Batching Online Shopping Orders By Clustering With Balance Constrains

Chengfei Wang czw0078@auburn.edu

Introduction

E-commerce booming brings challenge to traditional management of logistic. How to efficiently satisfy the requirements from customer becomes the priority of the company like Amazon. Improve efficient of process of orders by grouping of similar ones into one batch is a classic method used in company [3]. But it is impossible to manually do the task for large amount of orders (>10,000) meanwhile take all the factors into consideration, such as warehouse inventory status, locations of distribution centers and costumers, capacity and time constrains. A machine learning method, clustering solution, should be proposed to solve the problem of order batch optimization, therefore the best or the second best efficient can be achieved.

The logistics systems usually consist of warehouses, distribution centers and transportation vehicles. The stock keeping units (SKU) are the inventory in the warehouses catalog, and the catalogs of different warehouse may overlaps with each other. The distribution center collects items from warehouses according to the given set of orders and deliver them to customers in local region by transportation vehicles. For the seller, it draws great effort to lower the cost of operation in the sorting process, therefore the grouping of orders is the first method we can think of. The cluster of orders must minimize the sorting cost at warehouses and travel cost of vehicles while satisfying a certain constrains, both time and capacity.

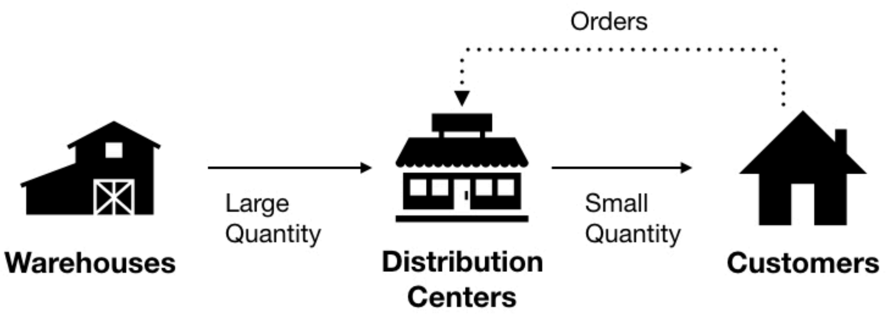


Figure 1: The major elements to consider in our model. The solid line shows how the products are delivered and the dashed line show how the orders are processed by distribution centers.

As figure 1 shows, the the products are first delivered from the large warehouse to the local distribution centers, and then to the online purchasing customers. In reality, we often observed following truths:

(1) The warehouses have larger capacity and their storage cost almost negligible compare to the other major cost.
(2) However, the warehouse locations are far away from the customers. The major cost of the logistics is the cost of transportation and storage in distribution centers.
(3) Similar products are usually stored in the same warehouse in large quantity for wholesale.
(4) The local distribution centers are close to the customer, but the distribution-centers-to-customer cost per unit distance is higher than the warehouses-to-distribution-centers cost, known as last milage problem.
(5) Similar orders are assigned to one local distribution center for process.
(6) Once the warehouses and the distribution centers are set up, the cost is a fixed investment in long-term, and the efficiency of usage determines the total cost.

Classic Optimization Problem

From above observations, we simplify our model and make following assumptions:

(a) The cost of storage and capacity constrains are not be considered in the model because of (1), that is, the warehouses are simply treated as “location source” of all products.
(b) The catalogs of different warehouses do not overlaps with each other because of (3).
(c) The cost of the distribution center is reflected implicitly as the “balance constrain”, because of (6). The model will not discuss the cost of distribution centers explicitly, but rather represented implicitly as an efficiency ration constrain M.

$$M = \frac{N_{tot}}{N_{center}} \times b$$

Number of data points in every cluster must lager than M, make sure every center has minimal work load, lowering the cost of distribution center. The meaning of b is that, if we want each distribution center can reach at least average capacity 95%, 85%, 75% and 65% to balance the load of each center, then we set b equals 0.95, 0.85, 0.75 and 0.65, therefore the value of M equals 500, 600, 700 and 800.

