HW2_Williams_Jacob

Jacob Williams

August 30, 2018

Problem 4

Using version control allows for mistakes to be made. Meaning that if I make a mistake and accidently delete all my code I still have access to the previous version. Keeping track of each version of the code you write also allows us to take a okay analysis of data and try to do even better without worrying about losing the okay analysis. And lastly if a group is working on a project for SAIG for instance and multiple members are writing code, this will allow us to work together more integrated.

Problem 5

```r
url1 <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
Data1 <- read.csv(url(url1))
Data1 <- data.frame(Data1[-1, ])
colnames(Data1) <- c("Operator")
rownames(Data1) <- 1:30
Data1$Operator <- as.character(Data1$Operator)
# Data1$item <- rep(1:10,each=3)
Data1$Operator[c(1, 4, 7, 10, 13, 16, 19, 22, 25)] <- substr(as.character(Data1$Operator[c(1,
    4, 7, 10, 13, 16, 19, 22, 25)]), 3, 21)
Data1$Operator[c(28)] <- substr(as.character(Data1$Operator[c(28)]), 4,
    22)
Data1 <- data.frame(cbind(rep(1:10, each = 3), matrix(unlist(strsplit(Data1$Operator,
    " ")), nrow = 30, ncol = 5, byrow = T)))
colnames(Data1) <- c("Item", "Operator1", "Operator2", "Operator3", "Operator4",
    "Operator5")
Data1$Operator1 <- as.numeric(as.character(Data1$Operator1))
Data1$Operator2 <- as.numeric(as.character(Data1$Operator2))
Data1$Operator3 <- as.numeric(as.character(Data1$Operator3))
Data1$Operator4 <- as.numeric(as.character(Data1$Operator4))
Data1$Operator5 <- as.numeric(as.character(Data1$Operator5))
str(Data1)
```

```
## 'data.frame':    30 obs. of  6 variables:
##  $ Item     : Factor w/ 10 levels "1","10","2","3",..: 1 1 1 3 3 3 4 4 4 5 ...
##  $ Operator1: num  4.3 4.3 4.1 6 4.9 6 2.4 3.9 1.9 7.4 ...
##  $ Operator2: num  4.9 4.5 5.3 5.3 6.3 5.9 2.5 3 3.9 8.2 ...
##  $ Operator3: num  3.3 4 3.4 4.5 4.2 4.7 2.3 2.8 2.6 6.4 ...
##  $ Operator4: num  5.3 5.5 5.7 5.9 5.5 6.3 3.1 2.7 4.6 6.8 ...
##  $ Operator5: num  4.4 3.3 4.7 4.7 4.9 4.6 2.4 1.3 2.2 6 ...
```

The steps taken were as follow: (1) Import the data set, delete unwanted rows, reassign values to rownames, and change the lists into characters. (2) For the lists that had the variables 1:10 on them on had to remove the 1:10, this was accomplished using the substr function. (3) I then unlisted the lists by spaces, put it into a 30 x 5 matrix, and reassigned the 1:10 variables. (4) I then had to convert the columns back into the desired format.

```
url2 <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
Data2 <- read.csv(url(url2), sep = " ")
Data2 <- data.frame(cbind(unname(unlist(list(Data2[, c(1, 3, 5, 7)]))),
    unname(unlist(list(Data2[, c(2, 4, 6, 8)])))))
colnames(Data2) <- c("Year", "Long_Jump")
Data2$Year <- Data2$Year + 1900
Data2 <- Data2[-c(23, 24), ]
str(Data2)
```

```
## 'data.frame':    22 obs. of  2 variables:
##  $ Year     : num  1896 1900 1904 1908 1912 ...
##  $ Long_Jump: num  250 283 289 294 299 ...
```

(1) I read the data in using sep = ‘’ this allowed the columns to be able to split.
(2) I then could grab the desired columns, put them into a list, unlist them and I would have a vector of values I wanted.
(3) Threw the two vectors into a dataframe, made new column names, removed NA’s and added 1900 to the years to get an easier to read variable.

```
url3 <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
Data3 <- read.csv(url(url3), sep = " ")
Data3 <- data.frame(cbind(unname(unlist(list(Data3[, c(1, 3, 5)]))), unname(unlist(list(Data3[,
    c(2, 4, 6)])))))
colnames(Data3) <- c("Body_Weight", "Brain_Weight")
Data3 <- Data3[-c(63), ]
str(Data3)
```

```
## 'data.frame':    62 obs. of  2 variables:
##  $ Body_Weight : num  3.38 0.48 1.35 465 36.33 ...
##  $ Brain_Weight: num  44.5 15.5 8.1 423 119.5 ...
```

