Introduction

Our dataset contains information on recipes including variables of interest to our analysis such as number of reviews for each unique recipe, reviews (scored from 2-5), average reviews, and number of steps to complete the recipe. We were interested in investigating people's preferences for different recipes in our dataset which we used ratings and average ratings as a proxy for and how preferences relate to the number of steps to carry out a recipe. Our analysis aims to uncover if people more strongly prefer easier recipes. We use ratings (average and number of ratings) to give us insight into preferences because the more ratings a recipe has allowed us to understand relatively how many people chose that recipe and the rating itself can point to satisfaction of the recipe. In addition we will look into the number of steps in the recipes in our dataset to get an understanding if number of steps correlates to higher ratings since people may have a higher preference for easier recipes it would follow that recipes with less steps are easier and if people prefer these recipes they would have higher average ratings. Exploring which recipes in this dataset were most highly preferred and their relative difficulty is of interest because cooking is an investment of time so we hope through our analysis readers can get a better understanding of which recipes reviews previously found to be the most worthwhile and hopefully uncover highly rated recipes with relatively few steps.

Our dataset includes 234,429 rows corresponding to a single review of a recipe (thus we see duplicate entries of a single recipe which correspond to different reviews of the same recipe) and 17 columns, however for our analysis we focus on a subset of 7 of these columns. The columns of interest to our research are name(string, name of recipe), id(int, unique recipe id we can use to identify individual recipes), minutes(int, time in minutes to complete a recipe start to finish), n_steps(int, number of steps to complete a recipe start to finish), n_ingredients(int, number of ingredients needed to make the recipe), rating(int, rating given by each submission of a review of a recipe), and average rating(float, average over all ratings for a given recipe).

Name and id are used simply for identification purposes in our analysis to give results of any sort of meaning. We are interested in understanding which recipes are found to be easier to complete and rated the highest and investigating if there is a strong relationship between these two variables. To investigate which recipes are "easy" we use the number of steps, time, and number of ingredients to get an overall idea of different measures of recipe complexity. Finally, to get an idea of which recipes are most popular we use the ratings and average rating of each recipe to explore and determine which recipes received the highest rating and we use this as an indicator of which of the recipes in our dataset are the most highly preferred.

Data Cleaning and Exploratory Data Analysis

In order to clean our data set we needed to first merge the recipes and ratings datasets to combine our variables of interest into one data frame that we could manipulate for further analysis into our question of interest. In order to do this we first renamed the 'recipe id ' column in ratings to

‘id’ when we loaded the ratings dataset and then we used recipes as our left data set and ratings as our right data set and merged on ‘id’ since these columns in each respective data set contained the unique id to identify the recipes and we performed a left merge so all the rows of unique recipes in the recipes data set were preserved in the merge and where these recipes have ratings in the ratings dataset the ratings were matched with the recipe they correspond to and where recipes didn’t have a rating these values are np.nan or missing values.

Our second step in the data cleaning process was to manipulate the values in different columns to make the data easier to work with in our eda and analysis steps. We first added a column to the data frame called average rating. To do so we grouped the data by unique recipe (grouping the data by id) and then averaging across the grouped data to get average rating for each recipe we then merge this back into the larger dataframe.

Additionally, we fixed the data type of the columns. First we made the ‘submitted’ and ‘date’ columns which gives information on the date the recipe was published and date a review was submitted respectively to date time objects rather than a string this way we can more easily access specific attributes about the date. We then converted ‘tags’ ,’nutrition’, ‘steps’, and ‘ingredients’ to lists using the eval method and applying that to the entire column. The eval method just analyzes anything inside a string as if it is a function so effectively for our data since all the information in the strings were encapsulated by brackets they look like lists so the eval method converted them to lists rather than strings. Finally, we converted the rating column to be Int8 since ratings are scored from 1-5 so decreasing the space this variable takes up should make our code run faster and these values will never be more than 2 byte data so allocating more storage to this column is not necessary.

Finally, we subsetted our dataframe to only include columns of interest. By dropping columns such as ‘nutrition’ and ‘date’ we are better able to complete analysis without having the processing burden of computation being carried on columns not relevant to the question at hand.

INCLUDE HEAD OF DF INSERTED IN WEBSITE

Univariate Analysis

NUMBER OF STEPS DISTRIBUTION

The number of steps is approximately normally distributed with the mean number of steps being around 11 and our plot shows a long right tail in the distribution, this suggests that most of our recipes have steps between around 9-13.

Bivariate Analysis

Plot: Binned Number of Ingredients vs. Average Rating

This plot displays the relationship between the number of ingredients (binned) and the average rating of recipes. We observe a slight positive linear trend: as the number of ingredients increases, the average rating also tends to increase. This suggests that more complex recipes may be rated more favorably by users.

