Project2

Shuyao Wang

2023-03-01

```
library(dplyr)
library(tidyr)
library(ff)
library(data.table)
library(ggplot2)
library(biglm)
```

1. Read the file using 'Divide-and-Conquer' Strategy.

I first read 219001 rows of data and took loops 7 times to read the rest. I also recorded the time to read these two parts separately. Then I renamed the column into the first row of the data and deleted that row. It took me 5246.82 second to read all the data in total.

```
library(R.utils)

nrow <- countLines("ss13hus.csv.bz2")[1]
filename <- "ss13hus.csv.bz2"
con <- bzfile(filename, "rt")

time1 <- system.time(
  DT1 <- read.csv(con, header = F, nrows = 219001)
)

time2 <- system.time(
  for (i in (1:7)){
      print(i)
      tmp <- read.csv(con, header = F, nrows = 1000000)
      DT1 <- rbind(DT1, tmp)
  }
)

time <- time1 + time2
colnames(DT1) <- DT1[1,]
DT1 <- DT1[-1,]
```

2. Randomly sampling 3,000,000 survey records and extract certain fields.

I set the seed and used the sample_n function in the dplyr package to do the sampling. After selecting the specific columns, I saved the result into a csv.

```
set.seed(1000)

DT_subset <- sample_n(DT1, 3000000)
```

```
DT_subset <- DT_subset %>%
  select(c(REGION, ST, ADJHSG, ADJINC, NP, ACR, BDSP, ELEP,
           GASP, RMSP, VEH, WATP, FINCP, HINCP))

write.csv(DT_subset, "./ss13hus_subset.csv")
```

3. Try 3 different functions of reading the data from Step2.

It took fewest time to read the file with fread function. The result read by read.csv.ffdf() is a list. Others are normal data frames.

```
time_ff <- system.time(DT_ff <- read.csv.ffdf(file="./ss13hus_subset.csv",
                                              header = TRUE, colClasses=NA))
time_ff
```

```
##    user  system elapsed
##  16.384   1.094  18.134
```

```
time_csv <- system.time(DT_csv <- read.csv("./ss13hus_subset.csv"))
time_csv
```

```
##    user  system elapsed
##  21.283   1.055  22.547
```

```
time_fr <- system.time(DT_fr <- data.table::fread("./ss13hus_subset.csv"))
time_fr
```

```
##    user  system elapsed
##   1.383   0.129   1.523
```

4. Scatter plot of BDSP and FINCP.

Since simply plotting the graph produces a warning of containing missing values, we need to deal with them first. I replaced missing values by median since distribution were quite skewed from the histograms. Then I adjusted FINCP to constant dollars and plotted the scatterplots with gam smoother. Since I did not get the graph after waiting for 2hr, I decided to draw 10000 samples instead.

```
# Take 10000 samples.
set.seed(1000)

DT_graph <- sample_n(DT1, 10000)

# Check for percent of missing values.
apply(DT_graph, 2, function(col)sum(is.na(col))/length(col))
# BDSP and FINCP have missing values.

# Turn into numeric format.
DT_graph$ADJINC <- as.numeric(DT_graph$ADJINC)
DT_graph$BDSP <- as.numeric(DT_graph$BDSP)
DT_graph$FINCP <- as.numeric(DT_graph$FINCP)

# Plot the distribution of variables which contains missing values.
hist(DT_graph$FINCP)
hist(DT_graph$BDSP)
# Since the distribution of FINCP is skewed, though does not include large outliers,
# I choose to use mode to replace missing values of it.
```

```
# Moreover, though the distribution of BDSP is not quite skewed, it contains large
# outliers. Thus, I also choose to use median to replace missing values of it.

# Deal with missing values.
DT_graph$FINCP[is.na(DT_graph$FINCP)] <- median(DT_graph$FINCP, na.rm=TRUE)
DT_graph$BDSP[is.na(DT_graph$BDSP)] <- median(DT_graph$BDSP, na.rm=TRUE)

# Adjust FINCP to constant dollars.
DT_graph$FINADJ <- DT_graph$FINCP*((1e-6)*DT_graph$ADJINC)

# Make the plot.
plot <- ggplot(DT_graph, aes(x = BDSP, y = FINADJ)) +
      geom_point() + geom_smooth(method = "loess", se = TRUE, linewidth = 1.2) +
      xlab("Number of bedrooms") + ylab("Family Income (Dollars)")
```

```
pdf("~/Desktop/plot.pdf")
plot
```

```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x7f7eccdbe670>
## <environment: namespace:base>
```

```
dev.off()
```

```
## pdf
##   2
```

5. Linear regression.

I set the seed with 1000 and did the sampling. Then I selected with specific columns and turned the results into numeric format. Since the result contained much NA value and the distribtuions were skewed, I replaced NA values with median values of the rest. Then I adjusted FINCP to constant dollars and fitted the result with lm model. The coeff of BDSP is 11216.31.

