Causal impact study

GB

February 5, 2017

1. Getting data file and setting experiment start, end and rollout times. Those times below can be changed.

```
#myfile<-file.choose()
myfile<-file('client_04-01-2015--11-21-2015.csv', 'r')
startime<-as.Date('2016-04-01')
endtime<-as.Date('2016-06-01')
rolout<-as.Date('2016-09-01')
mydata<-read.csv(myfile, header = TRUE)
close(myfile,'r')
```

checking if there is data prior to experiment- answer into variable data_for_strata

```
mydata$ga.date<-as.Date(mydata$ga.date)
sprintf('start date in file: %s', min(mydata$ga.date))
```

```
## [1] "start date in file: 2015-04-01"
```

```
sprintf('end date in file: %s', max(mydata$ga.date))
```

```
## [1] "end date in file: 2016-11-21"
```

```
if (min(mydata$ga.date) >= startime) { print('Warning: no data prior to experiment start!')
  data_for_strata<-FALSE} else {data_for_strata<-TRUE}
data_for_strata
```

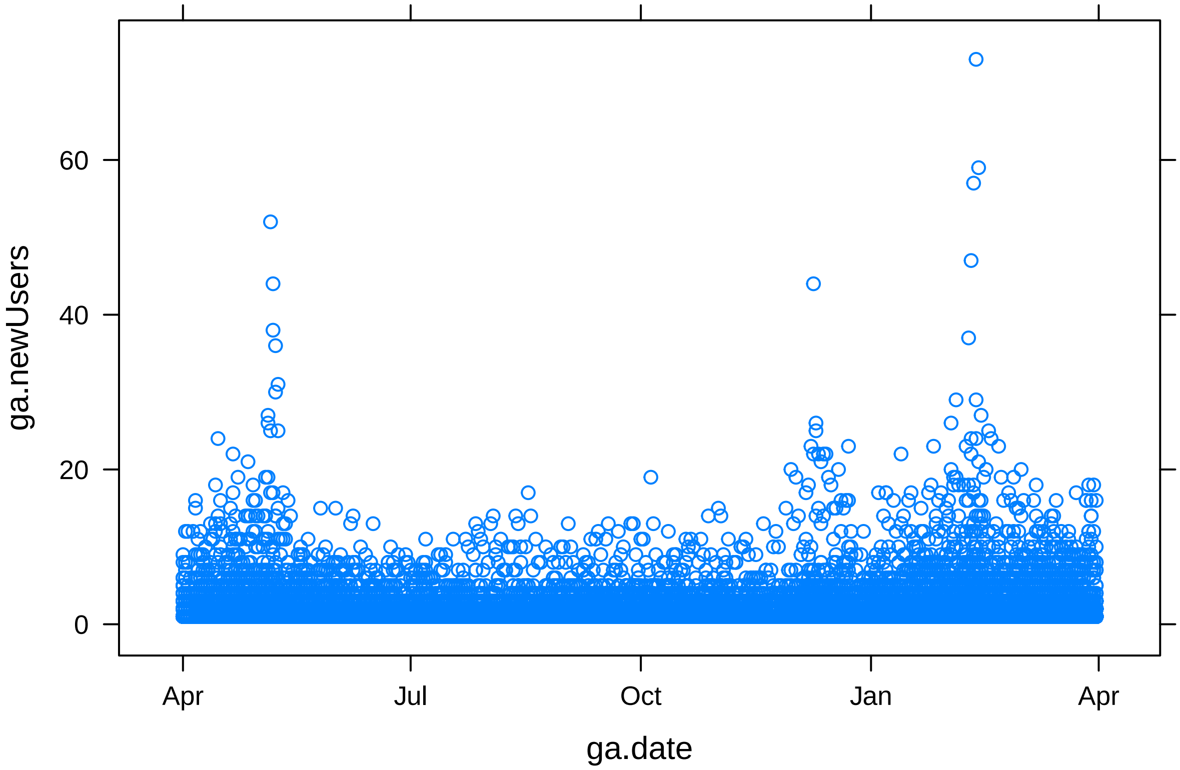
```
## [1] TRUE
```

```
library(CausalImpact)
library(dplyr)
library(lattice)
```

