Homework2

Yueyao Wang

September 10, 2017

Problem 4

Version control can help me keep track of how I coded throughout the whole analysis process. So when I would like to go back to a version of program it would be very easy. This also allows me to do different manipulation on the same dataset at the same time. I don't have to worry about ruining the data. I can also share my work with my teammates when doing the group projects.

Problem 5

a Sensory Data

The issue with the data is that the column headers are values, not variable names. "Person1",... "Person5" should be the values of variable person.

Table 1: Sensory data summary

Item	person	value
Length:150	Min. :1	Min. :0.700
Class :character	1st Qu.:2	1st Qu.:3.025
Mode :character	Median :3	Median :4.700
NA	Mean :3	Mean :4.657
NA	3rd Qu.:4	3rd Qu.:6.000
NA	Max. :5	Max. :9.400

b Long Jump Data

The issue with this data is that it save the same variable into several columns.

Table 2: Long Jump data summary

year_code	year	long_jump
Min. :-4.00	Min. :1986	Min. :249.8
1st Qu.:21.00	1st Qu.:2011	1st Qu.:295.4
Median :50.00	Median :2040	Median :308.1
Mean :45.45	Mean :2035	Mean :310.3
3rd Qu.:71.00	3rd Qu.:2061	3rd Qu.:327.5
Max. :92.00	Max. :2082	Max. :350.5

c Brain and Body Weight Data

The issue with this data is also save value of one variable into serveral columns.

Table 3: Brain/Body weight data summary

Brain	Body
Min. : 0.005	Min. : 0.10
1st Qu.: 0.600	1st Qu.: 4.25
Median : 3.342	Median : 17.25
Mean : 198.790	Mean : 283.13
3rd Qu.: 48.203	3rd Qu.: 166.00
Max. :6654.000	Max. :5712.00

d Tomato Data

This data saves multiple data into one cell.

Table 4: Tomato data summary

clone	replicate	value	variety
Min. :10000	Min. :1	Length:18	Ife#1 :9
1st Qu.:10000	1st Qu.:1	Class :character	PusaEarlyDwarf:9
Median :20000	Median :2	Mode :character	NA
Mean :20000	Mean :2	NA	NA
3rd Qu.:30000	3rd Qu.:3	NA	NA
Max. :30000	Max. :3	NA	NA

Problem 6

I created a new variable pH_Coverage = pH_Max - pH_Min to combine the information in pH_Min and pH_Max. When I want to measure the relationship between pH and foliage color, I first thought about predicting foliage color using the pH information. However this looks like a multi-dimension classification problem. So I think the other way around and decide to see if we can have a prediction for the pH when we have the foliage color.
So the model is following:
Model: $pH\_Coverage = Foliage\_Color + \epsilon$

	Estimate	Std. Error	t value	Pr(>|t|)
Foliage_ColorDark Green	2.176829	0.0868141	25.074616	0.0e+00
Foliage_ColorGray-Green	2.712000	0.1572270	17.248951	0.0e+00
Foliage_ColorGreen	2.367630	0.0298844	79.226393	0.0e+00
Foliage_ColorRed	1.925000	0.3930674	4.897379	1.2e-06
Foliage_ColorWhite-Gray	2.644444	0.2620449	10.091569	0.0e+00
Foliage_ColorYellow-Green	1.975000	0.1757851	11.235311	0.0e+00

```
##                Df Sum Sq Mean Sq F value Pr(>F)
## Foliage_Color   6   4607   767.9    1243 <2e-16 ***
## Residuals     826    510     0.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Problem 7

```
################## load datasets into R   ##################
library(data.table)

#this had the defect code and description
Car_Gebreken_select <- fread(input = "Open_Data_RDW__Gebreken.csv", header = T, select=c(1,6), showProg
#this has the license plate, inspection date and defect code
Car_Geconstat_select <- fread(input = "Open_Data_RDW__Geconstateerde_Gebreken.csv", header=T, select=c(
#this has the license plate, make and model of vehicle
Car_Person_select <- fread(input = "Personenauto_basisdata.csv", header=T, showProgress = F, select = c
Car_Geconstat_select_2017 <- Car_Geconstat_select[grep("2017",Car_Geconstat_select$"Meld datum door keu

