Homework 2

Tessa Anwyll

August 31, 2018

Homework 2

I have neither given nor recieved any unauthorized assistance on this assignment

Problem 4

One year in college I had a lengthy, multi-part final project for a class that counted as a large part of my semester grade. I spent a ton of time working on it and had only one part left to do. When I went to finish that one part, the entire rest of the project was gone! I spent an hour digging through my hard drives and trying to recover files to see where it had gone, but I couldn't find it. I ended up staying up all night until class the next day when it was due re-doing the entire thing. Because it had been on the computer, I had very limited notes to work off of so while I had figured everything out, all the formatting and retyping still took a long time. I wish I known about version control before this, if only for situations like these! Additionally, it seems really easy for collaboration, fixing/finding issues and returning to an old version when your "updates" turn out to be awful and cause everything to break. It seems like it really can provide some peace of mind and increase workflow since you don't necessarily have to spend hours digging through code to find the one thing that you changed that is causing everything to mess up; you could find it, or just revert to the last version that worked and start back over from there. Plus, if my computer is stolen or destroyed by dropping it, spilling something on it or having some other unfortunate event happen to it, my work will not be lost since it is backed up in many places, not just locally.

Problem 5

A

```r
# Read in the data as a list so I can split it up by row
tableA<-(readLines("https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"))
```

```
## Warning in readLines("https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/
## data/Sensory.dat"): incomplete final line found on 'https://
## www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat'
```

```r
# Split into individual data values
tableA<-strsplit(tableA, split = " ")

# Cut off first two vectors in table (they're just labels). So for each row there should be one value p
matA<-tableA[3:length(tableA)]
dfnames<-vector()
for(i in 1:length(matA)){

  if(length(matA[[i]])==6){
   dfnames<-c(dfnames, (matA[[i]][1]))
  matA[[i]]<-c(matA[[i]][2:6])
  }
}
```

```r
# Convert to 10x5 matrix then data frame and format (Adding row and column names and making columns the
aDf<- matrix(unlist(matA), nrow = 10, byrow = TRUE, ncol=5)
rownames(aDf)<-dfnames
colnames(aDf)<-c("Operator 1", "Operator 2", "Operator 3", "Operator 4", "Operator 5")

aDf<-data.frame(aDf)

aDf<-rownames_to_column(aDf, "Item")
aDf<-gather(aDf, key = Operator, value = Measurement, Operator.1, Operator.2, Operator.3, Operator.4, Op
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```

```r
aDf[,3]<-as.numeric(aDf[,3])
aDf<-group_by(aDf, Item)
aDf<-arrange(aDf, Item)
aDf
```

```
## # A tibble: 50 x 3
## # Groups:   Item [10]
##    Item  Operator   Measurement
##    <chr> <chr>            <dbl>
##  1 1     Operator.1         4.3
##  2 1     Operator.2         4.9
##  3 1     Operator.3         3.3
##  4 1     Operator.4         5.3
##  5 1     Operator.5         4.4
##  6 10    Operator.1         7.4
##  7 10    Operator.2         8.2
##  8 10    Operator.3         6.4
##  9 10    Operator.4         6.8
## 10 10    Operator.5         6
## # ... with 40 more rows
```

B

```r
# Read in the data
tableB<-read.csv("http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat")

# Look at the data
head(tableB)
```

```
##   Year.Long.Jump.Year.Long.Jump.Year.Long.Jump.Year.Long.Jump
## 1                     -4 249.75 24 293.13 56 308.25 80 336.25
## 2                      0 282.88 28 304.75 60 319.75 84 336.25
## 3                      4 289.00 32 300.75 64 317.75 88 343.25
## 4                      8 294.50 36 317.31 68 350.50 92 342.50
## 5                               12 299.25 48 308.00 72 324.50
## 6                               20 281.50 52 298.00 76 328.50
```

```r
## Separate the data into columns
tableB1<-separate(tableB, Year.Long.Jump.Year.Long.Jump.Year.Long.Jump.Year.Long.Jump, into = c("Y1", "Y
```

```
## Warning: Expected 8 pieces. Missing pieces filled with `NA` in 2 rows [5,
```

```
## 6].
```

```r
## Consolidate all year and long jump values into two variables and filter out NA
year<-as.vector(as.numeric(c(tableB1[,1], tableB1[,3], tableB1[,5], tableB1[,7])))
LongJump<-as.vector(as.numeric(c(tableB1[,2], tableB1[,4], tableB1[,6], tableB1[,8])))
matB1<-cbind(tableB1, year, LongJump)
tableB1<-select(matB1, year, LongJump)
tableB1<-filter(tableB1, !is.na(tableB1$year)&!is.na(tableB1$LongJump))