2. Getting to know more about the loaded datafile.

Selecting only data from time period before the experiment start as described in Etsy experiment, and plotting it. After some trials I noticed that largest counts were in the empty landingPagePath. Not sure if that was real path, eliminated it from stratas.

```
eliminate_empty_path=TRUE
if (eliminate_empty_path) {mydata<-mydata[!mydata$ga.landingPagePath==mydata$ga.landingPagePath[1],]}
prior_to_exp<-mydata[mydata$ga.date < startime,]
xyplot(ga.newUsers~ga.date, prior_to_exp)
```



Checking how many unique paths are in landingPagePath column

```
paths<-unique(mydata$ga.landingPagePath)   # selects unique landingPagePath's for the entire data file
sprintf('total different landing pages in the entire data file: %s', length(paths))
```

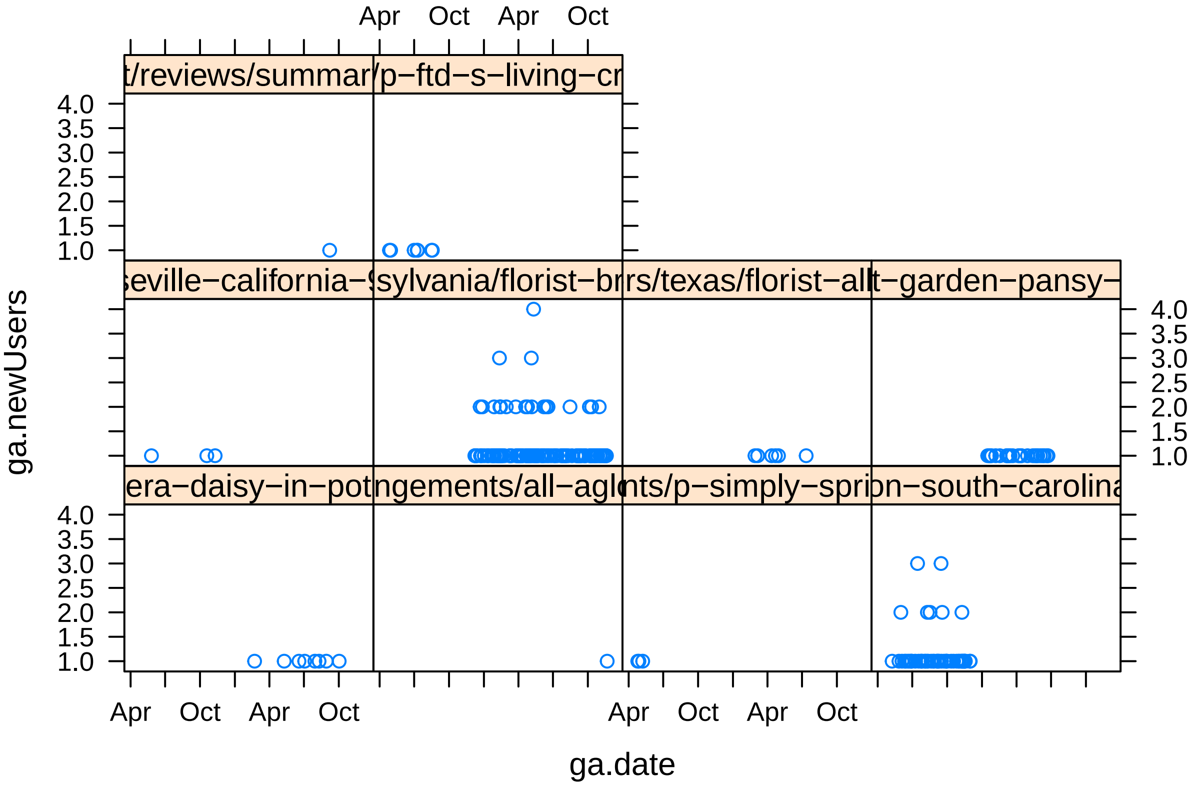
```
## [1] "total different landing pages in the entire data file: 7225"
```

```
if (data_for_strata) {paths2<-unique(prior_to_exp$ga.landingPagePath) #selects unique landingPagePath's
sprintf('total different landing pages prior to experiment: %s', length(paths2))}
```

```
## [1] "total different landing pages prior to experiment: 6102"
```

ploting some times series for the random NP=10 landing Pages, the number can be changed below, NP variable, user selectable

```
NP<-10
testpaths<-sample(length(paths), NP)
data2<-mydata[mydata$ga.landingPagePath %in% paths[testpaths], ]
xyplot(ga.newUsers~ga.date|ga.landingPagePath, data2)
```