(d) According to (1) the estimation of the total cost of total logistic is defined as follow: for each order m and the l-th item purchased in this order, we have the location of the warehouse of this item: $i = W(m, l)$ and the location of the customer of order $k = S(m)$ as known condition, we try to minimize the estimation of the total cost under balance constrain M:

$$E_{total,M} = \sum_{m} \sum_{l} \left(D_{i,j} \times Q_{m,l} \times C_1 + D_{j,k} \times Q_{m,l} \times C_2 \right)$$

By assigned each oder to certain distribution center j to minimize the objective function value $E_{total,M}$:

$$j = argmin(E_{total,M})$$

While the $D_{i,j}$ and $D_{j,k}$ represent the distance from warehouse i to j and distance from distribution center j to location of customer k respectively, and C_1 is the cost of delivering from warehouses to distribution per unit distance per quantity and C_2 is the cost per unit distance per quantity from distribution centers to customers. $Q_{m,l}$ is the quantity of the l-th item purchased in order m.

(e) Also because of (4), we should have $C_1 <= C_2$, in this experiment, we set $C_1 = 1.0$ and $C_1 = 2.0$ from experience.

Clustering Problem

So far, the question is still a classic optimization problem. However, even after several simplification, the task of $j = argmin(E_{total})$ optimization is still unsolvable due to large computation. So we want to have a clustering method under parameter α and balancing fraction M that minimize the object function:

$$j = C_{\alpha,M}(m) \approx argmin(E_{total,M})$$

Another reason that we can not solve it as an optimization problem is that we often only have the information of orders, and we do not have the information of warehouses storage, and the cost of delivery per unit of distance per quantity. We should first group orders from its list of shopping to have good approximation.

Naturally, the questions we should ask ourself as following:

Q1: How we set up the since we do not have the other information?

Like we mentioned before that before, we do not have the information of warehouses storage, and the cost of delivery per unit of distance per quantity.

Q2: How we solve the problem of high dimensions and sparsity property of data vector?

For each order if we represent the purchasing quantity of 3645 kinds of products as a vector, we will have extremely high dimension and sparse vectors, because 3921 customers with total 3645 kinds of products in total quantity 80996 means average only 20 kinds of product at most for each customer. This means there are a lots of zero of the 3645 size vector.

Q3: How to define a meaningful distance between every two orders?

We hope the total distortion function minimization in clustering can approximate the minimization of estimation of cost best, therefore the result of clustering can be applied to the assignment of orders to different distribution centers.

Q4: How we deal with the balance constrain during the clustering?

The classic clustering problem is a un-constrained optimization problem solved by coordinates descendent, how we modified the algorithm to have a constrained version of the clustering?

Finally the question that:

Q5: How we evaluate the result and how good it is?

Methods

Data set-up

To solve the Q1, the dataset of the experiment described as following. An on-line retail dataset in UCI Machine Learning repository [1] is used in our model. Those are UK-based online retail data of 16649 invoices/orders from 01/12/2010 to 09/12/2011, however, we do not have other data like we mentioned in Q3. Therefore, we randomly assign (1) locations for

