From Baskets to Networks: Uncovering Consumer Patterns in Instacart Shopping

Lucy Wang, Chris De Martinis, Kiley Price

1. Research Question

1.1 Problem Definition

Market basket analysis is a data mining technique of searching for meaningful associations in customer purchase data (Raeder & Chawla 2010). Although identifying association rules can lead to insightful discovery and has had proven success, this approach comes with some limitations. Association rules are often incredibly prolific and redundant, making it difficult to discern truly interesting, novel relationships (Raeder & Chawla 2010). We plan on expanding upon the traditional application of MBA by modeling transaction data via a product network and using skills garnered from this course to refine and assess product relationships. Additionally, we aim to detect communities of products within our product network and leverage these findings to create a model that may recommend what a customer is likely to purchase based on the items in their basket.

1.2 Motivation

Market basket analysis allows retailers to develop a better understanding of individual purchasing habits by associating customers with transactions. Identifying these patterns can promote data-driven product placement decisions, assist in designing personalized marketing campaigns, and inform the timing and extent of product promotions. Market basket analysis can also help a company understand purchasing habits to inform inventory management. When and how much of a product to have in stores and what a store should carry or get rid of are all additional benefits to a business for a market basket analysis.

2. Related Work

This market basket analysis focuses on a "next basket" approach which relies on implicit information only, purchases or clicks over time (Hu et al., 2008), rather than implicit information, ex. a rating system (Linden et al., 2003). There are different types of next basket methods: general, sequential, pattern-based, and hybrid, a combination of general and sequential. General techniques focus on a customer's preference, sequential focuses on recent purchases of a customer, pattern-based uses associations of products across all customers, and hybrid is a combination of general and sequential techniques (Guidotti et al., 2019). Some new techniques for pattern-based methods include classifying all users and only focusing on customer's data to make predictions about her next basket (Cumby et al., 2004) and co-occurrence of basket items which requires part of the next basket to make predictions (Guidotti et al., 2017).

Applications of market basket analysis include cross-selling, customer behavior analysis, website analysis and decision support (Pradhan et al., 2022). Cross-selling refers to co-purchasing activities and is influenced by shelf-space placement and specific customer targeting as in Chen et al. (2006) and Kim & Street (2004). Customer behavior analysis looks specifically at how and when customers purchase items which then in turn can produce product recommendations (E. Kim et al., 2003; Kwan et al., 2005; Liu & Shih, 2005). Website analysis looks to better inform website design to better manage customer preferences and relations (Albert et al., 2004; Tam &

Ho, 2005). MBA data extraction, implementation, and warehousing all lead to better decision support as evidenced by Kumar et al. (2007) and Lin et al. (2003).

One algorithm used in market basket analysis is from Aldino et al. (2021) which explored the performance of the Apriori and FP-Growth algorithms in mining transaction data using RapidMiner, offering insights into their efficacy in identifying associations between items. The study found that while the Apriori algorithm follows an iterative approach—where a k-itemset helps to generate (k + 1) - itemsets—the FP-Growth algorithm constructs a tree structure that can yield faster results for large datasets with a high number of transactions. Both methods assess the "interestingness" of association rules based on minimum Support and Confidence thresholds, ensuring that only relevant rules are highlighted. However, as Raeder and Chawla (2010) noted, the sheer volume of generated rules can result in redundancy, making it difficult to identify truly insightful associations within the data. To address the limitations of traditional association rule mining, researchers have proposed various enhancements. One approach involves modeling transactions as product networks, where nodes represent products and edges indicate co-purchase relationships. This network-based method facilitates the detection of communities or clusters within the product space, potentially revealing more intricate relationships that are often obscured by frequent itemset mining. Such clustering techniques allow for a more refined analysis of product interactions and have proven effective in creating targeted marketing strategies and product recommendations.

3. Data

3.1 Data Source

Several datasets are available for market basket analysis primarily for grocery store, food, and retail purchases. Kaggle has several datasets with one being for purchases at a bakery (Mittal) as well as a dataset for Instacart orders (Instacart). Another dataset on Github also contains data for grocery shopping (Market) as well as another dataset from Stanford SNAP on Amazon purchase data (Stanford University SNAP).

The Kaggle and Github datasets contain individual customer purchases and what items are in each purchase. These datasets with some manipulation can be turned into a network if needed. The Amazon purchases dataset contains an undirected network of frequently co-purchased items. Communities with less than three nodes were removed from the dataset. The Instacart dataset was selected for this analysis due to additional information provided that may be useful for further analysis and validation which is explained below.

