Homework 2

Due Wednesday Sep 16

Tsering Dolkar

9/6/2020

Problem 1

I finished the Primers titled "Work with Data" and "Tidy Your Data" on Rstudio cloud.

Problem 2

Rmarkdown of this HW is opened.

Problem 3

While I was working on this Homework, I was not able to commit the changes I made through trial and error and regretted not having a backup. I was going back and forth on which tidy version of data might work for me and for some problems, I lost my code when I made the changes and had to start over on it. As was pointed out in the two links to StackOverflow on why we should use version control, such issues could easily be addressed using version control i.e git.

Problem 4

a.

We are looking for sensory data from five operators from Wu and Hamada's book: http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat

First of all, we will get the data from the link above:

```
## getting http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat

url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
sensorydata_raw<- fread(url, data.table = FALSE, fill = TRUE, skip = 2, header = FALSE)
saveRDS(sensorydata_raw, "sensorydata_raw.RDS")
#Saves the object in it's native format without the name: When importing, it is easier for us.
sensorydata_raw <- readRDS("sensorydata_raw.RDS")
```

Need to tidy the data, basic issue is we need items, operators and each of their sensory data on the 10 items are columns.

```r
#if the first condition is T, return the index value
Na <- which(is.na(sensorydata_raw$V6),arr.ind=T)
df <- cbind(rep(1:10, each = 2), sensorydata_raw[Na,])
df$V6 <- NULL
#create a clone data frame of the raw data, to get a better aligned raw data
# and not change the original raw data.
new_sensorydata_raw <- sensorydata_raw
new_sensorydata_raw[Na,] <- df
colnames(new_sensorydata_raw) <- c("Item","Operator1","Operator2","Operator3","Operator4","Operator5")
as.data.frame(head(new_sensorydata_raw))
```

```
##   Item Operator1 Operator2 Operator3 Operator4 Operator5
## 1    1       4.3       4.9       3.3       5.3       4.4
## 2    1       4.3       4.5       4.0       5.5       3.3
## 3    1       4.1       5.3       3.4       5.7       4.7
## 4    2       6.0       5.3       4.5       5.9       4.7
## 5    2       4.9       6.3       4.2       5.5       4.9
## 6    2       6.0       5.9       4.7       6.3       4.6
```

```r
Operator <- stack(new_sensorydata_raw[,2:6])
sensorydata_tidy_br <- data.frame(Item=rep(new_sensorydata_raw$Item, 5),
                                  as.character(Operator[,2]), as.numeric(Operator[,1]))
colnames(sensorydata_tidy_br) <- c('Item', "Operator", "sensorydata")



head(sensorydata_tidy_br)
```

```
##   Item  Operator sensorydata
## 1    1 Operator1         4.3
## 2    1 Operator1         4.3
## 3    1 Operator1         4.1
## 4    2 Operator1         6.0
## 5    2 Operator1         4.9
## 6    2 Operator1         6.0
```

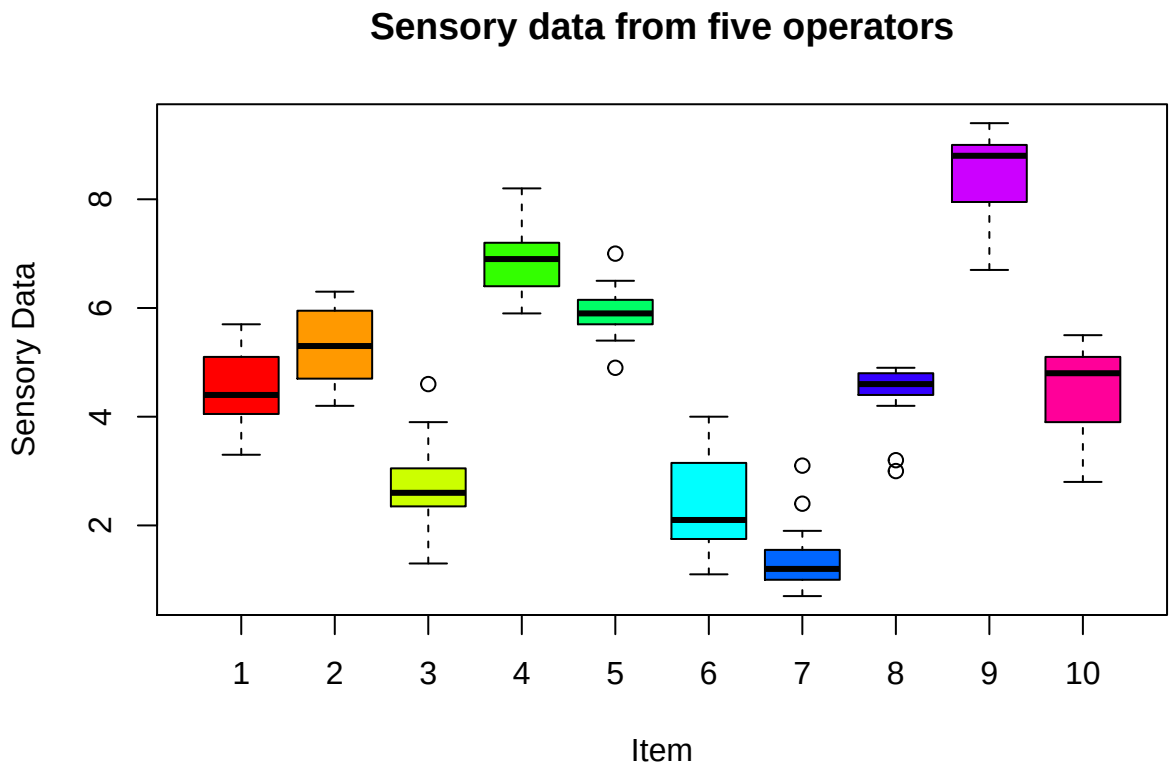
We have converted the dataframes to tidy dataframes using the base functions. Here is a summary of the data:

