HW2_Porter_Erica

Erica Porter

9/6/2017

Problem 4

Using version control will allow me to:

- Test code by using branching processes; I can backtrack to a working version if I have to abandon a portion of code
- Edit, modify, and copy pieces of code from other projects
- Compare my methods and the efficiency of my projects with those of classmates'
- Access projects and code when away from my personal computer, if necessary

Problem 5

a. The sensory data has formatting issues including:
   - The Item number is only listed for the first row of corresponding values;two thirds of the data are missing Item numbers.
   - The label "Operator" above the data set is extraneous since it is not a header with variable names; it may interfere with reading the data in neatly.
   - The values and operators span multiple columns; each variable needs to correspond to a separate column and each measurement to a single row, so the columns need to correspond separately to: Operator, value, and Item.
     - See Appendix for R code with comments about the data cleaning process.
     - Table 1 displays a summary of the Sensory data after cleaning/tidying the data.

b. The long jump data has formatting issues including:
   - Values for Year span multiple columns.
   - Values for Long Jump span multiple columns.
     - See Appendix for R code with comments about the data cleaning process.
     - Table 2 displays a summary of the Long Jump data after cleaning/tidying the data.

c. The Body/Brain Weight data has formatting issues including:
   - Values for Body Weight span multiple columns.
   - Values for Brain Weight span multiple columns.
   - Headings and values do not clearly align for each column.
     - See Appendix for R code with comments about the data cleaning process.
     - Table 3 displays a summary of the Body/Brain Weight data after cleaning/tidying the data.

Table 1: Sensory data summary

Item	Operator	value
Min. : 1.0	Length:150	Min. :0.700
1st Qu.: 3.0	Class :character	1st Qu.:3.025
Median : 5.5	Mode :character	Median :4.700
Mean : 5.5	NA	Mean :4.657
3rd Qu.: 8.0	NA	3rd Qu.:6.000

Item	Operator	value
Max. :10.0	NA	Max. :9.400

Table 2: Long Jump data summary

Year	Long Jump
Min. :-4.00	Min. :249.8
1st Qu.:21.00	1st Qu.:295.4
Median :50.00	Median :308.1
Mean :45.45	Mean :310.3
3rd Qu.:71.00	3rd Qu.:327.5
Max. :92.00	Max. :350.5

Table 3: Body & Brain Weight data summary

Body Wt	Brain Wt
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

Problem 6

To clean the plant data, I did the following:

- Saved the plant data as a text file and read the data in as a table.
- Removed the first unnecessary row.
- Assigned to the data more descriptive and clear column headings.
- Removed rows consisting of all NA values.

After cleaning the data, I converted the variable Foliage_Color to a numeric variable in order to test for a linear relationship between Foliage_Color and the pH values. First, I fit a simple model for Foliage_Color with pH_Max and pH_Min, but the coefficients in the model were not significant. See Table 4 and Table 5 for a summary of the fit and ANOVA of the fit, respectively. Next, I attempted the repeated the model, this time taking the logarithm of each variable; the coefficient for pH_Min was not significant, so I fit my final linear model with only log(Foliage_Color) and log(pH_Max). While I did not attempt every possible model, transformation, or combination of variables, the model that included only log(pH_Min) was most significant. See Table 6 and Table 7 for a sumary of the fit and ANOVA of this fit, respectively.

See the Appendix for the corresponding R code (data cleaning and model fits).

Table 4: Summary of fit with pHMax and pHMin

	Dependent variable:
	Foliage_Color
pH_Min	0.025 (0.046)
pH_Max	0.033 (0.045)
Constant	2.501*** (0.341)
Observations	813
R^2	0.001
Adjusted R^2	−0.001
Residual Std. Error	0.829 (df = 810)
F Statistic	0.595 (df = 2; 810)
Note:	*p<0.1; **p<0.05; ***p<0.01

Table 5: Summary of ANOVA for fit with pHMax and pHMin

Statistic	N	Mean	St. Dev.	Min	Max
Df	3	270.667	467.076	1	810
Sum Sq	3	185.639	320.829	0.365	556.101
Mean Sq	3	0.501	0.166	0.365	0.687
F value	2	0.595	0.090	0.531	0.658
Pr(>F)	2	0.442	0.035	0.417	0.466

Table 6: Summary of fit log(y) with log(pHMax)

	Dependent variable:
	log(Foliage_Color)
log(pH_Max)	0.302** (0.137)
Constant	0.400 (0.274)
Observations	813
R^2	0.006
Adjusted R^2	0.005
Residual Std. Error	0.362 (df = 811)
F Statistic	4.814** (df = 1; 811)
Note:	*p<0.1; **p<0.05; ***p<0.01

