Classifying Recipes into Cuisines using Context Vectors of Ingredients*

Nishant Mohan
Trinity College Dublin
mohanni@tcd.ie

Lujain Dweikat
Trinity College Dublin
dweikatl@tcd.ie

Rohan Bagwe
Trinity College Dublin
bagwer@tcd.ie

Aakash Kamble
Trinity College Dublin
kamblea@tcd.ie

Oommen Kuruvilla
Trinity College Dublin
kuruvilo@tcd.ie

April 14, 2020

Abstract

For classification problems, simple machine learning algorithms such as Logistic Regression have been extensively used with a numerical data. However, the current challenge is to convert non-ordinal textual data to a numerical data in a meaningful way that preserves semantics of the text. This paper presents a novel method to convert ingredients of a recipe into context vectors using Gensim's implementation of Word2Vec. The classification accuracy using the context vectors of ingredients was found to be 65%. This failed to support our hypothesis that the context vectors of ingredients would generate better results as compared to the traditional approach which is based on the Bag-of-Words model with 78% classification accuracy.

***Keywords*—** Text Analytics, Recipes, Cuisines, Context Vectors, Word2Vec, Classification

1 Introduction

With an introduction of the internet, people have been increasingly sharing food recipes on the web-blogs. This has caused a lot of websites to host these recipes and allow people to share, search and review the same with an ease. It has been observed that how world's cuisines have originated and developed over time is fascinating. Being able to label a recipe with a cuisine makes easier to find the recipes manually on the internet. For search engines, labelling helps in finding recipes from the same cuisine in a faster way. Additionally, such websites would probably be more beneficial to the user if they are able to recommend similar recipes from the same cuisine.

Classification is not a new problem. However, simple classification algorithms usually work best with numerical data. There are plethora of methods developed to transform textual data into numerical data. However, it has been observed that the semantics of the text is lost when converting textual information into the numerical space. Therefore, a novel method is required to preserve the meaning of the words used, which also creates consistent numerical sets that are appropriate for the machine learning models.

Kumar et. al.[1] have attempted to predict cuisines using the same data as in the proposed study. The method implements XGBoost,[2] a gradient boosting library and Random Forest[3] algorithms in order to achieve classification with 80% accuracy. The motive of this work is to investigate if representing ingredients of a recipe as word-embeddings aids in improving classification accuracy as compared to the traditional frequentist approach.

This paper presents an overview of traditional cuisine classification techniques which is part of converting textual data into numerical vectors. The paper depicts few concepts related to context vectors and word embedding.

*__Github Repository: Code, Data, MoM__ - https://github.com/mohannishant6/Recipe-Ingredients-as-Word-Embeddings

1. R. Kumar, M. Kumar, and Soman Kp, "Cuisine Prediction based on Ingredients using Tree Boosting Algorithms," *Indian Journal of Science and Technology* 9 (December 2016), doi:10.17485/ijst/2016/v9i45/106484.

2. *XGBoost Documentation*, https://xgboost.readthedocs.io/en/latest/.

3. *Random Forests Leo Breiman and Adele Cutler*, https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm.

The paper also illustrates previous attempts of classifying recipes into cuisines. The remaining paper explains features of the dataset used, baseline for the evaluation and the proposed algorithm to improve the classification accuracy. The paper concludes with the discussion of results and the future improvements.

2 Literature Review

It is obvious that a machine does not understand natural language the way humans do. Although advanced computing technologies allow the end-user to ignore this fact, at the lowest levels of implementation, a computer program or an algorithm deals with numeric data. Thus, an embedding is the numerical representation of words, sentences or corpus. A Word Embedding format generally tries to map a word using a dictionary to a vector of numbers. A dictionary may be the list of all the unique words in a sentence or a document, also known as Bag-of-Words (BOW). The corresponding vector representation of the word will be all zeros except one unique index from the Bag of Words for each word. This technique is called one-hot encoding.

Although word-embeddings created using bag-of-words are popular, they have two major weaknesses: - BOW models do not measure the relationship between different words (location information of a word) - BOW models generate high dimensional sparse matrix that requires heavy computation and storage resources. Since BOW removes order of words, heterogeneous sentences may have same vector representation if they consist of same words. For example, from our knowledge we know that “tea” should be closer to “milk” than “fish”, but “tea,” “milk” and “fish” are equally distant in vector representation. Thus, representing words using a simple BOW model leads to substantial data sparsity creating a need for a more sophisticated way of representing a word considering its context with other words. The method can be useful in some cases, however, for more complex language processing tasks the numerical representation loses the order of words and does not preserve semantics.

