PS6

ShubeiWang

10/22/2018

1

(a)

The goal is to investigate the finite sample performance and statistical properties of the likelihood ratio tests. The metics they use is as follows: the model is normal mixture distributed. They consider two cases where they tested a single normal versus a two-component normal mixture and a two-component normal mixture versus a three-component normal mixture. Then they consider simulated significance level, the rate of convergence and simulated powers and investigate relative properties.

(b)

They needed to decide the sample sizes, significance level, the distance between the two components, the number of components and mixing proportion.

(c)

Yes, because the power increases as distance, sample size or normal level increases. It means that the process is more likely to give correct identification when those factors increases.

(d)

Their tables are quite clear as it presents the simulation results and all the conditions together. An alternative way may be separate each factors so that it's easier to compare results between different levels for one factor.

2

The number of users who have asked Spark-related questions but not Python- related questions is 4647.

```
library(RSQLite)
drv <- dbDriver("SQLite")
dbFilename <- 'stackoverflow-2016.db'
db <- dbConnect(drv,dbFilename)
result <- dbGetQuery(db, "SELECT userid FROM questions Q JOIN users U on
                     U.userid = Q.ownerid JOIN questions_tags T on
                     Q.questionid = T.questionid
                     WHERE tag LIKE '%apache-spark%'
                     EXCEPT
                     SELECT userid FROM questions Q JOIN users U on
                     U.userid = Q.ownerid JOIN questions_tags T on
                     Q.questionid = T.questionid
                     WHERE tag LIKE '%python%'
                    ")
length(unlist(unique(result)))
```

```
## [1] 4647
```

3

I choose to explore the data related to financial crisis in 2008. Firstly I ssh to savio the process the data and then transfer the obtained text file to my local computer to conduct further analysis.

