Stat243: Problem Set 6

Linqing Wei

November 1, 2017

1 Problem 1

1. The goal of their simulation study is to propose a test procedure of testing the hypothesis of the number of components in a normal mixture. The questions they try to answer are: how fast the test statistic converge in distribution to asymptotic distribution, and what is the power?

The metrics they consider are the simulated significance level of unadjusted/adjusted test and simulated power of unadjusted/adjusted test. They set parameters u1 = 0, theta = 1 and the mixing proportion is pi (height of different gaussians) also the number of components. The constants are variance and nominal level.

2.The authors have to decide the simulation sample size, what correction to be applied for the adjusted test, through which algorithm to obtain the maximum likelihood estimates, from which distribution to draw the random sample, etc.

The number of drawn samples, the number of configurations, the nominal level, the values of mixing proportions and the chosen distance between two components might affect the statistical power of the test.

They set sigma to 1 without any further comments. They never mentioned the standard error out of the total number of simulations. Gaussian is not the only available option of generating the data but they chose gaussian without any further explanations.

3. Their tables provide sufficient information but there is no highlights marked in the table. It is not straightforward for readers to see the inference from the table directly.

They could mark the highlights in the table or maybe use plots to represent the data.

4. The results make sense. Interpretation: The initial step was to go from n samples to 2LR and then compared to asymptotic dist. For example, in 1000 repetitions, 0.009 means 9 rejected out of 1000 simulations. And the goal of each test is to ideally approach the nominal level they set. Based on the tables, the data are not approaching nominal level quickly.Table 2 and 4 represent samples drawn from alternative distribution. 1.8 means 18/1000 reject rate. When D = 1, we can tell the resolution is not good. Therefore, even with large sample size, we cannot reject H1.

5. A general principle is that changing the number of repetitions wont change the convergence rate but will change the precision. When the simulating results seem reasonable, steady and follow the results in previous papers, then the authors decided to use 1000 simulations. 10 simulations might not be enough since a small sample size would give a worse resolution in terms of power test. To decide whether or not 1000 samples are enough, we could look at the power test value. If it's close enough to the expected result, then it means 1000 samples are reasonablely good. Ohterwise, we would increase the number of simulations.

2 Problem 2

```
library(RSQLite)
```

```
## Loading required package:  DBI
## Warning:  package 'DBI' was built under R version 3.2.5

drv <- dbDriver("SQLite")
dir <- '/Users/weilinqing/Desktop'
dbFilename <- 'stackoverflow-2016.db'
db <- dbConnect(drv, dbname = file.path(dir, dbFilename))

result_users <- dbGetQuery(db, "select U.userid from users U
join questions Q on U.userid = Q.ownerid
join questions_tags T on Q.questionid = T. questionid
where T.tag = 'r' and userid not in
(select userid from users U2
join questions Q2 on U2.userid = Q2.ownerid
join questions_tags T2 on Q2.questionid = T2. questionid
where T2.tag = 'python')")

nrow(result_users)

## [1] 39195

head(result_users)

##   userid
## 1    740
## 2   1428
## 3   6722
## 4   9435
## 5   9435
## 6  17216

result_distinct <- dbGetQuery(db, "select distinct U.userid from users U
join questions Q on U.userid = Q.ownerid
join questions_tags T on Q.questionid = T. questionid
where T.tag = 'r' and userid not in
(select userid from users U2
join questions Q2 on U2.userid = Q2.ownerid
join questions_tags T2 on Q2.questionid = T2. questionid
where T2.tag = 'python')")

nrow(result_distinct)

## [1] 18611

head(result_distinct)

##    userid
## 1 4696169
## 2  575952
## 3 1513168
## 4 5733031
## 5 4887832
## 6 4875428
```

3 Problem 3

```
dir = '/global/scratch/paciorek/wikistats_full'
lines = sc.textFile(dir + '/' + 'dated')
import re
from operator import add

#The goal of this fetch function is to find the information associated with "Christmas"
def fetch(line, regex = "Christmas", language = None):
  vals = line.split(' ')
  if len(vals) < 6:
      return(False)
  temp = re.search(regex, vals[3])
  if temp is None or (language != None and vals[2] != language):
    return(False)
  else:
    return(True)
obs = lines.filter(fetch).repartition(480)
def stratify(line):
    vals = line.split(' ')
    return(vals[0] + '-' + vals[1] + '-' + vals[2], int(vals[4]))
counts = obs.map(stratify).reduceByKey(add)
def transform(vals):
    key = vals[0].split('-')
    return(",".join((key[0], key[1], key[2], str(vals[1]))))
mydir = '/global/home/users/linqing_wei'
outputDir = mydir + '/' + 'Christmas-counts'
counts.map(transform).repartition(1).saveAsTextFile(outputDir)
```

