Procedure to Analyse the Clients via the Proof of Concept Developed during the program S2DS 2017 London

Author:
Daniele Scopece – daniele.scopece.science@gmail.com
In case of query contact me via email (or LinkedIn) with subject: “S2DS-Criteo-Scripts”
During the event Science2DataScience London 2017-08/2017-09

What is this document about

During the program S2DS 2017 in London, we have developed a series of python scripts as a Proof of Concept to analyse the clients active in a given period of time. This allows to extract information about number of clients active, the total amount spend (i.e. Criteo’s revenues) and other. They also allow to plot graphs as a function of time. Finally we propose an alternative way to classify the clients that goes beyond the current classification into Tier1 and MMS.
In this document we describe the reasoning behind every step and how to use the scripts in order to allow further developments in the future.
Here we show for a selection of variables, the scripts can be modified to plot and analyse other features.

Business Questions tackled

- Is criteo neglecting some clients that deserve some treatment as the Tier1?
- How is one client performing with time wrt the other clients in the same RETAIL and Country?
- How can we identify the clients that are closer to a given value of the features?

Overview of the Possibilities

1. Extract the tables of the given periods via a python script = 01.01
2. Plot the data for a given set of features (in log scale) = 02.01
3. Select the clients whose features fall in a given range = 02.02
4. Plot sum of displays, clicks, revenues etc vs time = 02.03
5. Compute the correlations for the features = 03.01
6. Identify new classifier for the clients = classes on revenues, reach = 04.01
7. Plot the numbers of clients in a certain classifier and computes the percentage of them that are in Tier1 by Criteo = 05.01
8. Plot the trend with time of the classifier of a client = 06.01

Structure of the scripts

The scripts are kept separated, so that they can be run independently and for clarity of purposes.

File 01.01: extract the csv tables

1. Modify the file 00.00.PARAMETERS.txt
   Put the start, end date and the period.
   Important: start in the first column, do not leave spaces, otherwise it does not work.
2. Run in python the file
   01.01.Run_vertica_NEW.v04.cycle.py
   Explanation: this runs the file
   vertica_queries/Extract_Client_for_Clustering.v03.01.OK-variable-for-python.sql
   for the different dates specified in a loop
3. The csv files will be created in the folder 01.01.RES-CSV-TABLES (that has been created)

This extract the useful information of the clients and connects the client_id with the merchant_id, the name of the client and the fact whether they are tier1 or not according to Criteo.
We use the following tables from Vertica:

- lta.bi_client_galactica
- datamart.dim_client
- datamart.fact_client_stats_daily_euro
- datamart.dim_merchant

For sake of simplicity, we neglect the funnel.
Output files into: 01.01.RES-CSV-TABLES (that must be created before running the script)
Dependencies from files: none

File 02.01: Plot the trends

Here we plot the trends of a variable in the graph in order to visualize the data.
Script = 02.01.ClientID-Visualize.v05.py
We first decide the parameters to plot in the vector POI.
Then we plot them in the log scale.
Output files into: 02.01.RES-Plots (that is created automatically)
Dependencies from files: 01.01

File 02.02: Select the clients with a given set of features

Here we can select the clients whose (derived) features fall in a given range and we highlight them on the log-log plot.
Script = 02.02.ClientID-Visualize-Interval.v04.py
This is written for the same choice of parameters as in 02.01: an extension can be done seamingless if needed.
The user must insert the values desired into the variable Wanted_center_input and the error bar in the variable Wanted_radius_input. Then the program converts into log and plots the results. The user must insert np.nan for the variables that he does not want to select.
Reminder: The conversion to log shifts the center and the radius.
Reminder: Ignore the warning of division by 0: then we have is_finite and we ignore them.
It also writes a csv file with feature is_inside to identify with 1 the clients chosen and sorts them by distance in log scale: this means that it is sorted by log(point_1) – log(point_2).

Output files into: 02.02.RES-Plots-Interval (that is created automatically)
Output files:

- figure in log-log
- figure in log-log with highlighted the points inside the interval
- csv of the points inside not sorted by distance (in log scale)
- csv of the points inside sorted by distance (in log scale)
- csv file of number of points inside vs date
- figure of number of points inside vs date

Dependencies from files: 01.01

Execution: this script can take time, since it is checking the tables row by row

File 02.03: Plot the trend with time

Here we analyse the trend with time of the sum of the clients active in a certain period of time.

Script = 02.03.ClientID-Plot-Trend-Time.v04.py
Output files into: 02.03.RES-Plots-Trend-Time (that is created automatically)
Output files:

- plot number of active clients vs time
- plot sum all actual displays vs time
- plot sum all audience vs time
- plot sum all clicks vs time
- plot sum all potential displays vs time
- plot sum all revenues (for Criteo = spent for the client) vs time

Dependencies from files: 01.01

Execution: this script is very fast.

File 03.01: Compute correlations of the features

Here we analyse the correlation between the features of the clients in log scale.
In this way we can eliminate some of them for our analysis.

Script = 03.01.ClientID-Correlations.v03.py

Output files into: 03.01.RES-Correlations (that is created automatically)
Output files:

- one image with the correlation heat map per date chosen

Dependencies from files: 01.01

Execution: very fast

File 03.02: Compute Hierarchical Clustering features

Here we deduce the hierarchical clustering and report the dendrogram (one per date).
The features are considered, as usual, in log scale.
This could serve to determine the number of clusters, if a cluster analysis is required.

