Consumer Behaviour Analytics Tutorial - Practical Case

Statement

Using `data/LondonCustomer.csv` which contains the following information of the clients of a London bank:

- CONTACT_ID: client ID.
- AGE: age.
- FAMILYSIZE: family size.
- YEAREXPERIENCE: year of experience.
- ANNUALINCOME: annual income.
- EDUCATIONLEVEL_ID: education level ID.
- NETPRICE_PRO11_AMT: total consume of PRO11 product.
- NETPRICE_PRO12_AMT: total consume of PRO12 product.
- NETPRICE_PRO13_AMT: total consume of PRO13 product.
- NETPRICE_PRO14_AMT: total consume of PRO14 product.
- NETPRICE_PRO15_AMT: total consume of PRO15 product.
- NETPRICE_PRO16_AMT: total consume of PRO16 product.
- NETPRICE_PRO17_AMT: total consume of PRO17 product.
- name: office's name of London district where it belongs.

With `data/London_sports` data the following business cases have to be resolved:

Section 1. Company has 33 offices in London, each one per district, and in light of lack of profitability it need close three of these offices. Company has decided that it will do it with offices which has a lower business volume (sum of all products's consume) of clients lower than 55 years.

Section 2. Also, Company wants to know, for each closed districts, if there is an office or offices locate in a near district where it can move the clients if it is necessary. For that, it is considered that the offices are geolocated in the center of the its districts.

---

0. Read Data

0.0. Clean all and import libraries

```
# Clear all object
rm(list=ls())
# Set working directory
setwd("/home/jmssalas/git/master-bigdata-businessanalytics/08-analytical-applications/03-location-analy
```

```
# Import needed libraries
is.installed <- function(package) is.element(package, installed.packages())

if (!is.installed('rgdal'))
  install.packages('rgdal', dependencies = T)

if (!is.installed('rgeos'))
  install.packages('rgeos', dependencies = T)

if (!is.installed('tmap'))
  install.packages('tmap', dependencies = T)
```

```r
if (!is.installed('OpenStreetMap'))
  install.packages('OpenStreetMap', dependencies = T)

library(rgdal)
```

```
## Loading required package: sp
```

```
## rgdal: version: 1.3-6, (SVN revision 773)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.2.3, released 2017/11/20
##  Path to GDAL shared files: /usr/share/gdal/2.2
##  GDAL binary built with GEOS: TRUE
##  Loaded PROJ.4 runtime: Rel. 4.9.3, 15 August 2016, [PJ_VERSION: 493]
##  Path to PROJ.4 shared files: (autodetected)
##  Linking to sp version: 1.3-1
```

```r
library(rgeos)
```

```
## rgeos version: 0.4-2, (SVN revision 581)
##  GEOS runtime version: 3.6.2-CAPI-1.10.2
##  Linking to sp version: 1.3-1
##  Polygon checking: TRUE
```

```r
library(tmap)
library(OpenStreetMap)
```

0.1 From LondonCustomer CSV

```r
london_customer <- read.csv("data/LondonCustomer.csv", sep = ";", header = T)
head(london_customer)
```

```
##   CONTACT_ID AGE FAMILYSIZE YEAREXPERIENCE ANNUALINCOME EDUCATIONLEVEL_ID
## 1        395  25          4              1           49                 1
## 2        396  45          3             19           34                 1
## 3        397  39          1             15           11                 1
## 4        398  35          1              9          100                 2
## 5        399  35          4              8           45                 2
## 6        400  37          4             13           29                 2
##   NETPRICE_PRO11_AMT NETPRICE_PRO12_AMT NETPRICE_PRO13_AMT
## 1                  0                  1                  0
## 2                  0                  1                  0
## 3                  0                  0                  0
## 4                  0                  0                  0
## 5                  0                  0                  0
## 6                155                  0                  0
##   NETPRICE_PRO14_AMT NETPRICE_PRO15_AMT NETPRICE_PRO16_AMT
## 1                 16                  0                  0
## 2                 15                  0                  0
## 3                 10                  0                  0
## 4                 27                  0                  0
## 5                 10                  1                  0
## 6                  4                  0                  1
##   NETPRICE_PRO17_AMT               name
## 1                  0           Lewisham
## 2                  0            Enfield
```

```
## 3                  0       Waltham Forest
## 4                  0 Barking and Dagenham
## 5                  0              Hackney
## 6                  0               Barnet
```

