Personalized Recipe Recommendation System

Repository Link: https://github.com/sbogh/cse258-a2

Abstract

This paper presents a personalized recipe recommendation system based on user-recipe interaction data and individual preferences. We propose two hybrid models: (1) one that combines content-based filtering with collaborative filtering through an optimized Singular Value Decomposition (SVD) approach and (2) another that integrates content-based filtering with a logistic regression model. We evaluate these models against several baselines, including popularity-based, Jaccard and cosine similarity, and standalone collaborative filtering models. Our analysis reveals that the SVD model's performance was constrained by the sparsity of user interaction data, limiting the effectiveness of its collaborative filtering component. As a result, content-based filtering became the primary driver of recommendations. In contrast, the hybrid content-based filtering and logistic regression model outperformed baselines, achieving superior accuracy and precision. The results highlight the potential of hybrid models in capturing user preferences, with opportunities for future improvements in handling sparse datasets and enhancing personalization.

1 Exploratory Data Analysis

The selected dataset is from Food.com, containing eighteen years of user interactions including recipes, ratings, and reviews. It contains 231,637 recipes and 1,132,367 reviews from 226,570 users. Recipe parameters such as ingredients, number of ingredients, number of steps, and tags are provided within the dataset.

During the initial discovery phase, we uncovered biases within the user ratings in the dataset. Given the popularity metric, we sought to determine the divide between popular and unpopular recipes. For this, a threshold of 100 interactions was set to classify the dataset between the two groups. Of the unique recipes, only 769 were considered popular compared to 230,868 unpopular recipes. This means only 0.332% of recipes are considered popular under this metric. Both of these analyses informed us of the sparsity of the dataset.

Additionally, the ratings were heavily left-skewed, as seen in Figure 1, so we considered mitigating this skew through a balanced loss function or sampling the dataset.

The average recipe time was 9,398.546 minutes or approximately six days. This was of interest, given the extremely high value along with its substantial standard deviation. We investigated further and found that a subset of recipes had times that exceeded 24 hours. This distribution can be seen in Figure 2. Since there were no interactions with these recipes, it was deemed unnecessary to remove these outliers from the dataset, as they would have no influence on a collaborative filtering model.

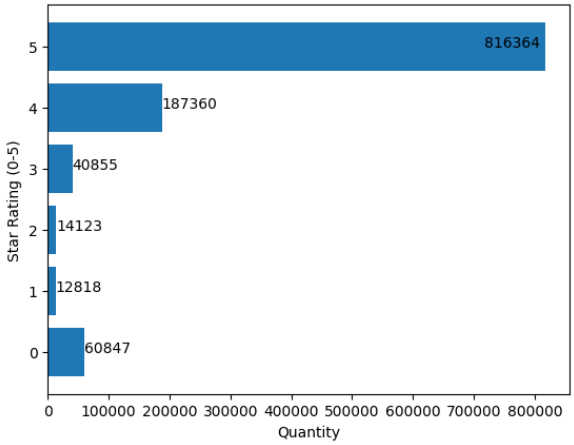


Figure 1: Interactions Rating Distribution

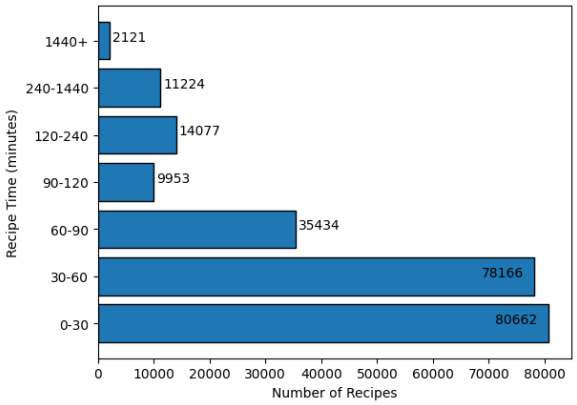
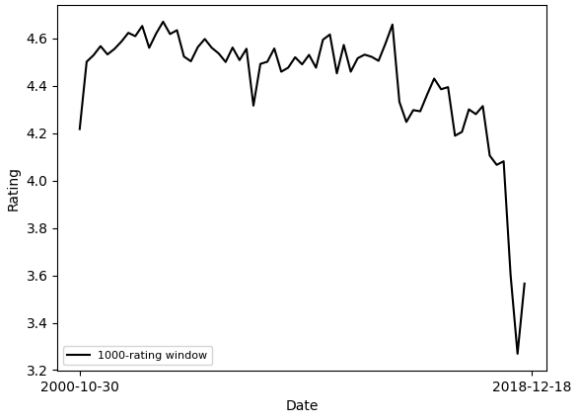


