CSC311 Machine Learning Challenge Report

Hanli Jiang, Rohan Bhalla, Vincent Wang, Yuchen Wang

University of Toronto

CSC311: Introduction to Machine Learning

Prof. Alice Gao

April 1, 2025

Data

Data Representation

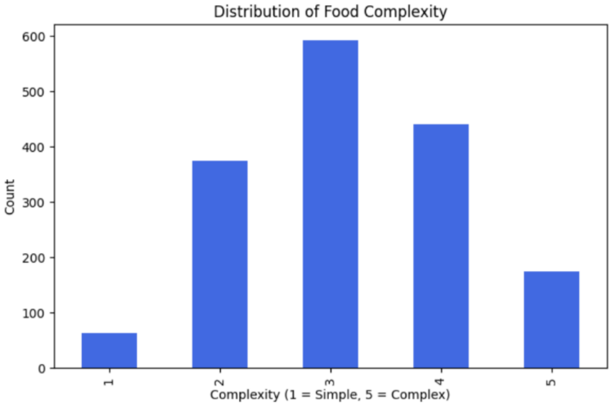
Given a raw dataset, we first need to do some preprocessing such as remove the NA values and convert string variables to integers so that the dataset after cleaning is consistent and easy for the ML model to interpret. A summary of how the dataset was preprocessed for each question is listed below:

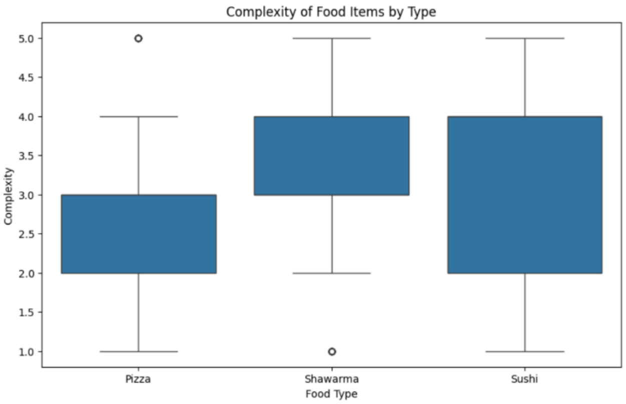
- **Q1 (Complexity, Single-select):** Encoded numerically from 1 (most simple) to 5 (most complex), as provided.
- **Q2 (Number of Ingredients, Free-text):** The original answer contains both text and numerical number. We convert all the text representations into numerical value. If the answer is a list, we use the average. Entries without a usable number were assigned NaN.
- **Q3 (Serving Setting, Multi-select):** Convert this categorical variable into one-hot encoding by creating binary columns for each setting option (e.g., "Weekday Lunch", "At a party"). Each column was marked with 1 if selected, otherwise 0.
- **Q4 (Price, Free-text):** Extracted numeric values or average ranges like what we did in Q2. Word-based numbers (e.g., "five dollars") were converted to numeric form. Non-convertible entries were marked NaN. We then replaced NaN values with the mean.
- **Q5 (Movie Association, Free-text):** Kept the 20 most frequent movie responses by first looping through each response. All others were grouped under the label "Other".
- **Q6 (Drink Pairing, Free-text):** Grouped similar drinks under broader categories (e.g., all variations of water were categorized as "Water"). Responses that didn't match any category were labeled "Other". Then we one-hot coded this categorical variable by creating binary (0/1) indicator columns for each drink category.
- **Q7 (Person Association, Multi-select):** Convert this categorical variable into one-hot encoding by creating binary columns for each person type (e.g., "Parents", "Friends"). Marked 1 if selected, 0 otherwise.
- **Q8 (Hot Sauce Amount, Single-select):** Encoded each response with a unique numeric value (e.g., 0 for "None", up to 4 for "I will have some of this food item with my hot sauce"), a higher value indicates higher hot sauce amount.

Data Exploration

After preprocessing the raw data, we visualized the data distribution and key features using various graphical representations.

Q1: Complexity of Food





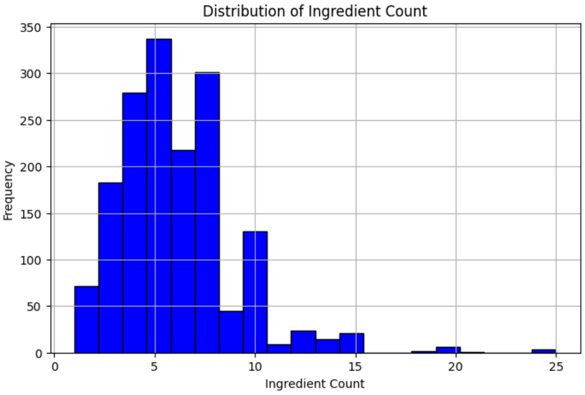
The histogram illustrates the distribution of food complexity, which approximately follows a normal distribution. Most foods have a moderate complexity level of 3, while complexity levels of 1 and 5 are relatively rare.