Table 7: ANOVA of fit log(y) with log(pHMax)

Statistic	N	Mean	St. Dev.	Min	Max
Df	2	406.000	572.756	1	811
Sum Sq	2	53.576	74.874	0.632	106.519
Mean Sq	2	0.382	0.354	0.131	0.632
F value	1	4.814		4.814	4.814
Pr(>F)	1	0.029		0.029	0.029

Problem 7

a. Loaded all three data sets into R by saving each as a .csv file in RStudio.

b. I used the `merge` function to merge data sets Personenauto_basisdata and Open_Data_RDW_Geconstateerde_Gebreken by (what I think is) license plate. Then, I merged this new data set with Open_Data_RDW_Gebreken.

c. After merging and organizing the data, I used a function to convert the first date column to character values and remove all except the final 2 digits (indicating year). If month or day were necessary for the subsequent analysis, I would have to use date conversions, but seeing as the original dates were given in a variety of different formats (mm/dd/yyyy, mm/dd/yy, and character text with no recognized date format), I was not able to convert all the dates to an identical format in the given time. Since the analysis asked only that we look at the year 2017, the last two digits indicating year were sufficient.

d. 71 unique car makes and 2746 unique car models for the year 2017. I found these values first by using the `aggregate` command to create a table of unique values for every year, and then by creating a subset of the data with only rows corresponding to 2017 and applying `length` and `unique` commands.

```
## [1] 71
```

```
## [1] 2746
```

```
##
##  K04  AC1  G05  K05  RA2
## 1452  983  671  577  554
```

