A social network of ingredients: differences between italian and american cuisines

Massimiliano Conte
massimiliano.conte.2@studenti.unipd.it

Francesco Giacomarra
francesco.giacomarra@studenti.unipd.it

Emanuele Zangrando
emanuele.zangrando@studenti.unipd.it

Abstract

Which are the main differences between italian and american cuisine? Which are the main ingredients of each one of the two? Is it true that there's a significant difference in the way in which the two approaches mix the ingredients? All these questions can be answered by investigating the structure underlying the way in which ingredients are used in these two cultures.
Given a list of ingredients, every dish can be loosely represented by the ones contained in it: this kind of representation suggests that peculiarities of these two worlds may be hidden in the hypergraphs telling us which ingredients are together in a recipe.
In this report we analyzed two graphical sample structures underlying these two cultures and we tried to answer all the questions posed at the beginning in order to find differences between the italian and american cooking behaviour.

Contents

1. Data collection

1.1. Data representation

Given that the information we decided to scrape from a recipe was the list of ingredients contained in it, we needed to decide a suitable way to represent this kind of data. Since the meaningful information is contained in the relations between ingredients, a natural representation would be an hypergraph.
A problem of this kind of representation is that the number of ingredients is changing among recipes: a solution to this would be to project the hypergraph to lower dimensional objects, like a k-homogenous hypergraph. With this decision we can represent each of the two cuisine samples as an hypergraph Y, whose vertices are the ingredients and whose hyperedges are their relationships. The entries of its sociotensor would be

$$Y_{i_1,..,i_k} = \sum_{\text{r} \in Recipes} I_{\{i_1,..,i_k \in r\}} \propto \widehat{P}\{i_1,..,i_k \in r\}$$

where k is the number of dimensions of the tensor (so the maximum "degree" of the interactions) and $i_1, ..., i_k$ are ingredients. Up to a proportionality constant, these quantities are estimators of the joint probabilities over the groups of k ingredients in the two different scenarios.
The choice of k depends on how much information we want to retain from the original structure, but unluckily the needed sampling size for a good estimation is increasing exponentially with it.
In the particular case where $k = 2$ we have the usual graph structure, that is the choice we landed on. We want to stress that even though our particular choice of k, all the models we tried can be extended to fit the sociotensor for an arbitrary value of it (like ergm for hypergraphs).
Our goal then was to capture "structural" differences in the american and italian cuisine by using these graph informations.

1.2. Scraping

We collected our data via a scraping procedure of two websites:

- *Giallozafferano*, one of the most popular cooking portal in Italy for the italian cuisine;
- *Tasty*, a cooking site belonging to the international Digital Company "Buzzfeed" that focus its recipes on the idea of comfort food for the american cuisine.

For both the sites we used the python libraries *Beautiful Soup* and *Selenium* that allowed us to scrape first the URLs of the recipes in the website and then scrape the ingredients for each single recipe.
We then used a specifically made dictionary to standardize our ingredient list in order to get a cleaner and more interpretable result (e.g. all the beef cuts were grouped into the macrocategory *Beef*, the same goes for *Seafood* that includes both crustaceans and molluscs). We scraped the same number of recipes for the two websites, in order to make easier the comparison.

1.3. Sociomatrices creation

We decided to stick to the classic network representation of the data, projecting all the higher degree interactions into couple relationships, for simplicity of analysis reasons. We then proceed in building the sociomatrices for each cuisine with this criterion: $Y^{(a)}$ corresponding to the website a is symmetric $n \times n$ matrix where n is the number of ingredients; the entry y_{ij}, with $i \neq j$ is equal to the number of recipes in our sample, from the site a, that includes both the ingredient i and the ingredient j. By the end this process we ended up with the symmetric sociomatrices $Y^{(giallo)}, Y^{(tasty)} \in \mathbb{N}^{n \times n}$.

2. Exploratory analysis

After the costruction of the two graphs from the collected data, we started to analyze and compare some basic statistics.

2.1. Characteristics of the dataset

2.1.1 Recipes

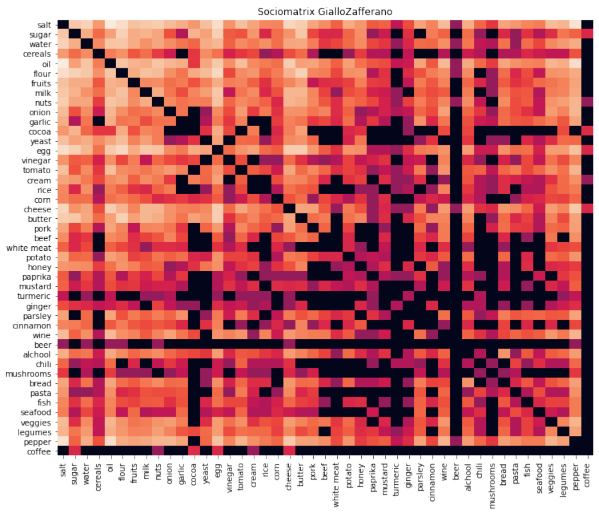
We scraped 1387 recipes per each website. Each website has its own division in categories, and each recipe belongs to one of them. In particular:

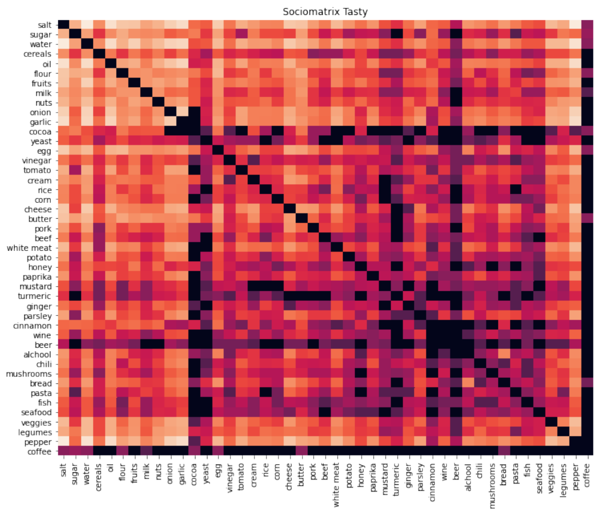
- *Giallozafferano*: *Appetizers, First Courses, Main Courses, Sweets and Desserts, Leavened products*
- *Tasty*: *Breakfast, Lunch, Dinner, Desserts, Snacks*

Already with this informations we started to see some differences: *Giallozafferano* divides the recipes based on their location in the meal, while *Tasty* divides the recipes based on the specific time of the day where that recipe is usually consumed.
We selected $n = 45$ ingredients for our analysis. The recipes from *Giallozafferano* have, on average, 7.41 different ingredients each, while the recipes from *Tasty* have 7.09. If we consider the number of ingredients as the complexity of the recipe, then we can't see relevant differences between the cuisines.

