ML Challenge Report: Ensemble Logistic Regression Model

Suhaima Ahmad, Sreya Sunil, Angelin Romieo

Department of Mathematical and Computational Sciences

CSC311: Introduction to Machine Learning

Dr. Lisa Zhang

April 4, 2025

ML Challenge

Understanding how people perceive different foods can offer valuable insights into their preferences and behaviors. In this project, we worked with a dataset of survey responses where participants answered questions about three different types of food: pizza, shawarma, and sushi. Our goal was to build a machine learning model that could predict one of these food based on answers to eight questions.

The survey data included a mix of numerical, categorical, and open-ended responses, requiring significant preprocessing and creative feature engineering. We explored the data in depth, reasoning through the usefulness of each feature, applying appropriate parsing and cleaning techniques, and conducting statistical tests where necessary. Multiple model families were evaluated, and after extensive experimentation, we selected an ensemble of logistic regression models as our final approach due to its strong performance.

Data Exploration

There were eight survey questions of various types. Below we will go over the exploration of each question.

Q1: From a scale 1 to 5, how complex is it to make this food? (Where 1 is the most simple, and 5 is the most complex)

This feature is represented as a numerical value (1-5). The distribution of this parameter for each label is the following:

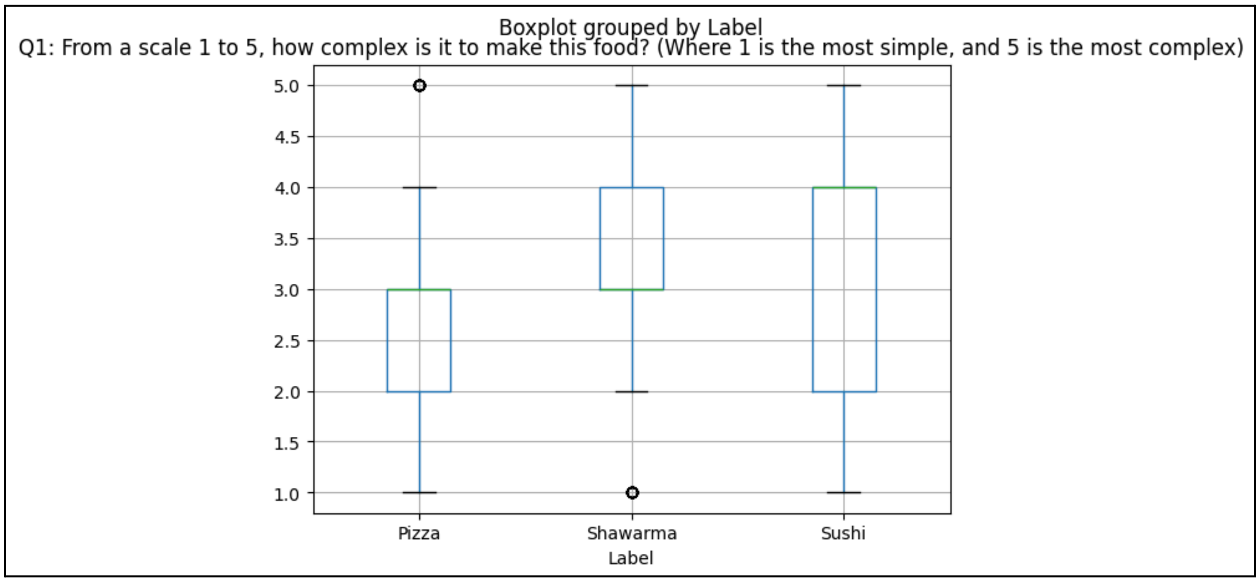

Figure 1: Box Plot of Question 1 Comparing the Rating of Complexity to Make Each Dish

The distributions for Pizza and Shawarma are well separated, however the distribution for Sushi overlaps the other two. We found that this feature lowered our model performance since it was possible for the model to predict Sushi even if the input parameter was in the range of Pizza or Shawarma.