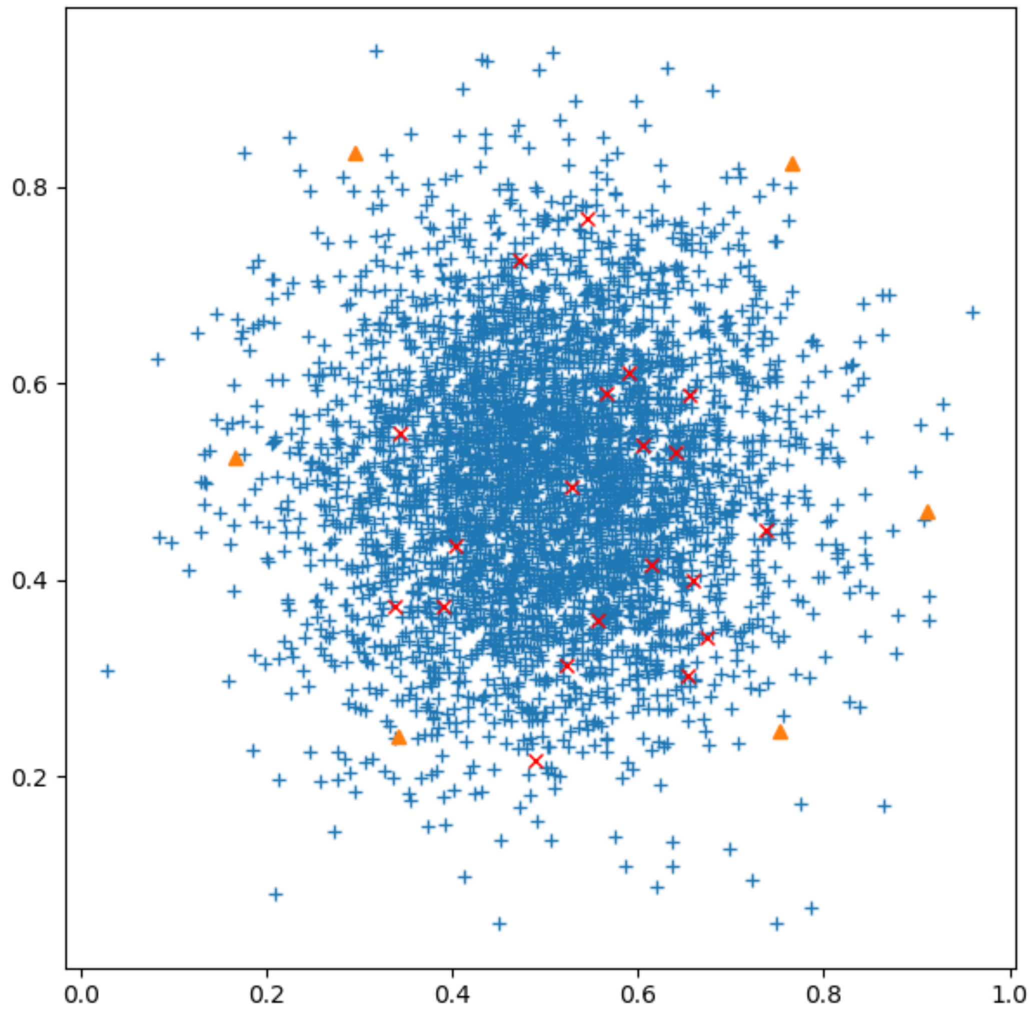


Figure 2: (1) Blue crosses: the location of 3921 customers. (2) Red Xs: the locations of the 20 distribution centers. (3) the storage of products not shown. (4) Yellow triangles: the locations of the 6 warehouses in suburb area.

3921 customers from 2D gaussian distribution with $\sigma = 0.13$ (see figure 2, blue cross), both X, Y coordinates range roughly 0~1. Then we set (2) 20 extra points follow the same distribution as the location of distribution centers (figure 2, red X). Also, in this data set, there are total 3645 kinds of different product, we (3) assign them into 6 warehouses (yellow triangles) from uniform distribution. Finally, we (4) manually set location of 6 warehouses in the suburb area of the London.

As we described above, the location of customers fits the experience that higher density of population in the downtown of city. The warehouses usually located remotely outside of downtown area with low cost. Finally, the local distribution centers should follow the distribution of population to closer to local customers.

Data cleaning and pre-process

We originally try to use k-modes to cluster because the large amount of category variables in the invoices, however, it does not work. Due to the problem we mentioned in Q2, all the data are aggregated into one cluster and the clustering results make none-sense. To solve this problem, we choose to group 3645 kinds of products into several “Topics” so the dimensions and the sparse are both reduced, and no need to use k-modes.

The topics are defined meaningfully according to which warehouses the product originally stored. In this experiment, we have 6 warehouses, therefore, the 3645 be compressed into 6 dimensions and each components are normalized as percentage from 0.0 to 1.0. As our later result shows, the clustering has a very good result after use topic method for pre-processing.

Workflow of the algorithm

We implemented a methods in Python similar to [2]. The whole flowchart shows in figure 3.

