Created by: Aritro Sen
aritro1011@gmail.com +91 9022314977

Recommendation System for Ecommerce Website using Content based Filtering

Git repo: https://github.com/aritro1011/E-CommerceRecommendationUsingContentBasedFiltering

Dataset link: https://www.kaggle.com/datasets/karkavelrajaj/amazon-sales-dataset

Drive folder: https://drive.google.com/drive/folders/130c-FootpXZjZ1j8BVCHK9lBnCsM0HoE?usp=sharing

Introduction

Overview of the project

A recommendation system for e-commerce using content-based filtering that suggests products based on user preferences and item features, analysing product descriptions and other metadata.

The Problem Statement also requires a real-time aspect where the results are based on user's search history or purchase history which can be simply created by

Capturing User Behaviour

- **Search History**: Track the keywords, product categories, or product IDs the user searches for.
- **Purchase History**: Maintain a record of the products purchased by the user, including metadata like product category, brand, and description.

Implementation Options:

- Use a **database** (e.g., MongoDB, MySQL) to store user behaviour.

Post that the product_id or any other feature can extracted to make the rest of the program work the way it is working now.

As of now the Flask WebApp requires user input in the format of product_id from the dataset which has been used.

Objective of the recommendation system

The objective of the recommendation system in an e-commerce setting is to personalize user experience by suggesting relevant products based on the user's preferences, behaviour, and historical interactions. the goals of the recommendation system:

- Increase Conversion Rate: Encourage users to make a purchase by presenting them with products they are more likely to be interested in.
- Enhance User Experience**:** Improve the overall shopping experience by offering personalized recommendations that align with individual user needs.
- Improve Product Discovery: Help users discover new and relevant products that they might not have encountered through traditional browsing or search methods.
- Maximize Customer Retention: Keep customers engaged by offering products tailored to their preferences, leading to repeat visits and long-term loyalty.
- Optimize Inventory Management**:** Promote products that are overstocked or are slow-moving, based on user behaviour and demand trends.

Why content-based filtering?

Created by: Aritro Sen
aritro1011@gmail.com +91 9022314977

Content-based filtering is a popular method for recommendation systems, especially in environments like e-commerce, where the objective is to recommend products based on the characteristics or features of the items themselves, rather than relying on collaborative behaviour (such as what other users have liked)

Understanding Content-Based Filtering

What is Content-Based Filtering?

Content-based filtering is a technique used in recommendation systems where recommendations are made based on the features or attributes of items that a user has shown interest in, such as products, movies, books, etc. The system suggests items that are similar to those the user has interacted with, based on the content characteristics (e.g., keywords, genres, categories).

In simpler terms, content-based filtering recommends items based on what the user has liked or interacted with in the past, focusing on the attributes of the items themselves rather than the behaviour of other users.

How it works

Profile Building:

- First, the system creates a profile of the user's preferences based on the items they have interacted with (e.g., products they've viewed, purchased, or rated highly).
- These items are described by various attributes like keywords, tags, categories, or features (e.g., genre of movies, colour and brand of clothing, features of an electronic device).

Item Representation:

- Items are represented by their content features (e.g., a movie can be described by its genre, cast, and director, while a product can be described by its brand, price, and category).

Similarity Calculation:

- The system computes the similarity between the items that the user has interacted with and other items in the catalogue, based on their features.
- This is typically done using techniques like cosine similarity or Euclidean distance to compare feature vectors (which represent item characteristics).

Recommendation Generation:

- Items that are similar in content to the user's previous interactions are then recommended.

Advantages and limitations

Advantages of Content-Based Filtering:

1. Personalization: Recommendations are directly tailored to the user's interests based on their past behaviour.
2. No Need for Large User Base: It works well even for new users or with limited data, unlike collaborative filtering, which relies on a large user base to make recommendations.

3. Transparency: The system can explain the reason behind the recommendations, which can enhance trust with users.
4. Works for Niche Preferences: Even if a user has specific tastes, content-based filtering can recommend items that match those tastes without requiring data from other users.

Limitations:

1. Limited Diversity: It may recommend only items similar to those the user has already interacted with, potentially leading to a lack of diversity in suggestions.
2. Requires Detailed Content Information: The effectiveness of content-based filtering depends on the availability of rich metadata about items.
3. Cold Start Problem for New Items: It may struggle to recommend newly added items that don't have enough interaction data.

