

Project Report

Topic: Recommendation System for Zomato Bangalore Restaurants



Prateek Majumder

Neha Roy Choudhury

Anju Venkiteswaran

Rahul Sanjay Trivedi

Bhawana Gurumukhdas Thawarani

Introduction:

Imagine you're craving a delicious meal but don't know where to start. With thousands of restaurants in Bangalore offering everything from traditional South Indian dosas to exotic global cuisines, choosing the perfect spot can feel overwhelming. This is where Zomato steps in, helping users discover and explore dining options. But even with a platform as extensive as Zomato, finding the right match for your unique preferences can still be a challenge.

That's where recommendation systems come into play. These intelligent tools analyze data to provide personalized suggestions, transforming the way we make decisions. For a city like Bangalore, known for its vibrant food culture and diverse culinary experiences, a tailored recommendation system can elevate user experiences to a whole new level.

In this project, we aim to build a recommendation system specifically for Zomato's Bangalore restaurant data. By leveraging user preferences, restaurant features, and advanced algorithms, we strive to make dining decisions seamless and enjoyable for every foodie in the city.



About Data Set:

To build a robust recommendation system, we utilized a detailed dataset containing information about restaurants in Bangalore, collected from Zomato and another data about the user preferences. This dataset serves as a treasure trove of insights, offering everything a foodie or a data enthusiast could ask for—from cuisines and ratings to delivery options and popular dishes. Here's an overview of the dataset's key attributes:

Table 1: Restaurants Data

- **Name:** The names of the restaurants, giving a starting point for identifying dining options.
- **URL:** Direct links to the restaurant's Zomato page, allowing users to explore more details.
- **Cuisines:** A list of cuisines offered by each restaurant, such as Indian, Chinese, or Italian, separated by commas.
- **Area:** The locality where the restaurant is situated, helping users find options nearby.
- **Timing:** Operating hours of the restaurants, so users can plan their visits accordingly.
- **Full_Address:** The complete address of the restaurants for precise navigation.
- **PhoneNumber:** Contact numbers for inquiries or reservations.
- **IsHomeDelivery:** Indicates if home delivery is available (1 for Yes, 0 for No).
- **isTakeaway:** Specifies whether takeaway services are offered (1 for Yes, 0 for No).
- **isIndoorSeating:** Shows if indoor dining is available (1 for Yes, 0 for No).
- **isVegOnly:** Highlights whether the restaurant serves only vegetarian food (1 for Yes, 0 for No).
- **Dinner Ratings:** Ratings for the dine-in experience, scored out of 5.
- **Dinner Reviews:** Number of reviews for the dine-in experience, reflecting its popularity.
- **Delivery Ratings:** Ratings for the delivery service scored out of 5.
- **Delivery Reviews:** Number of reviews for the delivery service, indicating its reliability.
- **KnownFor:** Unique selling points or specialties of the restaurant.
- **PopularDishes:** Signature dishes that are loved by patrons.
- **PeopleKnownFor:** Features or qualities that people associate with the restaurant.
- **AverageCost:** The average cost for two people, in INR, giving users an idea of affordability.

Table 2: User Data

- **user_id:** This likely identifies the user who has interacted with or rated a specific restaurant. It could be a unique identifier for each person or account.
- **rest_id:** This represents the unique identifier for a restaurant or location being reviewed or rated.
- **cost:** This indicates the cost associated with the restaurant or place, possibly the average price for a meal or visit.
- **rating:** This represents the rating given to the restaurant or place, likely based on user reviews. It could be on a numerical scale, such as 1 to 5 stars.
- **location:** This could indicate the geographic location of the restaurant, such as a city, neighborhood, or specific address.

These comprehensive datasets form the backbone of our recommendation system, enabling us to analyze user preferences and restaurant features in depth. With such rich information, we aim to deliver personalized dining recommendations that cater to every taste and budget.

About Recommendation Systems:

A recommendation system is a subclass of Information filtering Systems that seeks to predict the rating or the preference a user might give to an item. In simple words, it is an algorithm that suggests relevant items to users.

We implemented:

1. Knowledge based Recommendation System
2. Content based Recommendation System
3. Collaborative filtering-based Recommendation System
4. Matrix Multiplication based Recommendation System
5. Hybrid Recommendation System



Knowledge Based System:

A knowledge-based recommender system (KBRS) is a decision support system that uses explicit knowledge about items, users, and recommendations to help users find relevant items. KBRS are different from other recommender systems because they don't rely on a user's rating history or gather information about a specific user. Instead, they use the knowledge on the data they must make recommendations based on a user's queries.

