Analyzing Grocery Market Baskets and Recipes

CSC 466 Group Project

Team:
Jadyn Ellis, Alexander Arrieta, Christine Widden, Ryuhei Shida

Abstract.
This report introduces a data-centric approach to recommending additional items for grocery purchases by shoppers utilizing *Food.com* and *Instacart* data that comprises 11,000 unique ingredients over 178,000 unique recipes, and 3,000,000 grocery orders, respectively. The association rule mining section of our report results in insights into ingredient relationships, with a large focus on baking ingredients. The hierarchical clustering algorithm results in a total of 40 clusters when run on a subset of the grocery orders, revealing commonalities between sets of grocery orders in a given cluster. The findings of these two subproblems lay the ground for the development of our recommendation system that personalizes recommendations for additional item purchases of a shopper based on their grocery item preferences.

Introduction.
The goal of our analysis is to create a data-centric approach to recommending additional items to purchase for grocery shoppers. We believe it is necessary to first develop association rules between ingredients in given recipes in order to form an idea of what ingredients tend to be seen together, and what ingredients imply others in a recipe. We plan to use this information to recommend additional ingredients such that the shopper has more options for recipes to make. For example, if a shopper has purchased flour and eggs, and we discovered a recipe association rule {flour, eggs} → {sugar}, we would recommend an additional item of sugar. Another possible thing a shopper may want is actual recipe recommendations based on their current items, thus we will look at which recipes most closely match the items currently in the basket. Additionally, we desired to develop clusters of items that tend to be purchased together in grocery orders for further information on the tendencies of grocery orders and any other ingredients someone may desire to purchase regardless of useability in a recipe.

Dataset Information.
We utilized two datasets during our process. The first was sourced from *Kaggle*[1] and contains information on *Food.com* recipes, and the ingredients necessary for them. It contains over 11000 unique ingredients which were mapped down to 8000 unique IDs. Multiple different food items were mapped to the same ID. For example, both "Romaine Leaf Lettuce" and "Iceberg Leaf Lettuce" got mapped to "Lettuce". The list of ingredients for the 178,000 recipes in the dataset uses these IDs. We used this dataset to create ingredient market baskets to analyze and mine association rules.
The second dataset was also sourced from Kaggle[2] and contains Instacart grocery orders. This dataset was used to make clusters of common sets of grocery items. It contains over 3 million different Instacart orders, some of

[1] https://www.kaggle.com/datasets/shuyangli94/food-com-recipes-and-user-interactions?select=ingr_map.pkl
[2] https://www.kaggle.com/c/instacart-market-basket-analysis/data?select=departments.csv.zip

which are over 100 items large. The number of unique items is also staggering as the dataset differentiates between nearly identical products from specific brands, and there are also non-food items present. During both processes, we will obtain information that will aid in our creation of a recipe recommender engine.

Data Cleaning.
Before performing any of our analyses it was necessary to retrieve datasets in the format that our algorithms expect. First off, we needed to strip the ingredient information from the *Food.com* dataset to construct our version of market baskets to use in our analysis, based on ingredient IDs. Because not all ingredients in the *Food.com* dataset are also ingredients in the *Instacart* dataset, we converted any mismatched food names to use their *Instacart* counterparts according to whether or not the *Instacart* term was contained within the *Food.com* term as a whole word. "Baby Kale" would be converted to "Kale", "Pale Ale" would be converted to "Ale", etcetera. This did, unfortunately, lead to circumstances such as both "Ginger Ale" and "Pale Ale" being mapped to just "Ale" but due to the time frame of the project and the complexity of this problem, we did not attempt to preserve exact semantic integrity in the ingredients, as this issue was not overly common within the data.

These words were then mapped to their ingredient ID from the *Food.com* dataset so that our final resultant datasets contained only integer numbers for practical memory and runtime reasons. Thus we had one dataset where each row represented the list of ingredients in a recipe and another where each row represented a real market basket of an Instacart user, pruned down to only food items found in the *Food.com* dataset. Since the IDs of both datasets matched we could easily make inferences across datasets. For example, if we saw one row (5,6,7) in the recipes dataset and the same row (5,6,7) in the market baskets dataset, we could know that in that market basket, we could produce the recipe that needs ingredients 5,6,7, which we would map back to real names using the provided mapping in the original *Food.com* dataset, and ultimately the original recipe it came from.