Context Vector (CV)[4] is a type of word-embedding and a method of representing textual data. The concept was first introduced by Stephen Gallant in 1998. The author suggests that these vectors should have four goals in mind. Firstly, the vectors need to convey similarity of use between two different words. For example, the more similar two terms such as “car” and “auto” have the bigger the dot product, and vice versa. Secondly, context vectors need to provide a quick way of searching and verifying a word that relates to the contents of a document. Thirdly, the method should enable the user to use text data in a machine learning algorithm. Finally, context vectors should be feasible to create for new unlabeled text.

Context Vectors have a few shortcomings even though they achieve the above-mentioned goals. For instance, the syntax is not represented in CVs, therefore no discourse or logic can be concluded. Kumar

et. al suggests that using phrasal CVs and adding a second run of CVs could possibly be used to represent syntax. However, this study does not require to represent a syntax, since a recipe is just a list of ingredients and not a coherent sentence.

More modern forms of word embedding usually fall into one of the following categories:[5]

- Frequentist methods: Count Vectors, TF-IDF Vectors and Co-occurrence matrix
- Predictive methods: Continuous Bag of words (CBOW) and Skip-Gram models

Word2Vec[6] was first introduced by google in 2013. This embedding is a predictive method that uses the skip-gram model and the CBOW model to produce a distributed representation of words. The Word2Vec model evaluates texts in a sliding window methodology. The model presumes that every word has two vectors that are associated with it and tries to fit both these vectors based on the principle of maximum-likelihood. It can be trained on any custom corpus or a pre-trained model trained on a large corpus, such as Wikipedia, could be used to retrieve vectors for words. This generates a vector representation of words by taking into consideration its context.

GloVe[7] is another unsupervised learning algorithm created by researchers at Stanford University. It is trained on aggregated global word-word co-occurrence statistics from a corpus. Its main features include finding the nearest neighbor of a a word, for instance - frog’s nearest neighbor could be the word toad, and finding linear substructures that allow arithmetic operations on those vectors to conclude concepts like:

```
king - man + woman = queen
```

4. Stephen I. Gallant, “Context Vectors: A Step Toward a “Grand Unified Representation”,” in *Hybrid Neural Systems, Revised Papers from a Workshop* (Berlin, Heidelberg: Springer-Verlag, 1998), 204–210, ISBN: 3540673059.

5. Amit Mandelbaum and Adi Shalev, “Word Embeddings and Their Use In Sentence Classification Tasks,” October 2016,

6. Tomas Mikolov et al., “Distributed Representations of Words and Phrases and their Compositionality,” *CoRR* abs/1310.4546 (2013), arXiv: 1310.4546, http://arxiv.org/abs/1310.4546.

7. Jeffrey Pennington, Richard Socher, and Christopher D. Manning, “GloVe: Global Vectors for Word Representation,” in *Empirical Methods in Natural Language Processing (EMNLP)* (2014), 1532–1543, http://www.aclweb.org/anthology/D14-1162.

The end result of both GloVe and Word2Vec is a vector representation of a word. However, GloVe is not based on a neural network. Another difference is the fact that Word2Vec produces context-predicting vectors, while GloVe produces context-counting vectors. It has been previously shown[8] that context-predicting vectors are better than context-counting vectors. Based on this, we will be using Word2Vec for our experiment.

The dataset was obtained from a Kaggle competition, as such, extensive research has been conducted to predict a cuisine based on a recipe's ingredients as accurately as possible. Kumar[9] et. al. merged all ingredients into a single corpus. Then all words in the corpus are converted to lowercase letters. The words in the corpus are cleaned by removing punctuation, stop words, white spaces and converted to the lower case. This altered and cleaned corpus was then fed as a frequency matrix and classify using two algorithms, namely XGBoost[10] and Random Forests.[11] The former yielded a classification accuracy of 80.4% while the latter yielded an accuracy of 76.4%.

Thus as we see, many attempts have been made at prediction using classical models such as boosting trees and logistic regression. However, word-embeddings have still not been used to the best of our knowledge. This research problem aims to study the prediction capabilities of word-embeddings for recipes.

3 Methodology

3.1 Dataset

The corpus used in the study was obtained from Kaggle,[12] in partnership with Yummly.[13] Each recipe item has a unique ID, a single cuisine, and a list of ingredients.

