What's Cooking?

Bilal Ahsan (18k-1133)

Faizan Ahmed (18k-0201)

Tahir Asif (18k-1169)

National University of Computer & Emerging Sciences

FAST Main Campus

Abstract:

In this paper, we consider different strategies for identifying the cuisine, given its ingredients. This project aims to explore what combination of ingredients is helpful in identifying a cuisine if the recipe is not given. This has been tackled as a problem of cuisine classification. We also explore different classification algorithms in tandem with approaches like taking combination of multiple ingredients for an exhaustive analysis of the results obtained.

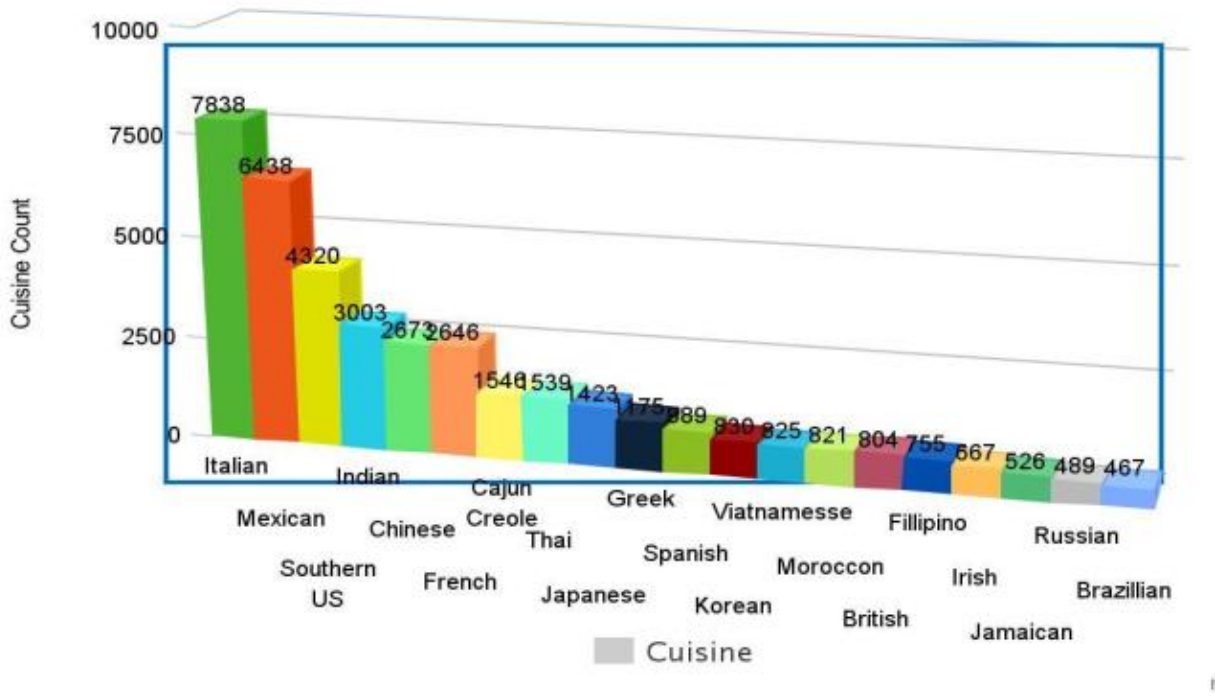
Keywords: Yummly, Cuisine, Ingredients, Classification.

Introduction:

All over the world, food recipes vary a lot even if they have the same ingredients. And different recipes belong to different cuisines. So, if someone is only given the ingredients, estimating a recipe is a problem that can be solved by looking at prior data (if there is sufficient data) but estimating a cuisine type (which is a superset of recipes, considering broader approach and wide variance in the use of ingredients) is not as easy. And what makes it even more interesting is, there could be many more versions of the same recipe [1]. However, we have followed the "standard cookbook" recipe approach for now, not considering all the variants at the same time.

After observing the dataset, we pondered upon the mining techniques that could be applied to this dataset. We explored the possibilities of application of regression, dimensionality reduction and found that only classification could be applied to this dataset. Since the data is limited for estimating cuisines based only on ingredients, we used some prior knowledge about recipes. E.g., an instance containing flour, butter and sugar would have a high probability of having eggs in it. This not only presents a pattern with respect to ingredient duplets and triplets but also opens up a lot of possibilities for exploration despite the size limitation of the dataset. Therefore, bigrams and trigrams (along with unigrams) of ingredients have been taken for classifying the cuisines.

While exploring the data, we could also consider a pattern amongst ingredients if we look at cooking techniques. E.g., for a dessert, there is almost always a requirement of sugar and flour for a particular cuisine. Due to limitation of time, we plan to leverage this factor in future.



