STATS506 Homework3

Haichao Ji

2022-11-30

```
rm(list = ls())
setwd(getwd())
```

(Q1)

Git link: https://github.com/SymmeJ/HW3

(1)

Global level git configuration files locate in $C:\backslash Users\backslash username\backslash .gitconfig$ and lobal level git configuration files locate in $< git-repo > \backslash .git\backslash config$ for Windows.\

credential.helper=osxkeychain

filter.lfs.clean=git-lfs clean – %f

filter.lfs.smudge=git-lfs smudge – %f

filter.lfs.process=git-lfs filter-process

filter.lfs.required=true

user.name=Symmes

user.email=haichao1121@gmail.com

(2)

```
readLines = function(n1,n2){
nms = read.delim("Data/2020_Business_Academic_QCQ.txt",header = FALSE, sep = ',', nrows = 1)
df = read.delim("Data/2020_Business_Academic_QCQ.txt",header = FALSE, sep = ',', nrows = n2 - n1, skip =
names(df) <- nms
df[df==""] <- NA
res = na.omit(df[,c(4,7,28,29,45)])
names(res) <- c("state","county_code","employee_size","sales_volume","census_tract")
res
}
```

(3)

```
res3 = data.frame(row.names = c("state","county_code","employee_size","sales_volume","census_tract"))
for(i in 1:15){
  df = readLines(20000*(i-1)+1,20000*i+1)
  res3 = rbind(res3,df)
}
res3 = res3[res3$state == 'AL',]
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
```

```
df1 = summarise(group_by(res3,census_tract),sum(employee_size),sum(sales_volume))
names(df1) <- c("census_tract","employee_size","sales_volume")
```

(4)

```
library(RMySQL)
```

```
## Loading required package: DBI
```

```
library(DBI)
mydb <- dbConnect(MySQL(), user='root', password='jhcbywzfb7', dbname='Hw3db',host = "localhost", port =
dbWriteTable(mydb, value = df1, name = "df1", overwrite = TRUE)
```

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

(5)

```
dbGetQuery(mydb, 'SELECT census_tract FROM df1 ORDER BY sales_volume DESC LIMIT 10')
```

```
##    census_tract
## 1          5101
## 2          4500
## 3          2700
## 4           201
```

```
## 5          10701
## 6          11200
## 7           3100
## 8           1402
## 9           2400
## 10          1200
```

(6)

On git by "git checkout -b new_branch".

(7)

```
dat7 = read.csv("Data/AL.csv", header = TRUE)
```

```
dat7 = dat7[,c(19,20,22,45,64,65)]
names(dat7) <- c("wealth_finder_score","find_div_1000","estimated_home_value_div_1000","state","census_
dat7 = dat7[dat7$estimated_home_value_div_1000 != 0,]
dat7 = summarise(group_by(dat7,census_2010_tract),mean(wealth_finder_score),mean(find_div_1000),mean(es
names(dat7) <- c("census_2010_tract","wealth_finder_score","find_div_1000","estimated_home_value_div_10
df2 = dat7
```

(8)

```
dbWriteTable(mydb, value = df2, name = "df2", overwrite = TRUE)
```

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

(9)

HEAD is the current branch.

(10)

```
library(tidycensus)
dat10white = tidycensus::get_decennial(geography = "tract", state = c('AL'), variables = c("H006002"), y
```

```
## Getting data from the 2010 decennial Census
```

```
## Downloading feature geometry from the Census website.  To cache shapefiles for use in future sessions
```

```
## Using Census Summary File 1
```

```
##   |                                                                              |
```

```
dat10black = tidycensus::get_decennial(geography = "tract", state = c('AL'), variables = c("H006003"), y
```

```
## Getting data from the 2010 decennial Census
```

```
## Downloading feature geometry from the Census website.  To cache shapefiles for use in future sessions
```

```
## Using Census Summary File 1
```

```
dat10asian = tidycensus::get_decennial(geography = "tract", state = c('AL'), variables = c("H006005"), y
```

```
## Getting data from the 2010 decennial Census
```

```
## Downloading feature geometry from the Census website.  To cache shapefiles for use in future sessions
```

```
## Using Census Summary File 1
```

```
tract = tidycensus::get_decennial(geography = "tract", state = c('AL'), variables = c("TRACT"), year = 1
```

```
## Getting data from the 2010 decennial Census
```

```
## Downloading feature geometry from the Census website.  To cache shapefiles for use in future sessions
```

```
## Using Census Summary File 1
```

```
dat10 = data.frame(GEOID = as.integer(dat10asian$GEOID), white = dat10white$value, black = dat10black$va
tract = data.frame(GEOID = as.integer(tract$GEOID), tract = tract$value)
df3 = merge(x = dat10, y = tract, by.x = "GEOID", by.y = "GEOID", all.x = TRUE, all.y = FALSE)
dbWriteTable(mydb, value = df3, name = "df3", overwrite = TRUE)
```