3.2 Data Collection

The data for our analysis comes from Instacart. Instacart collects extensive transactional data that captures various details about customer shopping behaviors. This data includes product selections, shopping patterns, and repeat purchases, enabling Instacart’s data science team to create models that predict user preferences such as likelihood of repurchasing an item, trying for the first time, or adding to cart in the next session.

To facilitate research and innovation in predictive modeling, Instacart has open-sourced anonymized data on over 3 million orders. This dataset, available through a Kaggle competition, includes order histories for multiple users, capturing patterns across different types of products

and orders over time. Key attributes in the dataset include order IDs, product names, aisle IDs and department numbers, providing a robust basis for investigating customer purchase behaviors and refining models for market basket analysis.

Our project leverages this rich dataset to identify patterns and communities within product networks, aiming to deepen the understanding of consumer shopping habits and develop actionable insights for personalized grocery recommendations.

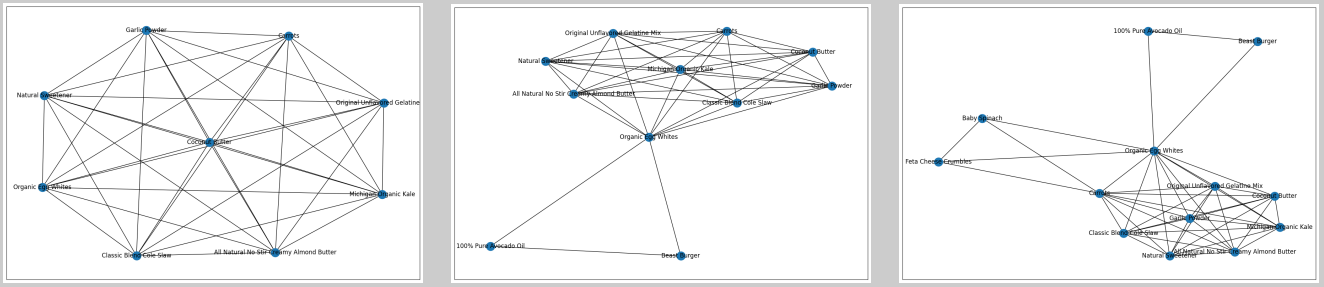
3.3 Data Analysis

We will now perform some initial exploratory analysis on the Instacart datasets to obtain a better understanding of the scope and structure of this data. The dataset provides product purchasing information for 4 to 100 orders for each customer. Each order contains the number of the customer's order (ranging from 1 to 100) alongside the day of the week and hour of the day each order was placed. The order data spans 3,421,083 orders across 206,209 Instacart users. On average, a single order includes 10 items but the number of items included in an order ranges from 1 to 145. InstaCart also provides department and aisle information; the products purchased in this dataset can be classified into 21 departments and 134 grocery aisles. In considering our plans to build a recommendation model, we decided to perform a stratified split based on customers and order number to achieve a prior/training and testing dataset which we will further explain as we discuss our methods. Instead of reserving some customers for training and testing, we included all Instacart users in our validation dataset but excluded each user's most recent order (as identified by the order_number column). Doing so left a prior/training set of 3,214,874 orders.

Upon segmenting our data into two datasets, we built an undirected, weighted network where each node represents a product and edges depict instances where two items existed in the same "basket". In order to do so, we grouped all items by order_id, and iterated through each order, creating all combinations of products in the same order. If an edge already existed between two items because of a previous order, the weight of the edge was increased by one, otherwise, a new edge was added to the graph with a weight of one. This process can be visualized on a small scale in figure 1. Additionally, each node has its corresponding aisle and department id numbers as attributes.

The resulting network has 49,674 nodes and 40,749,010 edges. The average weight of an edge in this network is 6 and the average degree of a node is 1,641; however both node degree

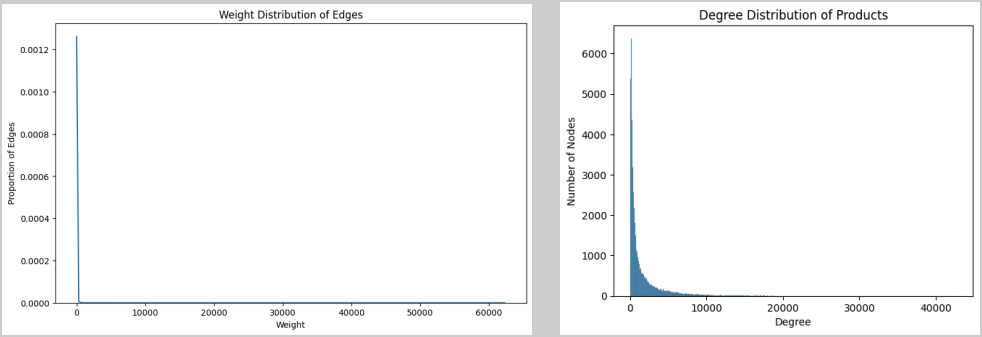
Figure 1
Shows the evolution of the product network on a small scale.



a) initialize basket 1 b) add basket 2 c) add basket 3

and edge weight are heavily right skewed, implying that there are a few products that are commonly purchased with a diverse range of other products and a few pairs of items that are frequently purchased together as seen in figure 2.

