Adaptive Recommendation Chatbot with RAG and Vector Database

Tasks:

1. Domain Selection: The domain chosen is Finance. It has information about Saving Strategies, Investment Options and Wealth Management Tips for individuals belonging to different income groups. The chat bot aims to provide guidance to people who have little to no knowledge about finance to people to have good knowledge about finance

2. Data Collection and Preprocessing: I have combined data from various sources into a single text document. I have preprocessed the data and indexed it in the vector database and converted it into the form that is suitable to be processed by LLM

3. Vector Database Implementation:

- The database chosen is qdrant cloud
- I have preprocessed the data by removing relevant information and indexed it and stored it as chunks so that it is easily preprocessed by the LLM. I have used OpenAI for this.

4. Application Development:

- I have used 3 google collab notebooks to implement this project. 'Intro_toQdrant' contains the code for data preprocessing and storing them as vectors. I have used streamlit for the frontend. 'run.py' contains code of the front end intergrated with 'Intro_toQdrant'. 'Rag_chat_bot' is used to run the application.

5. Evaluation and Testing:

- Few of the queries of different level of difficulties successfully answered by my chatbot during the demo are below:

Simple Queries

1. What is a high-yield savings account?
2. How do I create a monthly budget?
3. What is a Roth IRA?
4. What is the benefit of an emergency fund?
5. How can I avoid high-interest debt?

Nuanced Queries

1. How should I balance my contributions between a 401(k) and a Roth IRA if I expect my tax rate to be higher in retirement?

2. What strategies can I use to minimize capital gains taxes if I frequently rebalance my portfolio?
3. How should I structure my charitable contributions to maximize both my philanthropic impact and tax deductions?
4. What role can life insurance play in my overall financial strategy, particularly for estate planning and wealth transfer?
5. What role life insurance overall financial strategy, estate planning and wealth transfer?

Constraints based queries

1. I'm a student with an income below $100,000. Should I prioritize contributing to a Roth IRA or paying off my student loans?
2. How can someone with an income between $100,000 and $500,000 effectively use Health Savings Accounts (HSAs) as part of their retirement planning?
3. How can I protect my assets from potential liabilities if I have substantial investments and an annual income over $500,000?
4. I have an annual income of $150,000 and want to maximize my retirement savings. Should I focus on my 401(k) or open a SEP IRA?
5. With an income of $400,000, how can I effectively use tax-loss harvesting if most of my investments are in a taxable brokerage account?
6. With an income of $90,000 and limited investment experience, how can I start building a diversified portfolio without taking on too much risk?
7. With an annual income of $80,000 and no significant savings, how can I build an emergency fund while also starting to invest for the future?

Challenges and overcoming them

1. Domain Selection: Choosing an appropriate and specific domain was crucial to ensure the relevance and usefulness of the chatbot. After careful consideration, I selected the finance domain, focusing on saving strategies, investment options, and wealth management tips. This choice was driven by the broad interest and utility of financial guidance for different income groups and varying levels of financial knowledge.

2. Data Collection and Preprocessing: Aggregating and preprocessing data from various sources into a consistent and useful format for the chatbot. I combined data from multiple sources into a single text document and preprocessed it to ensure uniformity.

3. Vector Database Implementation: Choosing and implementing an efficient vector database to handle the high-dimensional data required for similarity search in the chatbot. I chose Qdrant Cloud as the vector database due to its capabilities in managing and querying vector data. The data was preprocessed, indexed, and stored as vectors to ensure seamless integration with the LLM. Qdrant's performance in handling large datasets and performing similarity searches made it a suitable choice.

4. Application Development: Developing and integrating various components of the chatbot application across different platforms. I used three Google Colab notebooks for different parts of the project. This modular approach allowed you to manage and debug each part separately, ensuring smooth integration and functionality.

5. Evaluation and Testing: Ensuring the chatbot could handle queries of varying difficulty and provide accurate and helpful responses. I tested the chatbot with a range of queries, including simple, nuanced, and constraint-based questions. Examples of successfully answered queries demonstrate the chatbot's capability to address different user needs:

- Simple Queries: Basic financial concepts and advice.

- Nuanced Queries: More detailed and specific financial scenarios requiring deeper understanding.

- Constraint-Based Queries: Specific financial situations with constraints.

- This thorough testing ensured the chatbot could handle real-world questions effectively.

By systematically addressing challenges related to domain selection, data collection and preprocessing, vector database implementation, application development, and evaluation/testing, I was able to create a robust finance domain chatbot. The successful handling of a wide range of queries demonstrates the chatbot's potential to provide valuable financial guidance to users with different levels of financial knowledge and varying income levels.