3.1 P3 Analysis

```
p3 = readLines("p3_output")
#Trim the trailing spaces in the output file
p3 = trimws(p3, which = c("both", "left", "right"))
p3 = lapply(p3, function(x) strsplit(x, ","))
p3 = lapply(p3, function(x) unlist(x))
#Make a dataframe from lists
Christmas_df = setNames(do.call(rbind.data.frame, p3),c("date", "time", "language", "hits"))
#Standardize the format of date, hits, and language
Christmas_df$date =  as.Date(as.character(levels(Christmas_df$date)), "%Y%m%d")[Christmas_df$date]
Christmas_df$hits = as.numeric(levels(Christmas_df$hits))[Christmas_df$hits]
Christmas_df$language = as.character(levels(Christmas_df$language))[Christmas_df$language]

##Month vs hits analysis
library(lattice)
library(gridExtra)
index <- sprintf('%0.2d', 10:12)
vec = c()
#Subset the hits data by month and then compute the total hits
```

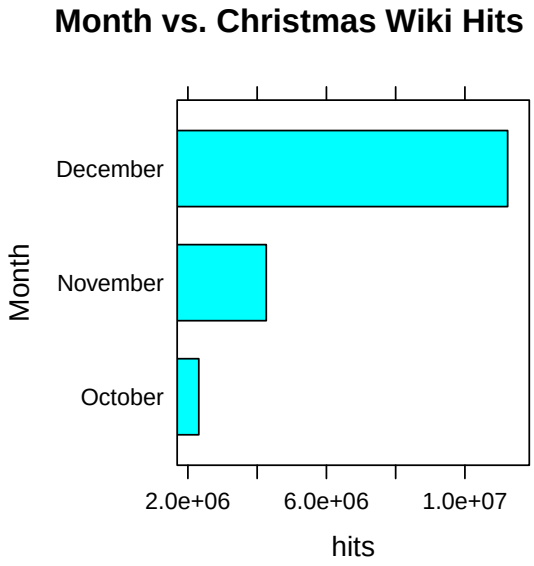
```r
for (i in index){
    m = subset(Christmas_df, format.Date(Christmas_df$date, "%m")==i)
    m_hits = sum(m$hits)
    vec = c(vec, m_hits)
}
#y contains 3 numeric values, each represents total hits for each month.
y = vec
y
```

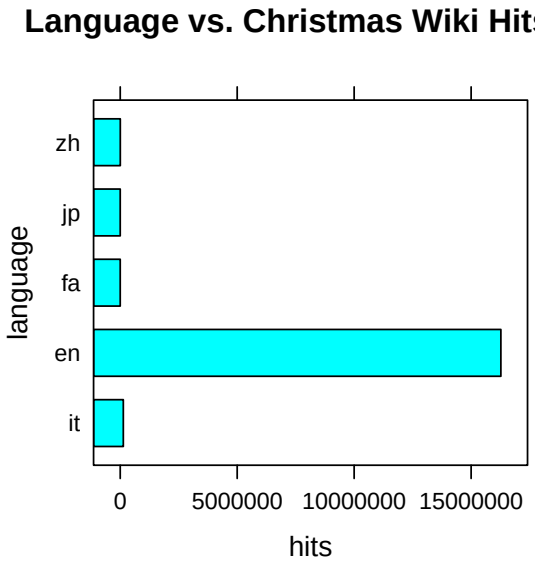
```
## [1]  2314515  4263079 11234924
```

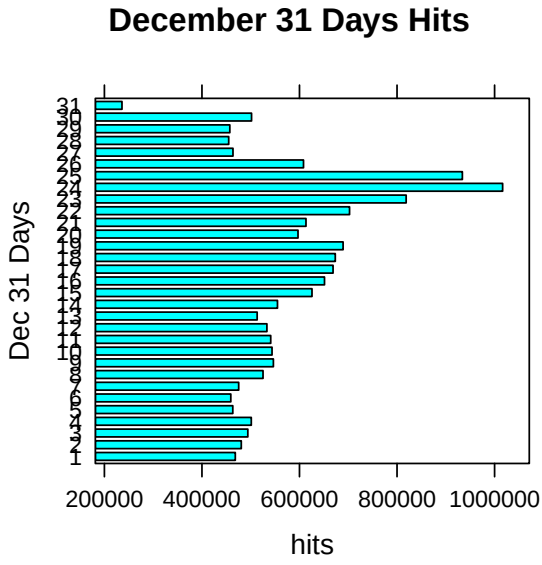
```r
names(y) = c("October", "November", "December")
plot1= barchart(y,ylab="Month", xlab = "hits", main="Month vs. Christmas Wiki Hits")


##Language vs hits analysis
lan_index = c("it", "en","fa", "jp", "zh")
lan_vec = c()
for (i in lan_index){
  lan = subset(Christmas_df, grepl(i,Christmas_df$language))
  lan_hits = sum(lan$hits)
  lan_vec = c(lan_vec, lan_hits)
}
#y2 contains the total number of hits for each language
y2 = lan_vec
names(y2) = lan_index
plot2= barchart(y2, ylab="language", xlab = "hits",main="Language vs. Christmas Wiki Hits")

## December 31 Days vs. Hits analysis
day_index <- sprintf('%0.2d', 1:31)
day_vec = c()
for (i in day_index){
  d = subset(Christmas_df, format.Date(Christmas_df$date, "%d")==i)
  d_hits = sum(d$hits)
  day_vec = c(day_vec, d_hits)
}
#y3 contains the total number of hits for each day in December
y3 = day_vec
names(y3) = c(1:31)
plot3= barchart(y3,ylab="Dec 31 Days", xlab = "hits", main="December 31 Days Hits")
grid.arrange(plot1, plot2, plot3, ncol=2)
```







