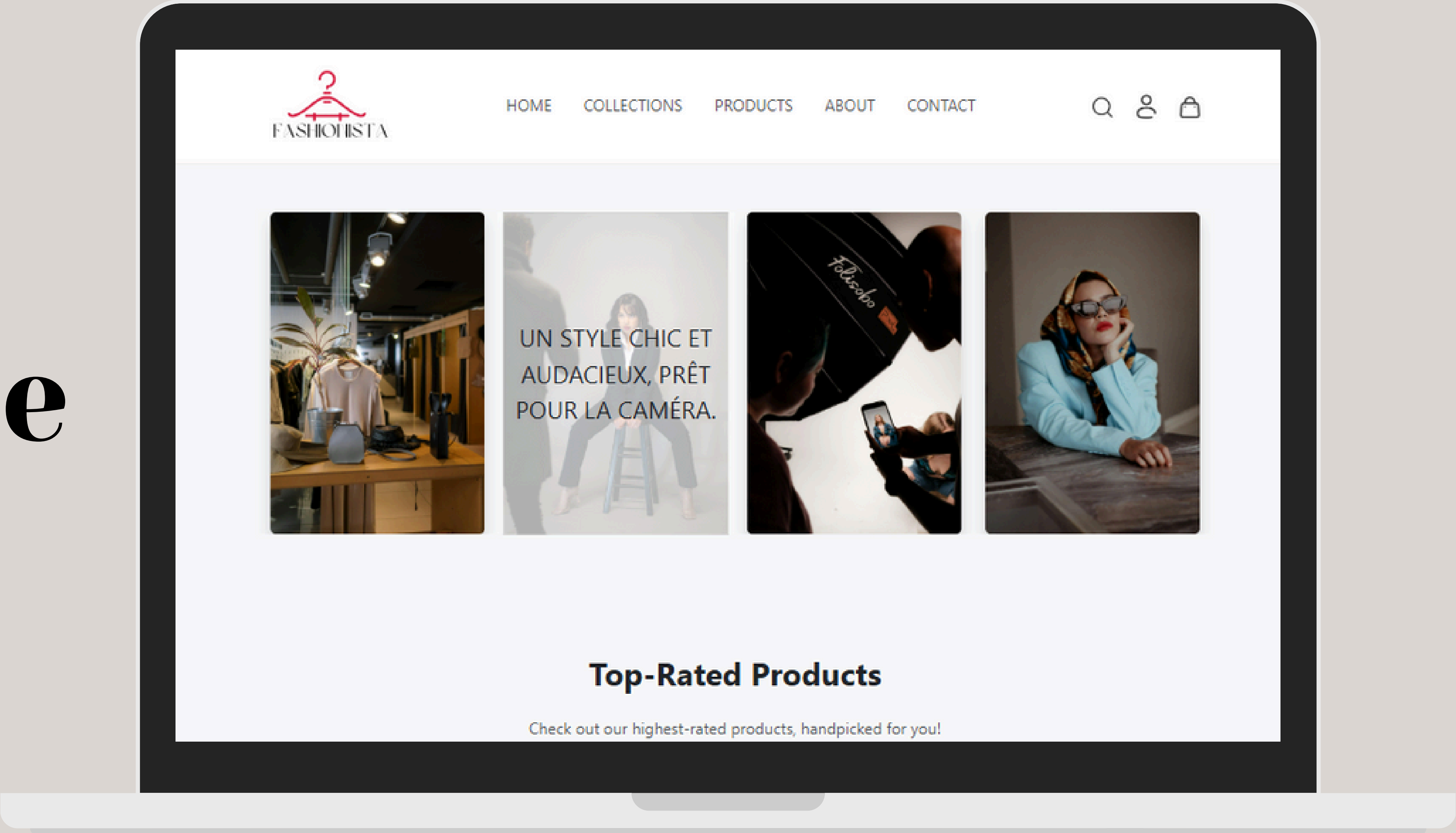
FASHIONISTA

Smart E-Commerce Platform



UN STYLE CHIC ET AUDACIEUX, PRÊT POUR LA CAMÉRA.

Top-Rated Products

Check out our highest-rated products, handpicked for you!

By: KALACH SIHAM
Supervised By :M. AZiZ KHAMJANE

Abstract

No Sells Yet ?

New Website ?

No Users ?