Plot: Binned Minutes vs. Average Rating

This plot explores the relationship between recipe duration (in minutes, binned) and average rating. No clear trend is evident, indicating that recipe length does not appear to significantly influence user ratings.

Interesting Aggregates:
Embed at least one grouped table or pivot table in your website and explain its significance.
Include both pivot tables

Pivot Table: Number of Steps vs. Ratings
The pivot table groups recipes by the number of steps and summarizes their average and individual ratings. There is no clear trend indicating that recipes with fewer steps consistently receive higher ratings. This suggests that the number of steps alone may not strongly influence how recipes are rated.

Pivot Table: Number of Ingredients vs. Ratings
Similarly, when recipes are grouped by the number of ingredients, we see no consistent pattern in average or individual ratings. Ratings fluctuate across ingredient bins, indicating that recipe complexity in terms of ingredients does not appear to be a strong predictor of satisfaction on its own.

Assessment of Missingness

After examining missingness in the dataset, we found that the columns rating, review, and description contain missing values.

We believe that the rating column is likely Not Missing At Random (NMAR). Ratings are submitted voluntarily by users, and the likelihood of leaving a rating may depend on unobserved factors such as the user's satisfaction or emotional response to the recipe. For example, users who had an average or negative experience might be less inclined to submit a rating. Since this missingness depends on internal, unmeasured experiences, it fits the definition of NMAR.

The review column likely follows a similar NMAR pattern. Whether a user chooses to leave a written review may depend on their strength of opinion, available time, or engagement — all of which are unobserved in our dataset. Reviews are missing at a much lower rate than rating and it is possible these two columns could be correlated for example if people who really like a recipe rate it high they also may be more inclined to leave a review and thus a relationship between missingness of reviews could be related to rating.

In contrast, the description column is provided by the recipe author, not by users submitting feedback. Missing values in description are likely due to the author choosing not to include one. This could be related to factors like how straight forward a recipe is and could be correlated with columns like number of ingredients or number of steps, if it's a super simple recipe the description may not be necessary to include.

MISSINGNESS DEPENDENCY

To explore the nature of missing values in our dataset, we focused on the review column, which has non-trivial missingness. We wanted to understand whether this missingness was related to observable data (indicating Missing At Random, or MAR), or if it might be Not Missing At Random (NMAR).

We hypothesized that users might be more inclined to leave a review when they leave a high rating, or when the recipe is relatively simple. To test these hypotheses, we conducted permutation tests to compare the distribution of rating and n_steps between rows where review is missing and rows where it is not.

1. Rating vs Review Missingness (Permutation Test) INCLUDE PLOT

We used a permutation test to compare the average rating between recipes that had a missing review and those that did not. The result was not statistically significant (p-value > 0.05), meaning we did not find strong evidence that rating and review submission are related. Therefore, we cannot conclude that review missingness is MAR with respect to rating.

2. n_steps vs Review Missingness (Permutation Test) INCLUDE PLOT

Interestingly, we also found a statistically significant relationship between the number of steps in a recipe (n_steps) and whether a review was submitted. Recipes that received reviews tended to have fewer steps on average. This suggests that recipe complexity may influence whether users leave a review, again supporting the MAR assumption.

Hypothesis Testing

Question:

We want to explore whether simpler recipes — operationalized here as having **fewer ingredients** — are rated more highly by users than more complex ones.

Hypotheses:

- **Null Hypothesis (H_0):** There is no difference in average rating between simpler recipes (those with fewer ingredients) and more complex recipes.

- **Alternative Hypothesis (H_1):** Simpler recipes have a higher average rating than more complex recipes.

Test Setup:

To assess whether recipe simplicity impacts average rating, we:

- Defined "simple" recipes as those with fewer ingredients than the median number in the dataset.

- Used the **difference in mean average rating** between simple and complex recipes as our **test statistic**.

- Performed a **permutation test** with 1,000 random shuffles of the rating data to simulate the null distribution.

- Chose a **one-sided test** since our research question asks whether easier recipes are rated *higher*.

Results:

- **Observed difference in mean ratings:** 0.0072

- **P-value:** 0.0000

Interpretation:

Since the p-value is far below the significance threshold of 0.05, we reject the null hypothesis. This result provides strong statistical evidence that recipes with fewer ingredients tend to receive slightly higher ratings than more complex recipes using our definition of complexity being represented through the number of ingredients.

Although the effect size is small (a difference of 0.0072 on the rating scale), this supports our overall research question: that easier recipes — as measured by number of ingredients — may be more favorably reviewed.

Why This Test Makes Sense:

A permutation test is a good choice here because:

- We do not assume any particular distribution of ratings.
- The test directly evaluates the likelihood of observing a difference in means under the null hypothesis.
- It is robust for comparing group differences, especially with large sample sizes like ours.