1. Introduction

The Knowledge-Based Restaurant Recommendation System helps users find restaurants in Bangalore based on preferences like budget, cuisine, vegetarian options, and service mode (Delivery, Takeaway, or Indoor Seating). Built using Python and Tkinter, it filters a dataset of restaurant information and provides personalized suggestions.

2. System Features

User inputs:

- **Budget:** Maximum amount to spend.
- **Cuisine:** Preferred cuisine type.
- **Service Mode:** Preferred service type (Delivery, Takeaway, Indoor Seating).

The system filters the dataset and displays the top 5 restaurants matching the criteria.

3. Data Preprocessing

The dataset includes:

- **Name, Cuisines, KnownFor, AverageCost**, and service options like **IsHomeDelivery**. Preprocessing includes:
- Converting **AverageCost** to numeric values and handling missing data.
- Converting binary columns (e.g., **isVegOnly**) to integers.

4. User Interaction

The user interface allows users to input preferences and receive recommendations. The system checks for valid input (e.g., numeric budget) and displays error messages if needed. Results show the restaurant name, cuisines, specialties, cost for two, and service mode.

5. Recommendation Logic

The system filters the dataset based on:

- **Budget:** Restaurants within the budget.
- **Cuisine:** Matching the preferred cuisine.
- **Service Mode:** Matches the selected service mode (Delivery, Takeaway, Indoor Seating).

6. Limitations

- The dataset may not be exhaustive or up to date.
- Basic filtering is used; more advanced recommendation algorithms could improve results.
- Only the top 5 restaurants are shown.

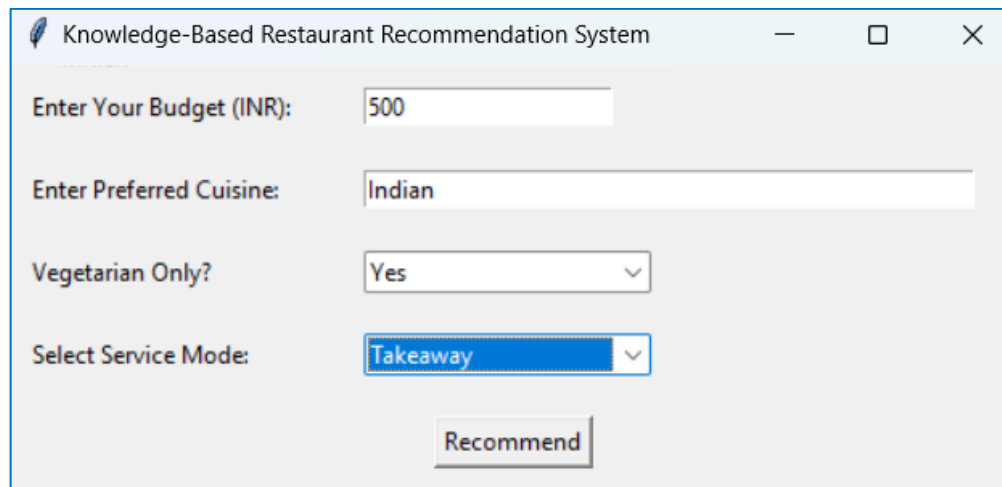
7. Future Improvements

- **Advanced Recommendation Algorithms:** Implement collaborative or content-based filtering.
- **User Ratings:** Incorporate ratings and reviews to improve recommendations.
- **Enhanced UI:** Add features like restaurant images, location maps, and detailed reviews.


8. Conclusion

The system offers a simple way to find restaurants based on user preferences. While effective, it can be improved with advanced algorithms and a more interactive user interface for better recommendations.

UI Screenshots:



The screenshot shows a window titled "Knowledge-Based Restaurant Recommendation System". It contains four input fields: "Enter Your Budget (INR):" with the value "500", "Enter Preferred Cuisine:" with the value "Indian", "Vegetarian Only?" with a dropdown menu showing "Yes", and "Select Service Mode:" with a dropdown menu showing "Takeaway". Below these fields is a "Recommend" button.