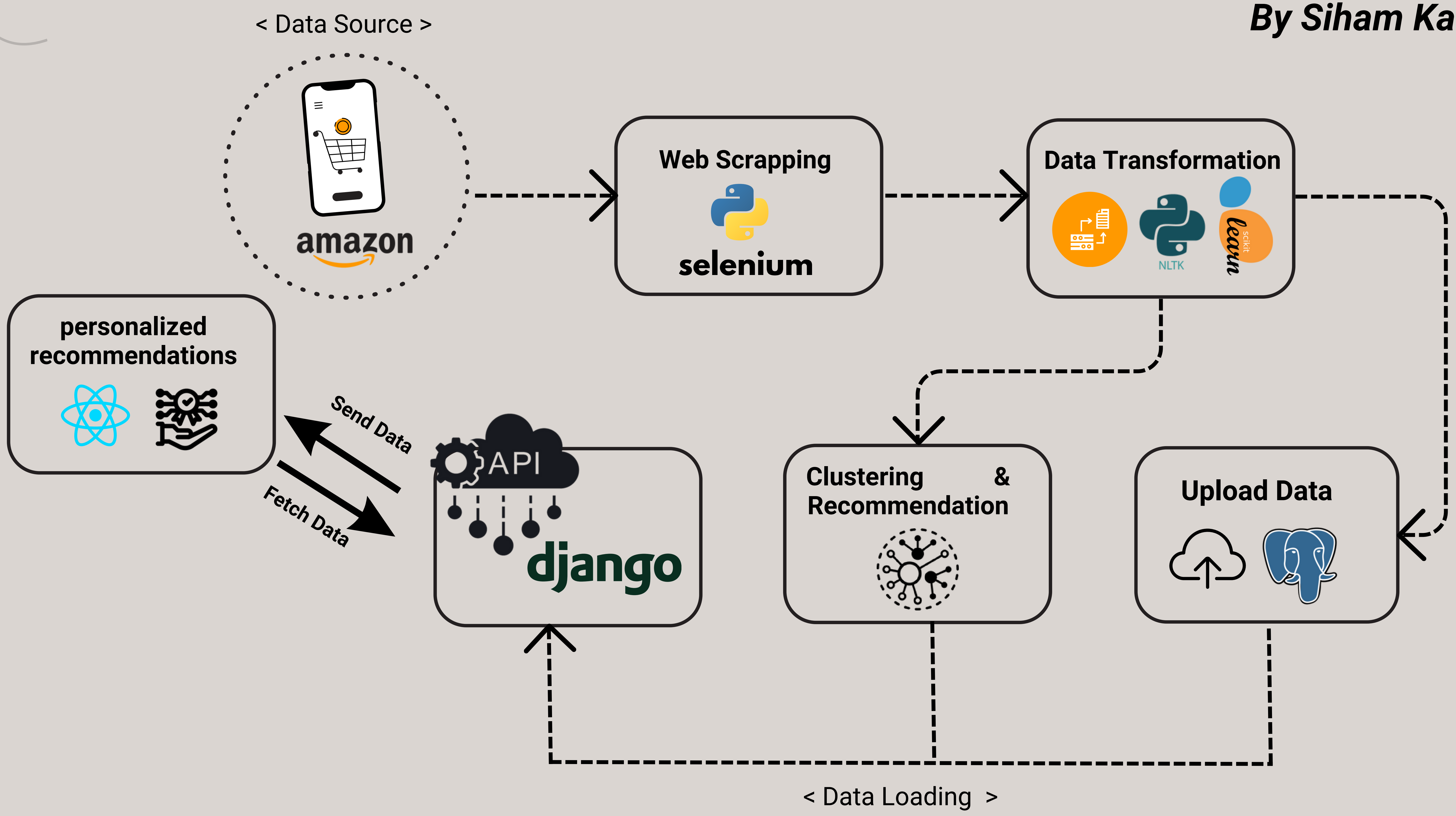
No Historical User-Item Interaction ?

You Want To Recommend To Your Client ?

