Tidy your data

Colby S-P

2023-10-24

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(readr)
library(ggplot2)
```

Question 1

Part A

```
# get url for the data
url_part_A <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/ThicknessGauge.dat"

# define new column names for if you look at the raw table
column_names <- c("Part", "Op1M1", "Op1M2", "Op2M1", "Op2M2", "Op3M1", "Op3M2")

# read the data, skip first two lines to get straight to the data
data_A <-read.table(url_part_A, col.names=column_names, skip=2)

# display the raw table
print(data_A)
```

```
##    Part Op1M1 Op1M2 Op2M1 Op2M2 Op3M1 Op3M2
## 1     1 0.953 0.952 0.954 0.954 0.954 0.956
## 2     2 0.956 0.956 0.956 0.957 0.958 0.957
## 3     3 0.956 0.955 0.956 0.955 0.957 0.956
## 4     4 0.957 0.957 0.958 0.957 0.957 0.958
```

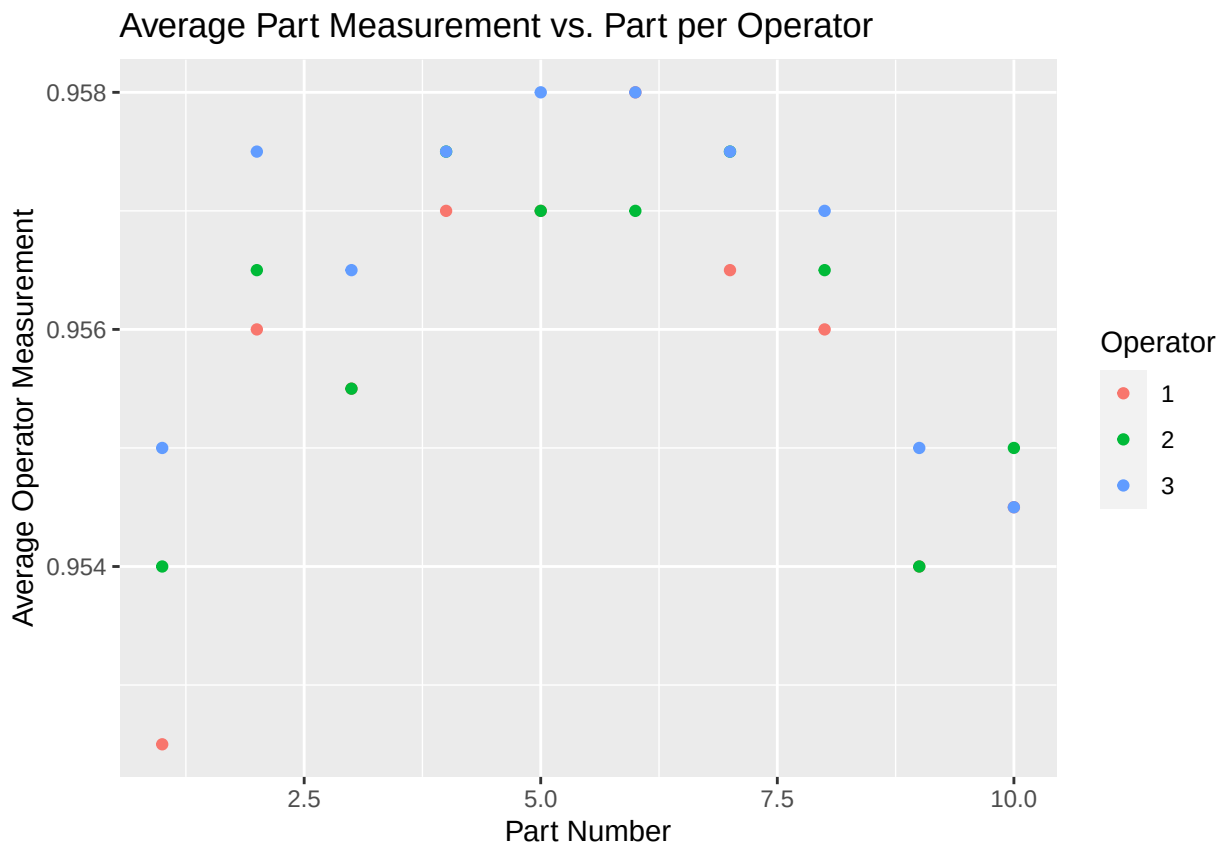
```
## 5     5 0.957 0.957 0.957 0.957 0.958 0.958
## 6     6 0.958 0.958 0.957 0.957 0.958 0.958
## 7     7 0.957 0.956 0.958 0.957 0.958 0.957
## 8     8 0.957 0.955 0.957 0.956 0.957 0.957
## 9     9 0.954 0.954 0.954 0.954 0.955 0.955
## 10   10 0.954 0.955 0.956 0.954 0.954 0.955
```

