HW2_Salem

Mohamed Salem

September 8, 2019

Problem 3 - How Can Version Control Help Me in the Classroom

Version control would help in: 1) in collaborations I'd like to do in class with my colleagues, where we would all have access to the code, being able to edit and review it in addition to being aware of who is working on what; 2) having the ability to compare and pinpoint exact differences between two or more versions of code without having to manually save versions using different files under different names; 3) picking up where we previously left off in class by knowing the latest working version; 4) being able to experiment with new features or personlizations without impacting the original source code

Problem 4 - Importing, Munging, Cleaning, and Summarising

```
# The next line specifies the url which the data will be imported from
url1 <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
```

```
# This creates a vector to hold the names available in the header of the data
name_placeholder <- read.table(url1, skip = 1, nrow = 1, stringsAsFactors = FALSE, sep = " ")

# This reads data from the source table after skipping the first line which is the header
Sensory_data_from_five_operators <- read.table(url1, fill = T, header = T, skip = 1, na.string = c("",
    "null", "NaN"), stringsAsFactors = FALSE, sep = " ")

# This applies our previously stored names to the imported dataframe
names(Sensory_data_from_five_operators) <- name_placeholder

# This creates a new matrix with number of rows and columns equal to that in the imported
# data with the same header names, and using modular arithmetic, adjusts the jagged-entry
# of the data based o row number, as the jagged pattern of data entry occurs in intervals
# of 3
```

```
Sensory_data_from_five_operators_cleaned <- data.frame(matrix(0,
    nrow = length(Sensory_data_from_five_operators$Item), ncol = length(Sensory_data_from_five_operators)))
```

```
names(Sensory_data_from_five_operators_cleaned) <- name_placeholder
for (i in 1:length(Sensory_data_from_five_operators$Item)) {
    for (j in 2:6) {
        if ((as.numeric(row.names(Sensory_data_from_five_operators[i, ])) + 2)%%3 != 0) {
            Sensory_data_from_five_operators_cleaned[i, j] <- Sensory_data_from_five_operators[i,
                j - 1]
            Sensory_data_from_five_operators_cleaned[i, 1] <- Sensory_data_from_five_operators[i -
                (i - 1)%%3, 1]
        } else {
            Sensory_data_from_five_operators_cleaned[i, j] <- Sensory_data_from_five_operators[i,
                j]
            Sensory_data_from_five_operators_cleaned[i, 1] <- Sensory_data_from_five_operators[i,
                1]
        }
    }
}
```

```
# This creates a new dataframe that combines data from all five operators into one column
# and creates an operator identifier column
```

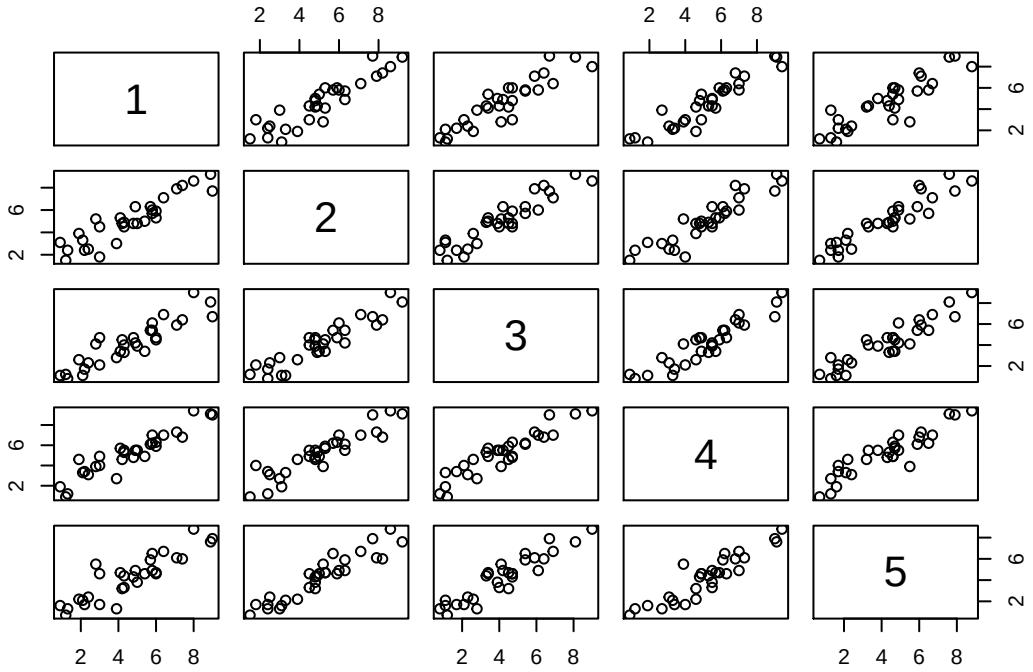
```
Sensory_data_from_five_operators_cleaned_gathered <- gather(Sensory_data_from_five_operators_cleaned,
    key = "Operator", value = "Item")
attach(Sensory_data_from_five_operators_cleaned)
```

