concordance=TRUE

Problem Set 6

Yue Hu

Oct 2017

1 problem 2

Use SQLite execute SQL command. first create two views, one contains userid who asked r questions and the other contains userid who asked python questions. Then left outerjoin these two, so for userid who asked r questions but not python questions, the coloumn python.userid will be Null. then we can select and count these distinct userids.

```
library(RSQLite)
drv <- dbDriver("SQLite")
dir <- './'
dbFilename <- 'stackoverflow-2016.db'
db <- dbConnect(drv, dbname =file.path(dir, dbFilename))

dbGetQuery(db, "create view userR as select distinct U.userid from Users U
           join questions Q on Q.ownerid = U.userid
           join questions_tags T on Q.questionid = T. questionid where T.tag = 'r'")

## Error in rsqlite_send_query(conn@ptr, statement):  table userR already exists

dbGetQuery(db, "create view userPy as select distinct U.userid from Users U
           join questions Q on Q.ownerid = U.userid
           join questions_tags T on Q.questionid = T. questionid where T.tag = 'python'")

## Error in rsqlite_send_query(conn@ptr, statement):  table userPy already exists

dbGetQuery(db, "select count(distinct userR.userid) from userR
           LEFT JOIN userPy on userR.userid = userPy.userid where userPy.userid IS NULL")

##   count(distinct userR.userid)
## 1                        18611

#18611
```

2 problem 3

I asked a question concerning rwo religious holidays in this period. Mowlid(Dec.1st) is a Islamic holiday for the birthday of Mohammad, and Christmas(Dec 25) day is Christan holiday for the birthday of Jesus. They are both important dates for their religion, and I want to know how much attension they receive by counting the numbers of website hit. And I also want to know what group of people is intersted in these two holidays by counting the distribution of language through a pie chart.

I first ran the following code on PySpark in Savio to filter the lines with key works "Mawlid" and "Christmas".

```
dir = '/global/scratch/paciorek/wikistats_full'
lines = sc.textFile(dir + '/' + 'dated')

import re
from operator import add

### filter to sites of interest
def find(line, regex = "Mawlid", language = None):
    vals = line.split(' ')
    if len(vals) < 6:
        return(False)
    tmp = re.search(regex, vals[3])
    if tmp is None or (language != None and vals[2] != language):
        return(False)
    else:
        return(True)


Mawlid = lines.filter(find).repartition(480)
Christmas = lines.filter(lambda x:find(x, regex = "Christmas", language = None)).repartition(480)


### map-reduce step to sum hits across date-time-language triplets
def stratify(line):
    # create key-value pairs where:
    #   key = date-time-language
    #   value = number of website hits
    vals = line.split(' ')
    return(vals[0] + '-' + vals[1] + '-' + vals[2], int(vals[4]))

countsMawlid = Mawlid.map(stratify).reduceByKey(add)
countsChristmas = Christmas.map(stratify).reduceByKey(add)


### map step to prepare output
def transform(vals):
    # split key info back into separate fields
    key = vals[0].split('-')
    return(",".join((key[0], key[1], key[2], str(vals[1]))))


### output to file ###

outputDirMaw = '/global/scratch/yue_hu/MawlidCounts'
countsMawlid.map(transform).repartition(1).saveAsTextFile(outputDirMaw)
outputDirChr = '/global/scratch/yue_hu/MawlidCounts'
countsChristmas.map(transform).repartition(1).saveAsTextFile(outputDirChr)
```

