Final Project Report

Recipe Recommendation Systems

Group - The Recipe Wizards

Naveen Kumar Reddy Veeramreddy | Avinash Vishnu

Yashkumar rajubhai Prajapati | Katasani Shashank reddy

DSC 478 Programming Machine Learning Applications

Prof. Roselyne Tchoua

March 10, 2025

Executive Summary

Many individuals struggle in meal planning as there are plenty of factors like personal preferences, available ingredients, perishable ingredients and possible meals that one can make, which one should consider, and it is not an easy task to manage everything and take a decision. Our application addresses the outlined problems with robust **Recommender Systems** that suggest complementary ingredients based on user input, dietary restrictions, nutritional goals, and past preferences.

Data

We leveraged two different datasets (Ingredients and Ratings) from Kaggle. **Ingredients**, the former dataset comprises Food Item, Veg/ Non-Veg, Cuisine type and List of ingredients columns. **Ratings,** the latter dataset primarily consists of User ratings with Food ID, User ID and Ratings (1-10) columns.

Analysis Approach

We implemented the following high-level steps throughout the entire project:

- Data Pre-processing
- Exploratory Data Analysis
- Hybrid Recommender Systems (Rule + Content based)
- Collaborative filtering (Model based & Memory based)
- Model based Collaborative filtering Evaluation

Evaluation metrics

We used multiple metrics which include **Cosine Similarity, Coverage** for each of the approaches to recommend the food items or ingredients. We evaluated our system only in Model based Collaborative filtering where the evaluation metrics used were **Root Mean Square Error (RMSE) and Mean Absolute Error (MAE)**. We calculated these on both training and test data to ascertain reliability and avoid overfitting, to confirm that our model performs well on unseen data.

Model based collaborative filtering (User)	**Train Data**	**Test Data**
RMSE	0.156	0.257
MAE	0.030	0.042

Introduction

Our project comes under the category of Food and Grocery which has been a major sector for a long time. In this project, we are building a recommender system that can help any individual to identify missing ingredients to make a food item or recommend food items based on user input and past preferences. We also consider the other segments of users and food items which are similar in making the recommendations.

Recommendation systems have been a widely used technique at various organizations to better serve customers by suggesting the content or items which are more personalized to an individual. The system that we build can be leveraged by grocery stores to cross sell the products, restaurants to better serve their customers and families to save money on ingredients wastage.

There are many types of recommender systems that are popularly used nowadays. It includes Rule based recommender systems where recommendations are purely based on only if user preferences matched with certain type of feature, it will be recommended. There is an extension of rule based which is Content based where suggestions are made based on matching criteria with certain threshold, from the similarity metric. In the content based there are no requirements to past preferences or interactions of the user are required. One of the popular recommendation systems is Collaborative filtering which primarily provides recommendations based on the user-item matrix. It depends on the user and other similar user preferences and robust compared to other systems. There are variations in Collaborative filtering like Memory based, Model based which are used based on the size of the data needs to be handled or the complexity of the problem.

In this project, we experiment with most of the above systems on datasets collected from Kaggle.

Data Description

Two different datasets (Ingredients and Ratings) are used for this project. The ingredients dataset primarily consisted of information on recipes whereas the second dataset included information about ratings of different on the food items in the first dataset.

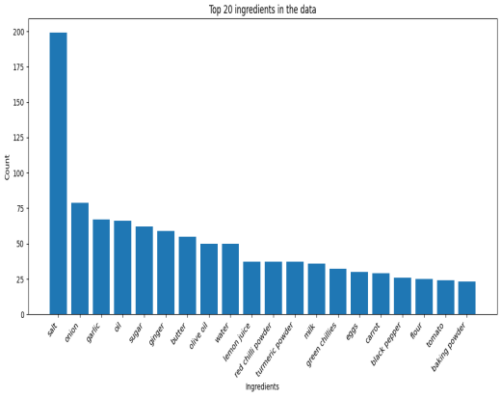
As ingredients data has information only about the dietary preferences and descriptions (list of ingredients) of food items, it can be used in Rule and Content based Recommender Systems.