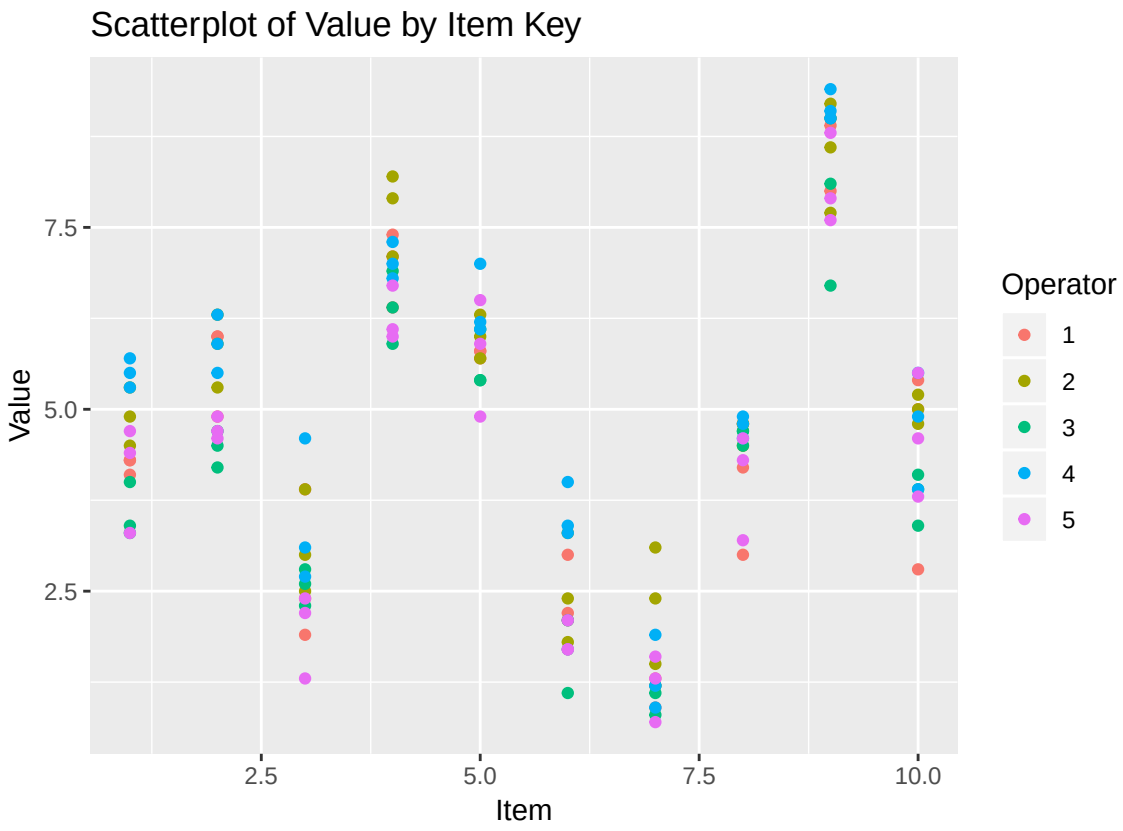
```
Sensory_data_from_five_operators_cleaned_gathered <- Sensory_data_from_five_operators_cleaned_gathered %>%
    cbind(cbind(c(Item, Item, Item, Item, Item)))
```

