CS 224W Information Networks
Angela Sy, Lawrence Liu, Blanca Villanueva
Dec. 9, 2014
Project Final Report

Finding Your Food Soulmate

1. Abstract

Individuals in social networks are often unaware of people who share similar tastes as they do. From our own personal experience, we believe that people often bond over food. The goal of this project is to consistently identify clusters wherein users can both discover new friends, and also identify members of their own friend lists with whom they may share common food tastes with. To achieve this, we took information from the Yelp dataset, identified network properties and generated a network model that recognizes food compatibility between Yelp users. We classify similarity in user attitudes towards restaurants (e.g. similar ratings and reviews of restaurants) as indicative of similarity in user food tastes as well. This allowed for the creation of an evaluation network against which we tested and optimised consideration of other factors of food-related experiences as indicative of similarity in food tastes (e.g. parking, cost, atmosphere). We finally run a clustering algorithm on a subgraph of our model and compare this with the real-world network for the purposes of friend recommendations centered on food.

2. Related Work

Significant research has been conducted on link prediction and community clustering for different networks. We evaluated two algorithms relating to community discovery and cluster assignment: the Girvan-Newman algorithm, and the Clauset-Newman-Moore algorithm. Advantages and disadvantages discussed range from the ability for an algorithm to capture belonging in multiple communities, to the computational viability of certain approaches for certain datasets.

The Girvan-Newman (GN) algorithm is based on the idea of modularity and edge betweenness. For any given division of a network into communities, the researchers propose a quality index to evaluate how well that particular division actually identifies groups based on the density of in-group edges and between-group edges. They propose a greedy algorithm to actually find a near-optimal such division that starts by treating the graph as consisting of n communities (wherein each of the n nodes is considered to be its own cluster), and joining communities until all nodes are connected (this is done by removing the edge with the highest value of edge betweenness). After each connection, the researchers evaluate the quality index, and can thus identify the division that had the highest quality index throughout the process.

The Clauset-Newman-Moore (CNM) clustering algorithm is also based on the idea of modularity. The algorithm identifies and merges the two communities that contribute the maximum possible value to global modularity. That is, the CNM clustering algorithm greedily optimises modularity. Similar to the GN algorithm, the CNM algorithm treats the initial graph as consisting of n communities (wherein each of the n nodes is again considered to be its own cluster). In contrast with the GN algorithm which removes edges based on betweenness centrality, the CNM algorithm joins two communities with the highest modularity.

David Liben-Nowell and Jon Kleinberg's research on The Link Prediction Problem for Social Networks is similar to ours in that it aims to accurately make claims about the evolution of a social network using features intrinsic to the network itself. Liben-Nowell and Kleinberg look specifically at five co-authorship networks and examine several aspects of their dataset such as interactions between users during a particular time period, graph distance, and Jaccard's coefficient. To evaluate these link predictors, Liben-Nowell and Kleinberg calculate several measures

such as the size of the intersection set of predicted links with existing ones. Different predictors are more accurate for different networks.

3. Preparing the Data

3.1. Raw Data

We use the Yelp Academic Dataset 2 (2013) to examine three types of objects: User objects, Business objects, and Review objects. We are focusing on the Yelp dataset for the city of Phoenix, Arizona–specifically the Yelp Phoenix Academic Dataset 2. Our raw data set size is as follows:

Object Type	Count
Users	70,817
Businesses	15,585
Reviews	335,022

3.2. Cleaning Up the Dataset

Because we are basing our analysis on food-related experiences, businesses and reviews within the Yelp dataset that are unrelated to food are removed.

Node edges are created based on review events shared between users. That is, they are based on reviews of the businesses users have commonly reviewed or visited. If a node has never had any user activity, it will become an isolated node, which is unhelpful in our analysis of communities. Nodes with fewer than one review attributed to them serve an unnecessary computational expense and are therefore removed.

3.3. Metadata

We created a layer of meta-data to allow us to calculate edge scores between nodes in order to generate the network. We maintain a map from each restaurant to a list of User IDs who have written a review for that restaurant, a map of reviews to star ratings and date, and, most importantly, a map of restaurant attribute preferences for each user. The metadata will be generated by examining the review data created by Yelp's userbase. Finally, we have a node ID that is the ID of the specific user in the graph (unique for each user in the network).

Concretely, for each user node, we will have:

- RestaurantMap: {businessID: [review1_ID, review2_ID, ...]}
- ReviewMap: {reviewID: (starRating, date)}
- AttributePrefCounter: {attribute: importance}
- nodeID

The Restaurant Map has a key for every business the user has written a review for, mapped to a list of review IDs for that specific business. The Review Map saves further information about each specific review and stores the star rating for that review as well as its date.