2.1.2 Sociomatrices

We recall that we obtained 2 sociomatrices $Y^{(giallo)}$, $Y^{(tasty)} \in \mathbb{N}^{n\times n}$, where each entry $y_{i,j}$ is the number of interaction of the i-th ingredient with the j-th ingredient in our corpus of recipes. We say that one ingredient is connected with another one if the number of interactions between the two are grater that 0.





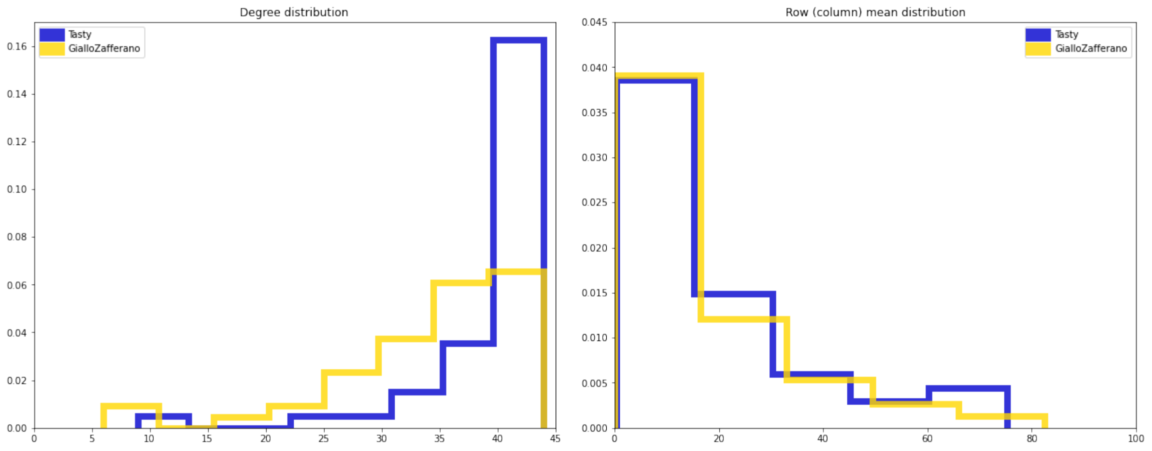
2.2. Descriptive statistics

The grand means are $\bar{y}^{(giallo)} = 17.81$ and $\bar{y}^{(tasty)} = 20.43$. This means that, despite the fact that recipes from *Tasty* have a lower average number of ingredients, the average number of interaction is greater.

More interesting than the grand means is the comparison of the edge densities of the two graphs: 77.78% for *GialloZafferano* and 89.9% for *Tasty*. This showed us how there are more combinations of ingredients that are never used in the Italian website, suggesting that maybe Italians have stricter rules for the choice of the combinations. Both graphs are pretty dense, so statistics like the maximum degree or the diameter are trivial.

2.3. Centrality measures

The easiest measure of centrality for a node is its degree, the number of ingredients with which it has at least one interaction. This measure does not account for the strength of the relationships, the number of interactions, so we put beside it a simple measure that accounts for this, the row mean. Note that, since the graphs are undirected (and the sociomatrices symmetric), the row mean is the same as the column mean. The row mean, for a node, is the total number of interactions of that ingredient normalized. In order to compare these statistics for the two cuisines, we oppose the empirical distribution with an histogram, and test the hypothesis of equal mean with the *t-test*.



The plot suggest us that the degree distribution of *Tasty* is more shifted to the right, with respect to the one of *GialloZafferano*. The row mean distribution, instead, does not show differences. These facts are confirmed by the *t-tests*: the hypothesis of equal mean, between the *GialloZafferano* and *Tasty* distributions, is rejected for the degree distributions (p = 0.001), while is not rejected for the row mean distributions (p = 0.51). While there are no difference in the distribution of the importance of the ingredients (row means), the way interactions are chosen are different, in particular Americans tend to mix the same ingredient with a wider range of foods in their recipes.

2.3.1 The ingredients of the difference

What are the actual ingredients that cause the difference between the cuisines? Since *Tasty* has a denser graph, some combinations of them are not present in *GialloZafferano* recipes. The majority of them is normal, but here we list some of the most strange (from an italian point of view): *ginger with pasta, mushrooms with fish, white meat with seafood*.
Moreover, we compare the *GialloZafferano* degree with the *Tasty* degree of the same ingredient, in order to see what are the ingredients with the most difference. Starting from the most different one, the list of the ingredients of the difference are: *beer, mushrooms, white meat, turmeric, chili, beef, paprika, ginger*.

2.3.2 Analysis of hubs

Another interesting aspect of a network is its hubs. A hub is a node with a degree much larger than the mean. Given the degree distribution of the graphs that we are considering, the usage of such definition would not discover any interesting result, since many nodes are connected with all the others.
A more useful measure of node importance, for the identification of the hubs, is the *betweenness*, since it's affected by the strength of the relationships. We recall that the betweenness of a node v is $g(v) = \sum_{v \neq s \neq t} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$, where $\sigma_{s,t}(v)$ is the number of shortest paths between s and t that pass through v and $\sigma_{s,t}$ is the total number of shortest paths between s and t, so in a certain sense it represents the probability that a geodesic connecting s and t passes through v.
It is clear that a crucial aspect is how the shortest paths are computed: the number of interactions represents the strength of the relationship, and it can't be used as distance measure in the shortest paths computation. A simple idea could be just using the inverse, $d(u, v) = \frac{1}{Y_{u,v}}$. Since the graphs are so dense, most of the shortest paths are of length 1 or 2, all passing through the few relevant ingredients, and so most of the ingredients have a *betweenness* equal to 0. Given that, we compare the ingredients from the two websites that have a *betweenness* greater than 0, considering them as the hubs of the graphs:

- *GialloZafferano*: *Salt, water, oil, eggs, cocoa, flour, pepper*;
- *Tasty*: *Salt, water, pepper, sugar, garlic, flour, butter, fruits, eggs*.

The relevant differences are in the hubs which are not in common, in particular Italians tend to use *Oil.* while Americans *butter*, *cocoa* is a hub only for *GialloZafferano* while *sugar, garlic, and fruits* are hubs only for *Tasty*. The other hubs are just fundamental ingredients in general.

3. Community detection

In this section we investigate communities in the network, in order to see if there is any culinar or ideological difference between the implicit groups of ingredients generated by the two corpus of recipes.