0.2. From London Sport SHP file

```
# Get London data in SHP format
i_data_gs <- readOGR(dsn = "data", layer = "london_sport")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "/home/jmssalas/git/master-bigdata-businessanalytics/08-analytical-applications/03-location-a
## with 33 features
## It has 4 fields
## Integer64 fields read as strings:  Pop_2001
```

```
i_data_gs@data$Pop_2001 <- as.numeric(as.character(i_data_gs@data$Pop_2001))
head(i_data_gs@data)
```

```
##   ons_label                 name Partic_Per Pop_2001
## 0      00AF              Bromley       21.7   295535
## 1      00BD Richmond upon Thames       26.6   172330
## 2      00AS           Hillingdon       21.5   243006
## 3      00AR             Havering       17.9   224262
## 4      00AX Kingston upon Thames       24.4   147271
## 5      00BF               Sutton       19.3   179767
```

Section 1.

1.1. Get the clients which has a age lower than 55

```
clients_lower_55 <- london_customer[london_customer$AGE < 55, ]
max(clients_lower_55$AGE)
```

```
## [1] 54
```

```
head(clients_lower_55)
```

```
##   CONTACT_ID AGE FAMILYSIZE YEAREXPERIENCE ANNUALINCOME EDUCATIONLEVEL_ID
## 1        395  25          4              1           49                 1
## 2        396  45          3             19           34                 1
## 3        397  39          1             15           11                 1
## 4        398  35          1              9          100                 2
## 5        399  35          4              8           45                 2
## 6        400  37          4             13           29                 2
##   NETPRICE_PRO11_AMT NETPRICE_PRO12_AMT NETPRICE_PRO13_AMT
## 1                  0                  1                  0
## 2                  0                  1                  0
## 3                  0                  0                  0
## 4                  0                  0                  0
## 5                  0                  0                  0
## 6                155                  0                  0
```

```
##   NETPRICE_PRO14_AMT NETPRICE_PRO15_AMT NETPRICE_PRO16_AMT
## 1                 16                  0                  0
## 2                 15                  0                  0
## 3                 10                  0                  0
## 4                 27                  0                  0
## 5                 10                  1                  0
## 6                  4                  0                  1
##   NETPRICE_PRO17_AMT                name
## 1                  0            Lewisham
## 2                  0             Enfield
## 3                  0      Waltham Forest
## 4                  0 Barking and Dagenham
## 5                  0             Hackney
## 6                  0              Barnet
```

1.1. Get the sum of all products's consume

```
# Products
products <- c("NETPRICE_PRO11_AMT", "NETPRICE_PRO12_AMT", "NETPRICE_PRO13_AMT", "NETPRICE_PRO14_AMT", "
```