## Convert years since 1900 into years
tableB1<-mutate(tableB1, Year = year+1900)
tableB1<-select(tableB1, Year, LongJump)
colnames(tableB1)<- c("Years (years)", "Long Jump (in)")
tableB1
```

```
##    Years (years) Long Jump (in)
## 1           1896         249.75
## 2           1900         282.88
## 3           1904         289.00
## 4           1908         294.50
## 5           1912         299.25
## 6           1920         281.50
## 7           1924         293.13
## 8           1928         304.75
## 9           1932         300.75
## 10          1936         317.31
## 11          1948         308.00
## 12          1952         298.00
## 13          1956         308.25
## 14          1960         319.75
## 15          1964         317.75
## 16          1968         350.50
## 17          1972         324.50
## 18          1976         328.50
## 19          1980         336.25
## 20          1984         336.25
## 21          1988         343.25
## 22          1992         342.50
```

C

```r
#Read in data as a list using fread
dataC<-fread("https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat")
```

```
## Warning in fread("https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/
## BrainandBodyWeight.dat"): Detected 12 column names but the data has 6
## columns. Filling rows automatically. Set fill=TRUE explicitly to avoid this
## warning.
```

```r
#Assign temporary column names to columns with values we want to keep and get rid of columns that are a
colnames(dataC)<-c("BodyWeight1", "BrainWeight1", "BodyWeight2", "BrainWeight2", "BodyWeight3", "BrainWe
dataC<-select(dataC, BodyWeight1, BodyWeight2, BodyWeight3, BrainWeight1, BrainWeight2, BrainWeight3)

#Stack all body weight and all brain weight vectors into one body weight and one brain weight vector
BodyWeight<-stack(as.vector(dataC[,c(1,2,3)]))
```

```r
BrainWeight<-stack(as.vector(dataC[,c(4,5,6)]))
BodyWeight<-BodyWeight[,1]
BrainWeight<-BrainWeight[,1]

# Coerce into a data frame
dataCFinal<-data.frame(BodyWeight,BrainWeight)

#Remove any rows that contain NA values
dataCFinal<-filter(dataCFinal, !is.na(dataCFinal$BodyWeight)|!is.na(dataCFinal$BrainWeight) )

dataCFinal
```

```
##    BodyWeight BrainWeight
## 1       3.385       44.50
## 2       0.480       15.50
## 3       1.350        8.10
## 4     465.000      423.00
## 5      36.330      119.50
## 6      27.660      115.00
## 7      14.830       98.20
## 8       1.040        5.50
## 9       4.190       58.00
## 10      0.425        6.40
## 11      0.101        4.00
## 12      0.920        5.70
## 13      1.000        6.60
## 14      0.005        0.10
## 15      0.060        1.00
## 16      3.500       10.80
## 17      2.000       12.30
## 18      1.700        6.30
## 19   2547.000     4603.00
## 20      0.023        0.30
## 21    187.100      419.00
## 22    521.000      655.00
## 23      0.785        3.50
## 24     10.000      115.00
## 25      3.300       25.60
## 26      0.200        5.00
## 27      1.410       17.50
## 28    529.000      680.00
## 29    207.000      406.00
## 30     85.000      325.00
## 31      0.750       12.30
## 32     62.000     1320.00
## 33   6654.000     5712.00
## 34      3.500        3.90
## 35      6.800      179.00
## 36     35.000       56.00
## 37      4.050       17.00
## 38      0.120        1.00
## 39      0.023        0.40
## 40      0.010        0.30
## 41      1.400       12.50
```

```
## 42    250.000      490.00
## 43      2.500       12.10
## 44     55.500      175.00
## 45    100.000      157.00
## 46     52.160      440.00
## 47     10.550      179.50
## 48      0.550        2.40
## 49     60.000       81.00
## 50      3.600       21.00
## 51      4.288       39.20
## 52      0.280        1.90
## 53      0.075        1.20
## 54      0.122        3.00
## 55      0.048        0.33
## 56    192.000      180.00
## 57      3.000       25.00
## 58    160.000      169.00
## 59      0.900        2.60
## 60      1.620       11.40
## 61      0.104        2.50
## 62      4.235       50.40
```

D

