Stat 5014 HW2

Bob Settlage

2017-09-11

Problem 4

Version control can assist in:

- first thought
- second thought
- third thought

The last way to make lists was more explicit and offers more control, but sometimes you just want a simple list or are targeting html so do it this way (note blank line and two spaces are important):

- another way
- list item
- list item

Problem 5

Here we will read in, clean and filter datasets with the final goal of creating tidy datasets. I am going to create a figure for each one to play with plotting functions.

CMM data

First, we will read in and create a tidy dataset. After tidying, a summary is in Table 1 with a boxplot in Figure 1. I will put this code in an Appendix.

Table 1: CMM data summary

part	operator	replicate	value
Min. : 1.0	Length:40	Min. :1.0	Min. :0.250
1st Qu.: 3.0	Class :character	1st Qu.:1.0	1st Qu.:0.289
Median : 5.5	Mode :character	Median :1.5	Median :0.301
Mean : 5.5	NA	Mean :1.5	Mean :0.302
3rd Qu.: 8.0	NA	3rd Qu.:2.0	3rd Qu.:0.317
Max. :10.0	NA	Max. :2.0	Max. :0.341

Finally, lets create a quick linear model to play with tables. Note, this analyis is not technically correct, you will learn more about why in the Design of Experiments class. We will use this model:

$$y_i = intercept + part_i + operator_i + \epsilon_i \tag{1}$$

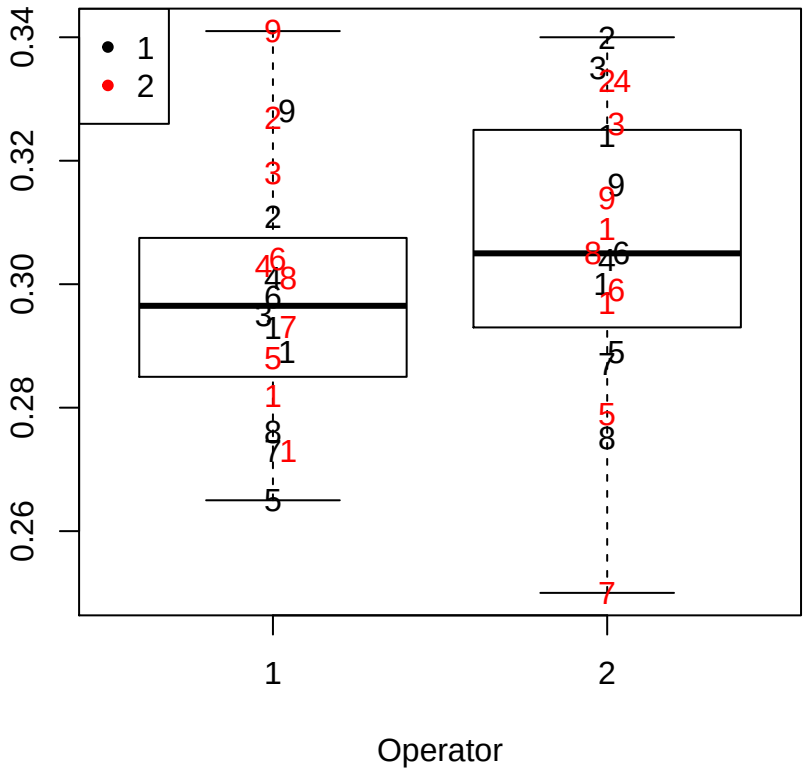

Figure 1: CMM data, boxplot by operator, color by replicate, label is part number.

Table 2: Playing with tables

	Dependent variable:
	value
as.factor(part)2	0.029*** (0.010)
as.factor(part)3	0.020* (0.010)
as.factor(part)4	0.012 (0.010)
as.factor(part)5	−0.019* (0.010)
as.factor(part)6	0.003 (0.010)
as.factor(part)7	−0.023** (0.010)
as.factor(part)8	−0.009 (0.010)
as.factor(part)9	0.026** (0.010)
as.factor(part)10	−0.006 (0.010)
as.factor(operator)2	0.008* (0.004)
Constant	0.295*** (0.007)
Observations	40
R^2	0.681
Adjusted R^2	0.571
Residual Std. Error	0.014 (df = 29)
F Statistic	6.191*** (df = 10; 29)
Note:	*p<0.1; **p<0.05; ***p<0.01

Now for the actual homework:

Problem 5

Part A: Sensory data

Table 3: Sensory data summary

Item	Person	value
Length:150	Length:150	Min. :0.70
Class :character	Class :character	1st Qu.:3.02
Mode :character	Mode :character	Median :4.70
NA	NA	Mean :4.66
NA	NA	3rd Qu.:6.00
NA	NA	Max. :9.40