Additionally, we noticed that there were certain replaced values that were not informative of the ingredient name within the recipe, such as flours being labeled as "flmy" or "all-purpose flmy".

Analytical Goals.
The ultimate goal of our data analysis is to create a recommender engine for the purchase of additional goods based on recipes one may be seeking to make. Before building the recommender engine, we must address a few subproblems to uncover the true nature of recipes and grocery market baskets. Our first subproblem is to develop association rules of recipe ingredients to discover which ingredients tend to be used together, and what ingredients imply others. Our second subproblem is to cluster sets of grocery market basket purchase items to find common groupings of grocery orders and what items they contain. Both of these subproblems will aid in our development of a recommender engine which will be inspected to see if an ingredient association rule developed from subproblem one applies and if it is similar to a known recipe ingredient list. We also wanted to look at the similarity to other user's market basket items in forming the recommendation but that ultimately did not prove to be useful.

Retrieval of Recipe Association Rules.

I. Methods

We decided to mine the ingredient market basket association rules utilizing the Apriori algorithm methodology. The Apriori algorithm discovers frequent itemsets within a market basket given hyperparameter minimum support, *minSup*, based on the Apriori principle:

If X is a frequent itemset in T, then every non-empty subset of X is also a frequent itemset in T[3]

This principle allows early pruning of candidate frequent itemsets, which in turn decreases the runtime necessary to discover frequent itemsets. For example, if we know that the set {14, 15, 16} is not frequent, then we know that sets {14}, {15}, and {16} are also infrequent based on minimum support. Moreover, we decided only to discover "skyline" frequent itemsets, which are frequent itemsets that are not subsets of any other frequent itemsets. This is mainly in the interest of constrained output in our algorithm.

When discovering association rules, we additionally implemented another hyperparameter, minimum confidence (*minConf*), so that we discover strong relationships between items. Thus, *minSup* and *minConf* are paired such that we discover association rules that commonly occur and that are strongly relational. Additionally, to further constrain our output, our mined association rules will only contain one item on the right side, since by rule, if {A, B, C} → {D}, then {A, B, D} → {C}, {A, B} → {C, D}, and so on.

II. Results

When developing association rules with a minimum support value greater than 0.01, we discovered that we were unable to obtain more than three association rules, which were not very informative about the ingredient sets in our dataset. Thus, we decided to implement our algorithm with a minimum support of 0.01. Additionally, with the same reasoning, we utilized a minimum confidence value of 0.7, such that we aren't including weak relationships, but also achieving more informative and well-rounded results. Finally, for just this section, we decided to omit any association rules in which the left side only contained one value, for the sake of constrained output while demonstrating how this algorithm is utilized late on in our recommender. Our developed association rules can be found in the Appendix, Section 1.

Most of our developed association rules utilizing *minSup = 0.01, minConf = 0.7* contain ingredients commonly found in baking recipes, such as {egg, baking soda, all-purpose flour} → {salt} (*support = 0.015, confidence = 0.8286)* and {vanilla, sugar, flour} → {egg} (*support = 0.014, confidence = 0.8413)*. This would make sense as these ingredients tend to be the base for most baking recipes, with a few alterations to others to make them unique. Additionally, we discovered other trivial association rules between sets of ingredients, such as {olive oil, pepper} → {salt} (*support = 0.0196, confidence = 0.8113)* and {celery, carrot} → {onion} (*support = 0.0107, confidence = 0.7289)*. The association rule with the largest confidence value was {butter, baking powder, baking soda} → {egg} with a confidence of 0.8858, while the association rule with the largest support was {onion, pepper} → {salt} with a support of 0.0358. Overall, with these *minSup* and *minConf* thresholds, our developed association rules are optimal at detecting baked good rules, while with more savory dishes, it is quite poor at developing association rules.

[3] https://users.csc.calpoly.edu/~dekhtyar/466-Fall2023/lectures/lec03.466.pdf

Development of Clusters of Grocery Items.