Q2: How many ingredients would you expect this food item to contain?

This feature contained a mix of both numerical and textual data, such as an estimated number of ingredients and an ingredient list. There were more samples that only included an estimated number of ingredients, hence we decided to take the average of all the numbers in each data sample. In doing so, 85 of the 1644 data samples are not considered. The average ingredient number for each label produced the following box plot.

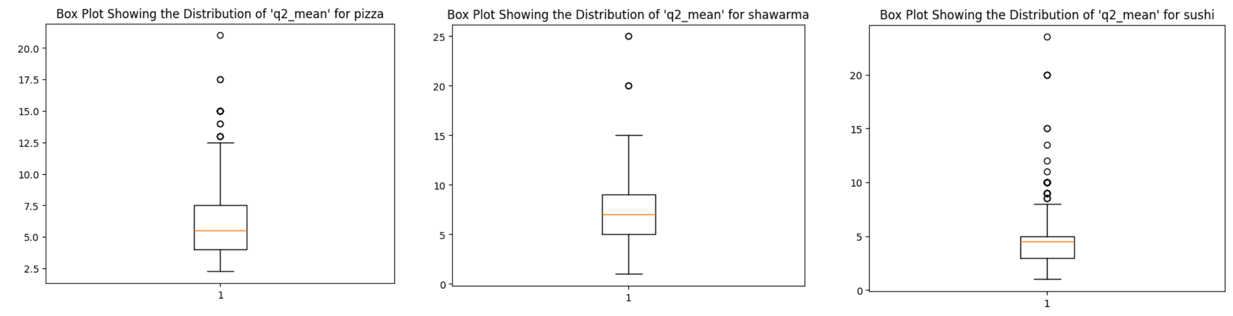

Figure 2: Box Plots Showing the Distribution of Average Number of Ingredients Per Dish

The average number of ingredients showed little variation across labels, indicating that it is not an informative predictor. Therefore, this feature was not used in the final model.

Q3: In what setting would you expect this food to be served? Please check all that apply

This feature was one-hot encoded with the parameters 'Week day lunch', 'Week day dinner', 'Weekend lunch', 'Weekend dinner', 'At a party', and 'Late night snack'. The chi square heat map showed strong correlation between each of the features and the labels.