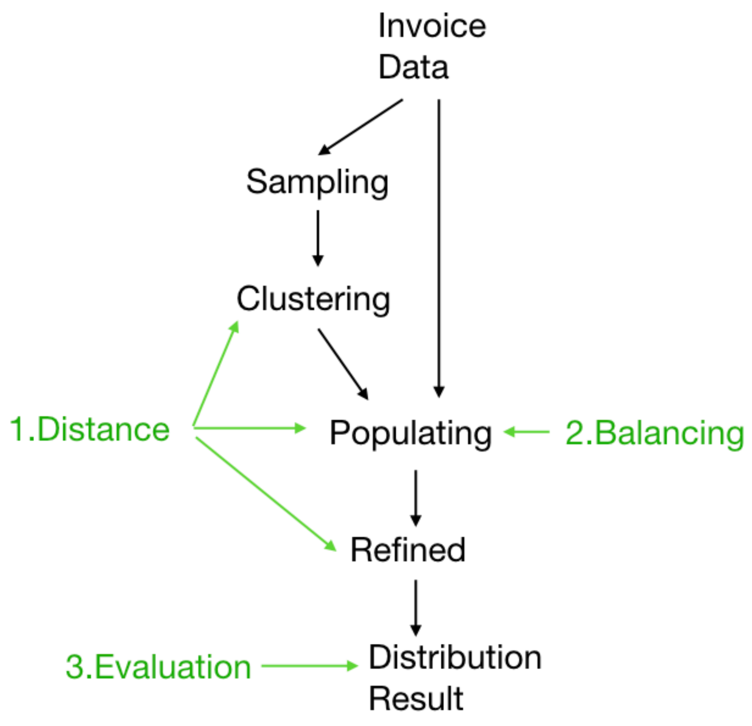


Figure 3: (1) Flowchart of each steps and (2) green arrows shows the critical step.

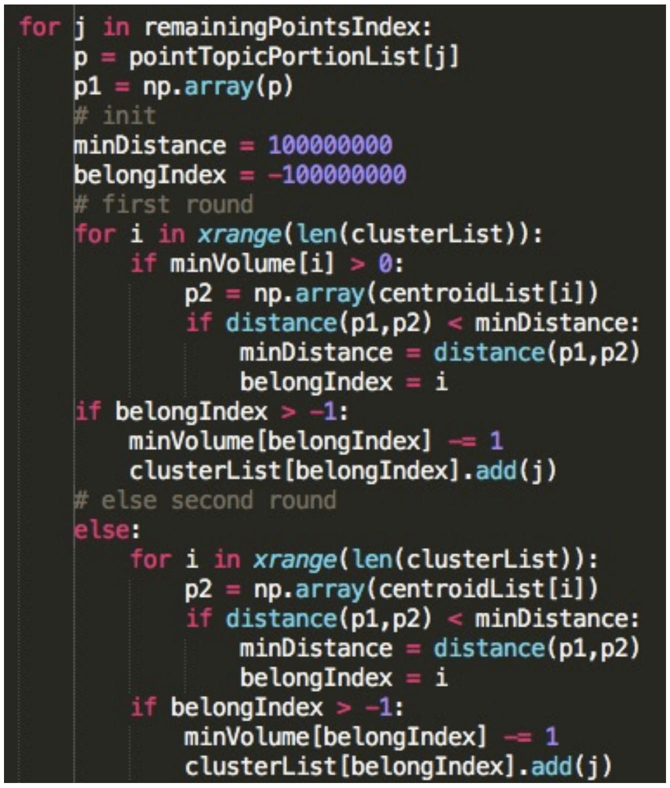
First we sampling small portion of compressed invoice data to do initial clustering. The number of clustering is exactly the same as the number of distribution centers, that is 20. Also, instead of random set up of initial value of centroids of clusters in k-means, we set the initial values according to the nearest warehouse of each distribution center and their own location. For example, if the distribution center located in (0.7, 0.5) has the nearest warehouses 1(most right yellow triangle in figure 2), then the centroids is [1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.7, 0.5]. In total, the data are now 8 dimensions vector: 6 percentages for the proportion of products from each warehouse and 2 values range 0~1.0 for the location in 2D space.