Figure 2
Shows the distribution of weight and degrees.



a) Weights b) Degrees

Due to the nature of our model, the resulting network has many edges with quite low weights (i.e. products are purchased together quite infrequently). In fact, 53.85% of the edges in our graph have a weight of one. Edges with extremely low weights may inaccurately suggest a relationship between two products and add unnecessary noise to our network. In order to remove some of the insignificant edges formed by coincidental purchases and improve the quality of our analysis, we decided to establish a minimum threshold r, such that an edge exists between two products only if they have been bought together at least r times. Determining this parameter is a careful balance between filtering out incidental edges and preserving meaningful relationships between products and is further described below.

4. Methods

4.1 Thresholding

Given our large dataset, we decided to perform statistical hypothesis testing to determine whether an edge with a given weight is likely due to random chance or represents a real

association. Since Poisson distribution is often used to model the number of events occurring within a fixed interval of time or space (in this case is the number of co-purchases between two products) we used this model to assess the significance of the observed number of times two products were purchased together. Here the poisson rate (lambda) represents the average number of co-purchases between all pairs of products in the network ($\lambda = 6$). To test the null hypothesis that the observed co-purchase frequency between two products is the result of random chance, we calculated the Poisson probability mass function for the observed value of r co-purchases, given lambda and then computed the p-value to find the cumulative probability for all values less than r before subtracting this value from 1 to get the upper-tail probability. We compared the resulting p-value to our significance level of alpha = 0.05 to decide whether we reject the null hypothesis that r number of co-purchases between two products is the result of random chance and repeated this process for every integer value of r from 5 to 20. The resulting threshold determined by this significant test is r = 11.

Although the Poisson distribution is commonly used to model the number of events occurring within a fixed interval of time or space, this approach is flawed in this application. The Poisson distribution assumes that events occur independently and the rate (lambda) is assumed constant across all product pairs. Given the nature of our dataset, we cannot safely assume that either of these are true. The co-purchase of one pair of products could influence the co-purchase of another pair, especially since each customer has multiple orders in this network. Additionally, as we examine our highly right skewed distribution of edge weights, we cannot assume all pairs of products are equally likely to be co-purchased. In our initial analysis of the edge and degree weight distributions shown in figures 2a and 2b, we observed a power-law like distribution where a small number of product pairs had extremely high co-purchasing levels. We will further explore this power law distribution by graphing our data against a power-law fit to determine if this distribution model could be a more accurate representation of our data.

Figure 3
Compares the distribution of the network to a standard powerlaw distribution.

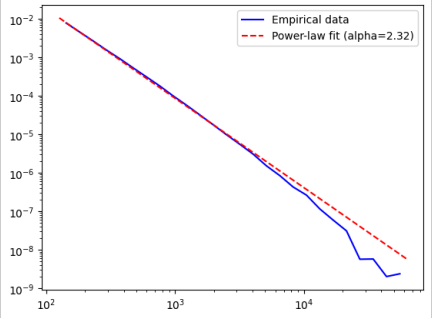


Figure 3 seems to confirm what we suspected about this network: the weights exhibit a power law distribution; thus, we continued to calculate the p-value for each observed co-purchase count k_{obs} in this network by comparing the observed count against the expected distribution of counts under the power-law model. First, we used a power-law fitting method to estimate the scaling parameter α and the minimum threshold k_{min} from our data and the power law library. Next, we calculate the cumulative distribution function (CDF) to determine the probability of observing a value less than or equal to k_{obs} using the equation $CDF = P(k_{obs}) = 1 - (k_{min}/k_{obs})^{\alpha-1}$. Then we calculated the p-value for each k_{obs} which is the probability of observing k_{obs} or greater under the fitted law distribution $p = 1 - P(k_{obs}) = (k_{min}/k_{obs})^{\alpha-1}$. Finally, we compared the corresponding p-value of every unique edge weight to our alpha level of 0.05 to decide whether the edge weight is significant or not and accumulated all significant edge weights in the list. The smallest "significant weight" serves as our threshold; therefore, this approach resulted in an edge weight threshold of r = 15 which limits the dataset to the top 5% of edge weights.