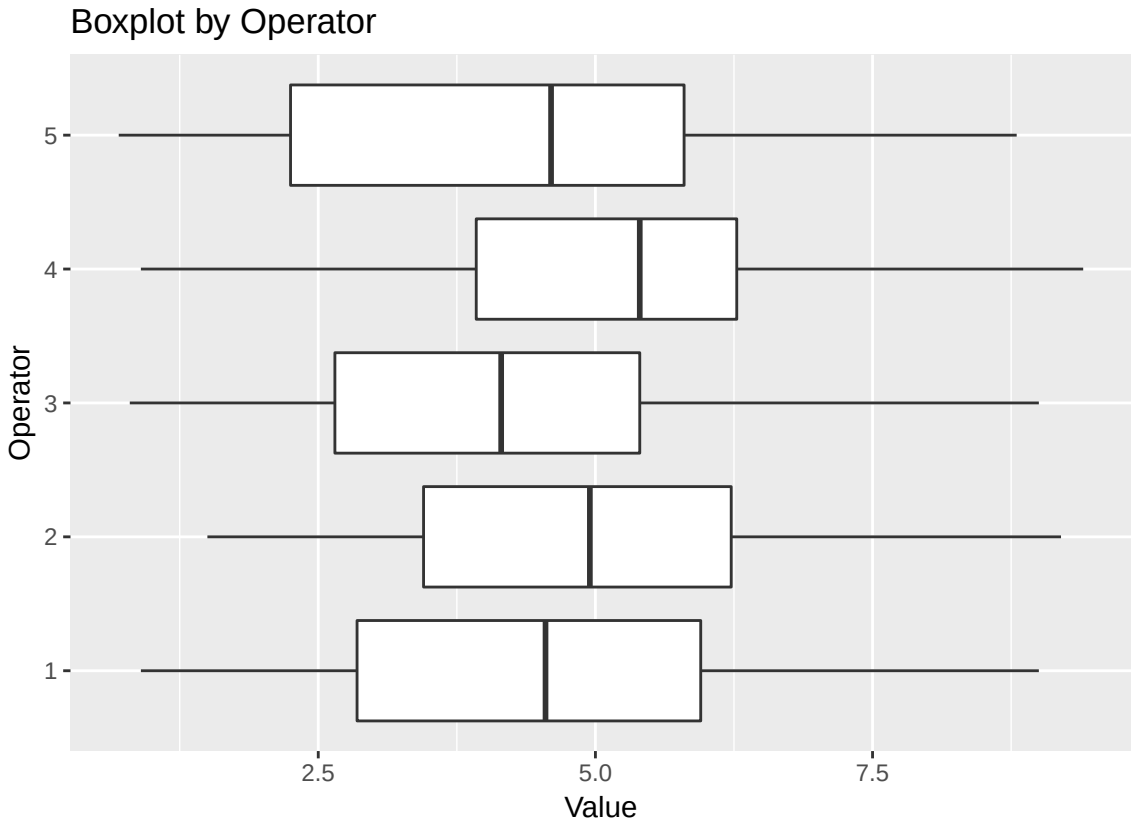
```
names(Sensory_data_from_five_operators_cleaned_gathered) <- c("Operator", "Value", "Item")

# Finally we create a summary table of our data grouped by operator...
summarise(group_by(Sensory_data_from_five_operators_cleaned_gathered, Operator), count = n(),
    mean(Value, na.rm = TRUE))
```

```
## # A tibble: 5 x 3
##   Operator count `mean(Value, na.rm = TRUE)`
##   <chr>    <int>                       <dbl>
## 1 1           30                        4.59
## 2 2           30                        5.06
## 3 3           30                        4.17
## 4 4           30                        5.19
## 5 5           30                        4.27
```