While analysing the exploratory data, we found that the corpus contains 39774 recipes, each belonging to one of the following cuisines: French, Mexican, Korean, Spanish, Chinese, Indian, Cajun Creole, Filipino, Brazilian, Italian, Irish, Moroccan, Southern US, Japanese, Russian, Jamaican, Greek, Thai, British, or Vietnamese. Overall, the dataset contained 6714 unique ingredients.

Cuisines were not represented equally in the training data. Figure 1 shows that Italian recipes were over-represented, comprising 19.7% of the dataset, while the training set had low composition Brazilian recipes, at around 1.17%.

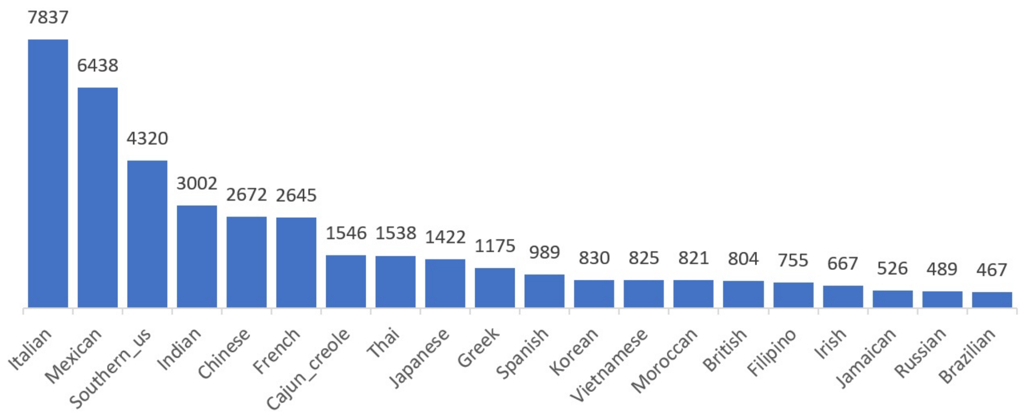

Figure 1: Recipe Count per Cuisine in Dataset

3.2 Establishing A Baseline

To compare the classification performance of context vectors of ingredients a baseline is needed. We chose traditional frequentest approach, BOW as our baseline. We use simple Logistic Regression[14] provided by scikit-

8. Marco Baroni, Georgiana Dinu, and Germán Kruszewski, "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors," vol. 1 (June 2014), 238–247, doi:`10.3115/v1/P14-1023`.
9. Kumar, Kumar, and Kp, "Cuisine Prediction based on Ingredients using Tree Boosting Algorithms."
10. *XGBoost Documentation*.
11. *Random Forests Leo Breiman and Adele Cutler*.
12. *What's Cooking?*, 2016, `https://www.kaggle.com/c/whats-cooking`.
13. *Personalized Recipe Recommendations and Search*, `https://www.yummly.co.uk/`.
14. Andriy Burkov, *The hundred-page machine learning book* (Andriy Burkov, 2019).

learn[15] for classification.

We used the bag-of-words model to convert the ingredients into numerical data. We derived a dataset with 6714 columns that count the frequency of occurrence of each ingredient in a recipe. This dataset was then split into 85% for training and 15% for testing. The classification accuracy for this method was found to be 78%.

3.3 Cleaning Up The Data

The proposed algorithm first cleans up the dataset. Initially, all the recipes were parsed individually at a time. All non-alphanumeric characters were removed for each ingredient. Also, most common words in the corpus such as large, salt, water, cold, etc, English stop words, and typographical errors were removed.[16] NLTK[17] has been implemented in order to get the singular form of nouns, then get the parts of speech of a word, and remove any words that are not nouns. This process reduced the total number of unique ingredients by 1729 from 6714 to 4985.

A vector has been created for each recipe using Gensim's[18] Word2Vec implementation. 300 features were chosen from the function, and we used a 10 unit wide window. Thus, a set of CVs containing one 300-length vector for each recipe is obtained.

3.4 Model Training

Using scikit-learn's,[19] we randomly split the data frame into two parts. The training set makes 85% of the original set, while the testing set is comprised of 15% of the original data frame. The split ratio is kept same as in the baseline model.

Next, scikit-learn's[20] Logistic Regression model is fitted using the clean data. The resulting model is used to classify the remaining validation set.