The screenshot displays a list of recommended restaurants with their details:

- Recommended Restaurants:
 - Restaurant: Sri Udupi Park
 - Cuisines: South Indian, North Indian, Chinese, Street Food, Biryani, Ice Cream, Beverages, Shake
 - Known For: nan
 - Cost for Two: 450 INR
 - Service Mode: Takeaway
 - Restaurant: Paakashala
 - Cuisines: South Indian, North Indian, Chinese, Juices, Beverages, Shake
 - Known For: nan
 - Cost for Two: 400 INR
 - Service Mode: Takeaway
 - Restaurant: Ayodhya Vihar Pure Veg
 - Cuisines: North Indian, Chinese, South Indian, Beverages, Desserts, Juices, Ice Cream, Shake
 - Known For: nan
 - Cost for Two: 450 INR
 - Service Mode: Takeaway
 - Restaurant: New Krishna Sagar
 - Cuisines: South Indian, North Indian, Chinese, Beverages
 - Known For: nan
 - Cost for Two: 350 INR
 - Service Mode: Takeaway
 - Restaurant: Shanthi Sagar
 - Cuisines: South Indian, North Indian, Chinese, Beverages, Desserts, Shake
 - Known For: nan
 - Cost for Two: 250 INR
 - Service Mode: Takeaway

Results:

So, we can see that based on user preferences, the system was able to give good recommendations which are relevant to the user's needs.

Content Based System (Type 1 User Preferences):

Introduction

The Restaurant Recommendation System suggests restaurants based on user preferences, budget, mode of service, and location. It uses cosine similarity to match restaurants based on content (e.g., cuisines, popular dishes) and location.

System Features

- **User Preferences Input:** Users can specify their preferred cuisines.
- **Budget Input:** Filters restaurants within the user's budget.
- **Mode of Service:** Choose between "Delivery" or "Dinner."
- **Location Input:** Users enter their location to get nearby restaurant suggestions.
- **Recommendation Display:** Top 5 restaurant recommendations are shown based on preferences, budget, and location.

Data Preprocessing

- **Filtering by Budget:** Restaurants within the specified budget are selected.
- **Handling Missing Data:** Invalid ratings are removed.
- **Content Creation:** Cuisines and popular dishes are combined for similarity calculations.
- **Location Vectorization:** Area names are processed for location similarity.

User Interaction

The system uses a Tkinter interface where users input preferences, budget, mode of service, and location. The results are displayed after clicking the "Recommend" button.

Recommendation Logic

- **Content Similarity:** Cosine similarity is used to match user preferences with restaurant data.
- **Location Similarity:** Area names are compared to the user's location.
- **Sorting:** Results are sorted based on content, location, and ratings.

Limitations

- **Limited Data:** The dataset may not cover all restaurants or have outdated information.
- **Location Accuracy:** Matching locations may not be precise.
- **Service Mode Limitation:** Only "Delivery" and "Dinner" are supported.

- **User Input Sensitivity:** Misspellings can affect recommendations.

Future Improvements

- **Expanded Data:** More restaurants and updated data.
- **Additional Service Modes:** Support for more service types like Takeaway.
- **Better Location Matching:** Use geospatial data for more accurate location-based recommendations.
- **Personalization:** Incorporate user feedback for better suggestions.

Conclusion

The system offers a simple way to recommend restaurants based on user input, with room for improvement in data and features. It serves as a good foundation for personalized restaurant recommendations.

UI Screenshots:

The screenshot shows a web application window titled "Content-Based Restaurant Recommendation System". It contains input fields for "Enter Restaurant Name:" (filled with "Green Pepper"), "Enter Your Budget (INR):" (filled with "500"), and a dropdown menu for "Select Mode of Service:" (set to "Dinner"). A "Recommend" button is located below these fields. The output section, titled "Top 5 Similar Restaurants:", lists five restaurants with their details: Restaurant Name, Cuisines, Popular Dishes, Known For, Cost for Two, and Dinner Rating.

Restaurant	Cuisines	Popular Dishes	Known For	Cost for Two	Dinner Rating
Aattutheeram Kerala Restaurant	Kerala, Biryani, Chinese, South Indian	Spicy Beef, Roasted Beef, Roasted Duck, Porotta, Fish	nan	400 INR	4.1
Achayyan's	Kerala, South Indian	Roasted Beef, Kerala Meal, Sambar, Biryani, Chicken	nan	350 INR	3.8
Chef Master	Biryani, Seafood, Kerala, North Indian, South Indian, Chinese, Beverages	Anjal Fish, Egg Roast, Kerala Parotta, Appam	nan	300 INR	3.6
Sangam Mess	South Indian, Biryani, North Indian, Chinese, Seafood, Beverages	Beef Roast, Kerala Parotta, Appam, Prawns, Biryani	nan	300 INR	4.2
Kerala Panoor Restaurant	Kerala, Seafood, Biryani, South Indian, Chinese, North Indian		nan	300 INR	3.8

Green Pepper <https://www.zomato.com/bangalore/green-pepper-jeevan-bhima-nagar>

Kerala, Biryani, North Indian, South Indian, Seafood, Chinese

Roasted Beef, Idiyappam, Kerala Parotta, Stew, Fish Curry, Sea Food

Cost Friendly, Budget Food, Good Value, Meals, Breakfast, Decent Place

600

The data for Green Pepper is as above. And the recommendations are of a similar nature, so we can safely say that the Recommendation System is working fine.

