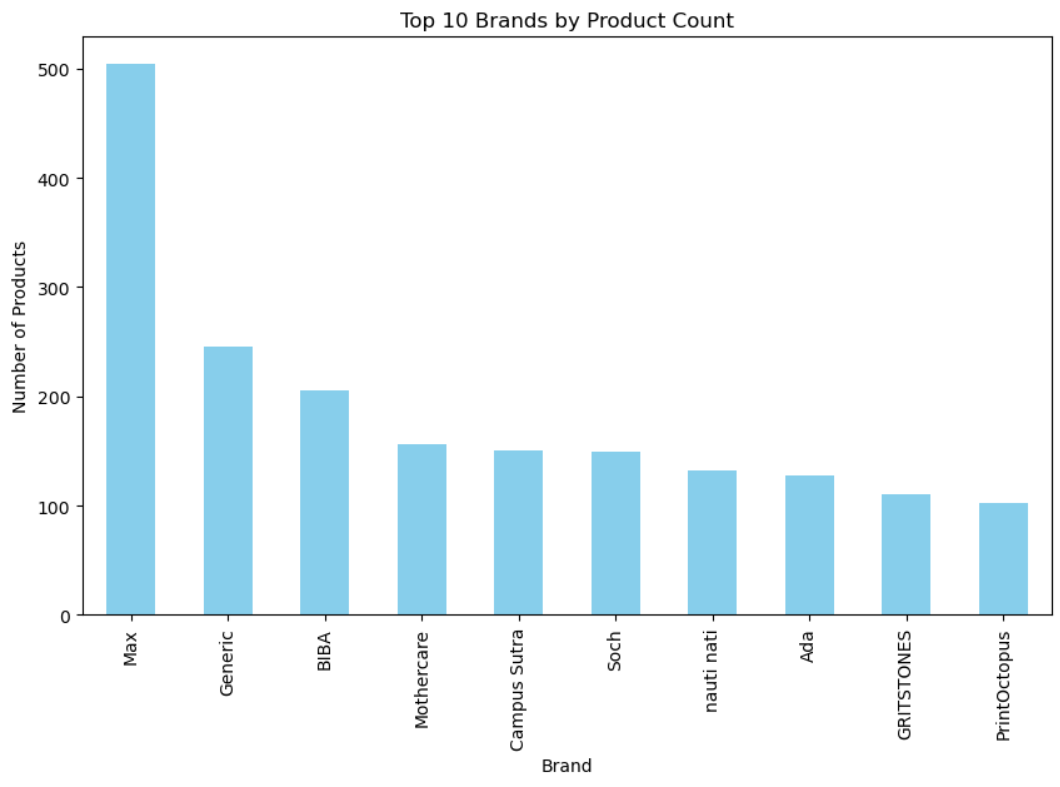
1. **Data Visualization:**

**Top Brands by Product Count:**

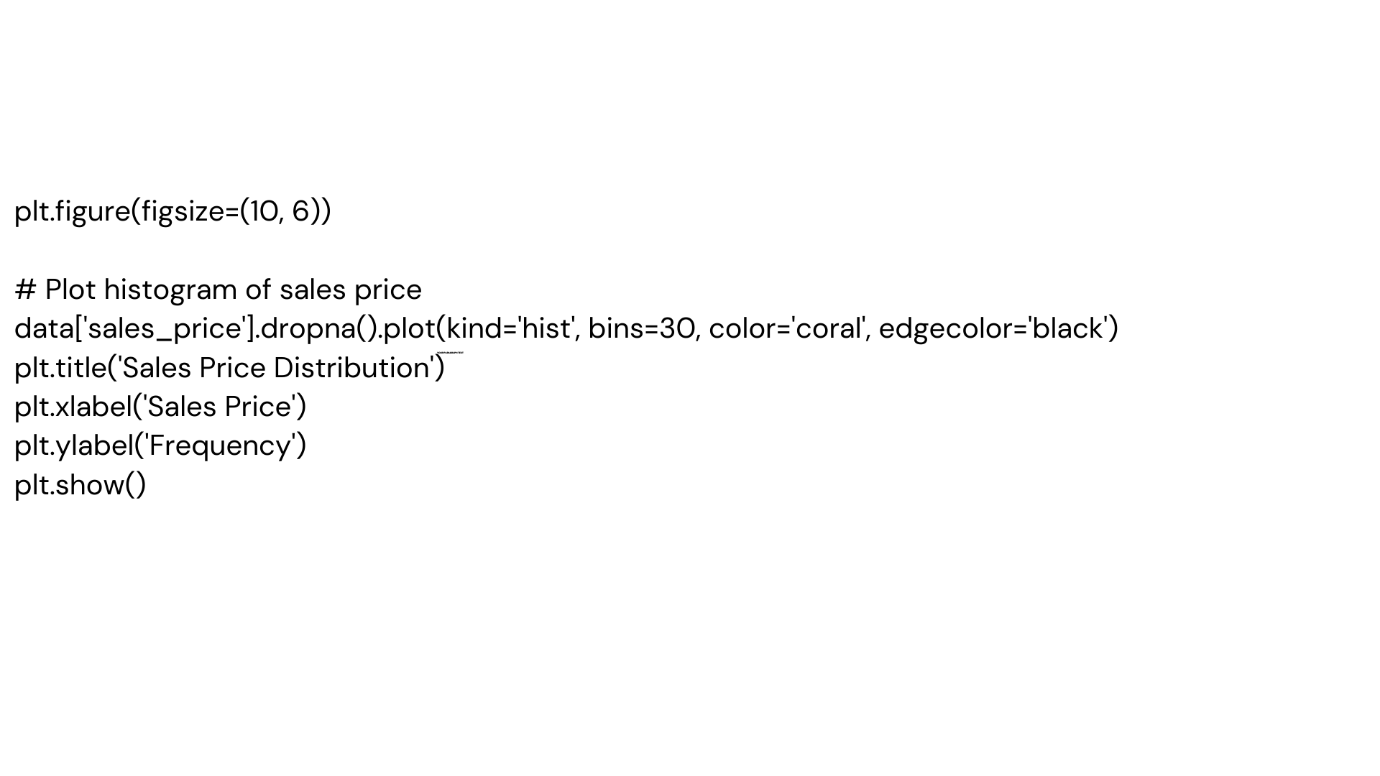
Visualizing data is a powerful way to uncover patterns and trends. Here, a bar chart is created to show the top 10 brands by product count.