3.1. Louvain method

We decided to extract the communities in both graphs using the *Louvain Method*[3], that is a greedy algorithm for community detection based on the maximization of the *Modularity*: We define the Modularity $Q \in [-1/2, 1]$ of a Graph with communities in the following way

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \tag{1}$$

Where:

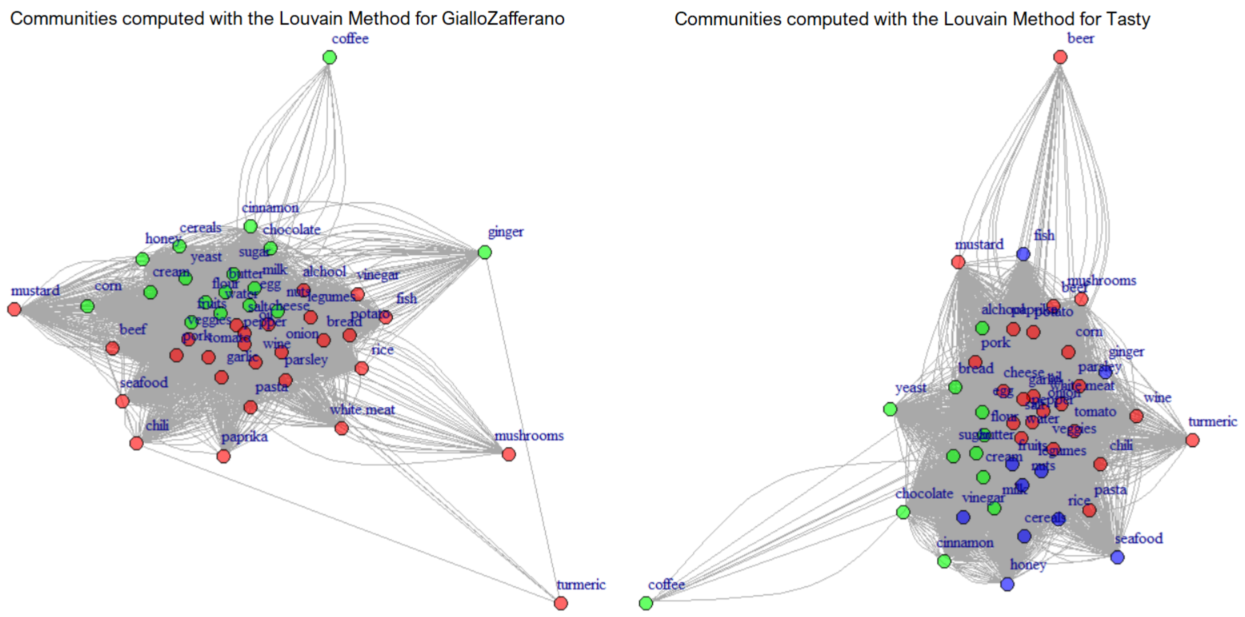
- A_{ij} represents the edge weight between nodes i and j;
- k_i and k_j are the sum of the weights of the edges attached to nodes i and j, respectively;
- m is the sum of all of the edge weights in the graph;
- c_i and c_j are the communities of the nodes;
- δ is the so called *Kronecker delta function* ($\delta(x, y) = 1$ if $x = y$, 0 otherwise)

So Q measures the density of links within the communities compared to the links between the communities. The goal of the algorithm is to maximize the modularity by partitioning the nodes into different communities. In order to do so, it starts with a different community for each node, then it alternates two phases until no changes occur:

- **Partitioning**: Cycle through each node i, calculating the improvement of Q if i is moved to the community of j, for every j connected with i (this is the key strength of this method, since this "gain" is easy to compute). Node i is moved to the community that maximize the Q improvement (it can also stay on the same community). The cycle keeps going until no changes occur for all the nodes, in a coordinate optimization fashion;
- **Aggregation**: A new network is built, where the nodes are the previous communities, and the edges between two nodes are the sum of the edges between the corresponding communities. Self loops are used to represent connections between nodes from the same community. Then the first phase is applied to the new graph.

This algorithm automatically finds the number of communities.

3.2. Results



Communities extracted with the Louvain Method

For *GialloZafferano* we have 2 communities, there is a clear distinction between ingredients used for sweet dishes and ingredients used for savour recipes, that is a classical discrimination done by the Italian cuisine.
On the other hand, for *Tasty*, we see 3 communities that we interpreted as: the savoury one, the sweet one and the healthy one. The healthy community includes those ingredients that are normally considered as healthy, like *fish, seafoods, cereals, nuts, fruits and veggies*. This reflects the cultural difference we saw also in the category division of the websites, that Italian tend to group recipes by their taste, while American tend to group recipes and ingredients by other properties, such as the healthiness. The healthy community seems coherent to the idea of typical American cuisine as less prone to use those ingredients by themselves, but very much conditioned by the desire of a part of the population to eat healthy.

4. AMEN modeling

4.1. Model definition

In order to capture the underlying generative mechanism behind each graph, we fitted some network models. Despite its popularity, ergm models like the p1 model [7],[6] provided us a bad fit, so we decided to focus on other possibilities.
In particular, we used [7] AME (Additive and Multiplicative Effects) models. The *AME* matrix decomposition is described by the following equations:

$$Y_{ij} = g(s_{ij}) \tag{2}$$

$$s_{ij} = \beta^T \mathbf{X}_{ij} + a_i + b_j + \alpha(\mathbf{u_i}, \mathbf{v_j}) + \epsilon_{ij} \tag{3}$$

Where we used the following notation:

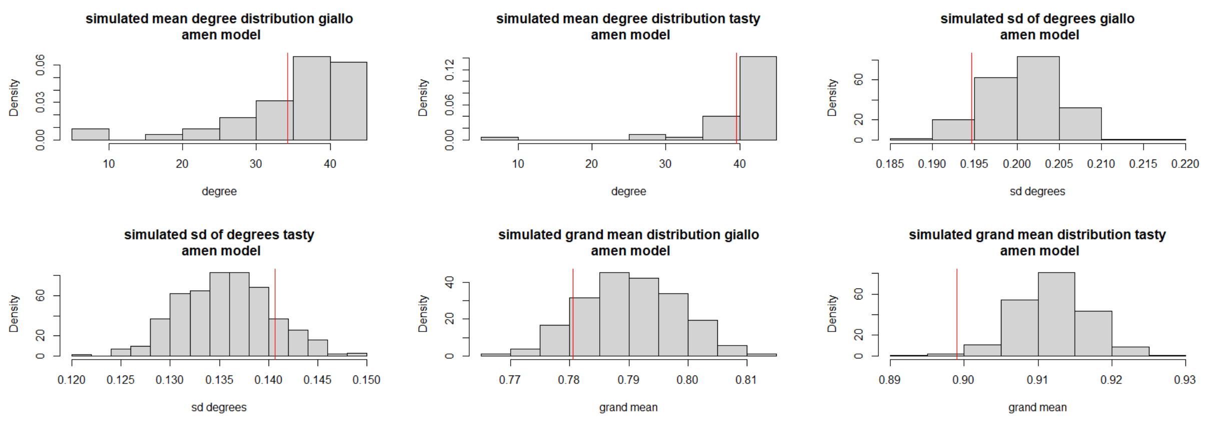
- g is the link function, needed when the sociomatrix entries are not representable by a Gaussian distribution;
- $\mathbf{X}_{ij}$ is a vector of nodal/edge features and β is the vector of coefficients associated to them;
- a_i, b_j represents respectively the row effect and the column effect and ϵ_{ij} accounts for the residual within dyads dependence, as seen in the classical RCE model;
- $\alpha(\mathbf{u_i}, \mathbf{v_j})$, called the multiplicative term, accounts for residual higher order dependencies (in particular third-order dependencies, that are dependencies that emerges within triads) that cannot be captured by the additive part described before. As examples, the **latent distance model** with $\alpha(\mathbf{u_i}, \mathbf{v_j}) = -||\mathbf{u_i} - \mathbf{v_j}||$ or the **multiplicative effects model** with $\alpha(\mathbf{u_i}, \mathbf{v_j}) = \mathbf{u_i}^T \Lambda \mathbf{v_j}$. The version we took under consideration was the multiplicative effects model.

