FashionAI

Table of Contents

Objective **2**

Dataset **2**

Playground **2**

Generative Search in Action **3**

System Design **3**

Layer 1: Build the vector store 4

Layer 2: Search & Rerank Layer 5

- Important Functions 5
- SemanticSearchWithReranking() 6

Layer 3: Generative Search 8

- Important Functions 8
- GenerativeSearch() 10

Example experiments with different Queries **11**

Query 1: *"Maroon blazer for casual occasion from Van Heusen"* 11

Sematic Search Result 11

Generative Search result 12

Query 2: *"White anarkali kurta with a rating of more than 4"* 13

Sematic Search Result 13

Generative Search result 14

Query 3: *"Denim jacket with atleast 4 pockets, a rating of 4 and price less than 3000"* 15

Sematic Search Result 15

Generative Search result 16

Conclusion **17**

Lessons learned 17

Challenges Faced 17

Objective

The goal is to develop an advanced search system that can intelligently sift through a vast collection of fashion product descriptions. This system should be able to understand a user's query and recommend the most suitable fashion items based on their preferences.

Dataset

The dataset used for this project is Myntra Fashion Product Dataset (kaggle.com).
It includes thousands of fashion products, each with detailed information such as description, price, rating, brand name, and other attributes.

The dataset is organized as follows:

1) **Fashion Dataset v2.csv:** This CSV file contains each product in a row, with various details spread across columns.
2) **Image Folder:** This folder contains sample images for each product. Each image file is named according to the corresponding product's 'p_id' value from the CSV.

Playground

To experiment with the search tool, follow these steps:

1. Open the Jupyter Notebook: Ensure you have the Jupyter Notebook for this project open and run all the cells for the first time after launch.

You can skip the cell with the comment "Add documents to the collection with a tqdm progress bar." in section 1.2 under heading "Build Chroma Store" in case you already have the chroma db downloaded"

2. Navigate to the Playground Section: Scroll to the end of the notebook until you find the section titled "Playground" or jump from the Table of Contents.

3. Interact with the Search Tool: In this section, you can test and experiment with the search tool. You can input various user queries to see how the system retrieves and recommends fashion products based on the descriptions, prices, ratings, brand names, and other attributes from the dataset.

In this section, two cells are provided to experiment with Semantic Search and Generative Search.

Playground

Try Yourself with some queries !

Run the below cell to experiment with Semantic Search and Reranking

```
In [ ]: SemanticSearchWithReranking()
```

Run the below cell to experiment with Generative Search

```
In [ ]: GenerativeSearch()
```

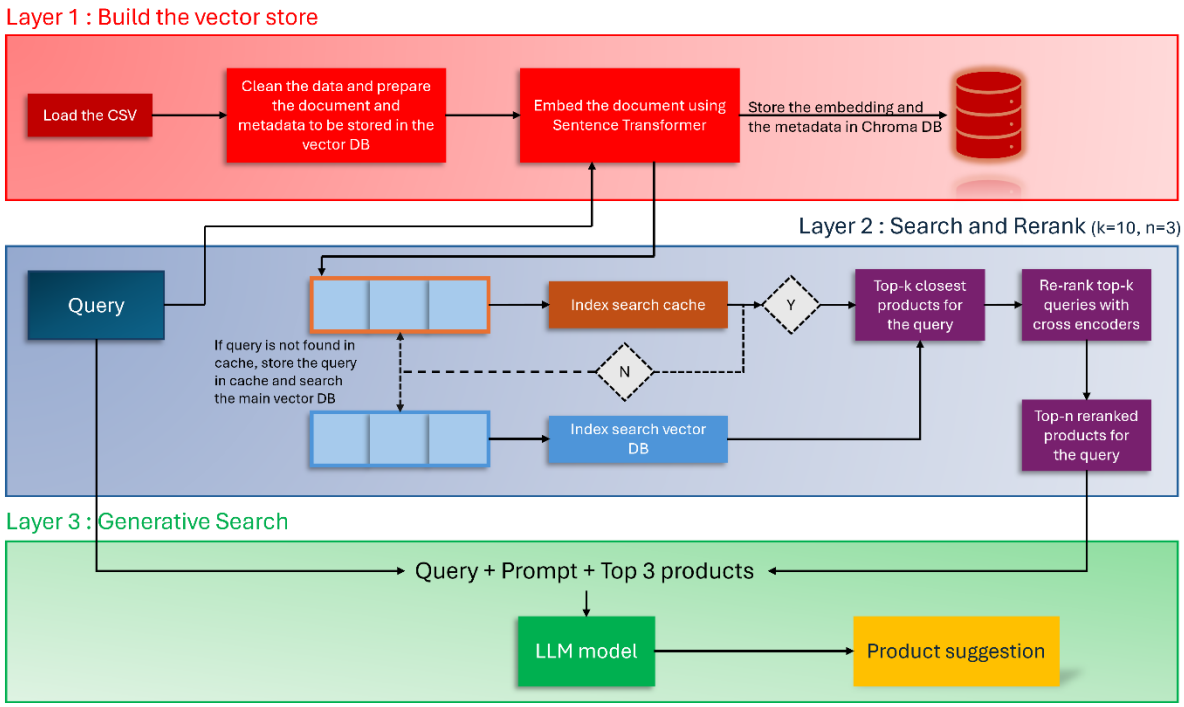
4. Analyse Results: Review the recommended products to understand how well the search tool matches the user query. This will help in assessing the accuracy and effectiveness of the search algorithm.