Problem Statement

[Real-Time Recommendation System: • Design a real-time recommendation system for an e-commerce platform. The system should provide product recommendations based on the user's recent browsing or purchase history using collaborative filtering or content-based filtering.]

This was the task I received from Settyl as an Assignment on Internshaala for a Machine Learning Internship.

Dataset Description

Source of the dataset

Name: amazon-sales-dataset

Format: comma-separated-values(.csv)

[https://www.kaggle.com/datasets/karkavelrajaj/amazon-sales-dataset]

Description of the columns

- product_id - Product ID
- product_name - Name of the Product
- category - Category of the Product
- discounted_price - Discounted Price of the Product
- actual_price - Actual Price of the Product
- discount_percentage - Percentage of Discount for the Product
- rating - Rating of the Product
- rating_count - Number of people who voted for the Amazon rating
- about_product - Description about the Product

- user_id - ID of the user who wrote review for the Product
- user_name - Name of the user who wrote review for the Product
- review_id - ID of the user review
- review_title - short review
- review_content - long review
- img_link - Image Link of the Product
- product_link - Official Website Link of the Product

Preprocessing the Data

Data cleaning

Handling Missing Values:

Missing values in the dataset were handled appropriately. Depending on the nature of the data:

- For numerical features (e.g., prices, ratings), missing values were either filled with the mean, median, or by applying domain-specific knowledge (such as zero for a missing discount).
- For categorical features (e.g., product categories, brands), missing values were either filled with a placeholder (e.g., "Unknown") or dropped if there was insufficient data.

Removing Duplicates:

Duplicate rows, which could distort the recommendation results, were identified and removed. This ensured there were no redundant entries for the same product or user interaction.

Correcting Data Types:

Columns with incorrect data types were converted to the appropriate format. For instance:

- Date columns were converted to datetime format.
- Price columns were converted to float or int to allow for numerical operations.

Data Pre-Processing

Text preprocessing

- Lowercased all text to maintain consistency.
- Removed punctuation, numbers, and special characters that don’t contribute to the analysis.
- Removed stop words (commonly used words like "the", "is", "in")
- using predefined stop word lists.
- Applied stemming or lemmatization to reduce words to their root form

Numerical data normalization

Used Standard Scaler to scale all the numerical features

Created by: Aritro Sen
aritro1011@gmail.com +91 9022314977

Feature Engineering

- Extracted features from textual data using TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) is a text vectorization technique that evaluates the importance of a word in a document relative to its frequency in a corpus.

Formula:

1. Term Frequency (TF):

$$\text{TF}(t,d) = \frac{\text{Number of times term t appears in document d}}{\text{Total number of terms in document d}}$$

2. Inverse Document Frequency (IDF):

$$\text{IDF}(t) = \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing term t}}\right)$$

3. TF-IDF:

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \times \text{IDF}(t)$$

This method assigns higher importance to terms that appear frequently in a specific document but are rare across the entire corpus.

Building the Recommendation System

Explanation of the algorithm

The content-based recommendation system works by suggesting items that are similar to what the user has previously shown interest in, based on item features such as product descriptions, categories, and other attributes. The system calculates the similarity between items and recommends the ones that are most similar to the items a user has interacted with, based on their features.

Step-by-Step Process:

1. Feature Extraction:
    - First, the relevant features of each item (e.g., product description, category, price, etc.) are identified.
    - For each product, these features are represented numerically. Textual features (like product description) are typically vectorized using TF-IDF (Term Frequency-Inverse

Document Frequency), while other features like category and brand might be encoded using one-hot encoding or label encoding.

- Cosine Similarity Calculation: Cosine similarity is a measure used to calculate how similar two items are based on their vectorized features. It measures the cosine of the angle between two vectors in the feature space, ranging from -1 (completely dissimilar) to 1 (completely similar).

$$\text{Cosine Similarity}(A, B) = \frac{A \cdot B}{\|A\|\|B\|}$$

where:

- $A \cdot B$ is the **dot product** of the two vectors.
- $\|A\|$ and $\|B\|$ are the **magnitudes** (or Euclidean norms) of the vectors.

Recommendation Generation:

- After calculating cosine similarities for all items, the system sorts these items in descending order of similarity to the target item.
- The top **N most similar items** are recommended to the user, based on their interaction history or the item they are currently interested in.