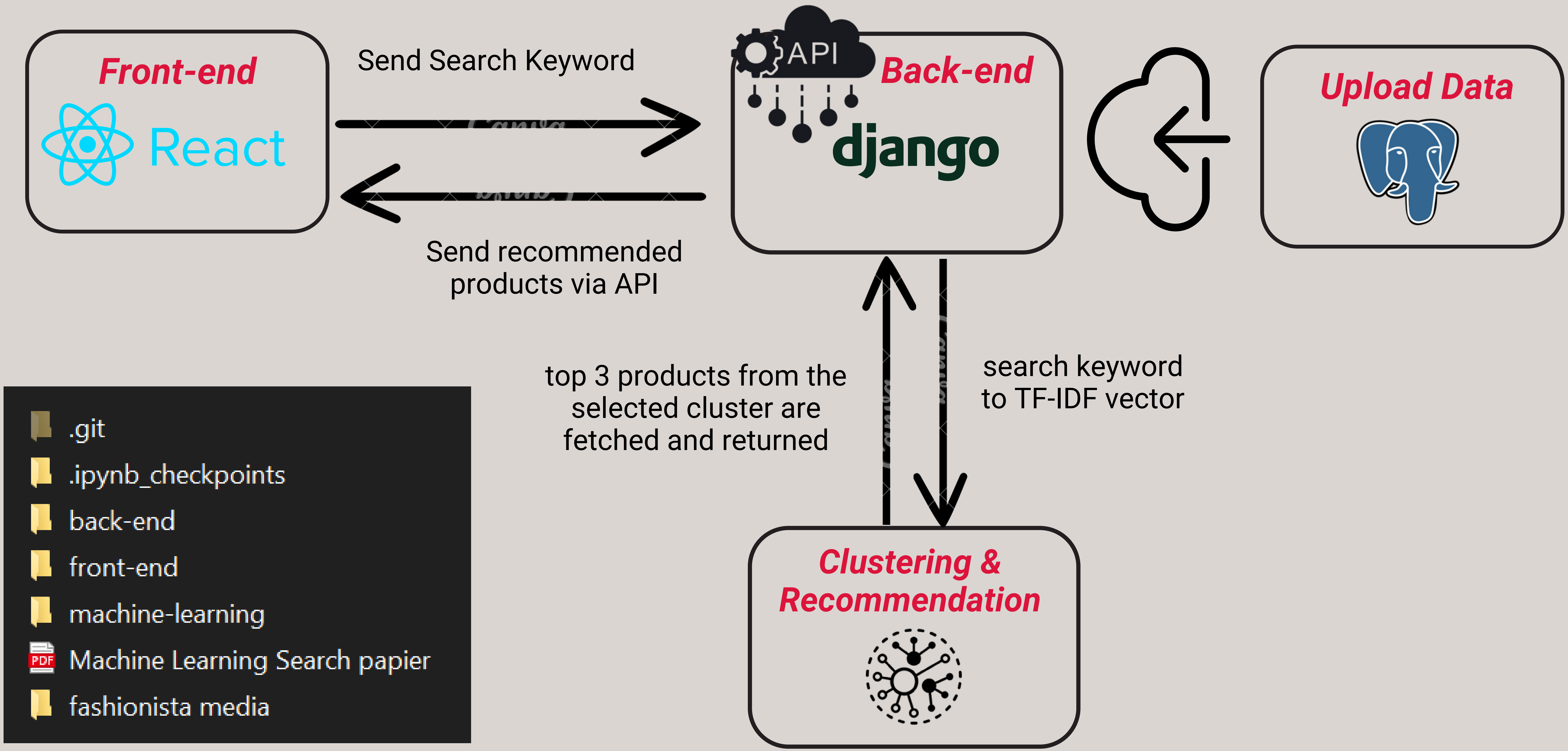
THE ARCHITECTURE

Pipeline Architecture for Smart FASHIONISTA E-Commerce Platform

By Siham Kalach



By Siham Kalach



Front-end

React

Send Search Keyword

Send recommended products via API

Back-end

API

django

Upload Data

top 3 products from the selected cluster are fetched and returned

search keyword to TF-IDF vector

Clustering & Recommendation

.git

.ipynb_checkpoints

back-end

front-end

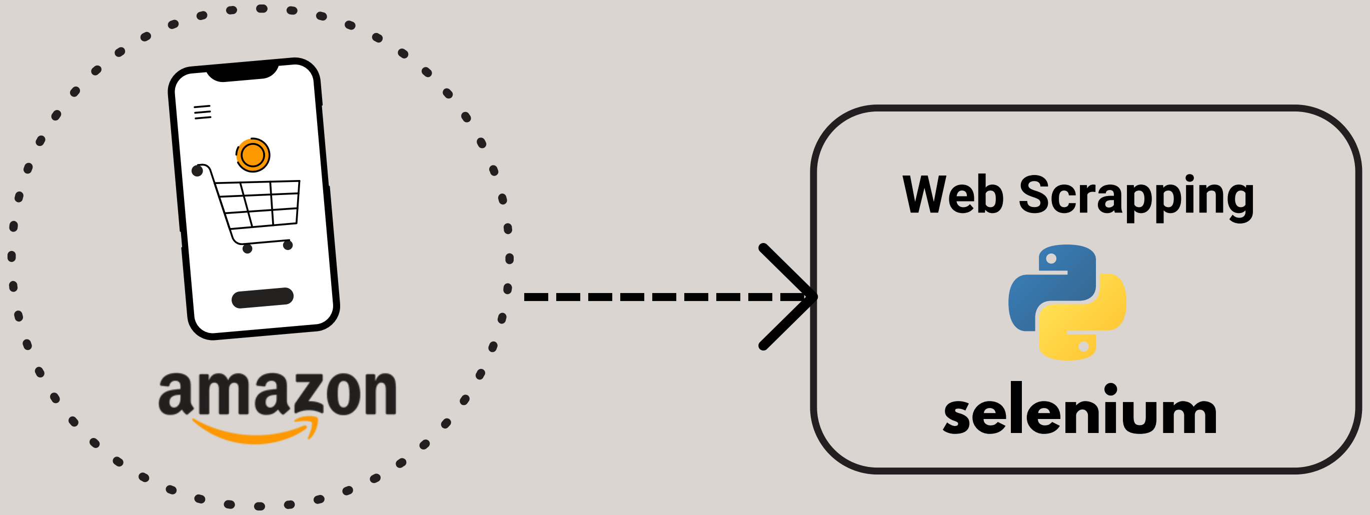
machine-learning

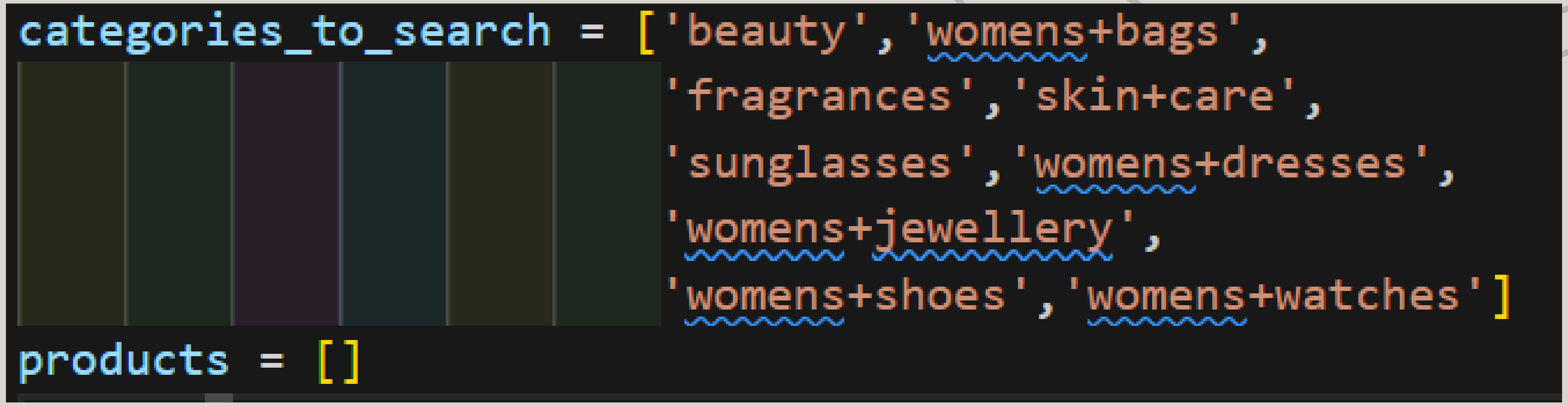
Machine Learning Search papier

fashionista media

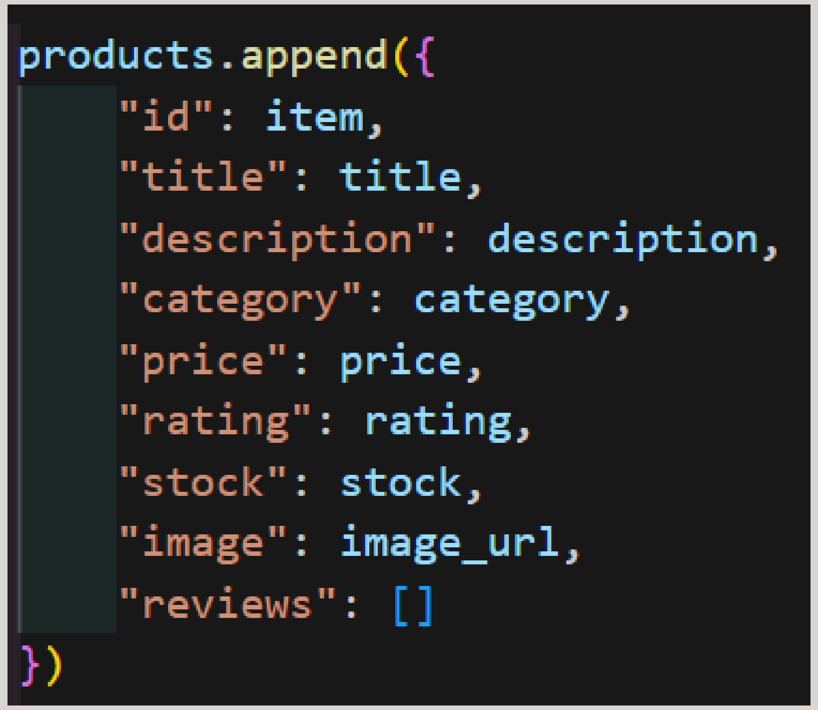
Data Collection

< Data Source >





```
categories_to_search = ['beauty','womens+bags',
                        'fragrances','skin+care',
                        'sunglasses','womens+dresses',