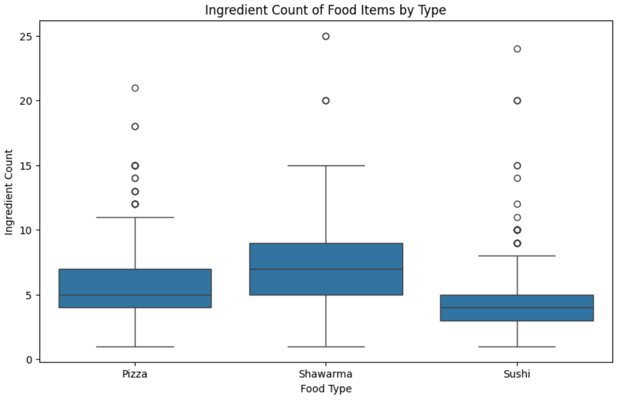
The boxplot, broken down by food type, reveals useful patterns for classification:

- Since the median complexity of shawarma is higher than pizza, we can infer that pizza generally has a lower complexity than shawarma.
- Shawarma and Sushi both exhibit higher median complexity scores around 4. However, Sushi has a wider interquartile range from 2 to 4, with many considering it either very simple or very complex, while Shawarma leans more consistently toward moderate to high complexity.

These patterns suggest that complexity could be a helpful feature in classifying the food item since each food has a slightly different distribution as well as different median.

Q2: Number of Ingredients





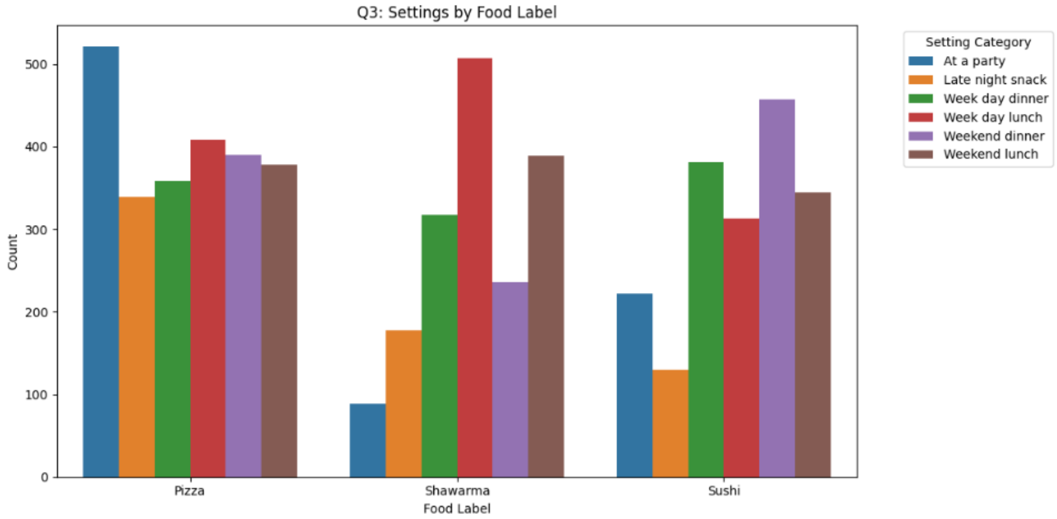
The overall distribution of ingredient count is right-skewed, with most values concentrated between 3 and 8 ingredients. A few outliers exceed 15, but they are relatively rare. This aligns with real-life scenarios, as these types of food typically do not contain an excessive number of ingredients.

The boxplot by food type reveals distinct patterns:

- Shawarma has highest median ingredient count of around 7–8 and a broader interquartile range compared to the others.
- Pizza follows with a slightly lower median around 5–6, though it has more extreme outliers (up to 21).
- Sushi shows the lowest median and the narrowest spread, suggesting a more consistent perception of simplicity in terms of ingredients.

Each food class has a relatively different distribution to distinguish from each other, indicating this is a valuable feature to include in our model. Given its usefulness, we prioritized careful preprocessing of this column.

Q3: Setting in Which Food is Served

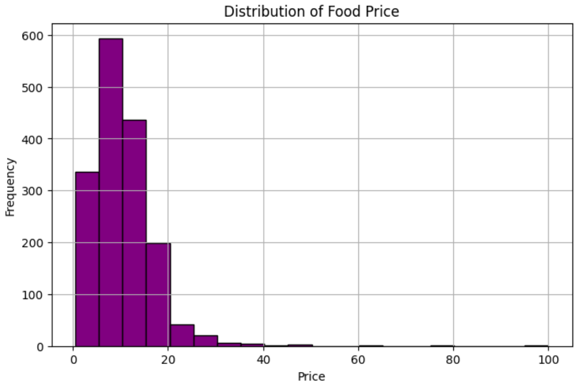


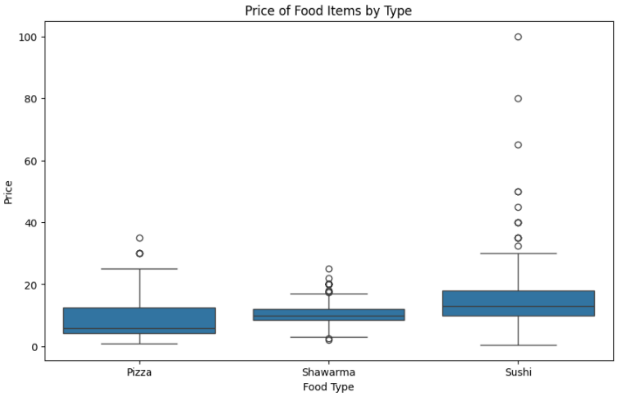
The bar chart reveals clear trends in setting preferences by food label:

- Pizza is most strongly associated with social or informal settings, particularly "At a party," where it had the highest count by far compared to other settings. It is also a popular choice overall, as it maintains a relatively high count across all setting categories.
- Shawarma stands out with strong associations to "Weekday lunch" and "Weekend lunch", with limited association to party or snack settings. This indicates it is perceived more as a meal than a snack or casual food.
- Sushi is more evenly distributed across "Weekday dinner", "Weekend dinner", and "Weekend lunch", but shows notably lower responses for "At a party" and "Late night snack", suggesting a more formal or sit-down perception.

These setting patterns are distinctive enough to make this a highly informative feature for classification. For example, strong associations like Pizza at parties or Shawarma for lunch could significantly improve the model’s ability to differentiate between food types. Since this question allows multiple responses, we represented each setting category as its own binary feature, which allows the model to pick up on these patterns independently and combine them with other features for better accuracy.

Q4: Price





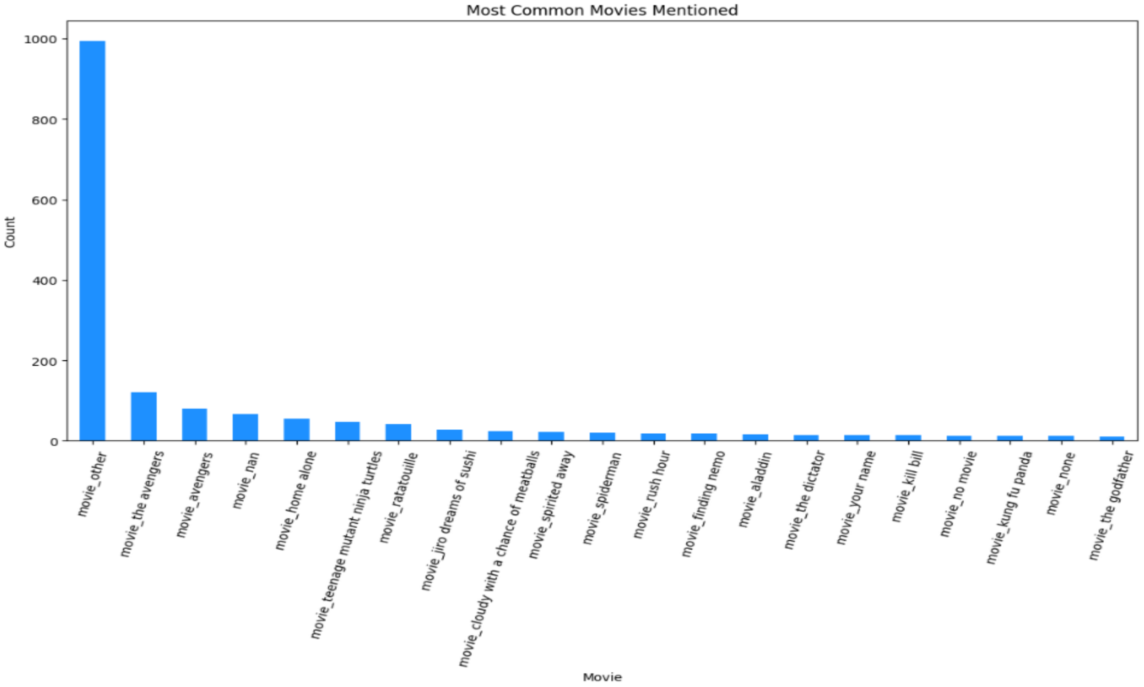
The overall distribution of price responses is heavily right-skewed. Most estimates fall between $5 and $20, with a long tail extending to high outliers up to $100. These high values are rare but noticeable and were preserved rather than removed, as they may reflect genuine perceptions (e.g., premium sushi).