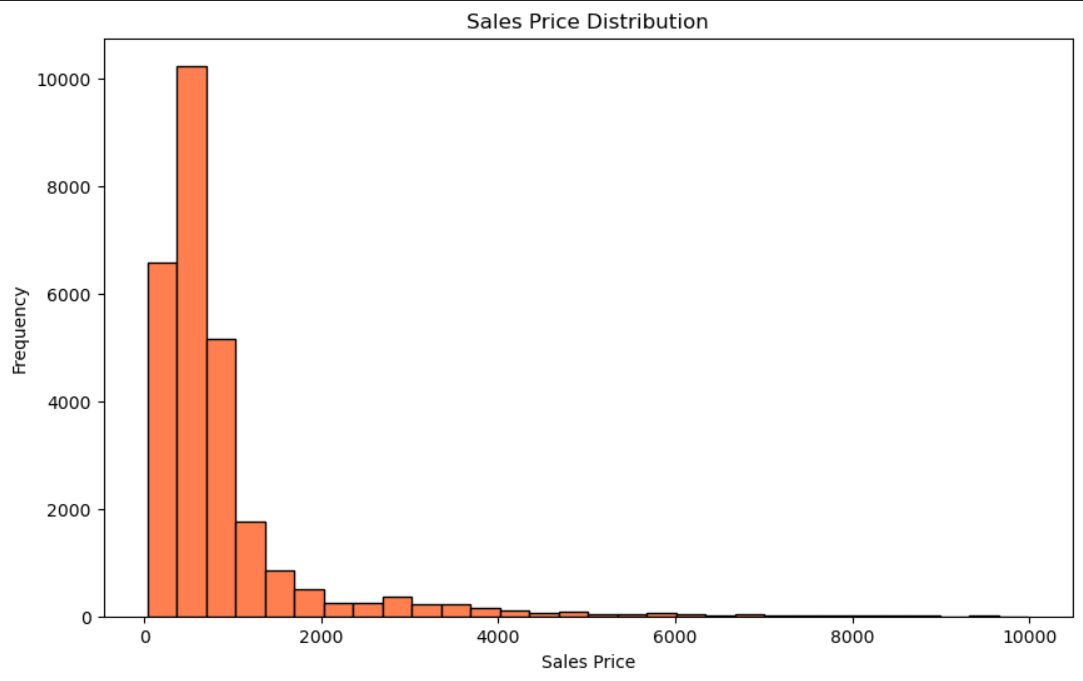


This code computes the frequency of each brand in the dataset, extracts the top 10, and plots the results using Matplotlib. The value\_counts() method efficiently counts occurrences, while the plot() function generates the bar chart.



SALES PRICE DISTRIBUTION

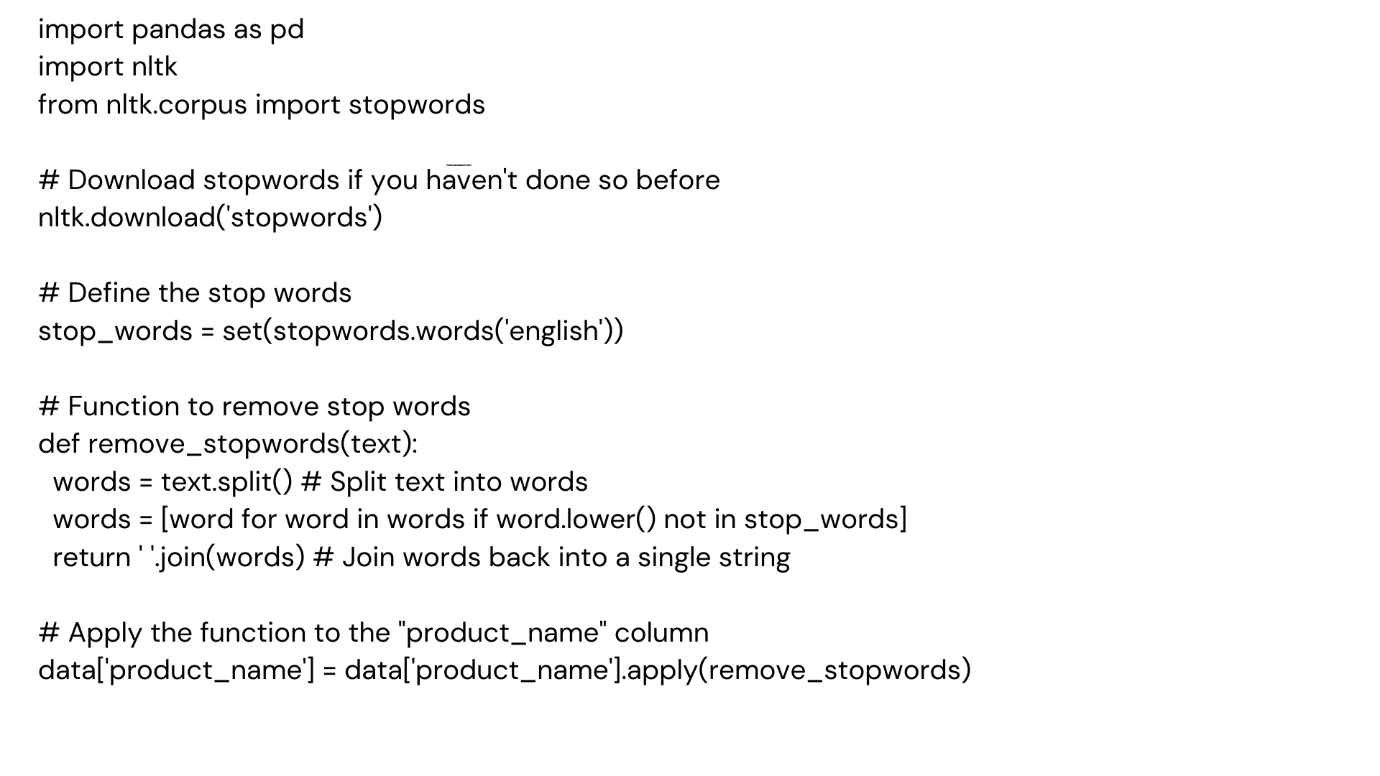


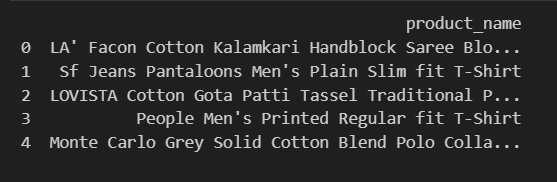


1. **Text Preprocessing:**

**1.Why Stop Words Are Removed in Coding:**

1. **Reduce Processing Load**:
   * Stop words make up a large portion of natural language but add little value to understanding the core meaning of text.
   * Removing them reduces the size of the data being processed, making algorithms faster.
2. **Enhance Model Accuracy**:
   * Models like classification or clustering focus on keywords that contribute to meaning. Removing stop words prevents them from overwhelming meaningful words.
3. **Simplify Algorithms**:
   * By removing unnecessary words, tasks such as keyword extraction or sentiment analysis become easier to compute.