We will now briefly describe what is the idea behind the multiplicative term: a standard ANOVA model cannot include third order dependencies pattern, for example like the stochastic equivalence[1] which can occur in our scenario since we include ingredients with very similar uses. To solve this problem we assume that each node has a non observable representation as a point in a $K-$dimensional space; in the $K-$dimensional space each node is represented as a vector of K latent factors that are non observable node specific variables which mediate its relationships. Within this framework the multiplicative term $\alpha(\mathbf{u_i}, \mathbf{v_j})$ is a function of $\mathbf{u_i}$, that is the latent row vector of node i and $\mathbf{v_j}$ that is the latent column vector of j.
Since our sociomatrix is symmetric and we have no node/edge features, the model becomes even simpler (we can neglect b_j, $\mathbf{X_{ij}}$ and $\mathbf{U} = \mathbf{V}$).
Given the impossibility of having a closed form for the likelihood of the generalized AMEN model, a Bayesian approach for fitting has to be applied in order to approximate the posterior distribution $P(Y|Y^{obs})$ (more details in [7]).
With this approach, any function of Y from the posterior distribution can be approximated by using Gibbs sampling.

4.2. Model validation

We fitted the model, on both networks, using the *R* package *amen*. Before using the model, it is necessary to check whatever the goodness of fit is reasonable. In order to validate the model, we chose a set of network statistics, in particular *grand mean* and *standard deviation of row/column mean*, then we simulated a bunch of networks with our model to see if the observed statistics are coherent with the model generated ones (best case scenario testing).

[1] A pair of nodes i and j are stochastically equivalent if the probability of i relating to, and being related to, every other node is the same as the probability for j. This refers to the idea that there will be groups of nodes in a network with similar relational patterns.



goodness of fit of the amen model through some statistics.

All the observed values (the red lines) fall in the regions with high density, meaning that the model is capturing at least some properties of our networks.

4.3. Latent factors interpretation

For each node i of our graph, we have its row and column representation on a latent space, u_i and v_i. As briefly mentioned in the model definition section, the model can be rewritten in the form

$$logoddsP\{Y_{ij} = 1|\mathbf{X}_{ij}, a_i, b_j, \mathbf{u_i}, \mathbf{v_j}; \beta\} = \beta^T \mathbf{X}_{ij} + a_i + b_j + \mathbf{u_i}^T \mathbf{v_j}$$

From this formulation we can understand the effect of the latent interaction term $\mathbf{u_i}^T \mathbf{v_j}$ on the probability of creating the edge (i, j) : the bigger the value of the projection of u_i onto v_j, the bigger the probability of having the edge $Y_{i,j} = 1$.
As we can read from the last equation, latent effects are invariant under rotations: in fact, if we transform $U, V \mapsto UO, VO$ with some orthogonal matrix O, we have $UO(VO)^T = UV^T$.
This lack of identifiability is a problem when trying to compare and interpret latent effects for the two different networks ($U^{(Giallo)}$ and $U^{(Tasty)}$), cause they may be similar but in a rotated version.
To account for this fact we fixed Tasty latent variables and we found the "best rotation" of *Giallo* to compare them: this has been done by solving the optimization problem

$$\min_{O \in O_k(\mathbb{R})} ||U^{(Giallo)}O - U^{(Tasty)}||_2^2$$

We reported the comparison of the latent effects for $k = 2$ in figure (2).
Since the relationships between ingredients in the latent space is captured by the whole configuration $U = (u_1, .., u_n)^T$, a better way to compare interaction effects of *GialloZafferano* and Tasty is to compute the interaction matrices $U^{(a)}U^{(a)T}$ for $a \in \{Giallo, Tasty\}$ and consider their difference $\Delta = U^{(Giallo)}U^{(Giallo)T} - U^{(Tasty)}U^{(tasty)T}$ to look for big discrepancies.
In particular as examples, the two way latent interaction between *rice* and *turmenic*, *turmenic* and *milk* and *chili* and *potato* is bigger in Tasty, probably because of the presence of typical american recipes not found in *GialloZafferano*.
Other informations of this kind can be revealed by looking at the (1): some ingredients (like *paprika, chili and wine*) have the opposite "fitness" effect in *Giallo* and *Tasty*.
By sampling from the posterior distribution on parameters one can even get some insights about the significance of this effects according to the model.