The features used for the cosine similarity matrix are:

1. about_product
2. review_content
3. category
4. discounted_price
5. actual_price
6. discount_percentage
7. rating
8. ratings_count

Workflow of the recommendation system

- Input: User query or product ID
- Process: Calculating similarity (cosine similarity)
- Output: Top N recommended products

8. Implementation

- Tools and libraries used:

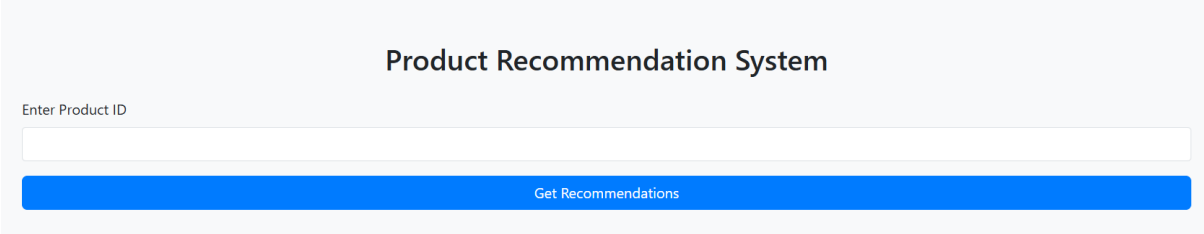
1. Pandas
2. NumPy
3. Sci-kit learn
4. Flask
5. nltk