4.2 Feature Exploration

Apart from product name, each node has an associated aisle and department ID number. Incorporating these attributes into analysis can provide valuable context, uncover hidden patterns, and make the large network more interpretable. Additionally, since products within the same department or aisle are likely to be more similar in terms of usage, customer preference or function, these features could help to make our recommendation model make more context-aware since aisle ID might indicate that products within the same aisle are often bought together, regardless of co-purchasing frequency, while department number generally represents broader categories (e.g., dairy, produce, deli) which could allow the model to prioritize recommendations based on product groupings.

Within our pruned network, we can explore these features to determine what products are most frequently co-purchased together in each aisle and department respectively. For each aisle ID, we created a list to contain co-purchasing frequency for products within said aisle. After iterating through all connected nodes in the aisle, we sorted our list of products and co-purchasing frequencies in descending order and added the pair of products purchased together most frequently to a new dataframe that held the most commonly co-purchased items in every aisle. We followed the same approach for department numbers.

4.3 Community Detection

By using community detection to identify clusters of frequently co-purchased products, we can identify more diverse groups and uncover new product categories outside of the given aisle and department attributes. Since items within the same community tend to be complementary or related in some way, community detection can offer valuable product bundling or cross-promotion opportunities from a marketing perspective. Additionally, community detection algorithms may discern more complex product connections by revealing hidden relationships between products that aren't immediately obvious from individual co-purchasing counts.

We took two approaches to community detection: greedy modularity and Louvain-based community detection. The greedy algorithm was implemented using the community module within Networkx algorithms. Behind the scenes, this function makes decisions to locally optimize the modularity score (defined as the density of edges within communities compared to a random distribution of edges) at each step by iteratively merging the most similar communities until the modularity score can no longer be improved. Although this process often leads to well-defined communities, it can be computationally expensive for large networks (which we have) and performs best in cases where communities are easily separable (which we may not have). Alternatively, the Louvain method was implemented using the standard community module. Louvain is also based on modularity optimization; however, it takes a mulit-level hierarchical approach by first optimizing modularity locally before proceeding to larger communities. This algorithmic structure makes the Louvain method more efficient and therefore more suitable for large, sparse networks.

Similar to our attribute analysis, detecting communities in our product network can enhance our recommendation system by suggesting products from the same community as the ones already in a customer's basket. We can allow our model to account for products' co-purchasing tendencies based on their community membership by using each product’s assigned community as a feature

in collaborative filtering models. Since products in the same community are often purchased together, this added layer can lead to more relevant recommendations.

4.4 Product Recommendation

Our goal is to provide informed product recommendations based on all items in an order. Since we cannot evaluate how our recommendation performs without user input, we will slightly shift our perspective to predicting a missing item in an order. We will use the same approach as we would for a recommendation model; however, this adjusted problem provides a ground truth we can use to evaluate our model's performance.

Given our network where nodes signify products, weighted edges represent co-purchasing frequency and nodes have features such as aisle ID, department number, and assigned community, we believe collaborative filtering is the appropriate choice for product recommendation. Akin to user-based collaborative filtering, we can apply a similar logic to recommend products based on item similarities (i.e. products that are frequently purchased together) which aligns well with our co-purchasing, product network. Furthermore, collaborative filtering can uncover underlying relationships between products that are not explicitly expressed through previously discussed product features like aisle, department or community. For example, products with shared attributes might be purchased together, but so might products with similar usage patterns or complementary functions; collaborative filtering can capture these types of patterns without requiring domain knowledge of each product. Another benefit of collaborative filtering includes its ability to handle sparsity. Since our pruned product network is quite large (24,001 nodes) and incredibly sparse (0.007 density), using matrix factorization techniques such as SVD to compress the co-purchase matrix into a lower-dimensional representation, allows our model to still make recommendations for products with few co-purchases or no direct relationships. Finally, combining collaborative filtering with attribute similarities presents opportunities for enhancements by incorporating additional information such as the product features mentioned earlier.

Our procedure involved first creating a co-purchasing (or adjacency) matrix from our pruned product graph where the value of each row-column pair was the weight of the edge between the two items. Over 99% of our co-purchasing matrix was sparse, thus we created a reduced matrix by applying SVD, a matrix factorization technique used to handle sparse data, with 50 components and a set random state for consistency. In order to leverage node attributes and community assignments previously discussed, we stored each node's aisle ID, department number and community assignment for both greedy and louvain modularity approaches into a dataframe. Next, we applied cosine similarity from the scikit learn module to our reduced matrix and converted it into a dataframe for easier manipulation. This cosine similarity data frame was then boosted using product attributes; if two products shared an aisle, department or community, their similarity score was increased by a set bonus parameter of 2.5. Finally we implemented our collaborative filtering recommendation model which takes a "basket" of items and cosine similarity matrix as inputs and returns a list of top recommended products. For each product in the provided basket, our model loops through all other products in the cosine similarity matrix. If the other product is not already in the basket, the function increments the cosine similarity (taken from the cosine similarity matrix) between the current product and this other product by its corresponding score. The similarity score of other products accumulates as the function iterates

over all products in the basket. Once all products in the basket have been considered, the aggregated similarity scores are sorted in descending order and the items corresponding to the top 15 scores are recommended.