I. Methods

We decided to cluster sets of grocery purchases utilizing hierarchical clustering, where a given "point" represents a full grocery order. Moreover, we performed our clustering analysis on a random subset of our *Instacart* data, as the original size of 3,000,000 rows resulted in high runtime and memory errors. We believed a random subset of the data would allow us to develop general ideas of the contents of each cluster. Once we formed the complete dendrogram of clusters, we implemented a cutoff threshold of 0.8, resulting in a total of 40 clusters of grocery orders. Additionally, when merging clusters, we used dice distance, which ranges from 0 to 1, to determine what two clusters are closest to one another.

$$D(P,\ Q)\ =\ \frac{2|P \cap Q|}{|P|\ +\ |Q|}\ where\ P\ and\ Q\ are\ grocery\ orders$$[4]

II. Results

Our hierarchical clustering algorithm with a cutoff threshold of 0.8 for dice distance resulted in a total of 40 clusters when run on a subset of 100 rows of the data. We discovered that 15/40 clusters only contained a singular grocery order, which does not tell us much about those certain clusters nor the similarities to other orders. Our largest cluster contained 7 total grocery orders, with every order containing water and the majority containing fruits such as pineapple, mango, or bananas. Another larger cluster of 6 points had item commonalities of avocados, ice, and fats while another 6-point cluster had items of gluten, broccoli, coconuts, and almonds in almost all grocery orders. Furthermore, we discovered that there were multiple clusters in which each grocery order contained multiples of items, such as in cluster 2, where one grocery order contained 3 pizzas and the other contained 2 pizzas. In cases such as these, it may be helpful to a grocery shopper to suggest duplicates of certain items. The full list of clusters and cluster contents can be seen in Appendix Section 2 at the end of this report. We did manage to run the code successfully on larger portions of data such as one thousand rows, but the output quickly became hard to interpret without finding an algorithmic approach to categorizing the clusters, which we did not have the time for. This also meant that it did not provide any easy way to incorporate the gained information into the recommendation system thus it was left out of that process from this project.

Creating a Recommendation System.

I. Methods

Given a table of all found association rules and a market basket, we can recommend other items that the user may wish to purchase. To generate these recommendations, all association rules where the items on the left are a subset of the items in the market basket are identified. Then, all items from the right-hand sides of these associations are collected, items already in the market basket are removed, and the remaining items are presented to the user as recommendations. This methodology was derived from the idea of information retrieval and matching a query to a document, just in this case our query was a market basket, and our document was the ingredients of a recipe. The small number of items in both the query and document meant the use of tf idf or other ways to weight information was unneeded and a simple percentage equality comparison was sufficient.

---

[4] https://towardsdatascience.com/17-types-of-similarity-and-dissimilarity-measures-used-in-data-science-3eb914d2681

II. Example of Use

Given the market basket of milk, water, butter, eggs, and sugar, the following association rules and recommended items are produced:

Association Rules	
Left	**Right**
{'water'}	{'soy sauce'}
{'sugar'}	{'garlic clove'}
{'milk'}	{'water'}
{'butter'}	{'garlic clove'}
{'sugar', 'butter'}	{'cinnamon'}
{'butter', 'milk'}	{'all-purpose flour}
{'water', 'butter'}	{'salt'}
{'water', 'sugar'}	{'salt'}
{'sugar', 'butter', 'milk'}	{'salt'}

Overall, garlic cloves, salt, soy sauce, cinnamon, all-purpose flour would be recommended purchases for this user. Considering that milk, butter, eggs, and sugar are all heavily associated with baking, it would then make sense to recommend other baking ingredients such as salt, cinnamon, and flour. Butter is a very common ingredient in sauces, so connecting it to garlic cloves, another common sauce ingredient, makes sense. The association from sugar to garlic cloves is the most unusual association rule here, but this could again be due to sauces, which sometimes incorporate sugar.

Conclusions.

For the development of association rules, we found that rules tended to contain items commonly used in baked goods, such as flour, sugar, eggs, and baking powder. Overall, we believed that this made sense, considering most baked items contain a similar base with a few alterations, while savory dishes tend to contain more unique sets of ingredients. Thus, our recommendation system, which utilizes these association rules, is heavily biased in recommending additional items pertaining to baking recipes. During clustering of grocery market baskets, we discovered certain tendencies of shoppers to purchase multiples of the same item in their purchases, such as purchasing two packages of chicken. Additionally, each cluster had random similarities between items, such as gluten, coconuts, broccoli, and almonds, which didn't prove to be all that useful when developing our recommendation system. During the clustering process, we additionally ran into memory and runtime issues, which are further explained in the *Limitations* section below. While the results from our recommendation system may be more simple at its foundation, we believe with further development this could prove to be a useful tool for grocery shoppers in the future.

Limitations.

