Using NLP and machine learning to classify cuisine type of recipes based on their ingredients.

Sunella Fernando | 21052642 | UFCFMJ-15-M MLPA Coursework Assignment

Abstract; This study attempts to identify the type of cuisine of a recipe based on its ingredients. A recipe dataset from Kaggle, originally scraped from the aggregation website Yummly was used for training and testing (Kaggle, 2016). The scope was limited to the ingredient name; quantity and preparation methods were not considered.

Source code: <link>

Keywords; NLP, machine learning, food, cuisine, classification.

I. Introduction

Food is a basic human need. The accessibility of a variety of recipes on the internet from multiple different locations and cultures transcends barriers and opens avenues to explore the world for those without the means and ability to travel; it helps bring people together by way of shared experience.

Issues associated with the current search, submission, and curation process for recipe websites include:

- Submitting recipes to most popular online sources require filling in multiple fields manually. This can be discouraging as it might be difficult for amateur cooks submitting recipes to identify which type of cuisine their recipe is likely to align with, and even experienced authors might be put off by the number of input fields to fill in (Krug, 2014).
- Deciding what type of recipe to search for or what to cook considering the ingredients that are available is a familiar problem to many, as evidenced by the ever-increasing Google trends graph for relevant search terms (Google Trends, 2004-2022).
- Low quality recipes that are uploaded to the public domain without proper moderation or tagged with incorrect labels and cuisine types to get more 'views' reduces the quality of recipe aggregation sites, misleads target audiences and contributes towards food waste.

Using machine learning to automatically predict cuisine type could help mitigate some of these issues, by potentially simplifying recipe submission processes, helping decide what type of cuisine to prepare with ingredients at hand, identifying low quality 'clickbait' recipes, and ultimately improving the quality and variety of food experiences available to all.

II. EDA and Pre-processing

Some exploratory analysis was performed to better understand the dataset and identify what pre-processing steps would be required.

The Kaggle 'What's Cooking?' dataset (Kaggle, 2016) contains two separate json files for training (tagged) and testing (untagged). The tagged training dataset which consists of 39774 recipes from 20 types of cuisines was used in this work to focus on supervised learning methods. The dataset does not include any 'fusion' recipes (i.e., each recipe belongs to a single distinct cuisine type), which simplifies the problem, even though this may differ in real-world contexts.

The number of recipes belonging to each cuisine type is not equally distributed; Italian is most prevalent with 7838 instances, followed by Mexican (6438 recipes), with Russian and Brazilian cuisine being on the lower end with 489 and 467 recipes respectively (Fig-1).

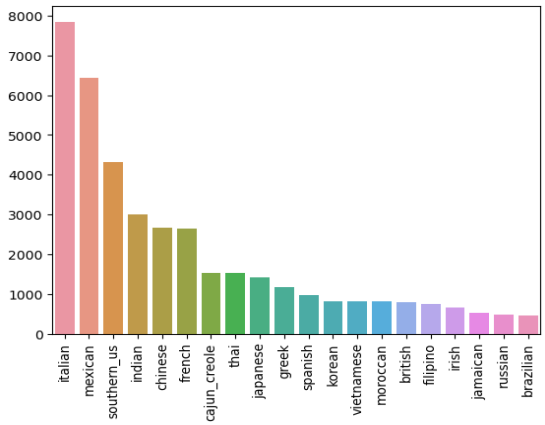

Figure 1- Cuisine counts

There are 6714 unique ingredients with the most common being salt (18049 occurrences), onions (7972), olive-oil (7972), and water (7457), and least common being more specialized ingredients with brand associations, specific cuts of meat or rare vegetables that occur only once such as lop-chong, kraft-cheese-crackers, lipton-iced-tea, and tongue. (Fig-2).

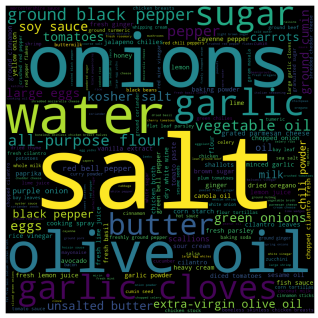

Figure 2- Ingredient word-cloud

An analysis was done to evaluate whether the number of ingredients could be used as a feature (i.e., whether there were any discernible patterns where, for example recipes of a certain type of cuisine were likely to have longer/shorter lists of ingredients than others), but the average number of ingredients appears to be similar for almost all cuisine types as evidenced in Table 1.

Cuisine	Average	Min	Max	SD
greek	10.18	1	27	3.72
southern_us	9.63	1	40	3.860
filipino	10	2	38	3.85
indian	12.71	1	49	5.01
jamaican	12.21	2	35	4.76
spanish	10.42	1	35	4.16
italian	9.91	1	65	3.80
mexican	10.88	1	52	4.65
chinese	11.98	2	38	4.04
british	9.71	2	30	4.16
thai	12.55	1	40	4.41
vietnamese	12.68	1	31	5.25
cajun_creole	12.62	2	31	4.61
brazilian	9.52	2	59	5.55
french	9.82	1	31	4.14
japanese	9.74	1	34	4.24
irish	9.3	2	27	3.70
korean	11.28	2	29	3.87
moroccan	12.91	2	31	4.79
russian	10.22	2	25	4.05

Table 1-Ingredient length distribution

It was observed that multiple cuisine types had occurrences of recipes with just one or two ingredients which might need to be removed. However, on further analysis, it was decided to retain such ingredients in the dataset since they might be relevant decision points as some ingredients appeared strongly correlated with certain types of cuisine (e.g., sushi-rice with Japanese cuisine).