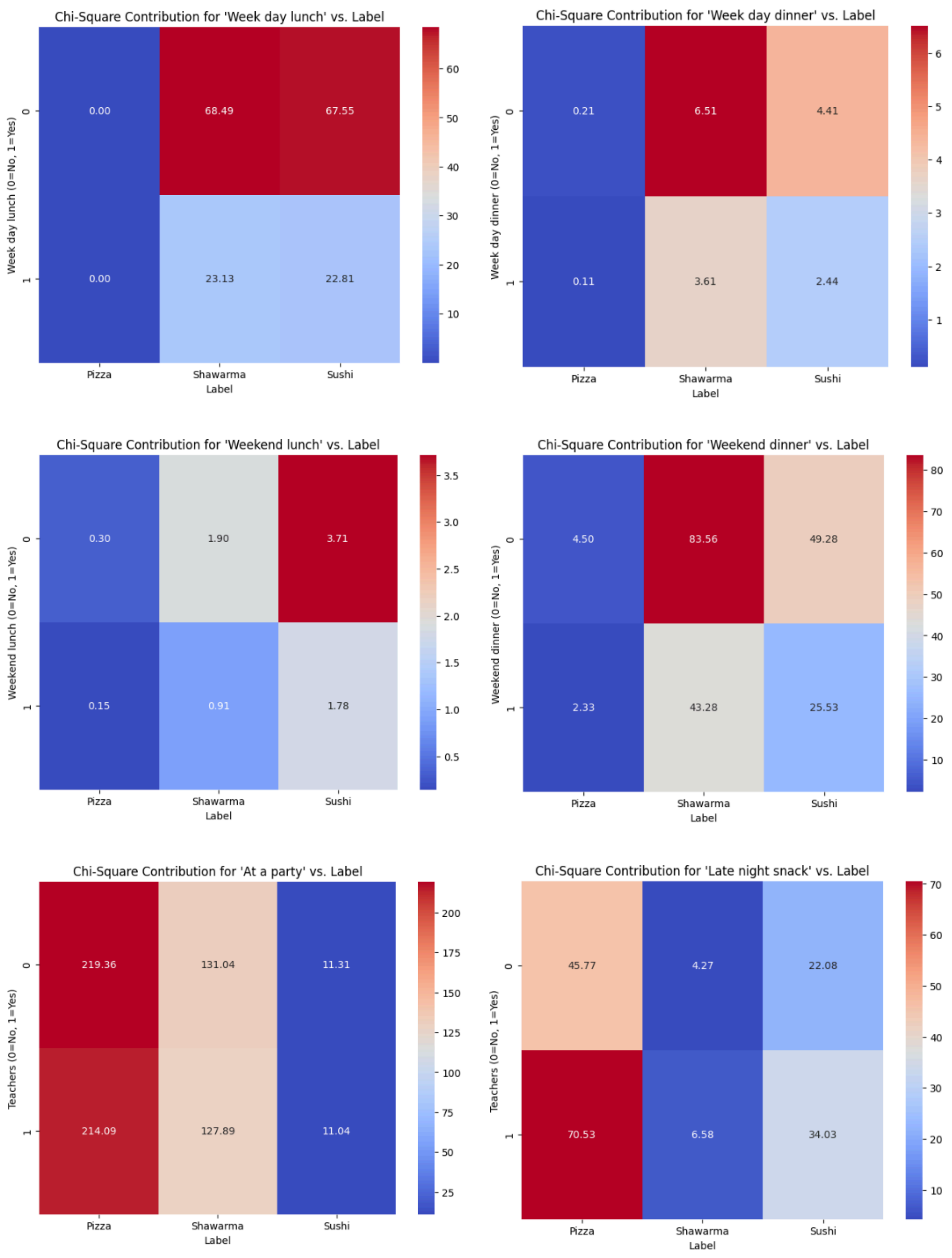

Figure 3: Chi-square Heatmaps Showing the Contribution of Each Parameter

As shown in the diagrams above, this question is an informative indicator for certain labels. For example, the feature 'At a party' has a high contribution to the 'Pizza' label, indicating its relevance.

Q4: How much would you expect to pay for one serving of this food item?

For question 4 in the survey, we encountered several challenges during data exploration. First, there were discrepancies in how participants interpreted the question. For instance, when asked about the price of pizza, many respondents provided the cost of an entire pizza ($20–$30) rather than a single serving.

A secondary challenge was that responses often contained full sentences mentioning multiple prices, rather than a single numeric value. For example, one participant wrote: “If it's in a fast food pizza store, a single slice costs around $6, and a full pizza costs around $15. For a full wood-fired pizza in a restaurant, it's around $25.” This made the data harder to parse.

To clean the data, we first applied a decoder and encoder to remove any non-Unicode characters. We then used regular expressions to extract all numerical values from the text. In cases where multiple numbers were mentioned, we generated a list of values (e.g., [6, 15, 25]) and took the **median** of the extracted values as the final estimated price for that response.

We considered alternative methods such as selecting the smallest extracted value or filtering based on nearby keywords like "slice" or "serving". However, for our purposes, the median method reduces computational complexity and works well.

We also created several graphs and box plots to get a feel for the data and make intuitive judgments on how it differs for each category.