The size of the data is far beyond our ability to process it. Luckily due to the principles of the Apriori algorithm, most combinations of ingredients are quickly removed as potential frequent itemsets. Yet, due to just the sparsity of the data it meant that even with just a minimum support of .01 and no bound on confidence, only 280 rules were produced. For clustering, we need to produce many sets of distances between the Instacart baskets. For the full set of distances (a 3 million by 3 million array), numpy needs over 70 terabytes of storage which we simply do not have access to. Additionally, to only produce subsets of distances at a time would be incredibly time-consuming without massive computational power which we also do not have access to. One last limitation we encountered was the lack of ground truth in associating the market baskets with recipes, as we utilized unsupervised learning.

Sources.

Food.com Recipes and Interactions
https://www.kaggle.com/datasets/shuyangli94/food-com-recipes-and-user-interactions?select=ingr_map.pkl

Instacart Market Basket Analysis
https://www.kaggle.com/c/instacart-market-basket-analysis/data?select=departments.csv.zip

Apriori Algorithm Lecture
https://users.csc.calpoly.edu/~dekhtyar/466-Fall2023/lectures/lec03.466.pdf

Information on Dice Similarity
https://towardsdatascience.com/17-types-of-similarity-and-dissimilarity-measures-used-in-data-science-3eb914d2681

Appendix.

Section 1.