Item	Operator1	Operator2	Operator3	Operator4	Operator5
Min. : 1.0	Min. :0.900	Min. :1.500	Min. :0.800	Min. :0.900	Min. :0.700
1st Qu.: 3.0	1st Qu.:2.850	1st Qu.:3.450	1st Qu.:2.650	1st Qu.:3.925	1st Qu.:2.250
Median : 5.5	Median :4.550	Median :4.950	Median :4.150	Median :5.400	Median :4.600
Mean : 5.5	Mean :4.593	Mean :5.063	Mean :4.167	Mean :5.193	Mean :4.267
3rd Qu.: 8.0	3rd Qu.:5.950	3rd Qu.:6.225	3rd Qu.:5.400	3rd Qu.:6.275	3rd Qu.:5.800
Max. :10.0	Max. :9.000	Max. :9.200	Max. :9.000	Max. :9.400	Max. :8.800

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

Then our boxplot of Item to sensorydata is:




```
#if the first condition is T, return the index value
Na <- which(is.na(sensorydata_raw$V6),arr.ind=T)
df <- cbind(rep(1:10, each = 2), sensorydata_raw[Na,])
df$V6 <- NULL
#create a clone data frame of the raw data, to get a better aligned raw data and not
# change the original raw data.
new_sensorydata_raw <- sensorydata_raw
new_sensorydata_raw[Na,] <- df
colnames(new_sensorydata_raw) <- c("Item","Operator1","Operator2","Operator3","Operator4","Operator5")
as.data.frame(head(new_sensorydata_raw))
```

```
##   Item Operator1 Operator2 Operator3 Operator4 Operator5
## 1    1       4.3       4.9       3.3       5.3       4.4
## 2    1       4.3       4.5       4.0       5.5       3.3
## 3    1       4.1       5.3       3.4       5.7       4.7
## 4    2       6.0       5.3       4.5       5.9       4.7
## 5    2       4.9       6.3       4.2       5.5       4.9
## 6    2       6.0       5.9       4.7       6.3       4.6
```

```
sensorydata_tidy_tv <-
 new_sensorydata_raw %>%
  gather(key = "Operators", value = "sensorydata", Operator1, Operator2, Operator3, Operator4,
         Operator5)
```

```
head(sensorydata_tidy_tv)
```