The ratings data has user-item interaction which can be considered for Model based and Memory based Collaborative Filtering.

Exploratory Data Analysis (EDA)

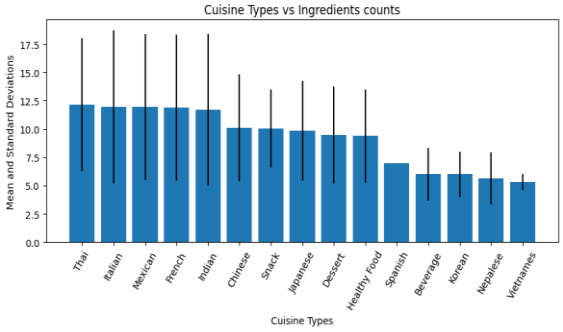
In this project, we dig deeper into the EDA part by analyzing both individual datasets and merged data. After pre-processing, we started with basic visuals to understand data distribution and identify any imbalances in the data followed by some complex visuals in finding patterns like common words, co-occurrences and rating variations in the data.

Top 20 ingredient counts



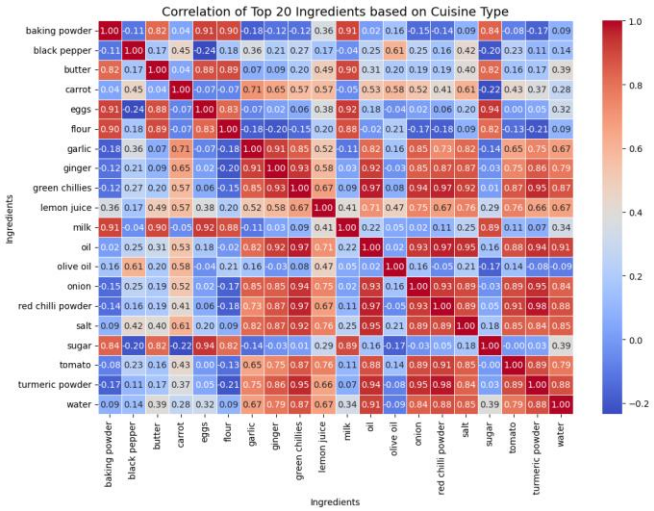
In the above bar plot, we visualized the most common items with their respective counts. The distribution in the bar plot highlights the imbalances in the presence of ingredients in the data showing the most common items like salt being the most common followed by onion, garlic, oil and sugar. It is the most important plot resulted to explore on different type of content-based recommender systems particularly Tf-Idf based recommender systems to ensure the most common items get penalized to highlight core ingredients of a food item.

Ingredient count variations of different cuisine types



This chart shows the average number of ingredients used by each cuisine, along with their standard deviations. Thai cuisine appears to use the highest number of ingredients on average, indicating more complex recipes. Vietnamese cuisine has a lower average, suggesting simpler dishes. This graph helped us in understanding the importance of Ingredients count feature that we engineered based on item counts.