                        'womens+jewellery',
                        'womens+shoes','womens+watches']
products = []
```



```
products.append({
    "id": item,
    "title": title,
    "description": description,
    "category": category,
    "price": price,
    "rating": rating,
    "stock": stock,
    "image": image_url,
    "reviews": []
})
```

Data Transformation and Cleaning

Price Formatting

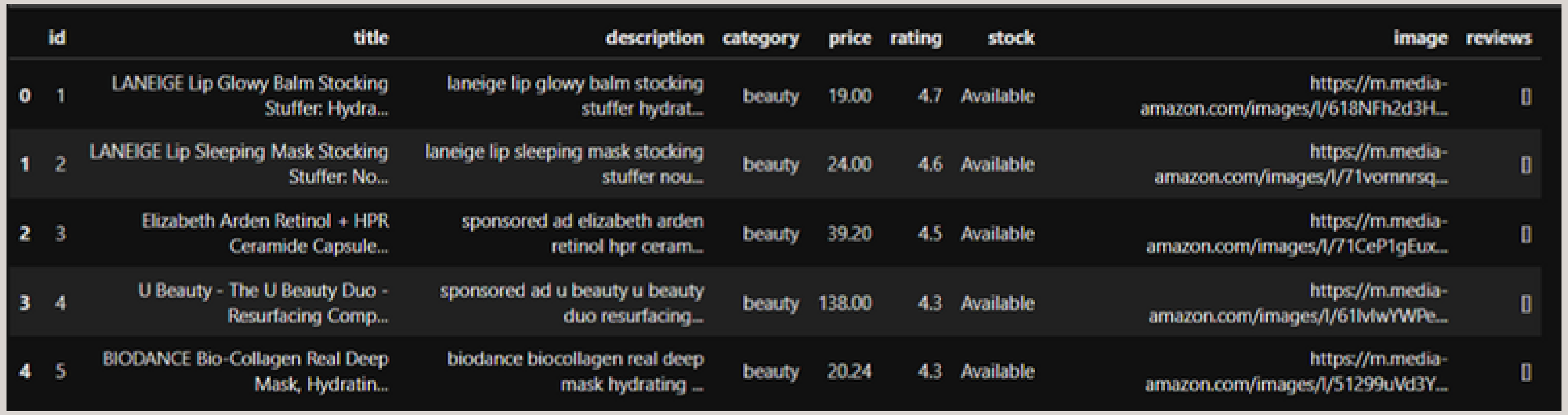
- Prices with invalid formatting, such as double dots (..), are corrected by replacing them with a single dot (.) to ensure the correct numerical representation.

Rating Processing

- Ratings are processed to extract the numerical value. Ratings with the phrase "out of 5 stars" are split to keep only the numerical part (e.g., "4.5" from "4.5 out of 5 stars").