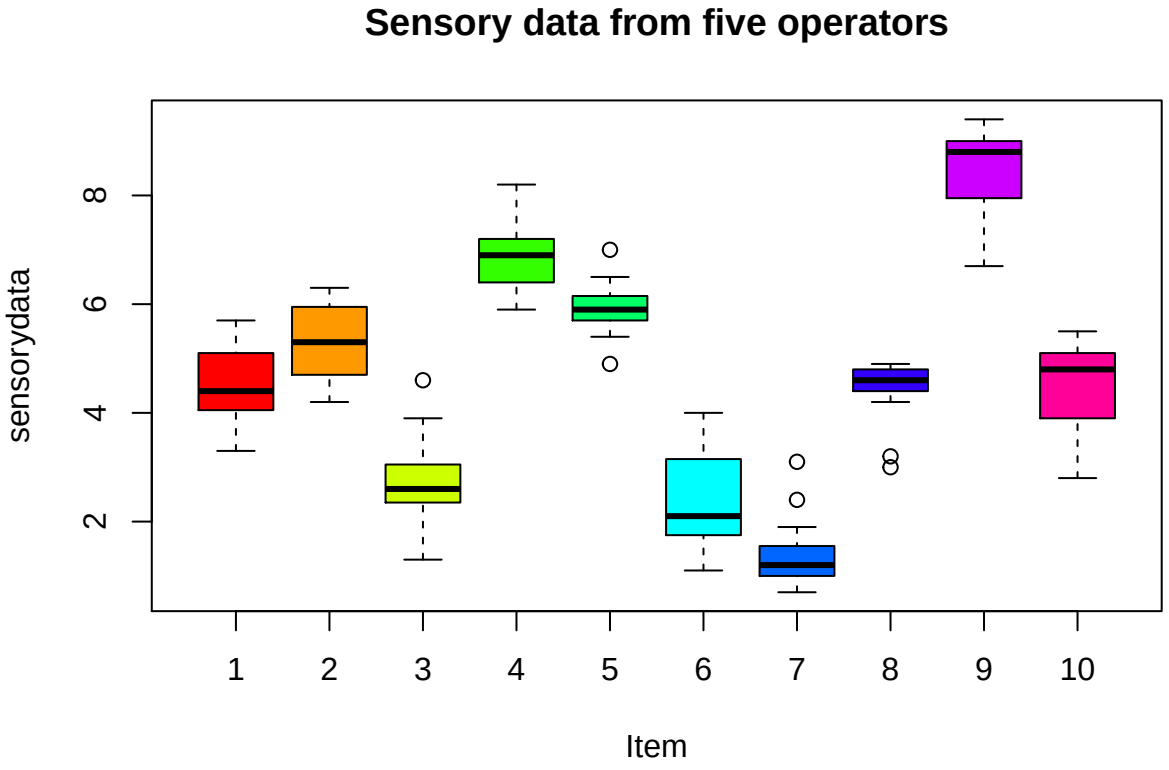
```
##   Item Operators sensorydata
## 1    1 Operator1         4.3
## 2    1 Operator1         4.3
## 3    1 Operator1         4.1
## 4    2 Operator1         6.0
## 5    2 Operator1         4.9
## 6    2 Operator1         6.0
```

We have converted the dataframes to tidy dataframes using the tidyverse functions. Here is a summary of the data:

	Item	Operator1	Operator2	Operator3	Operator4	Operator5
	Min. : 1.0	Min. :0.900	Min. :1.500	Min. :0.800	Min. :0.900	Min. :0.700
	1st Qu.: 3.0	1st Qu.:2.850	1st Qu.:3.450	1st Qu.:2.650	1st Qu.:3.925	1st Qu.:2.250
	Median : 5.5	Median :4.550	Median :4.950	Median :4.150	Median :5.400	Median :4.600
	Mean : 5.5	Mean :4.593	Mean :5.063	Mean :4.167	Mean :5.193	Mean :4.267
	3rd Qu.: 8.0	3rd Qu.:5.950	3rd Qu.:6.225	3rd Qu.:5.400	3rd Qu.:6.275	3rd Qu.:5.800
	Max. :10.0	Max. :9.000	Max. :9.200	Max. :9.000	Max. :9.400	Max. :8.800

Item	Operators	sensorydata
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
Max. :10.0	NA	Max. :9.400

Then our boxplot of Item to sensorydata is:



b.

We are looking at the Gold Medal performance for Olympic Men's Long Jump, year is coded as 1900=0 from Wu and Hamada's book: http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat

First, we will get the data from the link above:

```
## getting http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat

url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"

LongJumpData_raw<- fread(url, header = FALSE, fill = TRUE, skip = 1)
saveRDS(LongJumpData_raw, "LongJumpData_raw.RDS")
#Saves the object in it's native format without the name: When importing, it is easier for us.

LongJumpData_raw <- readRDS("LongJumpData_raw.RDS")
```

Need to tidy the data, basic issues are Year and Longjump are columns, need to push them into column.