> *Make sure to set the OpenAI Api key in 'OpenAI_API_Key.txt' file <*

Feel free to experiment with different queries and explore different aspects of the dataset to fully understand the capabilities of the search system.

Generative Search in Action

System Design



Layer 1: Build the vector store

1) **Loading and Exploring the CSV File**: The CSV file containing fashion product data is loaded and examined to understand its structure and contents.

2) **Exclusion of 'img' Column**: The 'img' column, which contains URIs for product images, is deemed unnecessary since the images are already available in the 'dataset/images' folder, named according to the 'p_id' column.

3) **Extracting Description Data**: The 'description' column contains data in HTML format. This HTML data needs to be converted into plain text strings using Python.

4) **Portfolio:** The product information is currently fragmented across various columns, hindering comprehensive keyword searches. - Since our query focuses solely on documents, this fragmentation poses limitations.
   There are two ways to address this:
   Option 1: Centralize all the product details as metadata. Develop a chatbot to extract metadata values, facilitating precise filtering. Utilize the filtered results to construct a RAG model.
   Option 2: Compile a cohesive document called as 'portfolio' incorporating all product details. Load this document entirely into the document section of the vector DB. This ensures vector DB is aware of additional keywords used in user queries.
   - For the present task, I opted for option 2 for enhanced efficiency.

5) **Analysing Portfolio Lengths**: The maximum sentence length in the descriptions is found to be **2764** characters, and the maximum number of portfolios is less *than 1833* characters.
   Since the maximum input lengths in our dataset are well within this limit, the model is suitable and there is *no need to split* (chunk) the product descriptions.

6) **Creating Metadata**: Metadata is created, containing only the 'p_id' value to facilitate displaying the corresponding product image.

7) **Selecting a Suitable Embedding Model**: After researching, the model " nomic-ai/nomic-embed-text-v1.5 · Hugging Face" is chosen due to its good MTEB results of **62.28** and its support for text lengths up to **8192** characters.

8) **Storing Data in ChromaDB**: After completing the above steps, a collection named *'ProductDetails'* is created in ChromaDB. All product information along with the corresponding metadata is added to this collection.

This structured approach ensures the data is properly prepared, relevant columns are identified, and an appropriate model is selected for embedding the text data. Finally, the data is efficiently stored in ChromaDB for subsequent retrieval and use in the search system.

Layer 2: Search & Rerank Layer

Layer 1 needs to be executed only for the first time and when we have a new data. From this layer, we retrieve the top 10 results for the query from the vector db and are reranked using cross encoder.

Setup a cache collection named *'ProductsCache'* in Chroma DB using the client.

- *Important Functions*
- ***query_from_main_or_cache_collection()***

The function *query_from_main_or_cache_collection* is designed to efficiently search for query results by checking a cache before querying the main collection. Here is a step-by-step breakdown:

1. Initialization:

- Initialize empty lists (*ids, documents, distances, metadatas*) and an empty DataFrame (*results_df*) to store query results.

2. Cache Search:

- Attempt to retrieve the query result from the *cache_collection*.
- If the cache is empty or the retrieved result's distance exceeds a predefined *threshold(0.2)* , proceed to query the main collection.

3. Main Collection Query:

- Query the '*ProductDetails'* collection for the **top 10** results.
- Store the query in the cache collection as a document for future reference, along with its corresponding results (*texts, ids, distances, and metadatas*).
- Construct a DataFrame (*results_df*) from the main collection query results.