Based on plot1, very reasonablly, in December people searched for "Christmas" way more than other months. Based on plot2, people who speak English searched Christmas more than other language speakers. Interestingly, asian countries, people who speak "zh" Chinese and "jp" Japanese also searched for Christams a lot. From plot3, people searched for "Christmas" with increasing hits approaching December 24. And it reached the peak on Dec 24. Interestingly, On Dec 30, the hits increased again. Perhaps on that day, people started missing their "almost gone" holiday so they searched again for "Christmas."

4 Problem 4

4.1 Problem 4A

```
library(parallel)
library(doParallel)
library(foreach)
nCores <- as.integer(Sys.getenv("SLURM_CPUS_ON_NODE"))
registerDoParallel(nCores)
#Create index in order to process files by file name.
index <- sprintf('%0.5d', 0:20)
result <- foreach(i = index ,
                  .combine = c,
                  .verbose = TRUE) %dopar% {
                    cat('Starting ', i, 'th job.\n', sep = '')
                    file_name = paste('/global/scratch/paciorek/wikistats_full/dated_for_R/part-',ind
                    dat <- readLines(file_name)
                    locations = grepl("Barack_Obama")
                    output = subset(dat, locations)
                    cat('Finishing ', i, 'th job.\n', sep = '')
                    print(output)
                  }
```

Since I used readLine, I tranformed the output into a dataframe using a bit format processing.

```
options(wideth = 200)
output = readLines("Obama.out")
output = output[grepl("\\[[0-9]{1,}\\].*", output)]
output = trimws(output, which = c("both", "left", "right"))
output = lapply(output, function(x) strsplit(x, " "))
output = lapply(output, function(x) unlist(x)[-1])
result <- setNames(do.call(rbind.data.frame, output), c("date", "time", "language",
    "page_info", "hits", "page_size"))
dim(result)

## [1] 91909     6

names(result)

## [1] "date"      "time"      "language"  "page_info" "hits"      "page_size"

print(result[1:10, ], right = FALSE)

##    date      time   language
## 1  "20081129 210000 pt
## 2  "20081014 190000 en
## 3  "20081108 190000 no
## 4  "20081128 190001 en
## 5  "20081110 160000 et
## 6  "20081101 110000 fr
## 7  "20081205 230000 en
## 8  "20081114 210000 en
## 9  "20081107 020000 de.n
## 10 "20081008 090001 commons.m
##    page_info
## 1  Barack_Obama
```

```
## 2  Special:AllPages/I_ran_Project_Vote_voter_registration_drive_in_Illinois,_ACORN_was_smack_dab_i
## 3  Bilde:Barack_Obama_2004.jpg
## 4  Early_life_and_career_of_Barack_Obama
## 5  Barack_Obama
## 6  Discuter:Barack_Obama
## 7  Image:20081102_Bruce_Springsteen_and_Barack_Obama_hug.JPG
## 8  Barack_Obama_Senior
## 9  Barack_Obama_will_auf_staatliche_Zusch%C3%BCsse_f%C3%BCr_seinen_Wahlkampf_verzichten
## 10 Image:Barack_Obama_and_supporters_5,_February_4,_2008.jpg
##    hits page_size
## 1  86   2032215"
## 2  2    25520"
## 3  1    7825"
## 4  16   760462"
## 5  4    55875"
## 6  1    20922"
## 7  2    18562"
## 8  1    17687"
## 9  1    8656"
## 10 2    11675"
```

4.2 4B

The running time of parallel R is: user 9079.448, system 6041.711, elapsed: 828.184. Approximately 13 min. If running on 4 times of cores, it would take 3 min. Therefore, running parallel R is more effective than running on PySpark.

4.3 4C

It runs faster when the tasks are dynamically allocated than statically allocated. The reason is that prescheduling reduces the communication costs. In this case, prescheduling wont make a large difference since the tasks are relatively equally distributed such that the running time of each file wont differ a lot.