Let's use the `str()` function to discover the products values:

```
str(clients_lower_55[, products])
```

```
## 'data.frame':    3641 obs. of  7 variables:
##  $ NETPRICE_PRO11_AMT: int  0 0 0 0 0 155 0 0 104 0 ...
##  $ NETPRICE_PRO12_AMT: int  1 1 0 0 0 0 0 0 0 0 ...
##  $ NETPRICE_PRO13_AMT: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NETPRICE_PRO14_AMT: Factor w/ 108 levels "0","1","10","100",..: 11 10 3 27 3 45 10 31 73 105 ...
##  $ NETPRICE_PRO15_AMT: int  0 0 0 0 1 0 0 1 0 0 ...
##  $ NETPRICE_PRO16_AMT: int  0 0 0 0 0 1 1 0 1 0 ...
##  $ NETPRICE_PRO17_AMT: int  0 0 0 0 0 0 0 0 0 1 ...
```

```
# Let's see the factor's levels
levels(clients_lower_55$NETPRICE_PRO14_AMT)
```

```
##   [1] "0"    "1"    "10"   "100"  "11"   "12"   "13"   "13,3" "14"   "15"
##  [11] "16"   "16,7" "17"   "17,5" "18"   "19"   "2"    "20"   "21"   "22"
##  [21] "23"   "23,3" "24"   "25"   "26"   "26,7" "27"   "27,5" "28"   "29"
##  [31] "3"    "30"   "31"   "32"   "32,5" "33"   "33,3" "34"   "35"   "36"
##  [41] "36,7" "37"   "38"   "39"   "4"    "40"   "41"   "42"   "42,5" "43"
##  [51] "43,3" "44"   "45"   "46"   "46,7" "47"   "47,5" "48"   "49"   "5"
##  [61] "50"   "51"   "52"   "53"   "53,3" "54"   "55"   "56"   "56,7" "57"
##  [71] "58"   "59"   "6"    "6,7"  "60"   "61"   "62"   "63"   "63,3" "64"
##  [81] "65"   "66"   "66,7" "67"   "68"   "69"   "7"    "7,5"  "70"   "72"
##  [91] "73"   "74"   "75"   "76"   "78"   "79"   "8"    "80"   "81"   "82"
## [101] "83"   "85"   "86"   "88"   "89"   "9"    "90"   "93"
```

As we can see, the `NETPRICE_PRO14_AMT` column is not a numeric value. So, we have to convert this factor value to numeric value, using first `gsub()` function to replace `,` to `.`.

```
clients_lower_55$NETPRICE_PRO14_AMT <- as.double(gsub(",", ".", clients_lower_55$NETPRICE_PRO14_AMT))
str(clients_lower_55[, products])
```

```
## 'data.frame':    3641 obs. of  7 variables:
```

```
##  $ NETPRICE_PRO11_AMT: int  0 0 0 0 0 155 0 0 104 0 ...
##  $ NETPRICE_PRO12_AMT: int  1 1 0 0 0 0 0 0 0 0 ...
##  $ NETPRICE_PRO13_AMT: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NETPRICE_PRO14_AMT: num  16 15 10 27 10 4 15 3 6 89 ...
##  $ NETPRICE_PRO15_AMT: int  0 0 0 0 1 0 0 1 0 0 ...
##  $ NETPRICE_PRO16_AMT: int  0 0 0 0 0 1 1 0 1 0 ...
##  $ NETPRICE_PRO17_AMT: int  0 0 0 0 0 0 0 0 0 1 ...
```

Now, we can already make the sum of all products

```
clients_lower_55$SUM_CONSUME <- rowSums(clients_lower_55[, products])
head(clients_lower_55[,c(products, "SUM_CONSUME")])
```

```
##   NETPRICE_PRO11_AMT NETPRICE_PRO12_AMT NETPRICE_PRO13_AMT
## 1                  0                  1                  0
## 2                  0                  1                  0
## 3                  0                  0                  0
## 4                  0                  0                  0
## 5                  0                  0                  0
## 6                155                  0                  0
##   NETPRICE_PRO14_AMT NETPRICE_PRO15_AMT NETPRICE_PRO16_AMT
## 1                 16                  0                  0
## 2                 15                  0                  0
## 3                 10                  0                  0
## 4                 27                  0                  0
## 5                 10                  1                  0
## 6                  4                  0                  1
##   NETPRICE_PRO17_AMT SUM_CONSUME
## 1                  0          17
## 2                  0          16
## 3                  0          10
## 4                  0          27
## 5                  0          11
## 6                  0         160
```