4. Cache Result Handling:

- If the cache contains a valid result within the distance threshold, extract the result data.
- Append the extracted data (*ids, documents, distances, metadatas*) to the respective lists.
- Create a DataFrame (*results_df*) from the cached query results.

5. Return Results: Return the constructed DataFrame (*results_df*) containing the query results.

This approach optimizes search efficiency by utilizing cached results when available and only querying the main collection when necessary.

- ***Improving Search Results with Cross-Encoder Reranking()***

Re-ranking the results obtained from a semantic search can often significantly enhance the relevance of the retrieved results. This process involves using a cross-encoder, which evaluates the relevance of each response in relation to the query. In this task, cross encoder model used is ***'ms-marco-MiniLM-L-6-v2'***.
Here's how this process works:

1. Initial Retrieval:

- Perform a semantic search to retrieve an initial set of results based on the query. This step quickly identifies potentially relevant documents.

2. Cross-Encoder Scoring:

- For re-ranking, pair the query with each of the retrieved responses.
- Pass each query-response pair into a cross-encoder model, which jointly processes the pair to capture the detailed interactions and context between them.
- The cross-encoder scores the relevance of each response with respect to the query, providing a refined relevance score.

3. Re-ranking:

- Use the relevance scores from the cross-encoder to reorder the initial set of retrieved results.
- Responses with higher relevance scores are moved to the top of the list, ensuring the most relevant documents are prioritized.

By leveraging the cross-encoder's ability to understand nuanced relationships between the query and each response, this re-ranking process leads to more accurate and contextually relevant search results.

- *SemanticSearchWithReranking()*

This is the final function of this layer which aims to perform a semantic search with an additional step of re-ranking the results using a cross-encoder model.

Here's a detailed explanation of how it works:

1. Query Input:

- If no query is provided (query=None), the function prompts the user to enter a query.

2. Initial Search:

- The function calls *query_from_main_or_cache_collection(query)* to retrieve an initial set of **top 10** semantic search results. This function queries either a main collection or a cache to get the results, which are returned in a DataFrame (*query_results_df*).

3. Preparing Inputs for Re-Ranking:

- The function prepares inputs for the cross-encoder model by pairing the query with each of the retrieved responses. This is done by creating a list of lists (*cross_inputs*), where each inner list contains the query and a response document.

4. Re-Ranking with Cross-Encoder:

- The cross-encoder model (*cross_encoder*) is used to predict relevance scores for each query-response pair. These scores reflect how relevant each response is to the query.
- The predicted scores are stored in the DataFrame (*query_results_df*) under a new column called '*Reranked_scores'*.

5. Selecting Top Results:

- The DataFrame is sorted by the re-ranked scores in descending order to prioritize the most relevant responses.
- The function selects the top n entries (top_n) based on these scores, where top_n defaults to 3.

6. Return Top Results:

- The top 3 re-ranked results are returned as the final output of the function.

Summary

This function enhances the relevance of search results by:
- Performing an initial semantic search to get a broad set of results.
- Using a cross-encoder model to re-evaluate and score these results based on their relevance to the query.
- Returning the top 3 results based on the refined relevance scores.
This two-step process ensures that the results are both comprehensive and highly relevant to the user's query.

Layer 3: Generative Search

Once we have obtained the final top semantic search results with re-ranking, we can enhance the user experience by leveraging GPT-3.5 to generate a direct answer to the query.

- *Important Functions*
- ***generate_response()***

This function utilizes OpenAI's GPT-3.5's ChatCompletion model to generate a response to a user query about fashion products. It considers the user's role, the query, and the search results from a DataFrame, and follows specific guidelines to ensure the generated response is relevant, informative, and compliant with the provided instructions.

Input Parameters:

query: The user's query for which a response needs to be generated.
results_df: A DataFrame containing search results from a corpus of fashion products. This DataFrame serves as the basis for generating the response.

Output:

Once the response is generated by the ChatCompletion model, the function extracts the content of the response and splits it into lines using the newline character ('\n'). The final response is returned as a list of lines, making it easily readable and accessible for further processing or display.

- ***display_image()***

This function is designed to display an image corresponding to a product result from the provided DataFrame (*result_df*). Let's break down its functionality:

1. Input Parameter:

- *result_df*: A DataFrame containing information about the product result, including metadata *'p_id'*

2. Metadata Extraction:

- The function extracts the metadata associated with the product result from the DataFrame. This metadata typically includes *'p_id'*, which is needed to locate the corresponding image file.

