Whats Cooking: Capstone Project

Mallesh

January 5, 2016

Introduction

It is a well known fact that every cuisine has a distinct quality that distinguishes it from the others. For e.g., Indian foods are genrally known to be spicy, Middle eastern has a lot of ingredients etc. Yummly has provided a unique dataset with list of dishes, the ingredients used in preparing the dish and cuisine it belongs to. This gives a good insight into how ingredients are spread across several cusines and dishes. In the process, a classification model is build which will predict the cuisine based on ingredients in a test file.

Datasets can be found here:

What's Cooking

Approach

We shall go through the standard sequence of steps mentioned below to understand and build a successful model.

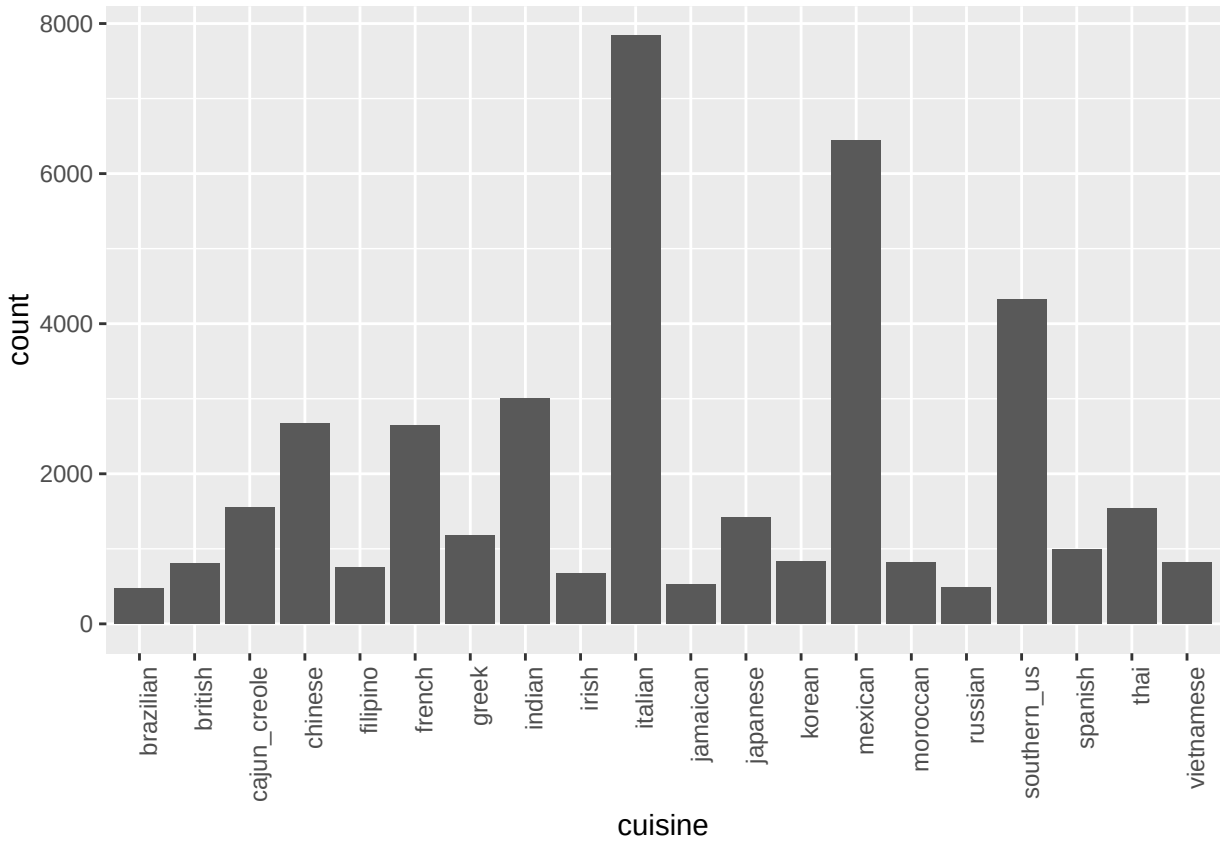
- Import the Json datasets and read them into a data frame.
- EDA to identify any patterns in the dataset.
- Data Wrangling and cleanup.
- Feature Engineering
- Model Building

Data Import and EDA::

After standard import and flattening the JSON file, we know that there are three attributes in training file, id - indicates a dish ID (from kaggle), cusine - indicates a type of cuisine and ingredients - a list of ingredients. In the testing file Cuisine attribute is excluded.