Left	Right	Support	Confidence
{'butter'}, {'baking powder'}, {'baking soda'}	{'egg'}	0.011096	0.885803851
{'vanilla'}, {'flour'}, {'baking soda'}	{'egg'}	0.01077	0.880330124
{'butter'}, {'sugar'}, {'salt'}, {'baking soda'}	{'egg'}	0.010905	0.879240163
{'butter'}, {'flour'}, {'baking soda'}	{'egg'}	0.012493	0.878500986
{'vanilla'}, {'sugar'}, {'baking powder'}	{'egg'}	0.01008	0.866441659
{'vanilla'}, {'sugar'}, {'baking soda'}	{'egg'}	0.01031	0.866163996
{'vanilla'}, {'salt'}, {'baking soda'}	{'egg'}	0.013923	0.862404448
{'vanilla'}, {'baking powder'}, {'salt'}	{'egg'}	0.01224	0.854011742
{'flour'}, {'pepper'}	{'salt'}	0.010984	0.853903184
{'butter'}, {'sugar'}, {'flour'}, {'baking powder'}	{'egg'}	0.010759	0.850554324
{'sugar'}, {'flour'}, {'salt'}, {'baking soda'}	{'egg'}	0.012493	0.850324551
{'cinnamon'}, {'salt'}, {'baking soda'}	{'egg'}	0.010933	0.849237473
{'flour'}, {'baking powder'}, {'baking soda'}	{'egg'}	0.010967	0.846686877
{'vanilla'}, {'flour'}, {'salt'}	{'egg'}	0.014047	0.846517918
{'butter'}, {'sugar'}, {'baking powder'}, {'salt'}	{'egg'}	0.013839	0.846312178
{'sugar'}, {'baking powder'}, {'salt'}, {'baking soda'}	{'egg'}	0.011113	0.845858241
{'butter'}, {'baking powder'}, {'all-purpose flour'}	{'egg'}	0.01026	0.844413666
{'vanilla'}, {'sugar'}, {'flour'}	{'egg'}	0.013682	0.841324595
{'brown sugar'}, {'salt'}, {'baking soda'}	{'egg'}	0.013328	0.839279407
{'butter'}, {'vanilla'}, {'flour'}	{'egg'}	0.012122	0.838897516
{'egg'}, {'pepper'}	{'salt'}	0.015011	0.83625
{'butter'}, {'flour'}, {'baking powder'}, {'salt'}	{'egg'}	0.011309	0.836167565
{'all-purpose flour'}, {'salt'}, {'baking soda'}	{'egg'}	0.015382	0.832169954
{'butter'}, {'pepper'}	{'salt'}	0.022108	0.829858918
{'egg'}, {'all-purpose flour'}, {'baking soda'}	{'salt'}	0.015382	0.828649139
{'sugar'}, {'egg'}, {'baking powder'}, {'baking soda'}	{'salt'}	0.011113	0.828177258
{'sugar'}, {'flour'}, {'baking powder'}, {'salt'}	{'egg'}	0.014546	0.828169914
{'sugar'}, {'baking powder'}, {'milk'}, {'salt'}	{'egg'}	0.010131	0.824657534
{'vanilla extract'}, {'baking soda'}	{'egg'}	0.010008	0.820229885
{'cinnamon'}, {'egg'}, {'baking soda'}	{'salt'}	0.010933	0.816164154
{'egg'}, {'baking powder'}, {'all-purpose flour'}	{'salt'}	0.017334	0.813800369
{'sugar'}, {'baking powder'}, {'all-purpose flour'}	{'egg'}	0.010928	0.812005002
{'milk'}, {'pepper'}	{'salt'}	0.011421	0.811478677
{'olive oil'}, {'pepper'}	{'salt'}	0.019634	0.811312007
{'butter'}, {'vanilla'}, {'salt'}	{'egg'}	0.014187	0.810836807
{'vanilla extract'}, {'all-purpose flour'}	{'egg'}	0.010625	0.810440736
{'onion'}, {'pepper'}	{'salt'}	0.035784	0.80879929
{'sugar'}, {'egg'}, {'baking powder'}, {'milk'}	{'salt'}	0.010131	0.806610094
{'brown sugar'}, {'baking powder'}, {'salt'}	{'egg'}	0.010142	0.804986643
{'butter'}, {'baking powder'}, {'milk'}	{'egg'}	0.010664	0.799747581
{'flour'}, {'sugar'}, {'egg'}, {'baking soda'}	{'salt'}	0.012493	0.799067097
{'brown sugar'}, {'egg'}, {'baking soda'}	{'salt'}	0.013328	0.798655462
{'vanilla'}, {'egg'}, {'baking soda'}	{'salt'}	0.013923	0.797557841
{'garlic clove'}, {'pepper'}	{'salt'}	0.018024	0.795100223
{'pepper'}, {'water'}	{'salt'}	0.012829	0.792446292
{'butter'}, {'sugar'}, {'all-purpose flour'}	{'egg'}	0.010518	0.791807432
{'baking powder'}, {'salt'}, {'all-purpose flour'}	{'egg'}	0.017334	0.791495902
{'flour'}, {'baking powder'}, {'milk'}	{'egg'}	0.010125	0.789588801
{'baking powder'}, {'cinnamon'}	{'egg'}	0.012145	0.787845706
{'butter'}, {'sugar'}, {'egg'}, {'baking soda'}	{'salt'}	0.010905	0.785137318
{'butter'}, {'vanilla'}, {'sugar'}	{'egg'}	0.014215	0.782823602
{'butter'}, {'sugar'}, {'flour'}, {'salt'}	{'egg'}	0.014686	0.781959379
{'brown sugar'}, {'egg'}, {'baking powder'}	{'salt'}	0.010142	0.780656304
{'vanilla'}, {'egg'}, {'baking powder'}	{'salt'}	0.01224	0.771570014
{'butter'}, {'sugar'}, {'egg'}, {'baking powder'}	{'salt'}	0.013839	0.769734789
{'sugar'}, {'salt'}, {'all-purpose flour'}	{'egg'}	0.014922	0.769008384
{'sugar'}, {'flour'}, {'milk'}	{'egg'}	0.011303	0.765286745
{'brown sugar'}, {'vanilla'}	{'egg'}	0.012021	0.754843255
{'sugar'}, {'salt'}, {'cinnamon'}	{'egg'}	0.011135	0.752463988
{'butter'}, {'flour'}, {'egg'}, {'baking powder'}	{'salt'}	0.011309	0.746943312
{'butter'}, {'egg'}, {'all-purpose flour'}	{'salt'}	0.015735	0.745217853
{'butter'}, {'vanilla extract'}	{'egg'}	0.012998	0.73672496
{'celery'}, {'carrot'}	{'onion'}	0.010709	0.728904162
{'flour'}, {'cinnamon'}	{'egg'}	0.012958	0.727559055
{'salt'}, {'vanilla extract'}	{'egg'}	0.015527	0.717284271
{'butter'}, {'sugar'}, {'milk'}	{'egg'}	0.013396	0.71497006
{'butter'}, {'flour'}, {'sugar'}, {'egg'}	{'salt'}	0.014686	0.701688555

Section 2.

