SWIGGY'S RESTAURANT RECOMMENDATION SYSTEM USING STREAMLIT

1. Introduction

Food delivery platforms generate large volumes of restaurant metadata, user ratings, and cuisine information. To improve user decision-making, personalized recommendation systems have become essential.

This project presents a Content-Based Restaurant Recommendation System using Cosine Similarity and deploys it using Streamlit. The system analyzes user preferences such as city, cuisine, rating, and cost, and recommends the most relevant restaurants.

2. Objectives

The main objectives of the project are:

- To clean and preprocess restaurant data extracted from Swiggy.
- To encode textual categorical fields using One-Hot Encoding (city) and Multi-Hot Encoding (cuisine).
- To build a Cosine Similarity–based recommendation engine.
- To deploy the system through an interactive Streamlit web application.
- To provide meaningful insights into restaurant options based on user selections.

3. Dataset Description

Column	Description
id	Unique restaurant ID
name	Restaurant name
city	Location
rating	User rating
rating_count	Number of reviews
cost	Price for two
cuisine	Comma-separated cuisine list
lic_no	License number
link	Swiggy link
address	Restaurant address
menu	Menu snippets

Data Types

- Categorical: name, city, cuisine
- Numerical: rating, rating_count, cost

4. Data Preprocessing

4.1 Data Cleaning

Performed using clean_data.py.

Steps included:

✓ Removed duplicate restaurants
✓ Dropped rows with missing name, city, cuisine, address
✓ Converted rating and cost to numeric
✓ Filled missing rating_count with 0

✓ Normalized inconsistent formatting
✓ Saved output → cleaned_data.csv

4.2 Feature Engineering & Encoding

Performed using preprocess.py.

City Encoding

- Used OneHotEncoder
- Each city → numeric binary vector
- Saved encoder → city_encoder.pkl

Cuisine Multi-Hot Encoding

- Split string: "Chinese, South Indian" → ["Chinese", "South Indian"]
- Generated binary columns for each unique cuisine
- Saved encoder → cuisine_encoder.pkl

Final Encoded Dataset

- Combined categorical + numeric features
- Saved final output → encoded_data.csv

Important:
Indices between cleaned_data.csv and encoded_data.csv remained perfectly aligned for accurate recommendation mapping.

5. Recommendation Methodology

Our system uses Content-Based Filtering with Cosine Similarity.

Steps:

➢ User Input (Streamlit UI)

User selects:

- City
- Cuisine
- Minimum rating
- Maximum cost

- Number of results (Top N)

➢ User Preference Vector Construction

- City → One-Hot encoding
- Cuisine → Multi-Hot encoding
- Rating & Cost → numeric values added to vector

➢ Cosine Similarity Model

The user vector is compared with each restaurant vector from encoded_data.csv:

$$Similarity = \frac{A \cdot B}{|| A || \cdot || B ||}$$

➢ Filtering

The system applies:

✔ City match
✔ Cuisines containing user-selected cuisine
✔ Rating ≥ min_rating
✔ Cost ≤ max_cost

➢ Ranking

Restaurants are sorted by:

1. Highest cosine similarity
2. Highest rating

➢ Output Display

Streamlit shows restaurant cards with:

- Name
- City
- Cuisines
- Ratings & reviews
- Cost
- Address
- Swiggy link

6. Key Results and Insights

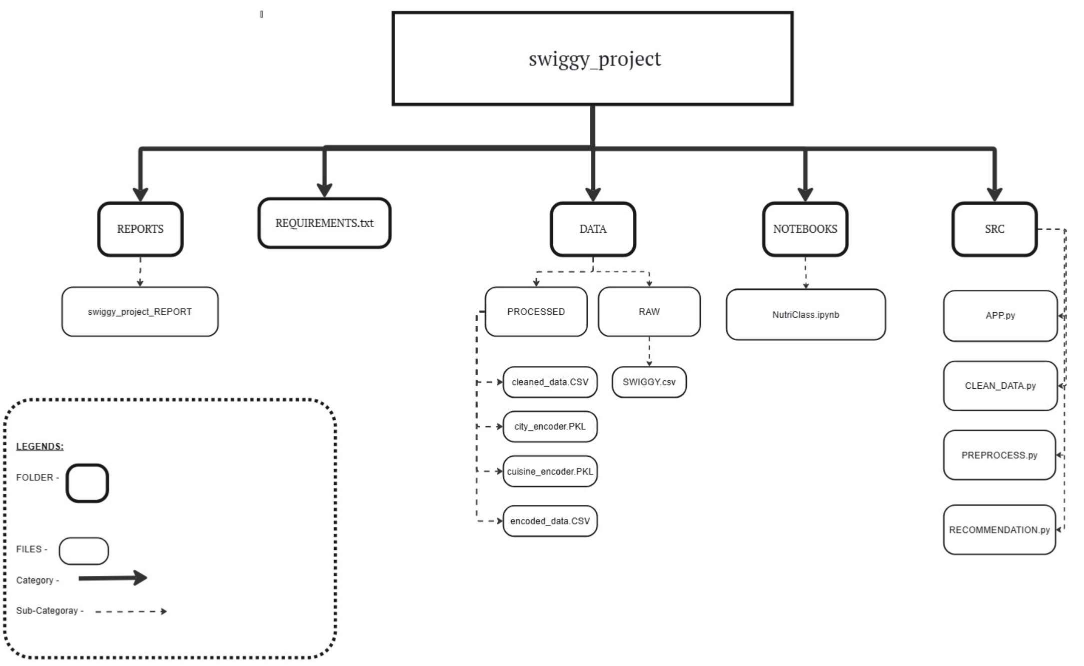
Results Achieved

- Developed a complete end-to-end recommendation engine.
- Cosine similarity delivered highly relevant recommendations.
- Multi-Hot cuisine encoding significantly improved match accuracy.
- The Streamlit UI provides fast, clean, and interactive user experience.

Insights

- Cosine similarity works extremely well for textual categorical features.
- Cuisine similarity contributes strongly to the final ranking.
- Clean data drastically improved filtering accuracy.
- One-Hot + Multi-Hot combination is ideal for restaurant recommendation.

7. System Architecture



8. Streamlit Application

Filters Provided

- Select City
- Select Preferred Cuisine
- Minimum Rating
- Maximum Cost
- Number of Recommendations

Output Display

Each recommendation card includes:

- Restaurant Name
- City
- Cuisine(s)
- Rating & Rating Count
- Cost for Two
- Full Address
- Direct Swiggy Link (clickable)

9. Evaluation Metrics (Qualitative)

Recommendation Quality

- High relevance due to cosine similarity
- Cuisine-aware ranking
- City-based filtering prevents irrelevant results

Application Usability

- Fast load time
- Clean UI
- Responsive filters

Data Alignment

- Encoded & cleaned CSV indices match

- Ensures accurate mapping from encoded → original details

10. Conclusion

This project successfully implemented a fully functional Restaurant Recommendation System using:

- Data Cleaning
- One-Hot & Multi-Hot Encoding
- Cosine Similarity
- Streamlit Deployment

The system provides accurate, personalized, and scalable restaurant suggestions based on user preferences.

Skills Demonstrated

✓ Python Programming
✓ Data Preprocessing
✓ Encoding Techniques
✓ Recommendation Systems
✓ Streamlit Application Development

11. Future Enhancements

These features can be added later:

- Add K-Means clustering
- Deployment on Streamlit Cloud
- Add restaurant images
- Add maps for location-based recommendations
- Integrate real-time API data
- Implement collaborative filtering
- Add sentiment analysis on customer reviews