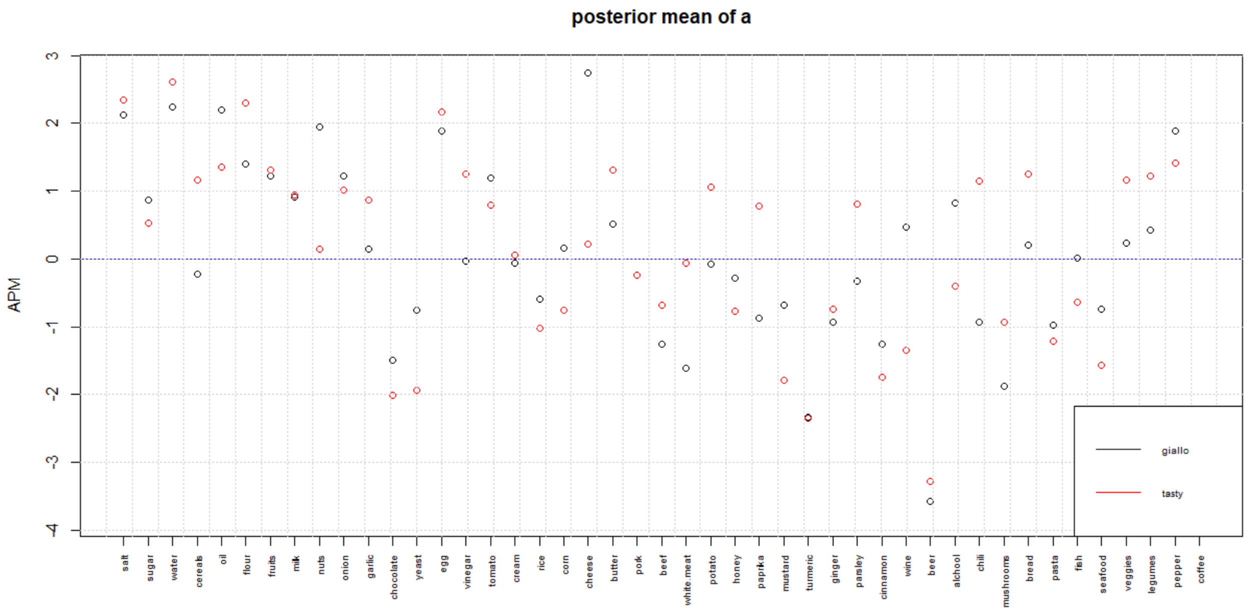


Figure 1. Posterior mean of row (column) effect for the two fitted models.

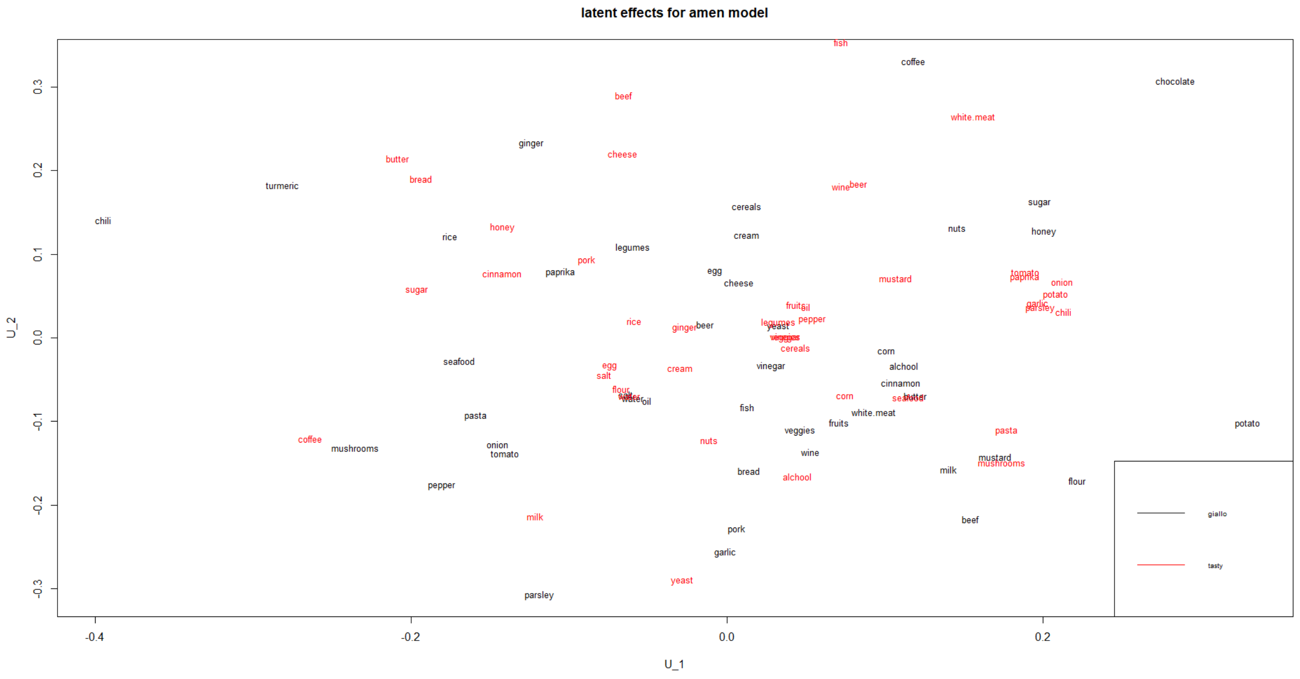


Figure 2. Latent variables scatterplot for the two networks (best rotated version).