Based on the learnings from EDA, some further pre-processing of the ingredient lists was done to clean the ingredient list and condense it into a form more suitable for analysis by a machine learning model (Brownlee, 2017), with the following steps:

- Converting all text into lowercase
- Removing leading and trailing whitespace
- Removing punctuation, numbers, and special characters
- Replacing plural words with the singular form (since quantities were not a feature)
- Lemmatizing using WordNetLemmatizer
- Vectorizing using TF-IDF algorithm.

TF-IDF was selected after comparing several other vectorization methods (Gupta, 2022) (Ragunathan, 2020). This algorithm measures how often a word occurs in the corpus (TF-term frequency) and multiplies it by the perceived importance of a term across the corpus (IDF-inverse document frequency) (Brownlee, 2017). TF-IDF is an unsupervised weightage algorithm, since it does not consider class information when deciding the weights (Carvalho & Paiva-Guedes, 2020). Applied to our context, this would assign a higher weighting for rare ingredients (which might be specific to a certain type of cuisine) and lower weighting for more frequent ingredients such as salt, water, onion (Fig-2). An unfortunate side effect might be bias towards ingredients frequently used in cuisines that have a stronger presence in the dataset (i.e., ingredients common to Italian cuisine being weighted lower than those common to Brazilian cuisine), however existing research suggests that adjusting weight according to the popularity of the cuisine does not significantly improve prediction results (Li & Yang, 2020).

III. Model Selection

The dataset is already labelled with a single cuisine type per recipe, making this a univariate supervised learning problem. A selection of algorithms from sckit-learn were considered (Pedregosa, et al., 2011). The expectation was to compare and evaluate scores, so models that might not be expected to perform well were also applied.

K-Nearest-Neighbor (KNN)

KNN uses a distance measure to calculate similarity between pre-classified data points and votes on where the new instance is likely to fit in based on a given 'k' number of neighbors (close data points). KNN is a 'lazy' algorithm and is comparatively fast when handling multi-class problems, however, is more vulnerable to the curse of dimensionality (scikit-learn, n.d.).

Multiclass Logistic Regression

Logistic regression uses a sigmoid function to model a linear relationship between the input and output and is generally used for text classification due to its simplicity to interpret, and low potential for overfitting. It is sensitive to feature-scaling and multicollinearity; however, this can be mitigated in the context of this problem by pre-processing and vectorization using TF-IDF (Asar, 2017). The scikit-learn algorithm for logistic regression applies regularization by default, which makes it less prone to overfitting for a large high-dimensional dataset such as this (scikit-learn, n.d.).

Naïve Bayes

Naïve Bayes is a linear classifier based on Bayes Theorem, that assumes that the features are independent (hence, 'naïve'). The algorithm is fast, is less likely to suffer from the 'curse of dimensionality', and is computationally less expensive to run, making it a popular choice for classification problems, even though it is only applicable on linearly separable data, and the underlying assumption (naivete) can be seen as unrealistic (Ng, n.d.). Multinomial naïve bayes is chosen rather than the Gaussian version due to its proven results with classification problems based on fractional counts such as TF-IDF (Chavez & Heffernan, 2020).

Random Forest

Random forest is an ensemble learning algorithm, which uses the output from multiple decision trees (which are simplistic models that make 'decision rules' to split the data). Random forests are known for low potential to overfit, generally high accuracy even though the training process can be time-consuming and computationally expensive when parallel processing is not available. Parameter tuning is not required but the model can be difficult to interpret and can be biased towards classes that occur more frequently (i.e., Italian cuisine in the context of our dataset) (Liberman, 2017).

Deep Learning (rejected)

Deep learning is vastly popular but 'over-hyped' by some accounts due to its application in multiple unrelated fields regardless of suitability (Leetaru, 2018). Deep learning requires informed tuning and is computationally very expensive in terms

of resource consumption (computing power) and is not open to interpretation at all (Cordero, 2017). It is a collection of methods rather than one single algorithm, none of which were applied on this dataset.

IV. Evaluation and Comparison

V. Conclusion

This work proposes a supervised learning approach to automatically tag the cuisine type of a recipe based on its ingredients. Future work might consider a different vectorization algorithm, further parameter tuning, using PCA to reduce the curse of dimensionality introduced after vectorization, including the quantities as a weight, considering other parameters such as instructions and cooking time.

A potential limitation is that the human-tagged recipe dataset used for training might be biased, labelled incorrectly and subject to issues outlined in the introduction. However, there is no straightforward way to overcome this limitation; to end with the words of Sir. Terry Pratchett “Real stupidity beats artificial intelligence every time” (Pratchett, 1996).

Word count -

VI. References