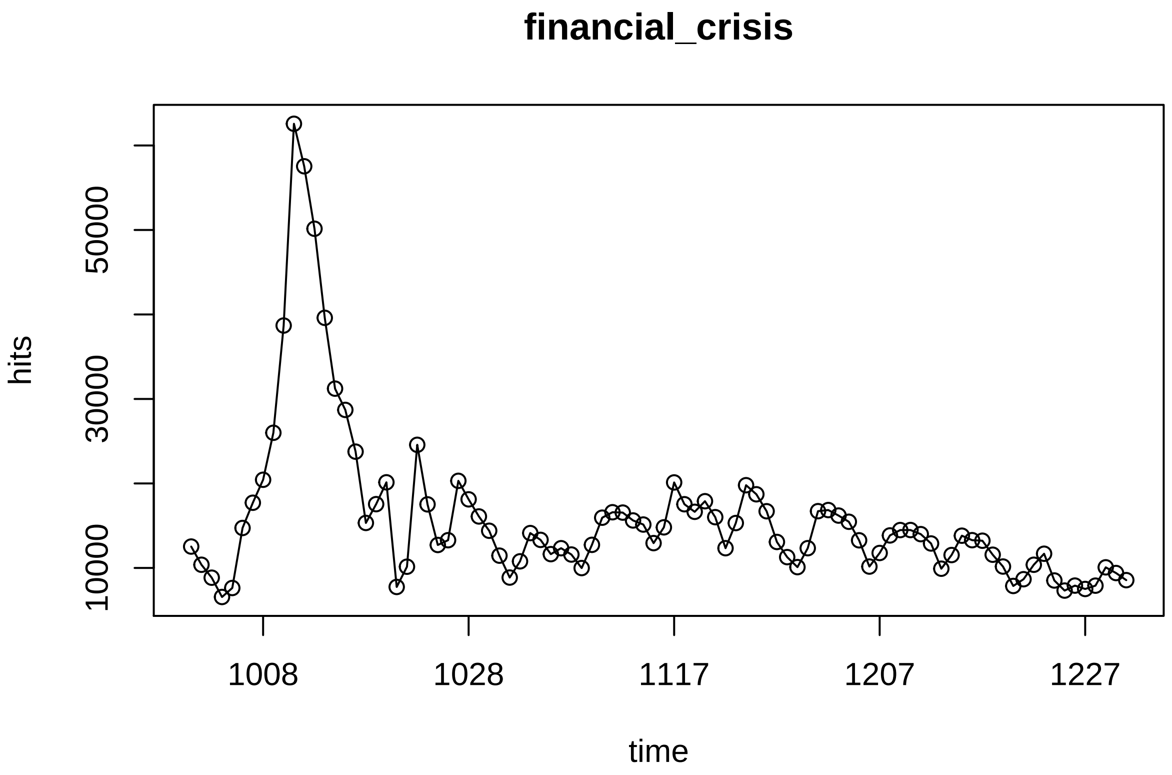
```
## on bash
srun -A ic_stat243 -p savio2 --nodes=4 -t 2:00:00 --pty bash
module load java spark/2.1.0 python/3.5
source /global/home/groups/allhands/bin/spark_helper.sh
spark-start
pyspark --master $SPARK_URL --conf "spark.executorEnv.PYTHONHASHSEED=321"  --executor-memory 60G
```

```
## on pyspark
## load data
dir = '/global/scratch/paciorek/wikistats_full'
lines = sc.textFile(dir + '/' + 'dated')

import re
from operator import add

## find the data related to financial crisis

def find(line, regex = "Financial_crisis", language = None):
    vals = line.split(' ')
    if len(vals) < 6:
        return(False)
    tmp = re.search(regex, vals[3], re.IGNORECASE)
    if tmp is None or (language != None and vals[2] != language):
        return(False)
    else:
        return(True)

fin_crisis = lines.filter(find)


def pair(line):
    # create key-value pairs where:
    #   key = date-language
    #   value = number of website hits
    vals = line.split(' ')
    return(vals[0] + '-' + vals[2], int(vals[4]))

# sum number of hits for each key
counts = fin_crisis.map(pair).reduceByKey(add)

## transform to output
def transform(pair):
    key = pair[0].split('-')
    return(','.join((key[0],key[1],str(pair[1]))))

outputDir = '/global/home/users/shubei_wang/stat243/ps6/' + 'financial_crisis_counts'
counts.map(transform).repartition(1).saveAsTextFile(outputDir)
```

```
## plot the hits against date

library(dplyr)
```

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

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

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

```
library(chron)
data <- read.csv("fin_crisis_count",header = FALSE,col.names = c("date","language","hits"))
data$date <- as.character(data$date)
data <- data %>% filter(data$language == 'en')
data$chron <- chron(data$date, format = c(dates = 'ymd'))
plot(data$chron, data$hits, type = 'l', xlab = 'time', ylab = 'hits', main = "financial_crisis")
points(data$chron, data$hits)
```



4

(a)

```
## on bash
srun -A ic_stat243 -p savio2 --nodes=1 -t 2:00:00 --pty bash
module load r r-packages
R

## use R
library(parallel)
library(doParallel)
library(foreach)
registerDoParallel(as.integer(Sys.getenv("SLURM_CPUS_ON_NODE")))

n <- 959
result <- foreach(i=0:n,
                  .packages = c("readr","stringr","dplyr"),
                  .combine = rbind, # combine the results
                  .verbose = TRUE) %dopar% {
                    filepath <- paste(
                    "/global/scratch/paciorek/wikistats_full/dated_for_R/part-",
                    str_pad(i,width=5,side="left",pad="0"),sep = "")
                    data <- read_delim(filepath,delim=" ",col_names=FALSE)
                    Obama <- data[grep("Barack_Obama",data$X4),]
                    Obama
                  }
dim <- dim(Obama)
write.table(dim, file = "/global/home/users/shubei_wang/result.txt")
write.table(Obama, file = "/global/home/users/shubei_wang/Obama.txt")
```

(b)

The time I used with one node, which has 24 cores, is about 1 hour and 50 minutes. So if I run it with 96 cores it would take about 28 minutes which is much greater than 15 minutes. It shows it's more efficient to use PySpark to do the filtering.