Text Preprocessing

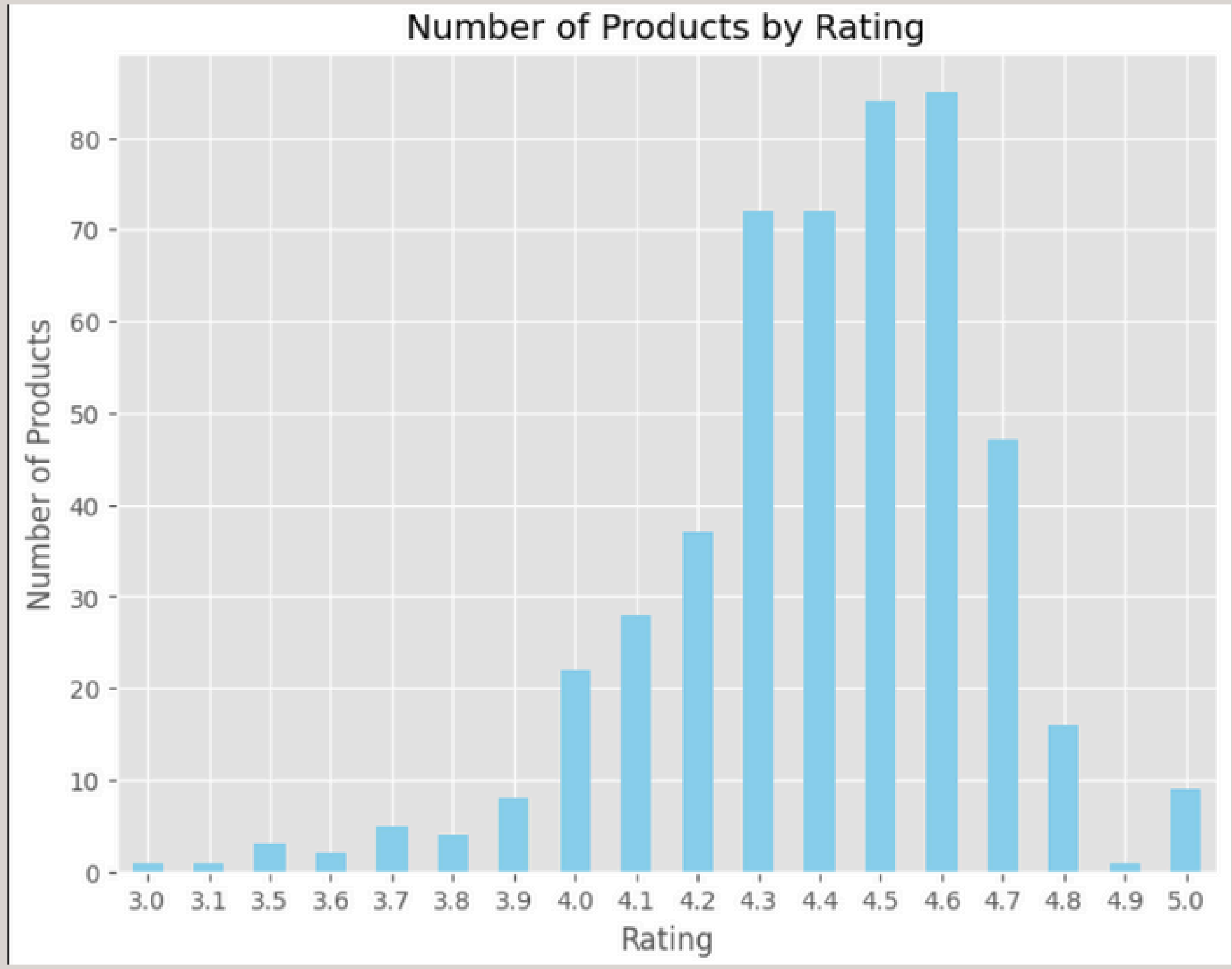
- Lowercasing: All text is converted to lowercase .
- Removing Punctuation : to focus on the textual content.
- Stop Words Removal: Common stop words (e.g., "the", "is", etc.)
- Tokenization: The text is split into individual tokens (words) to facilitate further analysis.



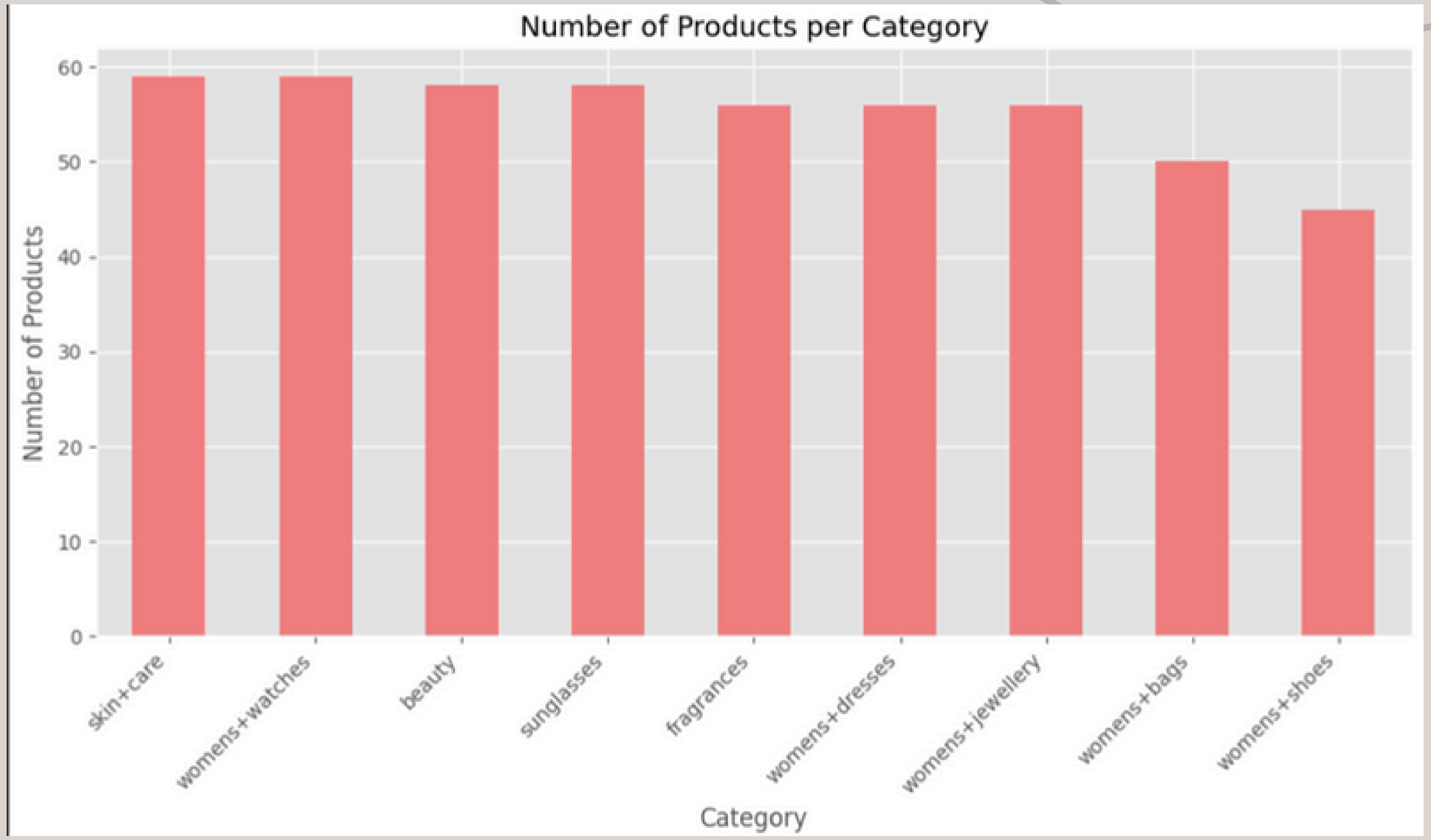
	id	title	description	category	price	rating	stock	image	reviews
0	1	LANEIGE Lip Glowy Balm Stocking Stuffer: Hydra...	laneige lip glowy balm stocking stuffer hydrat...	beauty	19.00	4.7	Available	https://m.media-amazon.com/images/I/618NFh2d3H...	[]
1	2	LANEIGE Lip Sleeping Mask Stocking Stuffer: No...	laneige lip sleeping mask stocking stuffer nou...	beauty	24.00	4.6	Available	https://m.media-amazon.com/images/I/71vornnrsq...	[]
2	3	Elizabeth Arden Retinol + HPR Ceramide Capsule...	sponsored ad elizabeth arden retinol hpr ceram...	beauty	39.20	4.5	Available	https://m.media-amazon.com/images/I/71CeP1gEux...	[]
3	4	U Beauty - The U Beauty Duo - Resurfacing Comp...	sponsored ad u beauty u beauty duo resurfacing...	beauty	138.00	4.3	Available	https://m.media-amazon.com/images/I/61lvlwYWPe...	[]
4	5	BIODANCE Bio-Collagen Real Deep Mask, Hydratin...	biodance biocollagen real deep mask hydrating ...	beauty	20.24	4.3	Available	https://m.media-amazon.com/images/I/51299uVd3Y...	[]