The AttributePrefCounter has a list of restaurant attributes and the importance of each attribute to the user. The importance is calculated by the number of restaurants that the user has reviewed which have this attribute divided by the total number of restaurants the user has reviewed.

Other generated metadata include:

- GlobalBusinessMap: {businessID: [userID, userID, userID]}

The GlobalBusinessMap is created at the same time as the RestaurantMap. The keys of this map are of all the businesses that have been reviewed at least once, and its keys are the userIDs of those users that have reviewed the business. This allows us to keep track of all the users who have reviewed a particular restaurant.

4. Rating Network Model

We create a rating-based network and use this to train our food network model which we create using our attribute similarity score. The true/rating-based network rating model is described in Section 4 and the food attribute network model is described in Section 5.

Our rating network is an undirected graph representing Yelp user food friendship circles. We consider this the "true" network because, by definition, reviews of common restaurants between two users represents their opinions on the *same* dining experience. To train our food attribute network model, we seek to devise an algorithm that produces the maximum number of correct classifications--that is, our algorithm should classify edges the same way the ratings-based network classifies food compatibility.We calculate the accuracy of the food attribute network model by comparing the edges created in the food attribute network model using the Attribute Similarity score against the actual edges in the test rating network using the Rating Similarity score.

4.1 Constructing a Rating Network with Rating Similarity Scores

Intuitively, the entire population is a network wherein each node is a user, and two users share an edge when they have compatible food tastes. However, we need to qualify the notion of "compatibility" between food tastes. When do we draw the edge between users? To construct this true network, we use data that we know with certainty reflect users' opinions on the same food: user reviews on the same restaurants. We want to draw an edge between two users when, on average, they share similar food tastes.

To do this, we identified any two users that have eaten at the same location. These users have shared the same dining experience, and since dining is a self-selective process (people choose to eat at restaurants where they suspect they will enjoy the food), we started to construct our model by examining only edges wherein both incident nodes had been to at least one common restaurant. In other words, we only examine edges wherein the incident nodes have a non-zero Jaccard Restaurant Similarity score.

Score 1: Jaccard Restaurant Similarity

The Jaccard similarity is calculated by getting the size of the intersection of the two restaurant sets (i.e. the set of restaurants that users A and B have both written reviews of) and dividing this by the size of the union of the same two sets. That is:

$$JaccardSim(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

However, there is an inherent flaw to using Jaccard Restaurant Similarity scores to determine whether an edge should be drawn: two users can have visited and reviewed the same restaurant, but have vastly different ratings of that restaurant. One might enjoy the food and another might find it distasteful. The self-selection into the restaurant is insufficient for capturing true compatibility--compatibility must take how favorable the user rates the restaurant once they've both visited. To account for the ratings, we derived a Rating Similarity score from the Jaccard Restaurant Similarity score.

Score 2: Rating Similarity

This term calculates how similar the ratings are of restaurants that both users have rated. The rating similarity is calculated by getting the intersection set of the two restaurant sets, and then comparing the star ratings of each of the reviews in the intersection set.

$$RatingSim(A,B) = \sum_{r \in R_A, R_B} (2 - |r_A - r_B|) \left(\frac{|r_A - 3|^2 + |r_B - 3|^2}{2} \right)$$

- $2-|r_a-r_b|$: Calculates the difference between the reviews given to the same restaurant, and normalizes the difference around 3, a neutral rating. So, a star rating of 1 is normalized to -2, and a star rating of 5 is normalized to +2. This penalizes reviews of largely different ratings.
- $((|r_a-3|^2+|r_b-3|^2)/2)$: Finds the average distance of the star ratings from normal (3), and gives exponentially more weight to more extreme ratings. So, a shared rating of 5 for a given item is more indicative of similarity than a shared rating of 4. Essentially, we are calculating r-squared on two data points against the "mean" of a 3 rating.

  In the case that multiple reviews exist for a common restaurant in either of the restaurant sets, then the most recent review will be used in the above calculation.

Notice that Rating Similarity score can be calculated strictly on the same set of edges that have a non-zero Jaccard Similarity because it means that the restaurant set R_A and R_B have at least one shared restaurant r. With this new score, we can actually model true compatibility: users whose average Rating Similarity score exceeds 0.25 have a positive compatibility (they enjoy the same food, with average ratings deviating by no more than 1 star), while those below -0.25 are incompatible (one likes food the other dislikes), and those between 0.25 and -0.25 are neutral. A network can be constructed by drawing all edges on the network where the compatibility is positive.