Heatmap – Correlation of common ingredients

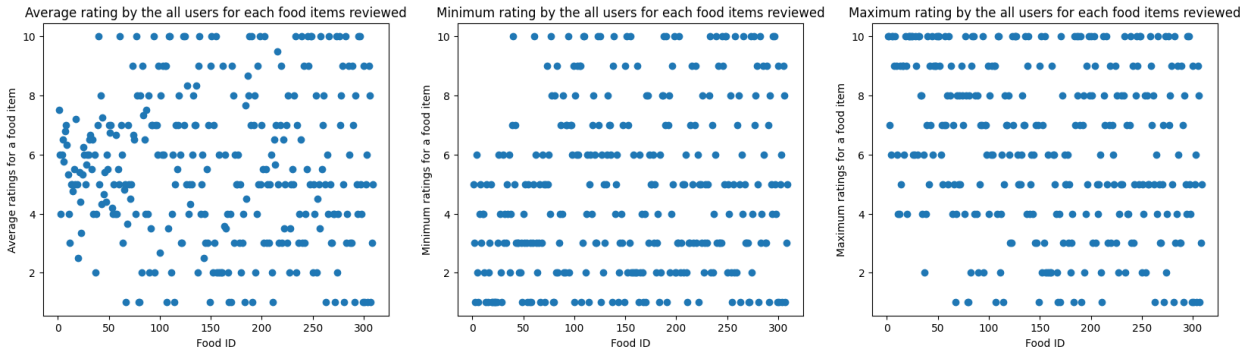


This heatmap reveals how often certain ingredients appear together across different cuisines. Strong positive correlations (in warm colors) mean those ingredients tend to be used together frequently. Blue or cooler shades indicate weaker or negative correlations, showing they rarely appear in the same recipes.

Word Clouds (Ingredients by Cuisine)

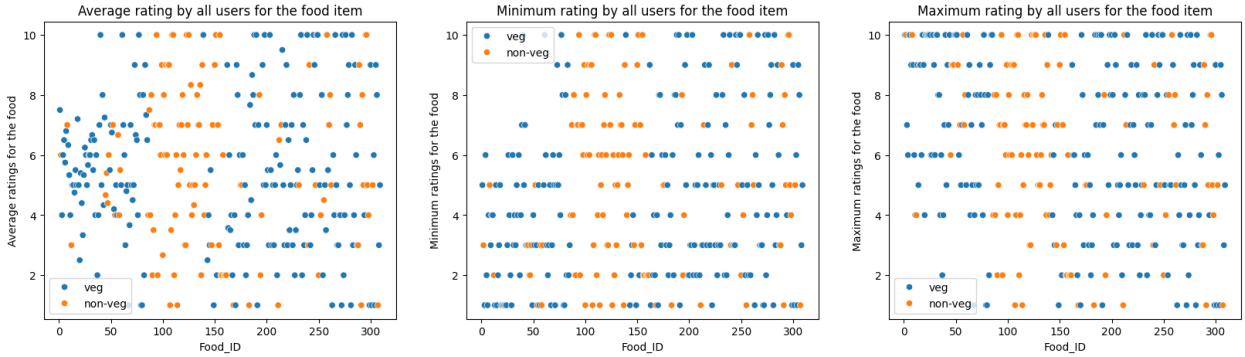
Each word cloud highlights the most common ingredients for a particular cuisine. Basic staples like salt, onion, and garlic are prominent in many cuisines, showing their universal appeal. Certain ingredients, such as soy sauce or sesame, stand out in Asian cuisines and reflect specific flavor profiles.

Scatter Plots (Average, Minimum, and Maximum Ratings vs Food ID)



These plots show how each dish is rated by users, from the lowest to the highest score. Even though there is no clear pattern this helped us in identifying there are not outliers or strong biases included in the data.