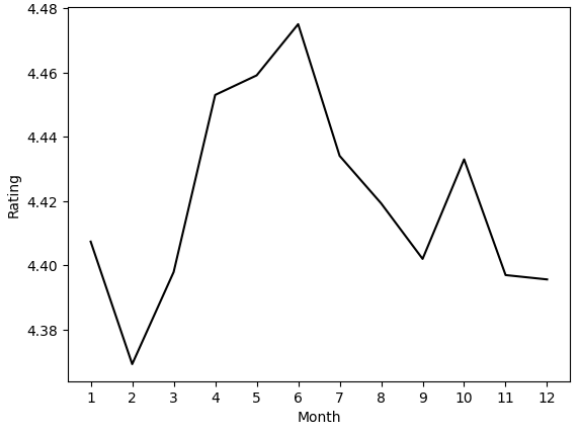
Figure 2: Recipe Time Distribution

Next, we dove into seasonal trends and how they may affect the data, as seen in Figures 3. This would inform whether temporal information needed to be a focus of the model. Figure

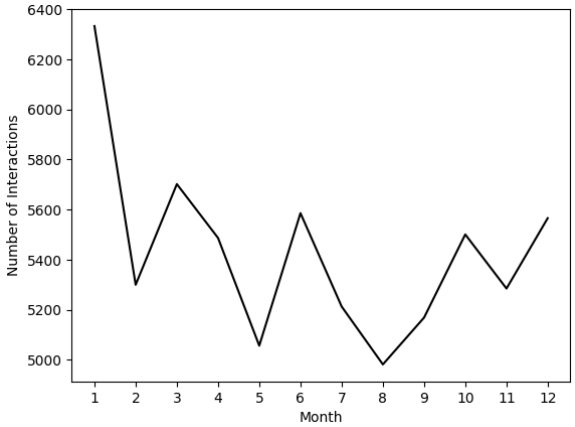
3a displays the ratings over time given a 1000-rating sliding window. There is a significant drop near 2017, which is potentially related to Food.com temporarily changing their name to Genius Kitchen from 2017 to 2019 (Spangler, 2017). This may have had an effect on perception of the website, causing users to expect higher quality recipes and decreasing the ratings. Figure 3b displays rating fluctuations over seasons. As seen from the small range of the ratings axis, there is not a significant fluctuation over time. Finally, Figure 3c shows the seasonal trend of the number of interactions. There is a substantial spike in January, possibly due to reviews of holiday recipes



(a) Ratings over time (sliding window)

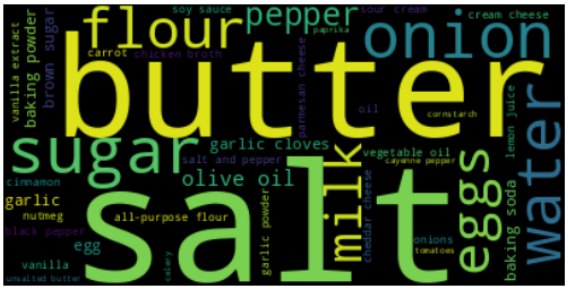


(b) Seasonal Trend - Ratings

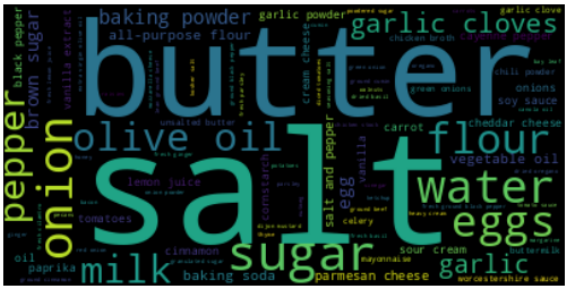


(c) Seasonal Trend - Interactions

Figure 3: Temporal Trends



(a) Winter



(b) Spring

(c) Summer

(d) Fall

Figure 4: Ingredient Word Clouds by Season

alongside a potential seasonal marketing push by Food.com.