```r
# get the average part measurment per operator
Op1_avg <- rowMeans(data_A[, 2:3])
Op2_avg <- rowMeans(data_A[, 4:5])
Op3_avg <- rowMeans(data_A[, 6:7])

# create a dataframe out of the average data
df <- data.frame(x = data_A$Part,
                 y1 = Op1_avg,
                 y2 = Op2_avg,
                 y3 = Op3_avg)

# reshape the dataframe for plotting
df_reshaped <- data.frame(x = df$x,
                      y = c(df$y1, df$y2, df$y3),
                      Operator = c(rep("1", nrow(df)),
                                   rep("2", nrow(df)),
                                   rep("3", nrow(df))))

# plot the data using ggplot
ggplot(df_reshaped, aes(x, y, col = Operator)) +
  geom_point() +
  labs(x = "Part Number", y = "Average Operator Measurement", title = "Average Part Measurement vs. Par
```



Part B

```
# get url for the data
url_part_B <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"

# load in the data skipping first line and separating on spaces
data_B <- read.table(url_part_B, header = FALSE, skip=1, sep = " ", fill = TRUE, na.strings = "")

# make a new vectors combining each of the redundant columns
reshaped_B <- data.frame(
  "BrainWeight" = c(data_B$V1, data_B$V3, data_B$V5),
  "BodyWeight" = c(data_B$V2, data_B$V4, data_B$V6)
)