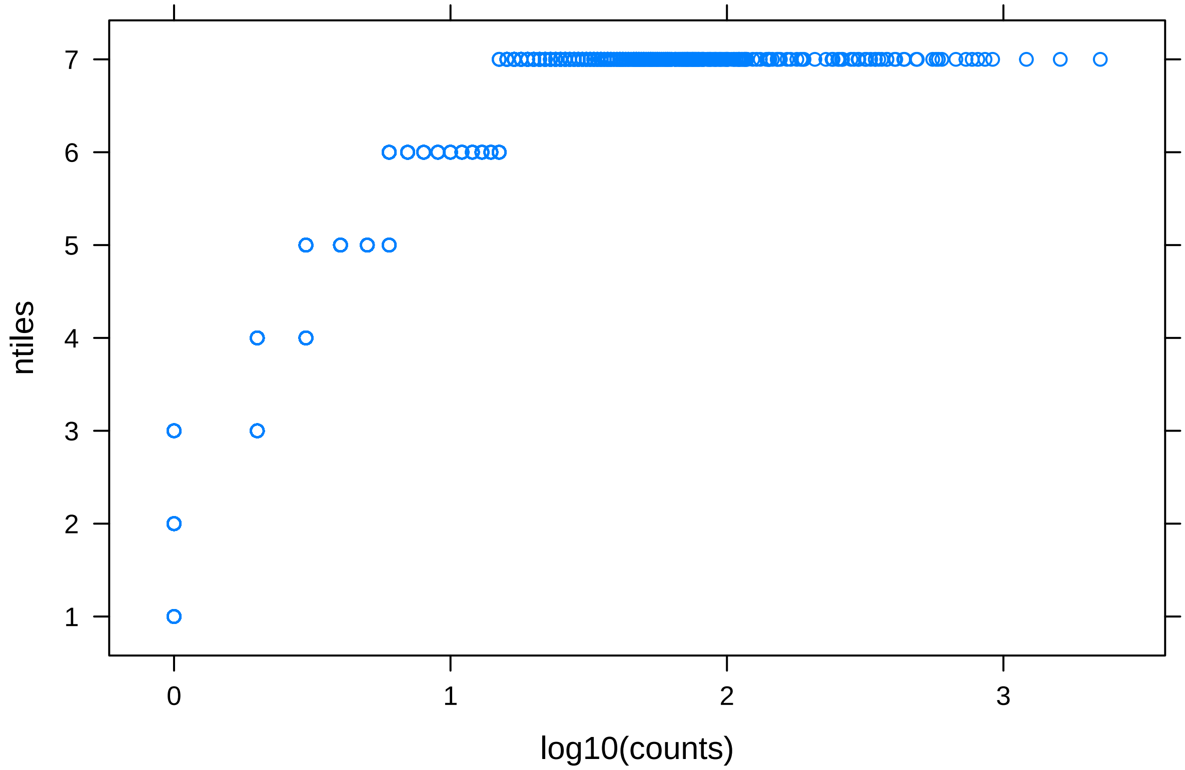


```r
rm(data2)
```

3. Building stratas

Generating ntiles and plotting ntiles versus counts. counts here are total NewUsers per page. Number of ntiles is kn=10. can be changed below, user selectable.

```r
kn=7 # numer of ntiles, can be changed
sumdata<-group_by(prior_to_exp, ga.landingPagePath) %>% summarise(counts=sum(ga.newUsers)) # getting
sumdata<-sumdata[order(-sumdata$counts), ]    # ordering in descreasing order
sumdata$ntiles<-ntile(sumdata$counts, kn)  # creating ntiles column and assinging ntile number to eac
xyplot(ntiles~log10(counts), sumdata)
```



repackaging data into different kg=7 groups. Number kg can be changed. number of unique landingPagePath's per ntile is printed out

```
kg=5  # kg = to number of test groups

  sumdata$group<-0   #creating group marker column
  groupeddata<-data.frame()  # new data frame where results will be
  for (i in 1:kn) {   # looping through ntiles
    sampledsumdata<-filter(sumdata, ntiles==i)   # selecting data for specific ntile only
    set.seed(12345)      #
    sampledsumdata2<-sampledsumdata[sample(nrow(sampledsumdata)),]    # reshufling rows randomly in tha
    print(length(sampledsumdata$ga.landingPagePath)) # prints the number of paths per ntile
    sampledsumdata2$group<-rep_len(1:kg, length(sampledsumdata2$counts))    # assign group to randomize
    groupeddata<-rbind(groupeddata, sampledsumdata2) # combine into final table
    rm(sampledsumdata, sampledsumdata2) # remove extra variables
  }
```