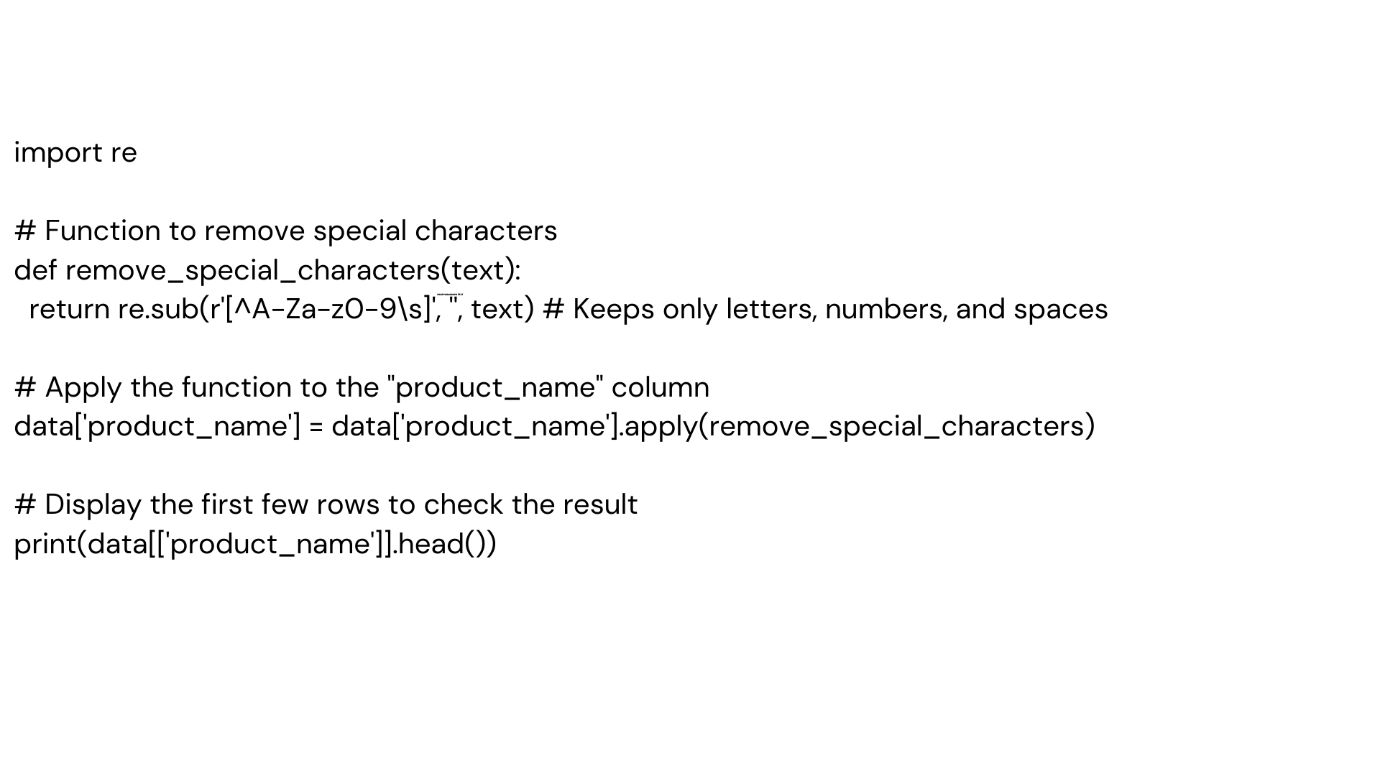


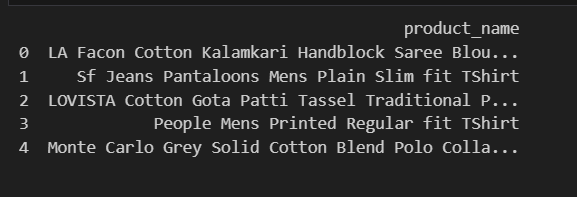
**2.Why Remove Special Characters in Text Preprocessing?**

Special characters (like !, @, #, &, etc.) are often removed in text preprocessing for tasks like machine learning, natural language processing, or text analytics. Here's why:

**Key Reasons:**

1. **Noise Reduction**:
   * Special characters often don't add meaningful information.
   * For example, smart-phone! and smartphone mean the same thing, so the - and ! are unnecessary.
2. **Standardization**:
   * Helps create uniform text data by eliminating inconsistent symbols that may interfere with downstream processing.
3. **Avoid Errors**:
   * Special characters can cause issues in tokenization, feature extraction, or when working with regular expressions and other string operations.



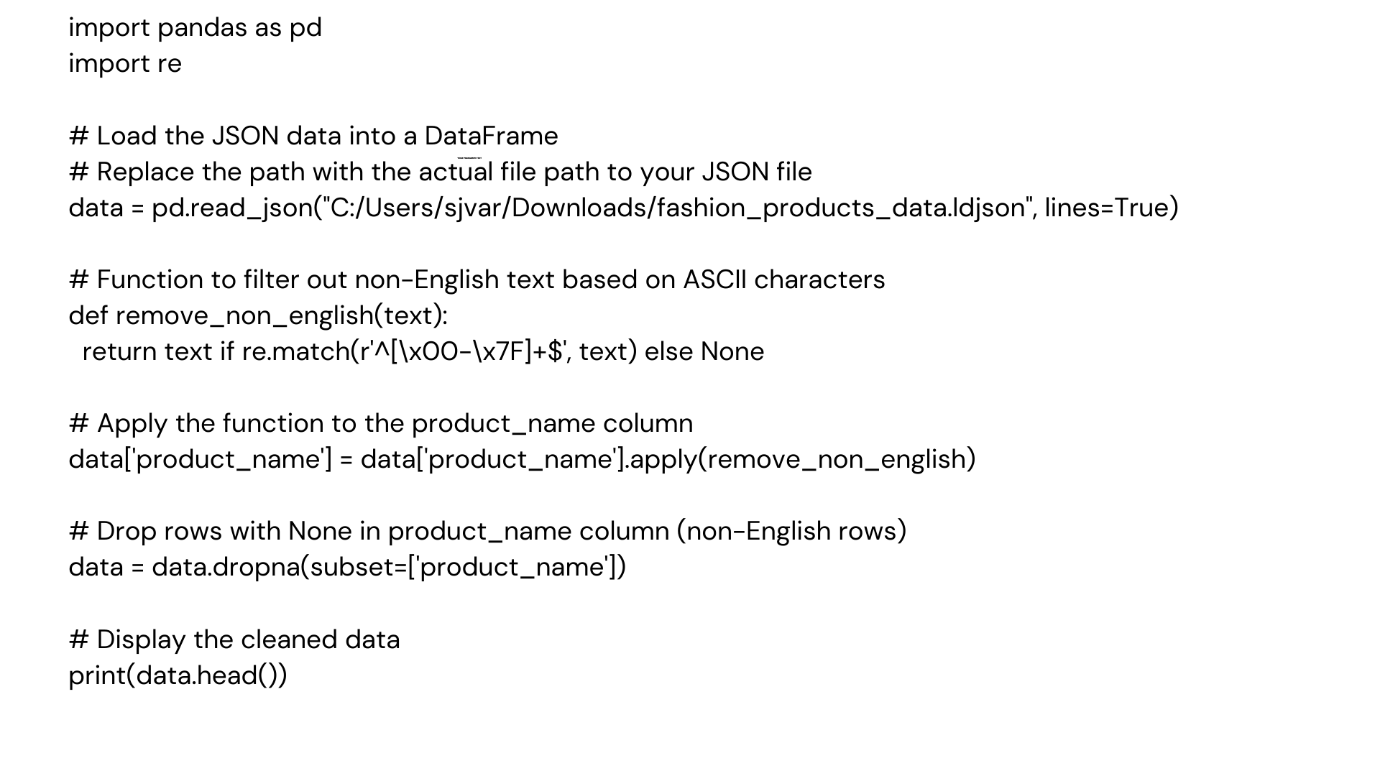


**3. Purpose of Removing Non-English Words or Text**

In text preprocessing, filtering out non-English content is essential for tasks that involve analyzing English-only data. Here's why:

**Why Remove Non-English Words/Text?**

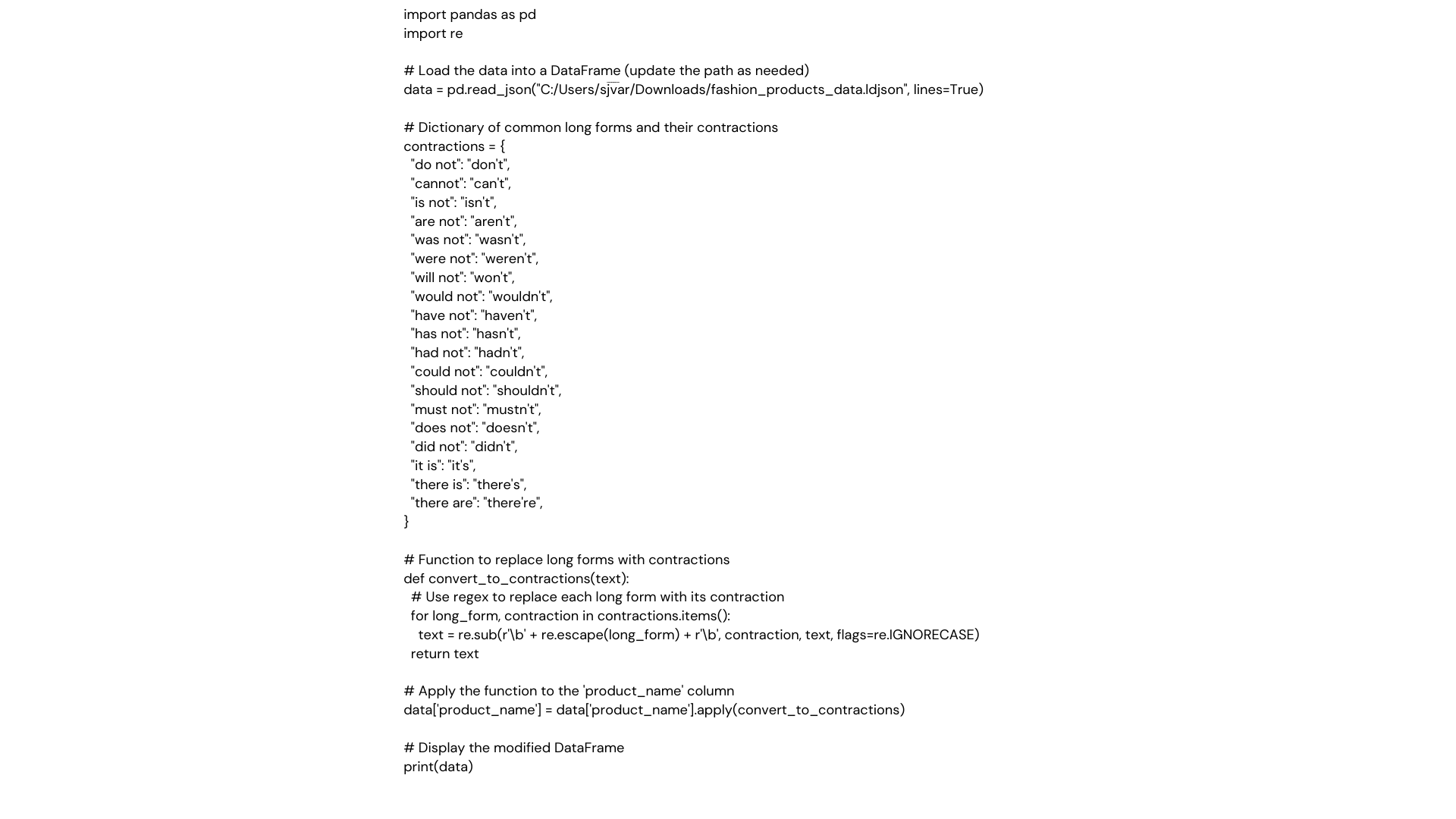
1. **Focus on Target Language**:
   * If the analysis is specific to English (e.g., keyword analysis, sentiment detection, or recommendation systems for English-speaking users), non-English words may introduce irrelevant data.
2. **Reduce Noise**:
   * Mixed-language content can confuse models, making it harder to extract meaningful patterns.
3. **Simplify Preprocessing**:
   * Ensures uniformity in the dataset, avoiding the need for additional steps like language detection or multilingual processing.
4. **Optimize Model Performance**:
   * Language-specific models (e.g., English sentiment analysis) perform best when trained and tested on text in the same language.
5. **Improved User Experience**:
   * For applications like search engines, chatbots, or e-commerce platforms catering to English speakers, excluding non-English content enhances relevance.

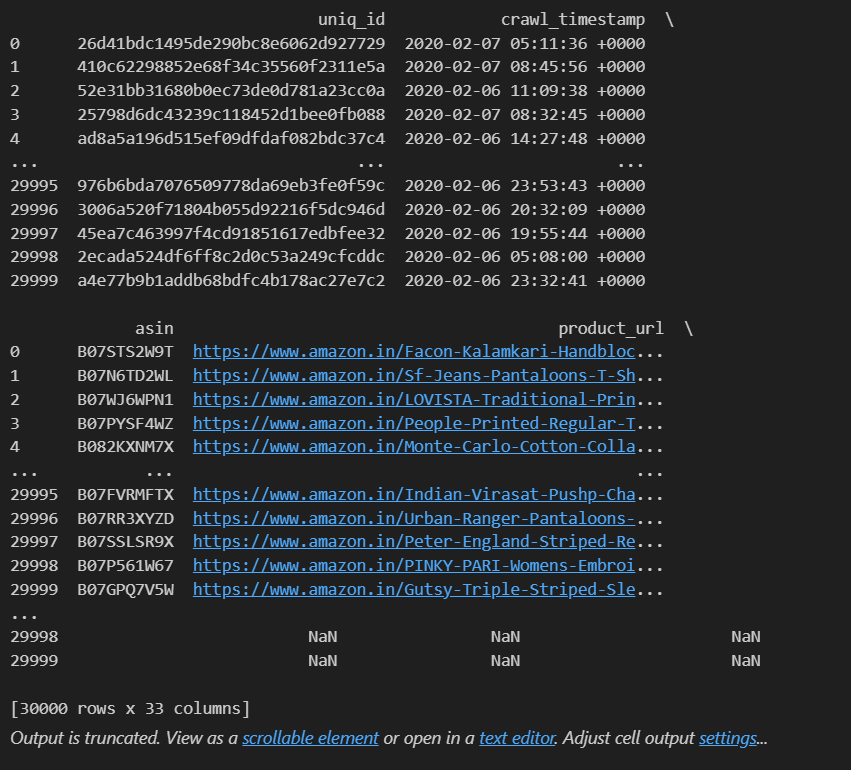




**4.Why Use Contractions?**

1. **Consistency**:
   * Standardizes text by replacing multiple representations of the same meaning (e.g., "do not" and "don't") with a single, consistent form.
2. **Alignment with Common Usage**:
   * Contractions are often used in casual text like product names, reviews, or social media content. Handling them improves relevance and user understanding.
3. **Improved Text Analysis**:
   * In tasks like sentiment analysis or language modeling, consistent contraction usage ensures better feature representation.





**5.Lemmatization**

**Lemmatization** is a natural language processing (NLP) technique used to reduce words to their base or root form, called the **lemma**. Unlike stemming, which often chops off word endings arbitrarily, lemmatization takes into account the word's context and ensures that the reduced form is a meaningful word in the language.

For example:

* The words **"running"**, **"ran"**, and **"runs"** would all be lemmatized to **"run"**.
* The word **"better"** might be lemmatized to its root form **"good"**, depending on the part of speech.