We analyzed the most popular ingredients for each season. As seen in Figure 4, the top five ingredients were the same, regardless of season: salt, butter, sugar, onion, and water. Because of this, we decided to implement TF-IDF vectorization to downweight common terms.

Finally, we observed a mismatch between the ratings and review text for certain entries. Specifically, some reviews with a rating of 0 contained text expressing overly positive sentiments. To investigate this further, we conducted sentiment polarity analysis on the review text for entries with a rating of 0 and reassigned ratings based on the sentiment scores. A comparison of the old and new ratings can be seen in Figure 5.

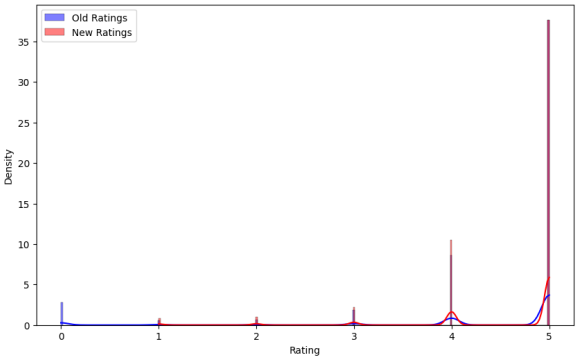


Figure 5: New vs Old Rating Distributions

2 Predictive Task

2.1 Identifying a Predictive Task

We analyzed our dataset to identify a meaningful predictive task that aligns with user-centered goals. This led us to focus on creating a recommender system that suggests recipes tailored to 3 key points: a list of ingredients users have on hand, any optional tags or keywords such as 'easy', 'vegan', etc., and the likelihood of a user engaging with or liking a given recipe. This task reflects a common real-world scenario where users seek personalized recipe recommendations that cater to their dietary preferences or time constraints.

2.2 Our Proposed Model

In developing our recipe recommender system, we aimed to create a hybrid model that combines the strengths of content-based and collaborative filtering to help find recipes that match a user's available ingredients and align with their interactions or historical ratings of other recipes.

2.3 Selected Baseline Models and Features

To evaluate the model, we identified several meaningful baselines for comparison, including a popularity-based model, Jaccard, and cosine-based models, and a collaborative filtering model. Some key features driving the model include recipe ingredients, optional identifying tags, user ratings, popularity scores, similarity scores, and user interactions. User ratings, ingredients and tags were provided as data in columns in the provided interactions and recipes dataset files. Other features such as popularity scores, similarity scores, and user interactions had to be calculated from these data columns. For example, popularity scores were calculated using the number of interactions for a recipe.

2.4 Assessing Model Prediction Validity

The primary evaluation metrics for these baseline models were accuracy and mean-squared error (MSE). For our content-based filtering model, we used qualitative validation methods, such as checking if ingredients of the returned top recipes matched the input. For the hybrid models, we aimed to utilize other validation metrics such as precision, recall and F1-scores. The predictions generated by the model are treated as a binary classification task: relevant or irrelevant to the user. For the content based-SVD hybrid model, the model generates a ranked list of recipe recommendations for each given user. A recipe is considered relevant if it matches the user's preferences and irrelevant otherwise. Precision measures the proportion of recommended recipes that are relevant, while recall evaluates how many of the user's relevant recipes were successfully recommended. The F1-score, otherwise known as the harmonic mean of precision and recall, provides a balanced evaluation metric, especially when relevance classes are imbalanced. We compared the model's predictions against our dataset of user interactions or preferences to obtain these metrics.

3 Models

3.1 Content-Based Filtering Model

We developed a content-based filtering model using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization to represent recipes and

their attributes in the Food.com dataset. TF-IDF highlights the most important terms within a recipe by emphasizing contextually significant words while downweighting common terms that appear across recipes. This approach reduces the recipe content to a meaningful set of features that are used to calculate similarities.

For our implementation, we utilized the ingredients and tags features from the recipes dataset, transforming them into TF-IDF vectors. These vectors allowed us to compute cosine similarity between a user query, which contains ingredients and tags, and the recipes in the dataset. The model ranks recipes based on their similarity scores and provides recommendations accordingly. To improve usability, we incorporated optional filters for caloric content and cooking time, since time constraints and nutritional goals are often important considerations when searching for recipes as seen in Figure 6.