4.5 Association Rules Mining

Apriori is an algorithm used to identify frequent item sets. It does so by a 'bottom up' approach, where it first scans every item in the database, then enumerates the possible subsets, check whether their frequency is above the minimum support, followed by finding frequent itemsets, then calculating the confidence of association. Apriori pruning principle states that if any itemset is infrequent, none of its superset need to be considered (Agrawal & Srikant 1994). Once the item sets have been generated using apriori, we can start mining association rules. Given that we are only looking at item sets of size 2, the association rules we will generate will be of the form {A} -> {B}. One common application of these rules is in the domain of recommender systems, where customers who purchased item A are recommended item B.

Our procedure first started with exploring the dataset, which comprised customer orders, product details, and transactions. Due to memory constraint we only selected order id that's less than 10000. After data preprocessing, we created a transactional matrix to represent customer orders as baskets of items. This matrix served as the input for the Apriori algorithm. Then we can start the association rule mining.

The three key metrics to consider when evaluation association rules are support, confidence, and lift. Support is the percentage of orders that contains the item set. Since there could be thousands of distinct items in InstaCart dataset and an order can contain only a small fraction of these items, setting the support threshold to 0.01% is reasonable. Given two items, A and B, confidence measures the percentage of times that item B is purchased, given that item A was purchased. This is expressed as: confidence{A->B} = support{A,B} / support{A}. Confidence values range from 0 to 1, where 0 indicates that B is never purchased when A is purchased, and 1 indicates that B is always purchased whenever A is purchased. Note that the confidence measure is directional. Lift indicates whether there is a relationship between A and B, or whether the two items are occurring together in the same orders simply by chance (ie: at random). Unlike the confidence metric whose value may vary depending on direction (eg: confidence{A->B} may be different from confidence{B->A}), lift has no direction. This means that the lift{A,B} is always equal to the lift{B,A}.

5. Results

There are a total of 124 aisles and 20 departments in our pruned dataset; thus, categories are more specific when grouping by aisle number which can result in more similar, less informative product pairings while 20 departments may run the risk of overgeneralizing, leading to potentially less meaningful results as well. For some aisles like "butter" or "refrigerated", product pairings are seemingly variations of the same thing (Pure Irish Butter and Unsalted Pure Irish Butter and Trilogy Kombucha Drink and Organic Raw Kombucha Gingerade), but further insights can be gained from other, more diverse aisles such as "spreads", "fresh vegetables" and "condiments". In these aisles, we see more diverse, yet classic pairings such as Organic Strawberry Preserves and Organic Creamy Peanut Butter or Organic Ketchup and Organic Yellow Mustard. Similarly, departments like "snacks" and "canned goods" only provide

information on one type of snack (fruit snacks) and one type of canned good (beans) while other departments such as “produce” and “pantry” present more unexpected pairings like Organic Hass Avocados and Bag of Organic Bananas or Crescent Rolls and Cinnamon Rolls with Icing.

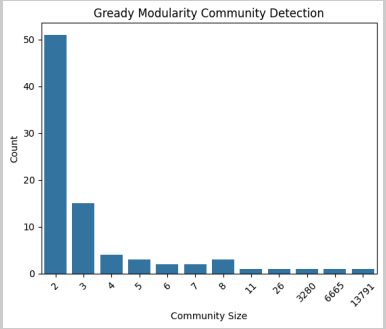
Apart from common product pairings, we can also extract customer behaviors from our results. By examining the magnitude of product pairing purchases for each aisle/department, we can infer where customers may have certain shopping patterns or are more likely to make co-purchases. For example, the greatest number of co-purchases among the “fresh fruit” aisle is 62,341 whereas the maximum number of co-purchases in the “egg” aisle is 610. Since these two aisles contain common household products, it seems reasonable to assume that customers are more likely to make co-purchases in the produce section when compared to the eggs section of the store.

Overall, applying node attributes in this way was insightful and illuminated frequently co-purchased product pairings in a structured, interpretable way; however, this method only considers two items and neglects relationships that may span departments or aisles such as flour, sugar, butter and eggs, chips and salsa, or bagels and cream cheese. For this analysis, we turn to community detection.