5. Food Preference Network Model

Although the rating network gives a certain amount of guaranteed-accurate information and insight into food friends and food communities, it is not enough to determine true food taste compatibility between all users because we are constrained by Rating Similarity scores only being calculable for users who have visited the same restaurants. As such, the graph based on Rating Similarity score alone is much sparser than the true network. Hence, we come up with a better model generalizing food preferences using users' most-valued restaurant categories and restaurant attributes.

5.1. Limitations of Rating Network

The Rating Similarity score network is not the complete network representing food relationships between people.

Nodes	30255
Non-Zero Degree Nodes	9522
Edges	37873
Closed Triangles	167723
Frac. of closed triads	0.030233
Connected component size	0.846146
Strong conn. comp. size	0.846146
Approx. full diameter	10
90% effective diameter	4.439134
Density (from formula below)	8.275e-5

$D = \frac{2\|E\|}{\|V\|\,(\|V\|-1)}$	

We observe that the fraction of closed triads is very low and the size of the connected components is similarly low. We can conclude that the density of the Rating Similarity score graph is small, which we confirmed by calculating the edge density of the graph. However, the lack of edges is not as a consequence of incompatibility. Instead, many users that are compatible simply have not reviewed the same restaurants, so they do not have an edge in the Rating Similarity graph. The sparsity of the graph based on Rating Similarity scores leads to over-segmentation of food communities. Thus, while we can be confident that the edges that *do* exist in the Rating Similarity network *are* accurate indicators of shared food taste when present, we also know we are missing edges between compatible users. Hence, we have created another score that more accurately identifies edges between food friends.

5.2. Constructing a Food Network using Attribute Similarity Scores

We want to generalize food compatibility to incorporate general food preferences, including cuisine types (e.g. Italian, Indian, Mediterranean, etc.) as well as other aspects of the food experience (e.g. ambience, delivery, suitability for groups or children, outdoor seating, parking availability, etc.). Business objects in the Yelp data each have these attributes associated with them, so we define a metric for evaluating the importance of each attribute to every user (See AttributePrefCounter in *Section 3.3 Metadata*). More similar users will more often consider the same attributes similarly important, so that attribute will show up more frequently in the businesses they review. Drawing edges based on these attributes gives us a more complete compatibility graph since two users can be evaluated against these attributes even if they've never reviewed the same restaurants. To draw the graph, we compute the following Attribute Similarity score.

Score 3: Attribute Similarity

Businesses each have attributes associated with them, so we define a metric for evaluating the importance of any attribute by its frequency among the elements in the intersection set of the two restaurant sets. More similar users will more often consider the same attributes similarly important.

Example attributes: *{"Take-out": false, "Wi-Fi": "free", "Noise Level": "average", "Takes Reservations": true, "Ambience": {"romantic": false, "intimate": false, "touristy": false}, "Parking": {"garage": false, "street": false, "validated": false, "lot": true, "valet": true}, "Good for Kids": false, "Good For Groups": true, "Price Range": 2}*

$$AttributeSim(A,B) = \left(\sum_{a \in attributes} \min\left(\frac{A_a}{B_a}, \frac{B_a}{A_a}\right) \right)$$

A_a is the importance of attribute a to user A while B_a is the importance of attribute a to user B. Importance of attribute *a* is calculated as the proportion of all businesses that a user has visited that takes on attribute *a*.

The ratio A_a/B_a gives the relative importance of the attribute *a* to user A compared to user B. The larger the ratio, the more compatible the two users are because they give an attribute the same importance. We take the minimum of A_a/B_a and B_a/A_a (which should always be less than 1) because we want to take the more constraining term since the true compatibility is only as strong as the weaker perceived compatibility along any given criteria, the attribute *a*. The average over all attributes is the final attribute similarity score.