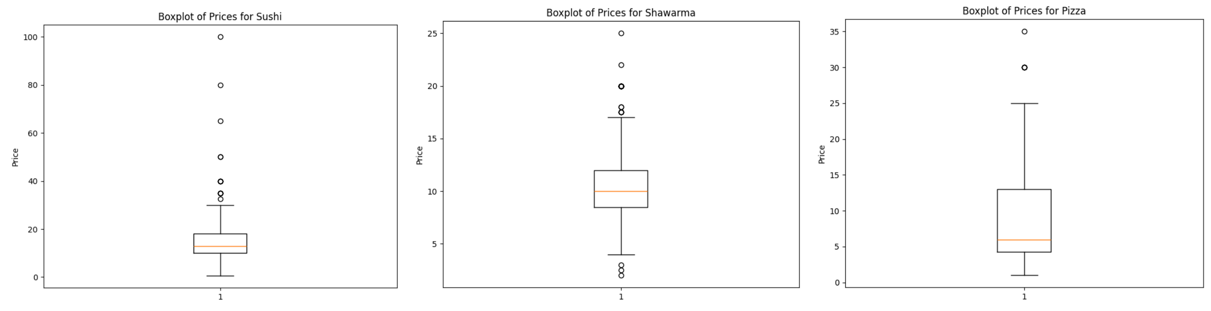

Figure 4: Box Plots Showing the Distribution of Average Price for Each Label

Q5: What movie do you think of when thinking of this food item?

This feature is represented using a bag-of-words. After preprocessing that included removing stop words, newlines, and capitalizations, we found some of the most common phrases associated with each label. For instance, Home Alone appeared 53 times under the label Pizza, Avengers appeared 187 times under Shawarma, and Jiro Dreams of Sushi appeared 28 times under Sushi. Since there were recurring movies in each of these labels, we chose to represent this feature as a bag-of-words. To limit the size of the input vertex, we included only words that appeared more than three times in the dataset.

Q6: What drink would you pair with this food item?

For question 6, we focused on standardizing open-text responses to create a manageable set of drink categories. Participants' answers varied widely in format, including brand names, drink types, misspellings, and full sentence responses. To clean the data, we first applied Unicode normalization and basic text preprocessing by removing extra spaces, hyphens, and capitalization differences. We then parsed each response by applying a regular expression search and selecting the first drink mentioned.

After extracting all unique drink responses, we manually grouped similar entries together to form a smaller set of high-level categories. The possible matches for each of the categories included different names of the same drink (i.e. carbonated water and sparkling water), similar drinks that belong in the same category (Sprite and 7UP belong in Soda), and common misspellings. Our final categories were: [Alcohol, Cola, Juice/Lemonade, Soda, Soup, Water, Other, and None]. This grouping allowed us to reduce noise and variability in the data while preserving important distinctions. By generating bar graphs for each drink category, we can see a correlation of certain drinks with certain labels, suggesting a correlation.