```r
# Read in data as a list using fread. Yields a 2x3
dataD<-fread("http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat", sep = " ")
```

```
## Warning in fread("http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/
## tomato.dat", : Detected 3 column names but the data has 4 columns (i.e.
## invalid file). Added 1 extra default column name for the first column which
## is guessed to be row names or an index. Use setnames() afterwards if this
## guess is not correct, or fix the file write command that created the file
## to create a valid file.
```

```r
# Separate each column into 3 columns, giving a 2x9
dataDSep<-separate(dataD, "10000", into = c("10000A", "10000B", "10000C"), sep = ",", convert = TRUE)
```

```
## Warning: Expected 3 pieces. Additional pieces discarded in 1 rows [2].
```

```r
dataDSep<-separate(dataDSep, "20000", into = c("20000A", "20000B", "20000C"), sep = ",", convert = TRUE
dataDSep<-separate(dataDSep, "30000", into = c("30000A", "30000B", "30000C"), sep = ",", convert = TRUE

# Gather each set of 3 pieces at density level into one column for each level
density1<- gather(dataDSep,key="ABC",value="10000","10000A":"10000C")
density1<- select(density1, "V1", "10000")
density2<- gather(dataDSep,key="ABC",value="20000","20000A":"20000C")
density2<- select(density2, "20000")
density3<- gather(dataDSep,key="ABC",value="30000","30000A":"30000C")
density3<- select(density3, "30000")
dataDGather<-cbind(density1, density2, density3)

# Combine each density column into one variable and make labels for each one, then group and sort by ty
dataDGather<- gather(dataDGather,key="Plant_Density",value="Yield","10000", "20000", "30000")
colnames(dataDGather)<- c("Tomato_Type", "Plant_Density", "Yield")
```

```
dataDGather<-group_by(dataDGather, Tomato_Type)
dataDGather<-arrange(dataDGather, Tomato_Type)

# Converting density from string to double
odd<-vector()

  for(i in 1:nrow(dataDGather)){
  if(dataDGather[i,2]=="10000"){
  odd[i]<-10000}
  else{
    if(dataDGather[i,2]=="20000"){
      odd[i]<-20000}
    else{
      if(dataDGather[i,2]=="30000"){
      odd[i]<-30000}
    }
  }
  }
dataDGather[,2]<-odd
dataDGather
```

```
## # A tibble: 18 x 3
## # Groups:   Tomato_Type [2]
##    Tomato_Type    Plant_Density Yield
##    <chr>                  <dbl> <dbl>
##  1 "Ife\\#1"              10000  16.1
##  2 "Ife\\#1"              10000  15.3
##  3 "Ife\\#1"              10000  17.5
##  4 "Ife\\#1"              20000  16.6
##  5 "Ife\\#1"              20000  19.2
##  6 "Ife\\#1"              20000  18.5
##  7 "Ife\\#1"              30000  20.8
##  8 "Ife\\#1"              30000  18
##  9 "Ife\\#1"              30000  21
## 10 PusaEarlyDwarf         10000   8.1
## 11 PusaEarlyDwarf         10000   8.6
## 12 PusaEarlyDwarf         10000  10.1
## 13 PusaEarlyDwarf         20000  12.7
## 14 PusaEarlyDwarf         20000  13.7
## 15 PusaEarlyDwarf         20000  11.5
## 16 PusaEarlyDwarf         30000  14.4
## 17 PusaEarlyDwarf         30000  15.4
## 18 PusaEarlyDwarf         30000  13.7
```

Problem 6

Import the data and take a look at what we are working with:

```
# Look at the data
head(plants)
```

```
##                Scientific_Name          Duration Active_Growth_Period
## 1                  Abelmoschus              <NA>                 <NA>
## 2       Abelmoschus esculentus Annual, Perennial                 <NA>
```

```
## 3                        Abies              <NA>                   <NA>
## 4                Abies balsamea         Perennial    Spring and Summer
## 5 Abies balsamea var. balsamea         Perennial                   <NA>
## 6                     Abutilon              <NA>                   <NA>
##   Foliage_Color pH_Min pH_Max Precip_Min Precip_Max Shade_Tolerance
## 1          <NA>     NA     NA         NA         NA            <NA>
## 2          <NA>     NA     NA         NA         NA            <NA>
## 3          <NA>     NA     NA         NA         NA            <NA>
## 4         Green      4      6         13         60        Tolerant
## 5          <NA>     NA     NA         NA         NA            <NA>
## 6          <NA>     NA     NA         NA         NA            <NA>
##   Temp_Min_F
## 1         NA
## 2         NA
## 3         NA
## 4        -43
## 5         NA
## 6         NA
```

The biggest issue is all the "NA"s. My next move is to remove each row that does not have NAs in any of my variables I will be testing (so no NAs for Foliage Color or either of the pH measures)