3. Metadata Conversion:

- The extracted metadata, represented as a string, is converted to a dictionary format using the *ast.literal_eval* function. This allows easier access to specific metadata values.

4. Image Number Extraction:

- From the metadata dictionary, the function retrieves the image number (*'p_id'*) associated with the product result.

5. Image Path Construction:

- Using the image number, the function constructs the file path to the corresponding image file. The path is typically based on a predefined directory structure where images are stored.

6. Image Display:

- The function utilizes the display function to showcase the image in the Jupyter Notebook environment. The image is loaded from the constructed file path and displayed with a specified width of "250" pixels.

Additional Notes:

- The images are stored in a directory named "*dataset/images*" and follow a specific naming convention, where the image file name corresponds to the '*p_id*' of the product followed by the ".jpg" extension.

- It's important to ensure that the DataFrame contains accurate metadata, especially the 'p_id', to successfully locate and display the corresponding image.

Thus, the function enables users to visualize the products associated with the search results, enhancing the understanding and interaction with the data.

➢ ***print_response_and_display_image()***

The function serves as a convenient utility for presenting the generated response and displaying the associated image of a suggested product. By combining textual information with visual representation, this function enhances the user experience and facilitates better understanding and interaction with the suggested products.

Input:

- *result_df*: This parameter represents the DataFrame containing information about the suggested product, including metadata such as 'p_id'
- *response*: This parameter contains the generated response that needs to be printed.
- *result_text*: This parameter specifies the text to be used as a header before printing the response. It defaults to "Response:" if not provided.

➢ *GenerativeSearch()*

The final function of this layer performs a generative search by combining semantic search with reranking and generating a response.

Let's delve into its functionality:

1. Input Parameter:

- *query*: This parameter allows the user to input a query directly. If no query is provided, the function prompts the user to input one via the console.

2. Semantic Search with Reranking:

- The function calls the *SemanticSearchWithReranking* function to perform semantic search with reranking for the given query. It retrieves the **top 3** most relevant results and stores them in a DataFrame named *semantic_search_df*. The index of the DataFrame is reset to ensure proper iteration.

3. Response Generation and Display:

- The function iterates over each result in the semantic_search_df DataFrame.
- For the first result (index 0), it generates a response using the *generate_response* function and prints the response along with displaying the corresponding image using the *print_response_and_display_image* function. This result is labelled as the "Top product".
- For subsequent results, it checks if the current result's reranked score is close enough to the previous result's reranked score. If it is within a certain threshold (0.5 in this case), it generates a response and displays the corresponding image, labelling the result as a "Similar product". If the rerank score difference is larger than the threshold, the loop is terminated to avoid suggesting further similar products.

4. Previous Reranked Score Tracking:

- The function keeps track of the reranked score of the previous result to compare it with the reranked score of the current result in the next iteration. This allows for determining the similarity between consecutive results.

Summary:

The **GenerativeSearch** function is the main function of this whole search system which orchestrates the process of performing generative search by integrating semantic search with reranking and response generation. It iterates over the top results obtained from semantic search, generates responses based on the query and retrieved information, and displays the corresponding images. This iterative process provides users with relevant product suggestions and allows for exploration of similar products based on reranked scores.

Example experiments with different Queries

Query 1: *“Maroon blazer for casual occasion from Van Heusen”*

Sematic Search Result

```
In [36]: SemanticSearchWithReranking("Maroon blazer for casual occasion from Van Heusen")
```

```
**************************************************
Query :  Maroon blazer for casual occasion from Van Heusen
**************************************************

[INFO] : Not found in cache. Found in main collection.
```

Out[36]:

	Metadatas	Documents	Distances	IDs	Reranked_scores
0	{'p_id': 16442258}	Name : Van Heusen Woman Women Maroon Solid Single-Breasted Formal Blazer\n Category : Blazer\n Price : 2599\n Color : Maroon\n Brand : Van Heusen Woman\n Rating : 4.58\n Description : Maroon solid regular-fit casual blazer, has a notched lapel, long sleeves, single-breasted with double button closure, two flap pockets, and a double-vented back hem 90% Polyester and 10% Spandex Dry clean Regular-fit The model (height 6') is wearing a size XS\n Attributes : {'Body Shape ID': '443,324,333,424', 'Body or Garment Size': 'Garment Measurements in', 'Closure': 'Button', 'Collar': 'Notched Lapel', 'Fabric': 'Polyester', 'Fabric Type': 'NA', 'Fit': 'Regular Fit', 'Front Styling': 'Single-Breasted', 'Length': 'Regular', 'Lining Fabric': 'Satin', 'Occasion': 'Formal', 'Pattern': 'Solid', 'Sleeve Length': 'Long Sleeves', 'Sustainable': 'Regular', 'Wash Care': 'Dry Clean'}	163.145508	12832	6.865547
1	{'p_id': 18152576}	Name : Van Heusen Woman Maroon Solid Single-Breasted Casual Blazer\n Category : Blazer\n Price : 3999\n Color : Maroon\n Brand : Van Heusen Woman\n Rating : nan\n Description : Maroon solid slim-fit casual blazer, has notched lapel, single-breasted with button closure, long sleeves, two welt pockets 64% Polyester, 34% Viscose and 2% Elastane Dry-clean Slim-fit The model (height 5'8") is wearing a size XS\n Attributes : {'Body Shape ID': '324,333,424', 'Body or Garment Size': 'Garment Measurements in', 'Closure': 'Button', 'Collar': 'Notched Lapel', 'Fabric': 'Polyester', 'Fabric Type': 'NA', 'Fit': 'Slim Fit', 'Front Styling': 'Single-Breasted', 'Length': 'Regular', 'Lining Fabric': 'Polyester', 'Occasion': 'Casual', 'Pattern': 'Solid', 'Sleeve Length': 'Long Sleeves', 'Sustainable': 'Regular', 'Wash Care': 'Dry Clean'}	170.755951	12716	6.635722
4	{'p_id': 16398418}	Name : Van Heusen Woman Women Navy Blue Checked Casual Blazer\n Category : Blazer\n Price : 3999\n Color : Navy Blue\n Brand : Van Heusen Woman\n Rating : 3.00\n Description : Navy Blue checked regular-fit single breasted front-open casual blazer, has long sleeves and single vented back hem 64% Polyester, 34% Viscose and 2% Spandex Dry Clean Regular-fit The model (height 5'8") is wearing S\n Attributes : {'Body Shape ID': '443,324,333,424', 'Body or Garment Size': 'Garment Measurements in', 'Closure': 'Button', 'Collar': 'Shawl Collar', 'Fabric': 'Polyester', 'Fabric Type': 'NA', 'Fit': 'Regular Fit', 'Front Styling': 'Single-Breasted', 'Length': 'Regular', 'Lining Fabric': 'Polyester', 'Occasion': 'Casual', 'Pattern': 'Checked', 'Sleeve Length': 'Long Sleeves', 'Sustainable': 'Regular', 'Wash Care': 'Dry Clean'}	195.148499	12800	4.584153

Generative Search result

```
In [85]: GenerativeSearch("Maroon blazer for casual occasion from Van Heusen")
```

```
**************************************************
Query :  Maroon blazer for casual occasion from Van Heusen
**************************************************

[INFO] : Found in cache!

**************************************************
Top product
**************************************************
Certainly! The Maroon blazer from Van Heusen is a solid choice for a casual occ
asion. It features a regular fit with a notched lapel, long sleeves, and a sing
le-breasted design with a double-button closure. The blazer also includes two f
lap pockets and a double-vented back hem for a stylish touch.

This versatile blazer is made of 90% Polyester and 10% Spandex, ensuring both c
omfort and durability. The fabric type is suitable for dry cleaning, maintainin
g the blazer's quality. The satin lining adds a luxurious feel to the garment.

Overall, this Maroon blazer is an excellent option for a casual setting, offeri
ng a perfect blend of style and comfort.

The suggested product looks like
```

```
**************************************************
Similar product
**************************************************
The Van Heusen Woman Maroon Solid Single-Breasted Casual Blazer is a stylish ch
oice for a casual occasion. This slim-fit blazer features a notched lapel, sing
le-breasted button closure, long sleeves, and two welt pockets, making it both
fashionable and functional. The blend of 64% Polyester, 34% Viscose, and 2% Ela
stane ensures a comfortable fit. It is recommended to dry clean this blazer for
maintenance. Perfect for a casual outing, this blazer exudes a chic vibe while
providing a sophisticated look.

The suggested product looks like
```