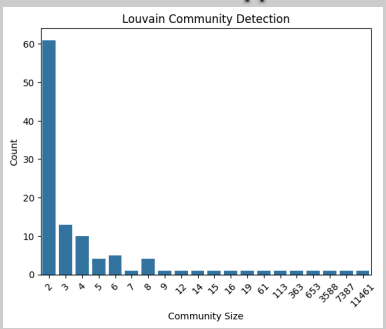
Both methods of community detection yielded somewhat similar results. The Greedy modularity approach produced 85 communities, with the largest one containing 13,791 (or 57%) of the products in our network while the Louvain method identified 111 communities and the largest one contained 11,461 (or 47%) of products. Additionally, the two approaches had similar distributions of community sizes as displayed in figure 4 where both community detection methods display vague power law properties with the majority of communities including two products.

Figure 4

Shows the distribution of community sizes for two different approaches.



a) Greedy Modularity Detection



b) Louvain Community Detection

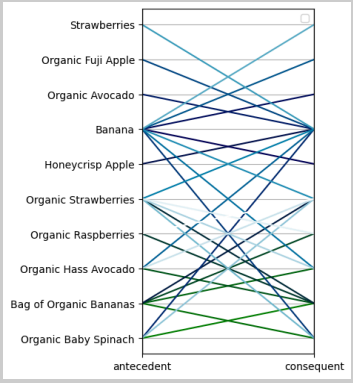
As we saw with our aisle and department analysis, communities of size two are not extremely insightful; however, extremely large communities with thousands of products are uninterpretable and not very informative either. Instead of examining the communities at either extremes, we will focus on interpreting those communities that lie in the middle in terms of size. When looking at communities of size 11 and 26 detected by greedy modularity, we noticed significant product trends. In the community of size 26, all nodes seemed to be baby related while most products in the community of size 11 seemed to be related to dog food but not all. In fact, an interesting phenomenon occurred in the second community where two of the products were completely unrelated to pets: mellow mango sweet dreams and sonic blood orange passion flavored dietary supplement. When looking at communities of size 19 and 15 detected by the Louvain method, we noticed significant product trends again. The community of size 15 was

similar to the community of size 11 found by the greedy algorithm in that it was a homogenous group of products (in this case, ice creams); however, the community of size 19 uncovered some new insights. This community included products related to cat food; however, similar to the community of dog foods, it too included some unexpected products like white cheddar cheese, squeezed lemonade fruit juice and core hydration water.

In order to evaluate the results of our recommendation system, we took our reserved testing set and randomly removed one product from each order. Given the extremely large size of our network, we randomly selected 1,000 of these basket-to-missing item pairings to test our recommendation method. For each basket, our model recommended 15 other products; every randomly removed product included in this list of 15 recommended items contributed to the model's precision score. Despite experimenting with hyperparameter tuning, implementing matrix factorization via singular value decomposition and incorporating product attributes, our precision score was between 0.036 and 0.041 which is quite low. This result suggests that achieving a high precision is quite difficult for a product recommendation model due to the inherent nature of the product data and unpredictable buyer behaviors.

The generated association rules reveal interesting relationships between products in the dataset. Bag of Organic Bananas are highly associated with organic Hass avocado, with 15.99% of the transactions that include "Bag of Organic Bananas" also include "Organic Hass Avocado." Lift of 2.468 indicating that this rule is over twice as likely compared to random co-occurrence. Similarly, Banana is also highly associated with Organic Strawberries, 1.86% of transactions include both items.

Figure 5
Parallel coordinates plot shows relationships between antecedents and consequents



The association rules were visualized using a parallel coordinates plot, where the relationships between antecedents, consequent, and the rules they formed were clearly mapped. One of the most interesting insights was the identification of specific product pairs frequently purchased together with high confidence and lift, such as {Banana} → {Strawberries}, indicating that customers buying product A were significantly more likely to buy product B.

The Apriori algorithm proved to be an effective method for uncovering patterns in transaction data. By identifying frequent itemsets and deriving association rules, we gained valuable insights into customer purchasing behaviors. These findings can be leveraged to enhance business strategies, from improving product placement in stores to optimizing online shopping experiences through recommender systems.

6. Challenges

Establishing and assessing a pruning threshold to remove unwanted noise is difficult and affects the rest of our analysis. Although we tried to make an informed, statistically-backed decision, we recognize that different thresholds could lead to drastically different results. A threshold that is

too aggressive could lead to loss of critical information and unfairly favor more popular products while a threshold that is too timid increases computational complexity and the risk of overfitting. By disregarding all edges with a weight lower than 15, we removed approximately 95% of our edges; however, in future analysis, we would like to experiment with how maintaining 1% of our edges (i.e. removing all edges below weight 66) may affect our community detection and recommendation system. Perhaps a more aggressive threshold would lead to smaller, more informative communities.