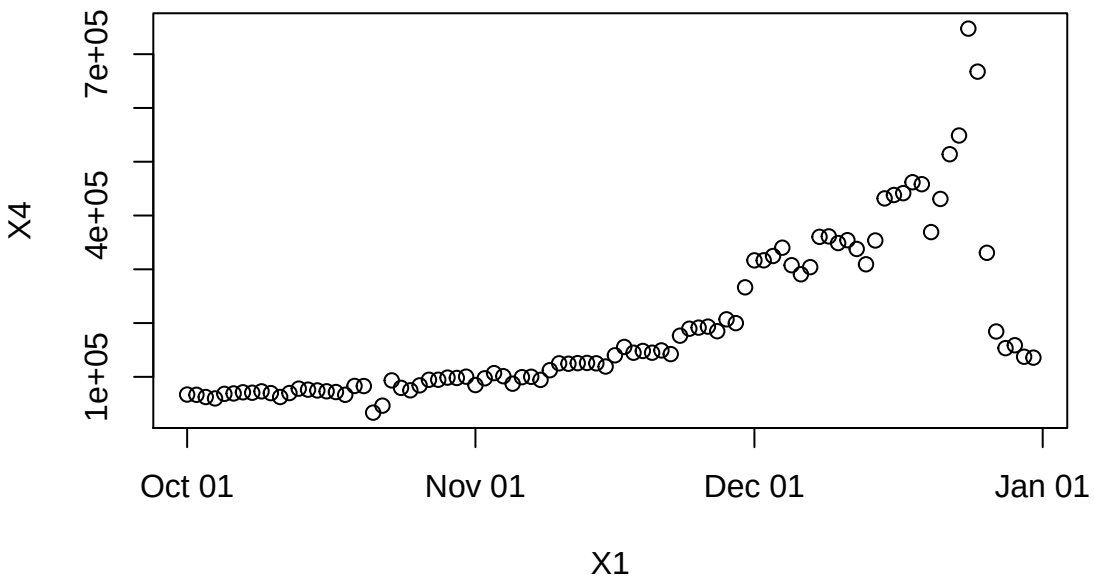
Then I downloaded the file to my local dir through bash shell

```
scp yue_hu@dtn.brc.berkeley.edu:/global/scratch/yue_hu/ChristmasCounts/part-00000 ChrismasCounts

scp yue_hu@dtn.brc.berkeley.edu:/global/scratch/yue_hu/MawlidCounts1/part-00000 MawlidCounts
```

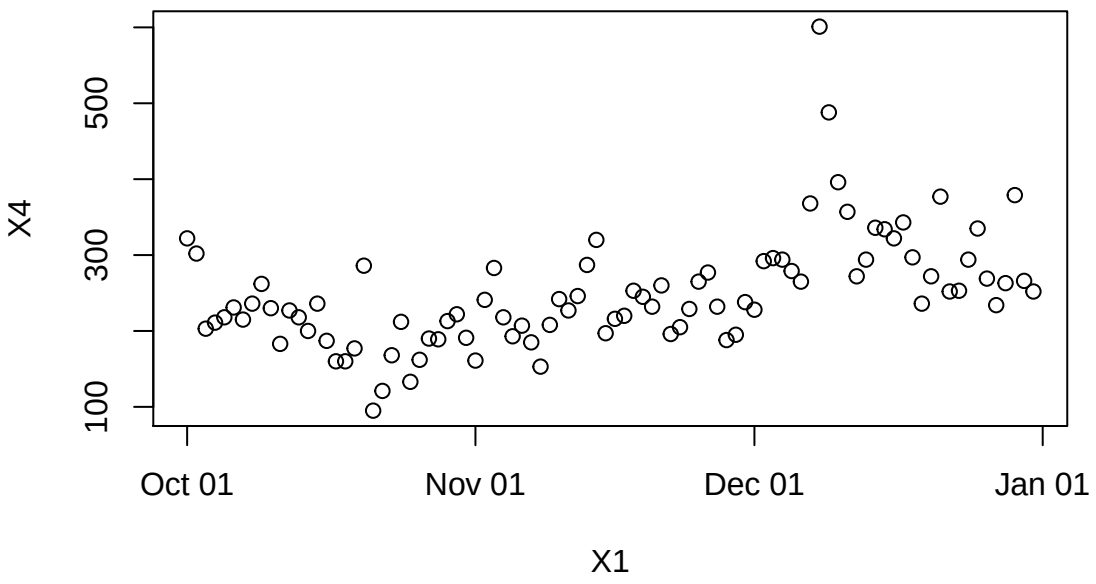
Then I parse it with r

```
library(readr)
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

# read in the dataframe
Mawlid <- readr::read_delim("MawlidCounts",",", col_names = F)

## Parsed with column specification:
## cols(
##  X1 = col_integer(),
##  X2 = col_character(),
##  X3 = col_character(),
##  X4 = col_integer()
## )

Christmas <- readr::read_delim("ChristmasCounts",",", col_names = F)

## Parsed with column specification:
## cols(
##  X1 = col_integer(),
##  X2 = col_character(),
##  X3 = col_character(),
##  X4 = col_integer()
## )

# show some features of the dataframe
head(Mawlid)

## # A tibble: 6 x 4
##         X1     X2    X3    X4
##      <int>  <chr> <chr> <int>
## 1 20081031 090000    nl     3
## 2 20081118 020000    en     7
## 3 20081229 020001    en     4
## 4 20081120 210000    sv     1
## 5 20081012 090000    it     1
## 6 20081226 050000    pt     3

head(Christmas)

## # A tibble: 6 x 4
##         X1     X2    X3    X4
##      <int>  <chr> <chr> <int>
## 1 20081019 040000    te     2
## 2 20081209 180000    hu    11
```

