CSCI 5220 H01 Social Networks & Informatics

Final Project Report
10th December 2022
Odyssey Villagomez
Samuel Wozinski

Amazon Recommendation System

Problem Statement and Background

For brick-and-mortar stores, the products that companies sell are constrained by physical space and budgets. Companies need to know which products are most likely to sell so that they can allocate space and money to those items. Knowing which products are purchased together allows companies to know what products they should carry. It also allows companies to recommend specific products to users or create bundles of products that are likely to sell together. The main motivation here is to recommend products that are useful to the shopper, based on a selected item, and to sell more products by making recommendations.

We would like to create a product recommendation system based on groupings of products from the "Customers Who Bought This Item Also Bought feature" Amazon dataset (Leskovec, Adamic, & Adamic, 2003). The Amazon dataset is a network of products that are connected to other products with a directed edge if they are co-purchased.

Methods

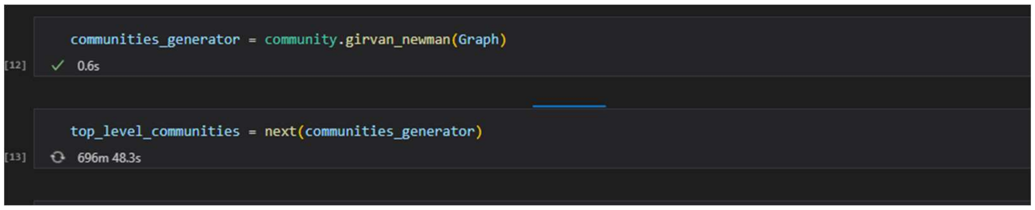
Given an Amazon network based on "Customers Who Bought This Item Also Bought feature" we propose to extract groups of products from the network. The groups will be composed of products that are frequently bought together. These groups will then be used to recommend products. For example, if a product is bought and is a member of a group, the other products in that group will also be recommended. We will use the NetworkX python package to load and analyze the dataset (Sphinx 0.5, 2008). From there we will look at different metrics like edge betweenness, node betweenness, node importance, and node degree to identify clusters of products

Implementation

Before we started extracting groups from the dataset, we first used the NeworkX package to compute some general network characteristics about the graph. This included calculating the average clustering coefficient, which measures how much the nodes in the graph cluster together. The graph had an average clustering coefficient of 0.35, We also plotted a histogram to visualize the shape of the dataset. The histogram revealed that most items have 0 - 15 neighbors, with 9-10 being the most common value. Additionally, the degree of the graph has a standard deviation of 5.92 and a mean of 9.42.This means that 68% of items are bought with 3.5 to 15.34 items.

Extracting Groups

To avoid the cold start problem where there are 0 initial products to recommend to the store owner, we propose to extract the most popular groups of products from the data. To do this we tried a variety of methods, including the Girvan Newman algorithm, PageRank analysis, and computing the degree of all nodes. The Girvan Newman algorithm uses edge betweenness to create communities by iteratively removing the edge with the highest edge-betweenness until there are no edges remaining. Due to the size of the dataset, 262,111 nodes with 1,234,877 edges, the Girvan-Newman algorithm was unable to run in a reasonable amount of time, see the figure below. We left the algorithm running overnight, and it still did not successfully complete. This makes sense given the time complexity of the Girvan-Newman algorithm.



```
[12] communities_generator = community.girvan_newman(Graph)
     ✓ 0.6s

[13] top_level_communities = next(communities_generator)
     696m 48.3s
```

Second, we analyzed the network using the PageRank algorithm. In this dataset, all incoming links to a node i represent all other products bought with item i. The more important a node i is, the more often it is bought with other products. This helps us identify what products are commonly bought with other items in the dataset. Using the PageRank algorithm, we found the nodes with the most importance, however due to the large number of nodes in the dataset, the PageRank scores were very low for each node. Lastly, we looked at the nodes with the highest degree of neighbors. The degree of each node tells us how many other products that item was bought with. The product with the highest degree connected to 425 other products.

Recommending Products

After we identified some of the popular products, we then created our recommendation system. We created a Content-based recommendation system that uses a history of previously bought items to recommend additional products. This involves keeping track of individual purchases, which allows the shop owner to later update the main graph with the number of items purchased. The code for this individual tracking and recommendation system can be seen in the attached jupyter notebook under the heading, “Create User Profile.”