Achieving a high precision score for a product recommendation model is difficult in this context due to many factors. Perhaps most apparent is the diverse catalog of product offerings in our network as there exist many variations of the same product. For example, the missing product may have been “Carrots” but only “Organic Carrots” were included in the list of recommended items. Although these two items are virtually the same, our evaluation metric does not consider this a correct recommendation. Since precision is a relatively strict measure, the score can be low if the top recommendations are not exactly perfect, even when the system does an adequate job at making recommendations otherwise. Another challenge includes unpredictable buyer behaviors. The user may be buying for a household with different dietary restrictions and taste preferences, making it challenging for our model to accurately target the most relevant products in the top 15 recommendations, even with the implementation of attribute-based filtering. In future analysis, we would consider experimenting with GCN for high order relationships using node features and because of its ability to utilize the underlying graph structure to improve recommendations. We could then try implementing ensemble models to blend GNN with SVD and traditional collaborative filtering approaches to hopefully robustify our model. Given the dataset provided by Instacart, we could also incorporate contextual features, such as the hour of the day, day of the week and days since last order, to improve our model’s precision. Furthermore, we could try implementing deep learning models such as neural collaborative filtering to better learn complex patterns in data.

Due to limitations in the machine capacity, the apriori algorithm only considered product id that is less than 10000, which included mainly the fresh produce. The order ID goes up to 49,688, which included other product categories such as home, alcohol, pets etc. that Instacart users bought, if we are able to run the apriori algorithm for all products the results might yield more item set pairs with higher lift that will be valuable in predicting the next basket.

7. Conclusions

In this study, we explored the Instacart dataset to understand product purchasing patterns and build a recommendation model tailored to customer preferences. Through a combination of network analysis, feature exploration, community detection, and recommendation algorithms, we uncovered valuable insights into customer behaviors, product relationships, and co-purchasing trends.

We began by segmenting the dataset into prior/training and testing sets, ensuring a robust structure for model evaluation. Using co-purchasing data, we constructed a weighted product network, applying thresholding techniques to remove noisy edges and preserve meaningful relationships. Statistical approaches such as the Poisson and power-law models were utilized to refine edge weights, ensuring that our network analysis focused on significant product associations.

Feature exploration revealed that product attributes like aisle and department IDs could enhance the interpretability and effectiveness of the recommendation model. Community detection provided additional insights, uncovering clusters of related products that went beyond predefined categories, such as aisles and departments, and illuminated novel product relationships.

Our collaborative filtering model demonstrated the value of leveraging co-purchasing trends to predict missing items in orders. By incorporating matrix factorization and boosting similarity scores with product attributes, we achieved a balance between interpretability and computational efficiency. However, challenges such as data sparsity and variations in product nomenclature posed limitations to precision.

Additionally, association rule mining with the Apriori algorithm identified frequent itemsets and strong product associations, offering actionable insights for both in-store and online product recommendations. Despite memory constraints limiting the scope of this analysis, the results highlighted important co-purchasing patterns that could inform strategies like product bundling and targeted marketing.

While our methods achieved meaningful insights and laid a strong foundation for recommendation systems, some challenges persist. The sparsity of the dataset, the existence of similar yet distinct product variations, and computational constraints limited the full potential of our analysis. Future work could address these challenges by implementing more scalable algorithms, leveraging additional data sources, and incorporating user feedback into model evaluation.

Overall, this analysis showcases the potential of data-driven methods to enhance customer experience through personalized recommendations. By combining network analysis, feature exploration, community detection, and collaborative filtering, we have created a framework capable of uncovering complex relationships in transactional data and improving the relevance of product recommendations. These findings not only contribute to better understanding consumer behavior but also hold practical implications for optimizing e-commerce and retail strategies.

Citations

Agrawal, R., Srikant, R. (1994). “Fast algorithms for mining association rules in very large databases”. In: Proceedings of the 20th International Conference on VLDB. Santiago, Chile, pp 487–499

Albert, T. C., Goes, P. B., & Gupta, A. (2004). GIST: A Model for Design and Management of Content and Interactivity of Customer-Centric Web Sites. MIS Quarterly, 28(2), 161–182. https://doi.org/10.2307/25148632

Aldino, A. A., E. D. Pratiwi, E. D., Setiawansyah, Sintaro, S., and Dwi Putra, A. "Comparison Of Market Basket Analysis To Determine Consumer Purchasing Patterns Using Fp-Growth And Apriori Algorithm," 2021 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE), Banyuwangi, Indonesia, 2021, pp. 29-34, doi: 10.1109/ICOMITEE53461.2021.9650317.