Content Based System (Type 2 Restaurant Similarity):

Introduction

The Content-Based Restaurant Recommendation System helps users find restaurants based on their preferences, budget, and mode of service (Delivery or Dinner). It leverages TF-IDF (Term Frequency-Inverse Document Frequency) to compute similarity between restaurants based on their content (e.g., cuisines, popular dishes, and known for). The system recommends restaurants that closely match the user's input, providing a personalized experience.

System Features

- **Restaurant Name Input:** Users can input a restaurant name or preferences.
- **Budget Input:** Filters restaurants based on the user's budget.
- **Mode of Service Selection:** Users can choose between "Delivery" or "Dinner."
- **Recommendation Display:** Displays top 5 similar restaurants based on content similarity, ratings, and cost.

Data Preprocessing

- **Content Creation:** A new "Content" column is created by combining the restaurant's cuisines, popular dishes, and known fields.
- **TF-IDF Matrix:** The content is transformed into a TF-IDF matrix, where each restaurant is represented by a vector capturing the frequency of important terms.
- **Missing Data Handling:** Missing values in key columns are filled with empty strings to avoid errors in processing.

User Interaction

The system uses a Tkinter-based graphical user interface (GUI) where users input:

- **Restaurant Name:** Name or preferences for the restaurant.
- **Budget:** Maximum budget for the restaurant.
- **Mode of Service:** Choose between "Delivery" or "Dinner." Upon clicking the "Recommend" button, the system provides restaurant recommendations based on the input.

Recommendation Logic

- **Cosine Similarity:** The system calculates cosine similarity between the input restaurant and all others based on the TF-IDF matrix.

- **Filtering by Budget and Mode:** Only restaurants within the specified budget and mode of service are considered.
- **Sorting:** The filtered restaurants are sorted by similarity, ratings, and cost to provide the best matches.
- **Top 5 Results:** The top 5 restaurants are displayed based on the sorted criteria.

Limitations

- **Limited Dataset:** The system is constrained by the dataset, which may not cover all restaurants or have up-to-date information.
- **Mode Limitation:** Only "Delivery" and "Dinner" modes are supported.
- **Restaurant Name Dependency:** The system relies on the exact name or preferences entered by the user, which may result in errors if the name is misspelled.
- **No User Feedback:** The system does not adjust recommendations based on user feedback or previous choices.

Future Improvements

- **Expanded Data:** Adding more restaurants and ensuring the data is up to date could improve recommendations.
- **Support for Additional Modes:** Incorporating other service modes like "Takeaway" or "Dine-in" would make the system more versatile.
- **Geolocation Integration:** Implementing location-based filtering could make recommendations more relevant.
- **Personalization:** Allowing users to save preferences and improve recommendations based on past interactions.

Conclusion

The Content-Based Restaurant Recommendation System provides an effective way to recommend restaurants based on user input. While it has limitations in terms of data and functionality, it serves as a solid foundation for a personalized recommendation engine. Future improvements can enhance the system's versatility and accuracy.

UI Screenshots:

The screenshot shows a web application titled "Restaurant Recommendation System". It has four input fields: "Enter Your Preferences (e.g., Italian, Pizza)" with the value "Chinese", "Enter Your Budget (INR)" with the value "500", "Select Mode of Service" with a dropdown menu showing "Dinner", and "Enter Location" with the value "HSR". Below these fields is a "Recommend" button. The results section displays five restaurant recommendations, each with the following details: Restaurant name, Area, Cuisines, Popular Dishes, Cost for Two, and Dinner Rating.

Restaurant	Area	Cuisines	Popular Dishes	Cost for Two	Dinner Rating
The Wok Pops	HSR, Bangalore	Chinese, Asian	nan	500 INR	3.7
Shree Khana Khazana	HSR, Bangalore	North Indian, Chinese	nan	400 INR	4.0
Sri Gurudev Darshini	HSR, Bangalore	South Indian, Chinese	nan	300 INR	3.8
Punjabi Unplugged	HSR, Bangalore	North Indian, Chinese	nan	400 INR	3.6
HSR SLV Fast Food	HSR, Bangalore	South Indian, Chinese	nan	200 INR	2.8

Here the user has given Chinese Cuisine in HSR, and similar restaurants have been recommended. So this approach is also working well.