As Q3 mentioned, In this and later steps , we define a distance between two data vectors a and b as following.

$$D(a,b) = \sqrt{\alpha \sum_{i=0}^{5} (a_i - b_i)^2 + (1-\alpha) \sum_{i=6}^{7} (a_i - b_i)^2}$$

Notice again that the first 6 values is the product proportions in each “topics” and the last 2 values are the spacial information, the α means the weight of each part. Since we do not know the value of α, it will set as a parameters for our clustering model so we can find the best definition of distance.

After first sampling and clustering, we have new centroids of each clusters. In the second step, we continue assign the remaining un-sampled points to clusters according to the same definition of distance. However, to solve the balancing problem as Q4 mentioned, we need to adjust our k-means algorithms as follow:



```
for j in remainingPointsIndex:
    p = pointTopicPortionList[j]
    p1 = np.array(p)
    # init
    minDistance = 100000000
    belongIndex = -100000000
    # first round
    for i in xrange(len(clusterList)):
        if minVolume[i] > 0:
            p2 = np.array(centroidList[i])
            if distance(p1,p2) < minDistance:
                minDistance = distance(p1,p2)
                belongIndex = i
    if belongIndex > -1:
        minVolume[belongIndex] -= 1
        clusterList[belongIndex].add(j)
    # else second round
    else:
        for i in xrange(len(clusterList)):
            p2 = np.array(centroidList[i])
            if distance(p1,p2) < minDistance:
                minDistance = distance(p1,p2)
                belongIndex = i
        if belongIndex > -1:
            minVolume[belongIndex] -= 1
            clusterList[belongIndex].add(j)
```

Figure 4: The new algorithms consider the balance constrains.

Notice in the new algorithms, we calculate the shortfall (*minVolume* in figure 4) to reach the minimal capacity for each cluster, that is the balance fraction M. The remaining points are assigned in two round: in the first round, only the cluster still has the shortfall can be the candidate, and the points are assigned to the nearest cluster by the distance we discussed previously. After several assignments, all clusters reach to the minimal requirement of capacity for balancing. If there are still un-assigned points, those points are simply greedily assigned to the nearest cluster. In this way, the balancing constrains are satisfied.

In the population step, only the *minVolume* counts down, the centroids of clusters are not updated until the next step refining. In the refining step, each point is re-assigned to other cluster if the total distortion function derived from distance function decrease and balancing still satisfied. Each step the centroids updates, until the change of total distortion lower than a threshold, then the whole job done and we calculate the total cost. By those two steps of updating, the computation cost can decrease significantly comparing to update each time.