While this formed the foundation of our recommendation system, model validation posed challenges, since the model predicts textual outputs, such as the name of the recommended recipe and the steps required, rather than a numerical target. We initially considered incorporating user ratings for validation; however, the ratings distribution was highly left-skewed, suggesting that a rating-based prediction model would not perform reliably. Thus, we focused on building upon the content-based filtering approach to validate it.

Example Input:

```
# Specify available ingredients, search tags
available_ingredients = ['chocolate', 'peanut
butter', 'caramel']
search_tags = ['gluten free', 'easy']

# Specify optional filters
max_calories = None
max_time = 30
```

Example Output:

```
Recipe Name: flourless peanut butter cookies
Ingredients: peanut butter  white sugar  egg
baking soda  vanilla chocolate chips
Tags: 30-minutes-or-less  time-to-make
course  main-ingredient  preparation
for-large-groups  snacks  dietary
gluten-free
Calories: 98.9
Time to Cook: 20 minutes
Steps: preheat oven 350, mix peanut butter
with sugar, beat egg and add baking soda
and vanilla to it, add eggs to peanut butter,
drop teaspoon size cookies on ungreased
baking sheet, bake 10-12 min, let cool at
least 3 min
```

Figure 6: Example input and top recommendation based on Content-Based Filtering Model

3.2 Logistic Regression Model

We selected the logistic regression model to predict user interaction with new recipes. Logistic regression is particularly well-suited for modeling probabilities, making it an ideal choice for predicting whether a user will interact with a recipe (`label = 1`) or not (`label = 0`).

3.2.1 Model Comparison

Prior to selecting the logistic regression model, we experimented with baseline models with simpler implementations: popularity thresholds, Jaccard similarity, and cosine similarity. While these models provided valuable insights into user-recipe interactions, we found the predictive accuracy to be insufficient compared to logistic regression. For example, while the accuracy on our test set of the popularity threshold was 74%, the Jaccard and cosine similarity approaches failed to outperform the baseline accuracy of predicting all labels as 0, which was 50%.

The models mentioned previously failed to capture nuanced interactions between users and recipes, and were thus limited in their predictive power. Therefore, we selected the more robust logistic regression model to capture interactions along with a diverse range of features, which led to a significantly higher accuracy (83%).

3.2.2 Our Model

To achieve an improved accuracy on our test data, the features implemented in the logistic regression model represent various user, recipe, and interaction characteristics. These features were carefully designed to capture the nuances of user-recipe dynamics and improve the prediction performance. Below is a detailed description of each feature:

- **Popularity Score:** The normalized measure of how frequently a recipe has been interacted with by users.
- **User Activity:** A measure of how active a user is based on their total number of interactions.
- **Average Jaccard Similarity Scores:** The average Jaccard similarity between the target

recipe and the user's top 2, 4, and 6 previously interacted recipes. The formula is:

$$\text{Jaccard Similarity} = \frac{|U_r \cap U_s|}{|U_r \cup U_s|}$$

where U_r and U_s are the sets of users who interacted with recipes r and s, respectively.

- **Average Cosine Similarity Scores:** The average cosine similarity between the target recipe and the user's top 2, 4, and 6 previously interacted recipes. The formula is:

$$\text{Cosine Similarity} = \frac{|U_r \cap U_s|}{\sqrt{|U_r|}\sqrt{|U_s|}}$$

- **Weighted Similarity:** A combined metric that blends the average top-K Jaccard and cosine similarities.

$$S_{\text{weighted}} = 0.5 \cdot S_{\text{Jaccard}} + 0.5 \cdot S_{\text{Cosine}}$$

3.2.3 Challenges

To mitigate overfitting, the model used L2 regularization. Cross-validation was employed to ensure generalizability. We experimented with class weights, but the change did not impact our results, suggesting that the features adequately addressed imbalance.

3.3 Logistic Regression on Content-Based Filtering Results

The hybrid model combines content-based filtering with logistic regression to enhance prediction accuracy. Content-based filtering identifies a subset of relevant recipes by matching user preferences to recipe features using TF-IDF vectorization and cosine similarity. Logistic regression then builds on this subset, predicting interactions by modeling relationships between user engagement patterns and recipe characteristics.