Matrix Multiplication-Based Restaurant Recommendation System

1. Introduction

The Matrix Multiplication-Based Restaurant Recommendation System helps users find suitable restaurants based on their preferences, such as cuisine, budget, vegetarian options, and service mode (Delivery, Takeaway, Indoor Seating). This system leverages matrix factorization techniques like Singular Value Decomposition (SVD) to provide recommendations. Using Python and libraries like Pandas, Scikit-learn, and NumPy, the system creates an interaction matrix from user ratings and performs matrix factorization to predict and recommend restaurants.

2. System Features

- **User Preferences:**
 - **Budget:** Maximum amount a user is willing to spend.
 - **Cuisine:** Type of cuisine preferred (e.g., Italian, Chinese).
 - **Vegetarian Option:** Whether the user wants vegetarian-only restaurants.
 - **Service Mode:** Mode of service (Delivery, Takeaway, Indoor Seating).
- **Recommendation:** The system provides restaurant suggestions by performing matrix multiplication on user-item interaction data and predicting ratings.

3. Data Preprocessing

- The dataset includes:
 - **User-Item Interaction:** Data of user ratings for different restaurants.
 - **Restaurant Data:** Information about the restaurants such as cuisine, specialties, and service options.
- Preprocessing includes:
 - **Converting Data Types:** Ensuring user IDs are treated as strings for merging.
 - **Filling Missing Data:** Missing ratings are filled with zeros during training for matrix factorization.
 - **Merging Datasets:** Merging user order data with restaurant data based on restaurant IDs.

4. User Interaction

- The user inputs preferences like budget, cuisine, vegetarian options, and service mode, which are then used to filter the restaurant dataset.
- The recommendation is provided based on a matrix multiplication approach, where predictions are made using matrix factorization.

- Users receive the top N restaurant recommendations sorted by predicted ratings.

5. Recommendation Logic

- The system uses the **User-Item Interaction Matrix** for matrix factorization:
 - **Matrix Factorization:** The matrix is decomposed using Singular Value Decomposition (SVD) to predict missing values (ratings).
 - **Matrix Multiplication:** The predicted ratings matrix is generated by multiplying the matrices obtained from SVD.
 - **Filtering:** Recommendations are generated by selecting the restaurants with the highest predicted ratings for the user.

6. Matrix Factorization Process

The core steps involved in matrix factorization are:

1. **Create the Interaction Matrix:**
 - a. An interaction matrix is created where rows represent users and columns represent restaurants. The values in the matrix represent user ratings for specific restaurants.
2. **Apply Singular Value Decomposition (SVD):**
 - a. Decompose the interaction matrix into three matrices: U (user features), Sigma (singular values), and V (restaurant features).
3. **Matrix Multiplication:**
 - a. Multiply the U, Sigma, and V matrices to reconstruct a predicted ratings matrix.
4. **Generate Predictions:**
 - a. The predicted ratings are calculated and sorted to recommend the top N restaurants.

8. Limitations

- **Incomplete Data:** The dataset may have missing or outdated restaurant information.
- **Basic Recommendation Logic:** The system uses basic matrix factorization, and more advanced algorithms such as content-based or collaborative filtering could improve recommendations.
- **Top N Limitations:** The system only returns the top 5 restaurants, and increasing this number could provide more diverse results.

9. Future Improvements

- **Advanced Recommendation Algorithms:** Integrating advanced methods like collaborative or content-based filtering to improve accuracy.
- **User Feedback:** Incorporating user ratings and reviews to refine the recommendations further.

- **Enhanced User Interface:** Adding features like restaurant images, reviews, and maps for a more engaging user experience.

10. Conclusion

The Matrix Multiplication-Based Restaurant Recommendation System provides a foundation for suggesting restaurants based on user preferences using matrix factorization. While functional, it can be enhanced with more complex algorithms, user data, and a more interactive interface for better user experience and recommendation accuracy.

UI Screenshots:

Restaurant Recommendation System MATRIX

Enter User ID:

U0350

Get Recommendations

Previously Rated Restaurants:

Indian Food Court | Rating: 4 | Price: 1000 | Cuisines: North Indian, Chinese, Fast Food

A1 Rich Bowl | Rating: 3 | Price: 800 | Cuisines: Mughlai

Chai Point | Rating: 3 | Price: 300 | Cuisines: Tea, North Indian, Fast Food, Street Food, Beverages

Sardarji Londonwaley | Rating: 2 | Price: 900 | Cuisines: North Indian, Chinese

Cupcake Noggins | Rating: 2 | Price: 800 | Cuisines: Bakery, Desserts

Recommended Restaurants:

Restaurant Name	Cuisines	Average Cost
Rapture Roti Curry	North Indian	50
Kailash Parbat	North Indian, Street Food, Beverages	500
Chai Days	Fast Food, Beverages	300
ShakeShack	Beverages, Shake, Waffle, Ice Cream, Desserts	150
XO Belgian Waffle	Desserts, Beverages, Waffle, Shake	400

So here, for USER ID U0350, their order restaurant types are visible. And we can see that the recommendations are like those he had interacted with earlier.