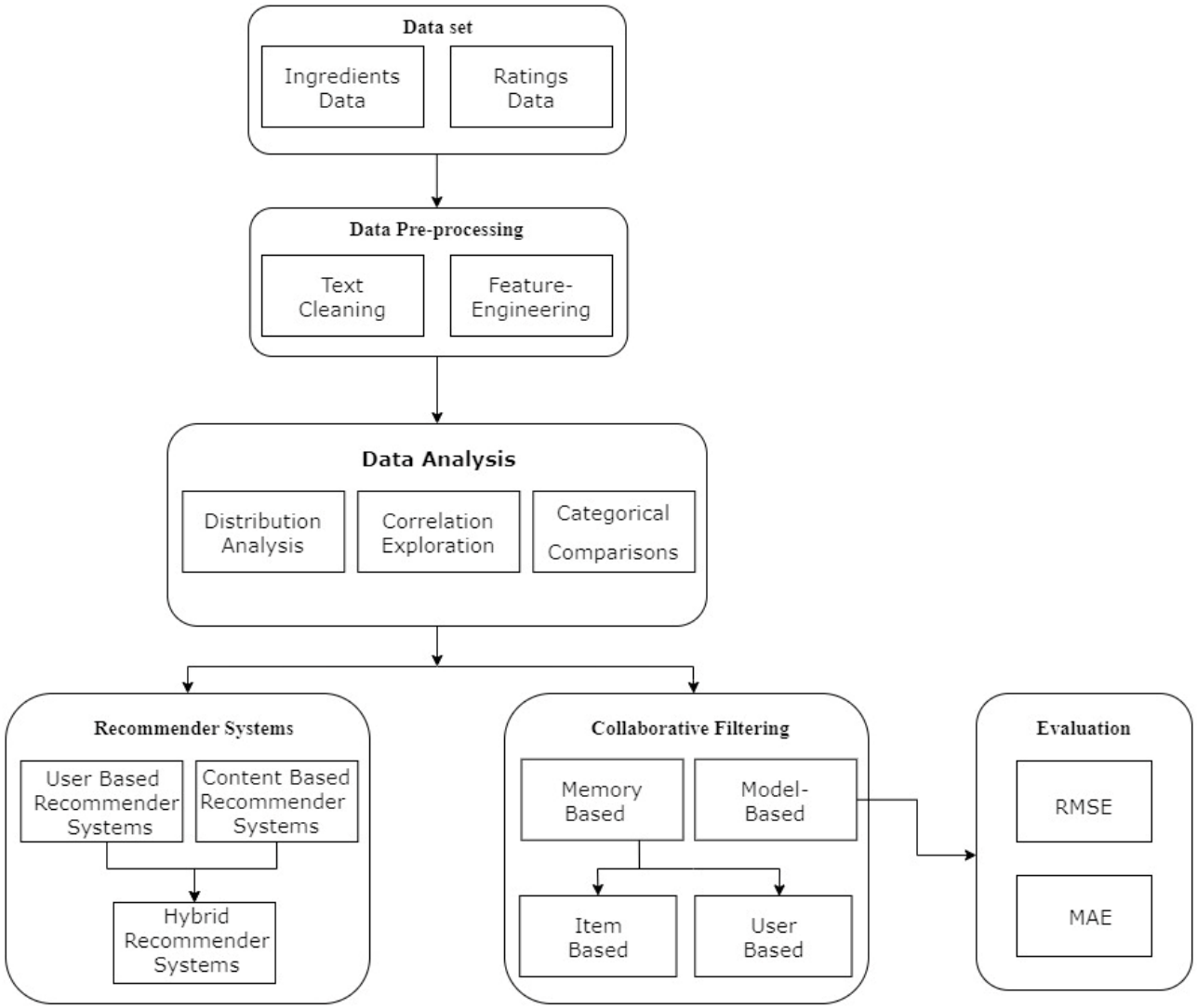
Scatter Plots (Veg vs Non-Veg Ratings)



These color-coded points compare how vegetarian and non-vegetarian dishes are rated. We can spot whether one category tends to be rated higher overall or if they overlap significantly. The variety in both categories suggests that personal taste plays a big role, regardless of the dish type. This plot also suggests that there are not outliers or strong biases included in the data.

Process Flow:

The following are very high-level steps implemented in the project. It includes end-to-end life cycle.



Hybrid Recommender Systems (Rule based + Content based)

Initially, we started with rule-based recommender systems, to recommend the users based on hard rules defined based on some dietary features in the data and user preferences. In this part, we did not include any information about the ingredients used or user-item interactions to recommend food items. It was completely based on the dietary features like Veg/ Non-Veg, Cuisine Types and Count of ingredients.