Created by: Aritro Sen
aritro1011@gmail.com +91 9022314977

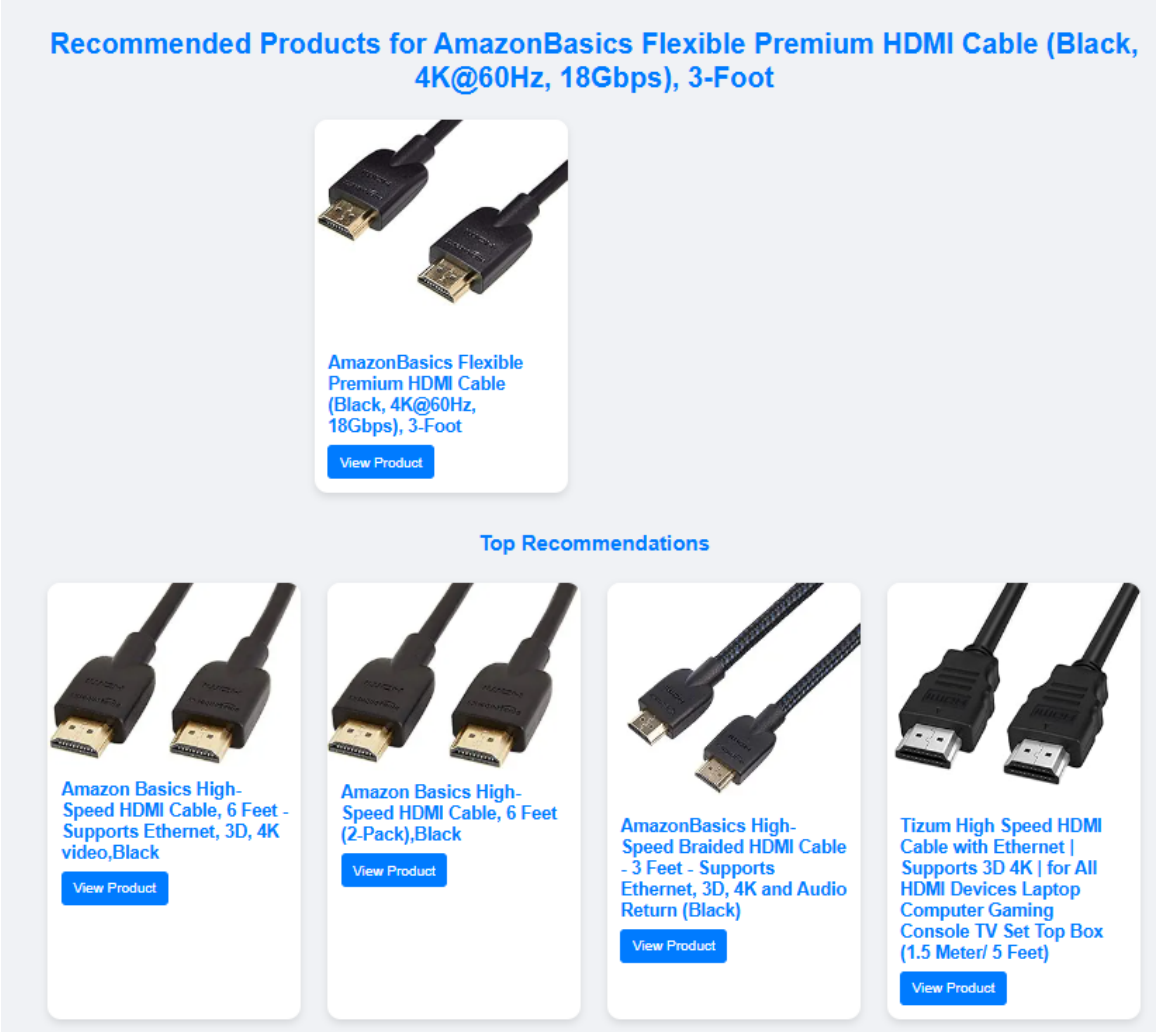
Output/Results

- Examples of recommendations for different products:
  - Since a basic webapp has been created with Flask, the output is presented in a well structured yet very basic manner.
  - Since the Dataset contains very old data too, some of the img_links are corrupted.

Landing Page:

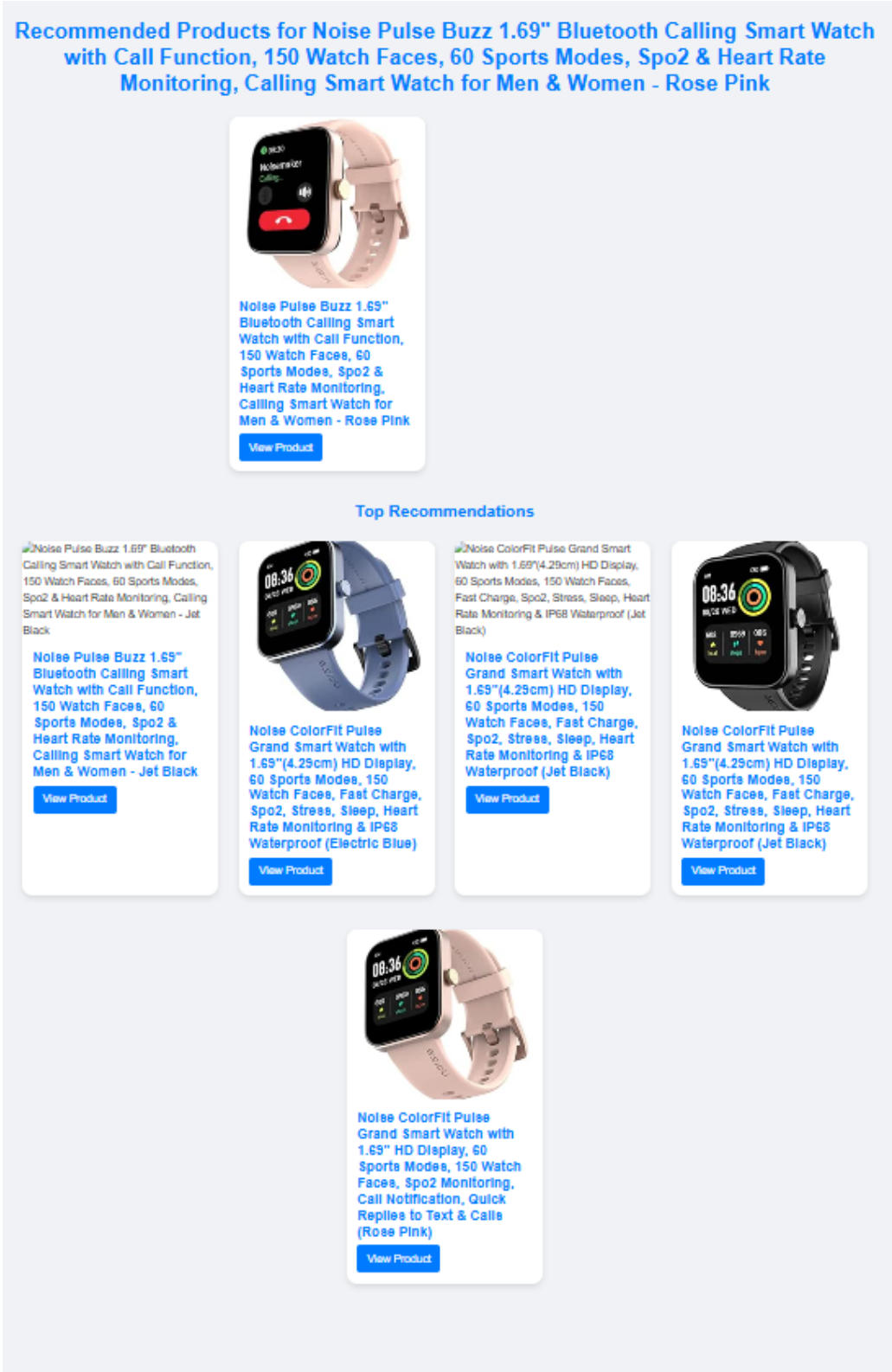


1)Product_id=' B07KSMBL2H'



Created by: Aritro Sen
aritro1011@gmail.com +91 9022314977

2) Product_id= ‘B09PLFJ7ZW’



Created by: Aritro Sen
aritro1011@gmail.com +91 9022314977

Challenges and Solutions

Issues Faced During Development:

1. Data Quality and Completeness:
    - Challenge: The data used for building the recommendation system contained incomplete or inconsistent records, especially for product descriptions and categories.
    - Impact: Missing or incomplete data could have led to inaccurate recommendations and affected the system's overall performance.
2. Feature Engineering and Vectorization:
    - Challenge: Converting unstructured text data (like product descriptions) into numerical features for machine learning models was time-consuming and computationally expensive.
    - Impact: Inadequate feature extraction could have resulted in poorly represented item characteristics, negatively affecting the recommendation accuracy.
3. Cosine Similarity Calculation:
    - Challenge: Calculating cosine similarity between products was slow, especially as the product catalogue grew, leading to delays in generating recommendations.
    - Impact: Longer processing times reduced the responsiveness of the system and hindered real-time recommendations.

Steps Taken to Resolve Them:

1. **Improved Data Preprocessing**:
    - **Solution**: Cleaned and pre-processed the data by handling missing values, standardizing formats, and filling in gaps where possible using techniques like imputation or default values for missing fields.
    - **Outcome**: Enhanced the quality and completeness of the dataset, leading to better feature representation and more accurate recommendations.
2. **Efficient Feature Vectorization**:
    - **Solution**: Used **TF-IDF** for text vectorization to efficiently represent product descriptions as numerical features, and applied **one-hot encoding** for categorical variables like brand and category.
    - **Outcome**: Created a more efficient and compact feature representation, enabling faster processing and better handling of textual data.
3. **Optimized Cosine Similarity Calculation**:

- **Solution**: Implemented **pre-computed similarity matrices** for commonly viewed or related items, allowing for faster retrieval of similar items without recalculating similarity on the fly for every request.
- **Outcome**: Significantly reduced the computational load and processing time for similarity calculations, improving system response times.

Future Scope

The real time element of the problem statement as mentioned in the introduction can be implemented (with the correct user interaction data)

More Filtering methods can be implemented (like collaborative filtering)

Weights can be added to features to give more priority and importance to specific features more than others

Conclusion

Summary of results

The content-based recommendation system successfully achieved its goal of providing relevant product recommendations to users based on their interaction history and item features. Key outcomes include:

Accurate Recommendations: The system effectively analysed product features such as descriptions, categories, and price to identify and recommend items similar to those previously viewed or purchased by users. The use of **TF-IDF** for text-based features and **cosine similarity** for comparing items ensured that the recommendations were highly relevant.

Data Preprocessing: By addressing data quality issues and implementing effective preprocessing techniques, the system was able to handle missing or inconsistent data, contributing to more reliable recommendations.

Final thoughts

The content-based filtering recommendation system has proven to be a powerful tool for personalizing product suggestions in an e-commerce platform. The combination of feature extraction, TF-IDF vectorization, and cosine similarity allows for a robust and scalable system. However, there are still opportunities for improvement, particularly in enhancing the handling of more complex user preferences and incorporating more diverse data sources for even richer recommendations. Future work could explore the integration of **collaborative filtering** methods to combine the strengths of both content-based and collaborative approaches, thereby further enhancing the accuracy and relevance of recommendations. Additionally, ongoing monitoring and optimization of the system will be essential to maintain its performance as the user base and product catalogue grow.

Created by: Aritro Sen
aritro1011@gmail.com +91 9022314977

Note: Kindly provide feedback to the project, even if my candidature isn't considered as it helps improve my work and helps me understand areas I require improvement in.