# drop na values from the resultant dataframe
reshaped_B <- na.omit(reshaped_B)

# display the raw dataframe
print(reshaped_B)
```

```
##    BrainWeight BodyWeight
## 1        3.385      44.50
## 2        0.480      15.50
## 3        1.350       8.10
```

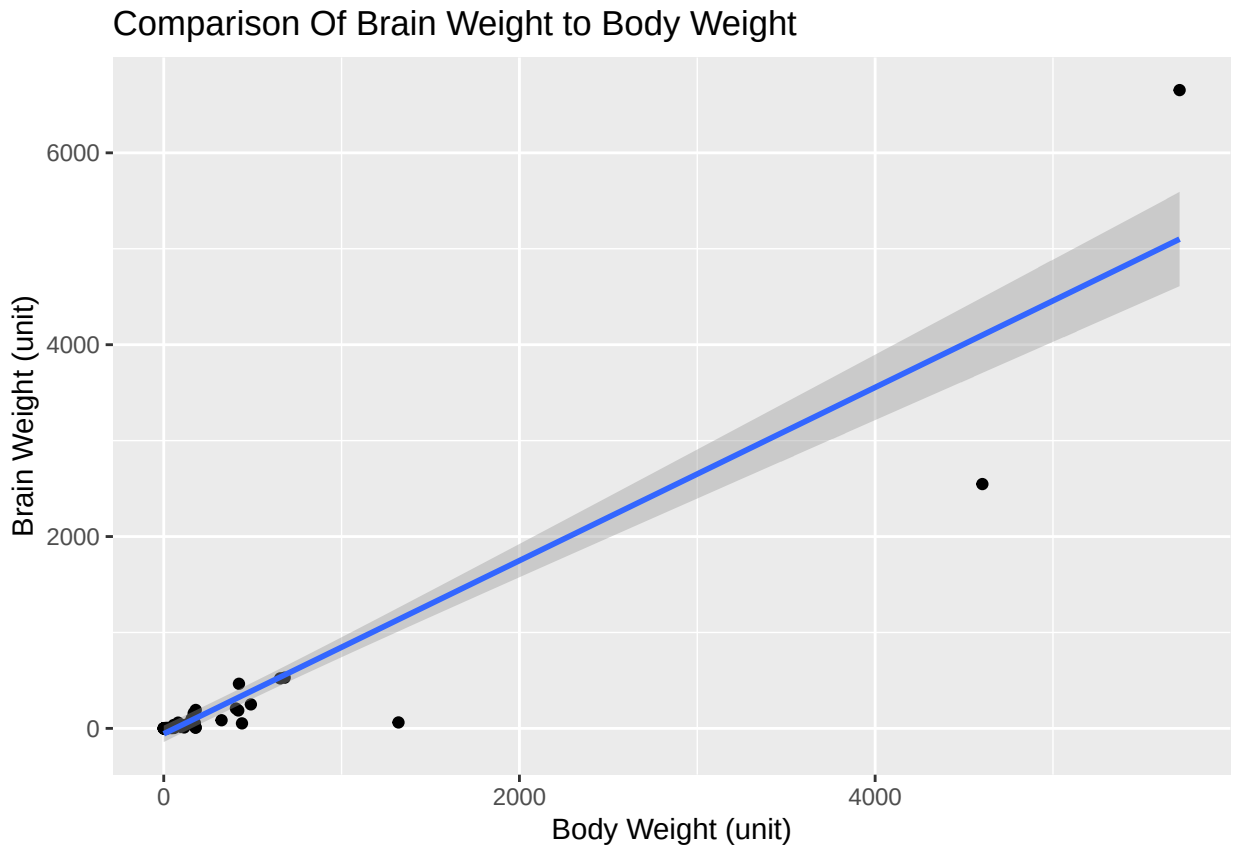
```
## 4      465.000     423.00
## 5       36.330     119.50
## 6       27.660     115.00
## 7       14.830      98.20
## 8        1.040       5.50
## 9        4.190      58.00
## 10       0.425       6.40
## 11       0.101       4.00
## 12       0.920       5.70
## 13       1.000       6.60
## 14       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
```

```r
# rewrite the table as a dataframe for ggplot
data_B <- data.frame(x = reshaped_B$BodyWeight,
                     y = reshaped_B$BrainWeight)

# display the data with x and the body weight and y as the brain weight
ggplot(data_B, aes(x=x, y=y)) +
  geom_point()+
  geom_smooth(method=lm) + # add a linear regression line for fun
  labs(x = "Body Weight (unit)", y = "Brain Weight (unit)", title = "Comparison Of Brain Weight to Body
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Part C

```r
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(tidyr)
```

```
# get url for the data
url_part_C <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"

# read in the data using fread this time
# there is no hearder in this file and we need to skip the first line for its broken column labels
data_C <- fread(url_part_C, header=FALSE, skip=1, fill=TRUE)

# reshape the data combining the redundant columns
reshaped_C <- data.frame(
  "Year" = c(data_C$V1, data_C$V3, data_C$V5, data_C$V7),
  "Long Jump" = c(data_C$V2, data_C$V4, data_C$V6, data_C$V8)
)

# drop na values from the resultant dataframe
reshaped_C <- na.omit(reshaped_C)

# add 1900 to the year so it is more readable
reshaped_C$Year <- reshaped_C$Year + 1900

# display the raw table
print(reshaped_C)
```