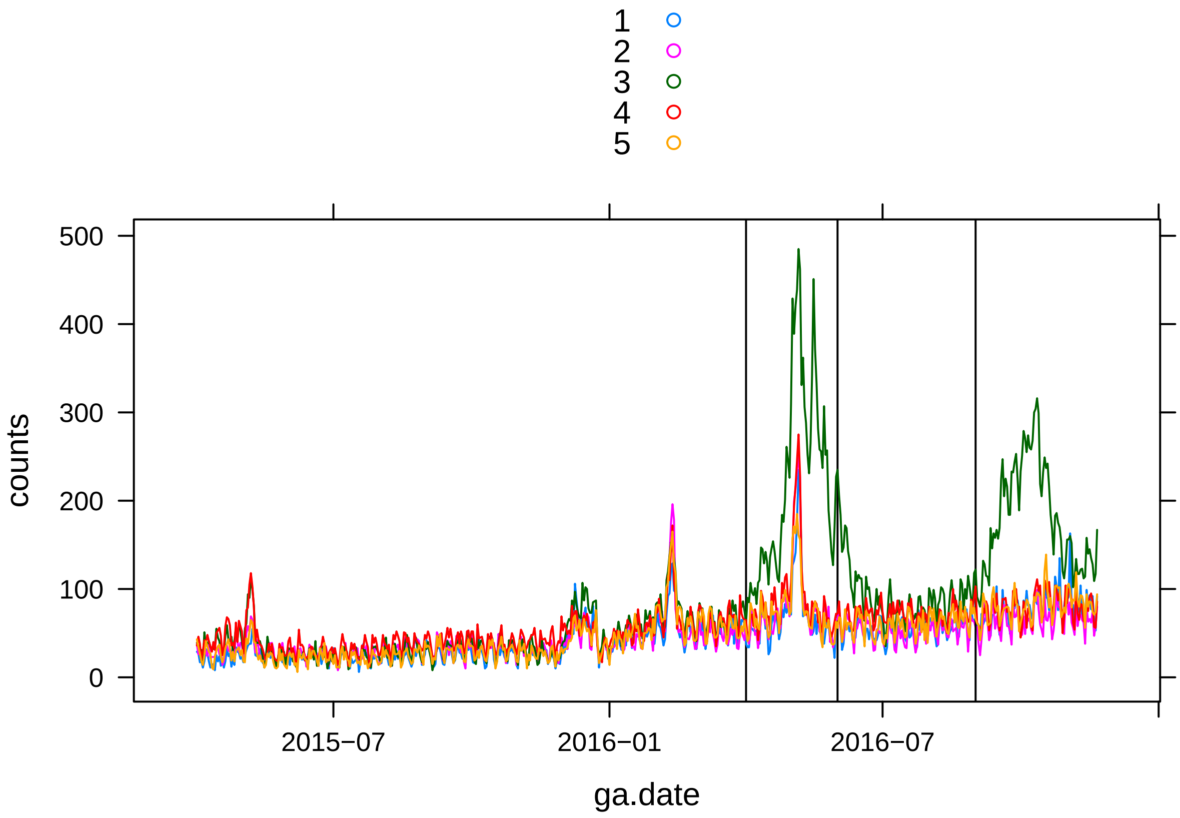
```
## [1] 872
## [1] 872
## [1] 872
## [1] 871
## [1] 872
## [1] 872
## [1] 871
```

here groupeddata contains landingPagePath, total counts per that path, ntile and group for that landingPagePath. this table is used to map landingPagePath to a group in the original data file

```
x<-groupeddata[match(mydata$ga.landingPagePath,groupeddata$ga.landingPagePath),4,drop=F]
mydata$group<-as.factor(x$group)
rm(x)
```

