UDC 330.36.012.4

COMPETITORS DETECTION USING GRAPH ANALYSIS

Hnot T.

Online markets could be analysed accurately by Internet shops, which propose wide range of products from different manufacturers. It gives them an ability to access purchases data of manufactures of the same product type, so – to detect competitors and as a result – perform marketing in a clever way. In this thesis we are proposing an approach to detect clusters of competitors in an unsupervised way by accessing transactional customers' data.

As an example dataset lets use generated transactional dataset of restaurants orders. To simplify explanation, restaurants business was used in the thesis, but this approach could be easily extrapolated on different markets. Example of data is in Table 1.

Table 1. Example of data for analysis

Order ID	Restaurant ID	Order Coordinates (lat;long)
1	95723	(41.85380; -87.85715)
2	272266	(41.04881; -87.66910)
3	184276	(41.88268; -87.32582)
4	275233	(41.23933; -87.50034)
5	164019	(41.53582; -87.83427)

Algorithm, which is described here to divide restaurants on competitors groups contains two stages:

1. Build graph[1], where each node represents restaurant and edge between nodes – strength of connection between restaurants. Higher weight of edge between restaurants indicates that these restaurants more probably should be placed in the same group. Weights of edges are calculated in the next way:
    a) built lists of restaurants, from which orders were made to the same place (based on coordinates of orders). We took into account that restaurants can't delivery further their delivery zones;
    b) for each list built all combinations of restaurants from list;
    c) weight of each combination equals minimum of number of orders to these restaurants.

For example, we have 5 orders to point (x, y) from "Restaurant_A", "Restaurant_A", "Restaurant_B", "Restaurant_B", "Restaurant_C". As a result we are receiving 3 combinations: Restaurant_A–Restaurant_B, Restaurant_A–Restaurant_C, Restaurant_B–Restaurant_C with weights: 2, 1, 1. Weight between Restaurant_A and Restaurant_B is 2, because we have 2 orders to each of the restaurants (min(2, 2)=2).

    d) after, combinations from all lists are combined together by summing weights;

2. Cluster graph by using graph techniques and tune clustering using distances between restaurants.

To cluster graph label propagation algorithm[2] was used. It works pretty fast and returns reasonable clusters. There is no need to specify number of clusters or any tuning parameter. One disadvantage of this algorithm: it does not return stable clusters. To deal with that, algorithm was run multiple times and result with minimum average within cluster distance between restaurants was chosen (to make restaurants in one group be more crammed together).

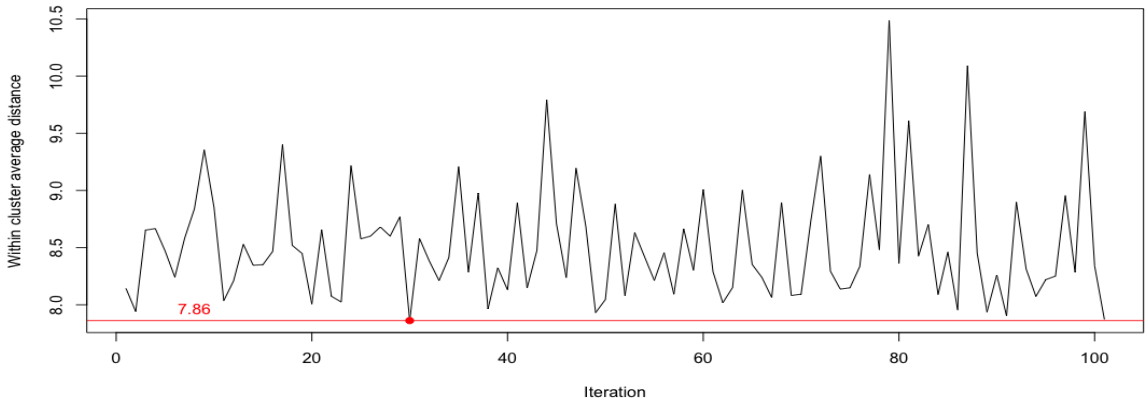

Figure 1. Clustering running results

Below are two visualizations. First one shows all restaurants with dots colored by competitors group index. Second – centers, calculated based on restaurants' coordinates.

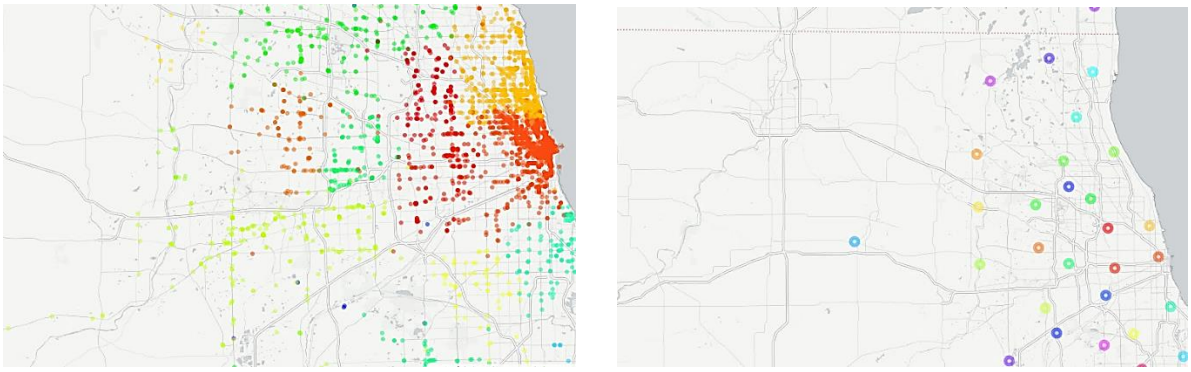

Figure 2. Detected competitors groups

Clustering of restaurants in described way gave us an ability to detect groups of restaurants, which are characterized by orders from the same customers. So, there could be 2 reasons of that:

1. Restaurants are characterized by the same food type, so they are really competitors, as customers order from both of them;
2. Restaurants are characterized by different food type, so they could not be completely treated as competitors. Here we need additionally analyse preferences of customers.

As could be view on Figure 2, restaurants of the same clusters are crammed together. It's because users often order foods from restaurants, which are near their places.

In the following example we have used ~ 10 millions transactions, and label propagation clustered graph just in few seconds. So, such approach also could be used as starting point for competitors detection in huge datasets.

REFERENCES

[1] Graph theory: https://en.wikipedia.org/wiki/Graph_theory

[2] Raghavan, Usha Nandini and Albert, R'eka and Kumara, Soundar. Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E, 2007