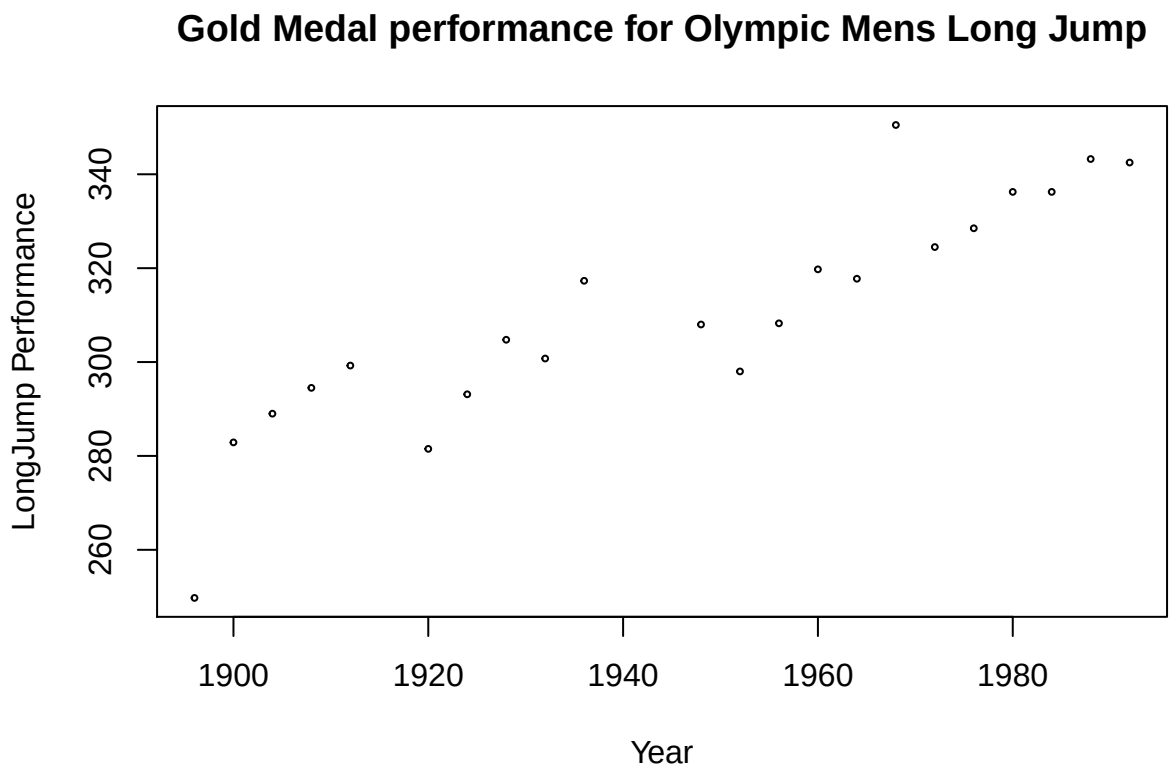
```
Year <- c(LongJumpData_raw$'V1', LongJumpData_raw$'V3', LongJumpData_raw$'V5', LongJumpData_raw$'V7')
LongJumpData <- c(LongJumpData_raw$V2, LongJumpData_raw$V4, LongJumpData_raw$V6, LongJumpData_raw$V8)
Year <- as.numeric(Year)
LongJumpData <- as.numeric(LongJumpData)
Year <- Year + 1900
longjump_tidy_br <- cbind(Year, LongJumpData)
longjump_tidy_br <- longjump_tidy_br[1:22,]
longjump_tidy_br <- as.data.frame(longjump_tidy_br)

head(longjump_tidy_br)
```

```
##   Year LongJumpData
## 1 1896       249.75
## 2 1900       282.88
## 3 1904       289.00
## 4 1908       294.50
## 5 1912       299.25
## 6 1920       281.50
```

We have converted the dataframes to tidy dataframes using the base functions. Here is a summary of the data:

Year	LongJumpData
Min. :1896	Min. :249.8
1st Qu.:1921	1st Qu.:295.4
Median :1950	Median :308.1
Mean :1945	Mean :310.3
3rd Qu.:1971	3rd Qu.:327.5
Max. :1992	Max. :350.5

Then our plot of year to Gold Medal performance for Olympic Men's Long Jump is:




```
lj_year<-
  LongJumpData_raw %>%
    gather(key = "Vector2", value = "Year", 'V1','V3','V5','V7', convert = TRUE)%>%
    select(Year)%>%
    slice(1:(n()-2))%>% #cuts off the NAs at the end
    mutate(Year = 1900 + Year)
lj_data<-
  LongJumpData_raw %>%
```