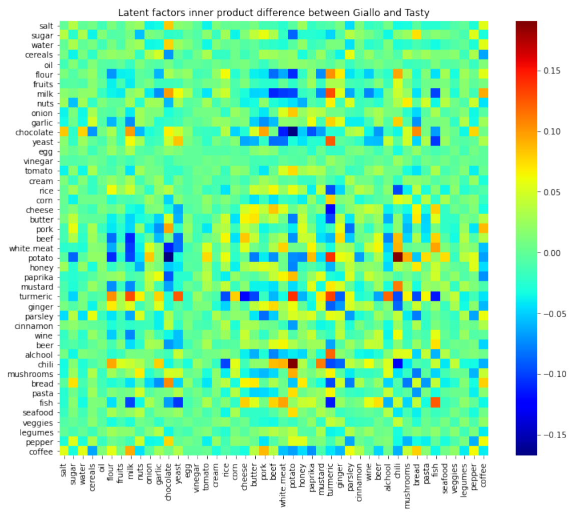


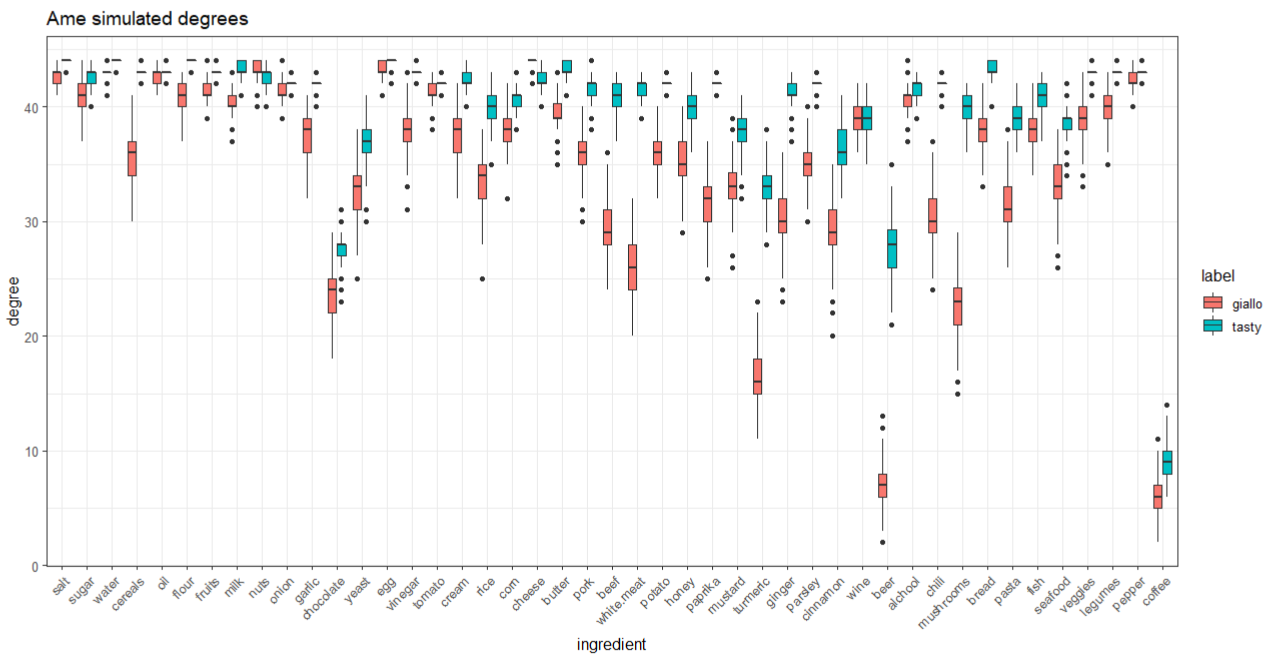
Figure 3. Latent factors inner product difference between Tasty and Giallo

4.4. Simulations

In this section we take advantage of the capacity of the model to simulate new networks, keeping the node properties intact, thanks to the addictive and multiplicative effects, that are node specific. In this way, we can still refer to the each node as one specific ingredient.

4.4.1 Degree distribution for each node

Since many nodes in our network have the maximum degree possible, we sampled new networks from the model, in order to get the distribution of degree for each individual node. We simulated several networks, and for each of them we computed the degree across the nodes. In this way we can access the variability and compare better the degree between nodes in the two scenarios.



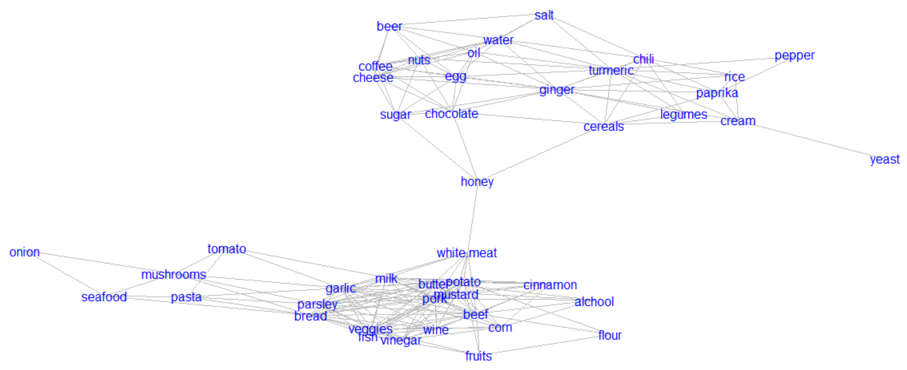
Simulated marginal degree distributions of the nodes.s.

For example, the degree of *oil* is equal for the two websites, but in this plot it seems higher from *Tasty*. Same for *nuts*, but here the degree seems higher for *GialloZafferano*.
The exact same kind of simulation can be done with any desidered nodal statistic (like the Pagerank for example), and it may lead to some more insights about difference of importance of the ingredients in the two cultures from a point of view different from the degree.

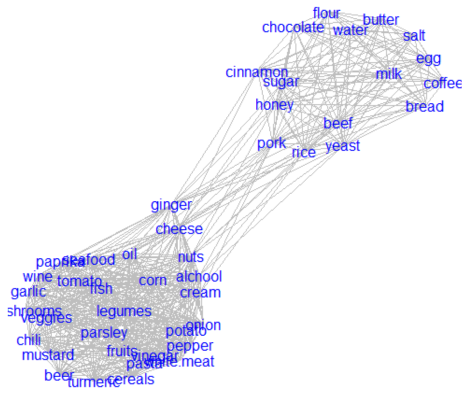
4.4.2 Assess the stability of the community structure through simulations

After fitting AME models for the two sociomatrices, we decided to go for model based community detection.
If we make the assumption, albeit it's a strong one , that our models are correctly specified with respect to the actual processes that generate our data, we can use simulations for the models to have a better insight on how the communities are extracted. In fact the Louvain method that we applied just assigns a community label to each of the node, so we cannot,for example, tell if some nodes are "borderline" between two or more communities and more in general we cannot say anything about the overall stability of the communities.
With this in mind, our approach was the following: for both posterior distributions we simulated i.i.d matrices $Y^{(i)} \sim P(Y|Y^{obs})$ and we made Louvain community detection on them to obtain a Monte Carlo estimate of the matrix $Q_{ij} = P(\text{i and j in the same community})$.
Then we built two new networks, one per each website. The nodes are the ingredients, and two of them are connected if the number of times they were in the same community is above one half of the times.

Giallo community interaction graph



Tasty community interaction graph



From these graphs we can get a probabilistic idea on how two ingredients, on average, are collocated into the same community.
As we can see in the picture, models fitted on the *GialloZafferano*'s sociomatrix tend to create a distinction between savoury dishes and the sweet ones, as happened for the actual graph, however it seems that spices are most of the time included in the same community as the sweet ingredients, moreover we can see that the "sweet" component of the graph actually includes all types of seasoning because of the presence of *salt*[2], *oil* and *sugar* and also *legumes*, probably because legumes in the Italian cuisine are treated as a dish per se and not often combined with meat, fish or other vegetables; fruits are on the converse more often included in the "Savoury Community", likely this is due to the fact that it is typical to mix sour fruits with vegetables, fish and seafoods, and sweet fruit with cheese.
For *Tasty* we have a pretty different behavior of the communities we can still see the *Sweet* and *Savoury* components in the graph, however the distinction is less clear, which can be interpreted as a good sign of fit to reality, since, as we said before, we expect less strictness for the American cuisine when mixing ingredients with each

[2]the presence of salt in the sweet communities might seem strange, however we must recall that salt is also included in almost all the bakery preparations!

other. We must also recall that by applying the *Louvain* method to the actual graph we ended up with three communities, *Sweet, Savoury* and one that we called *Healthy*; the latter one seems to be included in the Savoury component, which seems coherent since most of the ingredients in the "healthy community" were commonly used in savoury dishes. Another seemingly oddity is the presence of *Beef* and *Pork* in the sweet component of the graph, but in our opinion this could be explained by the usage of sweet-based seasoning that the Americans use on meat.