Cluster 0:
1 Points:
['infant formula'],
Cluster 1:
2 Points:
['parmesan'], ['parmesan'],
Cluster 2:
2 Points:
['pizza' 'pizza' 'pizza'], ['pizza' 'pizza' 'olive'],
Cluster 3:
1 Points:
['roma' 'barbecue sauce' 'hamburger' 'hamburger' 'hamburger'],
Cluster 4:
1 Points:
['peanut' 'tortilla'],
Cluster 5:
1 Points:
['lettuce'],
Cluster 6:
1 Points:
['bacon' 'cereal' 'oatnut bread' 'yogurt'],
Cluster 7:
2 Points:
['kale' 'yogurt' 'rice' 'tofu' 'yoghurt' 'yoghurt' 'yoghurt'], ['kale' 'whole milk' 'banana' 'rice'],
Cluster 8:
7 Points:
['peppermint' 'water' 'water' 'cider' 'banana'], ['cacao' 'lettuce' 'spelt' 'wheat' 'banana' 'onion' 'water' 'seed'], ['coconut' 'pineapple' 'mango' 'berry' 'mango' 'water'], ['roast' 'water' 'red raspberry' 'cinnamon'], ['paper' 'coke' 'steak' 'water'], ['water'], ['water' 'water'],
Cluster 9:
2 Points:
['liquid' 'lemon' 'cloth' 'paper' 'syrup'], ['cookie' 'paper'],
Cluster 10:
1 Points:
['navel orange' 'rhubarb' 'bok choy' 'liquid' 'coconut'],
Cluster 11:
1 Points:
['cherry' 'coke' 'liquid' 'lavender' 'starch' 'hot dog'],
Cluster 12:
2 Points:
['onion' 'green' 'pepper' 'pepper' 'banana' 'bratwurst' 'paper'], ['fat' 'sugar' 'baking powder' 'sea salt' 'pepper' 'green'],
Cluster 13:
2 Points:
['vanilla' 'coffee' 'bagel' 'ground chicken' 'snap pea' 'broccoli' 'strawberry' 'paper' 'arugula' 'tomato' 'green'], ['green' 'fruit' 'coffee' 'coffee'],
Cluster 14:
2 Points:
['corn'], ['corn' 'garlic' 'cheese' 'cream'],
Cluster 15:
3 Points:

['hazelnut' 'cilantro' 'tea' 'corn' 'fig' 'rosemary' 'spinach' 'thyme' 'tarragon' 'dill' 'rosemary' 'green'], ['salt' 'green' 'jalapeno' 'celery' 'hot salsa' 'broccoli' 'basmati rice' 'paper' 'cilantro' 'corn' 'coconut'], ['whole milk' 'corn' 'popcorn' 'cilantro' 'broccoli' 'kale' 'kale' 'celery' 'roma'],

Cluster 16:

3 Points:

['chicken'], ['garlic' 'yoghurt' 'lemon' 'chicken' 'pork' 'milk'], ['cannellini' 'meal' 'fat' 'milk' 'chicken' 'eggplant' 'bread'],

Cluster 17:

2 Points:

['arugula' 'fig' 'prosciutto' 'broccoli' 'fat' 'lemon'], ['lemon' 'olive' 'chicken' 'broccolini' 'sea salt' 'sweet cherry' 'popcorn' 'roasted turkey' 'fat' 'hot sauce'],

Cluster 18:

1 Points:

['fat' 'cucumber' 'mint' 'parsley' 'lettuce' 'corn' 'parmesan' 'vanilla' 'fat' 'fat' 'strawberry' 'mix' 'fillet' 'gluten' 'gluten' 'crostini' 'popcorn' 'vanilla bean' 'blue cheese'],

Cluster 19:

1 Points:

['bread' 'cranberry sauce' 'kefir' 'fat' 'juice' 'cornstarch' 'bosc pear' 'whole milk' 'broccoli' 'potato' 'roma' 'garlic'],

Cluster 20:

2 Points:

['fat' 'vanilla' 'vanilla' 'cinnamon' 'brownie' 'cherry' 'tortilla''salted butter' 'mango'], ['fat' 'tomato ketchup' 'kiwi' 'brownie' 'potato'],

Cluster 21:

6 Points:

['seasoning' 'avocado' 'string' 'kale' 'fat' 'granola'], ['ice' 'fat' 'grain' 'avocado' 'cheddar' 'vinaigrette'], ['ice' 'banana' 'meal' 'fat'], ['mustard' 'banana' 'fat' 'cheese' 'cheese' 'coconut' 'ice' 'fat'], ['fat' 'mix' 'fat'], ['gluten' 'cream' 'fat'],