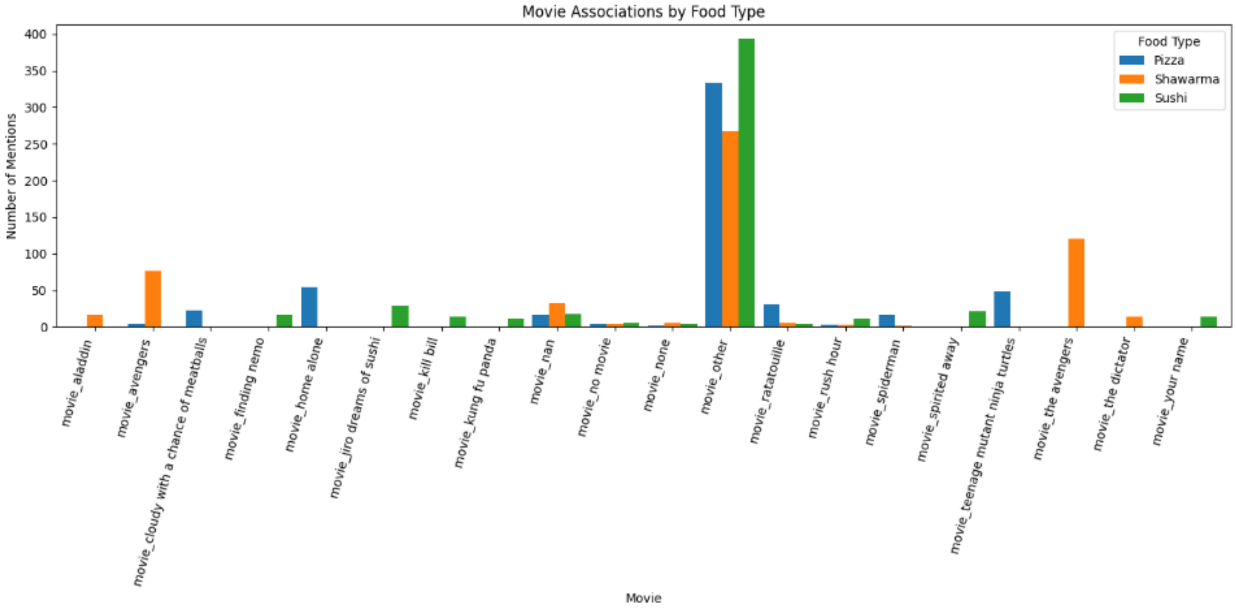
The boxplot broken down by food type reveals the following trends:

- Shawarma has the narrowest price range and the most tightly clustered values, with a median around $10. This suggests it is perceived as a relatively consistent, affordable meal.
- Pizza shows a wider spread with a lower median (around $7–8) but more variability in the upper range, likely due to differences in size (e.g., single slice vs. full pie).
- Sushi stands out with both the highest median price (approximately $15) and the widest range, including many extreme high outliers. This supports the idea that sushi is perceived as a more premium or variable-priced item.

These differences suggest that price is a meaningful feature, particularly for identifying Sushi. Since Shawarma prices are the most tightly distributed, the model can learn to associate mid-range, consistent prices with Shawarma, while high variability in Sushi may make it identifiable even when other features are ambiguous.

Q5: Movies





The first plot shows the top 20 most common movie responses across all food items. The "Other" category is by far the most frequent, indicating a high diversity in responses. Still, some patterns emerge from the structured responses:

- The most frequently mentioned specific movies include *The Avengers*, *Teenage Mutant Ninja Turtles*, *Kung Fu Panda*, and *Ratatouille*.
- The grouped bar chart by food type helps highlight associations:

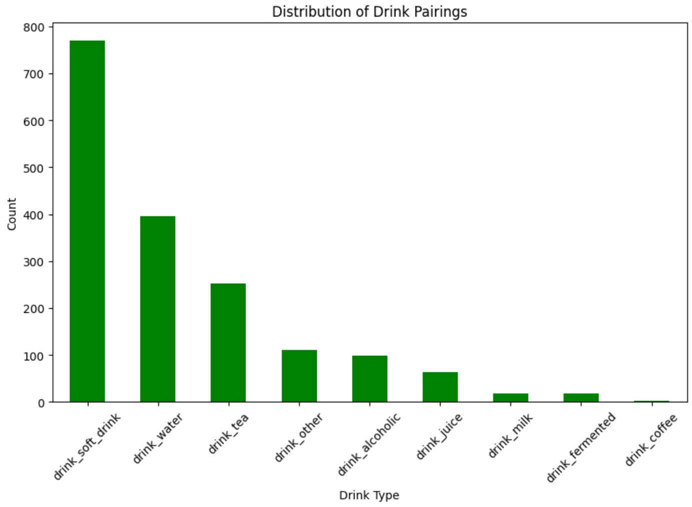
- *Teenage Mutant Ninja Turtles* and *Spiderman* are primarily linked to **Pizza**, likely due to cultural references in children's media.
- *The Dictator* and *Aladdin* are more often associated with **Shawarma**, possibly due to Middle Eastern settings in the films.
- *Jiro Dreams of Sushi* and *Spirited Away* were uniquely associated with **Sushi**, with *Jiro* standing out as an especially relevant and direct match.