Scatterplot of Operators



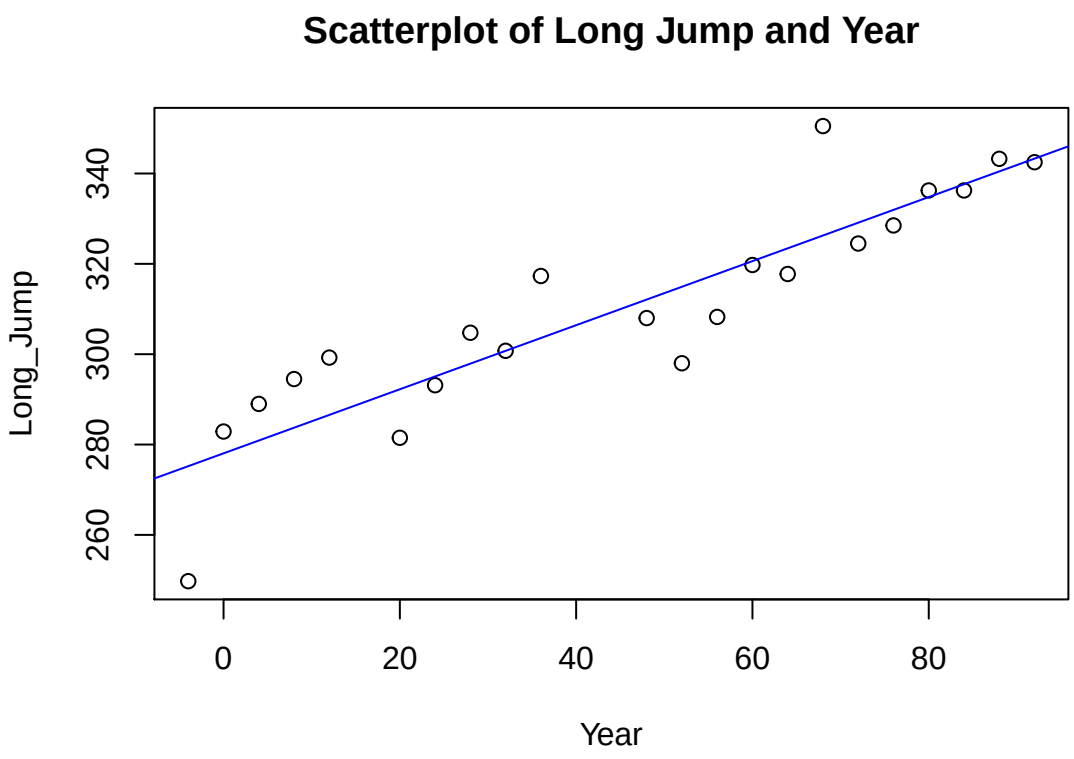





```
# The next line specifies the url which the data will be imported from
url2 <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
```

```
# This reads data from the source table
Gold_medal_performance_long_jump <- read.table(url2, fill = T, header = T, na.string = c("",
    "null", "NaN"), stringsAsFactors = FALSE, sep = " ")

# This creates a vector to hold the names we want to use as headers of the data
name_placeholder2 <- c("Year", "Long_Jump")

# This creates a new matrix with number of rows and columns equal to the number of
# obsesrvations and parameters (Years, Long Jump), we then use modular arithmetic to cycle
# through the columns in rounds of six, since the data is arranged to be recycled across
# 6x2 matrices
Gold_medal_performance_long_jump_cleaned <- data.frame(matrix(0, nrow = 22, ncol = 2))
names(Gold_medal_performance_long_jump_cleaned) <- name_placeholder2
for (i in 1:22) {
    for (j in 1:2) {
        x <- 6 * as.numeric(i%%6 == 0) + i%%6
        y <- 2 * ((i - 1)%/%6) + j
        Gold_medal_performance_long_jump_cleaned[i, j] <- Gold_medal_performance_long_jump[x,
            y]
    }
}
# Finally we create a summary table of our data ...
summary(Gold_medal_performance_long_jump_cleaned$Long_Jump)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   249.8   295.4   308.1   310.3   327.5   350.5
```




```
# The next line specifies the url which the data will be imported from
url3 <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
```

```r
# This reads data from the source table
Brain_weight_and_body_weight <- read.table(url3, fill = T, header = T, na.string = c("", "null",
    "NaN"), stringsAsFactors = FALSE, sep = " ")

# This creates a vector to hold the names we want to use as headers of the data
name_placeholder3 <- c("Body_Wt_kg", "Brain_Wt_g")

# This creates a new matrix with number of rows and columns equal to the number of
# obsesrvations and parameters (Brain Weight in g's, Body Weight in kg's), we then use
# modular arithmetic to cycle through the columns in rounds of twenty one, since the data
# is arranged to be recycled across 21x2 matrices
Brain_weight_and_body_weight_cleaned <- data.frame(matrix(0, nrow = 62, ncol = 2))
names(Brain_weight_and_body_weight_cleaned) <- name_placeholder3
for (i in 1:62) {
    for (j in 1:2) {
        x <- 21 * as.numeric(i%%21 == 0) + i%%21
        y <- 2 * ((i - 1)%/%21) + j
        Brain_weight_and_body_weight_cleaned[i, j] <- Brain_weight_and_body_weight[x, y]
    }
}
```