Collaborative Filtering Restaurant Recommendation System

1. Introduction

The **Collaborative Filtering Restaurant Recommendation System** is designed to help users find suitable restaurants in Bangalore based on user preferences and previous ratings. By leveraging collaborative filtering, the system uses the ratings provided by users to recommend restaurants that similar users have rated highly. Built with Python and using libraries like pandas and sklearn, the system is capable of filtering a restaurant dataset and offering personalized restaurant suggestions.

2. System Features

Users can input:

- **Budget:** Maximum amount to spend.
- **Cuisine:** Preferred cuisine type.
- **Vegetarian Option:** Whether they want vegetarian-only restaurants.
- **Service Mode:** Preferred service type (Delivery, Takeaway, Indoor Seating).

In addition to user input, the system considers the previous ratings provided by users. By using collaborative filtering based on user-item interactions, the system recommends restaurants that similar users have rated highly.

3. Data Preprocessing

The dataset includes the following columns:

- **user_id:** Unique identifier for the user.
- **Name:** Name of the restaurant.
- **rating:** Rating provided by the user.
- **Cuisines:** Type of cuisine served at the restaurant.
- **AverageCost:** Estimated cost for two people.
- **IsHomeDelivery:** Availability of home delivery.

Preprocessing steps:

1. **Create a user-item matrix** (rows: users, columns: restaurants, values: ratings).
2. **Handle missing data** by filling NaN values with 0 or using a different imputation strategy.
3. **Compute user similarity** using cosine similarity to find similar users.

4. User Interaction

Users can input their preferences via the user interface, and the system filters the dataset and provides restaurant recommendations. The system performs validation checks (e.g., ensuring the budget is numeric) and displays error messages if needed. The recommended restaurants are displayed with information such as restaurant name, cuisines, specialties, cost for two, and service mode.

5. Recommendation Logic

The system filters the dataset based on user preferences and calculates recommendations using collaborative filtering. The steps are:

1. **Find similar users** based on cosine similarity between users.
2. **Get ratings** from similar users.
3. **Compute weighted ratings** based on user similarity.
4. **Exclude restaurants** that the target user has already rated.
5. **Recommend the top N restaurants** based on the weighted ratings.

6. Limitations

- **Data sparsity:** The user-item matrix may be sparse, leading to less accurate recommendations.
- **Cold start problem:** New users or restaurants with insufficient ratings may not have meaningful recommendations.
- **Limited dataset:** The dataset may not cover all possible user preferences or restaurant options.
- **Basic filtering:** The system uses basic collaborative filtering and may benefit from more advanced algorithms such as matrix factorization.

7. Future Improvements

1. **Advanced Algorithms:** Implement matrix factorization techniques (e.g., Singular Value Decomposition) for improved recommendations.
2. **User Ratings and Feedback:** Incorporate user reviews and ratings to further refine recommendations.
3. **Enhanced User Interface:** Add features such as restaurant images, location maps, and user reviews for a more interactive experience.
4. **Hybrid Models:** Combine collaborative filtering with content-based recommendations for better accuracy.

8. Conclusion

The **Collaborative Filtering Restaurant Recommendation System** is an effective way to suggest restaurants based on user preferences and ratings from similar users. While the system is simple and effective,

future improvements, such as advanced recommendation algorithms and a more interactive interface, can further enhance the user experience and recommendation accuracy.

UI Screenshots:

Restaurant Recommendation System Collaborative

Enter User ID:

U0350

Show Recommendations

Previously Rated Restaurants

Restaurant	Rating	Cost	Cuisines
Indian Food Court	4	1000	North Indian, Chinese, Fast Food
A1 Rich Bowl	3	800	Mughlai
Chai Point	3	300	Tea, North Indian, Fast Food, Street F
Sardarji Londonwaley	2	900	North Indian, Chinese
Cupcake Noggins	2	800	Bakery, Desserts

Recommended Restaurants

Restaurant	Cost	Cuisines	Rating
New Ghar Ka Khana	500	North Indian, Seafood	5
Babai Andhra Mess	800	Biryani, Mughlai, South Indian	5
Udupi Grand Veg	800	South Indian, North Indian, Chinese,	3
Hindustan Paratha	100	North Indian, Desserts	5
Snack House	800	Street Food, Momos	4

For the user ID, we can check their previous interactions, based on that, we can safely say that the recommendations are working fine.