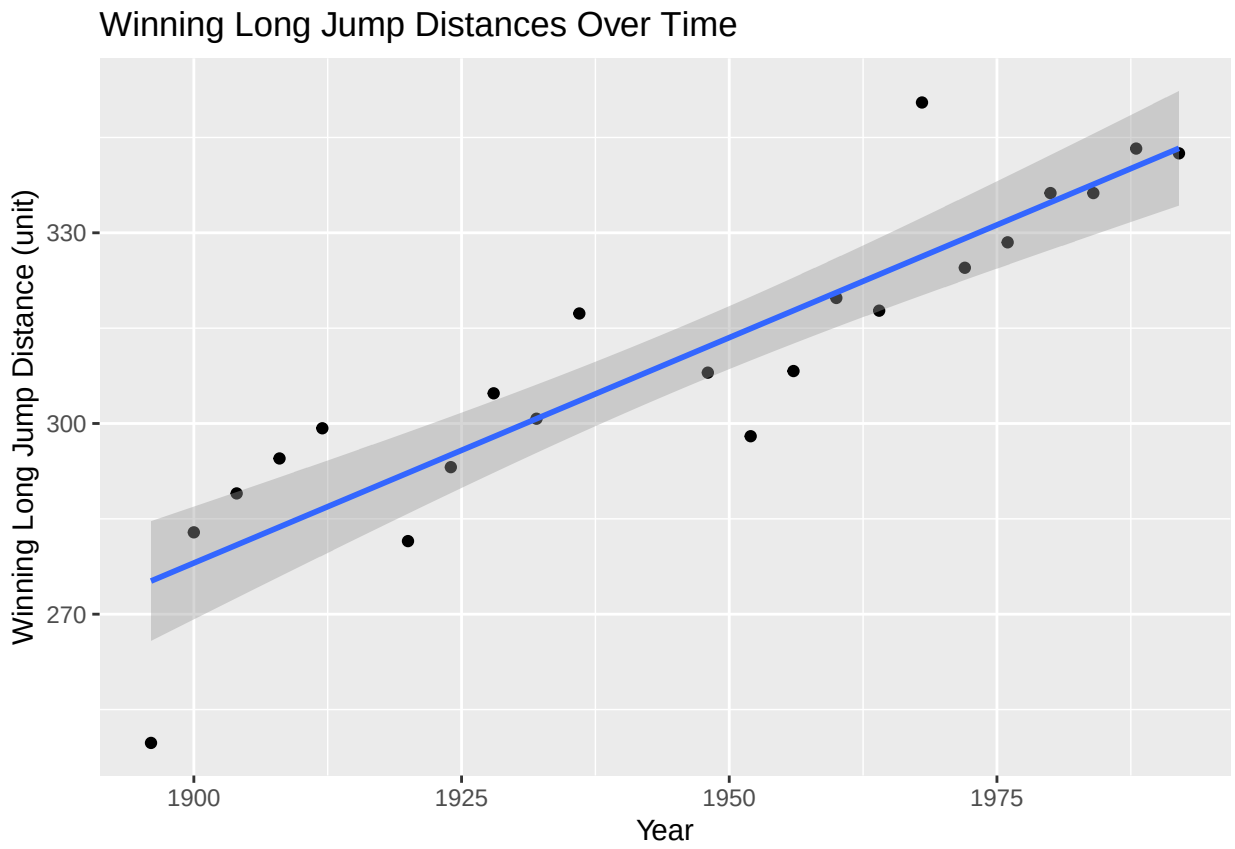
```
##    Year Long.Jump
## 1  1896    249.75
## 2  1900    282.88
## 3  1904    289.00
## 4  1908    294.50
## 5  1912    299.25
## 6  1920    281.50
## 7  1924    293.13
## 8  1928    304.75
## 9  1932    300.75
## 10 1936    317.31
## 11 1948    308.00
## 12 1952    298.00
## 13 1956    308.25
## 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
```

```r
# rewrite the table as a dataframe for ggplot
data_C <- data.frame(x = reshaped_C$Year,
                     y = reshaped_C$Long.Jump)

# display the data with x and the body weight and y as the brain weight
ggplot(data_C, aes(x=x, y=y)) +
  geom_point()+
  geom_smooth(method=lm) + # add a linear regression line for fun
  labs(x = "Year", y = "Winning Long Jump Distance (unit)", title = "Winning Long Jump Distances Over Ti
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Part D

