Almost Famous: Analyse campaign query combinations

Cindy Lamm

12:35, Wednesday 21st January, 2015

Load variable names and types:

```
nameTypeDataFile  <- "../../data/raw_variables.csv"
variableNames <- read.csv(nameTypeDataFile, header=TRUE, stringsAsFactors=FALSE)
variableNames

##          name      type
## 1    visit_id    factor
## 2         uid    factor
## 3    campaign    factor
## 4      tstamp character
## 5 experiments    factor
## 6      action    factor
## 7       query    factor

factorIdx <- which(variableNames$type=="factor")
factorNames <- variableNames$name[factorIdx]
```

Read the per visit aggregated web log data:

```
visitFile <- "../../data/web_visits.csv"
visitData <- read.csv(visitFile, stringsAsFactors=FALSE, col.names=variableNames$name,
                      colClasses=variableNames$type, na.strings=c("NA",""))
visitData$tstamp <- as.POSIXct(visitData$tstamp)
str(visitData)

## 'data.frame': 1482602 obs. of  7 variables:
##  $ visit_id   : Factor w/ 1482602 levels "10000024498",..: 1252062 128641 583195 349394 830165 690964
##  $ uid        : Factor w/ 1064214 levels "100000493","100000682",..: 858988 92339 95584 929716 656934
##  $ campaign   : Factor w/ 10 levels "103","127","14",..: 7 1 7 1 4 8 1 1 1 2 ...
##  $ tstamp     : POSIXct, format: "2014-09-18 05:43:18" "2014-09-16 21:24:08" ...
##  $ experiments: Factor w/ 4 levels "[1 3]","[1 4]",..: 2 1 4 1 3 2 1 3 2 1 ...
##  $ action     : Factor w/ 8 levels "[landed adclick adclick adclick]",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ query      : Factor w/ 5 levels "advanced analytics",..: 1 5 1 5 3 5 5 5 5 4 ...
```

```
summary(visitData)

##         visit_id              uid              campaign          tstamp
##  10000024498:      1   102486699:      7   558     :324872   Min.   :2014-09-15 00:00:01
##  10000032484:      1   123618732:      7   103     :324027   1st Qu.:2014-09-18 16:32:04
##  10000079220:      1   143588980:      7   59      :232002   Median :2014-09-22 16:55:36
##  10000092303:      1   159226004:      7   31      :231685   Mean   :2014-09-22 20:33:11
##  10000132469:      1   168873739:      7   127     : 92681   3rd Qu.:2014-09-26 19:41:15
##  10000206890:      1   171898393:      7   94      : 92436   Max.   :2014-09-30 23:53:20
##  (Other)    :1482596   (Other)  :1482560   (Other):184899
##  experiments                                   action
##  [1 3]:370018   landed                            :1291256
##  [1 4]:371852   [landed signup]                   :  84889
##  [2 3]:370082   [landed order]                    :  43930
##  [2 4]:370650   [landed adclick]                  :  28233
##                 [landed adclick adclick adclick]:  14956
##                 [landed adclick adclick]          :  14875
##                 (Other)                           :   4463
##                         query
##  advanced analytics         :463687
##  building predictive models: 92454
##  data science               : 92445
##  data science training      :185117
##  predictive modeling        :648899
##
##
```

What are the actions per visit??

```
table(visitData$action)

##
## [landed adclick adclick adclick]          [landed adclick adclick]
##                             14956                             14875
##                  [landed adclick]                    [landed order]
##                             28233                             43930
##           [landed signup adclick]             [landed signup order]
##                              1045                              3418
##                   [landed signup]                            landed
##                             84889                           1291256
```

Look at visits with orders:

```
orderLevels <- names(unlist(sapply(levels(visitData$action), FUN=grep, pattern="order")))
orderLevels

## [1] "[landed order]"        "[landed signup order]"

isOrderIdx <- sort(union(which(visitData$action == orderLevels[1]),
                      which(visitData$action == orderLevels[2])))
totalOrders <- length(isOrderIdx)
```

I conclude from the factor levels for `action` that there is at most 1 order per visit and overall 47348 orders. I cross check with a simple grep on the command line on the unaggregated web data which gives us the same result:

```
$ grep -o order web.log | wc -l
$ 47348
```

Add the number of orders per visit as variable to the data frame:

```
nbOrder <- rep(0, nrow(visitData))
nbOrder[isOrderIdx] <- 1
visitData$nb_orders <- nbOrder
```

```
prop.table(table(visitData$nb_orders))

##
##          0          1
## 0.96806425 0.03193575
```

There are 96.8064255% of visits that don't have an order and only 3.1935745% that do.

How many orders are there per campaign-query combination?

```
combinations <- expand.grid(queries=levels(visitData$query), campaigns=levels(visitData$campaign))
length(combinations)

## [1] 2

webAggCampaignQuery <- aggregatePerCQ(visitData)
o <- order(webAggCampaignQuery$mean_orders, decreasing=TRUE)
webAggCampaignQuery[o,]

##    campaign                      query nb_visits nb_uids total_orders mean_orders
## 3        14 building predictive models     46252   45738         3065  0.06626740
## 2       127      data science training     92681   90761         5101  0.05503825
## 10       94      data science training     92436   90510         5071  0.05485958
## 5       203 building predictive models     46202   45711         2511  0.05434830
## 8       558        predictive modeling    324872  301174        10628  0.03271442
## 1       103        predictive modeling    324027  300394        10425  0.03217324
## 4        17               data science     46308   45814         1052  0.02271746
## 9        59         advanced analytics    232002  219833         4396  0.01894811
## 6        23               data science     46137   45630          858  0.01859679
## 7        31         advanced analytics    231685  219610         4241  0.01830503
##    sd_orders
## 3  0.2487516
## 2  0.2280561
## 10 0.2277072
## 5  0.2267061
## 8  0.1778884
## 1  0.1764603
## 4  0.1490029
## 9  0.1363421
## 6  0.1350975
## 7  0.1340523
```

Write the result into csv file:

```r
write.csv(webAggCampaignQuery, file="../q3_campaign_query_combi/webAggCampaignQuery.csv",
          row.names=FALSE, quote=FALSE)
```

I use a Python script to put the result in the required json format (because even if I would use the package jsonlite to format the result in R, the `sink()` method, which I would need to write it to the file system, does not work in combination with knitr.)