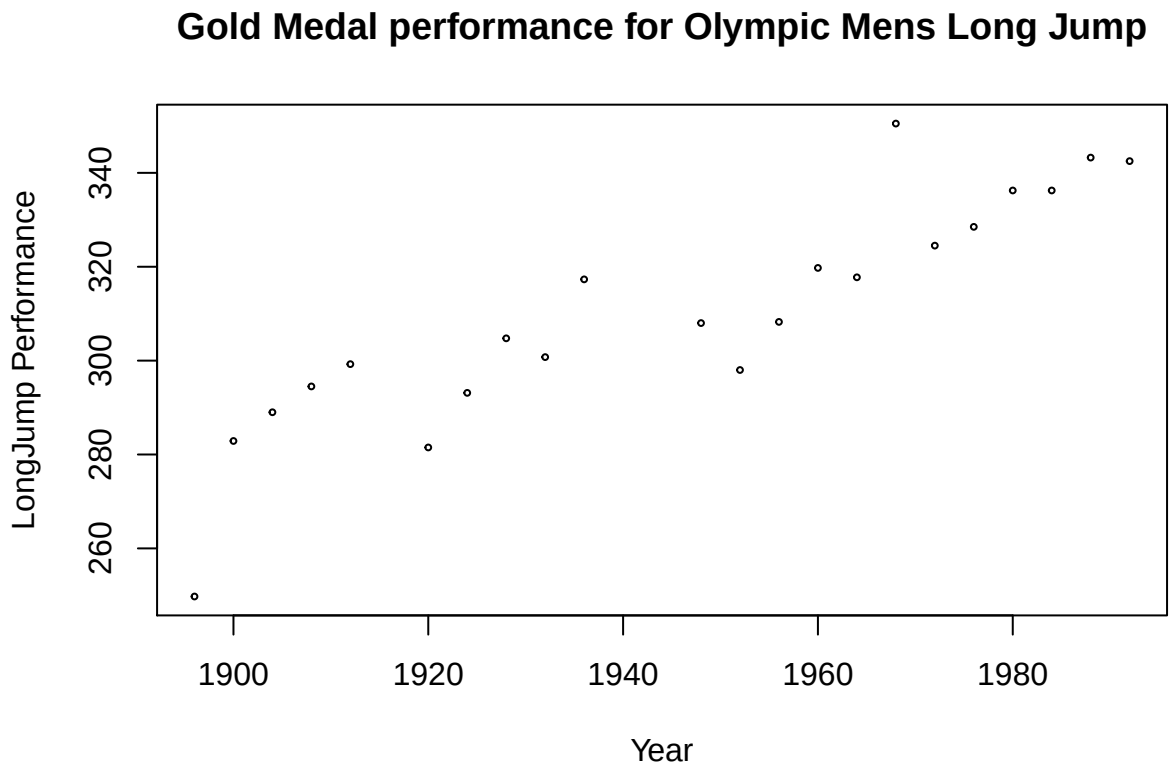
```
    gather(key = "Vector1", value = "LongJumpData", 'V2','V4','V6','V8', convert = TRUE) %>%
    select(LongJumpData)%>%
    slice(1:(n()-2)) #cuts off the NAs at the end

longjump_tidy_tv <- cbind(lj_year, lj_data)
head(longjump_tidy_tv)
```

```
##   Year LongJumpData
## 1 1896       249.75
## 2 1900       282.88
## 3 1904       289.00
## 4 1908       294.50
## 5 1912       299.25
## 6 1920       281.50
```

We have converted the dataframes to tidy dataframes using the tidyverse functions. Here is a summary of the data:

Year	LongJumpData
Min. :1896	Min. :249.8
1st Qu.:1921	1st Qu.:295.4
Median :1950	Median :308.1
Mean :1945	Mean :310.3
3rd Qu.:1971	3rd Qu.:327.5
Max. :1992	Max. :350.5

Then our plot of year to Gold Medal performance for Olympic Men's Long Jump is:



c.

We are looking at brain weight (g) and body weight (kg) for 62 species from Wu and Hamada's book: http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat

First, we will get the data from the link above:

```
## getting http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat

url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"

brainandbodyweight_raw<- fread(url, header = FALSE, fill = TRUE, skip = 1)
saveRDS(brainandbodyweight_raw, "brainandbodyweight_raw.RDS")
#Saves the object in it's native format without the name: When importing, it is easier for us.

brainandbodyweight_raw <- readRDS("brainandbodyweight_raw.RDS")
```

Need to tidy the data, basic issues are brain weight and body weight are columns, need to push them into column.