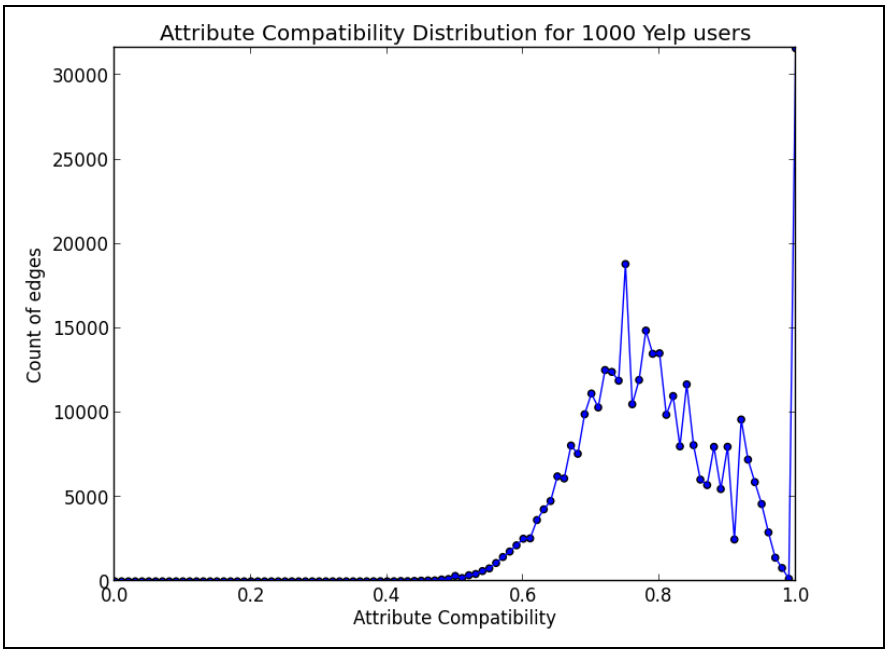
This removes the constraint of the two users only having a calculable food taste Similarity score if they have been to the same restaurants and instead compare the *types* of restaurants the two users regularly visit. In other words, eating at different restaurants that are of the similar type will still result in high compatibility scores.

5.3. Evaluating Accuracy of Food Network

We can see that the edge scores alone do not provide enough resolution to differentiate food communities. Instead of finding small communities with specific tastes, we find large clusters composed of many nodes. This is because these users have been to many restaurants with common attributes and thus have high edge scores. We can tackle this problem by choosing a threshold value for each score. If a pair of nodes have an edge score greater than the threshold, an edge is created between them (i.e. their food tastes are similar *enough* to warrant a food taste connection).

At this point, the biggest unknown is the resemblance of the Attribute Similarity scores to the true network, as modeled by the Rating Similarity scores. Because there are so many attributes, it is uncertain at what threshold of Attribute Similarity we should draw an edge to best create a resemblance to the true network. When using Rating Similarity, the natural threshold of 0.25 determined compatibility because it is the minimum score when two users rate restaurants differently by at most 1 star; we cannot as confidently give such a threshold with the Attribute Similarity score. We can generate a graph to investigate the distribution of compatibility scores over a random generation of 1000 nodes.

Graph: Attribute Compatibility Scores on 1000-node Food Network



 # Non-Zero Degree Nodes = 843, # Edges = 354896
Peak Attribute Compatibility Score = 0.75

We see that the graph is skewed towards the right. Every pair of users have a compatibility score of at least 0.4 because many restaurants have similar attributes, hence many users have similar attribute preferences. Hence, we need to provide more resolution and remove compatibility score values which represent very common attributes and do not necessarily signify a similarity in preference of two users. The peak attribute score is a good benchmark for our threshold value as lower compatibility scores are more likely to be coincidental while higher compatibility scores are better signifiers of food taste similarity.

To determine the appropriate threshold, we can randomly sample edges in our Rating Similarity scores network and calculate the Attribute Similarity score for each pair of incident nodes. We then classify the node based on a spectrum of thresholds, and pick the threshold with the greatest accuracy over many trials of sample edges.

From that, we can infer a true network modeled by the Attribute Similarity score and appropriate threshold with greater confidence. We can subsequently draw edges to form a network to run clustering algorithms on, and this will be more complete because Attribute Similarity scores can be calculated as long as a user has reviewed *any* restaurants, while Rating Similarity requires common restaurants between users.