Then, we moved to content-based recommender systems to include the information of list of food ingredients used to make a food item. Here, we created a custom metric similar to Jaccard Similarity and Dice coefficient, but we did not penalize the recommendations for having additional ingredients. It is primarily based on how many of the required ingredients are available.

Similarity metric could be expressed as follows:

A = Count of Available ingredients with a user that are also required to make a recipe (Available ∩ Required)

R = Required ingredients to make a recipe (Required)

Similarity score = A/R

In the content-based recommender systems, the top 5 items are recommended based on the Similarity scores. Here, we calculated the similarity scores between the user inputs and the entire data which was then sorted based on the score and filtered top 5 to make recommendations.

Finally, we have developed a hybrid recommendation system that consists of both capabilities (Rule & Content based with a custom metric). In this method, we initially filter the entire data based on the hard rules on the dietary features and then content based on the ingredients list.

Overall, hybrid recommender system is making decent recommendations considering the choice of design and the evaluation metrics. It is stricter in the user preferences, and it is also little lenient in the ingredients match scores which shows the balance of emphasizing user preferences and trying to put as many recommendations as possible even if the ingredients list is not completely matches.

It also has limitations which include some of the vice versa of the above advantages. One of the most important limitations is that it cannot overcome the ingredient imbalance issue which we handled with Content based TF-IDF approach. Another limitation is the evaluation of the system. Even though we have similarity metrics to make recommendations, it is hard to use any evaluation metrics as there is no training in the hybrid recommendations systems that we developed.

Content Based Recommender Systems

Content-based recommender systems suggest recipes based on a user's input ingredients and dietary preferences. In this project, we implemented a TF-IDF vectorization approach combined with cosine similarity to identify and rank the most relevant recipes based on their ingredient lists.

The dataset consists of recipes with ingredients in textual form. To make this data useful for recommendations, we applied TF-IDF vectorization to convert ingredient lists into numerical feature vectors. Stopwords were removed to enhance accuracy, ensuring that only relevant ingredients influenced recommendations.

When a user inputs ingredients, the system transforms them into a TF-IDF vector and compares it to recipes in the dataset using cosine similarity. Recipes with the highest similarity scores appear at the top, ensuring the best matches.

To refine recommendations, we added filters for cuisine type (e.g., Italian, Chinese, Indian) and dietary preferences (Veg/Non-Veg). Only recipes meeting these criteria are included in the final results. The system returns the top N most similar recipes (default: 8), providing details like recipe name, cuisine, ingredient list, and similarity score to help users decide what to cook.

This system offers personalized recommendations, aligning suggestions with available ingredients. The TF-IDF and cosine similarity approach ensures fast and efficient matching. Additionally, the model is scalable, handling large and diverse recipe datasets.

Despite its effectiveness, some limitations exist. TF-IDF treats all ingredients equally, even though some (like primary proteins or spices) are more critical. The cold start problem affects new recipes with limited overlap in ingredients. Additionally, ingredient substitutions (e.g., "tomato sauce" vs. "tomato paste") are not accounted for, which may impact similarity calculations.

Collaborative Filtering - Model-Based User-Based

Model based collaborative filtering algorithms utilize algorithms to predict users' preference on the basis of their past behavior and are an ideal solution to the sparsity and scalability problem that is very common in traditional collaborative filtering.

Here in this project, we first preprocess our data strictly by validating our datasets by casting 'User_ID' and 'Food_ID' from float to integer and by deleting all the rows with missing values. We did that so that we are able to preserve the accuracy of our further analysis and modeling. In our attempts to properly test our model and make sure that it will be capable of generalizing to new, unseen data, we split our data into separate training and test sets. This was necessary to the point where we would be able to estimate how our model would behave in real conditions, simulating the way that it would be running in a living environment.