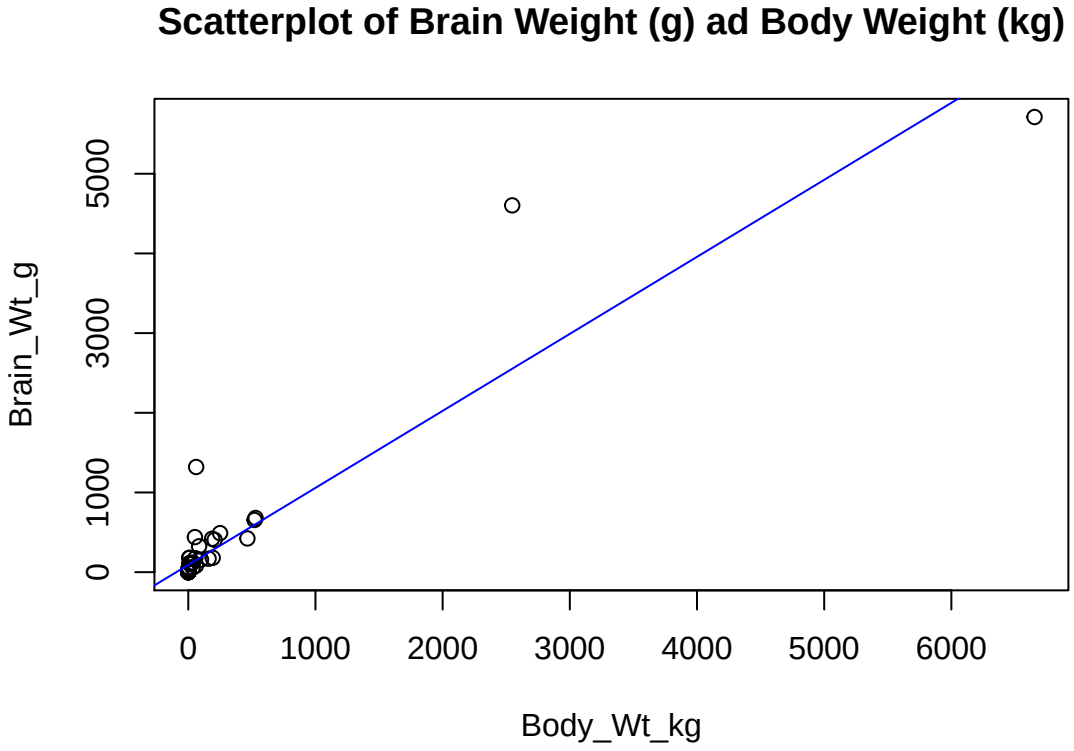
```r
Brain_weight_and_body_weight_cleaned <- mutate(Brain_weight_and_body_weight_cleaned,
    Body_Wt_g = Brain_weight_and_body_weight_cleaned$Body_Wt_kg *
        1000)
```

```r
# Finally we create a summary table of our data ...
summary(Brain_weight_and_body_weight_cleaned$Body_Wt_kg)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    0.005    0.600    3.342  198.790   48.203 6654.000
```

```r
summary(Brain_weight_and_body_weight_cleaned$Brain_Wt_g)
```

