Vickie Ip, Brendan Seto, Leonard Yoon
STAT 495: Advanced Data Analysis

Kaggle: What's Cooking?

Introduction

For our final project, we used recipes from the *What's Cooking?* Kaggle competition to predict the type of cuisine based on a list of ingredients. After converting the original recipes from JSON to CSV, we find that there are 39,774 rows in the dataset, and three columns (ID, cuisine, ingredients). Each row corresponds to an ingredient in a recipe. We first used One-Hot Encoding to transform the data such that rows became different recipes, and columns were binary indicators for each unique ingredient. This increased the size of our dataset to 6,715 columns. In this project, we explore decision trees (CART and Random Forest) with categorical variables. We grappled with maximizing categorization accuracy while reducing runtime by employing different data reduction strategies.

Figure 1: Word cloud of most popular ingredients in dataset (by frequency of occurrence)

Major Limitation: Running a Random Forest model using the entire dataset takes over two hours. Thus, we sought to reduce the dimensions of our data to lessen the runtime burden while improving our accuracy as much as possible.

Models:

We ran five different types of classification tree models: CART, Random Forest (RF) with no dimension reduction, RF with no identifying ingredients, RF with reduced number of predicted categories, and RF with simple string cleaning.

Model	Accuracy (% correct)	Runtime	Number of Variables Used
Original Random Forest	66.7%	>2 hours	6,715
CART	46.9%	10 minutes	6,715
RF: No Identifying	68.0%	~1.5 hours	4,115
RF: String Cleaning	69.4%	15 minutes	1,190

A. CART

We start with one decision tree that can have a maximum of 20 nodes (ideally one for each cuisine). After using *k* = 2-fold cross-validation to obtain an optimized complexity parameter α, we find that this rudimentary model can correctly classify about 47% of the pseudo-test set.

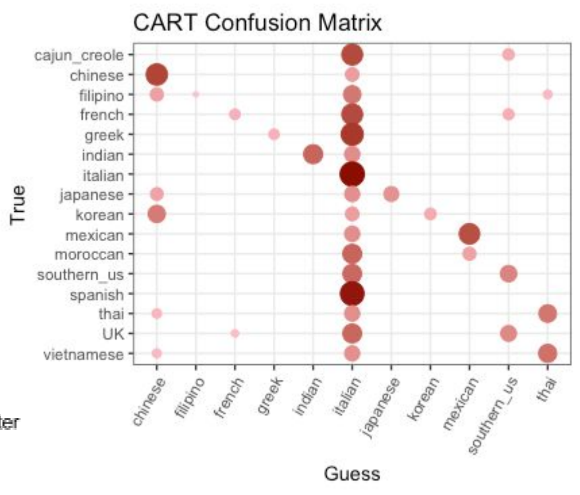

Figure 2: The red dots on the confusion matrix sum to 1 going from left to right. Darker and larger dots correspond to greater frequency of predictions by the model.

The accuracy is low because the model can only classify results into 11 cuisines. We also find that this model has a systematic overprediction of Italian cuisine.

B. Random Forest (baseline model)

Random Forest is a bootstrapping algorithm with a classification tree model. It builds multiple CART models with different samples and different initial variables. Our baseline model is a random forest with no dimension reduction.

1) Trim 1: No Identifying Ingredients

Some ingredients are only present in one cuisine. We termed these ingredients “identifying”, as their presence allows us to classify the recipe without having to create the model. They also allow us an opportunity to perform a pre-screening dimension reduction as we do not have to include these ingredients in our model.

As with all of our analyses, we did no manual filtering of the data. Identifying ingredients were determined through dplyr grouping methods (group_by ingredients and count cuisine) and then filtered. There are several limitations to this procedure. Most notably, just because an ingredient in our training set only shows up in one type of cuisine, we cannot say for certain that it is never used in other cultures. There is also the potential issue of very detailed ingredients (e.g. 6 oz Bologna Tomato paste) that may only appear in one cuisine by virtue of the fact that they were only used in a couple recipes total. While we acknowledge this limitation, we believe that it has little practical impact. If the ingredient is truly detailed beyond reason, it probably will not show up in the test set. Thus it will simply never be used. In this case, it is good that we have already removed it, as it is a waste to train on useless variables.

2) Trim 2: String Cleaning

The issue of ingredients detailed to the point of being useless motivated our second trim: a simple automated string cleaning. This method sought to group variables that are analogous in practice, if not in name. Examples include 6 oz tomato sauce and Barilla Traditional Tomato Sauce.

Kaggle Submission
Ultimately, we only made a Kaggle submission with our slimmest model (String cleaning trim), as we felt it was the only one that had some practical significance. Our score was significantly lower than our out-of-sample tests while training. We believe that there was a mix-up in Ids, or that there was a qualitative difference between the training and testing data that we cannot account for.

Conclusion:
We were able to make accurate out-of-sample predictions using our training data, but our success did not carry over to our Kaggle submissions. We were also able to significantly reduce runtime while slightly improving accuracy with dimension reducing techniques.