1.2. Get the three offices which have a lower business volume

```
# Import dplyr library
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:rgeos':
##
##     intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Get the sum of all product's consumes of all clients of each districts
clients_lower_55 %>%
  group_by(name) %>%
  summarise(SUM_CONSUME = sum(SUM_CONSUME)) %>%
  arrange(SUM_CONSUME)
```

```
## # A tibble: 33 x 2
##    name                    SUM_CONSUME
##    <fct>                         <dbl>
##  1 City of London                 62.7
##  2 Wandsworth                    501
##  3 Ealing                        714
##  4 Bromley                       776.
##  5 Hammersmith and Fulham       1011
##  6 Lambeth                      1042.
##  7 Haringey                     1174
##  8 Greenwich                    1223.
##  9 Hounslow                     2162
## 10 Southwark                    2825.
## # ... with 23 more rows
```

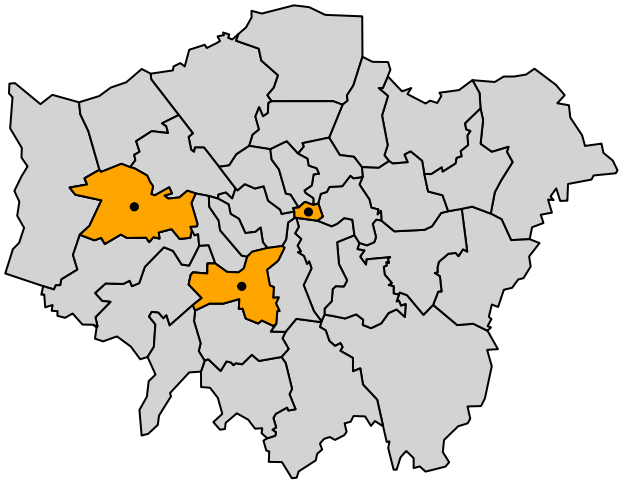
With that, we can say that the three offices which have a lower business volume are:

- City of London
- Wandsworth
- Ealing

Section 2

As we assume that the offices are geolocated in the center of its districts, we are going to get the centroid of each closed office:

```r
# Set the closed offices
closed_offices <- c("City of London", "Wandsworth", "Ealing")

# Get its information
i_data_closed_offices <- i_data_gs[is.element(i_data_gs@data$name, closed_offices), ]

# Get the Centroid of these offices
cent_closed_offices <- gCentroid(i_data_closed_offices, byid = T,  id = closed_offices)

# Plot the location of theses offices
plot(i_data_gs, col = "lightgrey")
plot(i_data_closed_offices, add = T, col = "orange")
points(cent_closed_offices, cex = 0.5, col = "black", pch = 19)
```



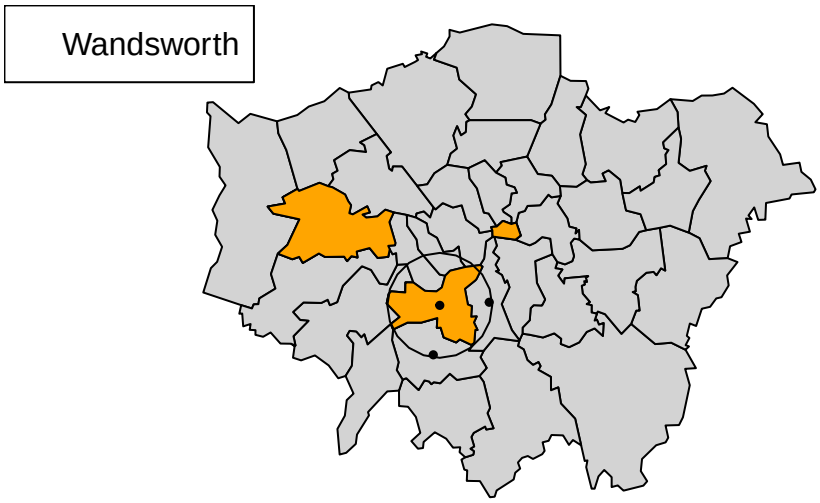
Now, we are going to get what open offices are near to the closed offices. For that, we are going to assume that a office is near to other if there are 5km between them. First at all, let's get the centroids of all open offices.