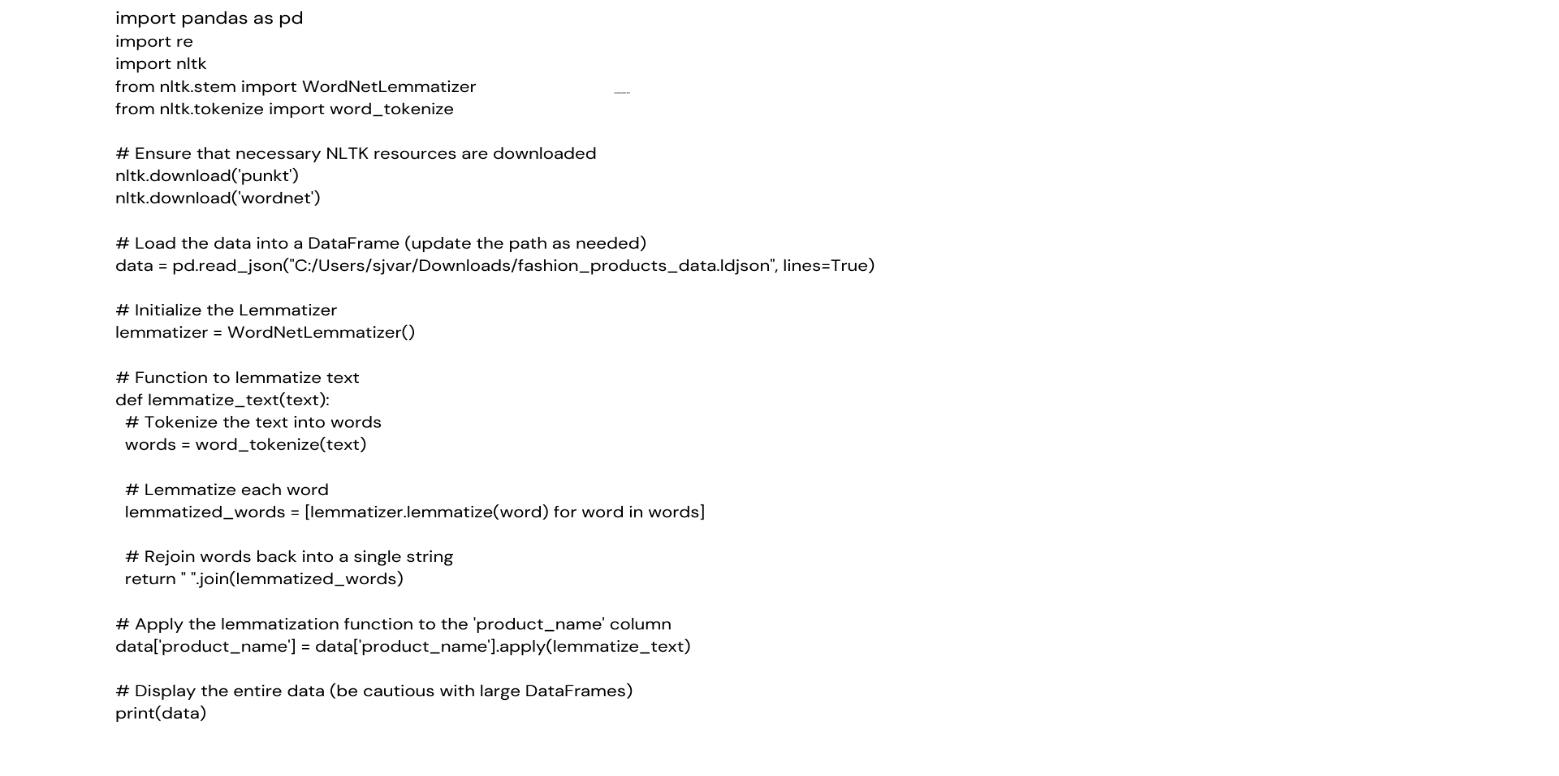
**Why Use Lemmatization?**

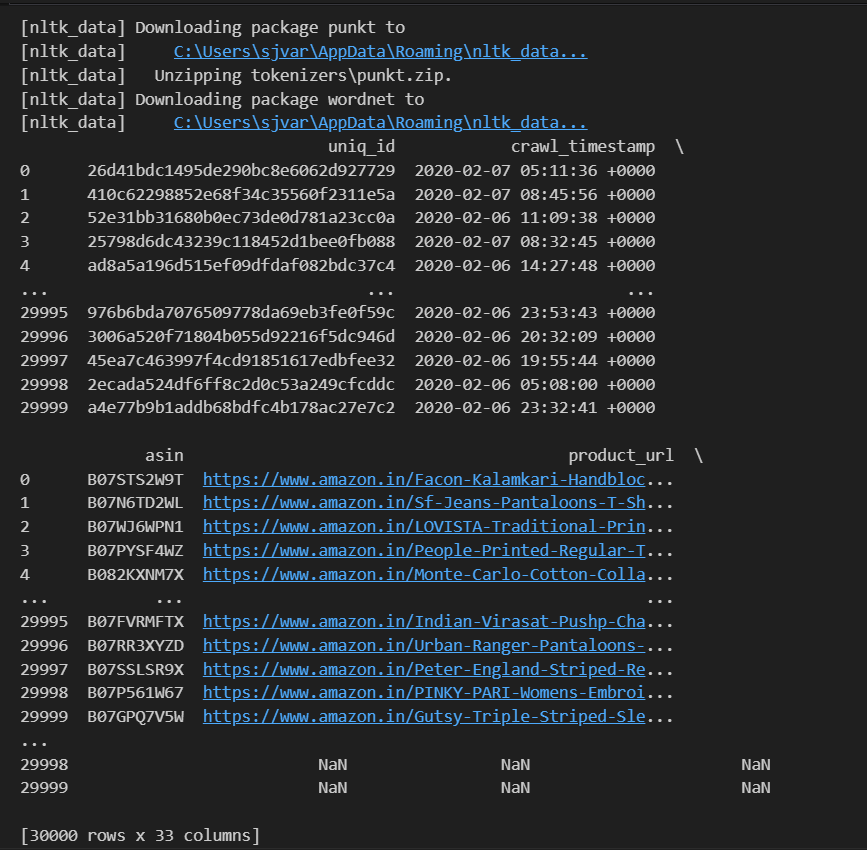
* To normalize words for better text analysis or preprocessing in NLP tasks.
* To reduce redundancy and variability in text data while retaining meaningful context.
* Helps improve the performance of tasks like sentiment analysis, information retrieval, and text classification.

In the provided code:

1. **WordNetLemmatizer** is used to perform lemmatization.
2. Each word in the product\_name column is tokenized and lemmatized individually.
3. Lemmatized words are rejoined into a single string and stored back in the column.

This process ensures that product names are standardized and easier to analyze, potentially reducing variability caused by different word forms.





* 1. **Content Based Filtering:**

**Overview of Recommendation Methods**

1. Bag of Words (BoW)
2. TF-IDF (Term Frequency-Inverse Document Frequency)
3. Word2Vec

**Bag of Words (BoW):**

**What is Bag of Words?**  
Bag of Words (BoW) is a natural language processing (NLP) technique used to represent text data as numerical features. It involves converting textual descriptions or documents into fixed-length vectors based on the frequency of each word in the text.

In the context of collaborative filtering, BoW helps to extract features from product descriptions (e.g., names, specifications) and use these features for similarity-based recommendations.

**Why Use Bag of Words?**

* **Simplicity:** BoW is easy to implement and understand.
* **Textual Feature Extraction:** Useful when product data includes textual descriptions.
* **Versatility:** Works well when paired with similarity measures like cosine similarity to find similar products.

**How Bag of Words Works**

1. **Tokenization**: Splits text into individual words or tokens.
2. **Vocabulary Creation**: Creates a unique set of words from the corpus.
3. **Vectorization**: Represents each document as a fixed-length vector based on word frequency or presence.

**Advantages of Bag of Words**

1. **Simple and Easy**: Easy to implement and understand.
2. **Effective Feature Extraction**: Useful for basic text analysis and recommendation systems.
3. **Versatile**: Works well with similarity measures like cosine similarity.

**Limitations of Bag of Words**

1. **Loses Context**: Ignores word order and semantics.
2. **High Dimensionality**: Requires large memory for extensive vocabularies.
3. **Sparse Data**: Results in sparse vectors, complicating computations.

**Code:**

A screen shot of a computer program

Description automatically generated

**Output:**

**Query Image:**

A person wearing a striped shirt

Description automatically generated

**Recommendations:**

A person wearing a striped shirt

Description automatically generatedA person in a black shirt

Description automatically generated

A person in a red shirt

Description automatically generatedA person in a green shirt

Description automatically generated

**TF-IDF (Term Frequency-Inverse Document Frequency)**

**Definition:**  
TF-IDF is a statistical technique used to measure the importance of a word in a document relative to a collection of documents (corpus). It helps to identify unique terms that are more relevant to a specific document while ignoring common terms across the corpus.

**How It Works:**

* **Term Frequency (TF):** Measures how often a word appears in a document.
* **Inverse Document Frequency (IDF):** Reduces the weight of common words across documents.