4 Results and Discussion

We evaluated the cosine similarity for the similar ingredients after training the proposed corpus. We used the ingredients ”ginger root” and ”ginger”, which results in a 0.9526417 similarity, indicating that the model trained is behaving correctly.

Figure 2 depicts the context vectors where 300 dimensions were transformed to 2D dimensional space. In the visualization, CVs are converging and forming different clusters and each of these clusters are based on different cuisines. The visualization shows accumulation of similar cuisines, for example, the lower left of the Figure 2 shows the accumulation of Chinese, Thai, Japanese and Vietnamese cuisines which are all Asian cuisine. This is due to the large similarity in ingredients used in all the Asian dishes. We see that there is a large overlap of different cuisines which may affect our classification accuracy.

Logistic regression yielded 65.35% accuracy on the test data. This is lower than the baseline which uses bag of words, wherein the classification accuracy was 78%. In the following section we discuss some possible explanations for the drop in accuracy.

Exploratory analysis of the text corpus reveals an imbalance in the dataset with an over-representation of Italian and Mexican recipes. These cuisines have lower misclassification rates:

Mexican	11.53	Southern US	31.81	French	58.56	Irish	81.41
Indian	13.26	Cajun Creole	43.92	Filipino	64.64	Russian	84.87
Italian	15.69	Moroccan	45.31	Korean	66.63	Spanish	93.02
Chinese	20.77	Japanese	52.39	Jamaican	67.3	British	94.28
Thai	26.79	Greek	52.68	Vietnamese	72.24	Braziallian	94.86

Table 1: Misclassification rates per cuisine

15. *scikit-learn: Machine Learning in Python*, `https://scikit-learn.org/stable/`.
16. nightowl21, *nightowl21/Extra-Projects*, `https://github.com/nightowl21/Extra-Projects/tree/master/Yummly-Kaggle`.
17. *Natural Language Toolkit*, `https://www.nltk.org/`.
18. *Gensim: Topic Modelling for Humans*, `https://radimrehurek.com/gensim/`.
19. *scikit-learn: Machine Learning in Python*.
20. Ibid.

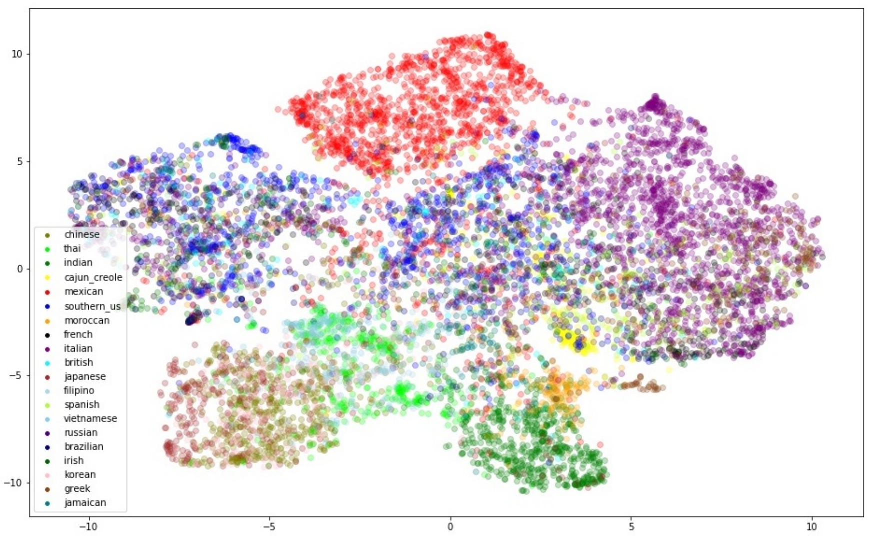

Figure 2: 2D visualization of a recipe context vectors

On comparing Figure 1 and Table 1, it can be observed that we obtained the lowest misclassification rates for the three most represented cuisines, while the least represented cuisine got the highest misrepresentation score. This can be visually represented using the confusion matrix shown in Figure 3, where cuisines on both axes have been ordered by the percentage of representation in the dataset.This indicates that there is a fundamental problem with the lack of representation of other cuisines in the dataset. The problem could potentially be solved using a diverse dataset.