I repeated the process for 1000 times, but may be due to the issue of my computer's battery, I took me really long time to process and my computer went die for running 100 runs. To deal with the issue, I tried the following ways: First, I had to divide the process into 6 parts and run them by different equipments by borrowing university's laptops. The sum of the running time is nearly 14 hr, but I don't think the time is very precise since the rent Mac air just took 20 min to process with 100 runs, but others like my Mac Pro takes around 2 hr with exact the same lines of code. I was thinking about using berkeley ssh to process, but unfortuntely it cannot process with files large than 55MB, the same error happen while using thinkpad.

```
set.seed(1000)
# Sampling
DT_subset2 <- sample_n(DT1, 1000000)

# Select columns
DT_subset2 <- DT_subset2 %>%
  select(c(ADJINC, BDSP, VEH, FINCP))

# Turn into numeric format.
DT_subset2$ADJINC <- as.numeric(DT_subset2$ADJINC)
DT_subset2$BDSP <- as.numeric(DT_subset2$BDSP)
DT_subset2$VEH <- as.numeric(DT_subset2$VEH)
DT_subset2$FINCP <- as.numeric(DT_subset2$FINCP)
```

```r
# Check NA
apply(DT_subset2, 2, function(col)sum(is.na(col))/length(col))

# BDSP, FINCP, VEH contains missing values
hist(DT_csv2$BDSP)
hist(DT_csv2$FINCP)
hist(DT_csv2$VEH)
# Since the distributions of FINCP and VEH are skewed, though does not include
# large outliers, I choose to use mode to replace missing values of it.
# Moreover, though the distribution of BDSP is not quite skewed, it contains large
# outliers. Thus, I also choose to use median to replace missing values of it.

# Deal with missing values.
DT_subset2$FINCP[is.na(DT_subset2$FINCP)] <- median(DT_subset2$FINCP, na.rm=TRUE)
DT_subset2$BDSP[is.na(DT_subset2$BDSP)] <- median(DT_subset2$BDSP, na.rm=TRUE)
DT_subset2$VEH[is.na(DT_subset2$VEH)] <- median(DT_subset2$VEH, na.rm=TRUE)

# Adjust FINCP to constant dollars.
DT_subset2$FINADJ <- DT_subset2$FINCP*((1e-6)*DT_subset2$ADJINC)

# Fit the model
model_lm <- lm(FINADJ ~ BDSP + VEH, data = DT_subset2)
model_lm$coefficients[2]
# The estimated coefficient for BDSP is 11216.31.
```

```r
# Repeat 1000 time with different random seed.
result700 <- c()

time_repe <- system.time(for (i in 911:1000){
  print(i)
  set.seed(i)

  # Take the samples.
  DT_subset_i <- sample_n(DT1, 1000000)

  # Select the columns.
  DT_subset_i <- DT_subset_i %>%
                 select(c(ADJINC, BDSP, VEH, FINCP))

  # Turn into numeric form.
  DT_subset_i$ADJINC <- as.numeric(DT_subset_i$ADJINC)
  DT_subset_i$BDSP <- as.numeric(DT_subset_i$BDSP)
  DT_subset_i$VEH <- as.numeric(DT_subset_i$VEH)
  DT_subset_i$FINCP <- as.numeric(DT_subset_i$FINCP)

  # Replace NA.
  DT_subset_i$FINCP[is.na(DT_subset_i$FINCP)] <- median(DT_subset_i$FINCP, na.rm=TRUE)
  DT_subset_i$BDSP[is.na(DT_subset_i$BDSP)] <- median(DT_subset_i$BDSP, na.rm=TRUE)
  DT_subset_i$VEH[is.na(DT_subset_i$VEH)] <- median(DT_subset_i$VEH, na.rm=TRUE)

  # Adjust FINCP to constant dollars.
  DT_subset_i$FINADJ <- DT_subset_i$FINCP*((1e-6)*DT_subset_i$ADJINC)
```

```r
  # Fit model.
  model_lm_i <- lm(FINADJ ~ BDSP + VEH, data = DT_subset_i)

  # Record results.
  result700[i] <- model_lm_i$coefficients[2]
})

result700[1:100] <- result
result700[101:200] <- result100[101:200]
result700[201:300] <- result200[201:300]
result700[301:600] <- result300[301:600]
result600 <- readRDS('result600.RData')
result700[601:700] <- result600[601:700]
result700[701:1000] <- result700[701:1000]

saveRDS(result700, file = 'finalresult.Rds')
```