5. Conclusions

5.1. Summary

In this work we analyzed two different networks, built from scratch starting from the recipes in the websites *GialloZafferano* and *Tasty*.
In the exploratory analysis, we found evidence that recipes in *Tasty* use a wider variety of combinations of ingredients, suggesting a weaker set of rules in composing a recipe. Then we saw how some ingredients are less used in *GialloZafferano*, in particular *beer, mushrooms, white meat, turmeric, chili, beef, paprika, ginger*. Analyzing the hubs, we found the most important ingredients of the two cuisines, *sugar, garlic*, and *fruits* are more used in *Tasty*, while *cocoa* in *GialloZafferano*. Moreover, *Oil* is the principal fatty seasoning for *GialloZafferano*, while for *Tasty* it is the *butter*.
Examining the communities discovered by the *Louvain method*, we saw how *GialloZafferano* tends to split recipes in sweet or savoury dishes, while a third community, that we interpreted as healty dishes, appears in *Tasty*.
We then fitted an *AMEN* model in order to explain the generative mechanism of the data. In particular, we adapted one different model for each network. We then checked the goodness of fit, by comparing the simulated distribution of some statistics, with the same statistics computed on the observed data. Firstly, we analyzed the additive effects and compared them among the two graphs to look for differences. Then we analyzed the latent representations generated by the model, rotated in such a way that the two set of points are comparable. The last thing we did is to exploit the generative capabilities of the models, in order to simulate the distribution of degree for each node, and also for assessing the stability of the community detection made before.

5.2. Critical issues

We realized that the analysis made in this work show a list of critical aspects:

- All the work is much sensitive to the choice of the grouping dictionary: we had to define a list of categories in order to generalize the data, but different choices could dramatically change the results. Moreover, ingredients from the same category can have similar usage for most of the time, and sometimes they can be used in a very different way (for example *cocoa* and *chocolate*: sometimes *cocoa* is used as spice in savoury recipes).
- Rigorous interpretation is hard when dealing with network data: we have only one realization of one network. Many methods (such as the *Louvain* one), do not come from a probabilistic framework. Interpretation of simulated data is risky, since we are making the strong assumption that the model is correcly specified.
- We stretched things a bit while interpreting and using the *AMEN* models: we know that some claims we did in this section are, to say the least, with a bit of overinterpretation, but we tried our best to find new possibilities in the analysis of these models.

5.3. Further improvements

What can be done in order to improve or extend our work? We asked ourselves this question, and what we would like to do is:

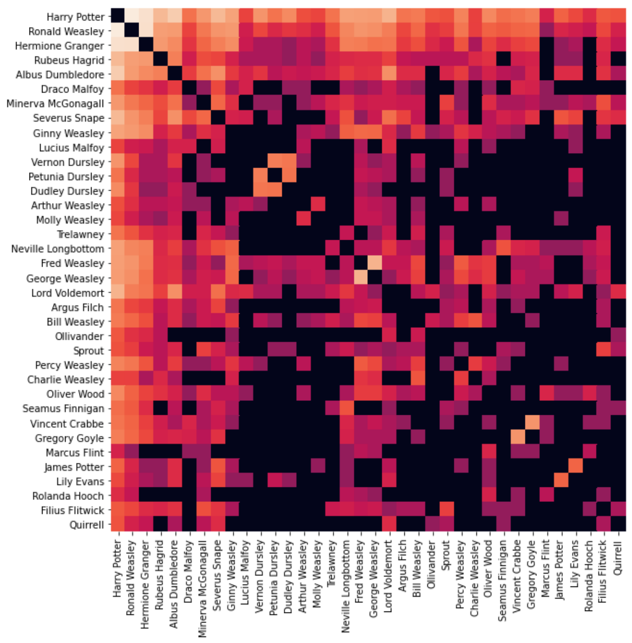
- Obtain new networks for other cuisines, for example the *Japanese* or the *Chinese* one;
- Granularize and expand the grouping dictionary, in order to see what are differences and to discriminate better between ingredients;
- Improve the rigorousness of the *AMEN* part and extend the experiments we did using simulations.

We want to thank you for your attention, hoping that you enjoyed our project, and keep in mind: Italians do it better!

6. Appendix

Harry Potter and the cursed ingredient

What does *Harry Potter* have to do with cuisines and ingredients? Well, the same graph data structure can be used in order to represent its social network. Taken all the books, then splitting the entire corpus into a list of sentences, and then just counting the number of times each couple of characters appear together in the same sentence. Of course some technical tricks are needed, like the creation of a dictionary of aliases. In the end the sociomatrix is $Y \in \mathbb{N}^{n \times n}$, where n is the number of characters, and $y_{i,j}$ is the number of sentences in which both character i and j are present. This is just an example of how data from completely different areas can use the same graph representation.