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

(11)

```
df23 = dbGetQuery(mydb, 'SELECT df3.white, df3.black, df3.asian, df3.tract,
                  df2.wealth_finder_score, df2.find_div_1000,
                  df2.estimated_home_value_div_1000
                  FROM df2 LEFT JOIN df3 ON df3.tract = df2.census_2010_tract')
df11 = dbGetQuery(mydb, 'SELECT * FROM df1')
df = merge(x = df23, y = df11[,-1], by.x = "tract", by.y = "census_tract",all.x = TRUE, all.y = FALSE)
```

(12)

% git log commit c88a02deacfb3835cf9e64c6fb68c42d6c0f416b (HEAD -> new_branch) Author: Symmes < *haichao*1121@*gmail.com* > Date: Wed Nov 30 17:00:48 2022 -0500

question 12

commit e19cf3dbef688897739729bcc9cadccf78d9f8a3 Author: Symmes < *haichao*1121@*gmail.com* > Date: Wed Nov 30 15:34:01 2022 -0500

question 9

commit e3ac6ea6be081c06adbb8b1cb6673a92ca6f2a51 Author: Symmes < *haichao*1121@*gmail.com* > Date: Wed Nov 30 15:33:46 2022 -0500

question 9

commit c11774854b06372dfd3a30c80d8548022a1472e1 (origin/master, master) Author: Symmes < *haichao*1121@*gmail.com* > Date: Wed Nov 30 15:07:06 2022 -0500

I used "git reset new_branch" to reset the repository to the older version.

(13)

```
mod = lm(sales_volume~.-tract,data = df)
summary(mod)
```

```
##
## Call:
## lm(formula = sales_volume ~ . - tract, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4436803  -226280  -132302   125414  4310962
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   238524.167  76309.107   3.126 0.001817 **
## white                            42.479     22.719   1.870 0.061770 .
## black                           159.797     42.459   3.764 0.000176 ***
## asian                          -301.419    558.675  -0.540 0.589627
## wealth_finder_score            -127.210     95.851  -1.327 0.184710
## find_div_1000                   696.022   1401.415   0.497 0.619525
## estimated_home_value_div_1000   511.049    421.294   1.213 0.225356
## employee_size                    72.705      2.519  28.867  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 505300 on 1169 degrees of freedom
##   (2 observations deleted due to missingness)
## Multiple R-squared:  0.4327, Adjusted R-squared:  0.4293
## F-statistic: 127.4 on 7 and 1169 DF,  p-value: < 2.2e-16
```

There is a racial bias at the level of sales volume. Larger size of black and white people, larger the sales volume.

(Q2)

(1)

A compute node offers resources such as processors, volatile memory (RAM), permanent disk space (e.g. SSD), accelerators (e.g. GPU) etc. A core is the part of a processor that does the computations. (Ref: https://stackoverflow.com/questions/65603381/slurm-nodes-tasks-cores-and-cpus#:~:text=A%20compute%20node%20offers%20resources,processor%20that%20does%20the%20computations.)

Log-in node is the connection between users and the server. Compute nodes are used to compute and there are a lot of compute nodes in a HPC.

(2)

$ sdev -h sdev: start an interactive shell on a compute node.

Usage: sdev [OPTIONS] Optional arguments: -c number of CPU cores to request (OpenMP/pthreads, default: 1) -n number of tasks to request (MPI ranks, default: 1) -N number of nodes to request (default: 1) -m memory amount to request (default: 4GB) -p partition to run the job in (default: dev) -t time limit (default: 01:00:00) -r allocate resources from the named reservation (default: none) -J job name (default: sdev) -q quality of service to request for the job (default: norma

$ srun -N 1 -n 4 –m 32GB –t 03:00:00

(3)

The original directory won't change.