Result

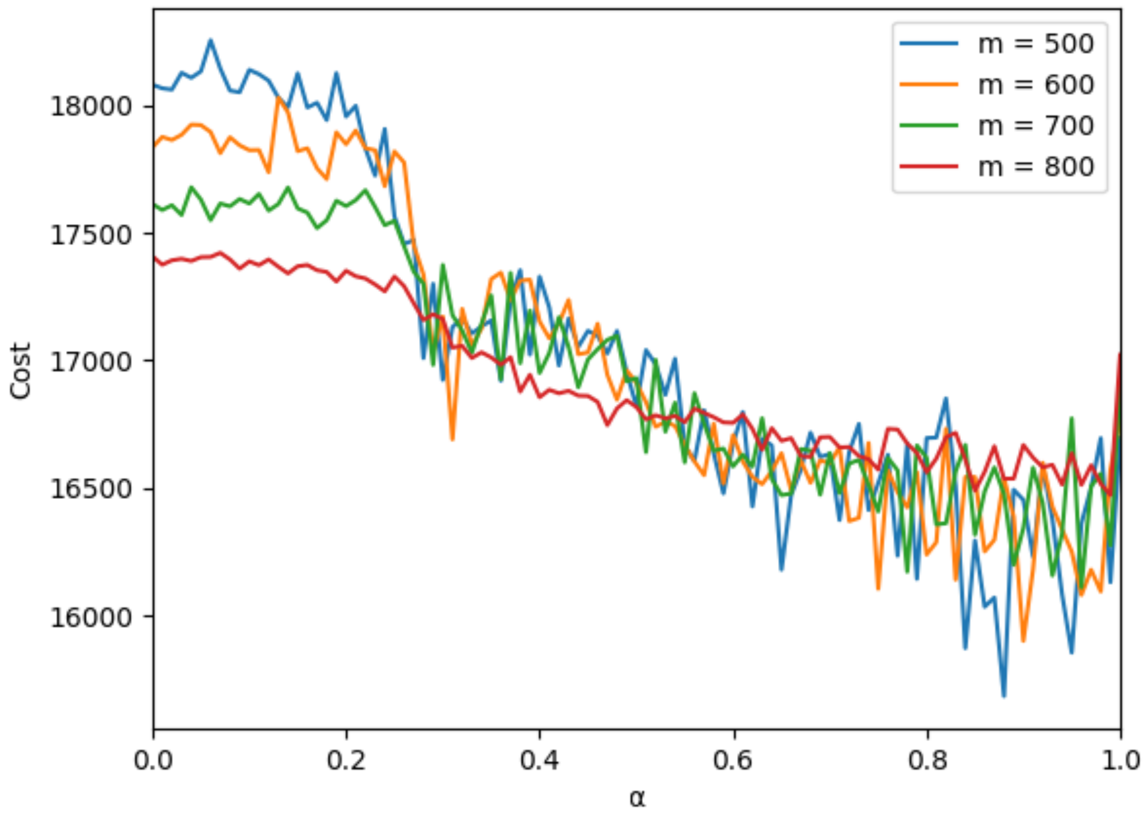


Figure 5: The real cost from different cluster results

The figure 5 shows how the result of our experiment, the “real” cost if we follow each clustering results to batch the orders. The α is the weight of product portion part in distance, and the curves of different color shows the corresponding balancing constrains.

Discussion

We can clearly see several interesting things from the figure 5:

(1) At the left side of the picture, when α equals 0, the clustering only depends on the distance of customers to distribution centers, as you can see, the result of cost is overall higher than the others considerations of distance. That means indeed we can not simply assigned each customer from their location, but also the similarity of the shopping list.

(2) As the α value increase, the cost drops quickly, and when α reaches about 0.9, all curves reaches at their lowest point. That means, to our surprise, the similarity of shopping list of the orders is actually play more important role in this experiment!

(3) When $\alpha = 1.0$, which is right edge of the picture we can see a dramatical increase of the cost, that tells us that consider only considering the similarity of shopping list either not works. In general, the curves do have "U" shapes as we expected. Although we empirically set $C_1 = 1.0$, $C_1 = 2.0$ in this simulation and α may changes if in real situation the cost of delivery changes, we can still use the same analysis framework to get best value α if we collects financial cost of delivery in real company operations. When α in its best value, the cost can drop 18000 to16000, nearly 10%~12%, which is a huge saving for logistics company!

(4) Except in extreme cases when $\alpha < 0.3$, which are obviously too high cost in all, in general, the less balancing constrains will lower the cost. But as we discussed, the storage cost is implicitly represented by the balancing fraction. In business terms, this investment and storage cost is called "Fixed Costs", which in total do not increase with respect to increase of sales and costs per product can be "diluted", while the delivery cost is "Variable Costs" and it will increase as sales increase. We should consider both costs in practice and may also consider other factors when apply our methods to the real case, but in most business cases, if the company can find the lowest "Variable Costs" and make sure the price higher than the "Variable Costs", the "Fixed Costs" will eventually diluted, in other words, companies consider "Variable Costs" more important than "Fixed Costs".Thus, our method has more practical meaning in business.

From above discussions, we can finally draw to a conclusion and answer the question Q5. Indeed, by using relatively simple clustering and choosing proper α value, we could help company saves money, and it performs at least better than simple batching by shopping-list similarity or simple batch by customer locations.

References

[1] UCL Machine Learning Repository: http://archive.ics.uci.edu/ml/datasets/online+retail

[2] Chapter 8: Basu S, Davidson I, Wagstaff K L. Constrained Clustering [M]. CRC press. 2008: 171-229.

[3] Neilson Marketing Research. Category Management: Positioning Your Organization to Win. McGraw-Hill, 1993.

Appendix

The source code of the project is attached in this report: see attached file Code.zip