**Why Use TF-IDF in Recommendations?**

* Highlights unique words in product descriptions or reviews.
* Reduces noise from common words like "the" or "and."
* Improves similarity-based recommendations by providing better weightings.

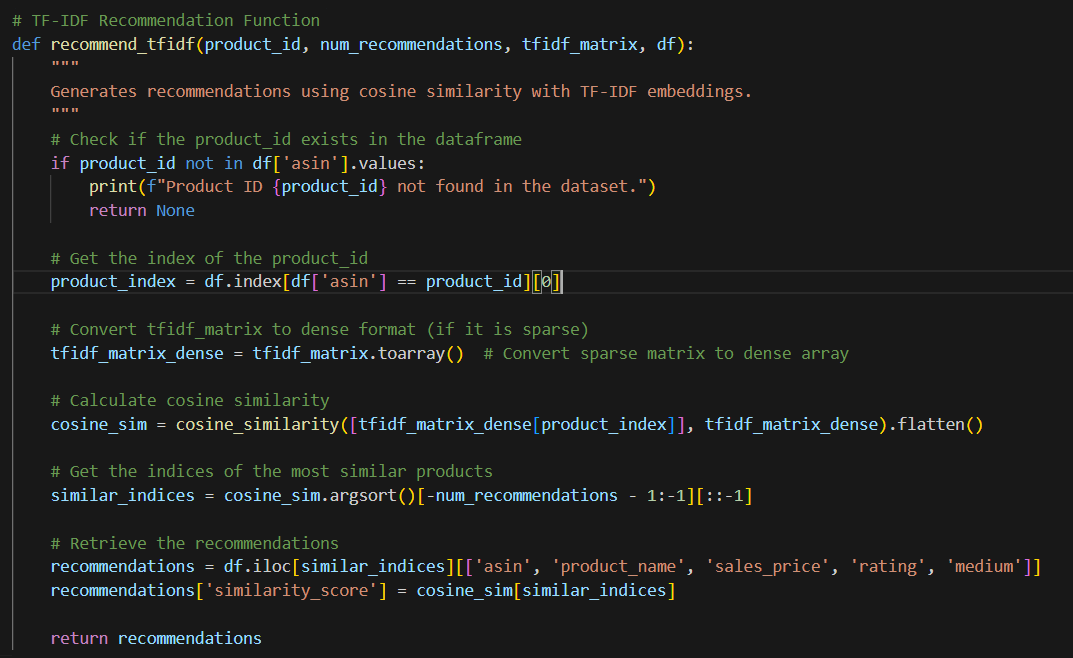
**Advantages:**

1. Balances word frequency and uniqueness.
2. Effective for text-heavy datasets.
3. Easy to implement and interpret.

**Limitations:**

1. Cannot capture the semantic meaning of words.
2. Computationally expensive for large datasets.

**Code:**

****

**Output:**

**Query Image:**

**A person in a striped shirt

Description automatically generated**

**Recommendations**

**A person wearing a blue shirt

Description automatically generatedA person wearing a grey and black shirt

Description automatically generated**A person in a blue shirt

Description automatically generated

**Word2Vec:**

**Definition:**  
Word2Vec is a word embedding technique that represents words as dense numerical vectors in a multi-dimensional space, capturing their semantic relationships. Developed using neural networks, Word2Vec learns the context of words based on their usage in a given text corpus.

**How Word2Vec Works:**

1. **Neural Network Training:**
   * Word2Vec uses either a **Skip-Gram** or **Continuous Bag of Words (CBOW)** model.
     + **Skip-Gram:** Predicts the context words from a target word.
     + **CBOW:** Predicts the target word from its surrounding context words.
2. **Vector Representation:**
   * Each word is mapped to a vector in a continuous vector space.
   * Words with similar meanings or contexts have vectors close to each other in this space.

**Why Use Word2Vec in Recommendations?**

* Captures semantic relationships (e.g., "king" and "queen" are close in vector space).
* Handles synonyms and context effectively.
* Provides a compact and meaningful representation of textual data.

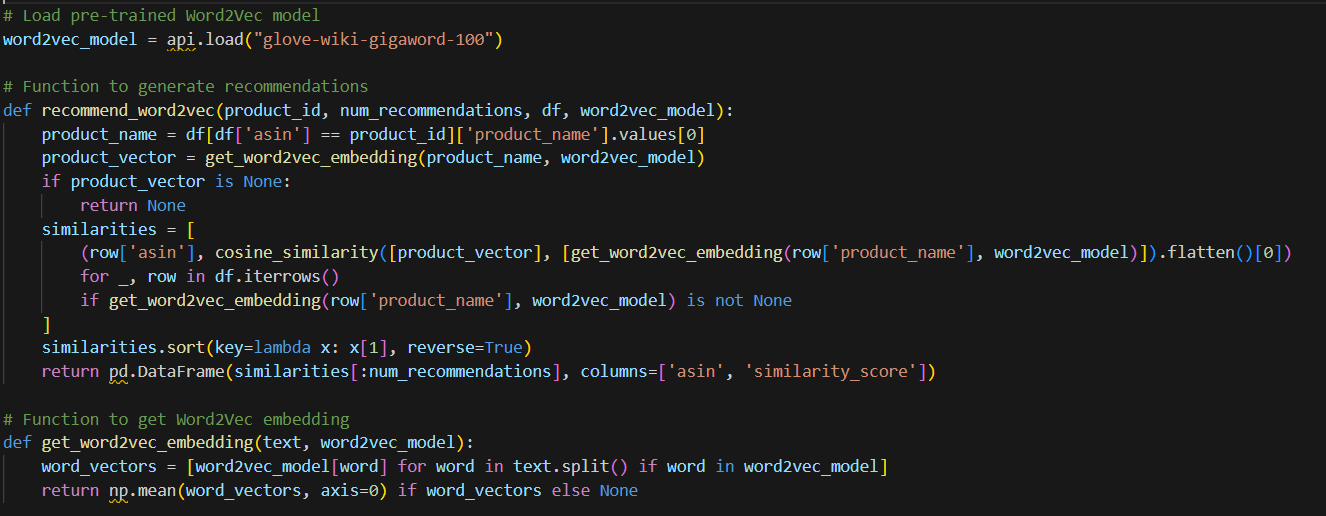
**Advantages of Word2Vec:**

1. Learns semantic and syntactic relationships.
2. Generates dense and compact word embeddings.
3. Handles large text datasets effectively.

**Limitations:**

1. Requires a large dataset for training to perform well.
2. Computationally intensive compared to simpler techniques like TF-IDF.

Code:



**Output:**

**Query Image**



A person in a blue dress

Description automatically generatedA person in a red dress

Description automatically generatedA person in a blue dress

Description automatically generated**Recommendations:1**

A person in a colorful dress

Description automatically generatedA person in a pink sari

Description automatically generated

**Hybrid Recommendation System Using TF-IDF and Word2Vec**

The code combines **TF-IDF** and **Word2Vec** to generate a hybrid recommendation system. This approach leverages the strengths of both methods:

* **TF-IDF** captures the relative importance of terms in product descriptions.
* **Word2Vec** captures the semantic relationships between words, enabling better context-based recommendations.

**Advantages of the Hybrid Approach**

* **Improved Accuracy:**  
  Combines term relevance from TF-IDF and semantic context from Word2Vec.
* **Versatility:**  
  Works well for both textual and semantic analysis.
* **Customization:**  
  Weighting parameters allow adjusting the contribution of TF-IDF and Word2Vec based on the dataset's characteristics.

Code:

A screen shot of a computer code

Description automatically generated

**Output:**

**Query Image**

A person in a black shirt

Description automatically generated

**Recommendations:**

A person in a black shirt

Description automatically generatedA person in a black shirt

Description automatically generatedA person in a black shirt

Description automatically generatedA person wearing a black shirt with a skull on it

Description automatically generatedA person in a white shirt

Description automatically generated

**Brand-Based Recommendation System**

The **brand-based recommendation system** generates recommendations by filtering products that belong to the same brand as the selected product. This method is useful for customers who have a strong preference for certain brands, ensuring the recommendations align with their brand loyalty.