Count of all cuisine in training set

We observe that Italian dishes dominate the charts. Therefore, we already have the baseline in which we predict Italian cuisine all the time.

Background:

Before training any machine learning algorithms, we first examined the data to get a feel for its general structure. The training file contains 39,774 entries described in 666,921 lines. The testing file contains 9,944 entries across 157,117 lines. A cursory glance at 5 random entries in the training and test file showed that the structure and contents of the files are similar, and that the ingredients list does not contain stop words. However, some of the ingredients have accents. We decided to normalize these words by turning them into their unaccented counterparts to prevent issues where one version of the word is accented and another is not.

Since the data was taken from a Kaggle Competition, not much cleaning was required. However, the problem of multi-class text classification required cleaning and adjustment of data according to our needs and we worked on a general framework for achieving that. Input and output of data was done through Pandas library. All the text was converted into lowercase.
Initial approaches involved the use of NLTK library and stemming the given words but it did not

affect the dataset much. So, this part was done only for the ingredient lists present in the dataset and not for cuisines as it would take up more time and not be much useful as there were only 20 unique cuisines. There was another interesting aspect of the problem – some ingredients have more than one words in their designation like ‘star anise’, ‘garam masala’, ‘cinnamon powder’ etc. Our standard data cleaning approaches posed a problem with respect to this as star anise was further reduced to two ingredients namely “star” and “anise”. This problem is also evident in figure 1 where ingredients like “fresh” and “ground” are present within the list of top 10 ingredients – they could have been a part of multiple ingredients like “ground pepper”, “ground black pepper”, “ground cumin” and others for the word “ground” & “fresh lemon juice”, “fresh cilantro”, “chopped fresh chives” for the word “fresh”.

Lastly, 100 out of these cuisine-ingredients’ pairings were duplicate which meant that there were a total of 39674 unique examples given in the training set.

Methods and Materials

Our main objective is to apply the visualization and modeling learned in Data Science. At first to explore data we visualized the data using matplotlib to understand the trends and numbers of the data. After demonstrating the data, we applied the supervise and unsupervised classifier to our data.

We implemented different modules to our data and checked their accuracy. Every module works on its own, and gives accuracy accordingly. We used following modules:

1. K Nearest Neighbors
2. Logistic Regression
3. Random Forrest
4. Decision Tree

1. **K-Nearest Neighbors**: measures the distance from an example to every other training example, identifies k smallest distances to each class and outputs class label based on the most represented class in these k classes. Since running this algorithm with sparse features becomes very time-consuming after a couple of trials, we did not implement this algorithm.

2. **Logistic Regression**: binary classification algorithm using sigmoid function, i.e., h (x) (θ x). By using one-vs-rest (OvR) scheme and cross-entropy loss, we are able θ = g T = 1 1 + e−θ x T to solve multi-class problems.

3. **Decision Tree Classifier**: non-parametric supervised learning method used for classification and regression; the goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

4. **Random Forest**: meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True (default)

Data and Results:

The dataset for these recipes has been obtained from a Kaggle Competition [2] “What’s cooking?” hosted by Yummly. It is a popular website and application which provides recipe recommendations tailored to the individual's preferences, semantic recipe search and a digital recipe box. It contains 39774 instances of cuisine and ingredient list pairings, in which each cuisine has several ingredients. There are a total of 2965 distinct ingredients in the entire training dataset for a total of 20 cuisines. A sample from the training set typically looks like

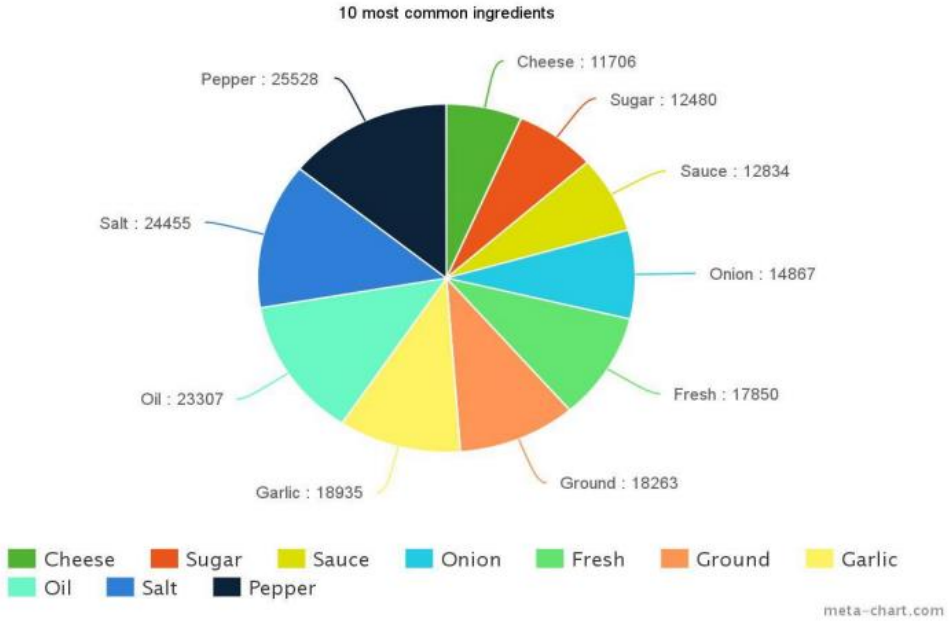
["id": 1, "cuisine": "greek", "ingredients": ["romaine lettuce", "black olives", "grape tomatoes"]]

