Almost Famous: Analyse Newsletter Signup Rate Per Experiment

Cindy Lamm

17:59, 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
##
##
```

1 Newsletter Signup Rate Per Experiment

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 signups:

```
signupIdx <- getPatternIndex(visitData$action, "signup")

## Concerned pattern levels are [landed signup adclick], [landed signup order], [landed signup]

totalSignups <- length(signupIdx)
```

I conclude from the factor levels for `action` that there is at most 1 signup per visit and overall 89352 signups. I cross check with a simple grep on the command line on the unaggregated web data which gives us the same result:

```
$ grep -o signup web.log | wc -l
$ 89352
```

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

```
nbSignup <- rep(0, nrow(visitData))
nbSignup[signupIdx] <- 1
visitData$nb_signups <- nbSignup
```

There are 93.97% of visits that don't have a signup and only 6.03% that do.
Checkout experiment information:

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

##
##     [1 3]     [1 4]     [2 3]     [2 4]
## 0.2495734 0.2508104 0.2496166 0.2499997
```

Split up the experiment information into separate variables

```
expIdx1 <- getPatternIndex(visitData$experiments, 1)

## Concerned pattern levels are [1 3], [1 4]

totalExp1 <- length(expIdx1)
expIdx2 <- getPatternIndex(visitData$experiments, 2)

## Concerned pattern levels are [2 3], [2 4]

totalExp2 <- length(expIdx2)
expIdx3 <- getPatternIndex(visitData$experiments, 3)

## Concerned pattern levels are [1 3], [2 3]

totalExp3 <- length(expIdx3)
expIdx4 <- getPatternIndex(visitData$experiments, 4)

## Concerned pattern levels are [1 4], [2 4]

totalExp4 <- length(expIdx4)
```

and add them pairwise to the data frame:

```
stopifnot(!any(intersect(expIdx1, expIdx2)),
          totalExp1 + totalExp2 == nrow(visitData),
          !any(intersect(expIdx3, expIdx4)),
          totalExp3 + totalExp4 == nrow(visitData))

experiment12 <- rep(1, nrow(visitData))
experiment12[expIdx2] <- 2
```

```
visitData$experiment_12 <- factor(experiment12, levels=1:2)

experiment34 <- rep(3, nrow(visitData))
experiment34[expIdx4] <- 4
visitData$experiment_34 <- factor(experiment34, levels=3:4)
```

Checkout experiment distribution:

```
prop.table(table(visitData$experiment_12))

##
##         1         2
## 0.5003838 0.4996162

prop.table(table(visitData$experiment_34))

##
##         3         4
## 0.4991899 0.5008101
```

How many signups are there per experiment?

```
visitAggExp12 <- aggregatePerExperiment12(visitData)
visitAggExp12

##   experiment_12 nb_visits nb_uids total_signups total_non_signups signup_rate
## 1             1    741870  532225         45145            487080  0.08482315
## 2             2    740732  531989         44207            487782  0.08309758

visitAggExp34 <- aggregatePerExperiment34(visitData)
visitAggExp34

##   experiment_34 nb_visits nb_uids total_signups total_non_signups signup_rate
## 1             3    740100  531345         46819            484526  0.08811413
## 2             4    742502  532869         42533            490336  0.07981887
```

Write the result into json file:

```
library(jsonlite)
overallSignupRates <- c(visitAggExp12$signup_rate, visitAggExp34$signup_rate)
names(overallSignupRates) <- paste("experiment", 1:4, sep="")
jsonString <- toJSON(as.data.frame(t(overallSignupRates)), dataframe="rows", pretty=TRUE)
jsonString

## [
##     {
##         "experiment1": 0.0848,
##         "experiment2": 0.0831,
##         "experiment3": 0.0881,
##         "experiment4": 0.0798
##     }
## ]
##
```

```
write(jsonString, file="../q4_newsletter_signup/out/overallSignupRates.json")
```

2 Performance Of Experiments

I assume that the performance of an experiment is measured by the number of signups for the newsletter.

```
gTestTable12 <- cbind(visitAggExp12$total_signups, visitAggExp12$total_non_signups)
rownames(gTestTable12) <- paste("Experiment",1:2,sep="")
colnames(gTestTable12) <- c("signups","ignorations")
gTestTable12

##             signups ignorations
## Experiment1   45145      487080
## Experiment2   44207      487782

gTestTable34 <- cbind(visitAggExp34$total_signups, visitAggExp34$total_non_signups)
rownames(gTestTable34) <- paste("Experiment",3:4,sep="")
colnames(gTestTable34) <- c("signups","ignorations")
gTestTable34

##             signups ignorations
## Experiment3   46819      484526
## Experiment4   42533      490336
```

Run G-Test, which is a test of independence (just like the Chi-Square test). The null hypothesis is that the two variables are independent, which is rejected if the p-value is smaller than a chosen significance level α. Usually one uses $\alpha = 5\%$ but with such a large sample size I would opt for $\alpha = 0.001$, which means that if we decide to reject the null hypothesis and assume the variables are dependent (i.e. in this case this means one experiment is better than the other), we might be wrong for $\alpha \times n = 1064.214$ observations.

```
library(Deducer)

## Loading required package:  ggplot2
## Loading required package:  JGR
## Loading required package:  rJava
## Loading required package:  JavaGD
## Loading required package:  iplots
## Note:  On Mac OS X we strongly recommend using iplots from within JGR.
## Proceed at your own risk as iplots cannot resolve potential ev.loop deadlocks.
## 'Yes' is assumed for all dialogs as they cannot be shown without a deadlock,
## also ievent.wait() is disabled.
## More recent OS X version do not allow signle-threaded GUIs and will fail.
##
##
## Please type JGR() to launch console.  Platform specific launchers (.exe and .app) can also
be obtained at http://www.rforge.net/JGR/files/.
##
##
## Loading required package:  car
```

```
## Loading required package:  MASS
##
##
## Note Non-JGR console detected:
## Deducer is best used from within JGR (http://jgr.markushelbig.org/).
## To Bring up GUI dialogs, type deducer().
##
##
## Attaching package:  'Deducer'
##
## The following object is masked from 'package:stats':
##
##    summary.lm
```

```
test12 <- likelihood.test(gTestTable12)
test12
```

```
test34 <- likelihood.test(gTestTable34)
test34
```

I would interpret the results as follows: Experiments 1 and 2 are indistinguishable based on the number of signups - it doesn't matter whether the color scheme of the page is blue or green.

However there is a statistically significant difference between Experiment 3 and 4 - based on the number of newsletter signups the promotional blurb by Josh Willis performs better.