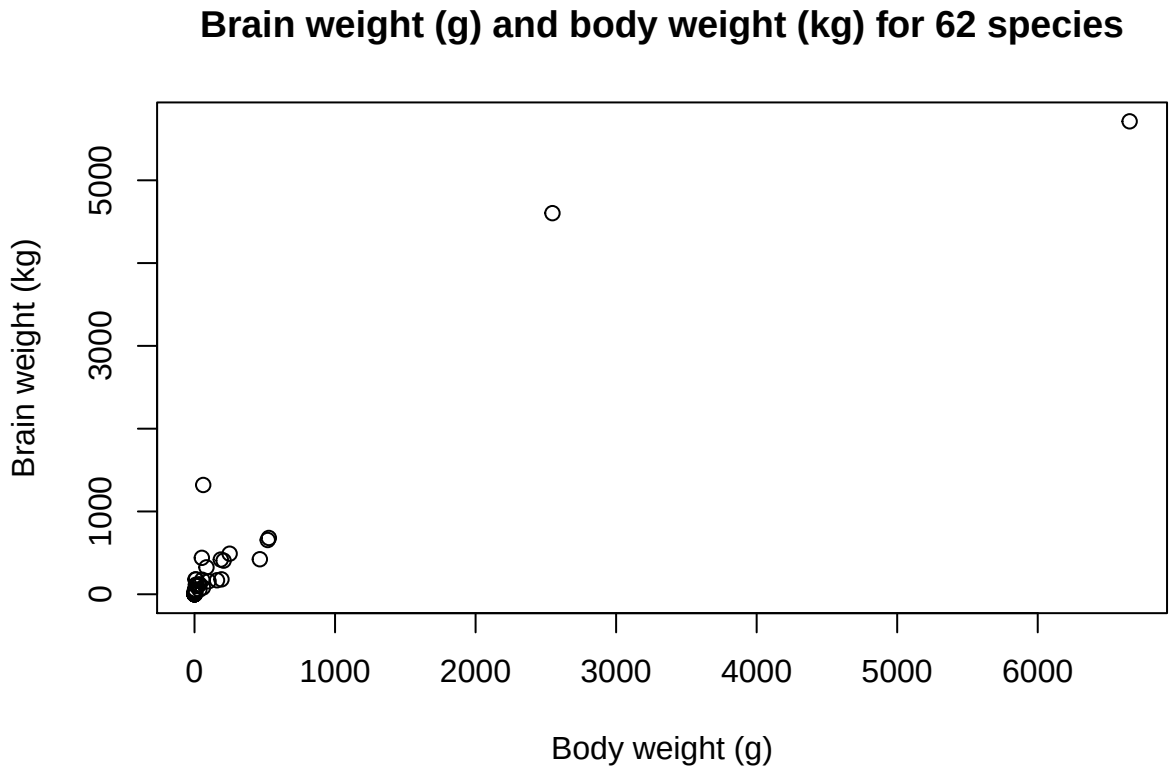
```
bodyweight <- c(brainandbodyweight_raw$'V1', brainandbodyweight_raw$'V3', brainandbodyweight_raw$'V5')
brainweight <- c(brainandbodyweight_raw$V2, brainandbodyweight_raw$V4, brainandbodyweight_raw$V6)
bodyweight <- as.numeric(bodyweight)
brainweight <- as.numeric(brainweight)
brainandbodyweight_tidy_br <- cbind(bodyweight, brainweight)
brainandbodyweight_tidy_br <- brainandbodyweight_tidy_br[1:62,]
brainandbodyweight_tidy_br <- as.data.frame(brainandbodyweight_tidy_br)

head(brainandbodyweight_tidy_br)
```

```
##   bodyweight brainweight
## 1      3.385        44.5
## 2      0.480        15.5
## 3      1.350         8.1
## 4    465.000       423.0
## 5     36.330       119.5
## 6     27.660       115.0
```

We have converted the dataframes to tidy dataframes using the base functions. Here is a summary of the data:

bodyweight	brainweight
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.: 48.202	3rd Qu.: 166.00
Max. :6654.000	Max. :5712.00

Then our plot of Brain weight (g) and body weight (kg) for 62 species:




```
bodyweight<-
  brainandbodyweight_raw %>%
    gather(key = "Vector2", value = "bodyweight", V1, V3, V5, convert = TRUE)%>%
    select(bodyweight)%>%
    slice(1:(n()-1)) #cuts off the NAs at the end

brainweight<-
  brainandbodyweight_raw %>%
    gather(key = "Vector1", value = "brainweight", V2, V4, V6, convert = TRUE)%>%
    select(brainweight)%>%
    slice(1:(n()-1)) #cuts off the NAs at the end

brainandbodyweight_tidy_tv <- cbind(bodyweight, brainweight)


head(brainandbodyweight_tidy_tv)
```

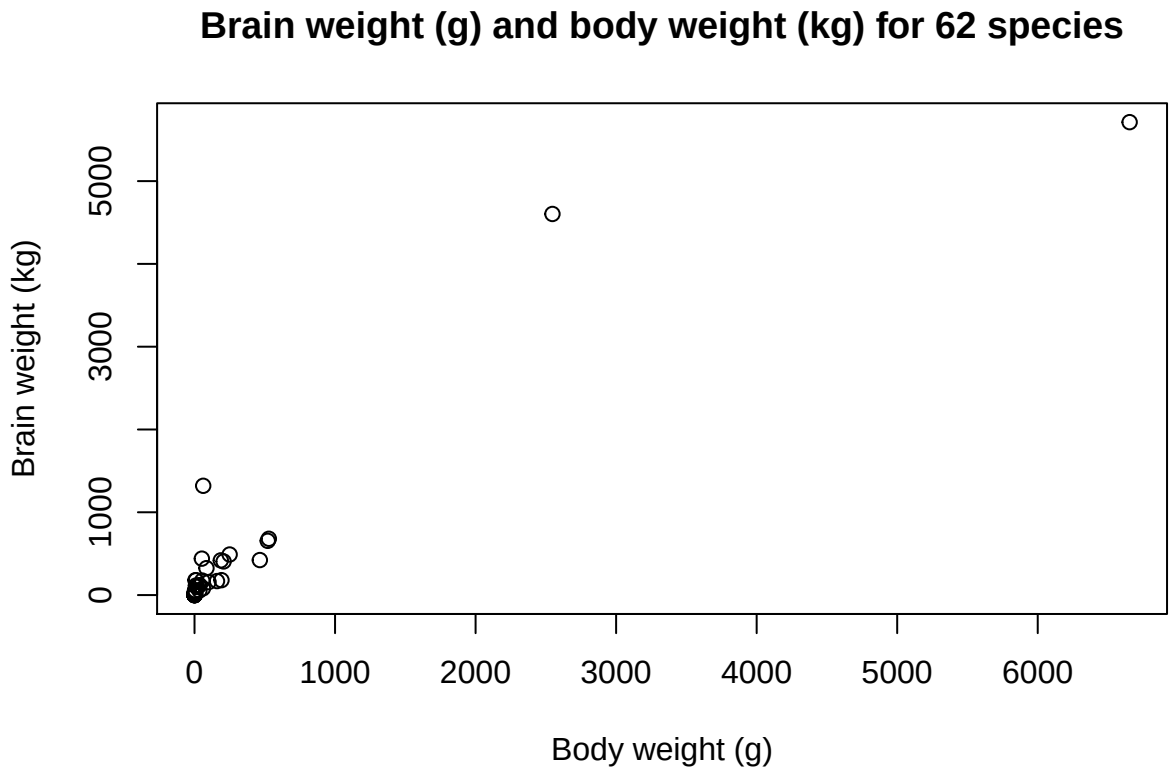
```
##   bodyweight brainweight
## 1      3.385        44.5
## 2      0.480        15.5
## 3      1.350         8.1
## 4    465.000       423.0
## 5     36.330       119.5
## 6     27.660       115.0
```

