**Restaurants Recommendation Bot**

Table of Contents

[1. Introduction 1](#_Toc181076119)

[1.1 Project Goals 1](#_Toc181076120)

[1.2 Scope 2](#_Toc181076121)

[2. AI Model 2](#_Toc181076122)

[2.1 Model Selection 2](#_Toc181076123)

[2.2 Model Training 2](#_Toc181076124)

[2.3 Model Performance 2](#_Toc181076125)

[3. Data details 3](#_Toc181076126)

[4. Key Design Stages 3](#_Toc181076127)

[4.1 Innovation and creativity 4](#_Toc181076128)

[4.2 Collecting user preferences 4](#_Toc181076129)

[4.3 Prepare recommendation 4](#_Toc181076130)

[4.4 Final recommendation 5](#_Toc181076131)

[5. Challenges 5](#_Toc181076132)

[6. User Experience and Evaluation 6](#_Toc181076133)

[7. Conclusion 7](#_Toc181076134)

[7.1 Summary 7](#_Toc181076135)

[7.2 Future Work 7](#_Toc181076136)

# 1. Introduction

## 1.1 Project Goals

* **Objective:** Create a smart bot that suggests restaurants based on user food and price preferences.
* **Target Audience:** Individuals seeking restaurants that meet specific criteria.
* **Key Features:**
  + Personalized recommendations.
  + Simplify the restaurants selection process based on following :
    - Cuisine
    - Ambiance
    - Area
    - Rating
    - Alcohol Served
    - Smoking Permitted
    - Price Range

## 1.2 Scope

* **In-Scope:** Static chat using Jupyter Notebook data
* **Out-of-Scope:** Flask application

# 2. AI Model

## 2.1 Model Selection

* **Chosen Model:** gpt-3.5-turbo by OpenAI.

## 2.2 Model Training

* **Initial Dataset:** Pre-trained on car data collected from Kaggle. The link to the dataset is provided below.

<https://www.kaggle.com/datasets/uciml/restaurant-data-with-consumer-ratings>

* **Fine-Tuning Data:**
  + Detailed prompts outlining each step and scenario.
  + Restaurant metadata including cuisine type, pricing, and operating hours.

## 2.3 Model Performance

**Metrics:**

* Compare bot data with expected data.
* Precision and recall in recommendation accuracy.
* User satisfaction ratings.
* Response time and relevance.

# 3. Data details

|  |  |
| --- | --- |
| **Column** | **Description** |
| Name | Restaurant Name |
| Address | Restaurant Address |
| Cuisine Variety | Cuisine Variety like 'Mexican', 'Bar', 'Cafeteria', etc… |
| Alcohol Served | Either yes or no |
| Ambiance | Familiar or Quite |
| Area | Open or closed |
| Smoking Permitted | Either yes or no |
| Price Range | Low / Medium / High |
| Operating Hours | Value can be like 09:00-12:00; or 18:00-23:30; |
| Rating | Low / Medium / High |
| Operating Days | Values can be like Mon;Tue;Wed;Thu;Fri; |
| gps\_link | A google map may look like <https://www.google.com/maps/search/?api=1&quer>.. |
|  |  |

# 4. Key Design Stages

A diagram of a chatbot system

Description automatically generated

## 4.1 Innovation and creativity

* The model's response was provided as JSON. I formatted it for better readability using the following display: `display(pd.DataFrame([json\_data]))`.
* To clarify the intent, I also added code to print the intent dictionary with the messages.
* In some cases, the result did not show the top 3 restaurants. Including the intent information proved helpful.
* The function `dialogue\_mgmt\_syste**m()` was** tested for several cuisines.
* Attempts were made to improve the output.

## 4.2 Collecting user preferences

The initial step involves interacting with the user to collect their preferences and requirements. This is accomplished through a series of questions or interactions where the bot gathers data such as user interests, previous interactions, specific needs, and contextual information. The bot uses natural language processing (NLP) to understand and extract relevant details from user responses, ensuring accurate and comprehensive data collection. This information serves as the basis for personalized recommendations.

A diagram of a chat bot

Description automatically generated

## 4.3 Prepare recommendation

After collecting user preferences, the system filters initial recommendations from the database. The suggested product must meet or exceed the user's specifications within the stated budget.

A diagram of a product mapping

Description automatically generated

## 4.4 Final recommendation

After presenting the initial recommendations, the bot remains interactive, allowing the user to provide feedback or ask follow-up questions. It refines the suggestions by incorporating additional user input, clarifying preferences, and addressing any ambiguities. This iterative process ensures that the final recommendations are highly customized and satisfactory, providing a more precise and user-centric experience.

A diagram of product recommending

Description automatically generated

# 5. Challenges

**Case 1:-**

* **Issue:** When the intent was "no," the bot did not provide reasons, even if all six key values were filled. The bot should present the collected data without asking follow-up questions. It was simply saving the intent as "no."
* **Solution:**
  + Added prompts to encourage the bot to ask questions until all dictionary values are filled.
  + Started displaying the dictionary values.
  + Included more sample conversations with varying questions.
  + Used `pd.DataFrame()` to present JSON in a clean tabular format.