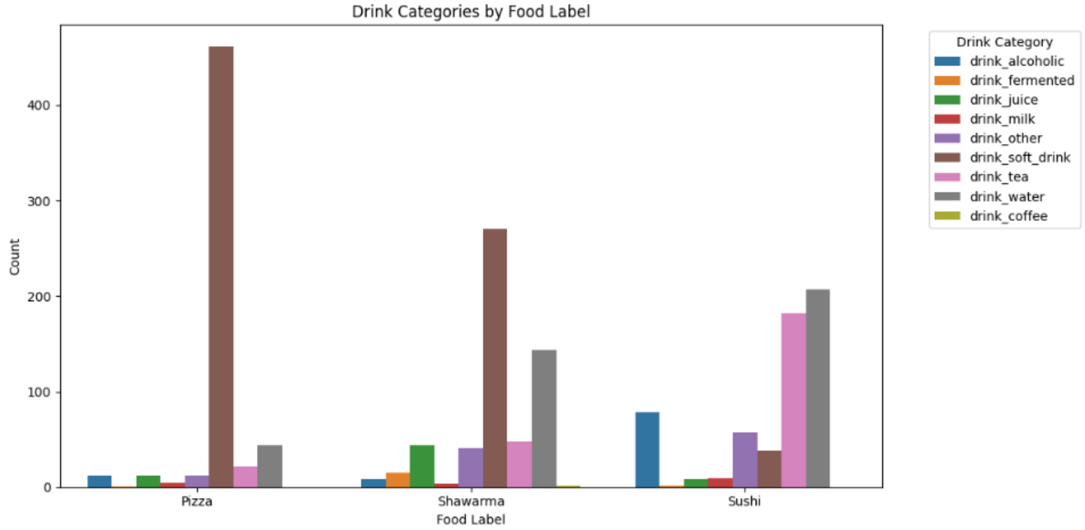
This feature introduces high variance due to its open-ended nature. Due to the sheer number of movies out there it also becomes difficult to group responses. For example, when a respondent answers “None”, do they mean there is no movie they associate this film with, or are they referring to a movie called None? Additionally, several films share the same title, making it difficult to discern which movie is actually being mentioned, possibly creating noise in the dataset.

The subjectivity of the question, combined with the overwhelming dominance of the "Other" category, makes this feature noisy and potentially less informative than others. There is also a risk of overfitting if the model relies too heavily on niche movie references.

As a result, we plan to evaluate the impact of this feature by training models both with and without the movie features. This will allow us to empirically assess whether including these variables contributes meaningfully to classification performance, or whether it introduces unnecessary noise.

Q6: Drinks





The overall distribution chart shows that the most common drink categories were:

- Soft drinks, with the largest share by far
- Water, tea, and other, which made up a significant portion of responses
- Less common responses included alcoholic drinks, juice, milk, and fermented drinks

The second chart, which breaks drink preferences down by food type, highlights some clear patterns:

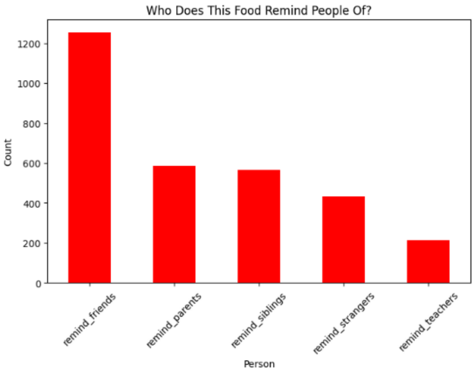
- Pizza is overwhelmingly associated with soft drinks, followed by some associations with tea and water

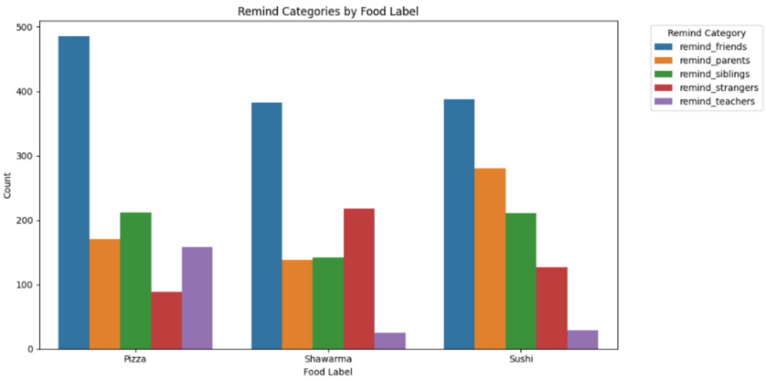
- Shawarma shows more varied pairings, with soft drinks, water, and tea as the top choices, but also notable mentions of juice and other
- Sushi stands out for having relatively high mentions of water, tea, and alcoholic drinks, and much lower association with soft drinks compared to the other two

These differences indicate that drink pairing could be a useful categorical feature for classification. For example, a response of "soft drink" strongly increases the likelihood that the food is Pizza, while tea or alcohol is more frequently associated with Sushi.

That said, the features still carry some noise, especially in less common categories—and may be sensitive to cultural bias or individual interpretation. We will retain this feature for modeling but also evaluate feature importance to assess whether it meaningfully contributes to performance. If it proves less helpful, we may consider collapsing less frequent categories even further or testing the model's accuracy with and without this input.

Q7: Who does this food remind you of?





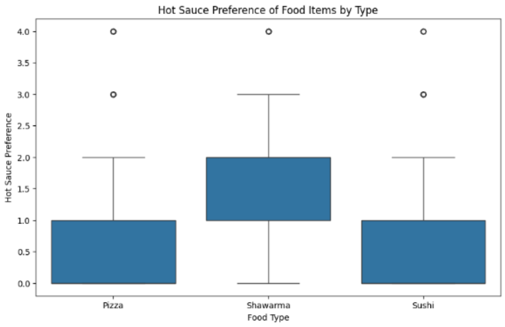
The overall distribution reveals that friends were the dominant choice by a large margin, followed by parents and siblings. Strangers and teachers were less common, suggesting most people associate food with family and personal relationships.