**Key Points:**

1. Products are filtered based on the brand of the input product.
2. Recommendations are sorted by rating (if available) to prioritize highly-rated items.
3. Relevant product details like name, price, and rating are displayed along with images.

**Advantages:**

* Promotes brand loyalty by recommending similar products.
* Simple and efficient for real-time applications.
* Improves user experience by focusing on brand-specific products.

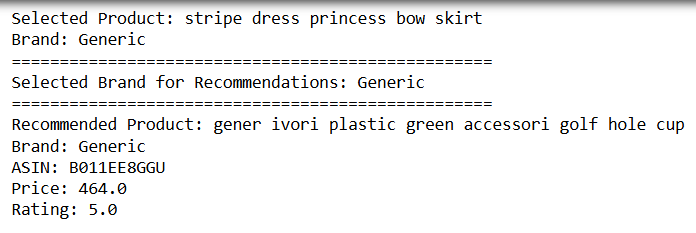
**Core Code:**

A screen shot of a computer program

Description automatically generated

**A white cylinder with a white background

Description automatically generatedOutput:**



A black dress with a red rose

Description automatically generatedA black text on a white background

Description automatically generatedA black text on a white background

Description automatically generated

A screen shot of a computer

Description automatically generated A white t-shirt with black text on it

Description automatically generated

A white background with black text

Description automatically generated A white shirt with black text on it

Description automatically generated

**Image-Based Recommendation System**

The **image-based recommendation system** recommends visually similar products by analyzing image features. A pre-trained deep learning model extracts meaningful features from images, which are then compared to determine similarity. This system enhances user experience by focusing on products that share similar visual aesthetics.

**Key Points:**

1. **Feature Extraction:** Uses a pre-trained EfficientNetB0 model to extract image features.
2. **Clustering:** Employs K-Means clustering to create synthetic labels for training a custom model.
3. **Similarity Computation:** Utilizes cosine similarity between image feature vectors to identify visually similar products.
4. **Recommendation:** Ranks products based on similarity scores and presents the top matches.

**Core Code:**

A screen shot of a computer program

Description automatically generated

**Output:**

**Query Image:**

****

**Recommendations:**

**** **A person in a red dress

Description automatically generated** A person in a pink and white dress

Description automatically generated A red and gold pajama

Description automatically generated

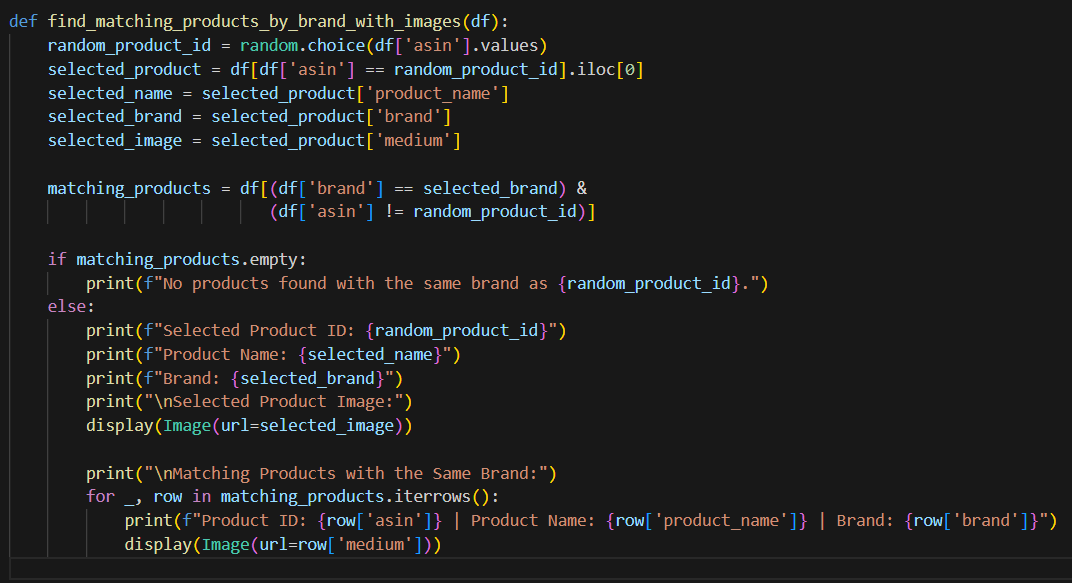
**Combination of Product Name and Brand-Based Recommendation**

This method combines product name and brand to enhance the relevance of recommendations. It identifies products from the same brand as the selected product and provides their details, including names and images. This approach ensures users find products with similar branding and potentially complementary features.

**Concept Highlights:**

1. **Selected Product:** A random or specific product is chosen from the dataset.
2. **Matching by Brand:** Filters out products of the same brand, excluding the selected product.
3. **Recommendation Details:** Displays matching product names, brands, and images for easy identification.

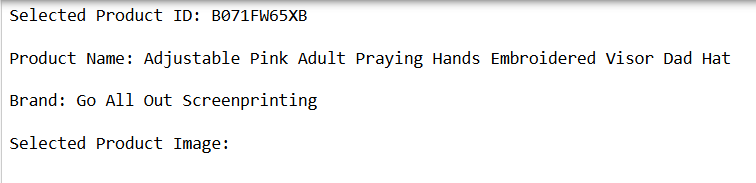
**Core Code:**



**Output:**

**Query Image:**

**A pink hat with a cross on it

Description automatically generated**

**A white visor with a pink ice cream cone on it

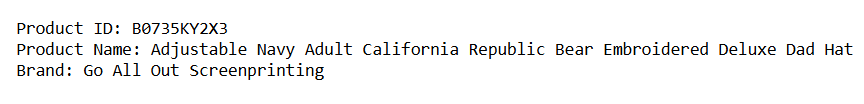
Description automatically generated**

**A white background with black text

Description automatically generated**

****

**A close up of words

Description automatically generatedA close up of a text

Description automatically generated** ****

****

1. **Collaborative Filtering**

**Overview**

Recommendation systems are essential tools in e-commerce, streaming platforms, and other industries. They analyze user preferences and product attributes to suggest items users are likely to enjoy. This report delves into collaborative filtering as the core approach, augmented by advanced techniques leveraging pre-trained deep learning models like VGG16 and MobileNet for embedding generation and product recommendations.

**Collaborative Filtering**

**Definition**

Collaborative filtering (CF) is a recommendation approach that predicts a user’s preference for an item based on past interactions and the preferences of similar users or items. It operates under the assumption that users who agreed in the past will agree in the future.

**Types of Collaborative Filtering**

**1. User-Based Collaborative Filtering**

* **Methodology**: Finds users with similar preferences ("neighbors") and recommends items preferred by these neighbors.
* **Advantages**:
  + Easy to understand and implement.
  + Works well for users with many interactions.
* **Challenges**:
  + Suffers from scalability issues in large datasets.
  + Cold start problem for new users.

**2. Item-Based Collaborative Filtering**

* **Methodology**: Identifies items that are similar based on user interaction patterns. Items similar to what a user has previously liked are recommended.
* **Advantages**:
  + More stable than user-based filtering.
  + Handles new users better than user-based approaches.
* **Challenges**:
  + Requires efficient computation of item similarity.

**Metrics Used**

* **Cosine Similarity**: Measures the cosine of the angle between two non-zero vectors (user/item embeddings). Values range from −1 to 1, where higher values indicate greater similarity.