(1) I read the data in using sep = ‘’ this allowed the columns to be able to split.
(2) I then could grab the desired columns, put them into a list, unlist them and I would have a vector of values I wanted.
(3) Threw the two vectors into a dataframe, made new column names, and removed NA’s.
(4) Same as above

```
url4 <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
Data4 <- read.delim(url(url4))
Data4$Column <- as.character(Data4$X.this.needs.reformatting.to.read.into.Splus)
Data4$TenThousand <- NA
Data4$TwentyThousand <- NA
Data4$ThirtyThousand <- NA
Data4 <- Data4[-c(1), -c(1)]
Data4$Column[1] <- gsub("  ", "", Data4$Column[1])
Data4$Column[2] <- gsub("  ", "", Data4$Column[2])
Column1 <- as.numeric(unlist(strsplit(unlist(strsplit(Data4$Column, " "))[-c(1,
    5)], ",")))
Column2 <- rep(rep(c(10000, 20000, 30000), each = 3), 2)
Column3 <- rep(c("IFE", "PusaEarlyDwarf"), each = 9)
Data4 <- data.frame(cbind(Column1, Column2, Column3))
colnames(Data4) <- c("Data", "Density", "Varieties")
Data4$Data <- as.numeric(as.character(Data4$Data))
str(Data4)
```

```
## 'data.frame':    18 obs. of  3 variables:
```

```
##  $ Data     : num  16.1 15.3 17.5 16.6 19.2 18.5 20.8 18 21 8.1 ...
##  $ Density  : Factor w/ 3 levels "10000","20000",..: 1 1 1 2 2 2 3 3 3 1 ...
##  $ Varieties: Factor w/ 2 levels "IFE","PusaEarlyDwarf": 1 1 1 1 1 1 1 1 1 2 ...
```

(1) Import data, create empty columns, put column in the right format, remove unwanted values.
(2) Seperate lists of values by " “, split by spaces, unlist the string, then remove first 5 elements of vector, the string split each element by comma, then unlist again, this gets the data.
(3) Create two categorical variables, combine the three vectors, rename columns in data frame, then make the data numeric.

This problem would have been more easily solved by simply typing out the matrix into vectors, to little data for this much manipulation.

Problem 6

```r
library(swirl)
```

```
## Warning: package 'swirl' was built under R version 3.3.3
```

```r
# Path to data
.datapath <- file.path(path.package("swirl"), "Courses", "R_Programming_E",
    "Looking_at_Data", "plant-data.txt")
# Read in data
plants <- read.csv(.datapath, strip.white = TRUE, na.strings = "")

# Remove annoying columns
.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")
frt <- vector()
for (i in 1:nrow(plants)) {
    frt[i] <- ifelse(is.na(plants$pH_Min[i]) | is.na(plants$pH_Max[i]) |
        is.na(plants$Foliage_Color[i]), NA, i)
}
plants <- plants[frt[!is.na(frt)], -c(1, 2, 3, 7, 8, 9, 10)]
rm(frt)
plants$average_ph <- (plants$pH_Min + plants$pH_Max)/2

lm1 <- lm(average_ph ~ Foliage_Color, data = plants)
library(knitr)
kable(lm1$coefficients, caption = "Table of Coefficients")
```

Table 1: Table of Coefficients

	x
(Intercept)	5.9993902
Foliage_ColorGray-Green	0.4126098
Foliage_ColorGreen	0.1847138
Foliage_ColorRed	0.1631098
Foliage_ColorWhite-Gray	0.4450542

	x
Foliage_ColorYellow-Green	-0.0618902

```r
kable(anova(lm1), caption = "Anova Table")
```

Table 2: Anova Table

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Foliage_Color	5	5.747631	1.1495262	3.958224	0.0014895
Residuals	826	239.882486	0.2904146	NA	NA