The difference in means is a natural and interpretable test statistic that aligns well with our research goal of comparing average ratings across simple vs. complex recipes.

Framing a Prediction Problem

Prediction Type:

This is a regression problem, since our goal is to predict a continuous numerical outcome— rating.

Response Variable (Target):

- rating: The numerical score (on a 1 to 5 scale) given by an individual user in their review of a recipe.

We chose rating because it is a direct, quantifiable measure of user satisfaction, and predicting it allows us to understand what makes a recipe more or less appealing to users. This connects well with our central theme of exploring how recipe characteristics (such as simplicity) relate to user preferences.

Evaluation Metric:

We will evaluate model performance using Root Mean Squared Error (RMSE). RMSE is a widely used metric for regression problems because:

- It penalizes larger errors more heavily, which is important when predicting human ratings where large deviations are more noticeable.
- It is interpretable in the same units as the response variable (rating), making it easy to assess how close predictions are on average to actual values.

We chose RMSE over metrics like Mean Absolute Error (MAE) or R^2 because RMSE is particularly helpful when we want to emphasize minimizing large individual prediction errors, which could reflect particularly poor model performance on certain types of recipes.

Features Used (Available at Time of Prediction):

To avoid data leakage, we only use features that would be known before a user has left a rating. These include:

- minutes: Time it takes to complete the recipe.
- n_steps: Number of steps required to make the recipe.
- n_ingredients: Number of ingredients in the recipe.

We do not use features like average rating or the presence of a review (review column), since those are only known after multiple users have interacted with the recipe.

Baseline Model Description and Evaluation

Our final model is a linear regression model trained to predict the individual rating a user might give a recipe, using features that are available at the time of recipe posting.

Features in the Model:

Feature Name	Type	Description	Transformation
n_steps	Ordinal	Number of steps in the recipe	Binarized into "few" (≤5) vs. "many" (>5) steps using Binarizer
minutes	Quantitative	Total time required to complete the recipe in minutes	Binarized into "short" (≤30) vs. "long" (>30) recipes using Binarizer
n_ingredients	Quantitative	Number of ingredients used in the recipe	Used as-is (no transformation)

- Quantitative features: n_ingredients
- Ordinal features: n_steps, minutes (after binarization)
- Nominal features: None

- All necessary encoding was handled using sklearn's ColumnTransformer and Binarizer.

Model Performance:

Metric	Value
Mean Absolute Error (MAE)	0.4988
Mean Squared Error (MSE)	0.6064
R^2 Score	0.0034

Interpretation:

- The R^2 Score of 0.0034 suggests that the model explains less than 1% of the variance in recipe ratings, indicating very weak predictive power.
- While the MAE of ~0.5 suggests that on average the model's predictions are within half a star of the actual rating, this is only slightly better than always predicting the mean rating.
- These results suggest that our current model is not particularly good.

Why might this be?

- Recipe ratings may depend heavily on unobserved user-level preferences or subjective factors like taste, presentation, or cultural familiarity that aren't captured in the dataset.
- More predictive features—such as review content, author reputation, or number of past reviews—might improve performance.

Final Model

Feature Engineering

For the final model, I engineered two new features from the original numeric columns:

1. Standard Scaled minutes:
   I applied StandardScaler to normalize the minutes column. This transformation centers the data to have mean 0 and variance 1, which can help many models (especially tree-based

ones) distinguish extreme values better.

2. Quantile Transformed n_ingredients:
   I applied QuantileTransformer to n_ingredients, which spreads out frequent values and maps the distribution to follow a uniform one. This helps reduce skewness and improve the model's ability to handle outliers or uneven distributions.

These choices are informed by the data-generating process:

- The number of ingredients and prep time (minutes) are not uniformly distributed, which makes them harder for models to interpret directly.

- These features plausibly affect recipe rating — longer or overly complex recipes may deter high ratings, and ingredients influence complexity as well.

Final Model and Hyperparameter Tuning

I tested four models using RandomizedSearchCV for efficient hyperparameter tuning:

- Linear Regression

- Decision Tree Regressor

- Random Forest Regressor

- Support Vector Regressor

The final model selected was a Decision Tree Regressor, which performed best on the test set.

Hyperparameter Selection Method:

I used RandomizedSearchCV with 5 iterations and 3-fold cross-validation to sample a few combinations quickly.

Metric	Baseline (Linear)	Final (Decision Tree)
R^2 Score	0.0036	0.0220

MAE	0.4985	0.4879
MSE	0.6044	0.5933

Although the improvement is modest, the final model outperforms the baseline across all metrics. This suggests that the tree model captures some nonlinear relationships the linear model could not.