```r
library(dplyr)
# Store new data frame to use in my test with only rows with no "NA"s for my variables of interest
newplants<-filter(plants,!is.na(plants$pH_Min)&!is.na(plants$pH_Max)&!is.na(plants$Foliage_Color))

# Create a new variable that takes the average of the min and max pH reported in the data set
newplants<-mutate(newplants, pHAvg = (pH_Min+pH_Max)/2)


# Group by foliage color, my predictor
newplants<-group_by(newplants, Foliage_Color)

# Take a look at the data set now
head(newplants)
```

```
## # A tibble: 6 x 11
## # Groups:   Foliage_Color [1]
##   Scientific_Name Duration Active_Growth_P~ Foliage_Color pH_Min pH_Max
##   <fct>           <fct>    <fct>            <fct>          <dbl>  <dbl>
## 1 Abies balsamea  Perenni~ Spring and Summ~ Green            4      6
## 2 Acacia constri~ Perenni~ Spring and Summ~ Green            7      8.5
## 3 Acalypha virgi~ Annual   Spring, Summer,~ Green            5.9    7
## 4 Acer negundo    Perenni~ Spring and Summ~ Green            5      7.8
## 5 Acer nigrum     Perenni~ Spring and Summ~ Green            4.5    7.3
## 6 Acer pensylvan~ Perenni~ Spring and Summ~ Green            4.4    6.5
## # ... with 5 more variables: Precip_Min <int>, Precip_Max <int>,
## #   Shade_Tolerance <fct>, Temp_Min_F <int>, pHAvg <dbl>
```

```r
# Make a table just to display that summarizes the means of each average ph and counts for each foliage
dispnewplants <- summarize(newplants, n(), mean(pHAvg))
dispnewplants
```

```
## # A tibble: 6 x 3
##   Foliage_Color `n()` `mean(pHAvg)`
##   <fct>         <int>         <dbl>
```

```
## 1 Dark Green       82          6.00
## 2 Gray-Green       25          6.41
## 3 Green           692          6.18
## 4 Red               4          6.16
## 5 White-Gray        9          6.44
## 6 Yellow-Green     20          5.94
```

```r
# Store regression model in a new variable
ANOVAtestdata<-lm(pHAvg~Foliage_Color, newplants)
RGC<-coefficients(ANOVAtestdata)
RGC<-as.list(RGC)

# Run ANOVA Test
ANOVAResults<-anova(ANOVAtestdata)
a = list(RGC, ANOVAResults)
# Bind regression results and ANOVA results into one table
FinalTable<-data.frame(rbindlist(a, use.names=FALSE, fill = TRUE, idcol = NULL))
```

```
## Warning in rbindlist(a, use.names = FALSE, fill = TRUE, idcol = NULL):
## Resetting 'use.names' to TRUE. 'use.names' can not be FALSE when
## 'fill=TRUE'.
```

```r
rownames(FinalTable)<-c("Coefficient", "Foliage Color (Predictor)", "Residual")
FinalTable
```

```
##                           X.Intercept. Foliage_ColorGray.Green
## Coefficient                    5.99939               0.4126098
## Foliage Color (Predictor)           NA                      NA
## Residual                            NA                      NA
##                           Foliage_ColorGreen Foliage_ColorRed
## Coefficient                        0.1847138        0.1631098
## Foliage Color (Predictor)                 NA               NA
## Residual                                  NA               NA
##                           Foliage_ColorWhite.Gray
## Coefficient                             0.4450542
## Foliage Color (Predictor)                      NA
## Residual                                       NA
##                           Foliage_ColorYellow.Green  Df     Sum.Sq
## Coefficient                             -0.06189024  NA         NA
## Foliage Color (Predictor)                        NA   5   5.747631
## Residual                                         NA 826 239.882486
##                             Mean.Sq  F.value      Pr..F.
## Coefficient                      NA       NA          NA
## Foliage Color (Predictor) 1.1495262 3.958224 0.001489531
## Residual                  0.2904146       NA          NA
```

With a p value (0.0015) so much lower than my significance level (.05), we can concluded that there is a statistically significant relationshipe between foliage color and pH.