Query 2: *"White anarkali kurta with a rating of more than 4"*

Sematic Search Result

```
In [38]: SemanticSearchWithReranking("White anarkali kurta with a rating of more than 4")
```

```
**************************************************
Query :  White anarkali kurta with a rating of more than 4
**************************************************

[INFO] : Not found in cache. Found in main collection.
```

Out[38]:

	Metadatas	Documents	Distances	IDs	Reranked_scores
0	{'p_id': 7156696}	Name : Vishudh Women White & Navy Blue Printed Anarkali Kurta\n Category : Anarkali Kurta\n Price : 2149\n Color : White\n Brand : Vishudh\n Rating : 4.26\n Description : White and navy blue printed anarkali kurta, has a scoop neck, three-quarter sleeves, flared hemThe model (height 5'8") is wearing a size SCotton Machine-wash\n Attributes : {'Body Shape ID': '443,333,324,424', 'Body or Garment Size': 'To-Fit Denotes Body Measurements in', 'Colour Family': 'Indigo', 'Design Styling': 'Pleated', 'Fabric': 'Cotton', 'Fabric 2': 'NA', 'Fabric Purity': 'Pure', 'Hemline': 'Flared', 'Length': 'Calf Length', 'Main Trend': 'NA', 'Neck': 'Scoop Neck', 'Occasion': 'Daily', 'Ornamentation': 'NA', 'Pattern': 'Printed', 'Print or Pattern Type': 'Ethnic Motifs', 'Shape': 'Anarkali', 'Sleeve Length': 'Three-Quarter Sleeves', 'Sleeve Styling': 'Regular Sleeves', 'Slit Detail': 'NA', 'Stitch': 'Ready to Wear', 'Technique': 'Screen Print', 'Wash Care': 'Machine Wash', 'Weave Pattern': 'Regular', 'Weave Type': 'Machine Weave'}	201.807709	69	6.691045
4	{'p_id': 15616828}	Name : Myshka Women White Anarkali Kurta With Dupatta\n Category : Kurta, Dupatta\n Price : 4999\n Color : White\n Brand : Myshka\n Rating : 4.04\n Description : Colour: white Solid Mandarin collar Long, regular sleeves Anarkali shape with panelled style Ankle length with flared hem Knitted regular cotton Comes with a dupatta Cotton Machine washDupatta Length: 2 meter (approx) The model (height 5'8) is wearing a size S\n Attributes : {'Body Shape ID': '333,324,424', 'Body or Garment Size': 'Garment Measurements in', 'Care for me': 'NA', 'Colour Family': 'Earthy', 'Design Styling': 'Panelled', 'Fabric': 'Cotton', 'Fabric 2': 'NA', 'Fabric Purity': 'Pure', 'Hemline': 'Flared', 'Length': 'Ankle Length', 'Main Trend': 'NA', 'Neck': 'Mandarin Collar', 'Number of Pockets': 'NA', 'Occasion': 'Daily', 'Ornamentation': 'NA', 'Pattern': 'Solid', 'Print or Pattern Type': 'Solid', 'Shape': 'Anarkali', 'Sleeve Length': 'Long Sleeves', 'Sleeve Styling': 'Regular Sleeves', 'Slit Detail': 'NA', 'Stitch': 'Ready to Wear', 'Sustainable': 'Regular', 'Technique': 'NA', 'Wash Care': 'Machine Wash', 'Weave Pattern': 'Regular', 'Weave Type': 'Knitted'}	232.027161	388	6.567142
6	{'p_id': 8033765}	Name : Varanga Women Off-White & Golden Printed Anarkali Kurta\n Category : Kurta, Anarkali Kurta\n Price : 3299\n Color : Off White\n Brand : Varanga\n Rating : 4.14\n Description : Off-White and golden printed anarkali kurta, has a mandarin collar with half button placket, three-quarter sleeves, flared hemThe model (height 5'8") is wearing a size SViscose Rayon Hand-wash\n Attributes : {'Body Shape ID': '333,424', 'Body or Garment Size': 'To-Fit Denotes Body Measurements in', 'Colour Family': 'Monochrome', 'Design Styling': 'Empire', 'Fabric': 'Viscose Rayon', 'Fabric 2': 'NA', 'Fabric Purity': 'Synthetic', 'Hemline': 'Flared', 'Length': 'Ankle Length', 'Main Trend': 'NA', 'Neck': 'Mandarin Collar', 'Number of Pockets': 'NA', 'Occasion': 'Festive', 'Ornamentation': 'NA', 'Pattern': 'Printed', 'Print or Pattern Type': 'Ethnic Motifs', 'Shape': 'Anarkali', 'Sleeve Length': 'Three-Quarter Sleeves', 'Sleeve Styling': 'Regular Sleeves', 'Slit Detail': 'NA', 'Stitch': 'Ready to Wear', 'Sustainable': 'Regular', 'Technique': 'Foil Print', 'Wash Care': 'Hand Wash', 'Weave Pattern': 'Regular', 'Weave Type': 'Machine Weave'}	232.543777	301	6.252642