Script = 03.02.ClientID-HierarchicalClustering.v03.py

Output files into: 03.02.RES-HierarchicalClustering (that is created automatically)
Output files:

- one image with the dendrogram per date chosen

Dependencies from files: 01.01

Execution: very fast

Conclusion: The clients cluster in a similar manner for all dates

File 04.01: Divide in Quartiles, novel classification of the clients

Here we define other classes of clients that goes beyond the classification in Tier1 and non-tier-1.
Based on the observation of File 03.01 we could exclude some variables.
Based on the observation of File 02.01, we have seen that the clients segment automatically in the plot log_sum_revenue_euro vs log_audience with the colouring given by the reach.
We hence segment them based on this observation with a classification that should be the same for all dates.
Additionally Criteo is interested in classifying the client based on their performance (and payment) in comparison to the other clients active in the same time, region, classification.

Hence we decide to classify the clients based on Quartiles of reach and log_sum_revenue_euro.
This means that we consider the max and min of the reach obtained in a given period by all clients active in that period (e.g. in the period ending 2017-08-01 that lasts 30 days). Then we divide it in 4 intervals of equal extension in the reach and attribute to each client a label based on their position in that period as follows.
For the log_sum_revenue_euro we have 4 groups:
Group 1: containing the clients with the value of this feature between 0% of the extension of the max-min and 25% of this interval
Group 2: lists of the client with this feature between 25.x% and 50% of this interval
Group 3: lists of the client with this feature between 50.x% and 75% of this interval
Group 4: lists of the client with this feature between 75.x% and 100% of this interval
In this way the higher the index of the group, the higher the performance of the clients inside it.
We do this classification for log_sum_revenue_euro and the reach independently of each other, so that an independent search can be made.
The script can be modified to include other features in the classification.

We save the files with the client_id, merchant_id and a lable to check if they are classified into tier1 by Criteo. This would be useful to make an analysis.

Script = 04.01.ClientID-Quartiles.v03.py

Output files into: 04.01.RES-Quartiles (that is created automatically)
Output files:

- csv file of the classification indexes into quartiles for reach and log_sum_revenue_euro for each date
- csv file of the clients in each set of index reach and log_sum_revenue_euro with the merchant_id and the label to identify whether they are classified as Tier1 by Criteo
- csv file for each date with the number of clients in each combination of sectors
- Figure to highlight the points in each subgroup in the plot

Dependencies from files: 01.01

Execution: fast

Conclusion: Good classification

File 04.02: Take the Quartiles Classification and Plot the points on a graph

Here we take the classification made in File 04.01 and plot the graphs.
Also we see how many of the clients in each sector are classified as Tier1 by Criteo: this answer to one of the business questions above.

Script = 04.02.ClientID-Quartiles-Plots-e-Graphs.v05.py

Output files into: 04.02.RES-Quartiles-Plots-e-Graphs and subfolders (that are created automatically)

Output files:

- Figure with the tier1 points highlighted for each date
- Figure to highlight the points in each subgroup in the plot for each date
- Heatmap for each date of
  the number of clients in each sector
  the number of clients that are Tier1 in each sector
  the percentage of clients that are Tier1 in each sector
-

Dependencies from files: 01.01 + 04.01

Execution: fast

Conclusion: Analyse the trends

Limitations: this version of the script works for 4 quartiles only

File 04.03: Take the Quartiles Classification and Plot the trend with time of the population of a group

Here we take the classification made in File 04.01 and plot the graphs of the evolution of the number of clients in each group and of tier 1 in each group.

Script = 04.03.ClientID-Quartiles-Plots-Population-vs-Time.v03.py

Output files into: 04.03.RES-Quartiles-Plots-Population-vs-Time and subfolders
(that are created automatically)

Output files:

- Csv files with date, period, max_sector, number of clients in each sectors, number of clients in each sectors that are tier1 for each combination of index of sectors
- Figure of number of clients in sector 1,1 and client of tier 1 vs date
- Same for all other sectors

Dependencies from files: 01.01 + 04.01

Execution: fast

Conclusion: Analyse the trends of the clients with time statistically

Limitations: this version of the script works for 4 quartiles only

File 04.04: Take the Quartiles Classification and Plot the trend with time of a specified client list

Here we take the classification made in File 04.01 and plot the graphs of the evolution of the Reach and Revenue index as a function with time.

Script = 04.04.ClientID-Quartiles-Plots-Idx1Client-vs-Time.v06.py

Output files into: 04.04.RES-Quartiles-Plots-Idx1Client-vs-Time (that is created automatically)

The list of the client_id are read from the file 04.04.LIST-Client_id.txt that must have the followi9ng format:
first line = comment
from second line onwards: number of the client_id, one client_id per line, leave one blank space before carriage return.

Output files:

- Images of the evolution of the index with date
  The index is 0 or -1 when the client is not active in the given date
  The figures also show the classification of the client as Tier 1: 0 = is not tier 1, 0.5 = it is tier1

Dependencies from files: 01.01 + 04.01

Execution: fast

Conclusion: Analyse the trends of the clients with time

Limitations: this version of the script works for 4 quartiles only