# Reproduce the group information of products using Amazon product co-purchasing network data

***CSE 881***

***mini project 2***

Mengling Hettinger

## Abstract

In this paper, I present a reproduction of item groups using Amazon product co-purchasing network data. This is a problem of graph clustering. We first assume that according to the co-purchase links, we test the result, it shows that 58% of the two vertices between a link belong to the same group. Based on this assumption, we use the link information to reproduce the group information for each product in Amazon0601 file. In the first approach, we only use the link information, by using k mean clustering, we get the accuracy is 73% and the total sum square is 360553. In the second approach, other than the link information, we also apply the information of co-customer comments, i.e., if there are comments from the same customer between the two vertices, we update the adjacency matrix by adding 1. The results is : accuracy 47% and total sum square 359286.

## Introduction

The data was collected by crawling Amazon website. Here we used two files:

1. Amazon product co-purchasing network metadata: this is a review information about 548,552 different products, each of which includes the information of Title, Salesrank, List of similar products, Groups,  Detailed product categorization and Product reviews.
2. Amazon product co-purchansing network, June 1 2003: it is based on customers who bought this item also bought feature of the Amazon website. If a product I is frequently co-purchansed with product j, the graph contains a directed edge from I to j. There are 403394 nodes and 3387388 edges in the dataset.

My goal is to reproduce the group of each product using graph clustering analysis using both files (not the ground truth) and compare the results with ground truth.  Graphs are the structures that formed by vertices (in this case, products) and edges (in this case, the relation between two product, whether they have the co-purchase relation or not). Graph clustering is the task to group vertices together based on the knowledge of edges.

Because the datasets we deal with here is massive, any algorithms that have a expensive computational complexity in either time domain or memory domain will not be applicable. The methodology we use will be presented in the next chapter. Here we only briefly explain the hypotheses and approaches we use in this paper. We first prove the hypothesis: nodes directly linked to each other are very likely to belong to the same product group. Next, based on this hypothesis, we first only use the link information itself and apply clustering algorithms to predict the group of each product in Amazon0601 file. Finally, we extract additional useful features from the meta data set and use these informations to improve our clustering results. In this case, we assume that if the two products purchased together have the common customer review, they are more likely to be in the same group. We then update the link weights, and apply the clustering method and predict the group again.

Next section, we discuss the preprocessing of the datasets we have.

## Preprocess

Based on the nature of our task, some of the nodes from Amazon0601 are removed because we do not have the ground truth of the group information about these nodes in the metadata set.  In this step, we discard 75,833 edges from the original 3,387,388 data, thus 3,311,555 are left.

In the next step, we further made a graph for all the data points. It was found that there are 28 clusters 27 of them only have less than 14 nodes. We consider these data points outliers since we are more interested in the major structure of the large graph. Another reason is being that from the meta data set we know that the products numbers are the following : Books:393561, DVDs: 19828, Music CDs: 103144 and Videos: 26132, all of which are way beyond the small clusters we found. Consequently, we also removed these outliers. This leaves us 3,311,186 nodes in total.

From the Amazon0601, I extract the groundtruth result: Book 291554, Music 75814, Video 19194, DVD 12023,  CE 4, Toy 6, Baby 1, Software 2, Sports 1, Games 0.  Finally, I apply the k mean algorithm again and finally get 10 clusters.

## Methodology

We start with the hypothesis that nodes directly linked to each other are very likely to belong to the same product group. In order to prove our hypothesis, we extracted the group information from the meta data, and directly applied to each edge to compare the two groups. We found that there are 1917763 out of 3311554 transactions are the from the same group. The ratio is 58% which directly proved our hypothesis.

We then apply clustering using only link information. In this part, we used igraph package in R (http://igraph.sourceforge.net/). First of all, we cluster all the links with the strongly connected component. In the mathematical theory of directed graphs, a graph is said to be strongly connected if every vertex is reachable from every other vertex. The strongly connected components of an arbitrary directed graph form a partition into subgraphs that are themselves strongly connected. This gives us overall 2346 clusters. Next, we binarize the group information by putting 1 to one of the 2346 columns where the vertex belongs to and setting the rest are 0.