Also, looking at some of the values, it can be observed that there are several variations of same kind of ingredients. Which indiactes there is some level of cleaning to happen to be able to build a good model.

Let us look at the number of recipes in each cusine to see how the data is spread:



The graph indicates that there a lot of recipes in Italian. So, lets mark all the recipes in test file as Italian as a benchmark. Also, add indicators to both test and training datasets, so that we can combine and split them as needed.

Create a combined dataset , this would enable us to build the sparse matrix without the problem of having new variables from test dataset. These datasets would be spearated before building the model.

Data Wrangling and clean up:

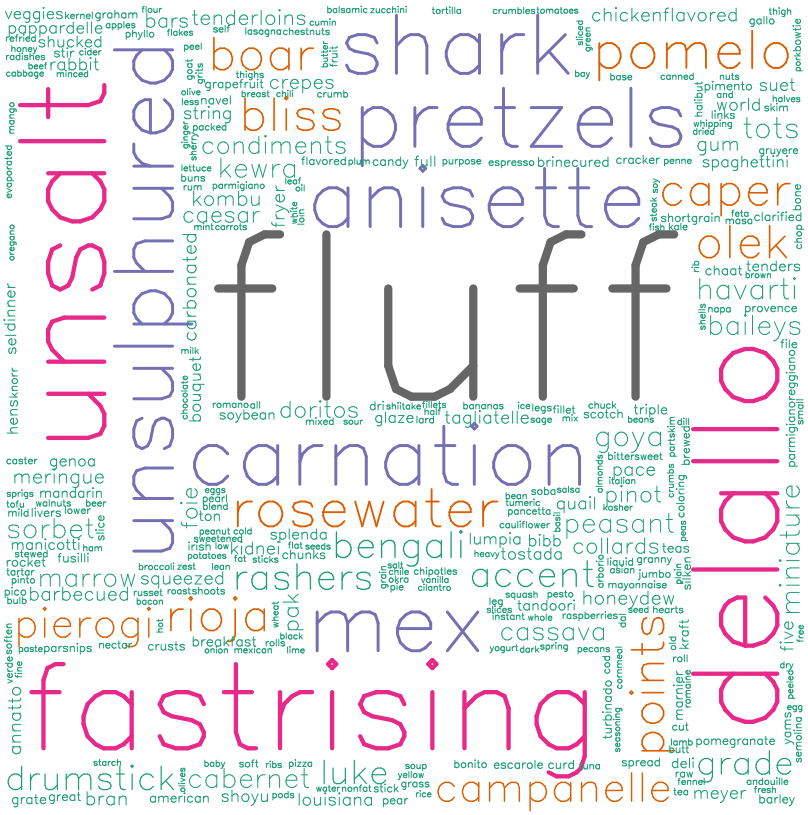
It is observed that some of the ingredients have measurements like 1 oz etc and some of the ingredients have punctuations like "."" and parentheses.

create a corpus of ingredients for further processing. Using TM package:

For doing further analysis, a bag of words is created which makes it simpler to clean-up the words. Following cleaning steps are applied in that order: -Stem the words -Remove the Punctuation -Remove the Numbers -Convert to lower case

Plot a word cloud with the least occuring ingredient names and most occuring ingredient names to ensure any bad data is cleaned.

Least Common Terms:



Most common terms:

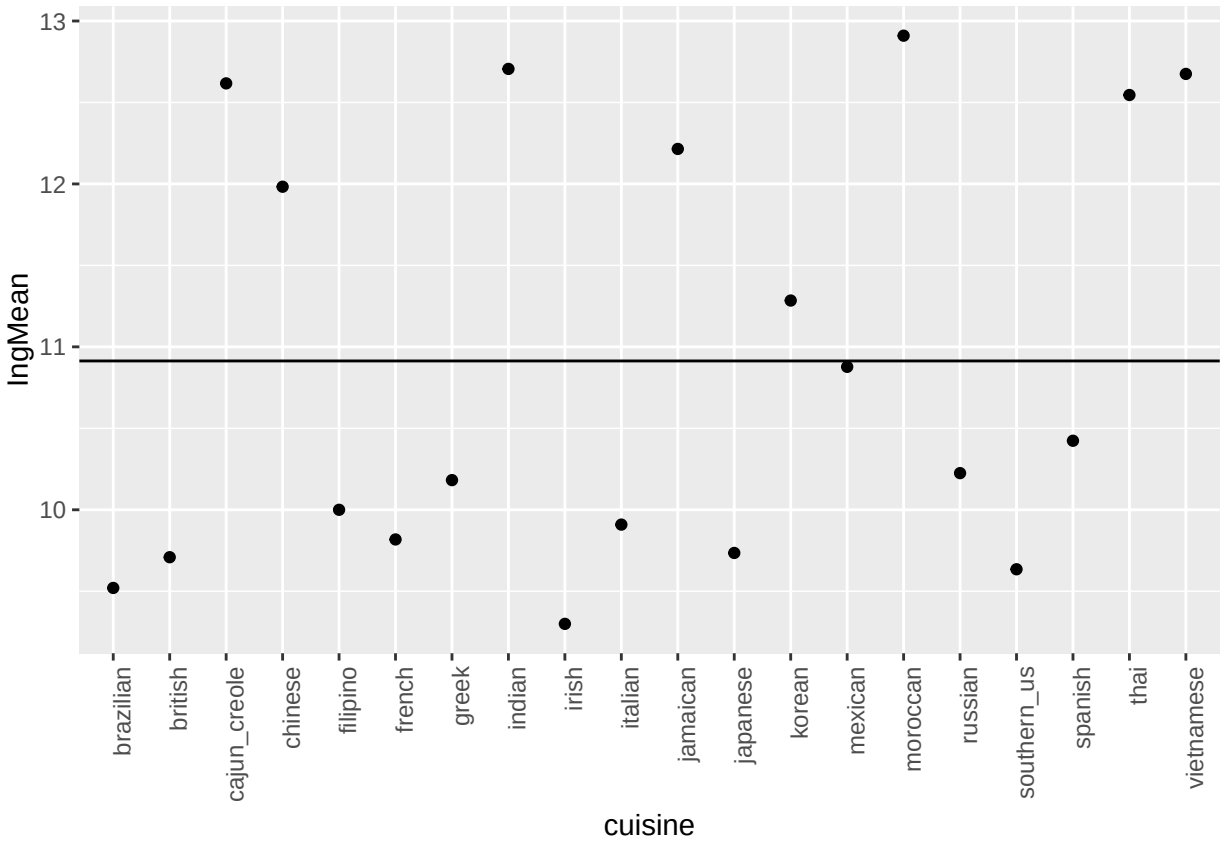
Based on the above word cloud, we can remove some of the terms that look like company names and measurements etc. List of words observed are:

“the”,“frozen”,“fresh”,“oz”,“reposado”,“medallion”,“foster”,“smithfield”,“simpli”,“home”,“zinfandel”,“reducedsodium”,“sand”,“
“dasti”, “torn”, “indian”, “lake”, “dash”,“razor”, “telem”,“delallo”, “torn”, “earl”,“for”, “zero”,“into”

As a next step, remove the sparse terms, i.e., include only those terms that have appeared atleast few 100 times in the corpus. We have come to this conclusion, since there are atleast 400 recipies for each cuisine and if an ingredient doesn’t appear atleast 25% of the time, then it wouldn’t be a good indicator of the cuisine. This would also remove any anomolies we might find that were missed during the manual step explained above.

Feature Engineering:

While some dishes are simple to prepare and some are difficult to prepare, the number of ingredients play a role in the complexity of the dish. Plotting the average number of ingredients across the cuisines:



The graph above indicates that the number of ingredients might play a role in the model evaluation. So, lets add an additional column to the training and testing datasets to include the number of ingredients in each of the dish.

Model Building:

4 different models were built using different algorithms:

In the order of their performance:

Decision trees: Using the rpart package in R, a multi classifier model is built to predict the cuisine based on the final set of ingredients. The result set submitted achieved a score of 0.40185.

Random Forest: After not so great score on Decision Trees, lets check if a Random Forest model performs any better. When the results are predicted based on a Random Forest model the prediction accuracy is around 0.47

SVM: Now, lets apply support vector machine SVM algorith to see of the success ratio improves. After building the model and checking for the ratio, its at 0.73592. Quite an improvement over the RandomForests.

XGBOOST: Finally, lets check if the performance improves with the Gradient Boosting. We need to build a datamatrix and convert the classifiers into Numericals as Gradient boosting doesn't support character classifiers. After rearranging the data and building an xgboost model, the success ratio is at 0.76368