Project Report Semantic Spotter Fashion Recommendor

Your Name

Aparana Bhatt

Table of Contents

- 1. Ideation
- 2. Data details
- 3. Implementation Details
- 4. System Design
- 5. Challenges and Future Scope
- 6. Conclusion

1. Ideation

While we explore online women's fashion clothes we get confused as to which option to choose from and find top preferred style. If we have our trained custom recommendation assistant life would be easier and our time will also be saved.

If we know the correct keyword, we can search for good products with less effort but if we have a requirement and our search query is very long then we do not get required suggestions and end up wasting a lot of time searching for the right product.

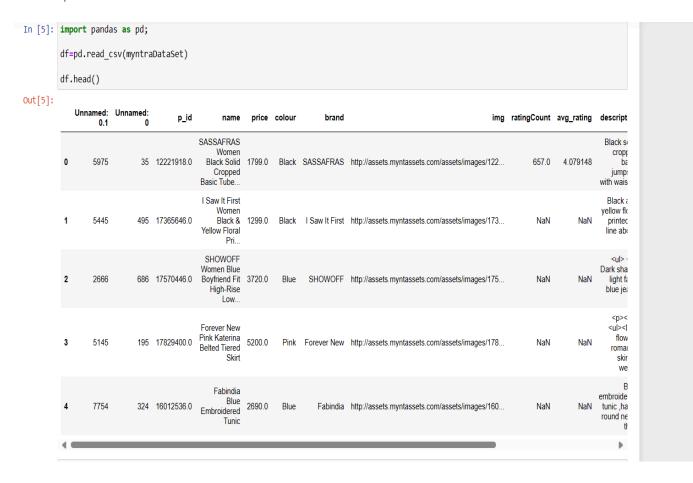
This is where semantic spotter fashion recommender can be of extreme use because we can write our needs to it and it will understand and return good suggestions in a short time and we would need less time to search for the product online.

It will greatly streamline our choices and help us select our favorite style on the go.

2. All About Data

I have used a publicly available Myntra dataset and sampled it according to the project requirement. The data set is available at Myntra Fashion Product Dataset (kaggle.com)

Data consists of a list of women products with description,name,brand and other metadata . Some of which have been used to create a minimal instruction/decision set for the helpmate to take help with.



The data is processed and sampled to 500 entries only for simplicity. We have also filtered the rows having empty description columns as description is the major source of semantic context in the process. We have also added MetaData column, text length column int the dataset during processing and saved this sampled data in a final csv file, which is being used in the next steps.

	Unnamed:	p_id	name	price	colour	brand	ima	ratingCount	avg rating	description	n attribi
0	0	17048614.0	Khushal K Women Black	5099.0			http://assets.myntassets.com/assets/images/170	4522.0	4.418399	Black printed Kurta with Palazzos with dupatta	{'Add-C 'NA', 'E Shape
1	1	16524740.0	InWeave Women Orange Solid Kurta with Palazzos	5899.0	Orange	InWeave	http://assets.myntassets.com/assets/images/165	1081.0	4.119334	Orange solid Kurta with Palazzos with dupatta<	{'Add-('NA', 'I Shape '443,333
2	2	16331376.0	Anubhutee Women Navy Blue Ethnic Motifs Embroi	4899.0	Navy Blue	Anubhutee	http://assets.myntassets.com/assets/images/163	1752.0	4.161530	Navy blue embroidered Kurta with Trousers with	{'Add-('NA', 'I Shape '333,42
3	3	14709966.0	Nayo Women Red Floral Printed Kurta With Trous	3699.0	Red	Nayo	http://assets.myntassets.com/assets/images/147	4113.0	4.088986	Red printed kurta with trouser and dupatta 	{'Add- 'NA', ' Shap '333,42
4	4	11056154.0	AHIKA Women Black & Green Printed Straight Kurta	1350.0	Black	AHIKA	http://assets.myntassets.com/assets/images/110	21274.0	3.978377	Black and green printed straight kurta, has a	{'Body SI ID': ' 'Boo Garr S

3. Implementation Details

The code is written in python and various other components and apis used are mentioned below

- 1) Python3
- 2) Openai library
- 3) LangChain
- 4) Chroma DB
- 5) Gpt-3.5 Turbo used for recurring chats
- 6) Jupyter notebook

4. System Design

System design contains below major layers and steps for a successful response.

- Data Layer: This layer is the first layer and first step where relevant data is sourced and preprocessed.
- **2) Text Embedding Layer:** In this layer the data loaded via data loader is converted to embeddings by using OpenAlEmbeddings module and then stored in a vector store called chroma db.
- 3) Semantic Search Layer: We use chroma db as a semantic search store here. This is where the document embeddings are stored pooled from the data layer after preprocessing.
- 4) LangChain Chains for prompting: In this layer we use a stuff chain to take input from a user, convert into prompt template and then pass to LLM for providing a solution for user query.
- **5) Generation Layer:** This layer makes use of promptTemplate of lang chain using a simple chain to generate the results.

Code overview and semantic spotter example

Below is the overview of main methods used in the code which are empowering the Movie Assistant .