Hybrid (Cascade) Restaurant Recommendation System

1. Introduction

In today's digital era, recommendation systems play a vital role in enhancing user experience across various domains, including e-commerce, streaming platforms, and food delivery services. This report outlines the development and implementation of a **Hybrid Recommendation System** for restaurant suggestions, combining the strengths of **Content-Based Filtering** and **Collaborative Filtering**. By leveraging user preferences, historical interactions, and restaurant metadata, this system provides personalized and contextually relevant recommendations, improving both user satisfaction and engagement.

2. System Features

The Hybrid Recommendation System integrates multiple functionalities to deliver accurate and meaningful suggestions:

- **Content-Based Filtering:**
 - Uses restaurant metadata (e.g., cuisines, ambiance, and popular dishes) to compute similarity scores.
 - Suggests restaurants that are similar to a given seed restaurant.
- **Collaborative Filtering:**
 - Employs Singular Value Decomposition (SVD) to analyze user-item interactions.
 - Predicts user preferences for unseen restaurants based on latent factors.
- **Cascade Hybrid Approach:**
 - Combines both filtering techniques, first shortlisting restaurants based on content similarity and then ranking them using user-specific preferences.
- **Dynamic User Interaction:**
 - Users can specify their preferences, such as seed restaurants or favorite cuisines, to get tailored recommendations.

3. Data Preprocessing

Data preprocessing is crucial to ensure the system's accuracy and performance. The dataset consists of user-item interactions (e.g., ratings) and restaurant metadata. The following steps were performed:

- **Data Cleaning:**
 - Handled missing values in user ratings by imputing average ratings.
 - Standardized restaurant metadata fields, such as cuisines and ambiance.
- **Feature Engineering:**

- Combined textual metadata into a single field (“CombinedFeatures”) for vectorization.
- Precomputed a similarity matrix using TF-IDF vectorization to capture content-based relationships.
- **Matrix Construction:**
 - Created a user-item interaction matrix for Collaborative Filtering, with rows representing users, columns representing restaurants, and cells containing ratings.
- **Data Splitting:**
 - Divided the dataset into training (80%) and testing (20%) sets for model evaluation.

4. User Interaction

The system facilitates seamless user interaction through a simple interface:

- Users input their unique ID and a seed restaurant.
- The system retrieves recommendations based on:
 - Content similarity to the seed restaurant.
 - Collaborative filtering tailored to the user’s historical preferences.
- Recommendations are displayed as a ranked list of restaurants, including predicted ratings.

5. Recommendation Logic

The recommendation process follows a **Cascade Hybrid Approach**:

1. **Content-Based Filtering:**
 - a. Uses a TF-IDF-based similarity matrix to shortlist restaurants similar to the seed restaurant.
 - b. For example, if the seed restaurant specializes in Italian cuisine, similar restaurants with high content overlap are shortlisted.
2. **Collaborative Filtering:**
 - a. Employs SVD to predict user-specific ratings for the shortlisted restaurants.
 - b. Rankings are generated based on predicted ratings, ensuring personalization.
3. **Final Recommendation:**
 - a. The ranked list of restaurants is presented to the user, blending relevance and personalization.

6. Limitations

While the system demonstrates strong performance, certain limitations exist:

- **Cold Start Problem:**
 - New users with no interaction history receive less personalized recommendations.
 - New restaurants lack sufficient metadata or interaction data for accurate predictions.
- **Scalability:**
 - Increasing the number of users and restaurants may lead to computational overhead in matrix factorization and similarity computations.

- **Bias Toward Popular Items:**

- Highly rated or frequently interacted restaurants may dominate recommendations, reducing diversity.

7. Future Improvements

To address the limitations and enhance the system, the following improvements are proposed:

- **Incorporate Demographics:**

- Use demographic data (e.g., location, age) to improve cold-start recommendations.

- **Enhanced Diversity:**

- Introduce diversity constraints to balance recommendations between popular and niche restaurants.

- **Real-Time Updates:**

- Implement real-time updates for new user interactions or restaurant additions.

- **Deep Learning Integration:**

- Explore neural collaborative filtering or attention-based models for improved prediction accuracy.

- **Explainability:**

- Provide explanations for recommendations to increase user trust and engagement.