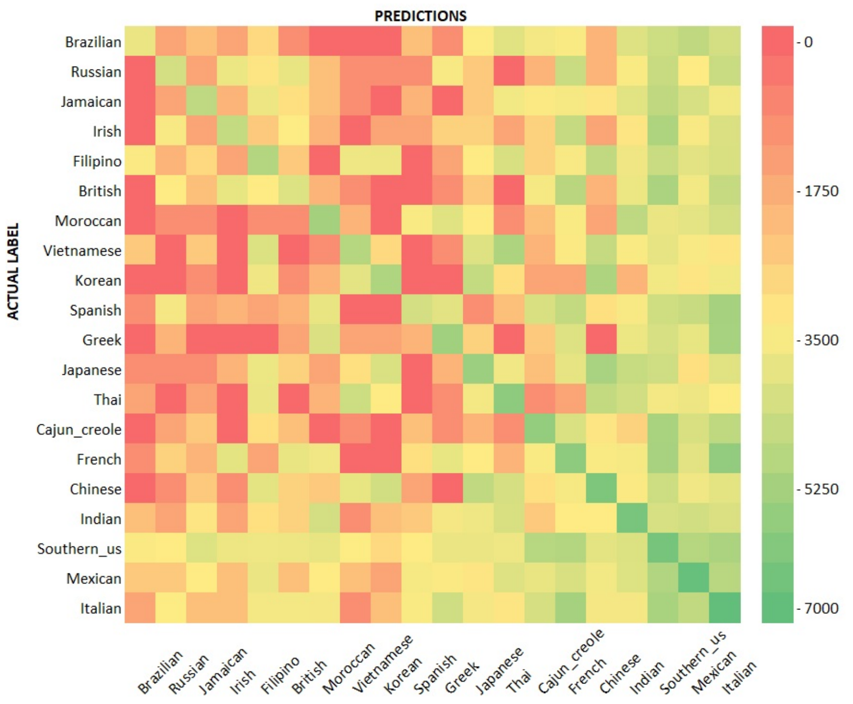

Figure 3: Confusion matrix for classification using Word2Vec

The product of the scaled misclassification rate and the scaled representation was taken to understand which cuisines were classfied correctly, despite their level of representation in the original dataset.This approach rank French cuisine as the best candidate for further analysis. It has been observed that the recipes which were actually French, were misclassified as Italian the most. This is because Italian cuisine has highest presence ratio in the data. However,the notable observation is the Italian cuisine were misrepresented as French cuisine the most. This was surprising considering French cuisine was not among those having high presence ratio in the dataset. Also, Italian was not misidentified as Irish cuisine in high numbers, but French was. We, therefore, took a closer look at the top ingredients of these three cuisines.

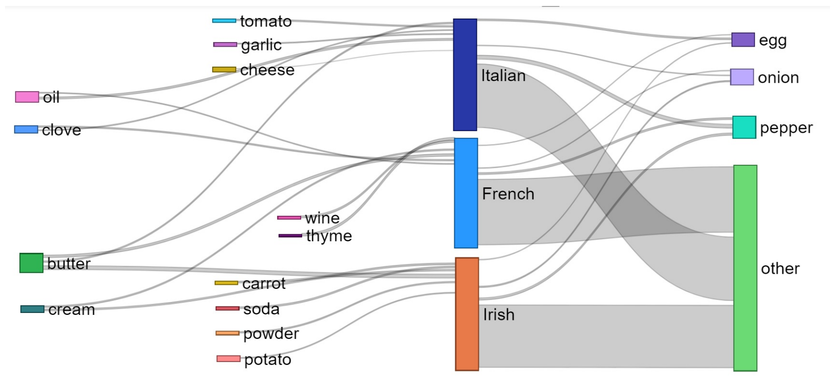

Figure 4: Top 10 ingredients w.r.t cuisines

Figure 4 depicts the most common ingredients in the three cuisines under scrutiny. It can be observed that French cuisine shares two ingredients, oil and clove, with Italian cuisines. French cuisine also shares two main ingredients, cream and butter with Irish cuisine. Interestingly, Italian cuisine and Irish cuisine do not share many ingredients other than the ones shared by all three cuisines, which are pepper, onion and others. This further strengthens our observation that French cuisine has similarities with Italian and Irish cuisine, however, Italian and Irish cuisine are not as similar. Therefore, misclassification among Italian and French cuisine is natural, since they closely resemble each other based on the ingredients used in their recipes.

5 Conclusion

Although recipes and recipe books have existed for a very long time, the widespread use of the internet started appearing recipes on the web. Aggregated recipe-book simulates the websites with a variety of cuisines. Knowing the cuisine a recipe belongs to could be helpful for recipe recommendations and browsing.

