**TD leads model**

* **M. Kaleeswaran, Analytics Intern (Swiggy Delhi), under guidance of Mr. Vinit Kumar**

**Objective:**

To provide leads to sales team for four consecutive weeks (a month) of the best set of restaurants that would increase the Swiggy’s Trade Discount (TD) order coverage, by considering various parameters determining the buyers’ behavior.

**Basic outline:**

Basic outline of the model is that it works on a points system. Suppose **k**, number of restaurants out of **n**, restaurants are set as target to run TD in a zone, the model samples all combinations of **r** (**r = k + x; x >= 0**) out of **n** (a total of number of combinations) and decides which sample has the highest points (based on the equation to be discussed later) and provide it as the leads to the sales team.

Note: The number of restaurant to be given as leads is set at **r** instead of **k** to enable easy conversion.

**Parameters under consideration:**

The various parameters under consideration and its usage will be discussed here.

* **Number of orders:** The number of orders done by a restaurant is an indicator of its demand and reputation with the people. This parameter will be factored into the equation as, the average number of orders done by all the restaurant in the particular sample of the combinations.
* **Latitude and Longitude:** The purpose of using the geographical location of a restaurant is to break a zone into closer and equally sized sub-divisions and sample the combinations based on this sub-division, rather than sampling the combinations as a whole zone. This would ensure that the restaurants are not been chosen to run TD on is not been clubbed in a particular part of the zone only. This would get much clearer as I discuss the execution part of the model.
* **Cuisines:** Cuisine is one the most important parameters of the model. The equation ensures that the final leads of restaurants been provided, maximizes the total number of cuisines been provided and these cuisines are equally distributed in frequency. The equal distribution in frequency is to ensure that there is no skewness to the number of each cuisines been provided.
* **Cost For Two (CFT):** CFT has a similar implication as Cuisines. Goal is to maximizes the total number of CFT ranges been provided and ensure that these CFTs are equally distributed in frequency. The restaurant CFT is been made into ranges in this model.
* **Freshness:** Freshness coefficient, **α**, is to ensuring that there is no number of restaurants above a threshold value is passed on from the previous week to the next week. This makes sure customers get to see a TD running on a new set of restaurant each week.

The mathematically freshness coefficient will be:

**The equation:**

The equation this model works on is as below:

in jth sample

= total number of unique cuisines in jth sample

Calculation of this is a bit trickier. First a frequency distribution is created for each of the cuisines from jth sample and each of these values are subtracted from the mean of the frequencies and summation of this is a measure of how dispersed the frequencies are from the mean. This value would equal

And are CFT alternatives to similar measure for cuisine.

Ws would be corresponding reasonable weightages assigned to each of the factors.

This point would be calculated across all samples and the one with maximum value would be given as leads for that particular week.

**The implementation:**

The whole model is been implemented in R-programming language. It is advised to read this section along with the program been attached as a supplementary reading material. This section would be explained with briefing each of the functions in the program.

The functions in use:

* Separate\_data
* as\_radians
* calc\_dist
* direchletClusters\_constrained
* Sample\_Compilation

**separate\_data:**

This function takes in the data we use, the zone which is to be we need the TD leads on and the size of the equally sized clusters we need. This makes a call to the function direchletClusters\_constrained (will be explained) with the zonal data. This returns the zonal data with the additional factor column as cluster number the each restaurant in the zone belongs to.

**direchletClusters\_constrained:**

This function is called from the separate\_data function with the arguments as the Latitudes and Longitudes of all the restaurant belonging to the zone. This function is a modification to the K-means algorithm (knowledge of k-means algorithm makes understanding of this function easier). K-means is an iterative classification algorithm, which classifies the variables based on the distances (the closest getting clustered together). It might happen that in K-means, clusters might not be of unequal sizes and this makes our model irrelevant. So modification is made to the K-means algorithm where-in at the end of each iterative step, from the cluster where cluster size is greater than that of the desired size, the farthest point from that cluster’s center will be pushed to the next closest cluster. This process in the same iterative step will happen for all the cluster with sizes greater than the desired. Now the next iterative step where-in the adjustment of cluster centers, allocation of points to cluster center and the previously described steps happen. These iterations continue until there is not much significant movement of cluster centers taking place (centers comes to a standstill) or when the maximum number of iterations been passed as an argument has been reached. This function returns the latitude and longitude data with the corresponding clusters to the separate\_data function and the cluster column is bind to the zone data there.

**as\_radians:**

This is a simple function taking in the arguments as latitudes and longitudes and converting it to radians. Call to this function is made from the direchletClusters\_constrained function.

**calc\_dist:**

This function takes in the latitudes and longitudes in pairs and returns the geographical distances between them to the directletClusters\_constrained function to implement the modified k-means algorithm. The formula used is Haversine formula which accounts for elliptical nature of earth.

**Sample\_Compilation:**

This is the most fundamental and primary function to our model. This would be discussed in much brief. It takes in call from the global environment (the final for-loop in the program) with arguments passed as each separate clusters of the zone. Now each of these cluster are sampled into all the combinations of the desired size. These samples are made a list. Inside the loop the various parameters been mentioned would be calculated as per the formulated equation for each of the samples in the list. These results are compiled within the loop as a dataframe called “decider”. The sample in the decider dataframe which has the highest value of points will be considered as the leads to be given for that cluster belonging to a zone for week 1. Now complication arises as we need to consider the freshness as a factor for giving leads for week 2. This is overcome by selecting only those samples from the clusters, which has the number of matching restaurant id, between each sample and the lead been provided on week 1, less than the threshold value as per the freshness coefficient been allotted. Similarly between week 3 and week 2, also between week 4 and week 3. For week 3 and week 4, the samples with 2nd highest point will be chosen. This is from the basic logic that week 3 and week 1, also week 2 and week 4 restaurant list would be exactly similar, since there is freshness set between these.

**global\_environment:**

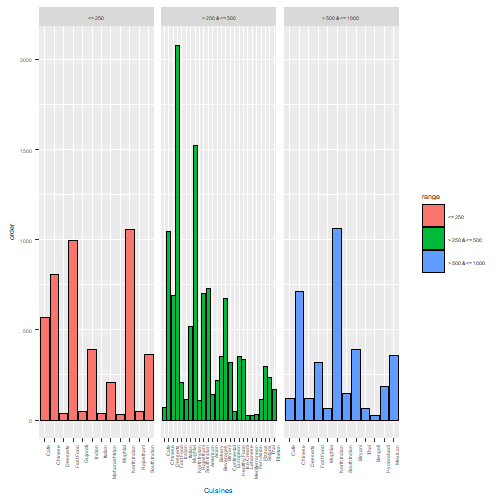
The global environment executes a for loop which would make a call to the separate\_data function for each of the zones and get the clustered data and make a call to the Sample\_Compilation function and receive the leads for week 1 to week 4 for the zones. This is also where it is ensured that none of the clusters exceed the size limit and small sized clusters are clubbed together, without exceeding the max size.

**Test run:**

The program is test run on all zones of Delhi, using the RHI June. The result of which is been attached. The weightages provided on the test run are as follows,

**Further developments and versatility:**

Plans are been made to create a distribution for cuisine and CFT which would exactly match the demand been observed. Analysis is been made in that direction.



This plot is the demand distribution in Cyber City during the month of May. It is been auto-generated for all the zones from R-function made as part of another project. The cuisine and CFT part of equation in the TD model is been worked upon to match this distribution, rather than be distributed equally, as discussed before. However each has its own obvious merits and demerits.

The model currently is highly versatile, the weights are highly flexible, it could tweaked well to optimize the final leads, new parameters could be easily added to equation.