This approach improves upon standalone similarity models by leveraging the logistic regression model's ability to integrate diverse features and capture nuanced relationships. The model is trained on filtered datasets, ensuring that logistic regression is applied to recipes most relevant to user preferences. The use of the logistic regression on the content-based filtering model results led to an accuracy of 95%, depending on the filter requirements, which offered a substantial improvement on the logistic regression model alone, which had an accuracy of 83%.

Overfitting risks are managed through L2 regularization, and the scalability of combining filtering with regression allows for handling larger datasets. While early attempts using only similarity metrics struggled to outperform baseline predictions, the integration of logistic regression significantly improved accuracy. The hybrid model's strength lies in uniting personalized filtering with a statistical model that generalizes well, achieving high predictive performance by capturing both general trends and user-specific interactions.

3.4 Collaborative Based Filtering

3.4.1 SVD Model Selection

Our goal is to create a hybrid content-based and collaborative filtering model, so we explored a baseline collaborative-filtering model. We chose the singular value decomposition (SVD) model, a matrix factorization technique that decomposes the user-item interaction matrix (user ratings of recipes) into three smaller matrices. These matrices capture the latent features of users and items that explain the observed interactions. The SVD model minimizes the error between the observed and predicted ratings generated from these matrices. Under SVD, a given matrix A of size m x n is expressed as:

$$A = U\Sigma V^T$$

where U and V are left and right singular values of A and Σ is a diagonal matrix of eigenvectors of AA^T (McAuley, 2022).

3.4.2 Implementation

After training our SVD model, we tested it on the validation set to predict ratings and evaluate its performance using mean squared error (MSE), which was 1.48. To improve the MSE, we then tuned the model parameters to optimize the model's performance on unseen data and avoid over or underfitting.

3.4.3 Model Optimization

We used sklearn's RandomizedSearchCV implementation to find the best combination of hyperparameters based on our predefined search space. After hyperparameter optimization, the MSE improved slightly to 1.46 as seen in Table 1.

n_factors	n_epochs	lr_all	reg_all	MSE
100	20	0.005	0.02	1.4906
50	25	0.005	0.1	1.4631
25	20	0.005	0.1	1.4778
75	25	0.005	0.1	1.4802

Table 1: Ablation Study with Varying Hyperparameters

3.4.4 Model Issues

One of the issues considered for this implementation was scalability. The model training time scaled with the dataset size, and as such by increasing the number of interactions, the process would become computationally expensive. Due to the existing skew in the dataset, the model was not effective in predicting recipe ratings, as seen from the MSE results discussed.

3.5 Hybrid Recommendation System

3.5.1 Hybrid SVD and Content-Based Filtering Model

Despite the SVD baseline model not performing as well as we hoped, we wanted to explore creating a hybrid recommendation system that leveraged both content-based filtering and collaborative filtering through the optimized singular value decomposition (SVD) model. This hybrid approach aimed to capitalize on the strengths of both models to improve recommendation accuracy and overcome the limitations of individual methods.

3.5.2 Implementation

The content-based filtering component begins by using available ingredients and identifying tags as input to recommend the top 5 recipes that match these criteria. The similarity between the ingredients and tags is measured using cosine similarity, generating a list of recommended recipes along with their similarity scores. Next, we apply the optimized SVD model to this list of recipes, where the user's ID is provided as input to predict the user's rating for each recipe. A weighted score is then computed by combining the similarity score and predicted rating. Initially, both factors were given equal weight. The recipes are then ranked based on the final weighted scores, with the top recommendation being a recipe that not only aligns with the ingredient and other optional requirements, but one that is also likely to receive a high rating from the user.

3.5.3 Results Validation Methodology

Validation of this hybrid model was challenging due to the complexity of combining two different recommendation techniques. To assess its performance, we focused on quantitative validation through precision@K, comparing the predicted ratings to the actual ratings of the recommended recipes. We set a threshold of 4.0 to indicate that a user truly liked a recipe, though in hindsight, this threshold may not effectively distinguish between a "really liked" and a "moderately liked" recipe, given the skewed nature of the dataset (with most ratings above 3.0 or 4.0).

We selected a subset of user IDs and used the optimized SVD model to predict ratings for the recipes output by the content-based filtering model. For recipes that a user had not rated, the global average rating was used as a proxy. As expected, due to the rating skew, the precision@K for our hybrid model was 1.0, meaning that all top K recommendations were highly relevant or liked by the user.

