API documentation

▼ Documentation About Apps API

User-Based Recommendation System API

This API allows users to get personalized movie recommendations based on user-based collaborative filtering using cosine similarity. The API is built with Streamlit, which provides an interactive interface for inputting user data and displaying recommended movies.

Prerequisites

- **Python**: Ensure you have Python installed (version 3.6 or higher).
- **Libraries**: Install the necessary libraries by running:

```
pip install streamlit pandas scikit-learn scipy request
s
```

How to Use

1. **Run the Application:**

To start the API, open your terminal or command prompt and run:

```
streamlit run your_script_name.py
```

Replace `your_script_name.py` with the actual name of your Python script file (e.g., `APPknn.py`).

2. **Access the API:**

   Once the application is running, it will provide a local URL (e.g., `http://localhost:8501`). Open this URL in your web browser.

3. **Provide User Inputs:**

   - **User ID**: Enter your `User ID` in the provided input field. Make sure the ID is within the valid range (i.e., between the minimum and maximum values shown).
   - **Number of Recommendations**: Use the slider to select the number of movie recommendations you want (between 1 and 10).

4. **Generate Recommendations:**

   Click the **"Get Recommendations"** button to generate personalized movie recommendations based on your input.

API Output

- The API will display a list of recommended movies for the given user ID, along with predicted ratings. Each movie will be ranked from highest to lowest predicted rating.

Example Usage

1. **Input:**
   - **User ID**: `5`
   - **Number of Recommendations**: `5`
2. **Output:**
   - A list of movie recommendations, such as:

```
1. The Shawshank Redemption (Predicted Rating: 4.75)
2. The Godfather (Predicted Rating: 4.50)
3. Pulp Fiction (Predicted Rating: 4.40)
4. The Dark Knight (Predicted Rating: 4.35)
5. Fight Club (Predicted Rating: 4.30)
```

How It Works

- **Data Loading**: The API fetches movie and rating data directly from raw GitHub URLs.
- **Data Processing**: A user-item matrix is created, representing the ratings of each user for each movie. This matrix is converted to a sparse format for efficient computation.
- **Cosine Similarity Calculation**: Computes the similarity between users based on their movie ratings.
- **Recommendation Generation**: For a given user ID, the API finds similar users, calculates weighted ratings, and suggests movies that the user hasn't rated yet.

Additional Notes

- This application currently uses simulated data to demonstrate the recommendation functionality. Once the app's data source is complete, it will use real-time data for generating recommendations.
- Ensure you have a stable internet connection since the application fetches data from external URLs.

Troubleshooting

- If you receive an error related to data loading, ensure that the URLs for the CSV files are correct and accessible.
- If the recommendations do not display, make sure that the user ID entered is valid and exists within the dataset.

Contributing

Feel free to fork this repository and suggest improvements or add features. Please submit a pull request with a clear explanation of your changes.

▼ **Documentation About Appknn API**

KNN Item-Based Recommendation System (Streamlit)

This Streamlit app provides movie recommendations using an Item-Based Collaborative Filtering approach, powered by the K-Nearest Neighbors (KNN) algorithm. Users can select a movie they like and receive a list of recommended movies based on the similarity between movies.

Prerequisites

- **Python**: Make sure Python (version 3.6 or higher) is installed.
- **Libraries**: Install the required libraries by running:

```
pip install streamlit pandas scikit-learn scipy
```

How to Use the App

1. **Run the Streamlit Application:**

   Open your terminal or command prompt and run:

   ```
   streamlit run your_script_name.py
   ```

   Replace `your_script_name.py` with the actual name of your Python script file.

2. **Access the Application:**

   After running the above command, a local URL (e.g., `http://localhost:8501`) will appear. Open this URL in a web browser.

3. **Select a Movie:**

- Choose a movie title from the dropdown list ("Select a movie you like"). The app loads movie titles from the dataset.

4. **Select the Number of Recommendations:**
   - Use the slider to select the number of recommendations you want to receive (from 1 to 20).
5. **Get Recommendations:**
   - Click the "Get Recommendations" button to display a list of movies similar to the selected movie.

How It Works

- **Data Loading**: The app loads movie and rating data from CSV files hosted on GitHub.
- **User-Item Matrix Preparation**: A matrix is created where rows represent users, columns represent movie titles, and the values are the users' ratings for those movies.
- **Model Training**:
  - The user-item matrix is transposed to form an item-item matrix (with movies as rows and users as columns).
  - The KNN model is trained on this matrix to find similar movies based on cosine similarity.
- **Movie Recommendation**:
  - When a user selects a movie, the app finds similar movies by performing a KNN search in the item-item matrix.
  - The top N similar movies are recommended to the user.

Caching

- The `@st.cache_data` decorator is used to cache data-related operations like loading CSV files and preparing the user-item matrix. This reduces the need to reload data every time the app is rerun.

- The `@st.cache_resource` decorator is used to cache the trained KNN model, preventing unnecessary retraining when the app is rerun.

Example Usage

1. **Select a Movie:**
   - Suppose you choose "The Shawshank Redemption" from the dropdown.
2. **Choose Number of Recommendations:**
   - Use the slider to select `5` recommendations.
3. **Get Recommendations:**
   - Click "Get Recommendations."
   - The app will display a list of 5 movies similar to "The Shawshank Redemption" based on the trained KNN model.

Additional Information

- **Data Source**:
  - `ratings.csv`: Contains user ratings for various movies.
  - `movies.csv`: Contains movie information, including titles and IDs.
- **Algorithm**:
  - K-Nearest Neighbors (KNN) with cosine similarity is used to compute the similarity between movies.

Troubleshooting

- **Data Loading Issues**: Ensure the URLs for `ratings.csv` and `movies.csv` are correct and accessible.
- **Streamlit App Errors**: Make sure all required libraries are installed. If you encounter errors, check for missing or incompatible library versions.