NTDS Project

Strategic Recipes

January 10, 2020

Authors:
Maël Wildi, Karen Preitner, Raphael Strebel, Elias Gajo

Network Tour of Data Science
Ecole Polytechnique Fédérale de Lausanne

1 Introduction

In recent years, the public has been more and more preoccupied by the environment, participating to climate strikes and changing their consumption habits.
Unfortunately, food waste remains an enormous problem in our society. Indeed, we still waste millions of tons of food each year. For example, in Switzerland, it goes up to 2.8 million of tons per year, i.e. 330kg per capita per year.
Our approach is to reduce the food waste from the consumer side. Indeed, up to one third of the waste come from the consumers. We decided to create a system that would suggest different recipes depending on the ingredients that the consumer has in his fridge. We want to maximize the number of ingredients that people already have at home, while minimizing the number of new ingredients that the consumer would have to purchase.

2 Getting the Data

The dataset used for this project is "Cooking recipes", available on [2]. It contains cooking recipes from various websites. The recipes are all HTML documents, but as they are from different websites, the HTML page architecture also changes, which increases the complexity of the scraping. Not all the documents are recipes, so we have to filter these pages during the scraping.
As the recipes come from different websites, we begin by grouping them (the HTML documents) by websites. After that, we remark that the 3 biggest website contain more than 50% of the recipes in the dataset. We choose to keep only the recipes belonging to these 3 biggest websites. This choice greatly reduced the scraping efforts and allow to obtain a dataset with more than 44 000 recipes. Note that we later only keep the first 2000 recipes, as the computational time to build the graph is too demanding on the entire dataset.

At this point we have a dataset of recipes of the format in Figure 1 below :

	Website	Recipe	Prepare time	Ranking	Reviews	Ingredients
0	food.com	Ecuadorean Quinoa and Vegetable Soup	75	4.86	31	[quinoa, olive oil, onions, salt, potato, red ...
1	food.com	Authentic Injera (aka Ethiopian Flat Bread)	4330	3.13	17	[teff, water, salt]
2	food.com	Healthy Vegan Coleslaw	10	5	1	[cabbage, vegan mayonnaise, apple cider vinega...
3	food.com	Grilled Flatbread	35	4.67	14	[active yeast, olive oil, flour, salt]
4	food.com	Baked Margarita Pie	20	5	3	[cracker, milk]

Figure 1: Recipes dataset

Since we focus on a recipe's and its ingredients, we will disregard the other columns. We also remove any recipe that has no ingredients, as well as recipes with the same list of ingredients as another.

3 Building the Graph

Once we have the cleaned data, we can start building the graph. To do so, we compare the ingredients of all pairs of recipes. There are multiple weighting functions that we could use, but since we deal with sets of ingredients, one popular choice is the Jaccard similarity [3]. This weighting method computes the size of the intersection over the size of the union between the two sets of ingredients.
When building the graph, we add an edge between two recipes if their Jaccard similarity is strictly greater than 0. We could threshold this value since there are around 2000 nodes and potentially 4 million edges, but then the clustering method might output different results.
A second weighting method we try is the following. We give each ingredient a weight of $w_i = \frac{1}{K_i}$ where K_i is the number of times ingredient i appears in a recipe. Then we modify the Jaccard similarity to output the sum of the weights of ingredients in the intersection of two recipes, divided by the sum of the weights of ingredients in their union. This gives a measure which takes into account the *importance* of an ingredient, in the sense that it more frequently used ingredients have smaller weight. This way we avoid high similarities between recipes that use for example sugar or salt in a small set of otherwise unrelated ingredients.

4 Exploration

4.1 Properties

Now, that we have constructed our graph, we can analyze its properties :

```
Graph properties :
 Nodes : 28136
 Edges : 1994
 Average clustering coefficient 0.40
 Diameter : infinity (there is some isolated nodes)
```

Figure 2: Different properties of the graph

We recognize from the degree distribution on Figure 3 the power-law curve, meaning that our graph is scale-free : indeed, there are multiple small-degree recipes, while only few hubs are present. Also, there are few nodes with a degree of $<k> = 28$ (the average degree).