################ join them together  ################
### First join Car_Person_select and Car_Geconstat_select by license plate
### Then join the merged one with Car_Gebreken_select by defect code
Merged_Geconstat_Person <- merge(Car_Geconstat_select,Car_Person_select,by = 'Kenteken',all = T)
Merged_All <- merge(Merged_Geconstat_Person,Car_Gebreken_select,by = 'Gebrek identificatie',all = T)

################ clean the data  ###################
### translate the variables and remove observations containg missing values for any of the variables
colnames(Merged_All) <- c("defect_code","liscence_plate","inspection_date","make","model","defect_descr
### s is a logic vector: whether the observation has any missing value of any of those variables.
s = apply(Merged_All,MARGIN = 1,function(x) sum(is.na(x)))>0
Merged_tidy <- Merged_All %>% mutate(has_missing_value = s) %>%
              filter(has_missing_value == FALSE)
```

d.how many DIFFERENT makes and models of cars you end with (?unique ?distinct ?duplicated) considering only year 2017

```
Merged_tidy_2017 <- Merged_tidy[grep("2017",Merged_tidy$inspection_date),]
diff_makes <- n_distinct(Merged_tidy_2017$make)
diff_models <- n_distinct(Merged_tidy_2017$model)
```

The different makes of cars is 503 and the different models of cars is 33470.

e. report a table of the 5 most frequent defects (translated) and the top make/models having that defect (?count) again considering only year 2017

```
summary1 <- Merged_tidy_2017 %>% group_by(defect_code,make) %>%
  summarise(make_frequency = n()) %>% group_by(defect_code) %>%
  mutate(defect_frequency = sum(make_frequency)) %>%
  mutate(top_make_number = max(make_frequency)) %>%
  filter(make_frequency %in% top_make_number) %>%
  arrange(desc(defect_frequency)) %>%
  select(-top_make_number)
kable(summary1[1:5,],caption = "Top 5 mose frequent defects and the top coresponding make")
```

Table 6: Top 5 mose frequent defects and the top coresponding make

defect_code	make	make_frequency	defect_frequency
AC1	VOLKSWAGEN	43490	385634

defect_code	make	make_frequency	defect_frequency
K04	PEUGEOT	27704	271273
RA2	OPEL	29089	223171
205	VOLKSWAGEN	18395	171255
497	PEUGEOT	16316	139978

f. use function lm to test for a relationship between number of defects observed by make, report both the coefficient and anova tables (2017 only)

```
summary2 <- Merged_tidy_2017 %>% group_by(make) %>%
  summarise(number_defect = n_distinct(defect_code))
kable(head(summary2),caption = "number of distinct defects observed by it make")
```

Table 7: number of distinct defects observed by it make

make	number_defect
A.C.L.	2
A.M.C.	17
ACCUBUILT	9
ACM	1
ACURA	12
ADRIA	64

```
#fit1 <- lm(number_defect~make,data = summary2)
```

g.repeat (f) by model (2017 only)

```
summary3 <- Merged_tidy_2017 %>% group_by(model) %>%
  summarise(number_defect = n_distinct(defect_code))
kable(head(summary3),caption = "number of distinct defects observed by it model")
```

Table 8: number of distinct defects observed by it model

model	number_defect
	95
-	5
—	5
—-	53
——	40
———	1

```
#fit2 <- lm(number_defect~model,data = summary3)
```

h.comment on this workflow and how you might be more computationally efficient

I think save those important intermediate dataset for those summarise analysis would improve the efficiency a lot.

Appendix: Code

```
library(dplyr)
library(tidyr)
library(readr)
library(knitr)
Sensory_data <- read.table("Sensory.txt", header = F, skip = 1, fill = T, stringsAsFactors = F)
Sensory_tidy1 <- Sensory_data[-1, ] %>% filter(V1 %in% 1:10) %>% rename(Item = V1,
    V1 = V2, V2 = V3, V3 = V4, V4 = V5, V5 = V6)
Sensory_tidy2 <- Sensory_data[-1, ] %>% filter(!(V1 %in% 1:10)) %>% mutate(Item = rep(as.character(1:10)
    each = 2)) %>% select(Item, V1:V5) %>% mutate(V1 = as.numeric(V1))