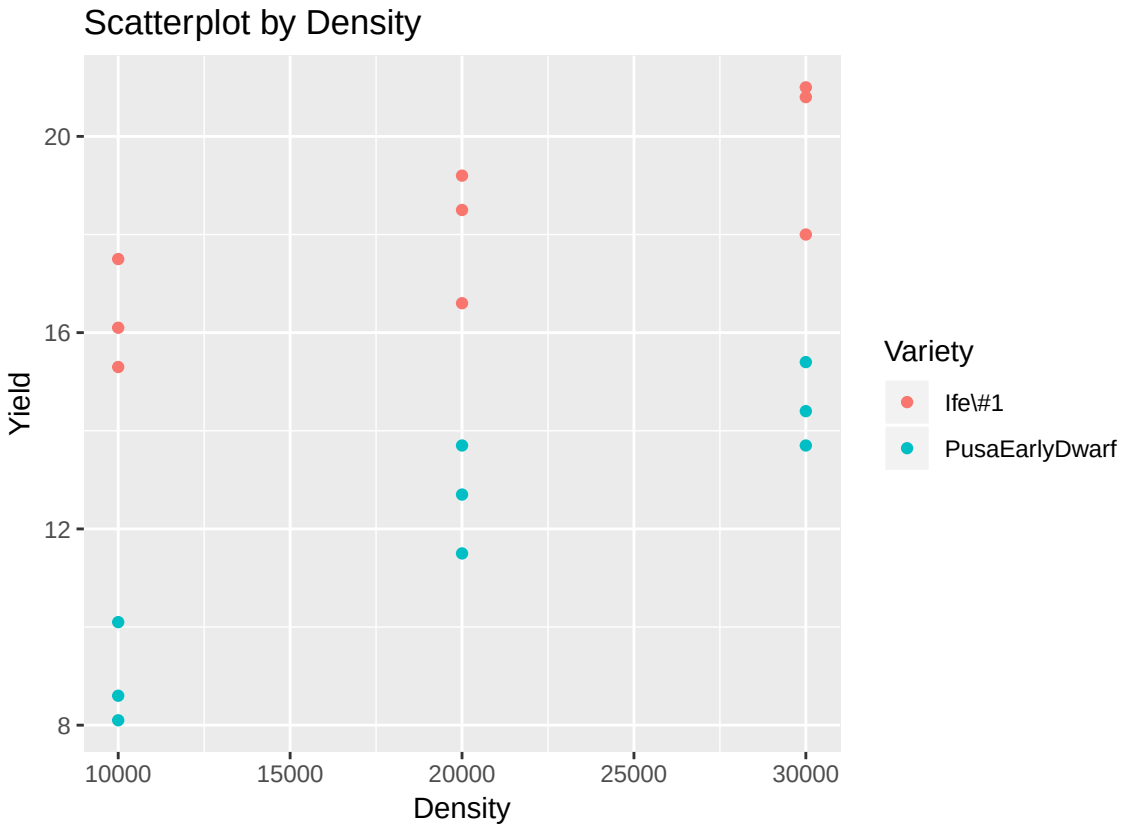
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.10    4.25   17.25  283.13  166.00 5712.00
```




```
# The next line specifies the url which the data will be imported from
url4 <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
```

```
# This reads data from the source table
Triplicate_measurements_of_tomato_yield <- read.table(url4, sep = ";")

# This creates a vector to hold the names we want to use as headers of the data
name_placeholder4 <- c("Variety", "Density", "Yield")

# This reads data from the source table
firstrow1 <- read.table(url4, skip = 2, nrow = 1, quote = "", comment.char = "", stringsAsFactors = T,
    sep = c(" ", ";"))
secondrow1 <- read.table(url4, skip = 3, nrow = 1, stringsAsFactors = FALSE, sep = c(" ", ";"))