We used two different approaches for our clustering analysis. Firstly, we used k mean approach to analyze the clusters because of the time complexity O(n). K mean clustering is a approach that k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. However, since the desktop I use only contains 8G memory, to cluster all the points at the same time becomes a problem. My solution for this problem is to use sampling strategy. I first partition all the vertices into 10 chunks each of which has 40000 points (the last one only has 398599 points). Next step, I apply k means clustering method to each of the chunk with 10 centers. After this step, I combine all the 100 clusters which contains all the 398599 vertices. Binarization is again used for this step. Secondly, we apply the CLARA approach. CLARA adopts a sampling approach, the quality of its clustering results depends greatly on the size of the sample. When the sample size is small, CLARA's efficiency in clustering large data sets comes at the cost of clustering quality. In this case we tried different size of the sample size, and report the result with random 100 samples to compromise the time consumption and accuracy.

Next, informations from the amazon-meta can be also extremely helpful when used correctly to predict groups for each product. Here we extract information from metadata to update the weight in the co-purchase data. The following hypothesis has been applied: I assume that the fact that same customer commented on different products increase the chance that these products are from the same group. To do this, we produce a dictionary where keys are the product ID and values corresponding to each key is a list of all the customers that commended on the product. From here we update the link weight by adding 1 to the adjacency matrix if there's common customer appears on the both vertices of a link in Amazon0601 dataset.

## Conclusion and Discussion

The first approach which we only apply link information to obtain the clustering information. When using the k mean algorithm, the size of 10 clusters we obtained from k means clustering are 398538, 5 , 2 , 9 , 10, 9 , 2 , 8 , 5 , 11.  The accuracy we get thus is 73%.  From the result, one can see, that only the first cluster is really large, which caused a large percentage of accuracy. However, this is not a scientific way to evaluate the clustering.  I also measure the total sum of squares in this case, it gives 360553.

Then the CLARA algorithm gives the clustering information is showed below:

```
      size   max_diss  av_diss     isolation
 [1,] 49492 1.414214 0.2770592       1
 [2,] 39734 0.000000 0.0000000       0
 [3,] 39703 0.000000 0.0000000       0
 [4,] 39602 0.000000 0.0000000       0
 [5,] 39558 0.000000 0.0000000       0
 [6,] 39364 0.000000 0.0000000       0
 [7,] 39124 0.000000 0.0000000       0
 [8,] 38945 0.000000 0.0000000       0
 [9,] 38384 0.000000 0.0000000       0
[10,] 34693 0.000000 0.0000000       0
```

For the second approach we used which we take into account the co - customer information. The 10 clustering results size are: 238786, 39931, 39994, 10, 189, 18, 39996, 2, 39626, 47. The accuracy result shows that by applying the customer information increased to 47%. The total sum of squares when taking into account the co-customer information, reduced to 359286. Forthermore, one can see, this approach gives a more spread out clustering structure which cluster 2, 3, 7 and 9 are all around the same size and above 10% of the total vertices.

The CLARA algorithm in this approach is given:

```
      size max_diss   av_diss isolation
 [1,] 42947 1.414214 0.09717429       1
 [2,] 39995 0.000000 0.00000000       0
 [3,] 39997 0.000000 0.00000000       0
 [4,] 39992 0.000000 0.00000000       0
 [5,] 39994 0.000000 0.00000000       0
 [6,] 39931 0.000000 0.00000000         0
 [7,] 39943 0.000000 0.00000000         0
 [8,] 39793 0.000000 0.00000000         0
 [9,] 39626 0.000000 0.00000000         0
[10,] 36381 0.000000 0.00000000         0
```

We believe that when using K mean algorithm, by adding the co-customer information will improve the clustering results because it reduced total sum  of squares. The result we got from the CLARA algorithm is not as promising as we expected since the data set is not well balanced among all the clusters.

In the future, the nonlinear features may use when predicting the clusters. For example, $A^T A$ can be used as a feature, where A is the adjacency matrix. Furthermore, the name of a product can be used as the feature to update the link between two vertices. For example, if the two items have the same word in common in their names, we will increase the weight between them.