**Challenges of Collaborative Filtering**

* **Cold Start Problem**: New users or items lack interaction data.
* **Data Sparsity**: Many users interact with only a small subset of items.
* **Scalability**: Computational demands grow with larger datasets.

**Image-Based Recommendations Using Pre-Trained Models**

**Why Image-Based Recommendations?**

Collaborative filtering relies heavily on user interaction data, but product attributes like visual features can provide additional insights. By using pre-trained models like VGG16 and MobileNet, we can extract meaningful embeddings from product images to augment recommendations.

**Pre-Trained Models**

**1. VGG16**

* **Description**: VGG16 is a deep convolutional neural network architecture developed by the Visual Geometry Group (VGG) at Oxford. It has 16 weight layers and is known for its simplicity and effectiveness.
* **Features**:
  + Uses a uniform architecture with small 3x3 filters.
  + Trained on the ImageNet dataset.
  + Outputs feature maps that are effective for image-related tasks.
* **Advantages**:
  + Produces high-quality embeddings suitable for diverse tasks.
  + Well-documented and widely supported.
* **Challenges**:
  + Computationally expensive due to its depth.
  + Larger model size compared to alternatives like MobileNet.
* **Code:**

**A white rectangular box with black text and green text

Description automatically generated with medium confidence**



**Output:**

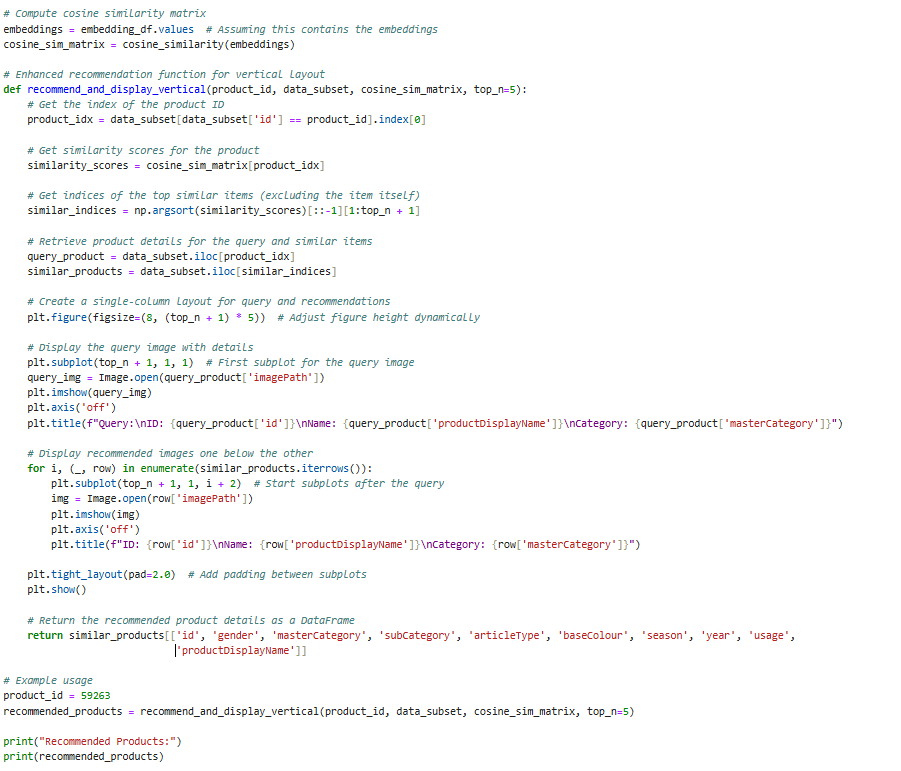
****

**2. MobileNet**

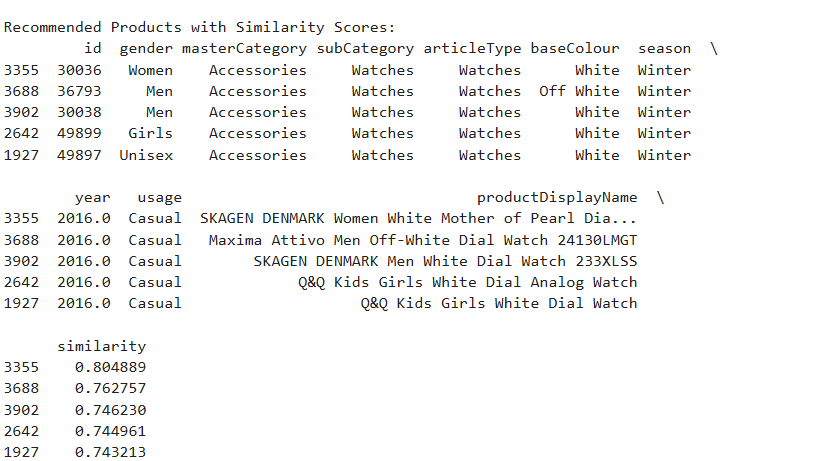
* **Description**: MobileNet is a lightweight convolutional neural network designed for mobile and embedded vision applications.
* **Features**:
  + Uses depthwise separable convolutions to reduce computational cost.
  + Trained on the ImageNet dataset for a variety of vision tasks.
* **Advantages**:
  + Optimized for speed and efficiency.
  + Suitable for deployment on resource-constrained devices.
* **Challenges**:
  + May not perform as well as larger models like VGG16 for some tasks.
* **Code:**

**A screenshot of a computer

Description automatically generated**



**Output:**

****

**Implementation Steps**

**1. Feature Extraction**

* **Objective**: Extract meaningful image embeddings using VGG16 and MobileNet.
* **Process**:
  1. Load the pre-trained model (VGG16 or MobileNet) with weights trained on ImageNet.
  2. Remove the top classification layer to retain only the feature extraction layers.
  3. Apply a global pooling layer to convert feature maps into compact embeddings.
  4. Generate embeddings for all product images.
* **Output**: A numerical vector (embedding) representing each image.

A screenshot of a computer

Description automatically generated

**2. Saving Embeddings**

* **Storage**: Save embeddings in a CSV or .npy file, aligning each embedding with its corresponding product ID and metadata.
* **Advantages**: Allows efficient similarity computations and avoids repetitive extraction.

**3. Computing Similarity**

* **Method**: Use cosine similarity to compare embeddings and identify visually similar products.
* **Implementation**:
  + Compute pairwise similarity between embeddings.
  + Rank items based on similarity scores.

**Code**:



**Outputs:**

**A white background with black numbers

Description automatically generated**

**4. Recommendation Pipeline**

1. **Collaborative Filtering**:
   * Generate an initial list of recommendations using user or item interactions.
2. **Image-Based Refinement**:
   * For each recommended item, identify visually similar products using embeddings.
   * Integrate metadata (e.g., product category, brand) for contextual relevance.
3. **Display Results**:
   * Show recommended products along with their details (name, category, brand, similarity score).

**Evaluation and Advantages**

**Evaluation Metrics**

* **Accuracy**: Measure how well the system predicts user preferences.
* **Diversity**: Assess the variety in recommendations.
* **User Satisfaction**: Evaluate the relevance and appeal of recommendations.

**Advantages of Combining Collaborative Filtering and Image-Based Methods**

* **Cold Start Solution**: Image embeddings provide insights for new items with limited interactions.
* **Enhanced Relevance**: Combines behavioral data with visual cues for better recommendations.
* **Scalability**: Efficient embedding storage and retrieval allow for large-scale deployment.

**Conclusion**

This hybrid approach, leveraging collaborative filtering and pre-trained deep learning models like VGG16 and MobileNet, addresses common challenges in recommendation systems. Collaborative filtering provides a foundation based on user interactions, while image-based methods add depth by incorporating visual features. Together, these techniques create a robust, versatile recommendation pipeline suitable for diverse applications in e-commerce and beyond.