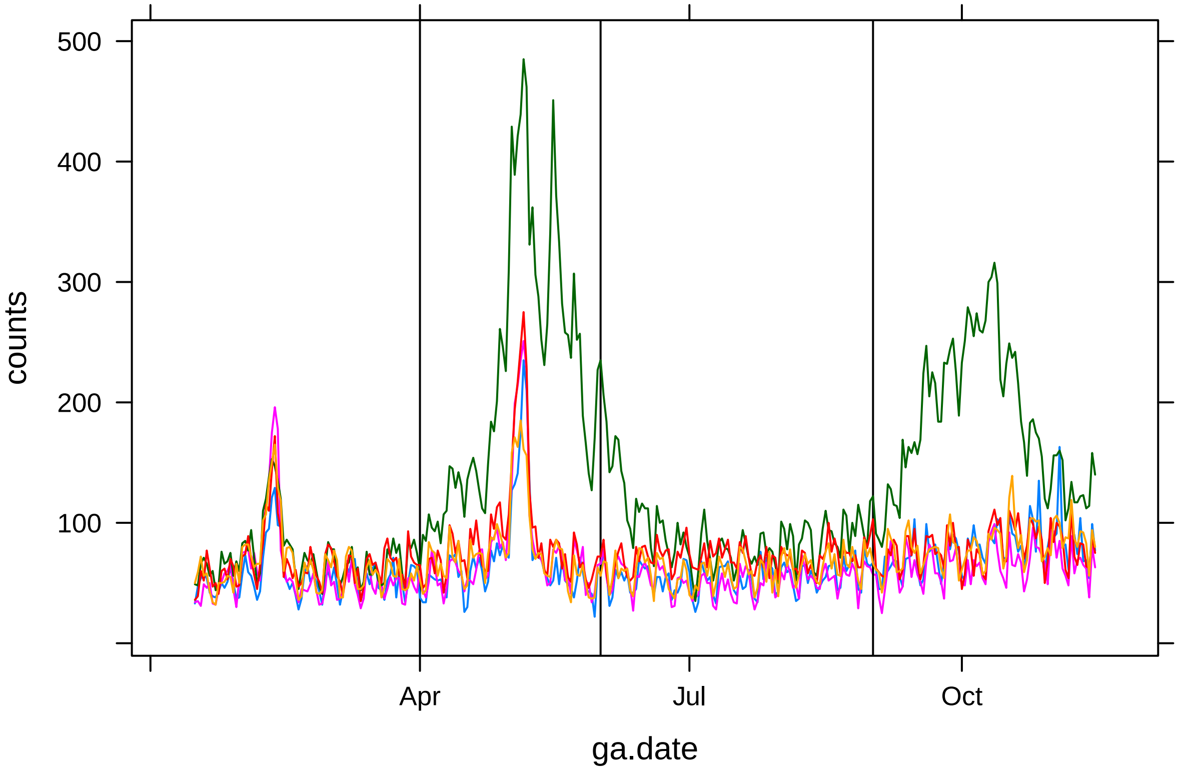
there were some landingPagePath values during experiment that were not existant prior to experiment. Therefore, those extra landingPagePath's do not get assigned a group.

```
mydata_na<-mydata[!is.na(mydata$group),]    #removes NA, puts result into mydata_na
my.lines<-c(startime, endtime, rolout) # marks vertical lines in plot
groupmydata<-group_by(mydata_na, group, ga.date) %>% summarise(counts=sum(ga.newUsers)) # agregates per
xyplot(counts~ga.date, groups=group, groupmydata, type='l',auto.key=T, panel = panel.superpose,
       panel.groups = function(..., group.number) {
           panel.abline(v = my.lines[group.number])
           panel.xyplot(...)})
```



plotting for specific time window, st and en define time period, values can be changed below:

```
st<-as.Date('2016-01-15')
en<-as.Date('2016-11-15')
groupmydata2<-groupmydata[groupmydata$ga.date<=en & groupmydata$ga.date>st, ]
xyplot(counts~ga.date, groups=group, groupmydata2, type='l', panel = panel.superpose,
       panel.groups = function(..., group.number) {
           panel.abline(v = my.lines[group.number])
           panel.xyplot(...) })
```



this text only for kg=5, kn=7 values and the above data file. For different values colors shift. During the experiment (between first two vertical lines) and after the rolout (the last vertical line) group Nr3 (green line) increased in values more than other groups. Prior to the experiment group Nr 3 was very similar to other groups see also the previous figure. It takes some time to recover in post-experiment. The same recovery delay was observed in Thumbtack report.