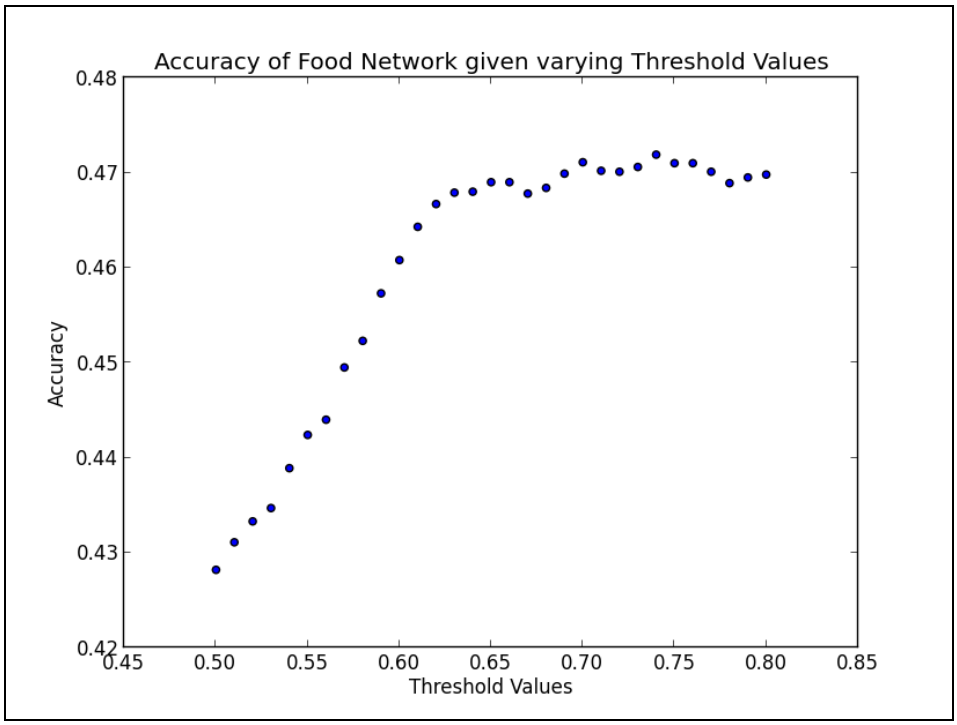
Table: Accuracy Calculation (for any given Attribute Score threshold) on 1000 sampled edges

	Edge with Rating Score	No edge with Rating Score
Edge with Attribute Score	+1/1000 (Hit)	0 (False positive)
No edge with Attribute Score	0 (Miss)	+1/1000 (Correct rejection)

5.4. Placing Threshold Values on Attribute Similarity Scores

A number of edges are sampled from the rating-based network and tested on the corresponding food network (made with the attribute similarity score). We use different threshold values to find which value creates a food network that most closely resembles the rating-based network. This accuracy can be calculated as the number of edges in our test food network that were correctly classified as an edge divided by the sample size (since each of the edges should exist in the true rating network).

Graph: Accuracy of Food Network with a sample of 10,000 edges



The graph above shows that our accuracy increases as the threshold value increases, peaks, and then begins a gradual, inconsistent decline. This is expected because a low threshold means we have a very liberal or loose classification criteria on Attribute Similarity scores that commits many Type I errors (false positives). On the other hand, a high threshold means we have a very conservative or strict classification criteria on Attribute Similarity scores that commits many Type II errors (misses or false negatives). There is a sweet spot between the extremes of liberal and conservative thresholds. From the graph we determined that the maximum accuracy was found at a threshold value of 0.74 with a 47.19% accuracy rate. This is very close to our earlier hypothesis threshold value from the peak value of Attribute Similarity scorefrequencies.

5.5. Final Food Preference Network Model

Hence, we have created the final Food Preference Network Model which creates a food network by investigating every pair of nodes in a Yelp user graph and creating an edge if the Attribute Similarity Score of the pair is above the threshold value of 0.74.

6. Best Food Friends

Finally, we investigate communities of Yelp users who have similar food tastes and who are already friends, henceforth called the 'friend food network'. We construct this graph using only nodes that have friends and create the rating network as before over this real-world network. Using the Food Preference Model on top of the rating network, we finally create a food network that specifically reflects Yelp users who are already friends and have similar food tastes. We will use the Clauset-Newman-Moore clustering algorithm (CNM algorithm) on the generated food network to identify communities with similar food tastes. Ultimately, we are testing our model on a real-world network and employing link prediction techniques to identify best food friends from within the network.

6.1. Clustering on Food Friends

We performed CNM Community Detection on the friend food network as well as on the rating network. The friend food network is expected to have values smaller than the rating network because we are focusing on even more specific food friendships within the rating network to create the friend food network.

Table: CNM Community Detection on Rating Network and Food Network

	Rating Network	Friend Food Network within Rating Network
Number of Communities	1031	533
Average Cluster Size	9.236	2.43
Max Cluster Size	2450	39
Min Cluster Size	2	2
Modularity	0.402	0.983

The table shows us that there are 533 food communities in a rating network of 30,255 nodes and 151,516 edges (see Appendix, Table 1). On average, there are 2.43 members in every food community, with a maximum of 39 and a minimum of 2 members. This shows about a third decrease in the average cluster size from the original rating network. These values are quite reasonable as most Yelp users have a small friend group, and to find a friend within the group who also has the same food taste becomes even more selective, creating smaller communities.