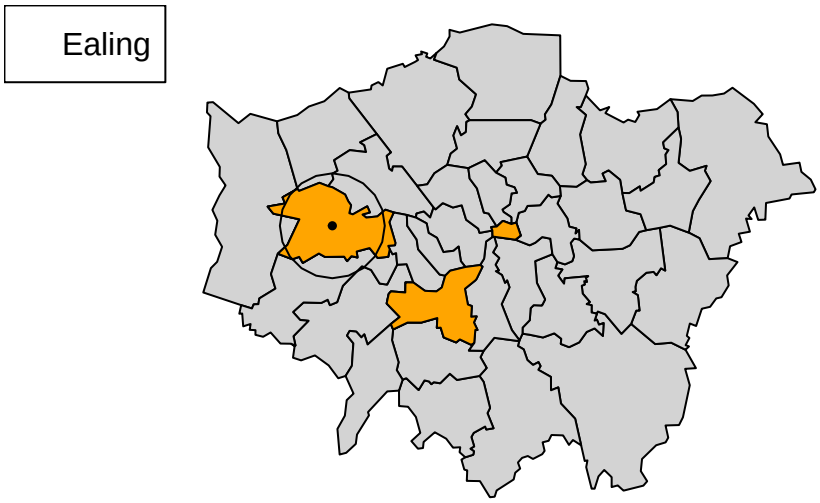
```
# Get all districts where Company has its offices
districts <- unique(london_customer$name)

# Remove closed offices
open_offices <- districts[!districts %in% closed_offices]

# Get its information
i_data_open_offices <- i_data_gs[is.element(i_data_gs@data$name, open_offices), ]

# Get the Centroid of these offices
cent_open_offices <- gCentroid(i_data_open_offices, byid = T,  id = open_offices)
```

Now, for each closed offices (using its centroid), we are going to get the near open offices:

```
near_offices = list()

for (closed in closed_offices)
{
  # Get information of the closed office
  i_data_closed_office <- i_data_gs[i_data_gs@data$name == closed, ]

  # Get the Centroid of that office
  centroid <- gCentroid(i_data_closed_office)
```

```
  # Get the buffer with 5km from each closed office
  i_data_closed_offices_buffer <- gBuffer(spgeom = centroid, width = 5000)

  # Get the open offices inside the buffer
  cent_open_offices_inside_buffer <- cent_open_offices[i_data_closed_offices_buffer, ]

  # Plot the completed graph
  plot(i_data_gs, col = "lightgrey")
  plot(i_data_closed_offices, add = T, col = "orange")
  plot(i_data_closed_offices_buffer, add = T)
  points(centroid, cex = 0.5, col = "black", pch = 19)
  points(cent_open_offices_inside_buffer, cex = 0.5, col = "black", pch = 19)
  legend("topleft", legend = closed)

  # Storage the open offices near to closed office
  near_offices[[closed]] <- rownames(cent_open_offices_inside_buffer@coords)
}
```


Wandsworth



With that, we already have the offices near to each closed office:

```
near_offices
```

```
## $`City of London`
## [1] "Bexley"                  "Kingston upon Thames"
## [3] "Brent"                   "Hammersmith and Fulham"
## [5] "Hounslow"
##
## $Wandsworth
## [1] "Redbridge"   "Westminster"
```

The offices which are open near to the closed offices are:

- City of London:
  - Bexley
  - Kingston upon Thames
  - Brent
  - Hammersmith and Fulham
  - Hounslow
- Wandsworth:
  - Redbridge
  - Westminster
- Ealing:
  - None