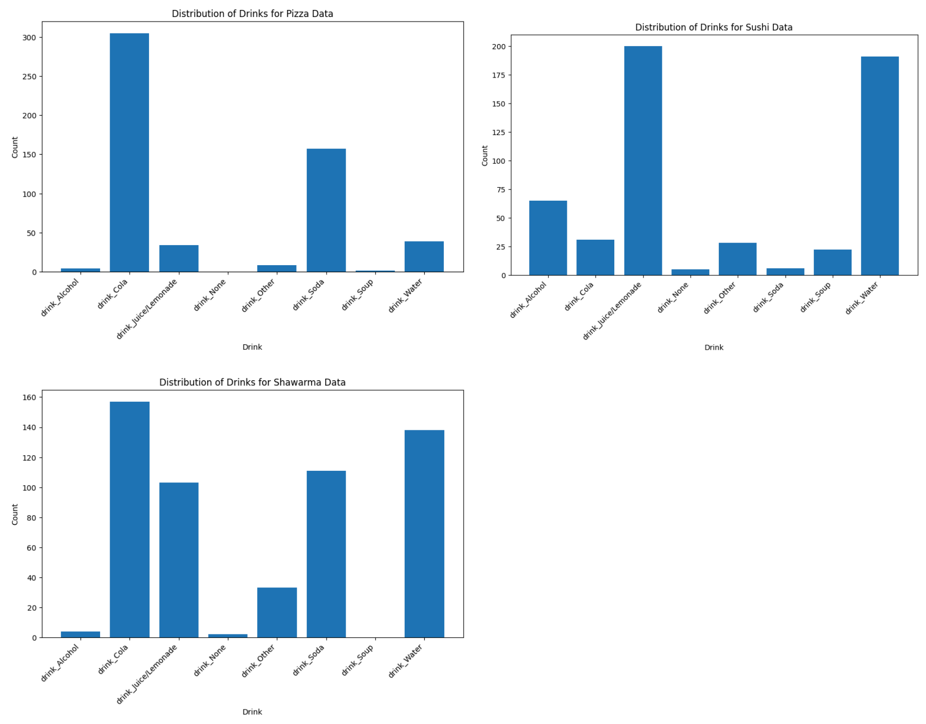

Figure 5: Bar Graphs Depicting the Distribution of Drinks for Each Label

Q7: When you think about this food item, who does it remind you of?

Since question 7 on the survey was a multiple choice question, there was no data processing required before it was fed into the model. However, the data exploration for this question involved reasoning out the usefulness of each question. Question 7 specifically was a multiselect multiple choice question that allowed people to answer with as many or as few options as they want.
To better understand the relationship between the selected options and the target variable, we conducted statistical tests to assess the significance of each choice. The p-values obtained for different categories (Teachers, Parents, Strangers, Friends, and Siblings) were all extremely small, with the largest p-value being approximately 2.2×10^{-6} for Siblings. These small p-values indicate that the choices participants made for this question are statistically significant and meaningfully correlated with the target. In particular, Teachers had the smallest p-value (4.09×10^{-41}), suggesting a very strong association.

Given the strong statistical significance across all options, we decided to retain all features generated from this question in our models without dropping or combining categories. This ensured that we preserved potentially valuable information related to participants' emotional or social connections to the food items.

To further explore this finding, we isolated the "Teachers" option and visualized its contingency with the target labels. The chi-square contribution heatmap shows that selecting "Teachers" is most strongly associated with the "Pizza" label, further supporting the observed statistical significance.

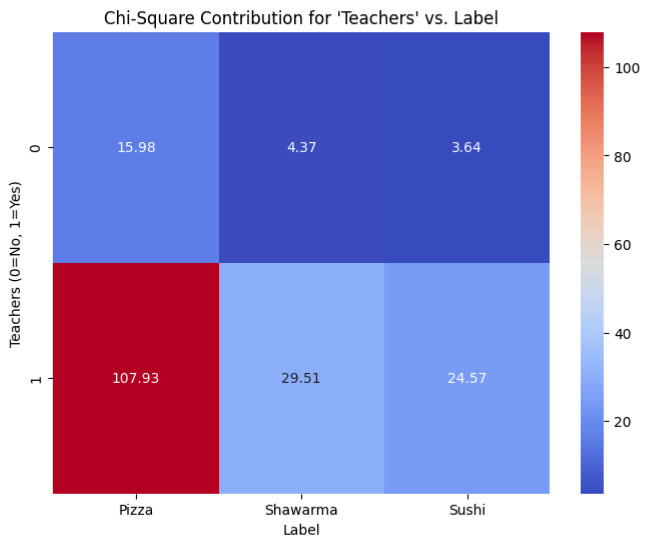

Figure 6: Chi-square Heatmap Showing the Contribution of the Parameter 'Teacher'

Q8: How much hot sauce would you add to this food item?

Question 8 was a multiple choice question, but unlike other survey questions, the available options had a natural numeric relationship (e.g., no hot sauce, a little, moderate amount, large amount). To facilitate model training, we mapped these choices to corresponding numeric values.