The interpretation of the above-mentioned example is obvious - for some arbitrary greek recipe, the ingredients used were romaine lettuce, black olives and grape tomatoes. An additional test set of 9944 instances has been provided with the training data in which cuisine value is missing and the task at hand for the competition was to predict the cuisine for each example.

["id": 234, "ingredients": ["Cheese", "Tomato", "Karela"]]

The distribution of the ingredients and cuisines is presented in the following graphs: Figure 1: Counts of most common ingredients in training set data Figure 2: Counts of all cuisines in training set data We observe that Italian dishes dominate the charts. Therefore, we already have the baseline in which we predict Italian cuisine all the time.

The distribution of the ingredients and cuisines is presented in the following graphs:



Counts of most common ingredients in training set data

Conclusion

Confusion Matrix

Confusion Matrix we evaluated the classification accuracy by computing the confusion matrix. Each row corresponds to the true cuisine label. We normalized the results by dividing by the number of recipes for each cuisine in the test data. The diagonal elements represent the proportion of samples for each cuisine whose predicted label was equal to the true label, while off-diagonal elements were mislabeled by the classifier. In other words, the higher the diagonal values of the confusion matrix the better since this indicates a greater number of correct predictions.

```
[108] print('Confusion Matrix: \n',confusion_matrix(y_test, predictions))
[[  1  14   1   2   0  10   5   0  36   8   2   1   0   3   0   1  16   2
    0   0]
 [  3  18   8   9   4 131  47   9  23 811   1   5   2  34  12   7  49  29
    3   2]
 [  4   0   1   4   4   5   0   4   0   5  36   1   0   4   2   1   8   2
    1   0]
 [  1   2   0  30   5  11   1  11   5   2   3 119  19   3   1   1   9   1
    1   9]
 [  1   0   1  16   6   1   0   2   2   2   0  14  78   0   0   1   3   0
    4   2]
 [  4   8   9  10   6  36  15  12   7  53   1   4   3 789  15   2  59  11
    3   5]
 [  1   3   2   1   0  14   6  15   5  15   1   0   0  12  58   3   7   1
    0   0]
 [  1   1   0   0   1  13   3   1   8   8   0   1   0   7   0  21  15   2
    1   0]
 [  2  10  33  13   6  49   3   5  23  60   4   4   1  42   2   2 381  20
    3   1]
 [  3   4   2   2   0  12   5   1   5  29   1   2   2  13   2   0  16  37
    1   1]
 [  1   1   0  20   6   0   1  15   0   6   5   7  11  16   0   1   5   1
  144  27]
 [  1   1   0  13   7   0   0   3   0   4   0   6   5   4   0   1   6   2
   20  62]]
```

Summary of the findings

When we began this problem, we did not pre-process any of the data. Our initial results had accuracies ranging from 62% to 77%. By putting accurate values of n_neighbors, we managed to gain 3% to 4% accuracy in KNN.

Classifiers	Accuracy
KNN	60% - 65%
Logistic Regression	75% - 78%
Rondom Forrest	71% - 74%
Decision Tree	61% - 64%

Summary of our Classifiers

References

[1] References from Dan Jurafsky, Stanford - LINGUIST 62N The Language of Food
[2] Yummly-Kaggle. 2015. Kaggle – What's Cooking? (2015). Retrieved November 28, 2015 from https://www.kaggle.com/c/whats-cooking
[3] Stack exchange – StackOverflow, Cross Validated

[4] Kevin K. Do Department of Computer Science Duke University Durham, NC 27708 kevin.kydat.do@gmail.com.
[5] TSDAE: Using Transformer-based Sequential Denoising Auto-Encoder for Unsupervised Sentence Embedding Learning