Table: Girvan Newman Community Detection on Rating Network and Food Network

	Friend Food Network within Rating Network
Number of Communities	534
Average Cluster Size	2.42
Max Cluster Size	32
Min Cluster Size	2
Modularity	0.982

We can see that running community detection using the Girvan Newman algorithm gives us very similar values.

6.2. Friend Recommendations

By applying the Food Preference model, which has now been trained on the optimal Attribute Similarity Score, we can find food relationships across the entire graph of Yelp users. In this way, the Food Preference model will find relationships that may not already exist in the Yelp user friend communities and recommend these people as food friends to a user. Thus, clusters found in the food network for a set of Yelp users can be used to generate friend recommendations for each user.

Since users can be grouped in a cluster with other users that are not first-degree friends, a consumer product leveraging our algorithm could surface users in their food community cluster but who are not immediate friends as recommended “food soulmates.” Such a product could also perform either community detection algorithm on the immediate friends of a user only to determine which of a user’s friends are best to grab food with (compared to other pastimes, like going to a concert, recreational sports, etc.).

7. Conclusion

The goal of the project is to consistently identify clusters wherein users can both discover new food friends and also identify members from their own friend lists who they may share common food tastes with. To achieve this, we trained a model to take a Yelp user friend network and modified its edges to instead represent food compatibility between users. We take into account the cuisine preference of each user as well as other more subtle restaurant characteristics such as parking, cost, and atmosphere. This model can then be applied to an entire graph of users and find food friend connections that are not limited to a user’s friend community, thus making new friend recommendations based on food taste similarity.

Our food network (made through Attribute Score calculations) was able to model the User Friend Network (made through Rating Scores) to within an accuracy level of 46%. It is key to note that the User Friend Network does not actually represent food taste similarity as friendship connections can exist without having the same food preference and so 100% model accuracy would not actually be representative of a well-functioning Food Network Model.

The model can be used to allow Yelp users across different areas to connect based on food similarity. Moving forward, other models can be created using different scores, such as the geographic location of restaurants that the users frequent. Food connects people. We hope that with this model more people can form friendships and enjoy food together. After all, the fastest way to a new friend’s heart is through the stomach!

8. Works Cited

Clauset, Aaron, M.E.J. Newman, and C. Moore. “Finding Community Structure in Very Large Networks.” (2004): n. pag. Web. 7 Dec. 2014. <http://www.ece.unm.edu/ifis/papers/community-moore.pdf>.

D. Liben-Nowell, J. Kleinberg. The Link Prediction Problem for Social Networks. Proc. CIKM, 2003. <http://snap.stanford.edu/class/cs224w-readings/nowell04linkprediction.pdf>.

Newman, M. "Fast Algorithm for Detecting Community Structure in Networks." *Physical Review E* 69.6 (2004): n. pag. Web. 12 Oct. 2014. <http://http://arxiv.org/pdf/cond-mat/0309508.pdf>.

Ng, Andrew Y., Michael I. Jordan, and Yair Weiss. "On Spectral Clustering: Analysis and an Algorithm." *NIPS* (2001): n. pag. Web. 11 Oct. 2014.

Rodriguez, A., and A. Laio. "Clustering by Fast Search and Find of Density Peaks." *Science* 344.6191 (2014): 1492-496. Web. 12 Oct. 2014.

9. Appendix

Table 1: Statistics on Rating Network

Ground Truth/Evaluation Network Statistics	
Nodes	*30255*
Edges	*151516*
Zero Deg Nodes	0
Unique undirected edges	*151516*
Closed triangles	*492166*
Open triangles	*17273862*
Frac. of closed triads	*0.027703*
Connected component size	*0.957759*
Strong conn. comp. size	*0.957759*
Approx. full diameter	*11*
90% effective diameter	*4.875198*

10. Individual Team Member Contributions

(Please distribute scores equally across team members.)

Angela: Data Preparation, Creation of Meta-Data, Attribute Score Calculation and creation of Distribution graphs, Food Network Generation, Clustering on networks

Blanca: Creation of Meta-Data, Rating Score Calculation, Evaluation Network Generation, Food Preference Model

Larry: Jaccard Similarity Calculation, Food Network Accuracy, Rating Network Model, Threshold Values on Attribute Score