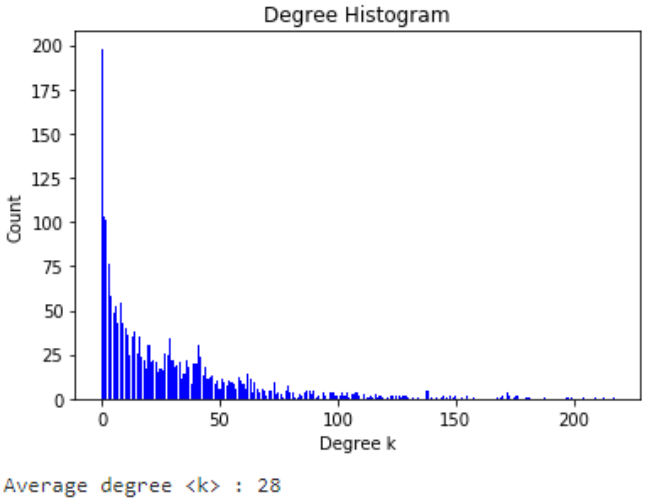

Figure 3: Degree distribution of the graph

4.2 Spectral analysis

To have an idea of the number of connected components and in order to do some dimensionality reduction, we compute the normalized Laplacian of the graph. Then we proceed to its spectral decomposition. The Figure 4 presents the results obtained.

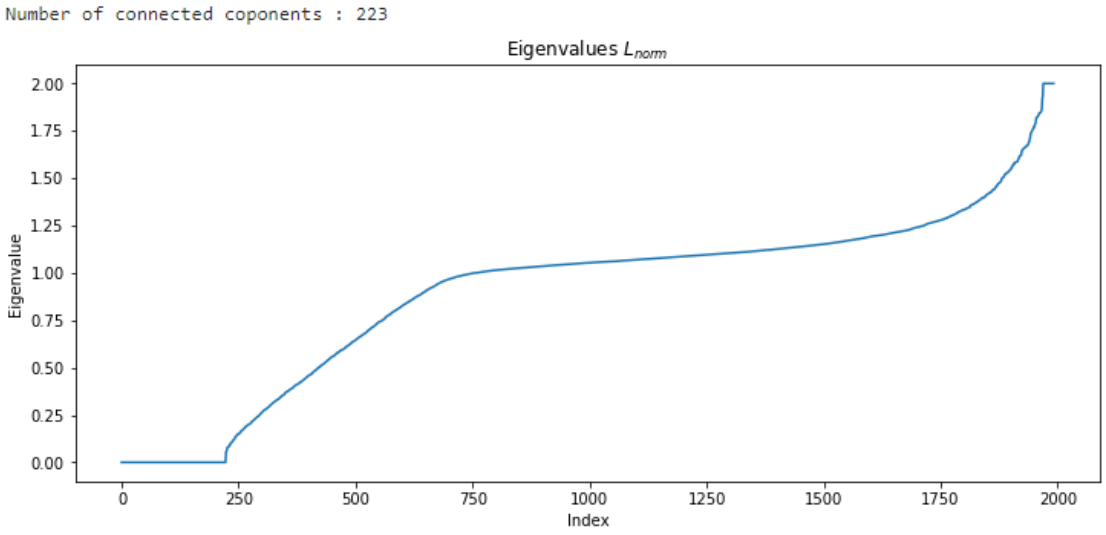

Figure 4: Spectral decomposition of the graph

We see that there are quite a lot of connected components, each of them corresponding to a trivial eigenvalue of the Laplacian. On the other hand, the biggest eigenvalues are associated with the least smooth eigenvectors. The plateau at the end is explained by the fact that eigenvalues of the normalized Laplacian are bounded at 2.

5 Exploitation

5.1 Dimensionality reduction

We use Laplacian Eigenmaps to reduce the dimensionality of the graph : this enables to preserve the local distances of the points even if the manifold is non-linear. We already have computed the normalized Laplacian and its spectral decomposition. We have now to keep the eigenvectors corresponding to the $n_{classes}$ smallest non-trivial eigenvalues, where $n_{classes}$ is the number of clusters into which we will assign the datapoints. We decide to take 5 classes.

5.2 Spectral Clustering

We can now apply the clustering on the projection of the normalized Laplacian that we found in the previous section. To have a better view of the result, we look for the most frequent ingredients contained in the recipes of each cluster. Figure 5 shows the results.

```
Cluster 0 has 537 recipes
{'salt': 216, 'sugar': 153, 'butter': 121, 'egg': 115, 'flour': 108, 'onion': 104, 'water': 102, 'milk': 76
Cluster 1 has 403 recipes
{'salt': 167, 'sugar': 131, 'butter': 115, 'flour': 97, 'egg': 80, 'onion': 74, 'water': 66, 'milk': 54
Cluster 2 has 402 recipes
{'salt': 142, 'sugar': 107, 'butter': 101, 'onion': 81, 'flour': 77, 'egg': 74, 'water': 73, 'olive oil': 54
Cluster 3 has 351 recipes
{'salt': 133, 'sugar': 100, 'butter': 91, 'flour': 68, 'onion': 67, 'water': 55, 'egg': 55, 'garlic clove': 44
Cluster 4 has 301 recipes
{'salt': 106, 'sugar': 89, 'butter': 64, 'water': 53, 'onion': 52, 'egg': 48, 'flour': 44, 'vanilla': 40
```