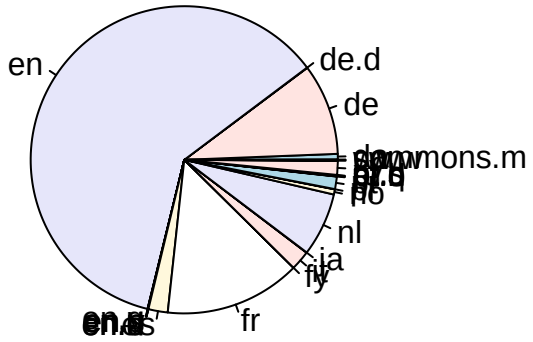
```
## 3 20081027 100000  fr.s     1
## 4 20081115 230001    tl     1
## 5 20081012 010001    zh     1
## 6 20081116 000000   lmo     1

nrow(Mawlid)

## [1] 7295

nrow(Christmas)

## [1] 88625

#group by dates(column X1) and sum the hits(column X4)
SumDateMawlid <- Mawlid %>%
  group_by(X1) %>%
  summarise(X4 = sum(X4))

SumDateChristmas <- Christmas %>%
  group_by(X1) %>%
  summarise(X4 = sum(X4))

# change date to datetime type
SumDateChristmas$X1 <- as.Date(as.character(SumDateChristmas$X1), "%Y%m%d")
SumDateMawlid$X1 <- as.Date(as.character(SumDateMawlid$X1), "%Y%m%d")

# plot the result
plot(SumDateChristmas)
```



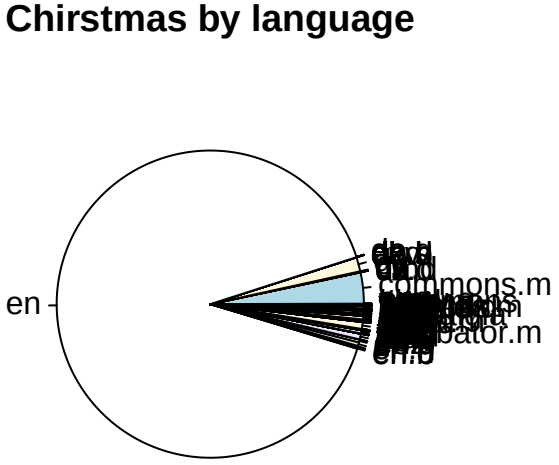
```
plot(SumDateMawlid)
```




```
#group by language(column X3) and sum the hits(column X4)
SumLanMawlid <- Mawlid %>%
  group_by(X3) %>%
  summarise(X4 = sum(X4))

SumLanChrist <- Christmas %>%
  group_by(X3) %>%
  summarise(X4 = sum(X4))

# plot pie chart
pie(c(SumLanMawlid$X4), labels = c(SumLanMawlid$X3), main = "Mawlid by language")
```




```
pie(c(SumLanChrist$X4), labels = c(SumLanChrist$X3), main = "Chirstmas by language")
```



From the plot we can see that both holiday received the highest attension at the date it is celebrated, while although the population of both population is large, christmas received more attention by orders of magnitude. One reason might be that Chrismas has gone far beyond religious meaning and is celebrated

world wide, and Mawlid is only mainly known to Islamic people.

more over, while langage for christmas is dominated by english, people interested in Mawlid speak franch, German and Dutsch, varying a lot more.