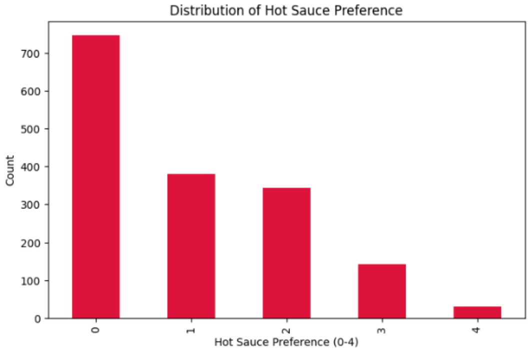
The breakdown by food type provides more insight:

- Pizza was overwhelmingly associated with friends, which aligns with its social and casual nature, often eaten at parties or shared with peers.
- Shawarma showed a more even spread across friends, siblings, and strangers, indicating a broader set of associations.
- Sushi was most often linked with friends and parents, suggesting it may be seen as a food enjoyed in both peer and family contexts.

Although this feature is inherently subjective, the multi-label format and differences in how people associate foods with social groups do offer useful signals. For example, strong associations with “teachers” were relatively unique to Pizza, and “strangers” were far more likely to be linked with Shawarma than other foods. These differences give the model some leverage to differentiate between classes, especially when used in combination with other features. We consider this a moderately useful feature and will retain all five binary indicators in our final model. While it may not drive the classification on its own, its ability to capture social context provides a subtle but meaningful layer of information.

Q8: Hot Sauce





The overall distribution is heavily skewed toward lower values. The majority of responses selected 0 (none), followed by moderate levels of heat (1 and 2). Very few respondents opted for the highest levels (3 or 4), suggesting that extreme spiciness is not commonly associated with any of the food types in this survey.

The boxplot broken down by food label reveals some subtle but consistent differences:

- Shawarma has the highest median hot sauce preference and a wider range, indicating that respondents are more likely to associate it with spicy customization.
- Pizza and Sushi both have lower medians and tighter distributions, suggesting that they are typically seen as foods not commonly paired with hot sauce.

Although the feature is not highly variable across food types, the distinction for Shawarma is notable enough to justify its inclusion. Its clear separation in hot sauce preference—especially at the higher end—adds discriminative value that can help the model distinguish Shawarma from the other two. For example, if a response indicates a strong preference for heat, it significantly boosts the likelihood that the food is Shawarma.

We consider this a low-dimensional but useful feature, particularly because it is easy for the model to interpret and introduces no noise from preprocessing or text normalization.

Feature Selection

We initially decided to retain all eight survey questions(Q1–Q8) as input features, as our previous data exploration section revealed that each question contributes information to distinguishing food items, with no two foods having highly similar distributions. This approach ensures that no valuable information from the original dataset is lost, ensuring that the classifier is efficient. However, we were uncertain about whether to include the movie associations question (Q5) due to its noisiness and sparsity, as the majority of responses were categorized as "Other." We suspected that incorporating this potentially less informative data might introduce more noise rather than capturing meaningful patterns related to food items, potentially leading to overfitting in our model.

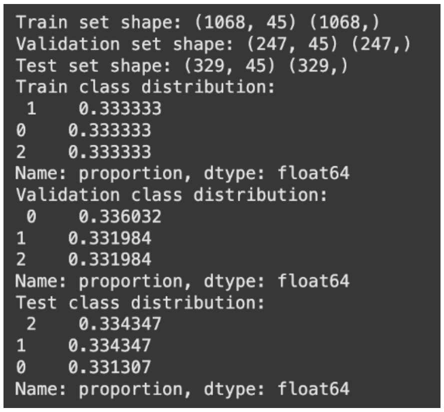
That being said, we fitted two models—one including Q5 and the other excluding it—while keeping all other parameters fixed. The results showed that the model with Q5 achieved a higher accuracy rate. Therefore, we decided to include Q5 in our final model.

No features were removed for simplicity; we prioritized keeping as much signal as possible and will rely on performance evaluation to guide any future exclusions.

Data Splitting

The dataset was split into training, validation, and test sets using a stratified approach to ensure that class distributions remained consistent across all subsets. First, the dataset was separated into features (x) and the target variable (y), with the target column labeled as "Label". Since the target variable contained categorical values, we converted the labels into numerical format for compatibility with different models. To create the data splits, we first allocated 20% of the dataset as the test set. The remaining 80% was then further divided, with 15% of the original dataset allocated to the validation set. This fraction was computed relative to the remaining data to ensure the final proportions resulted in 68% of the dataset for training, 12% for validation, and 20% for testing. Stratification was applied during each split to preserve the equal class distribution in all subsets.