3.5.4 Model Issues

The SVD model, as applied to our hybrid recommendation system, encountered challenges in generating personalized predictions for users, particularly for the top 5 recommended recipes. For users with limited or no prior ratings of similar items, the model struggled to produce accurate predictions and instead resorted to predicting a value close to the global average. This led to the same predicted rating for all top recipes within a given user—while the rating was consistent across these items, it varied between users. For example, User 1 might receive a predicted rating of 4.7 for all five recipes, while User 2 receives 4.4, but within each user's top recommendations, the ratings were uniform.

This uniformity of predictions for each user's top 5 recommendations can be attributed to the lack of sufficient user-specific interaction data. In these cases, the model relied heavily on the latent factors learned from other users and general patterns across the data, which led to similar predicted ratings across the recommended recipes for each individual user. However, the model still generated different ratings for different users, as latent factors were user-specific and reflected each user's unique preferences. This issue underlines the limitations of SVD when user interaction data is sparse, resulting in less personalized recommen-

dations. Hence, this model could be more effective in cases where user-interaction data is less sparse. For sparser datasets like ours, the content-based filtering plays a more dominant role in recommending recipes.

4 Related Literature

The dataset we selected was used in work published in EMNLP, which recommended personalized recipes based on user preferences inferred from their reviewed recipes (Majumder et al., 2019). This paper employs BERT for the predictive task to make predictions using the dataset, specifically training the model to generate recipes based on user preferences. This model architecture was based on an encoder-decoder architecture, in which the encoder has three embedding layers representing the recipe vocabulary, ingredients, and caloric level. The decoder implements a 2-layer GRU which is attended over a user's previously reviewed recipes to create bias towards user preferences.

A similar dataset is the Food Ingredients and Recipes Dataset with Images from the Epicurious website, which includes 13,500 recipes accompanied by metadata such as the dish title, ingredients, preparation instructions, and images. The images make it useful for research that combines natural language processing with computer vision, such as visual-based food recommendations or calorie estimations from images.

Another related dataset is the RecipePairs dataset, which was derived from the Food.com dataset. It consists of 6.9 million pairs of original and modified recipes. Each pair was annotated with a similarity score calculated using the longest common subsequence/union of recipe names and dietary labels (e.g., vegetarian, keto). This dataset has been used in studies on recipe transformations, where one recipe is adapted to meet dietary restrictions or preferences, or to evaluate how ingredient substitution impacts similarity between recipes (Li et al., 2022). The researchers in this paper utilized the Transformer Architecture to learn ingredient embeddings, predicting the likelihood of ingredient substitutions and generating corresponding recipe steps. Despite their challenges in recipe substitution tasks, their model was able to generate human-like recipes. Our work focuses on accommodating dietary restrictions through search tags such as "gluten free," and is an additional parameter in the recommendation process to accommodate users.

The paper "Intelligent Food Planning: Personalized Recipe Recommendation" recommendation approach is similar to ours, employing content-based filtering, collaborative filtering, and hybrid models using Pearson similarity based on recipes and food items (Freyne and Berkovsky, 2010). This paper uses a dataset from CSIRO Total Wellbeing Diet Book, primarily to promote healthier eating habits. The researchers concluded that a hybrid approach, based on recipe and user interactions, had the lowest MAE loss. This coincides with our findings as well since the hybrid model performed the best in our experimentation.

5 Results

5.1 Logistic Regression and Content-Based Filtering Conclusions

Model	Accuracy	Precision	Recall	F1
Baseline Model	0.5	0.0	0.0	0.0
Jaccard Similarity	0.5	0.0	0.0	0.0
Cosine Similarity	0.5	0.0	0.0	0.0
Popularity Threshold	0.74	0.76	0.69	0.73
Logistic Regression	0.83	0.86	0.78	0.82
Logistic Regression on Content-Based Filtering	0.95	1.0	0.9	0.95

Table 2: Model Evaluation Metrics

The results of these models highlight the significant improvement achieved through the hybrid model that combines content-based filtering with logistic regression. The **baseline** approach of predicting all interactions as 0 yielded an accuracy of 50%, with precision, recall, and F1 scores of 0.