# This creates a new matrix with number of rows and columns equal to the number of
# obsesrvations and parameters (Type, Density, Yield), we then use modular arithmetic to
# cycle through the data and fill our constructed matrix
Triplicate_measurements_of_tomato_yield_cleaned <- data.frame(matrix(0, nrow = 18, ncol = 3))
names(Triplicate_measurements_of_tomato_yield_cleaned) <- name_placeholder4
Triplicate_measurements_of_tomato_yield_cleaned[1:6, 2] <- 10000
Triplicate_measurements_of_tomato_yield_cleaned[7:12, 2] <- 20000
Triplicate_measurements_of_tomato_yield_cleaned[13:18, 2] <- 30000
for (i in 1:18) {
    if (i%%2 != 0) {
        Triplicate_measurements_of_tomato_yield_cleaned[i, 1] <- as.character(firstrow1[1,
            1])
    } else {
        Triplicate_measurements_of_tomato_yield_cleaned[i, 1] <- secondrow1[1, 1]
    }
}
phlist1 <- as.data.frame(strsplit(as.character(paste(firstrow1[1, 12], ",", firstrow1[1, 15],
```

```
    ",", firstrow1[1, 18])), split = ","))
names(phlist1) <- c("value")
phlist1 <- as.numeric(as.character(phlist1$value))
phlist2 <- as.data.frame(strsplit(as.character(paste(secondrow1[1, 4], secondrow1[1, 8], ",",
    secondrow1[1, 11])), split = ","))
names(phlist2) <- c("value")
phlist2 <- as.numeric(as.character(phlist2$value))

for (i in 1:18) {
    if (i%%2 != 0) {
        Triplicate_measurements_of_tomato_yield_cleaned[i, 3] <- phlist1[(i%/%2) + 1]
    } else {
        Triplicate_measurements_of_tomato_yield_cleaned[i, 3] <- phlist2[i/2]
    }
}
```



Problem 5 - Guidelines and Challenges for Reproducible Research

```
# 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")
```

```
# Creating a variable that holds the midpoint of the range
plants_cleaned <- filter(plants, !is.na(pH_Min), !is.na(pH_Max), !is.na(Foliage_Color))
plants_cleaned <- plants_cleaned %>% group_by(Scientific_Name) %>% mutate(pH_median = (pH_Max -
    pH_Min)/2 + pH_Min, foliage_color_coded = as.factor(Foliage_Color))

# Creating a set of dummy variables for the linear regression
plants_cleaned_dummies <- dummy_columns(plants_cleaned, select_columns = c("foliage_color_coded"))
plants_cleaned_dummies_only <- plants_cleaned_dummies[, 11:18]
```

```
# Running a linear regression analysis
reg <- lm(pH_median ~ . - foliage_color_coded - foliage_color_coded_Green,
    data = plants_cleaned_dummies_only)
```

```
# Producing a summary of the regression analysis
summary(reg)
```

```
##
## Call:
## lm(formula = pH_median ~ . - foliage_color_coded - foliage_color_coded_Green,
##     data = plants_cleaned_dummies_only)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.63750 -0.33410  0.00061  0.31590  2.01590
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        6.18410    0.02049 301.870  < 2e-16
## `foliage_color_coded_Yellow-Green` -0.24660    0.12223  -2.018  0.04396
## `foliage_color_coded_Dark Green`   -0.18471    0.06294  -2.935  0.00343
## `foliage_color_coded_White-Gray`    0.26034    0.18080   1.440  0.15026
## `foliage_color_coded_Gray-Green`    0.22790    0.10971   2.077  0.03809
## foliage_color_coded_Red            -0.02160    0.27023  -0.080  0.93630
##
## (Intercept)                        ***
## `foliage_color_coded_Yellow-Green` *
## `foliage_color_coded_Dark Green`   **
## `foliage_color_coded_White-Gray`
## `foliage_color_coded_Gray-Green`   *
## foliage_color_coded_Red
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5389 on 826 degrees of freedom
## Multiple R-squared:  0.0234, Adjusted R-squared:  0.01749
## F-statistic: 3.958 on 5 and 826 DF,  p-value: 0.00149
```

```
# running an ANOVA analysis
anv <- aov(pH_median ~ foliage_color_coded, data = plants_cleaned)

# Producing a summary of the ANOVA analysis
summary(anv)
```

```
##                      Df Sum Sq Mean Sq F value  Pr(>F)
## foliage_color_coded   5   5.75  1.1495   3.958 0.00149 **
## Residuals           826 239.88  0.2904
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```