After splitting, we standardized specific numerical features—"1. complexity," "2. ingredient count," "4. price," and "8. hot sauce". The scale was fitted on the training data and then applied to both the validation and test sets to maintain consistency. Finally, we confirmed the correctness of the splits by examining the dataset shapes and verifying that the class distributions were preserved across all subsets. This structured approach ensures that the training set provides a robust foundation for model learning, the validation set facilitates hyperparameter tuning, and the test set offers an unbiased evaluation of model performance.



```
Train set shape: (1068, 45) (1068,)
Validation set shape: (247, 45) (247,)
Test set shape: (329, 45) (329,)
Train class distribution:
 1    0.333333
0    0.333333
2    0.333333
Name: proportion, dtype: float64
Validation class distribution:
 0    0.336032
1    0.331984
2    0.331984
Name: proportion, dtype: float64
Test class distribution:
 2    0.334347
1    0.334347
0    0.331307
Name: proportion, dtype: float64
```

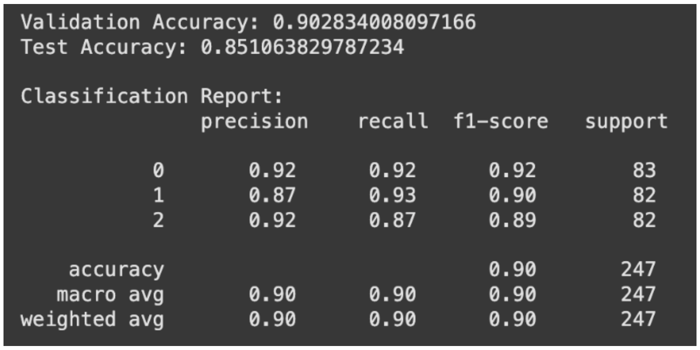
Model Exploration

As a group, we aimed to explore a diverse range of machine learning models, ensuring that each team member contributed to evaluating at least one approach. Our exploration included two different Multilayer Perceptron (MLP) models, a Random Forest, a Decision Tree, Logistic Regression, and an XGBoost model. The MLP models were chosen to assess the effectiveness of neural networks in capturing complex patterns within the data. We experimented with two architectures to understand how varying the number of layers and neurons impacted performance. Given their ability to learn intricate feature interactions, MLPs provided a strong deep-learning-based approach for comparison.

In addition to neural networks, we investigated tree-based models, including a Decision Tree and Random Forest. The decision tree served as a simple, interpretable model capable of handling non-linearity, though we recognized its tendency to overfit. To mitigate this, we included Random Forest, an ensemble method that constructs multiple decision trees and aggregates their predictions, improving generalization and robustness to noise. We further extended our tree-based exploration with XGBoost, a gradient-boosting algorithm known for its strong performance in structured data tasks. XGBoost incrementally improves weak learners, optimizing predictive accuracy while preventing overfitting through regularization.

To establish a baseline for comparison, we also implemented Logistic Regression, a linear classifier commonly used for classification problems. Logistic regression provided insight into how well a simple, interpretable model could perform relative to more complex methods. . Each model was selected with a clear rationale, allowing us to compare performance and determine the most effective method for our specific task.

Ultimately, we compared the performance of all the models and found that one of our MLP models achieved the best results. Further details on its performance and evaluation are provided in the next section.



```
Validation Accuracy: 0.902834008097166
Test Accuracy: 0.851063829787234

Classification Report:
               precision    recall  f1-score   support

           0       0.92      0.92      0.92        83
           1       0.87      0.93      0.90        82
           2       0.92      0.87      0.89        82

    accuracy                           0.90       247
   macro avg       0.90      0.90      0.90       247
weighted avg       0.90      0.90      0.90       247
```

Model Selection

We explored several model families—including logistic regression, random forests, Multi-Layer Perceptron (MLP), and XGBoost—during the early stages of this project. Our baseline logistic regression model achieved 87% validation accuracy, which confirmed the integrity of our preprocessing pipeline and data splits. Given the complex, non-linear interactions among our features (numeric, ordinal, one-hot/multi-hot encoded categorical variables), we anticipated that a non-linear model would capture these relationships more effectively.

After comparing evaluation metrics across various models, we selected the MLP as our final candidate. The MLP's flexibility in architecture and ability to learn non-linear decision boundaries made it particularly well suited for our multi-class classification task (Pizza, Shawarma, Sushi). Importantly, the evaluation metrics for all models were computed on the same validation set, ensuring that our comparisons were fair and directly comparable.

Evaluation Metrics

For model evaluation, we focused on the following metrics:

- **Accuracy:** The proportion of correctly classified examples. This provides a straightforward measure of overall performance.
- **Precision, Recall, and F1-Score:** These metrics are computed for each class.
    - **Precision** helps us understand how many of the predicted instances for a class are correct.
    - **Recall** indicates the proportion of actual instances of a class that are correctly identified.
    - **F1-Score** (the harmonic mean of precision and recall) provides a balanced measure that accounts for both false positives and false negatives.
- **Macro and Weighted Averages:** These averages allow us to compare performance across classes, especially when the classes are balanced.

We chose these metrics because our problem is a multi-class classification task, and it is essential to achieve balanced performance across all classes. The metrics give us a comprehensive view of the model's ability to correctly predict each food type while minimizing misclassifications.