Cluster 22:

3 Points:

['salami' 'almond' 'cheese' 'wasabi' 'snap pea' 'orange'], ['almond' 'tofu' 'bar'], ['almond'],

Cluster 23:

6 Points:

['gluten' 'broccoli' 'berry' 'gluten' 'cucumber' 'oil' 'cream' 'almond' 'brown rice' 'milk' 'roll' 'water'], ['cookies' 'spinach' 'ham' 'lime' 'chocolate' 'coconut' 'kiwi' 'string' 'cream' 'broccoli' 'gluten' 'chocolate' 'whole milk' 'almond' 'strawberry' 'whole chicken' 'fat' 'muffin'], ['potato' 'gluten' 'gluten' 'kale' 'onion' 'coconut' 'coconut' 'ketchup' 'mix' 'orange' 'lemon'], ['seed' 'jalapeno' 'corn' 'orange' 'coconut' 'water' 'coconut' 'green' 'gluten' 'american cheese' 'onion' 'date' 'corn' 'kefir' 'almond' 'fat' 'cereal' 'broccoli' 'zucchini' 'broth' 'tomato' 'seasoning' 'parmigiano'], ['lemon' 'tomato' 'gluten' 'orange' 'juice' 'almond' 'almond' 'paper' 'banana' 'coconut' 'green' 'ice'], ['gluten' 'banana' 'gluten' 'sage' 'broccoli' 'almond' 'rice' 'coconut'],

Cluster 24:

1 Points:

['coffee' 'lemon' 'macaroni' 'spinach' 'bean' 'spinach' 'tzatziki'],

Cluster 25:

5 Points:

['spinach' 'squash' 'string' 'sea salt' 'colby' 'fat' 'peach' 'basil' 'apple'], ['squash' 'grape' 'lemon' 'spinach' 'almond' 'basil' 'ginger'], ['spinach' 'tomato' 'goat cheese' 'cereal' 'burrito' 'banana'], ['banana' 'coffee' 'spinach' 'yam' 'almond' 'raspberry'], ['banana' 'pepper' 'apple' 'cucumber' 'spinach' 'almond' 'seed' 'cinnamon''ginger' 'tomatillo'],

Cluster 26:

5 Points:

['apple' 'tea' 'vitamin'], ['whole milk' 'sugar' 'apple' 'pineapple' 'garlic' 'water' 'tea'], ['burrito' 'pasta sauce' 'apple' 'diet soda'], ['lettuce' 'roma' 'apple' 'green'], ['green' 'apple' 'roll' 'candy'],

Cluster 27:

3 Points:

['apple' 'paper' 'spinach' 'caramel' 'hazelnut' 'water' 'water' 'cola' 'chicken'], ['corn' 'avocado' 'cotija cheese' 'lime' 'water' 'lime' 'blueberry' 'spinach' 'strawberry' 'kale' 'apple' 'coconut' 'cheese' 'smoked turkey' 'pizza' 'olive'], ['lime' 'water' 'banana' 'raspberry' 'whole milk' 'strawberry' 'string' 'apple' 'bartlett pear'],

