|  |
| --- |
| DuaSAeed |

**Car Recommendation Chatbot**

**Overview**

The Car Recommendation Chatbot is a Flask-based web application designed to help users find cars matching their preferences. By integrating a recommendation system, users can filter cars based on their budget, the minimum year of manufacture, and maximum mileage. The system dynamically provides recommendations using cosine similarity on processed car data.

**Features**

* **Interactive Chat Interface**: Users can interact with the chatbot by entering search parameters (budget, year, mileage) to receive tailored car recommendations.
* **Data-Driven Recommendations**: Filters cars based on the user's criteria and computes the similarity between car attributes.
* **Web-Based Implementation**: Provides a clean, user-friendly web interface using HTML, CSS, and JavaScript for real-time interaction.

**Dependencies**

* **Python Libraries**:
  + Flask: For building the web application.
  + pandas: For data preprocessing and handling.
  + scikit-learn: For computing cosine similarity.
* **Frontend Tools**:
  + HTML/CSS: For structuring and styling the user interface.
  + jQuery: For asynchronous communication between the frontend and backend.

**Data Preprocessing**

The dataset (USA\_cars\_datasets.csv) is preprocessed to enhance recommendation accuracy:

1. Removed unnecessary columns: Unnamed: 0, vin, lot, country, and condition.
2. Filled missing values using median imputation for numeric columns.
3. One-hot encoded categorical variables: brand, title\_status, color, and state.

**System Workflow**

1. **Dataset Preprocessing**:
   * Load the dataset and clean the data.
   * Encode categorical variables for machine learning compatibility.
2. **Backend Functionality**:
   * **Recommendation Function**:
     + Filters cars within the specified budget, year, and mileage.
     + Uses cosine similarity to identify the most similar cars.
     + Returns up to five recommendations as JSON data.
   * **Flask Routes**:
     + /: Renders the chatbot interface.
     + /get\_recommendations: Accepts user input and returns recommendations.
3. **Frontend Interaction**:
   * Users input search parameters (e.g., budget, year, mileage).
   * AJAX requests fetch car recommendations from the backend.
   * Recommendations are displayed dynamically in the chatbox.

**How to Run**

1. Ensure all dependencies are installed:

pip install flask pandas scikit-learn

1. Place the USA\_cars\_datasets.csv file in the same directory as the project.
2. Run the Flask application:

python chatbot.py

1. Open the application in a web browser:

http://127.0.0.1:5000

**File Descriptions**

1. **chatbot.py**:
   * Main Flask application.
   * Contains routes for rendering the UI and processing recommendations.
2. **USA\_cars\_datasets.csv**:
   * Dataset containing car details for recommendation.
3. **DATAbase.ipynb**:
   * (Potentially) Notebook for exploratory data analysis and preprocessing.

**Future Enhancements**

* Integrate advanced recommendation algorithms (e.g., collaborative filtering).
* Add user authentication for saving search preferences.
* Expand dataset with more car attributes.
* Implement a mobile-friendly UI design.