Part B: Long Jump data

Table 4: Long Jump data summary

YearCode	Year	dist
Min. :-4.0	Min. :1896	Min. :250
1st Qu.:21.0	1st Qu.:1921	1st Qu.:295
Median :50.0	Median :1950	Median :308
Mean :45.5	Mean :1945	Mean :310
3rd Qu.:71.0	3rd Qu.:1971	3rd Qu.:328
Max. :92.0	Max. :1992	Max. :350

Part C: Brain vs Body data

Table 5: Brain/Body weight data summary

Brain	Body
Min. : 0.00	Min. : 0.10
1st Qu.: 0.60	1st Qu.: 4.25
Median : 3.34	Median : 17.25
Mean : 198.79	Mean : 283.13
3rd Qu.: 48.20	3rd Qu.: 166.00
Max. :6654.00	Max. :5712.00

Part C: Tomato data

Table 6: Tomato data summary

Clone	Replicate	value	Variety
Length:18	Length:18	Length:18	Length:18

Clone	Replicate	value	Variety
Class :character	Class :character	Class :character	Class :character
Mode :character	Mode :character	Mode :character	Mode :character

Problem 7

In this problem, we are to munge and create some summary statistics for a large dataset. Going in, we know this is a large dataset, so I Google'd "read large dataset in r" and came up with https://rpubs.com/msundar/large_data_analysis as my top hit. At 2015, it is a *little* dated, but reading through it, it seems to have some good advice, so I am going to try it and see. The first suggestion is to read in a little data so that we know can set the column classes, so let's do that on all three datafiles (I downloaded them onto my local drive via wget)

```
## [1] "Gebreken"
```

```
##    Gebrek.identificatie     Ingangsdatum.gebrek        Einddatum.gebrek
##             "character"               "integer"               "integer"
## Gebrek.paragraaf.nummer   Gebrek.artikel.nummer     Gebrek.omschrijving
##               "integer"             "character"             "character"
```

```
## [1] "Geconstat"
```

```
##                          Kenteken Soort.erkenning.keuringsinstantie
##                         "character"                       "character"
## Meld.datum.door.keuringsinstantie  Meld.tijd.door.keuringsinstantie
##                         "integer"                         "integer"
##              Gebrek.identificatie      Soort.erkenning.omschrijving
##                         "character"                       "character"
##     Aantal.gebreken.geconstateerd
##                         "integer"
```

```
## [1] "Personen"
```

```
##                          Kenteken                      Voertuigsoort
##                         "character"                       "character"
##                              Merk                     Handelsbenaming
##                         "character"                       "character"
##              Datum.tenaamstelling                         Bruto.BPM
##                         "character"                       "integer"
##                     Cilinderinhoud               Massa.ledig.voertuig
##                          "integer"                        "integer"
## Toegestane.maximum.massa.voertuig            Datum.eerste.toelating
##                          "integer"                       "character"
##    Datum.eerste.afgifte.Nederland                    Catalogusprijs
##                         "character"                       "integer"
##                       WAM.verzekerd
##                         "character"
```

From the discussion, it looks to me as if the path of least resistance is to use fread and data.tables. As a cool aside, note that you can print data.tables and it only shows the first and last 5 lines AND if you have too many columns, it only shows what fits to screen. Data.tables are replacing data.frames. They are faster to manipulate and offer additional functionality. Continuing with this problem, I think we should limit the import to only those columns we are interested in to save some space. We could filter on date (2017) in line, but for this I am thinking we will convert to date and then filter. Actually, I hate dealing with dates, so I am filtering using grep.

Appendix 1: R code

```
########################### Problem5_CMM_analysis get data
url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/CMM.dat"
CMM_raw <- read.table(url, header = F, skip = 1, fill = T,
    stringsAsFactors = F)
CMM_tidy <- CMM_raw[-1, ]
colnames(CMM_tidy) <- c("part", "Op1_1", "Op1_2", "Op2_1",
    "Op2_2")
CMM_tidy <- CMM_tidy %>% gather(op_rep, value, Op1_1:Op2_2) %>%
    separate(op_rep, into = c("operator", "replicate"),
        sep = "_") %>% mutate(operator = gsub("Op", "",
    operator)) %>% mutate(replicate = as.numeric(replicate)) %>%
    mutate(part = as.numeric(part))
############################

########################### Problem5_CMM_analysis plot
boxplot(value ~ operator, data = CMM_tidy, xlab = "Operator")
beeswarm(value ~ operator, data = CMM_tidy, pwcol = CMM_tidy$replicate,
    pwpch = as.character(CMM_tidy$part), add = T)
legend("topleft", legend = levels(as.factor(CMM_tidy$replicate)),
    pch = 20, col = levels(as.factor(CMM_tidy$replicate)))
```