WEEK8-TRACK -1

Implementation

I started by setting up the environment in Google Colab and installing key libraries such as SpeechRecognition, pydub, gTTS, soundfile, and openai-whisper. The notebook's flow consisted of four stages:

1. **Audio Capture** – I recorded live voice input directly from the Colab interface using JavaScript integrated into Python. The audio was saved locally as voice.wav.
2. **Transcription** – Using the **Whisper base model**, the system accurately converted my recorded voice into text. This became the user query sent to the LLM.
3. **LLM Processing** – I connected to **OpenAI's GPT-4o-mini** through the openai API. The recognized text was passed as input to generate a coherent and contextual response.
4. **Speech Synthesis** – Finally, I applied **Google Text-to-Speech (gTTS)** to transform the LLM's reply into an audible MP3 file (response.mp3), completing the speech-to-text-to-speech loop

week8_track1_speech_llm_(1)

.

Learning & Observations

During testing, I experimented with microphone duration, model accuracy, and response latency. Whisper provided high-quality transcriptions even for noisy audio, and GPT-4o-mini generated relevant, human-like answers. I also handled errors like missing PyAudio in Colab and ensured stable performance by retrying failed LLM calls.

This integration demonstrated how speech recognition, natural-language reasoning, and speech synthesis can operate together to enable **conversational AI**. I also realized how model parameters like temperature influence response creativity versus factual precision.

Results

The system successfully completed all core tasks:

- Captured and saved voice input.
- Transcribed speech accurately into text.
- Generated contextual replies using the LLM.

- Spoke back the response in clear synthesized voice.

This achieved the Week 8 Track 1 goal of building an **end-to-end, speech-driven AI assistant** that mimics natural human interaction.

Reflection

From this experiment, I learned the importance of connecting multiple AI modalities—audio, text, and reasoning—within one pipeline. It deepened my understanding of how LLMs can interpret spoken language and respond conversationally. Overall, the project proved that integrating voice interfaces with LLMs can create more natural, accessible, and human-centered AI applications.

WEEK8/-TRACK2

Implementation

I began by loading the **medical_summary_agent_metrics.csv** dataset, which includes columns such as accuracy, latency, trust score, evidence coverage, and redundancy. The LLM interprets user prompts and converts them into structured plot specifications (like x, y, and kind), which are validated against df.columns to prevent unsafe code execution. Once validated, a single Matplotlib chart is generated using default styling to maintain reproducibility and simplicity.

This approach combines **language understanding (via LLM)** with **data visualization logic**, so the system behaves like a natural conversation between a human analyst and an AI data assistant.

From the medical_summary_agent_metrics dataset, I analyzed trends across multiple experimental runs of my summarization agent.

- **Accuracy** values consistently improved across model updates, with some exceeding **0.92**, showing refinement in summarization quality.
- **Latency** decreased as optimization layers were added, averaging around **1.2–1.5 seconds per request**.
- **Trust scores** and **evidence coverage** remained strong, proving that my multi-agent verification logic effectively reduces redundancy while improving factual reliability.
  This data-driven feedback loop helped me confirm that my model not only performs better quantitatively but also maintains stable reasoning quality.

key Learnings

- This track helped me understand how to bridge **analytical visualization** with **LLM-based natural language understanding**. It showed that enabling users to *“talk to their data”* enhances accessibility and interpretability. By integrating voice, visualization, and metrics, I created a cohesive multimodal system where agents can summarize medical data and visualize their own performance.

Reflection:

From both Track 1 and Track 2, I gained practical experience in **multimodal AI** — connecting speech, language, and visualization. This combined setup demonstrates how conversational AI can make technical results intuitive, interactive, and user-friendly. It reinforced the potential of **AI agents that can reason, summarize, visualize, and explain**, forming a strong foundation for my multi-agent project going forward.