We have converted the dataframes to tidy dataframes using the tidyverse functions. Here is a summary of the data:

bodyweight	brainweight
Min. : 0.005	Min. : 0.10

bodyweight	brainweight
1st Qu.: 0.600	1st Qu.: 4.25
Median : 3.342	Median : 17.25
Mean : 198.790	Mean : 283.13
3rd Qu.: 48.202	3rd Qu.: 166.00
Max. :6654.000	Max. :5712.00

Then our plot of Brain weight (g) and body weight (kg) for 62 species:



d.

We will look to triplicate measurements of tomato yield for two varieties of tomatos at three planting densities from Wu and Hamada's book: http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat

First, we will get the data from the link above:

```
## getting http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat

url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"

tomato_raw<- fread(url)
```

```
## Warning in fread(url): 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
saveRDS(tomato_raw, "tomato_raw.RDS")
#Saves the object in it's native format without the name: When importing, it is easier for us.

tomato_raw <- readRDS("tomato_raw.RDS")
```

Need to tidy the data, basic issue is we need a column of different densities, two column of measurements for each type of tomato and a column called M number to keep track of which of 3 measurement for each density we are looking at.

```r
#create a dataframe called tomato_tidy where there is 3 of the tomatovariety, the string of
# their measurements, and densities.
tomato_tidy_br <- data.frame(TomatoVariety=rep(tomato_raw$V1, 3),
                             stack(tomato_raw[,-1]))
colnames(tomato_tidy_br) <- c("TomatoVariety", "V1", "Density")

# we will now try to separate the string of measurement into individual measurements
M <- data.frame(strsplit(head(tomato_tidy_br$V1), ','))
colnames(M) <- c("1", "2", "3", "4", "5", "6")
rownames(M) <- c("M1", "M2", "M3")
M <- as.data.frame(t(M))

#Separate the measurements into columns of measurements for Tomato type 1 and type 2
Type1 <- as.data.frame(stack(M[c(1,3,5),]))
colnames(Type1) <- c("Ife1", "M_number")
# Since the measurement number is redundant, remove one column out of two M numbers
Type1$M_number <- NULL
Type2 <- as.data.frame(stack(M[c(2,4,6),]))
colnames(Type2) <- c("PusaEarlyDwarf","M_number")
# bind the two types together into one dataframe
df <- as.data.frame(cbind(Type1, Type2))
#change the type of the variables to reflect true type.
df$Ife1 = as.numeric(df$Ife1)
df$PusaEarlyDwarf=as.numeric(df$PusaEarlyDwarf)
df1 <- as.data.frame(subset(df, select = c(Ife1, PusaEarlyDwarf)))
df2 <- as.data.frame(subset(df, select = M_number))

#create a dataframe with column of density values and dataframe with the rest of the information
subset.df = data.frame(Density=rep(c(1000,2000,3000), 3))
subset.df <- as.data.frame(subset.df)
new.df <- data.frame(cbind(df2, df1))

# bind the density, m number, and measurements of the two types of tomatoes and reoder the
# rows of the dataframe to look more like the tidyverse dataframe(since I did that first)
# for easy comparison reason.
tomato_tidy_br <- cbind(subset.df, new.df)
colnames(tomato_tidy_br) <- c("Density", "M number", 'Ife1', "PusaEarlyDwarf")
tomato_tidy1 <- data.frame(tomato_tidy_br[c(1,4,7),])
tomato_tidy2 <- data.frame(tomato_tidy_br[c(2,5,8),])
tomato_tidy3 <- data.frame(tomato_tidy_br[c(3,6,9),])
tomato_tidy_br <- data.frame(rbind(tomato_tidy1,tomato_tidy2,tomato_tidy3))
#the automated row index was showing up as out of order. For aesthetics, I removed it.
rownames(tomato_tidy_br) <- NULL
```