```r
# get url for the data
url_part_D <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"

# read in the data using fread
# again, there is a header but it isn't very helpful, instead we will skip the first two rows to get st
# we are also going to separate based on spaces since later we can break based on the commas in the nes
data_D <- fread(url_part_D, header=FALSE, skip=2, fill=TRUE, sep=" ")

# separate each of the read columns into three new ones corresponding to their group (10k, 20k, 30k)
df <- data_D %>%
  separate(col = "V2", into = c("M1_10,000", "M2_10,000", "M3_10,000"), sep = ",") %>%
  separate(col = "V3", into = c("M1_20,000", "M2_20,000", "M3_20,000"), sep = ",") %>%
  separate(col = "V4", into = c("M1_30,000", "M2_30,000", "M3_30,000"), sep = ",")
```

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

```r
# convert all applicable columns to numeric
df[, 1:10] <- lapply(df[, 1:10], as.numeric)
```

```
## Warning in lapply(df[, 1:10], as.numeric): NAs introduced by coercion
```

```r
# display the raw table
print(df)
```

```
##   V1 M1_10,000 M2_10,000 M3_10,000 M1_20,000 M2_20,000 M3_20,000 M1_30,000
## 1 NA      16.1      15.3      17.5      16.6      19.2      18.5      20.8
## 2 NA       8.1       8.6      10.1      12.7      13.7      11.5      14.4
##   M2_30,000 M3_30,000
## 1      18.0      21.0
## 2      15.4      13.7
```

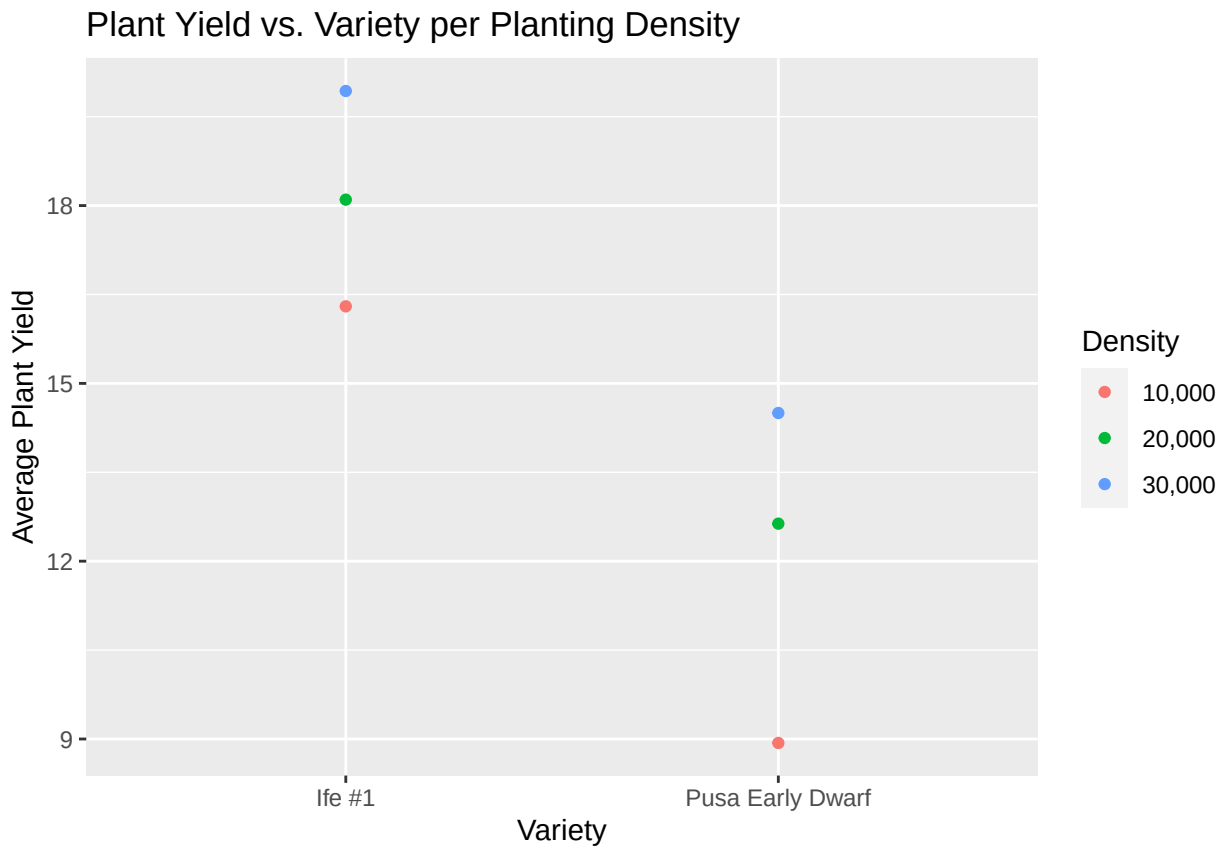
```r
# get the average part measurment per operator
tenk_avg <- rowMeans(df[, 2:4])
twentyk_avg <- rowMeans(df[, 5:7])
thirtyk_avg <- rowMeans(df[, 8:10])