Figure 5: Most frequent ingredients contained into each cluster's recipes

The clustering of the recipes is not a complete success : we can see that the most common ingredients are present in each cluster. This might be explained by the difficulty to separate recipes as there are always some ingredients that will link different categories of food together. Besides, having labels at disposition or a value for each node could have enabled us to use supervised learning or GFT methods which might have turned out to be more efficient. We also tried to use DBSCAN but it was either putting almost every datapoint into a single cluster or was considering half of the nodes as outliers if we increased the minimum number of samples, without an intermediate solution. Unfortunately, the weighted Jaccard similarity did not provide better results either, and so we keep the weight of an edge as the non-weighted Jaccard similarity.
Finally, we save for each cluster the recipes it contains as well as a "typical recipe" in which we put the most represented ingredients of the cluster. This will enable us in the next part to search for a recipe directly inside the cluster the most likely to contain the ingredients that we have in our fridge.

6 Recommending Recipes

For this part, we have to consider 3 things :

- The "typical recipes", which is the representation of each cluster computed above.
- The User ingredients list, which is all the ingredients that the user writes as input (the ingredients in his fridge).
- The unwanted ingredient list, which contains all the ingredients that the user does not wish for in the final recipes proposed.

As a first step, the program computes the Jaccard similarity between the user's ingredients list and the typical recipes of each cluster. We recompute the typical recipe lists such that their length is the same as the length of the user's ingredients list. Then we select the cluster with which the similarity is the highest. Now the Jaccard similarity is computed between the user's ingredient list and all the recipes belonging to the selected cluster. After that, we return the 10 recipes with the highest Jaccard similarity. Those recipes are filtered in order to not have the unwanted ingredients. If all the 10 recipes have an unwanted ingredient, we compute the 30 recipes with the highest Jaccard similarity. We filter again, and go on similarly until the constraint is satisfied.

```
salt added in ingredients
flour added in ingredients
olive oil added in ingredients
garlic clove added in ingredients
garlic clove added in unwanted ingredients
```

	Idx	Recipe	Ingredients	Similarity
67	67	Mullein Garlic Ear Oil	garlic clove,olive oil	0.500000
95	95	Safeway's Two Hour Turkey	olive oil,salt	0.500000
473	466	Skordalia	garlic,lemon juice,olive oil,potato,salt	0.285714
929	912	Popeye and Olive Oil's Fresh Spinach Salad	black pepper,lime,mint,olive oil,salt,shallot	0.250000
994	976	Quick & Simple Pesto (Classic Pesto Recipe) Sauce	basil,garlic clove,olive oil,parmesan cheese,p...	0.285714
1377	1349	Pizza Crust (All-Purpose Flour or Spelt Flour)	fast rising yeast,flour,salt,water	0.333333
1418	1389	Roast Leg of Lamb	black pepper,garlic clove,leg lamb,lemon juice...	0.250000
1752	1713	Pan-Seared Tilapia	basil,olive oil,parsley,salt	0.333333
1849	1809	Onion Bagels	active yeast,cornmeal,flour,salt,sugar,warm water	0.250000

Figure 6: Example of suggestions of recipes proposed by our application.

7 Conclusion

This project shows how we can apply some methods seen in the course to realize a fun application. It also shows that it is not always easy to find clear clusters in the data. However, the results of our program are satisfying since the application always finds accurate recipes to propose. To go further into the project, it would be nice to plan, for the following days, a set of recipes chosen such that the ingredients in the fridge are well exploited.

References

[1] Food waste, https://www.wwf.ch/fr/nos-objectifs/gaspillage-alimentaire

[2] Recipes dataset http://infolab.stanford.edu/ west1/from-cookies-to-cooks/recipePages.zip

[3] Similarity, S. Niwattanakul, J. Singthongchai, E. Naenudorn and S. Wanapu 2013. Using of Jaccard Coefficient for Keywords https://www.researchgate.net/profile/Ekkachai_Naenudorn/publication/317248581_Using_of_Jaccard_Coefficient_for_Keywords_Similarity/links/592e560ba6fdcc89e759c6d0/Using-of-Jaccard-Coefficient-for-Keywords-Similarity.pdf