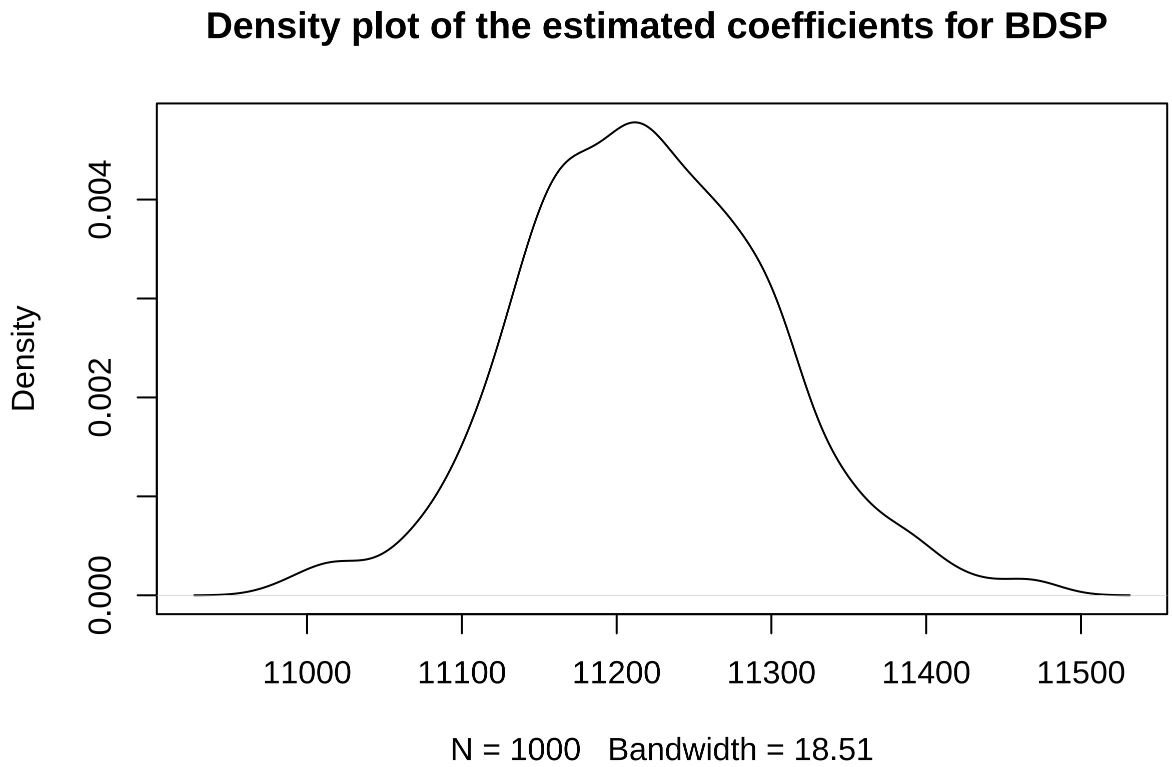
```r
finalresult <- readRDS(file = 'finalresult.Rds')

mean1000 <- mean(finalresult)
mean1000
```

```
## [1] 11219.46
```

```r
sd1000 <- sd(finalresult)
sd1000
```

```
## [1] 81.87784
```

```r
# Filled Density Plot
d <- density(finalresult)
plot(d, main = "Density plot of the estimated coefficients for BDSP ")
```


Density plot of the estimated coefficients for BDSP
Density
0.000
0.002
0.004
11000
11100
11200
11300
11400
11500
N = 1000   Bandwidth = 18.51