Ingredient dictionary

This is the dictionary that defines the alisases for each ingredient, and groups them in to categories.

```
ingredients_dictionary = {
   "salt": ["salt"],
   "sugar": ["sugar"],
   "water": ["water", "ice"],
   "cereals": ["cereal", "cereals", "oat", "barley", "malt", "flakes","quinoa"],
   "oil": ["oil"],
   "flour": ["flour", "starch"],
   "fruits": ["apple", "apples", "grape", "orange", "oranges", "strawberry", "strawberries",
              "raspberry", "pineapple", "cranberry", "cherry", "cherries", "banana",
              "bananas", "mango", "mangos", "cucumber", "peach", "peaches", "kiwi", "kiwis",
              "berries", "plum", "plums", "melon", "cider", "lemon", "lemons","figs","pear",
              "limes","lime","pomegranate"],
   "milk": ["milk","wholemilk"],
   "nuts": ["seed", "seeds", "nut", "nuts", "almond", "almonds", "peanut", "peanuts",
              "sesame", "hazelnut","pistachio"],
   "onion": ["onion"],
   "garlic": ["garlic"],
   "cocoa": ["cocoa", "chocolate", "dark-chocolate"],
   "yeast": ["yeast", "sourdough"],
   "egg": ["egg", "eggs","yolk","yolks"],
   "vinegar": ["vinegar"],
   "tomato": ["tomato", "passato", "passata","datterino","datterini"],
   "cream": ["cream"],
   "rice": ["rice", "rices","basmati","carnaroli"],
   "corn": ["corn"],
   "cheese": ["cheese", "cheddar", "mozzarella", "parmisan", "parmigiano", "grana",
              "pecorino", "gorgonzola", "taleggio", "caciotta", "ricotta", "provola",
              "provolone", "mascarpone", "montasio", "piave", "asiago", "puzzone",
              "quartirolo", "scamorza", "stracchino","robiola","roquefort","queso"],
   "butter": ["butter"],
   "pork": ["pork", "ham", "bacon", "salami","ribs","sausage","mortadella","wurstel",
             "bratwurst"],
   "beef": ["ribeye", "beef", "veal","roast-beef","bresaola","ossibuchi",
            "chunk","beefsteak"],
   "white meat": ["chicken", "chickens" "bun", "turquey", "fowl","quail","pigeon"],
   "potato": ["potato", "potatos"],
   "honey": ["honey"],
   "paprika": ["paprika"],
   "mustard": ["mustard"],
   "turmeric": ["turmeric"],
   "ginger": ["ginger"],
   "parsley": ["parsley"],
   "cinnamon": ["cinnamon"],
   "wine": ["wine","prosecco","moscato","pinot"],
   "beer": ["beer"],
   "alchool": ["sake","limoncello","marsala","rum","rhum","porto","brandy",
               "whiskey","whisky", "grappa","bourbon","liqueur"],
   "chili": ["chili", "jalapeno"],
   "mushrooms": ["mushroom", "mushrooms","mushroomsoaking","porcini"],
   "bread": ["bread","breadcrumbs","breadcrumb"],
   "pasta": ["pasta","tagliatelle","lasagne","farfalle","rigatoni","paccheri",
            "strozzapreti","macaroni","linguine","spaghetti","maniche","pappardelle",
            "tortellini","ravioli","gnocchi"],
   "fish": ["fish", "salmon", "swordfish", "cod", "codfish", "sardines", "tuna", "trout",
               "trouts", "mackerel", "mackerels", "herring", "herrings", "hailbut",
               "halibuts","anchovies"],
   "seafood": ["seafood", "lobsters", "mussels", "crab", "crabs", "shrimp", "crayfish",
               "prawn", "clams","oyster","scampi", "squid", "squids", "octopus"],
   "veggies": ["spinach", "fennel", "carrot", "carrots", "celery", "celeries", "peppers",
               "cabbage", "kale", "chicory","zucchini","eggplant","eggplants" ],
   "legumes": ["chickpea", "chickpeas", "pea", "peas", "beans", "bean", "soy", "lentils"],
   "pepper": ["peppercorns", "pepper"],
   "coffee": ["coffee","espresso","cappuccino"]
```

Code for scraping Tasty

```
from selenium import webdriver

options = webdriver.ChromeOptions()
options.add_argument('--ignore-certificate-errors')
driver = webdriver.Chrome("chromedriver", chrome_options=options)

import time
import requests
from bs4 import BeautifulSoup

URL = "https://tasty.co/topic/"
recipe_categories = ["breakfast", "lunch", "dinner", "desserts", "snacks"]

recipes_urls = []
n_recipes_per_cat = {cat : 0 for cat in recipe_categories}

for category in recipe_categories:
    print(f"Scraping {category}")
    full_url = URL + category
    driver.get(full_url)
    show_more = driver.find_elements_by_css_selector("button.button--tasty.bold.xs-block.xs-
    mx-auto.xs-col-12.md-width-auto")
    while len(show_more) > 0:
        driver.execute_script("arguments[0].click();", show_more[0])
        time.sleep(3)
        show_more = driver.find_elements_by_css_selector("button.button--tasty.bold.xs-block
    .xs-mx-auto.xs-col-12.md-width-auto")

    page_source = driver.page_source

    soup = BeautifulSoup(page_source, "lxml")
    results = soup.findAll("a")
    for link in results:
        if "/recipe" in link["href"]:
            recipes_urls.append(link["href"])
            n_recipes_per_cat[category] += 1

def get_ingredients_list(url):
    ingredients = []
    page = requests.get(url)
    soup = BeautifulSoup(page.content, "html.parser")
    results = soup.find_all("ul", class_ = "list-unstyled xs-text-3")
    if len(results) > 0:
        results = results[0].find_all("li", class_ = "ingredient xs-mb1 xs-mt0")

        for ingedient in results:
            ingredients.append(ingedient.text)

    return ingredients

recipes_ingredients = []

for url in recipes_urls:
    full_url = "https://tasty.co/" + url
    recipes_ingredients.append(get_ingredients_list(full_url))
```

Code for creating the sociomatrices

```python
import numpy as np
regex = re.compile('[^a-zA-Z ]')
import itertools

ingredients_list = np.array(list(ingredients_dictionary.keys()))
n = len(ingredients_dictionary)
socio_giallo = np.zeros((n,n))
socio_tasty = np.zeros((n,n))

# tasty_ingredients and giallo_ingredients are list of lists
# recipes x ingredients

for recipe in tasty_ingredients:
    ingredients_in_recipe = set()
    for ingredient in recipe:
        ingredient = regex.sub('', ingredient)
        for key_food, psuedo_list in ingredients_dictionary.items():
            for w in psuedo_list:
                if w in ingredient:
                    ingredients_in_recipe.add(key_food)
    for subset in itertools.combinations(ingredients_in_recipe, 2):
        i = np.argwhere(subset[0] == ingredients_list)
        j = np.argwhere(subset[1] == ingredients_list)
        socio_tasty[i, j] += 1
        socio_tasty[j, i] += 1

for recipe in giallo_ingredients:
    ingredients_in_recipe = set()
    for ingredient in recipe:
        ingredient = regex.sub('', ingredient)
        for key_food, psuedo_list in ingredients_dictionary.items():
            for w in psuedo_list:
                if w in ingredient:
                    ingredients_in_recipe.add(key_food)
    for subset in itertools.combinations(ingredients_in_recipe, 2):
        i = np.argwhere(subset[0] == ingredients_list)
        j = np.argwhere(subset[1] == ingredients_list)
        socio_giallo[i, j] += 1
        socio_giallo[j, i] += 1


# interactions are counted 2 times
socio_giallo /= 2
socio_tasty  /= 2
```

References

[1] *Exponential Random Graph Models for Social Networks: Theory, Methods, and Applications*. Structural Analysis in the Social Sciences. Cambridge University Press, 2012.

[2] Alain Barrat, Marc Barthélemy, and Alessandro Vespignani. *Dynamical Processes on Complex Networks*. Cambridge University Press, 2008.

[3] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[4] Peter D Hoff. Bilinear mixed-effects models for dyadic data. *Journal of the American Statistical Association*, 100(469):286–295, 2005.

[5] Peter D. Hoff. Dyadic data analysis with amen, 2015.

[6] Eric Kolaczyk and Gbor Csrdi. *Statistical Analysis of Network Data with R*. Springer Publishing Company, Incorporated, 2014.

[7] Shahryar Minhas, Peter D. Hoff, and Michael D. Ward. Inferential approaches for network analysis: Amen for latent factor models. *Political Analysis*, 27(2):208–222, 2019.