Documentation of Approach
AI Powered Product Recommendation Engine

Project Overview

This project implements a personalized product recommendation system powered by advanced prompt engineering using OpenAI's GPT-3.5-turbo. The primary objective is to recommend products tailored precisely to user preferences and browsing history by effectively structuring prompts to guide the generative model.

Approach and Methodology

Step 1: Data Collection and User Input

Collected user data through a React-based frontend.

Captured critical preferences, including:

- Product categories
- Preferred brands
- Desired price range

Utilized a clean, intuitive UI to simplify the data-gathering process.

Step 2: Candidate Product Selection

Identified a relevant subset of candidate products from a larger product database.

Implemented scoring logic to rank products based on alignment with user-specified preferences, ensuring relevance in the candidate pool.

Step 3: Advanced Prompt Engineering

Prompt engineering was central to the success of this recommendation system, designed through meticulous crafting and structured implementation:

Structured Prompt Creation:

Initially defined the assistant's role clearly, instructing GPT-3.5 explicitly to recommend exactly five tailored products.

Provided detailed information about the user's stated preferences, including desired categories, preferred brands, and price range constraints.

Comprehensive browsing history was included to contextualize user intent, highlighting product names, IDs, categories, price points, brands, ratings, and key product features and tags.

Candidate Product Selection:

Implemented a scoring algorithm to narrow down the product catalog, selecting up to 15 products based on alignment with user preferences and browsing patterns.

Evaluated products according to criteria such as category and brand alignment, price similarity, ratings, and feature/tag relevance to browsing history.

Strategic Recommendation Mix:

Instructed GPT-3.5 explicitly to organize its recommendations into three categories:

a. Core Recommendations (2-3 items): Closely matching user preferences and browsing behavior.

b. Complementary Products (1-2 items): Products enhancing previously viewed items.

c. Discovery Product (1 item): An unexpected but relevant recommendation intended to broaden the user's interests.

Quality Guidance:

Included examples explicitly contrasting high-quality versus poor-quality recommendation explanations.

Guided GPT-3.5 to ensure explanations were highly detailed, contextually relevant, and directly tied to user-specific behaviors and preferences.

Response Format Enforcement:

Clearly instructed GPT-3.5 to output responses in a structured JSON format to enable reliable parsing and smooth frontend integration:

```
[
{
"product_id": "prod123",
"explanation": "Clear reasoning for this recommendation connected to the user's specific interests",
"score": 8
},
...
]
```

Step 4: Model Integration
Integrated OpenAI's GPT-3.5-turbo using a Flask-based backend.
API endpoints were developed to: Accept user inputs and construct prompts accordingly
Handle interaction with GPT-3.5-turbo
Parse and process the AI model responses into a usable format

Step 5: Response Handling and Display
Parsed model-generated responses and formatted them for clear presentation.
Integrated with React frontend to display recommendations interactively.
Ensured smooth user experience through responsive design and clear visualizations.

Key Technologies and Tools Used:
Frontend: React, HTML, CSS
Backend: Flask, Python
Generative AI Model: GPT-3.5-turbo via OpenAI API