Appendix: R Code and comments

```
### Sensory data
library(tidyr,quietly=T,warn.conflicts=F)
library(dplyr,quietly=T,warn.conflicts=F)
# Read data into R, removing the first row with unecessary information
url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
sensory <- read.table(url, header = F, skip = 1, fill = T, stringsAsFactors = F)
sensory_new <- sensory[-1,]
# The data only lists Item number for the first measurement, so create a new Item column
Items <- rep(1:10,each = 3)
sensory_new <- cbind(sensory_new,Items)
n = ncol(sensory_new)
# Call a function that shifts replicate measurements into the correct columns
for(i in 1:30) {if ((sensory_new[i,1] != sensory_new[i,n]) == TRUE) {sensory_new[i,1:6] <-c("NA",sensory_ne
sensory_new <- sensory_new[,-1]
colnames(sensory_new) <- c("Op1","Op2","Op3","Op4","Op5","Item")
# Make the Operator variable into a column and append to data
sensory_tidy <- sensory_new %>% gather(Operator,value,Op1:Op5) %>% mutate(Operator = gsub("Op", "",
Operator))
```

```
### Long Jump data
library(tidyr,quietly=T,warn.conflicts=F)
library(dplyr,quietly=T,warn.conflicts=F)
library(stringr,quietly=T,warn.conflicts=F)
# Read data into R, remove heading since misaligned
url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
jump <- read.table(url, header = F, skip = 1, fill = T)
jump_new <- jump
# Separate columns for year and jump into two data sets
n=ncol(jump_new)
m = nrow(jump_new)
even_indexes<-seq(2,n,2)
odd_indexes<-seq(1,n,2)
year <- data.frame(x=jump_new[1:m,odd_indexes])
jumps <- data.frame(y=jump_new[1:m,even_indexes])
```

```r
# List all years in one column and all jump measurements in one column
jump_full <- cbind(stack(year[1:4]),stack(jumps[1:4]))
jump_full <- jump_full[,c(1,3)]
# Combine data and remove rows of NA
jump_full <- na.omit(jump_full)
colnames(jump_full) <- c("Year","Long Jump")
```

```r
### Weight data
library(tidyr,quietly=T,warn.conflicts=F)
library(dplyr,quietly=T,warn.conflicts=F)
library(stringr,quietly=T,warn.conflicts=F)
# Read data into R, removing first line
url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
weight <- read.table(url, header = F, skip = 1, fill = T)
weightnew <- weight
# Specify column names and group Body and Brain columns into two sets; stack the data
colnames(weightnew) <- c("Body_1", "Brain_1", "Body_2", "Brain_2", "Body_3", "Brain_3")
Body <- data.frame(a = c(weightnew[,"Body_1"], weightnew[,"Body_2"], weightnew[,"Body_3"]))
Brain <- data.frame(b = c(weightnew[,"Brain_1"], weightnew[,"Brain_2"], weightnew[,"Brain_3"]))
# Create tidy data set by pasting separate columns for Body weight and Brain weight
weight2 <- cbind(Body,Brain)
weight2 <- na.omit(weight2)
colnames(weight2) <- c("Body Wt","Brain Wt")
```

```r
### Tomato data
 # url1 <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
 # tomato <- read.table(url1,header=F,skip=2,fill=T,comment.char="",na.strings=F)
 # tomato_new <- tomato %>%
 # separate(V2,into=paste("C10000",1:3,sep="_"),sep=",", remove=T,extra="merge")
 #   %>%
 # separate(V3,into=paste("C20000",1:3,sep="_"), sep=",", remove=T,extra="merge")
 #   %>%
 # separate(V4,into=paste("C30000",1:3,sep="_"), sep=",", remove=T,extra="merge")
 #   %>%
 # mutate(C10000_3=gsub(",","",C10000_3)) %>%
 # gather(Clone,value,,C10000_1:C30000_3) %>%
 # mutate(Variety=V1, Clone=gsub("C","",Clone)) %>%
 # mutate(Variety=gsub("\\\\#"," ",Variety)) %>%
 # separate(Clone,into=c("Clone","Replicate")) %>%
 # select(-V1,Variety,Clone,value) %>%
 # arrange()
```

```r
### Plant data (Problem 6)
# Read data into R, remove unecessary columns, give descriptive variable names
plants <- read.table(file="~/Desktop/Fall_2017_Courses/Statistical_Programming/STAT_5014_homework/STAT_5014
.cols2rm <- c('X')
plants <- plants[,!(names(plants) %in% .cols2rm)]
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 rows with excessive NA values
sel <- apply(plants, 1, function(x) sum(is.na(x))>=1 )
plants_trim <- plants[!(sel),]

# Convert Foliage_Color to a numeric variable in order to test relationships
```

```r
plants_trim$Color.factor <- as.numeric(factor(plants_trim$Foliage_Color,levels=c("Dark Green","Gray-Green",
plants_trim2 <- plants_trim
plants_trim2$Foliage_Color <- plants_trim$Color.factor

# Fit Foliage_Color with pH_Min and pH_Max, drop variables and re-test fit
fit=lm(Foliage_Color~pH_Min+pH_Max,data=plants_trim2)
fit2=lm(Foliage_Color~pH_Min,data=plants_trim2)
fit3=lm(Foliage_Color~pH_Max,data=plants_trim2)

# Try linear fits with log of the variables
fit4=lm(log(Foliage_Color)~log(pH_Max)+log(pH_Min),data=plants_trim2)
fit5=lm(log(Foliage_Color)~log(pH_Max),data=plants_trim2)
```

```r
### Car data
# Read data sets into R and merge by variables they have in common
dat1 <- read.csv("~/Desktop/Fall_2017_Courses/Statistical_Programming/STAT_5014_homework/STAT_5014_homework
dat2 <- read.csv(file="~/Desktop/Fall_2017_Courses/Statistical_Programming/STAT_5014_homework/STAT_5014_hom
dat3 <- read.csv(file="~/Desktop/Fall_2017_Courses/Statistical_Programming/STAT_5014_homework/STAT_5014_hom
plates <- merge(dat1,dat3,by="Kenteken")
defects <- merge(plates,dat2,by="Gebrek.identificatie")

# Attempt to handle dates; convert to character and remove all except ending yy element
substrRight <- function(x, n){substr(x, nchar(x)-n, nchar(x))}
defects$Datum.tenaamstelling <- as.character(defects$Datum.tenaamstelling)
defects$Datum.tenaamstelling <- substrRight(defects$Datum.tenaamstelling,1)
unique_makes <- aggregate(Merk ~ defects$'Datum.tenaamstelling', defects, function(x) length(unique(x)))
      # year 2017 has 71 unique makes
unique_models <- aggregate(Handelsbenaming ~ defects$'Datum.tenaamstelling', defects, function(x) length(un
      # year 2017 has 2746 unique models

# below I create a smaller data set with only 2017 data
defect17 <- defects[which(defects$`Datum.tenaamstelling` == '17'),]
 # Can re-do unique counts from subset data; get same results (71 and 2746)
unique_make <- length(unique(defect17$Merk))
unique_model <- length(unique(defect17$Handelsbenaming))

 # Most common 2017 defects (by ID number)
sort(table(defect17$`Gebrek.identificatie`),decreasing=TRUE)[1:5]
```

```
##
##  K04  AC1  G05  K05  RA2
## 1452  983  671  577  554
```