AI AGENT FOR STORING AND RETRIEVING THE DATA

Problem:

The objective of the project is to analysing and creating Q&A system.

-Cleaning and preprocessing the raw data from dataset.

- standardize and Insights of dataset.

- Visualization of data.

- Develop a Q&A system that allows user to retrieve the data from LLMs and vector embedding.

Data Transformed:

- Columns are converted to lowercase, spaces are replaced with underscore '_' .
- Invalid data or 0 are placed it replaced with None.
- Generate insights.

Setup:

Ollama: It is local LLM Model, from ollama we pulled 2 models:

- llama3.2 for LLMs.
- mxbai-embed-large for embeddings.

Libraries:

1. pandas
2. matplotlib
3. tabulate
4. langchain-ollama
5. langchain-chroma
6. os
7. logging
8. datetime

Code:

data_transformation.py

This file handles the data loading, cleaning, standardization, insights and visualization.

- log: When function prints with it appends a timestamp log file in the log directory.
- Data Loading: It reads the uploaded dataset file into pandas data frame.
- Standardization: It converts columns names to lowercase and spaces are replaced with underscore.
- Remove Duplicates: Delete unnecessary data in dataset.
- Handling Negative Values: If 0 or less than 0 value present, it replaced with None(NaN).
- Insights Generation:
- Visualization: Generate a bar graph chart using matplotlib and save it as branches_chart.png.

vector.py:

This file is responsible for managing the chroma vector store, it is for efficient search of dataset.

- Embeddings: By using OllamaEmbedding with mxbai-embed-large model to convert text into numerical vector representation.
- db location: The chroma vector data store here.
- Document: It iterates through each row in dataset, create objects and handles Metadata.
- Vector Store: Initialize Chroma vector with a collection_name and persist_directory. Documents are added to vector store.
- Retriever: It will return the relevant documents for user query.

main.py:

This file is a interactive application for querying the data using the LLM and vector store.

- LLM: Initialized OllamaLLM with llama3.2 model.
- Prompt Template: ChatPromptTemplate used for structuring the user prompt for understanding of LLM.
- Chain: It combines the prompts and model for user queries.
- Interactive loop: Once enter the while loop, user need to ask question until user type q or exit.
- Retrieval Augmented Generation(RAG): User need retrieve the documents from vector store. The retrieved holds the context, these context and questions passed to chain.invoke(). Then LLM generates an answer for the user question.
- Result: Prints result in console.