Sensory_tidy <- bind_rows(Sensory_tidy1, Sensory_tidy2) %>% gather(person, value,
    V1:V5) %>% arrange(as.numeric(Item)) %>% mutate(person = parse_number(person))

kable(summary(Sensory_tidy), caption = "Sensory data summary")
LongJump_data <- read.table("LongJumpData.txt", header = T, fill = T)
LongJump_tidy <- LongJump_data %>% select(Year:Long.2)
## extract two columns
yearcode <- LongJump_tidy[, c(1, 3, 5, 7)] %>% gather(name1, year_code, Year:Year.2)
longjump <- LongJump_tidy[, c(2, 4, 6, 8)] %>% gather(name2, long_jump, Long:Long.2)
## combine them together and remove missing value
LongJump_tidy <- bind_cols(yearcode, longjump) %>% select(year_code, long_jump) %>%
    filter(!is.na(year_code)) %>% mutate(year = year_code + 1990) %>% select(year_code,
    year, long_jump)

kable(summary(LongJump_tidy), caption = "Long Jump data summary")
BrainBody_data <- read.table("BrainandBodyWeight.txt", header = F, fill = T,
    skip = 1)
colnames(BrainBody_data) <- rep(c("Brain", "Body"), times = 3)
BrainBody_tidy <- rbind(BrainBody_data[, 1:2], BrainBody_data[, 3:4], BrainBody_data[,
    5:6]) %>% filter(!is.na(Brain)) %>% arrange(Brain)

kable(summary(BrainBody_tidy), caption = "Brain/Body weight data summary")
Tomato_data <- read.table("tomato.txt", header = F, fill = T, skip = 2, comment.char = "")
## use number as column name is easy to have error, so combine the number
## with a character and then can parse the number if needed
colnames(Tomato_data) <- c("variety", "C10000", "C20000", "C30000")
## first gather the header clone into one column separate the value column
## into different replicate and then gather replicates
Tomato_tidy <- Tomato_data %>% gather(clone, value, C10000:C30000) %>% separate(value,
    into = paste("replicate", 1:3, sep = ""), sep = ",") %>% gather(replicate,
    value, replicate1:replicate3) %>% mutate(clone = parse_number(clone), replicate = parse_number(repli
    select(clone, replicate, value, variety) %>% arrange(variety, clone, replicate)

kable(summary(Tomato_tidy), caption = "Tomato data summary")
library(knitr)
library(swirl)
# Path to data
.datapath <- file.path(path.package("swirl"), "Courses", "R_Programming_E",
    "Looking_at_Data", "plant-data.txt")
plants <- read.csv(.datapath, strip.white = TRUE, na.strings = "")
# Remove annoying columns
```

```r
.cols2rm <- c("Accepted.Symbol", "Synonym.Symbol")
plants <- plants[, !(names(plants) %in% .cols2rm)]
# Make names pretty
names(plants) <- c("Scientific_Name", "Duration", "Active_Growth_Period", "Foliage_Color",
    "pH_Min", "pH_Max", "Precip_Min", "Precip_Max", "Shade_Tolerance", "Temp_Min_F")
# remove observations with missing value
plants <- plants %>% select("Foliage_Color", "pH_Min", "pH_Max") %>% mutate(pH_Coveragenterval = pH_Max
    pH_Min, Foliage_Color = as.factor(Foliage_Color)) %>% filter(!is.na(pH_Coveragenterval) &
    !is.na(Foliage_Color))
lm1 <- lm(formula = pH_Coveragenterval ~ Foliage_Color + 0, data = plants)
kable(summary(lm1)$coefficients)
av1 <- aov(pH_Coveragenterval ~ Foliage_Color + 0, data = plants)
summary(av1)
```