Fairness Analysis

Groups Defined

- **Group X**: Individuals with minutes ≤ 30 (short-duration recipes)
- **Group Y**: Individuals with minutes > 30 (long-duration recipes)

We chose this division because cooking time might reflect different types of recipes (e.g., quick meals vs. slow-cooked dishes), which could lead to unequal model performance if the model better learns patterns from one group over the other.

Evaluation Metric

We used **Root Mean Squared Error (RMSE)** to measure model performance across groups. RMSE is appropriate for regression problems and penalizes larger errors more heavily than MAE.

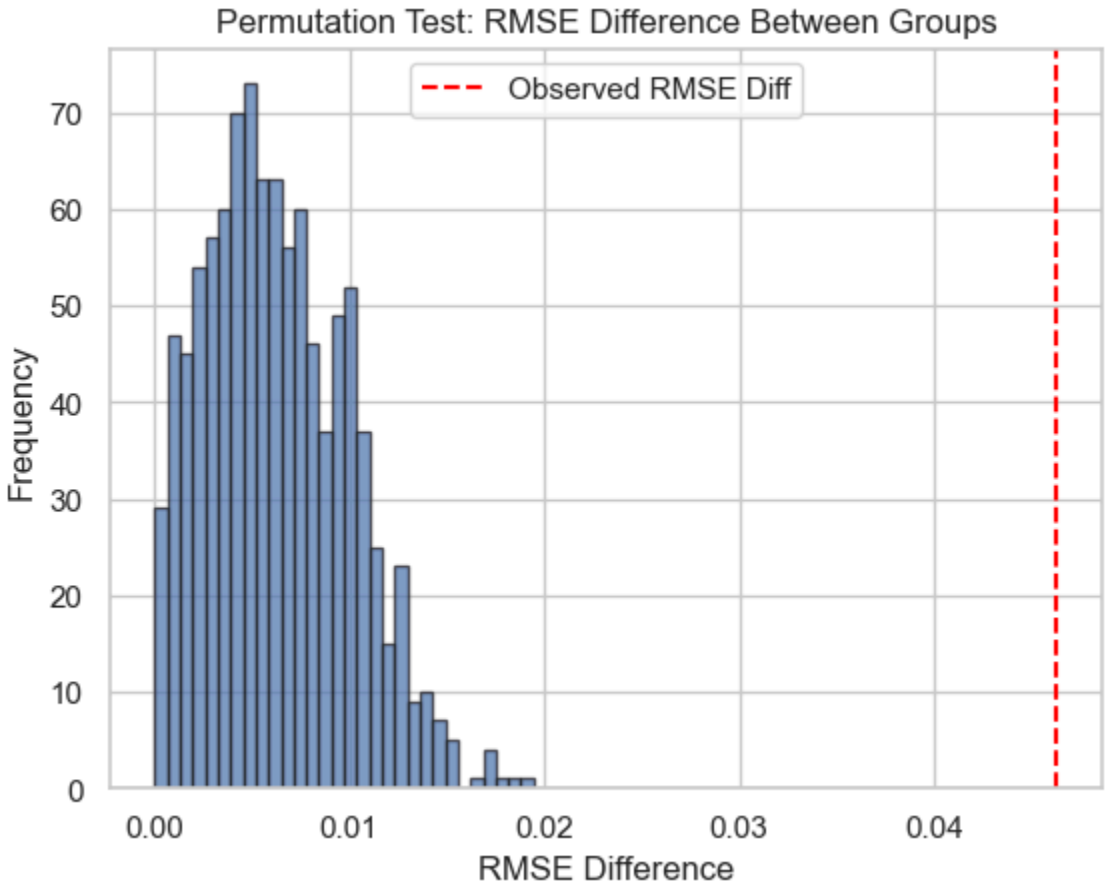
Hypotheses

- **Null Hypothesis (H_0)**: The model is fair. It performs equally well (in terms of RMSE) for Group X and Group Y. Any difference in RMSE is due to random variation.
- **Alternative Hypothesis (H_1)**: The model is **unfair**. It performs significantly worse (higher RMSE) for one group than the other.

Test Statistic and Procedure

- We used the **absolute difference in RMSE** between Group X and Group Y as our test statistic.
- We performed a **permutation test with 1000 permutations**: we shuffled the true `y_test` labels and recomputed the RMSE difference each time to simulate the distribution of RMSE differences under the null hypothesis.

Results



- **Observed RMSE difference**: 0.0463
- **P-value**: < 0.001

Conclusion

Since the p-value is significantly less than the common alpha level of 0.05, we **reject the null hypothesis**. This suggests that the model's performance **differs significantly** between the two groups — it is likely **less fair** with respect to recipe duration.