Whereas we experimented with several methods for missing value handling, such as imputations by means or zeros for the ratings, they introduced bias without or with insignificant accuracy improvement. We thus leveraged the natural feature of SVD to handle missing data. Our use of SVD experimentation was spurred by a necessity to solve sparsity and our data being of too high dimensionality. Our optimization of our model came through parameterizing how many latent factors captured at least 90% variance of the data without introducing any redundant additional complexity.

Throughout the project, we kept making incessant changes to our approaches in response to test results in a relentless quest to optimize the efficiency and effectiveness of our recommendation system. Through the iterations, we obtained a high-performance and high-efficiency recommender system that could be executed with big complex data sets.

The limitation of this current model is that it experiences the cold start problem to which is general for model based collaborative system, that means making no meaningful predictions for new users or products about which the ratings history does not exists. It is highly sensitive to the selection of hyperparameters, i.e., latent factor size, and they need to be properly tuned so that optimal performance is achieved. Also, the latent factors output obtained from the matrix factorization are not user-friendly, and for end-users as well as system administrators, it is hard to comprehend the rationale behind why particular recommendations should be made. This complexity can be utilized in order to conceal the decision-making process, as well as limit user trust and system transparency.

Collaborative Filtering – Memory Based

Collaborative filtering (CF) which is memory-based is a common recommendation framework used to produce food recommendations by evaluating how users differ in their preferences as well as evaluating similarities among the food item itself. Memory-based CF typically provides food recommendations through one or two standard methodologies-namely user-based collaborative filtering, or item-based collaborative filtering.

- **User-Based Collaborative Filtering (UBCF)**: This approach establishes ties between users with shared preferences regarding cuisine. Our model uses cosine similarity on historical food ratings to uncover shared users. Based on the similar users, it predicts the user's rating for the unrated food item by averaging the ratings of similar users.
- **Item-Based Collaborative Filtering (IBCF)**: This approach looks at item similarity according to user ratings and recommends items that are like ones that a user has rated highly. It is less volatile than user-item methods since the relationship between items remains relatively stable over time compared to user preferences.

First, we Ensured data consistency by handling missing values and duplicate records After that, converting user and food identifiers into a structured format we created a matrix where users are represented as rows, food items as columns, and ratings as values. Later we normalized the ratings using Min-Max scaling to bring them to a common scale. For item-based CF, we transposed the user-item matrix and calculated cosine similarity between items. For user-based CF, we computed cosine similarity between users directly and then computed weighted scores for unrated items based on user or item similarity. At last, we selected the top N most similar or highly rated food items as recommendations.

While memory-based CF has proven to be effective, there remain some challenges, such as data sparsity and cold-starts for new users and items. Improvements likely include hybrid solutions that combine content-based filtering or a deep learning approach for personalized accuracy improvements.

Collaborative filtering that is based on memory is a proven, intuitive process for personalized food recommendations. It works by using someone's preferences about food and item similarities to help others identify new food and types that may align with their likes.

Evaluation

While we tested our model-based collaborative filtering system, we implemented two of the most significant metrics: Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). RMSE is a measure of the size of

the prediction errors that is the square root of the average of squared differences of actual versus predicted ratings. MAE is an average absolute error between actual ratings and predicted ratings providing a simple representation of average error.

We select RMSE and MAE since they are typical in measurement of accuracy in recommendation system to check how good the prediction of the system matches the actual ratings by the users. These are also pragmatic since they can measure error in units of ratings, thus it is easy to interpret model performance.

MAE and RMSE were computed on training as well as test data. This prevented our model from merely fitting the data that it was being trained with and performing well on new, unseen data. This prevents overfitting, where a model performs well on training data but badly on test data.