Generative Search result

```
In [86]: GenerativeSearch("White anarkali kurta with a rating of more than 4")
```

```
**************************************************
Query :  White anarkali kurta with a rating of more than 4
**************************************************

[INFO] : Found in cache!

**************************************************
Top product
**************************************************
The white anarkali kurta from Vishudh featuring a navy blue print is a stylish and elegant choice for
your wardrobe. With a rating of 4.26, this kurta is not only visually appealing but also well-receive
d by customers. Made from pure cotton and designed with a scoop neck and three-quarter sleeves, this
kurta offers comfort and style. Its flared hem adds a touch of grace to the overall look. Perfect for
daily wear, this calf-length anarkali kurta is easy to care for with a simple machine wash.

The suggested product looks like
```

```
**************************************************
Similar product
**************************************************
The white Anarkali kurta by Myshka has a rating of 4.04, making it a highly rated choice. The kurta f
eatures a solid design with a mandarin collar, long sleeves, and an ankle-length flared hem. Made fro
m regular knitted cotton, it comes with a matching dupatta for a complete look. Perfect for daily wea
r, this Anarkali kurta is both stylish and comfortable.

The suggested product looks like
```

```
**************************************************
Similar product
**************************************************
The Varanga Women Off-White & Golden Printed Anarkali Kurta is a beautiful white anarkali kurta with
a rating of 4.14, slightly above 4. It features a mandarin collar, three-quarter sleeves, and a flare
d hem, making it a stylish and elegant choice. The kurta is made of Viscose Rayon and is ankle-lengt
h, suitable for festive occasions. The foil print technique adds a touch of sophistication to the eth
nic motifs design. To maintain its quality, it is recommended to hand wash the kurta.

The suggested product looks like
```

Query 3: *“Denim jacket with atleast 4 pockets, a rating of 4 and price less than 3000”*

Sematic Search Result

```
In [40]: SemanticSearchWithReranking("Denim jacket with atleast 4 pockets, a rating of 4 and price less than 3000")
```

```
**********************************************
Query :  Denim jacket with atleast 4 pockets, a rating of 4 and price less than 3000
**********************************************

[INFO] : Not found in cache. Found in main collection.
```

Out[40]:

	Metadatas	Documents	Distances	IDs	Reranked_scores
4	{'p_id': 10562374}	Name : The Roadster Lifestyle Co Women Blue Solid Denim Jacket\n Category : Denim Jacket\n Price : 2899\n Color : Blue\n Brand : Roadster\n Rating : 4.29\n Description : Blue solid denim jacket, has a spread collar, four pockets, button closure, long sleeves, straight hemThe model (height 5'8") is wearing a size SCotton Machine-wash\n Attributes : {'Add-Ons': 'NA', 'Body Shape ID': '443,333,424', 'Body or Garment Size': 'To-Fit Denotes Body Measurements in', 'Closure': 'Button', 'Collar': 'Spread Collar', 'Fabric': 'Cotton', 'Features': 'NA', 'Hemline': 'Straight', 'Length': 'Regular', 'Lining Fabric': 'Unlined', 'Number of Pockets': '4', 'Occasion': 'Casual', 'Pattern': 'Solid', 'Print or Pattern Type': 'Solid', 'Sleeve Length': 'Long Sleeves', 'Sport': 'NA', 'Surface Styling': 'Embroidered', 'Technology': 'NA', 'Type': 'Denim Jacket', 'Wash Care': 'Machine Wash'}	218.098251	12457	3.741729
0	{'p_id': 10573162}	Name : Tokyo Talkies Women White Solid Denim Jacket\n Category : Denim Jacket\n Price : 2249\n Color : White\n Brand : Tokyo Talkies\n Rating : 4.31\n Description : White solid Denim jacket, has a spread collar, 4 pockets, button closure, three-quarter sleeves, straight hem, and unlinedThe model (height 5'8") is wearing a size SCotton Machine-wash\n Attributes : {'Add-Ons': 'NA', 'Body Shape ID': '443,333,424', 'Body or Garment Size': 'To-Fit Denotes Body Measurements in', 'Closure': 'Button', 'Collar': 'Spread Collar', 'Fabric': 'Cotton', 'Features': 'NA', 'Hemline': 'Straight', 'Length': 'Regular', 'Lining Fabric': 'Unlined', 'Number of Pockets': '4', 'Occasion': 'Casual', 'Pattern': 'Solid', 'Print or Pattern Type': 'Solid', 'Sleeve Length': 'Three-Quarter Sleeves', 'Sport': 'NA', 'Surface Styling': 'NA', 'Technology': 'NA', 'Type': 'Denim Jacket', 'Wash Care': 'Machine Wash'}	212.812500	11752	3.361337
8	{'p_id': 10742380}	Name : Tokyo Talkies Women Blue Solid Denim Jacket\n Category : Denim Jacket\n Price : 2149\n Color : Blue\n Brand : Tokyo Talkies\n Rating : 4.33\n Description : Blue solid jacket, has a spread collar, 4 pockets, button closure, long sleeves, straight hem, and unlinedCotton Machine-washThe model (height 5'8") is wearing a size S\n Attributes : {'Add-Ons': 'NA', 'Body Shape ID': '443,333,424', 'Body or Garment Size': 'To-Fit Denotes Body Measurements in', 'Closure': 'Button', 'Collar': 'Spread Collar', 'Fabric': 'Cotton', 'Features': 'NA', 'Hemline': 'Straight', 'Length': 'Regular', 'Lining Fabric': 'Unlined', 'Number of Pockets': '4', 'Occasion': 'Casual', 'Pattern': 'Solid', 'Print or Pattern Type': 'Solid', 'Sleeve Length': 'Long Sleeves', 'Sport': 'NA', 'Surface Styling': 'NA', 'Technology': 'NA', 'Type': 'Denim Jacket', 'Wash Care': 'Machine Wash'}	219.908905	11743	3.276806

Generative Search result

```
In [87]: GenerativeSearch("Denim jacket with atleast 4 pockets, a rating of 4 and price less than 3000")
```

```
**************************************************
Query :  Denim jacket with atleast 4 pockets, a rating of 4 and price less than 3000
**************************************************

[INFO] : Found in cache!

**************************************************
Top product
**************************************************
The denim jacket by The Roadster Lifestyle Co Women in blue color fits your criteria. It has a rating
of 4.29, four pockets, and is priced at 2899. This jacket is made of cotton, has a spread collar, but
ton closure, long sleeves, and a straight hem. It is a solid denim jacket suitable for casual occasio
ns and can be machine-washed.

The suggested product looks like
```

```
**************************************************
Similar product
**************************************************
The Tokyo Talkies Women White Solid Denim Jacket meets your criteria with 4 pockets, a rating of 4.3
1, and a price of 2249. It has a spread collar, button closure, three-quarter sleeves, and is machine
washable in cotton fabric. A casual piece perfect for a stylish everyday look.

The suggested product looks like
```

```
**************************************************
Similar product
**************************************************
The Tokyo Talkies Women Blue Solid Denim Jacket meets your criteria for a denim jacket with at least
4 pockets, a rating of 4, and a price less than 3000. This jacket has 4 pockets, a rating of 4.33, an
d is priced at 2149. It features a spread collar, button closure, long sleeves, and a straight hem. T
he fabric is cotton, and it is unlined.


The suggested product looks like
```

Conclusion

Finally, with the function GenerativeSearch(), user can input a query to fetch a product from the vast database using natural language.

Lessons learned

- Explored various models to implement embedding.
- Explored various ways to prepare the document for query.
- Explored different prompts to generate appropriate response for the user query.
- Implemented cache technique for faster response.
- Hands-on experience with Chroma DB.
- Explored various datasets to experiment with RAG apart from this fashion dataset.

Challenges Faced

- In this scenario, I did not come with the use of chunking a single product data. This is an area to explore for the same problem statement.
- When using OpenAI embedding function, faced an error with '$' character input. But I did not choose the model anyways since I could find something else with similar performance.
- There is still a room to explore the search query. For an example, if the user queries for "red t-shirt" it always gives the same set of results.