Since the data was already ordinal, no complex preprocessing was needed beyond this encoding. However, we explored the distribution of these numeric values using boxplots grouped by the food label. As shown in the boxplot, responses for Pizza and Sushi were generally skewed toward lower hot sauce preferences, while Shawarma showed a higher median value, indicating participants typically preferred adding more hot sauce to Shawarma compared to the other food items.

These patterns suggest that the amount of hot sauce a participant would add may carry predictive information about the type of food item being described, justifying the inclusion of this feature in our models.

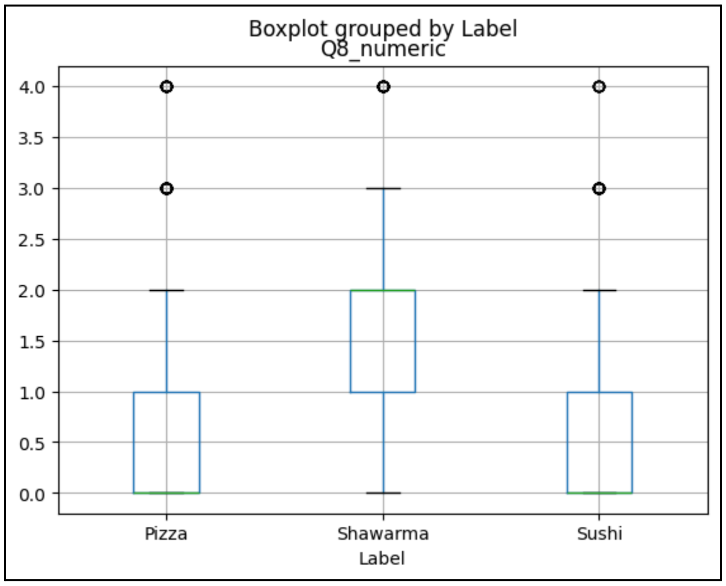

Figure 7: Box Plots Showing the Distribution of Hot Sauce Added for Each Label

Data Splitting

To ensure a fair evaluation and avoid data leakage, we carefully structured our data splitting process. Each survey respondent contributed three related data points — one each for pizza, sushi, and shawarma — which are inherently correlated. To reduce bias and prevent information about a participant from leaking into both the training and validation sets, we ensured that all three data points from a single respondent were kept together during splitting.

We first split the dataset into a training+validation set and a test set, allocating roughly 10% of the data to the test set. Then, we further split the training+validation set into a training set (60%) and a validation set (30%). We set shuffle=False in both splitting steps to maintain the original ordering of the data, which grouped each respondent's entries consecutively. This method ensured that models were evaluated on entirely unseen participants during validation and testing.

Model Family Exploration

Random Forest

We explored random forest models as part of the tree-based family of models. Random forests combine multiple decision trees and average their outputs, which helps reduce overfitting compared to a single decision tree. We trained the random forest model on the raw, cleaned features without applying standardization or normalization, as tree-based models are naturally scale-invariant.

During experimentation, we observed that random forests provided much more stable and reliable validation performance compared to single decision trees. To optimize performance, we tuned hyperparameters of the sklearn model, such as *n_estimators* (the number of trees) and *max_depth* (the maximum depth of each tree). We found that limiting the depth of individual trees helped prevent overfitting, while increasing the number of trees improved model stability. Random forests also handled noise in the features well, making them a strong candidate compared to simpler models. However, in terms of pure accuracy, this model did not perform as well as the others as we were able to achieve only an 87% accuracy at best.

Multilayer Perceptron (MLP)

To explore the neural network model family, we trained and evaluated our data using multilayer perceptron (MLP). We included all relevant features from the data section. Since not all features were one-hot encoded, we applied StandardScaler from scikit-learn to normalize the input data.

We used the MLP Classifier from scikit-learn to implement the model. After experimenting with hyperparameters, we found that a hidden layer size of (100, 10), Relu activation, 500 iterations and default hyperparameters of the model provided the best validation accuracy for this model.