**Case 2:-**

* **Issue:** The chatbot sometimes replied with 2 recommendations during QA in `dialogue\_mgmt\_system()`.
* **Solution:** Updated the prompt to show 3 recommendations.

**Case 3:**-

* + **Issue**: The chatbot was unable to automatically comprehend the intent in the presence of a spelling error. For example:

{

'Cuisine\_Variety': 'Mexican',

'Area': 'open',

**'Ambiance': 'quiet',**

'Rating': 'low',

'Alcohol\_Served': 'yes',

'Smoking\_Permitted': 'yes',

'Price\_Range': 'low'

}

* + **Solution**: I asked for the wrong key name, the chatbot provided it, and I corrected it. And also displayed all correct keys at the end of the chat

|  |
| --- |
| Sure! Let's analyze the keys one by one:  1. Cuisine\_Variety: Mexican cuisine matches the user's preference, so it's correct.  2. Area: The user prefers an open area, which is correctly filled in the dictionary.  3. Ambiance: The user prefers a quiet ambiance, so this is also correct.  4. Rating: The user did not specify any preference for the rating, so this key is filled based on an assumption and needs further clarification.  5. Alcohol\_Served: The user mentioned needing alcohol, so this key is incorrectly filled as 'yes'.  6. Smoking\_Permitted: The user mentioned wanting to smoke, so this key is incorrectly filled as 'yes'.  7. Price\_Range: The user specified a low budget, so this key is correctly filled.  The incorrect keys are 'Rating', **'Alcohol\_Served'**, and **'Smoking\_Permitted'**. Further clarification is needed for these keys.  After correcting  Here is the final updated dictionary with all values filled accordingly:  {'Cuisine\_Variety': 'Mexican', 'Area': 'open', 'Ambiance': 'quiet', 'Rating': 'low', 'Alcohol\_Served': 'no', 'Smoking\_Permitted': 'no', 'Price\_Range': 'low'} |

# 6. User Experience and Evaluation

* While executing the function dialogue\_mgmt\_system(), I asked for following preferences:

{

'Cuisine\_Variety': 'Japanese',

'Area': 'Open',

'Ambiance': 'Quiet',

'Rating': 'medium',

'Alcohol\_Served': 'no',

'Smoking\_Permitted': 'no',

'Price\_Range': 'low'

}

Recommendations from chatbot

|  |
| --- |
| Here is a summary of the restaurants based on your preferences:  1. Sushi Delight : Authentic Japanese cuisine, quiet ambiance, medium price range  2. Ramen House : Cozy Japanese setting, medium price range  3. Sakura Garden : Japanese cuisine, peaceful outdoor dining area, medium price range |

* Function evaluate\_model\_response() also performed well .
* I experimented with the chatbot to identify a second 'Mexican' cuisine restaurant. During this process, I intentionally provided incorrect keys and phrasing while expecting the correct response to these incorrect inputs. In my view, this was a valuable experiment.

|  |
| --- |
| Let's reassess the keys in the dictionary to ensure they accurately reflect your dining preferences:  1. Cuisine\_Variety: Mexican cuisine - **Correct**  2. Area: Open area - **Correct**  3. Ambiance: Quiet ambiance - **Correct**  4. Rating: Low - **Correct**  5. Alcohol\_Served: No - **Correct**  6. Smoking\_Permitted: No - **Correct**  7. Price\_Range: Low - **Correct**  All keys in the dictionary are now accurately filled based on your preferences. Thank you for providing clear instructions!  **Based on your user profile, here is a list of restaurants that match your preferences:**  **1. Tacos & Tequila : Authentic Mexican cuisine, outdoor seating, quiet ambiance, affordable prices** |

# 7. Drawback

The current chatbot implementation utilizes a dataset with a limited number of rows. Despite this, the recommendation process, once all user preferences are finalized, takes a significant amount of time in minutes. Given that the execution is sequential, it is necessary to adopt an asynchronous processing approach for the dataset to achieve faster response times for users.

# 8. Conclusion

## 8.1 Summary

* Developed a restaurant recommendation bot with advanced personalization features.
* Utilized OpenAI’s GPT for robust natural language understanding and interaction.

## 8.2 Future Work

* **Enhancements Functional:**

The output format of each layer is inconsistent. Use the function API of GPT to standardize it. The dialogue flow doesn't recall product extraction after an intent change. Add a feature to restart the chat after an intent change. Include a user satisfaction rating.

Incorporate feature to check if the user wants to avail dine-in or takeaway and recommend restaurants based on the preference.

* **Enhancements Technical**
  + Web application development.
  + UI development for better user interaction.
  + Add capability to chatbot to check and automatically resolve spelling errors.
  + Optionally store details in some Memcached (redisdb or mongodb) for proper user experience for returning users.