# create a dataframe out of the average data
df <- data.frame(x = c("Ife #1", "Pusa Early Dwarf"),
                 y1 = tenk_avg,
                 y2 = twentyk_avg,
                 y3 = thirtyk_avg)

# reshape the dataframe for plotting
df_reshaped <- data.frame(x = df$x,
                      y = c(df$y1, df$y2, df$y3),
                      Density = c(rep("10,000", nrow(df)),
                                  rep("20,000", nrow(df)),
                                  rep("30,000", nrow(df))))

# plot the data using ggplot
ggplot(df_reshaped, aes(x, y, col = Density)) +
```

```
  geom_point() +
  labs(x = "Variety", y = "Average Plant Yield", title = "Plant Yield vs. Variety per Planting Density")
```



Part E

```
url_part_E <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LarvaeControl.dat"
data_E <- fread(url_part_E, header=TRUE, skip=2, fill=TRUE)

# run a quick loop to change the names of the columns beyond the block
for (i in c(2:11)) {

  # if part of the first 5 then its treatment 1
  if (i < 7) {
    # change the name to age 1 followed by treatment number
    colnames(data_E)[i] <- paste("Age1_Treat", i-1, sep="")
  }
  # if part of the next 5 then its treatment 1
  else {
    # change the name to age 2 followed by treatment number
    colnames(data_E)[i] <- paste("Age2_Treat", i-6, sep="")
  }
}
```

