Title: Code Genei AI

Name: Ranjan Kumar

.....

Milestone 2: Testing ollama integration &chatbot Functionality

Github : [click here]

1. Project Overview

This project is a Streamlit web app that lets you chat with your dataset. It uses a large language model (LLM) running on your computer through Ollama. By giving the LLM a sample of the data, you can ask questions in plain language and get useful insights.

2. Features

- **Interactive Chat Interface:** A clean and intuitive chat window for seamless conversation.
- **Local LLM Integration:** Communicates with a local LLM server (Ollama), ensuring privacy and offline functionality.
- **Dataset Context:** Automatically feeds a sample of the dataset to the LLM to provide relevant context for responses.
- **Session State Management:** Retains chat history and loaded data across user interactions.
- **Chat History:** Allows users to start new chats and revisit previous conversations.

3. System Requirements & Setup

System Requirements

- **Python: 3.13.5**
- **Streamlit: pip install streamlit**
- **Pandas: pip install pandas**
- **Requests: pip install requests**
- **Ollama:** Must be installed and running on your local machine. Download from the Ollama website.
- **Ollama Model:** A model must be pulled. The application is configured for **llama3.2:1b**, but you can change it.
  ollama pull llama3.2:1b

Setup & Execution

1. **Save the Code:** Save the provided Python script as **chat_Bot.py.**

2. **Place the Dataset:** Place the **bengaluru_house_prices.csv** file in the same directory as the script.
3. **Run the App:** Open your terminal, navigate to the project directory, and execute the following command:
   streamlit run chat_Bot.py

4. Approach & Methodology

The application's core methodology is based on contextual prompting, where a large language model is given a small, but representative, sample of the dataset along with the user's query.

1. **Initialization:** Upon starting the app, the bengaluru_house_prices.csv dataset is loaded into a pandas DataFrame and stored in Streamlit's **st.session_state**.
2. **User Input:** When a user types a query, it is captured by the **st.chat_input** component.
3. **LLM Interaction:** Your question and the sample data are joined together into one prompt, which is then sent to the local Ollama server.
4. **State Management:** All messages are saved to st.session_state.messages, ensuring that the conversation history is maintained and can be displayed correctly.

5. Limitations & Future Enhancements

Limitations

- **Limited Context Window:** The current approach uses a small sample of the dataset as context. This may be insufficient for questions that require information from the middle or end of a large dataset.
- **Dependency on Local Server:** The application requires the Ollama server to be running locally, which may pose a challenge for users who are not familiar with the framework.

Future Enhancements

- **Improved Context Strategy:** Implement a more advanced context retrieval method, such as using a vector database to search for relevant rows based on the user's query and retrieve a more accurate and comprehensive data sample.
- **User Interface Improvements:** Add features like user authentication, the ability to upload different datasets, and customizable model settings.