For this experiment, we established a baseline using Logistic Regression,[21] as it is the simplest form of classification algorithms with no other variable to be controlled. Using bag-of-words, the baseline resulted in 78% classification accuracy.

We have used context vector representations of the recipes created using Gensim's[22] implementation of Word2Vec[23] after carrying out some data cleaning tasks. After classifying the data using Logistic Regression again, we got an accuracy rate of 65.35%, which is less than the baseline accuracy. After analysing the dataset, we found out that the cuisines in the dataset do not have similar representation ratios. This has lead to having higher accuracy rates for the top four most represented cuisines than the overall accuracy rate.We have also concluded that some cases of misclassification might be due to actual similarities between two cuisines, as we establish that Italian and French cuisines are similar, French and Irish cuisines are similar, however, Italian and Irish cuisines are as not similar, causing a lot of misclassifications between French and Italian cuisines.

6 Future Work

This research has been mainly focused on the use of context vectors of ingredients for classifying cuisines. Although the provided analysis and methodology failed to capture deeper relations between ingredients given the context in which they are used, some improvements can still be made. This section briefly describes some interesting research topics, which are worth investigating further :

1. Improvement in building context vectors: Word2Vec algorithm builds a deep neural network to predict a word based on its context. The size of the context is defined by the window size. Thus, if the window size is 2, the context of each world would be defined by the two words that are neighboring it. Our analysis uses the context window of size 10 which may not be big enough to yield better results. Also, the dimensionlity of the context

21. Burkov, *The hundred-page machine learning book.*
22. *Gensim: Topic Modelling for Humans.*
23. Mikolov et al., "Distributed Representations of Words and Phrases and their Compositionality."

vectors can be reduced to make them more compact. Different word-embeddings can be used to generate context vectors of ingredients, for example, Doc2Vec, Glove[], Gloss Vectors.

2. Balancing the output classes: Since there are bias issues in our dataset, for example, the non-uniform distribution of the types of cuisines. Getting more data to balance the classes could make improvements in the accuracy of the classifier. Also, new sources of data such as allrecipes.com could be explored if getting more data is not time consuming or expensive.

3. Refined choices of ingredients: It has been observed that despite performing data cleaning, there still exist some data leakages. Thus, the use of advanced NLP techniques that recognize the entity of a word and calculate its probability to be an ingredient should be explored.

References

Baroni, Marco, Georgiana Dinu, and Germán Kruszewski. "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors," 1:238–247. June 2014. doi:`10.3115/v1/P14-1023`.

Burkov, Andriy. *The hundred-page machine learning book.* Andriy Burkov, 2019.

Gallant, Stephen I. "Context Vectors: A Step Toward a "Grand Unified Representation"." In *Hybrid Neural Systems, Revised Papers from a Workshop*, 204–210. Berlin, Heidelberg: Springer-Verlag, 1998. ISBN: 3540673059.

What's Cooking?, 2016. `https://www.kaggle.com/c/whats-cooking`.

Kumar, R., M. Kumar, and Soman Kp. "Cuisine Prediction based on Ingredients using Tree Boosting Algorithms." *Indian Journal of Science and Technology* 9 (December 2016). doi:`10.17485/ijst/2016/v9i45/106484`.

Mandelbaum, Amit, and Adi Shalev. "Word Embeddings and Their Use In Sentence Classification Tasks," October 2016.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. "Distributed Representations of Words and Phrases and their Compositionality." *CoRR* abs/1310.4546 (2013). arXiv: `1310.4546`. `http://arxiv.org/abs/1310.4546`.

Natural Language Toolkit. `https://www.nltk.org/`.

nightowl21. *nightowl21/Extra-Projects.* `https://github.com/nightowl21/Extra-Projects/tree/master/Yummly-Kaggle`.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation." In *Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. 2014. `http://www.aclweb.org/anthology/D14-1162`.

Gensim: Topic Modelling for Humans. `https://radimrehurek.com/gensim/`.

Random Forests Leo Breiman and Adele Cutler. `https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm`.

scikit-learn: Machine Learning in Python. `https://scikit-learn.org/stable/`.

XGBoost Documentation. `https://xgboost.readthedocs.io/en/latest/`.

Personalized Recipe Recommendations and Search. `https://www.yummly.co.uk/`.