This yielded the same results as other similarity-based methods, **Jaccard** and **cosine** similarity, which also plateaued at 50%, with the same precision, recall, and F1 metrics of 0, as these models predicted every interaction as 0 when optimized. A simple **popularity threshold** of 0.7 improved accuracy to 74%, with precision of 0.76, recall of 0.69, and F1 score of 0.73, indicating that global popularity was a moderately effective predictor.

Logistic regression applied to the entire dataset, incorporating features like user activity, recipe popularity, and similarity metrics, achieved an accuracy of 83%, with precision of 0.86, recall of 0.78, and F1 score of 0.82. It demonstrated balanced performance across other metrics and was better at minimizing false positives than false negatives. However, the hybrid model, which applied **logistic regression to content-filtered results**, significantly outperformed the other models with an accuracy of $\sim$95% depending on filtering parameters, along with a precision of 0.95, recall of 1.0, and F1 score of 0.95. The results indicate near-perfect precision, reflecting the model's ability to minimize false positives, while recall improved substantially compared to standalone logistic regression, suggesting that the hybrid model improves the ability to minimize false negatives.

The hybrid model's superior performance can be attributed to its two-stage process. Content-based filtering narrowed the candidate recipes, reducing noise, while logistic regression effectively modeled the refined interactions. This combination captured both user preferences and general trends, leading to substantial gains in precision, recall, and overall accuracy.

5.2 SVD and Content-Based Filtering Conclusions

The solo SVD model and the hybrid SVD/content-based filtering model represent two complementary approaches to recipe recommendation. The solo SVD model initially performed with an MSE of 1.48, which improved to 1.46 after hyperparameter optimization. Despite these adjustments, the improvement was marginal, indicating that the model struggled to capture the complexity of user-recipe interactions effectively.

The hybrid SVD/content-based filtering model attempted to address these limitations by incorporating content similarity scores derived from TF-IDF vectorization and cosine similarity. This model narrowed down the set of recommended recipes using content features such as ingredients, tags, and user preferences before applying SVD for personalized rating predictions. The final hybrid score, combining content similarity and SVD ratings equally, aimed to balance relevance and personalization.

One of the major difficulties of the hybrid model is the lack of comprehensive validation metrics, as its combined scoring mechanism (merging content-based similarity and SVD predictions) does not align directly with standard metrics like MSE or RMSE. This makes it challenging to evaluate performance quantitatively. However, the hybrid model achieved a precision@k of 1.0, meaning that for the top k recommended recipes, all were relevant to the user's preferences. This metric highlights the model's strength in prioritizing highly relevant results, but does not fully address its limitations with unseen recipes. Thus, while precision@K offers a useful measure of relevance, it does not provide a complete picture of the model's ability to generalize or perform across broader scenarios.

The primary limitation of the SVD model was its inability to make accurate predictions for recipes the user had not previously interacted with, often defaulting to average ratings. This reflects a fundamental cold-start problem in collaborative filtering. This limitation diminished the hybrid model's overall effectiveness in providing truly personalized recommendations.

In terms of feature representation, content-based filtering worked well for identifying relevant recipes based on ingredient and tag similarity. However, SVD failed to exploit these similarities effectively for personalized predictions. The lack of synergy between the two components limited the hybrid model's success.

In conclusion, the hybrid model demonstrated potential by combining relevance and personalization, but its reliance on a limited SVD implementation hindered its performance. Future work should focus on integrating more robust collaborative filtering techniques or improving content-based features to address cold-start and sparsity challenges effectively.

References

Freyne, J. and Berkovsky, S. (2010). Intelligent food planning: personalized recipe recommendation. In *Proceed-*

ings of the 15th International Conference on Intelligent User Interfaces (IUI '10), pages 321–324, New York, NY, USA. Association for Computing Machinery.

Li, S., Li, Y., Ni, J., and McAuley, J. (2022). Share: a system for hierarchical assistive recipe editing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11077–11090. Association for Computational Linguistics.

Majumder, B. P., Li, S., Ni, J., and McAuley, J. (2019). Generating personalized recipes from historical user preferences. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5976–5982. Association for Computational Linguistics.

McAuley, J. (2022). *Personalized Machine Learning*. Cambridge University Press.

Spangler, T. (2017). Scripps networks launches 'genius kitchen' online food network, folding food.com recipes into the mix. www.variety.com/2017/digital/news/scripps-networks-genius-kitchen-food-com-1202563297/.