Logistic Regression

We explored logistic regression by first one hot encoding all of the data. We did this because most of the features are categorical other than question 4. Question 4 is about prices which is a continuous feature. Consequently, to one hot encode question 4, we categorized the prices based on price range. As a result, we were able to build a one hot encoded data matrix to train logistic regression models.

Additionally, we found that question 1 and question 2 features were decreasing the validation accuracy. Therefore, for further exploration, we did not include these features in the data matrix.

For model fitting, we used scikit-learn's logistic regression. From this, we tuned the hyperparameter C (inverse of the regularizer).

Logistic Regression Ensemble

By using scikit-learn's bagging classifier, we explored an ensemble of logistic regression models. Our reasoning to explore an ensemble is because they reduce bias and variance compared to a single model. Logistic regression can be sensitive to small changes in the training data. Therefore, by bootstrapping samples and averaging over multiple models, we can create a more stable model.

The hyperparameters we tuned for this model are C (inverse of the regularizer) and n_estimators.

Naive Bayes

To explore generative classifiers we used Naive Bayes. Since this model requires discrete inputs, we applied bag-of-words on all of the features. While this created quite a large vocabulary, it ensured that all relevant data points were captured.

Model Choice and Hyperparameters

We selected the Logistic Regression model with ensembling based on its performance on the test set (Table 1). This comparison is valid since all models were trained, validated and tested on the same data splits.

Model	Test Accuracy
Random Forest	87.3%
Logistic Regression	89.7%
Logistic Regression with Ensembling	90.3%
MLP	83.1%
Naive Bayes	80.6%

Table 1: Test accuracy for each model

The hyperparameters we tuned for the logistic regression model with ensembling are the inverse of the regularizer (C) and n_estimators. We exported the weights from the scikit-learn Logistic Regression model.

First, we tuned n_estimators by setting C = 1.

C (1/λ)	n_estimators	Validation accuracy
1	1	84.6%
1	5	86.2%
1	10	86.2%
1	20	87.4%
1	30	87.2%
1	35	87.2%
1	55	87.0%

Table 2: Tuning n_estimators for an ensemble of logistic regression models

Then, we tuned the regularizer by setting n_estimators = 30.

C (1/λ)	n_estimators	Validation accuracy
0.1	30	84.0%
0.5	30	87.4%
1	30	87.2%
1.5	30	87.0%
2	30	87.0%
2.5	30	87.4%
3	30	87.4%

Table 3: Tuning C for an ensemble of logistic regression models

Based on the validation accuracy, we decided to set C = 2.5. However, we further tuned the n_estimators hyperparameter with C=2.5.

C (1/λ)	n_estimators	Validation accuracy
2.5	29	87.4%
2.5	30	87.4%
2.5	31	87.2%
2.5	32	87.6%
2.5	33	87.6%
2.5	34	87.6%
2.5	35	87.6%

Table 4: Tuning hyperparameters for an ensemble of logistic regression models

Thus, our final model has the hyperparameters C = 2.5 and n_estimators = 32 as they yielded the highest validation accuracy of 87.6%. We extracted the weights of the 32 estimators from the scikit-learn Logistic Regression model. Additionally, we use 160 features from the processed data, where each feature is one hot encoded. Thus, each weight matrix is of the shape (3, 161) since there are three classes (pizza, shawarma and sushi). Note that we did not use the features extracted from question 1 and question 2. To make a prediction, the model uses the Softmax activation function since this is a classification problem. The model then takes the average of 32 predictions.

Prediction

We expect our model to have a generalization accuracy of 90%, as the test accuracy is a good indicator of the model performance on the training set. Since the validation accuracy (87–88%) closely matched the test accuracy (90%), this suggests that the model is neither underfitting nor overfitting, and we expect similar generalization performance on unseen data. As the test data set consists of samples that are not seen by the model during learning, it provides a good estimate on how the model will generalize on new data.