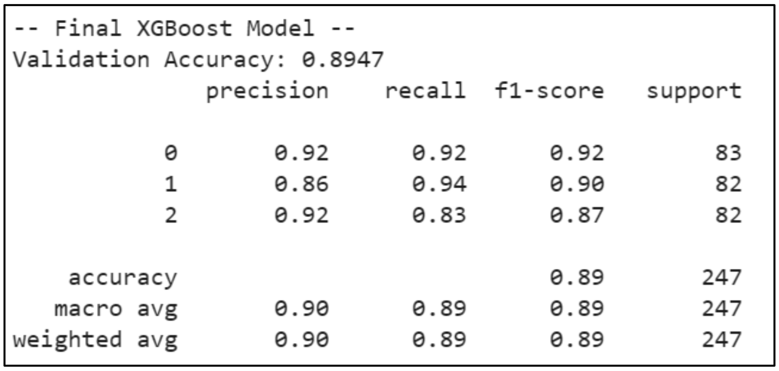
Hyperparameter Tuning

In our hyperparameter tuning process, we used Optuna, a Python library that helps find the best settings (hyperparameters) for machine learning models. Optuna uses a method called Tree-structured Parzen Estimator (TPE). This technique is a form of Bayesian optimization that learns from past results to choose better hyperparameter combinations over time. For example, if certain learning rates or neuron counts lead to better accuracy, Optuna will focus on testing more values in those ranges. This makes the tuning process more efficient and effective. We applied Optuna to both our XGBoost and MLP models by defining functions that tested different settings—like the number of trees in XGBoost or the number of neurons in the MLP—and returned the average accuracy from cross-validation.

For XGBoost, our search space included:

- **n_estimators:** Tested in increments from 50 to 300.
- **max_depth:** Ranged from 2 to 10.
- **learning_rate:** Sampled on a log scale between 0.001 and 0.3.
- **subsample:** Values between 0.6 and 1.0.
- **colsample_bytree:** Values between 0.6 and 1.0.
- **gamma:** Integer steps between 0 and 5.

The objective function maximized the average accuracy over the cross-validation folds. After running 30 trials, the best hyperparameters for XGBoost were determined (as shown by the output of our Optuna study), and these settings were used to train a final XGBoost model. The final model achieved the following classification report.

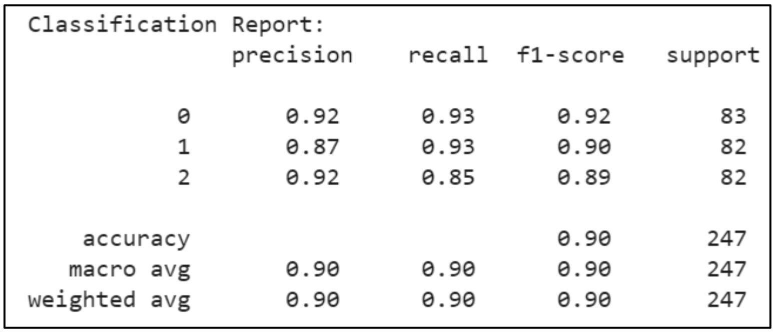


```
-- Final XGBoost Model --
Validation Accuracy: 0.8947
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        83
           1       0.86      0.94      0.90        82
           2       0.92      0.83      0.87        82

    accuracy                           0.89       247
   macro avg       0.90      0.89      0.89       247
weighted avg       0.90      0.89      0.89       247
```

For the **MLPClassifier**, we defined a search space including:

- **hidden_layer_num:** An integer from 1 to 3.
- **hidden_layer_size:** An integer from 10 to 200 (evaluated in steps of 10).
- **alpha:** The L2 regularization strength, sampled on a log scale between 1e-5 and 1e-1.
- **learning_rate_init:** Also sampled on a log scale between 1e-4 and 1e-1.
- **activation:** Options were "tanh" and "relu".
- **solver:** Either "adam" or "sgd".

Using a similar 3-fold cross-validation scheme and 30 trials, Optuna selected a configuration that yielded the following classification report.



```
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.93      0.92        83
           1       0.87      0.93      0.90        82
           2       0.92      0.85      0.89        82

    accuracy                           0.90       247
   macro avg       0.90      0.90      0.90       247
weighted avg       0.90      0.90      0.90       247
```

Since our MLP model achieved a higher average recall and f1-score than XGBoost, we decided to select it as our final model. The hyperparameters are recorded as follows:

Final Model

Our final model is an MLP model with the following hyperparameters:

- **hidden_layer_num:** 1
- **hidden_layer_size:** 120
- **alpha:** 0.022794022713833025
- **learning_rate_init:** 0.010609828889300572
- **activation:** “tanh”
- **solver:** "sgd"

Prediction

Performance Estimate

We estimate that our model will achieve an accuracy of 85.1% on the test set.

Reasoning and Justification

This estimate is based on our model's performance on a held-out test set, which was created using a stratified split to ensure that the class distribution matched that of the overall dataset. The test set was never used during training or hyperparameter tuning, making it a reliable indicator of how the model will perform on the unseen test set.