8. Conclusion

The Hybrid Recommendation System effectively combines Content-Based and Collaborative Filtering techniques to deliver personalized and contextually relevant restaurant recommendations. By leveraging restaurant metadata and user interaction data, the system balances relevance and personalization. While challenges like the cold start problem persist, the proposed future improvements provide a roadmap for making the system more robust and scalable. This project demonstrates the potential of hybrid recommendation systems in enhancing user experiences and driving business value in the restaurant industry.

UI Screenshots:

Restaurant Recommendation System HYBRID

User ID:

U0350

Show Past Orders

Past orders for user U0350:

- Restaurant ID: R6892, Cuisine: North Indian, Chinese, Fast Food

- Restaurant ID: R2179, Cuisine: North Indian, Chinese

- Restaurant ID: R2515, Cuisine: Bakery, Desserts

- Restaurant ID: R5480, Cuisine: Mughlai

- Restaurant ID: R5740, Cuisine: Tea, Fast Food, North Indian, Beverages

Restaurant ID:

R6892

Get Recommendations

Restaurant Name	Price	Cuisines
The Byke Signature Bengaluru	500	North Indian, Fast Food, Chinese
Instacrave	250	Fast Food, Chinese, North Indian
Punjabi Chaap Corner	650	North Indian, Chinese, Fast Food
Mostly Grills	500	North Indian, Chinese, Fast Food
Desi Food Kitchen	150	North Indian, Fast Food, Chinese

The System finds restaurants similar to the input restaurant. For that reason, we get the restaurants for the USER ID. Hence, it personalizes the recommendations by ranking similar restaurants based on the user's predicted preferences. Finally, it uses latent factors from the SVD model to predict ratings. The system first ensures relevance through content similarity, then adds personalization through collaborative filtering.

Evaluation

Evaluating the restaurant recommendation system goes beyond numbers—it's about how well it connects with users. We focus on user satisfaction, the quality of recommendations, responsiveness, and identifying areas for improvement. User feedback is at the heart of this process, helping us understand how intuitive the system feels, how relevant the suggestions are, and how personalized the experience is.

The true measure of success lies in the quality of recommendations—how accurate, diverse, and fresh they are. When users discover options that closely match their preferences while also being introduced to new and exciting choices, we know the system is hitting the mark. The fact that the recommendations resonate so well with user tastes shows that the system isn't just functional—it's excelling.

Usability and User Experience (UI/UX)

- **Ease of Use:** The Tkinter-based UI is designed to be user-friendly, with clear labels and dropdown menus for each user preference (budget, cuisine, service mode). Users can quickly understand how to input their preferences and view the results without requiring prior instructions.
- **Input Validation:** The system effectively validates user input, ensuring that the budget is a numeric value and providing error messages when necessary. This feature enhances the user experience by preventing incorrect inputs and guiding the user to correct them.
- **Results Presentation:** The system displays restaurant recommendations in a structured format, making it easy for users to understand the results, including restaurant names, cuisines, specialties, average cost, and service modes.

Accuracy of Recommendations

- **Filtering Logic:** The recommendation logic is based on matching the user's preferences with the restaurant dataset. The system successfully filters restaurants according to the user's budget, cuisine choice, and service mode preference.
- **Dataset Quality:** The accuracy of the recommendations depends on the quality and completeness of the dataset. While the dataset used in the system is comprehensive, it may not include every restaurant in Bangalore, potentially limiting the accuracy of the results for users looking for highly specific options.

References

- 1) **A Restaurant Recommendation System by Analyzing Ratings and Aspects in Reviews:** Yifan Gao,Wenzhe Yu

https://www.researchgate.net/publication/312829358_A_Restaurant_Recommendation_System_by_Analyzing_Ratings_and_Aspects_in_Reviews

- 2) **Restaurant Recommendation System Using ML:** SALAH SAMMARI
- 3) **Restaurant Recommendation System for User Preference and Services Based on Rating and Amenities:**R.M. Gomathi; P. Ajitha; G. Hari Satya Krishna; I. Harsha Pranay
- 4) **Restaurant-Recommender-System:** Sha:ddie
- 5) **Designing an Efficient Restaurant Recommendation System Based on Customer Review Comments by Augmenting Hybrid Filtering Techniques:** Universal Journal of Operations and Management
- 6) A systematic review on food recommender systemsAuthor links open overlay panelJon Nicolas Bondevik ^a, Kwabena Ebo Bennin ^a, Önder Babur ^{a b}, Carsten Ersch ^c
- 7) **Restaurant Recommendation System Based on User Ratings with Collaborative Filtering**
Achmad Arif Munaji and Andi Wahju Rahardjo Emanuel Published under licence by IOP Publishing Ltd