Cluster 28:
2 Points:
['chicken' 'butter' 'string' 'cheese' 'pepper' 'green' 'jalapeno' 'cilantro' 'lemon' 'ginger' 'garlic' 'spinach' 'parmesan' 'lettuce'
'lettuce' 'cucumber' 'roma' 'squash' 'zucchini' 'squash' 'tofu' 'green'
'tofu' 'tofu' 'sausage' 'ground turkey' 'broth' 'broth' 'salt'
'cauliflower' 'eggplant' 'harissa' 'basil' 'masala' 'kale' 'butter'
'milk' 'tomato' 'squash' 'avocado' 'homogenized milk' 'hearts of palm'], ['cider' 'green' 'grain' 'cucumber' 'chicken' 'colby' 'apple' 'grain'
'peach' 'turkey' 'banana' 'squash' 'baby food' 'puree' 'puree'
'bartlett pear' 'lemon' 'blueberry' 'broccoli' 'masala' 'apple' 'lime'
'apple' 'kiwi' 'cheese' 'roma'],
Cluster 29:
1 Points:
['strawberry' 'avocado' 'raisin' 'burrito' 'honey' 'gluten' 'berry' 'honey' 'mexican rice' 'apple' 'roasted turkey' 'american cheese' 'cola'
'coke' 'pancake' 'pretzel' 'sugar' 'syrup' 'mix' 'salsa' 'cream' 'cheddar'],
Cluster 30:
6 Points:
['coconut' 'fat' 'lemon' 'green' 'grain' 'basil' 'vanilla' 'gluten' 'cookie' 'chicken' 'blueberry' 'cheddar' 'apple' 'whole milk'], ['beef' 'ginger'
'apple' 'cookie' 'lemon' 'avocado'], ['grain' 'apple' 'peanut' 'blackberry' 'water' 'okra' 'apple' 'almond' 'concord grape' 'ravioli' 'ginger'],
['cucumber' 'cheddar' 'peanut' 'banana' 'almond' 'gluten' 'gluten' 'masala' 'jasmine rice' 'popcorn' 'peach' 'kale' 'broth' 'apple'
'roasted turkey' 'tomato' 'basil' 'cheese' 'corn'], ['water' 'spinach' 'almond' 'gluten' 'cheese' 'fat' 'avocado' 'berry' 'butter' 'cucumber'
'arugula' 'sesame' 'grain' 'basil' 'peanut' 'beef' 'apple'], ['cucumber' 'mint' 'cheddar' 'ham' 'almond' 'gluten' 'peanut' 'muenster cheese'
'watermelon' 'cinnamon' 'pretzel' 'avocado' 'cheese' 'apple' 'fat' 'grain' 'arugula'],
Cluster 31:
1 Points:
['butter' 'vitamin' 'bread' 'onion' 'heavy whipping cream' 'parsley' 'onion'],
Cluster 32:
1 Points:
['apple' 'honey' 'greek yogurt' 'honey' 'cheese' 'cheese' 'vitamin' 'fat' 'fat' 'milk' 'milk' 'raisin'],
Cluster 33:
1 Points:
['banana' 'vitamin' 'spice' 'potato' 'liquid' 'honey' 'liquid' 'salted butter' 'cheese' 'buttermilk' 'peanut' 'green' 'bread' 'vegetable' 'ice' 'ice'
'cream' 'tomato'],
Cluster 34:
3 Points:
['honey' 'wheat' 'greek yogurt' 'liquid' 'broccolini' 'avocado' 'chicken' 'cabbage' 'liquid' 'american cheese' 'liquid' 'lettuce'], ['milk'
'avocado' 'honey'], ['guacamole' 'wheat' 'avocado'],
Cluster 35:
1 Points:
['garlic' 'ginger' 'lime' 'lemon' 'watermelon' 'potato' 'gnocchi' 'pasta sauce' 'mayonnaise' 'olive' 'avocado' 'smoked chicken' 'gluten'
'gluten' 'lemon verbena' 'mint'],
Cluster 36:
5 Points:
['grapefruit' 'plain yogurt' 'lemon' 'cheese' 'avocado' 'garlic' 'lemon' 'salted butter' 'spinach'], ['salmon' 'whole milk' 'plain yogurt' 'cheese'
'whole milk' 'green' 'persimmon' 'garlic' 'parsley' 'dill'], ['zucchini' 'garlic' 'lemonade' 'avocado' 'onion' 'rosemary' 'sugar' 'unsalted butter'
'mushroom' 'corn' 'ice'], ['whole milk' 'veggie burger' 'lemon' 'spelt' 'tomato' 'spinach' 'quinoa' 'wheat' 'avocado' 'ice' 'coconut'],
['cilantro' 'green' 'garlic' 'ginger' 'lemon' 'cauliflower' 'onion' 'lettuce' 'water' 'whole milk' 'greek yogurt' 'wheat' 'avocado' 'ice'],
Cluster 37:
2 Points:
['fruit' 'cola' 'sea salt' 'beef'], ['beef' 'tea' 'turkey pepperoni' 'tea' 'ice']
Cluster 38:
2 Points:
['strawberry' 'grape' 'spinach' 'tea' 'cocoa' 'chocolate' 'broccoli' 'cottage cheese' 'chicken' 'roasted turkey' 'pizza' 'tuna'],

['cocoa' 'curd' 'tea']
Cluster 39:
2 Points:
['mix' 'curd' 'blueberry' 'strawberry' 'marinara sauce' 'broccoli'], ['strawberry' 'cheese' 'herb']