Additionally, we created an algorithm to get around the long runtime of traditional clustering algorithms. In this algorithm, the store owner would enter in the ID for their most popular product, and a cluster of nodes would be built around it. The way this works is using the degree of neighboring nodes, and a greedy approach. A new graph is created, starting with the item that the user specifies. After that, the surrounding items are added. If the desired number of items is still not reached, it begins to add on nodes based on the highest degree of each node. Code for this can be seen in the attached jupyter notebook under the heading, "Store Owner Recommendation System."

Results

Amazon Network Characteristics	
Nodes	262,111
Edges	1,234,877
Avg Clustering Value	0.35
Max Clustering Value	1.0

Table 1: Network Characteristics of Amazon dataset

Top 10 Degree Scores	
Node	**Degree**
14949	425
4429	409
33	366
10519	339
12771	335
8	298
297	280
481	280
5737	277
9106	232

Table 2: Top 10 nodes and their degree

Graphs

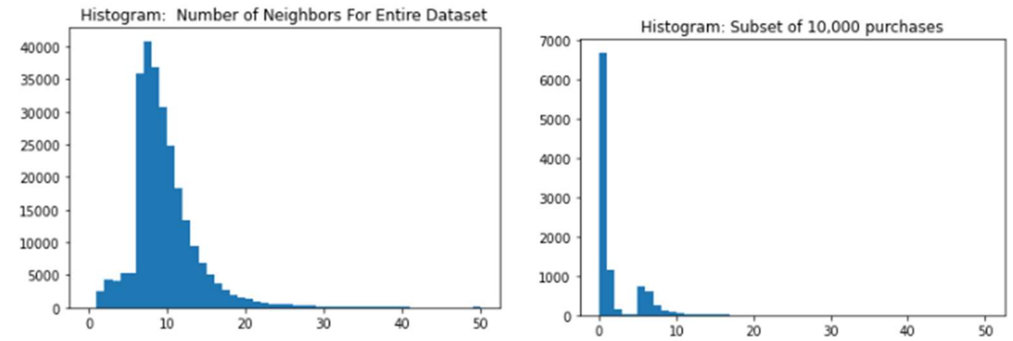


Figure 2: Histogram of neighbors for entire data set and subset of 10,000 purchases

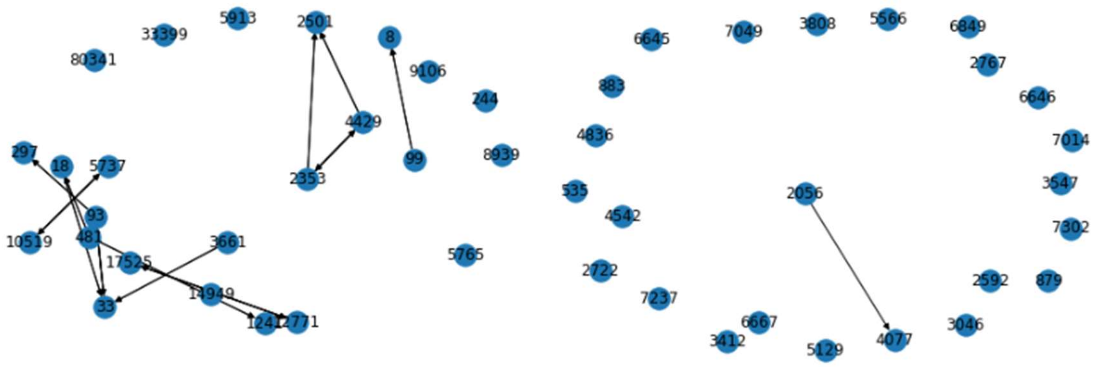


Figure 2: Subgraphs showing the nodes with the most neighbors and the PageRank most important nodes

Conclusion

Our exploration of the sample Amazon dataset gave several insights into analyzing graph data using the NetworkX package. We also learned that the runtimes of traditional clustering algorithms are often too high to run real-time on larger data samples using this package. Because of this we implemented a simple, greedy approach based on the degrees of neighboring nodes to maximize the number of related products. In the future, clustering of the dataset can be optimized by using advanced computational hardware and packages. These include GPU machines and Pytorch or Tensorflow libraries. Additionally, acquiring a dataset that includes more item features would allow a more item-specific recommendation system to be considered.

Overall, we preliminarily explored the Amazon dataset to find the most frequently bought products in the dataset and created a recommendation system to recommend these products to users.

References

Leskovec, J., Adamic, L., & Adamic, B. (2003, March 02). *Amazon product co-purchasing network*. Retrieved from snap.stanford.edu: http://snap.stanford.edu/data/amazon0302.html

Sphinx 0.5. (2008, Nov 18). *NetworkX*. Retrieved from networkx.org: https://networkx.org/documentation/networkx-0.99/