Data Visualization

1. Number of Products by Rating :



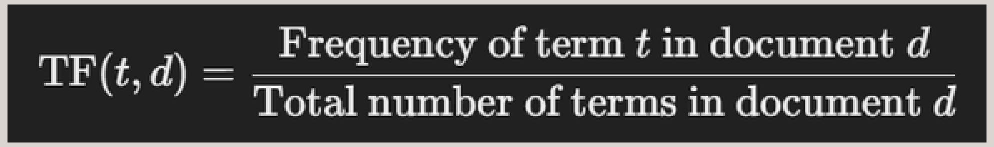
2. Number of Products per Category:



Clustering & Product Categorization

Term Frequency (TF)

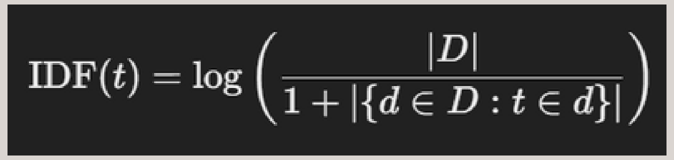
- Measures how often a term ttt appears in the combined "category + description" field of a product.



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

Inverse Document Frequency (IDF)

- Measures how unique a term ttt is across the entire product catalog (set of documents D).

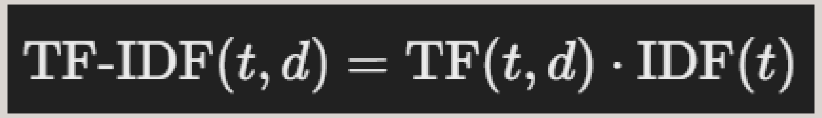


$$\mathrm{IDF}(t) = \log\left(\frac{|D|}{1 + |\{d \in D : t \in d\}|}\right)$$

- ∣D∣: Total number of documents (products) in the catalog.
- ∣{d∈D:t∈d}∣: Number of documents in which the term t appears.

TF-IDF Weight

- Assigns higher weights to terms that are frequent in the specific "category + description" combination but rare across the entire product catalog.



