**AI-Powered Data Analysis and Visualization 🤖**

**MADDELA RAVI TEJA**

**Objective:**

* The primary objective of this project is to develop a web-based data analysis application that leverages advanced AI Models to assist users in performing data analysis tasks.
* Specifically, the application aims to:

1. Enable users to upload CSV files and analyze the data within.
2. Provide functionalities for statistical analysis and data visualizations.
3. Utilize natural language processing to understand and respond user queries effectively.

**Approach:**

* The approach to achieving the objectives is outlined as follows:

**Technologies used:**

* **Frontend:** Streamlit is used to create an interactive and user-friendly web interface.
* **Backend:** LangChain Agents and Google Generative Ai (Chat Model) Model are utilized for processing and analyzing the data.

**Implementation steps:**

1. **Setup and Imports:** Import necessary libraries including Streamlit, Pandas, LangChain, Google Generative AI, io, and re.
2. **API Integration:** Configure and set the API key for Google Generative AI to enable natural language process capabilities.
3. **User Interface Design:** 
   1. Create a title and file uploader widget using Streamlit.
   2. Display uploaded CSV file contents using Pandas.
   3. Provided a dropdown menu for selecting tasks (statistics or plotting).
4. **Data Handling:**
   1. Read the uploaded CSV file contents using Pandas.
   2. Convert the Data Frame to a CSV buffer for processing.
5. **Agent Creation:**
   1. Use LangChain to create an AI agent configured with the Google Generative AI Model.
6. **User Interaction:**
   1. Create a text input for user queries and a submit button to execute the query.
7. **Query Processing:**
   1. Invoke the AI Agent with the user’s query and display the response.
8. **Dynamic Code Execution:**
   1. Extract and execute Python code from the AI Agent’s response to generate plots.
   2. Handle and display any errors encountered during execution.

**Conclusions:**

The developed application successfully meets its objective by interacting various technologies to provide a seamless and interactive data analysis experience. Key conclusions from the project include:

* **Effective use of AI:** Google Generative AI effectively understands and processes user Queries, enabling sophisticated data analysis tasks.
* **Interactive and User-Friendly:** Streamlit provides an interactive interface for users to interact with the application, upload files and view results.
* **Robust Data Handling:** The combination of Pandas and io modules ensures efficient data manipulation and buffering, facilitating smooth data processing.
* **Dynamic Visualization:** The ability to execute dynamic code blocks extracted from AI responses allows for flexible and powerful data visualization capabilities.
* **Error Handling:** Comprehensive error handling mechanisms ensure the application remains robust and User-friendly, even when unexpected issues arise.

**References:**

1. **Streamlit Documentation:** Provides comprehensive guides and references for building web applications using Streamlit.

<https://docs.streamlit.io/develop/api-reference>

1. **LangChain documentation:** Reference for creating and managing AI Agents and Google Generative AI Chat Model using LangChain.

<https://api.python.langchain.com/en/latest/langchain_api_reference.html>

1. **Python exec Function:** Official Python documentation for the `**exec**` function, which dynamically executes Python code.

<https://docs.python.org/3/library/functions.html>

1. **Python io Module:** Official python io module documentation for handling input/output operations.

<https://docs.python.org/3/library/io.html>

1. **Public GitHub Repository:** To get clear understanding regarding LangChain and for the process of creating agent, Generative AI models.

<https://github.com/bansalkanav/Machine_Learning_and_Deep_Learning/tree/master/Module%209%20-%20GenAI%20(LLMs%20and%20Prompt%20Engineering)>