Chen, Y.-L., Chen, J.-M., & Tung, C.-W. (2006). A data mining approach for retail knowledge discovery with consideration of the effect of shelf-space adjacency on sales. Decision Support Systems, 42(3), 1503–1520. https://doi.org/10.1016/j.dss.2005.12.004

Cumby, C., Fano, A., Ghani, R., & Krema, M. (2004). “Predicting customer shopping lists from point-of-sale purchase data. In KDD-2004 - Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 402-409)”. (KDD-2004 - Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining). Association for Computing Machinery (ACM). https://doi.org/10.1145/1014052.1014098

Guidotti, R., Rossetti, G., Pappalardo, L., Giannotti, F., and Pedreschi, D. "Personalized Market Basket Prediction with Temporal Annotated Recurring Sequences," in IEEE Transactions on Knowledge and Data Engineering, vol. 31, no. 11, pp. 2151-2163, 1 Nov. 2019, doi: 10.1109/TKDE.2018.2872587.

Guidotti, R., Monreale, A., Nanni, M., Giannotti, F., Pedreschi, D. (2017). Clustering Individual Transactional Data for Masses of Users. New York, NY, USA: ACM.

Hu, Y., Koren, Y., and Volinsky, C. "Collaborative Filtering for Implicit Feedback Datasets," 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 263-272, doi: 10.1109/ICDM.2008.22.

Instacart. “Instacart Market Basket Analysis.” Kaggle. https://www.kaggle.com/c/instacart-market-basket-analysis

Kim, E., Kim, W., & Lee, Y. (2003). Combination of multiple classifiers for the customer’s purchase behavior prediction. Decision Support Systems, 34(2), 167–175. https://doi.org/10.1016/S0167-9236(02)00079-9

Kim, Y., & Street, W. N. (2004). An intelligent system for customer targeting: A data mining approach. Decision Support Systems, 37(2), 215–228. https://doi.org/10.1016/S0167-9236(03)00008-3

Kumar, N., Gangopadhyay, A., & Karabatis, G. (2007). Supporting mobile decision making with association rules and multi-layered caching. Decision Support Systems, 43(1), 16–30. https://doi.org/10.1016/j.dss.2005.05.004

Kwan, I. S. Y., Fong, J., & Wong, H. K. (2005). An e-customer behavior model with online analytical mining for internet marketing planning. Decision Support Systems, 41(1), 189–204. https://doi.org/10.1016/j.dss.2004.11.012

Lin, Q.-Y., Chen, Y.-L., Chen, J.-S., & Chen, Y.-C. (2003). Mining inter-organizational retailing knowledge for an alliance formed by competitive firms. Information & Management, 40(5), 431–442. https://doi.org/10.1016/S0378-7206(02)00062-9

Linden, G., Smith, B., and York, J., "Amazon.com recommendations: item-to-item collaborative filtering," in IEEE Internet Computing, vol. 7, no. 1, pp. 76-80, Jan.-Feb. 2003, doi: 10.1109/MIC.2003.1167344.

Liu, D.-R., & Shih, Y.-Y. (2005). Integrating AHP and data mining for product recommendation based on customer lifetime value. Information & Management, 42(3), 387–400. https://doi.org/10.1016/j.im.2004.01.008

Mittal, A. “The Bread Basket.” https://www.kaggle.com/datasets/mittalvasu95/the-bread-basket/data

“Market-Basket-Analysis.” https://github.com/satishrath185/Market-Basket-Analysis

Pradhan, S., Priya, P., & Patel, G. (2022). Product bundling for ‘Efficient’ vs ‘Non-Efficient’ customers: Market Basket Analysis employing Genetic Algorithm. The International Review of Retail, Distribution and Consumer Research, 32(3), 293–310. https://doi.org/10.1080/09593969.2022.2047756

Raeder, T., & Chawla, N. V. (2010, August 28). Market basket analysis with networks.

Stanford University SNAP. “Amazon product co-purchasing network and ground-truth communities.” https://snap.stanford.edu/data/com-Amazon.html

Tam, K. Y., & Ho, S. Y. (2005). Web Personalization as a Persuasion Strategy: An Elaboration Likelihood Model Perspective. Information Systems Research, 16(3), 271–291. https://doi.org/10.1287/isre.1050.0058

University of Notre Dame. https://www3.nd.edu/~dial/publications/raeder2011market.pdf