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

TF-IDF Configuration

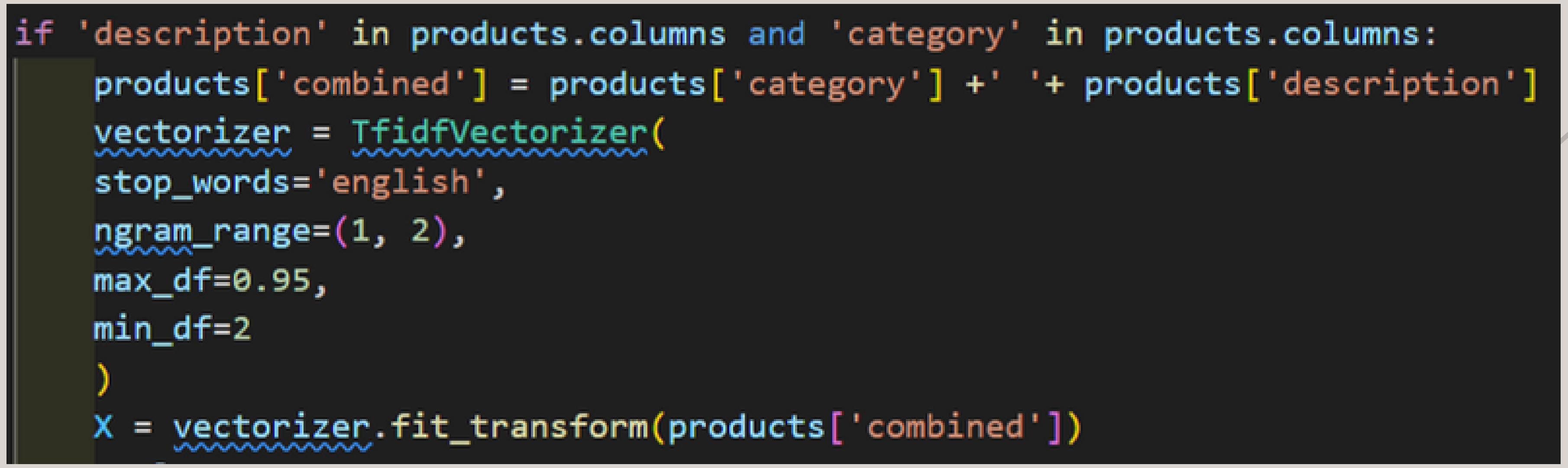
N-gram Range

The parameter ngram_range=(1, 2) is applied to capture both unigrams (single words) and bigrams (pairs of consecutive words). For example:

- Unigrams: "wireless," "headphones."
- Bigrams: "wireless headphones."

Maximum Document Frequency

(max_df=0.95): Filters out terms appearing in more than 95% of the product catalog