 DataLoader to load data and parse via langchain: Langchain provides various data loader to support different format. We have used a dataframe data loader to load the fashion dataset data.



2) **Embedding And Searching**: After loading data it is to be converted to embeddings and then semantic search is to be applied .

3)

```
In [27]: # Store the split document in ChromaDB
db = Chroma.from_documents(data, OpenAIEmbeddings())

Perform Similarity Search

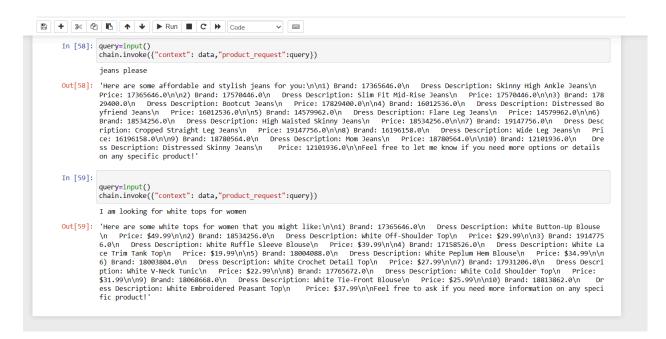
In [29]: query = "find red kurta for me "
docs = db.similarity_search(query)
print(docs[0].metadata['description'])

Trendy and versatile, this playsuit is a worthy pick to add to your casualwear closet. This playsuit comes with an alluring eth nic motifs print and V-neck.%nbsp;cul>li>Stylish black shadeli>Deautiful ethnic motifs prints/li>li>Perints/li>li>Perints/li>li>Perints/li>li>Perints/li>li>Perints/li>like knots, flared sleeves, plunge necklines, etc. in beautiful pastel palettes. It includes attractive prints and patterns, such as checks, floral, nautical stripes, and Nu Boheme, which enhance your overall appeal. The model (height 5'8) is wearing a size SPo lyester<br/>br>Machine wash<br/>br>Wash dark colour separately<br/>br>Do not bleach<br/>br>Do not bleach<br/>br>Warm iron if needed
```

4) Semantic Spotter Recommender example

Query Snapshots

```
| Ilm = ChatOpenAI(model_name="gpt-3.5-turbo")
| chain = create_stuff_documents_chain(llm, chat_prompt)
| In [56]: | query=input()
| need a few red kurta
| In [57]: | chain.invoke({"context": data, "product_request":query})
| Out[57]: | 'Here are some red kurta options for you:\n\n1) Brand: 17365646.0\n Description: Red Cotton Kurta\n Price: 17365646.0\n\n2)
| Brand: 17570446.0\n Description: Red Silk Kurta with Golden Embroidery\n Price: 17570446.0\n\n3) Brand: 17820400.0\n\n2)
| Brand: 17570466.0\n Description: Red Silk Kurta with Golden Embroidery\n Price: 17570446.0\n\n3) Brand: 17820400.0\n\n2)
| Brand: 17570466.0\n\n2) Price: 16012536.0\n\n5) Brand: 14579962.0\n Description: Red Crepe Kurta with Sequin Detailing\n Price: 14
| 579962.0\n\nFeel free to let me know if you need more options or details on any specific product!'
| In [58]: | query=input()
| chain.invoke(("context": data, "product_request":query))
| jeans please
| Out[58]: 'Here are some affordable and stylish jeans for you:\n\n1) Brand: 17365646.0\n Dress Description: Skinny High Ankle Jeans\n Price: 17365646.0\n\n2) Brand: 17570446.0\n\n2) Brand: 17570446.0\n\n3) Brand: 17809400.0\n\n4) Brand: 16012536.0\n\n5) Brand: 17809400.0\n\n4) Brand: 16012536.0\n\n5) Price: 17570446.0\n\n3) Brand: 17809400.0\n\n6)
| Brand: 15814256.0\n\n5) Price: 16012536.0\n\n5) Brand: 14579962.0\n\n6) Press Description: Skinny High Ankle Jeans\n Price: 17570446.0\n\n6)
| Brand: 15814256.0\n\n5) Price: 1797066.0\n\n5) Brand: 15106158.0\n\n5) Price: 11834256.0\n\n5) Price: 118314256.0\n\n5) Price: 118314356.0\n\n5) Price: 118314
```



Challenges and Future Scope

Challenges

- 1) Identifying minimal subset that would yield proper data for semantic search documents in the dataset .
- 2) The support code was written on the older openal version, the common modules using openal apis had to be rewritten to handle response structure and api call changes.
- 3) The score generation and query response is limited by the dataset size and user requirement. Since the sample dataset is small, the query suggestions are not much satisfactory however working well with respect to the kind of content that is present.
- 4) Frequent data updates in the chroma db database for very large datasets is taxing.

Future Scope

- 1) Store user shopping patterns and when the user logs in provide users with similar suggestions even if the user has not yet searched anything. This approach can be used to also refine user likes and dislikes.
- We can also include price in search response in future so that we can provide user with more refined searches. However since price keeps changing, we should not make it part of embeddings but of metadata.

4. Conclusion

This project showcases the utility of semantic search combined with the power langchain stuff chaining to improve and enhance user experience to more humane and also reduce online shopping time expenditure of the user .