3 problem 4

3.1 a)

First I tried to use read_delim to read in the files, each one would produce a couple warnings.

I used grep. but this is a bit slow and reading 1/10 of all files would cost 3889s, i.e. 64min.

```
install.packages("readr")
library(readr)
require(parallel)
require(doParallel)
library(foreach)

path = '/global/scratch/paciorek/wikistats_full/dated_for_R'
files <- list.files(path, full.names = T)

nCores <- 24
cl <- makeCluster(nCores)

ObamaDF <- foreach(file=sample, .combine = rbind) %dopar%{
  lines <- readr::read_delim(file, " ",col_names = F, quote = "")
  ObamaLines <- lines[grep("Barack_Obama",lines$X4),]
  return(ObamaLines)
}

outFile = "/global/scratch/yue_hu/sample2.csv"

write.csv(ObamaDF, file = outFile, row.names = FALSE)
```

Then I tried to use dplyr r package and pipe the dataframe to a filter. This is faster and I could read a quater of the files within time limit.

```
library(stringr)
library(dplyr)
require(parallel)
require(doParallel)

path = '/global/scratch/paciorek/wikistats_full/dated_for_R'
files <- list.files(path, full.names = T)

nCores <- 24
cl <- makeCluster(nCores)

ObamaDF <- foreach(file=files[1:240], .combine = rbind) %dopar%{
  readr::read_delim(file, " ",col_names = F, quote = "") %>%
    filter(str_detect(X4, "Barack_Obama"))

}

writeFile = "/global/scratch/yue_hu/Obama_pipe.csv"
```

```
write.csv(ObamaDF, file = writeFile , row.names = FALSE)
```

The result is like following

```
library(readr)
# read in the dataframe
Obama <- readr::read_delim("Obama_quater.csv",",", col_names = T)

## Parsed with column specification:
## cols(
##  X1 = col_integer(),
##  X2 = col_character(),
##  X3 = col_character(),
##  X4 = col_character(),
##  X5 = col_integer(),
##  X6 = col_integer()
## )

# show some features of the dataframe
head(Obama)

## # A tibble: 6 x 6
##         X1     X2    X3
##      <int>  <chr> <chr>
## 1 20081129 210000    pt
## 2 20081014 190000    en
## 3 20081108 190000    no
## 4 20081128 190001    en
## 5 20081110 160000    et
## 6 20081101 110000    fr
## # ... with 3 more variables: X4 <chr>, X5 <int>, X6 <int>

print(Obama$X4[1:6])

## [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"

nrow(Obama)

## [1] 109865
```

Lastly I tried readlines to read, but still too slow.

```
library(stringr)
library(dplyr)
require(parallel)
require(doParallel)

path = '/global/scratch/paciorek/wikistats_full/dated_for_R'
files <- list.files(path, full.names = T)
```

```
nCores <- 24
cl <- makeCluster(nCores)

ObamaDF <- foreach(file=files[1:240], .combine = rbind) %dopar%{
  Lines <- readLines(file)
  Obama <- Lines[str_detect(Lines,"Barack_Obama")]
}

writePath = "/global/scratch/yue_hu/Obama_str.csv"
write.csv(ObamaDF, file = writePath, row.names = FALSE)
```

3.2 b)

It took 6347 seconds, i.e. 105 min to run a quater of the files on 24 cores , so if we are on 96 cores to run the whole file it would coist 105*4/4= 105min, so this way it is less effective than Saprk code.