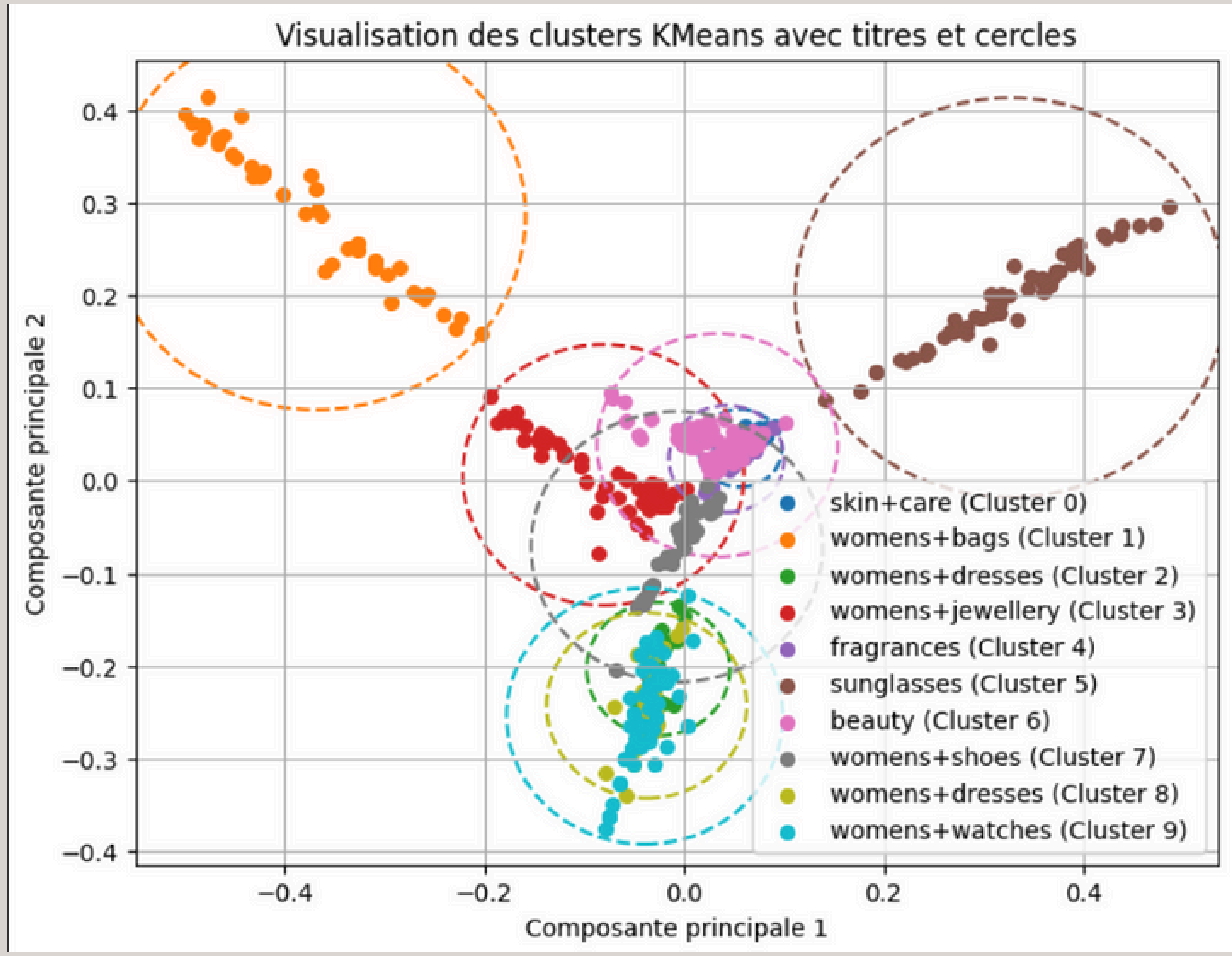


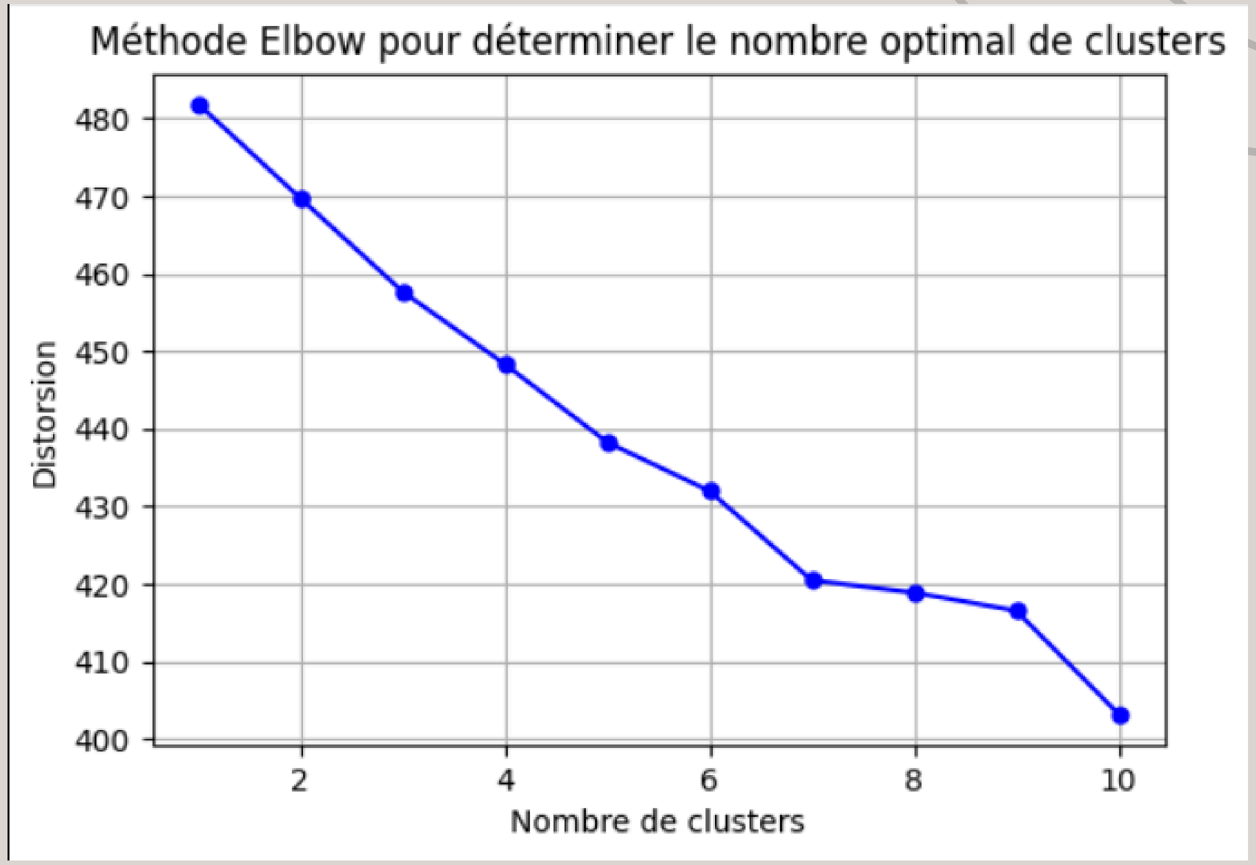
```
if 'description' in products.columns and 'category' in products.columns:
    products['combined'] = products['category'] +' '+ products['description']
    vectorizer = TfidfVectorizer(
    stop_words='english',
    ngram_range=(1, 2),
    max_df=0.95,
    min_df=2
    )
    X = vectorizer.fit_transform(products['combined'])
```

Minimum Document Frequency

(min_df=2): Filters out terms appearing in fewer than two product descriptions, removing noise from

K-Means Clustering Algorithm

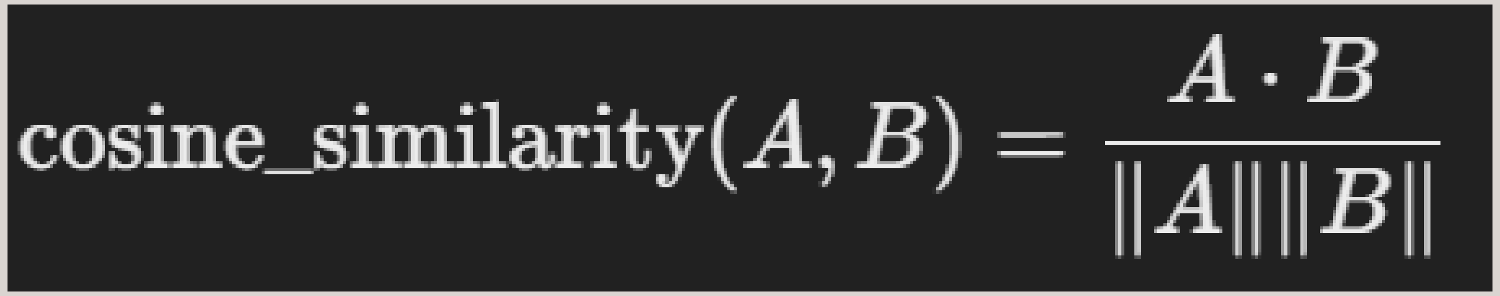




To find the best number of clusters we used the Elbow Method .
The Elbow Method involves plotting the sum of the squared distances between each data point and the center of the cluster it is assigned to. The lower the inertia, the better the points are grouped around their cluster centers.)

Assigning Products to the Nearest Cluster

Once the K-Means algorithm has clustered the products, we calculate the cosine similarity between the search query (e.g., "travel") and the centroids of each cluster. This step involves comparing the search term's vector representation with the cluster centroids (the average representation of products in each cluster).



$$\mathrm{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\| \|B\|}$$

Limitations and Future Work

Improving User Experience !

Analyzing User Transaction History !

Purchase-Based Recommendations !