```r
# display the raw dataframe
print(data_E)
```

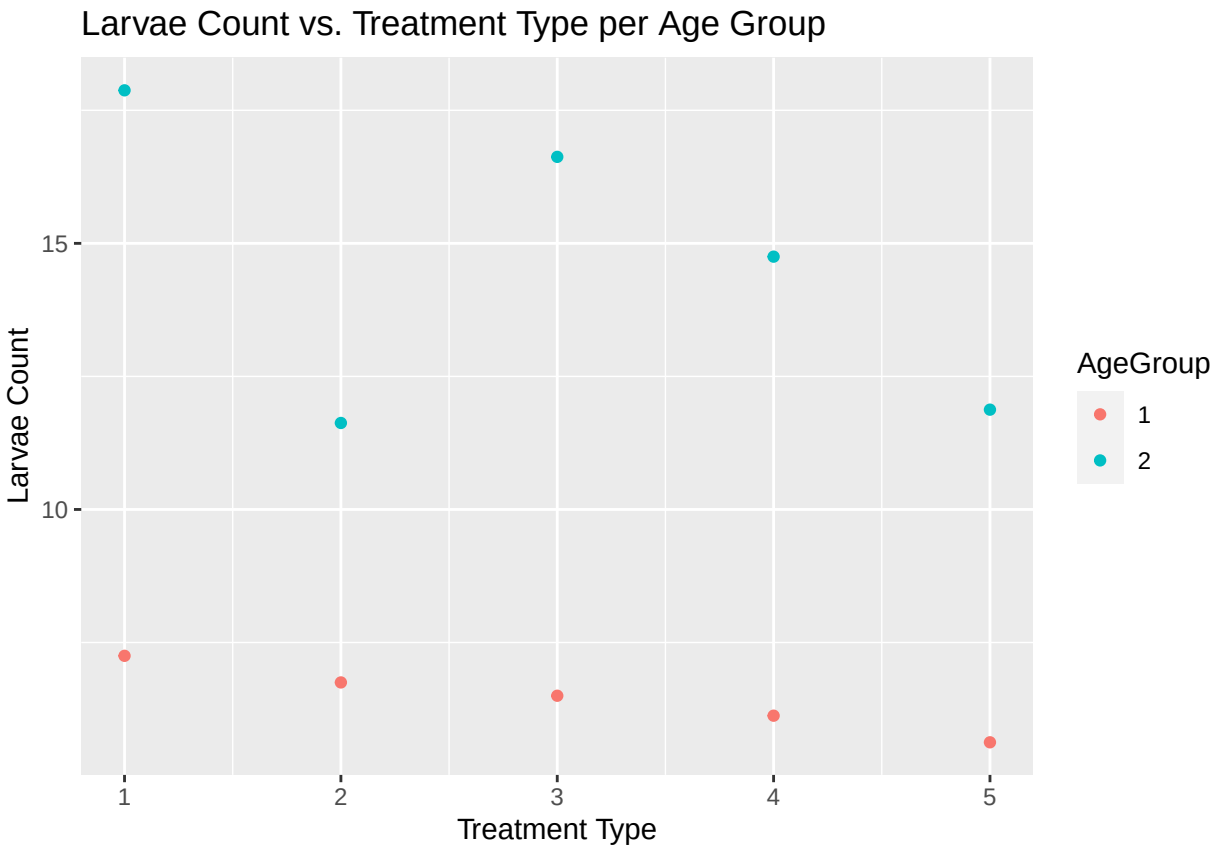
```
##    Block Age1_Treat1 Age1_Treat2 Age1_Treat3 Age1_Treat4 Age1_Treat5
## 1:     1          13          16          13          20          16
## 2:     2          29          12          23          15          17
## 3:     3           5           4           4           1           2
## 4:     4           5          12           1           5           3
## 5:     5           0           2           2           2           0
## 6:     6           1           1           1           3           5
## 7:     7           1           3           1           0           1
## 8:     8           4           4           7           3           1
##    Age2_Treat1 Age2_Treat2 Age2_Treat3 Age2_Treat4 Age2_Treat5
## 1:          28          12          40          31          22
## 2:          61          49          48          44          45
## 3:           7           2           4           5           2
## 4:          14           5          14           9           8
## 5:           3           3           2           7           0
## 6:           7           6           7           7           4
## 7:          10           5           8           3           6
## 8:          13          11          10          12           8
```

```r
# get the average part measurement per operator
averages <- colMeans(data_E)

# create a dataframe out of the average data
df <- data.frame(x = c(1:5),
                 y1 = averages[2:6],
                 y2 = averages[7:11])

# reshape the dataframe for plotting
df_reshaped <- data.frame(x = df$x,
                      y = c(df$y1, df$y2),
                      AgeGroup = c(rep("1", nrow(df)),
                                   rep("2", nrow(df))))

# plot the data using ggplot
ggplot(df_reshaped, aes(x, y, col = AgeGroup)) +
  geom_point() +
  labs(x = "Treatment Type", y = "Larvae Count", title = "Larvae Count vs. Treatment Type per Age Group
```


Larvae Count vs. Treatment Type per Age Group
Larvae Count
15
10
1
2
3
4
5
Treatment Type
AgeGroup
1
2