Model based collaborative filtering (User)	Train Data	Test Data
RMSE	0.156	0.257
MAE	0.030	0.042

Test results show that the performance of our model was adequate for training purposes as well as testing purposes. Training RMSE was 0.1563, meaning that the predicted ratings produced by the model were highly close to true user ratings on average. Training MAE was 0.0298, meaning that absolute difference between actual and predicted ratings averaged close to zero, suggesting effective learning of data. The RMSE on the test data was 0.2566, only slightly larger than the training RMSE, so the model was only slightly larger in its errors on new data but extremely accurate still. The test MAE was also 0.0419, which, while slightly larger than the training MAE, indicates that the model generalizes extremely well to new data. These results inform us that our model is robust in making good recommendations without overfitting.

Conclusion

Overall, EDA guided the choices of recommender systems to be chosen in the entire pipeline. In the entire project, every recommendation system has a unique ability to give recommendations as it is designed in a different approach. We experimented with different combinations to compare and identify the benefits and challenges of each recommender system. Evaluation metrics like cosine similarity and other metrics are used to make recommendations. MAE and RMSE are used to evaluate the model-based collaborative filtering where we notice very low errors and no overfitting highlighting the performance of recommender system.

We would like to conclude the project with some of the limitations of the implemented pipeline and future work left in the project:

Limitations

- Dataset has fewer than 400 food items which is very low compared to real-life food items.
- Data includes very few dimensions which does not include some of the major features like Cook time, Preparation time, Cost of Ingredients/ Food items.
- Rule, Content-based, and Hybrid recommender systems completely rely on ingredients. If user makes grammatical errors in the input ingredient list, recommendation systems either do not give any response or end up giving inaccurate recommendations.
- Collaborative Filtering completely relies on the past preferences which might struggle with new user and items.

Future Scope

- Integration of Advanced LLMs to identify the actual ingredient names from the user inputs by handling grammatical errors.
- Integrating the recommender systems to a grocery receipt text extraction pipeline can provide a user-friendly platform for ingredient/ food recommendations
- Explore and Research the existing literature review on the problems encountered in the process of building pipeline like Evaluation metrics for Content based Recommender Systems, Optimal process flow for Collaborative filtering etc.,

Individual Contributions

1. Naveen:

In this project, I contributed to various aspects of the project that include creating the project outline, developing scripts for **EDA** and **Hybrid Recommender Systems**, writing report, making presentation and managing the scheduling of meetings by coordinating with other team members. I enjoyed every aspect of the project working with all the individuals involved in the project, which includes brainstorming sessions, voluntarily presenting in the class. Everyone in the team contributed their share of work to the entire project.

2. Avinash:

In this project, I assisted in the development of a collaborative filtering system for food recommendations on the basis of memories, through both user-based and item-based filtering. I was responsible for cleaning and preprocessing the datasets, normalizing a user-item matrix, and calculating cosine similarities for both item-based and user-based recommendations. Moreover, I really got involved in writing the project report, the presentation, and organizing our meeting schedules by working closely with my teammates. I genuinely loved every part of the project, especially It was such a fulfilling experience to work with such a committed group and its great we got an opportunity to present voluntarily in class

3. Shashank:

In this project, my primary contribution was the development and implementation of the **Content-Based** Recommender System and drawing the architecture diagram discussing with everyone. Other than that, I contributed through regular meetings, ideas on what and how to perform analyses. Through these contributions, I gained helped shape the foundation of the **ingredient-based recommendation system**, enabling users to discover relevant recipes based on the ingredients they have.

4. Yashkumar Rajubhai Prajapati:

I contributed some of the important things to the project to enhance stability and quality for our recommendation system. I conducted **exploratory data analysis (EDA)** and good preprocessing had been performed to be able to do proper model training. I tried to employ the **model-based user-based collaborative** filtering framework with **Singular Value Decomposition** (SVD) and **evaluated** system testing with **RMSE** and **MAE**. I also contributed to reporting and documentation so that the model, process, and outcome were communicated well. I also assisted Shashank in simplifying the system flowchart further, managing tasks better, and making the entire project process easy to understand.