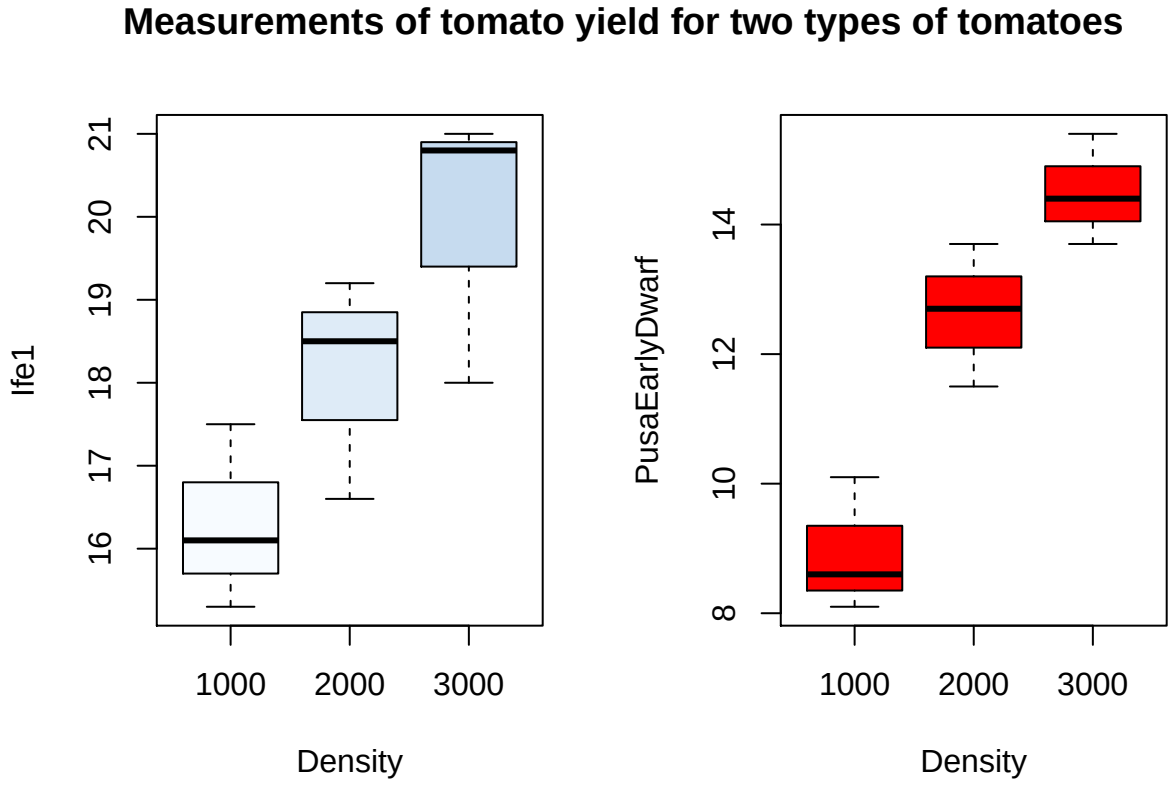
```
head(tomato_tidy_br)
```

```
##   Density M.number Ife1 PusaEarlyDwarf
## 1    1000       M1 16.1            8.1
## 2    1000       M2 15.3            8.6
## 3    1000       M3 17.5           10.1
## 4    2000       M1 16.6           12.7
## 5    2000       M2 19.2           13.7
## 6    2000       M3 18.5           11.5
```

We have converted the dataframes to tidy dataframes using the base functions. Here is a summary of the data:

Density	M.number	Ife1	PusaEarlyDwarf
Min. :1000	M1:3	Min. :15.30	Min. : 8.10
1st Qu.:1000	M2:3	1st Qu.:16.60	1st Qu.:10.10
Median :2000	M3:3	Median :18.00	Median :12.70
Mean :2000	NA	Mean :18.11	Mean :12.02
3rd Qu.:3000	NA	3rd Qu.:19.20	3rd Qu.:13.70
Max. :3000	NA	Max. :21.00	Max. :15.40

Then our plot of density to measurements of tomato yield for two varieties of tomatoes:




```
## [1] "Measurements of tomato yield for two types of tomatoes"
```

```
tomato_tidy_tv <-
  tomato_raw %>%
  gather(key = "Density", value = "Measurements", -V1, convert = TRUE) %>%
  rename(TomatoVariety = V1) %>%
```

```
  separate(Measurements, into = c("M1", "M2", "M3"), sep = ',', convert = TRUE) %>%
  gather(key = "M number", value = "Measurements", 3,4,5)%>%
  spread(key = TomatoVariety, value = Measurements)
```

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

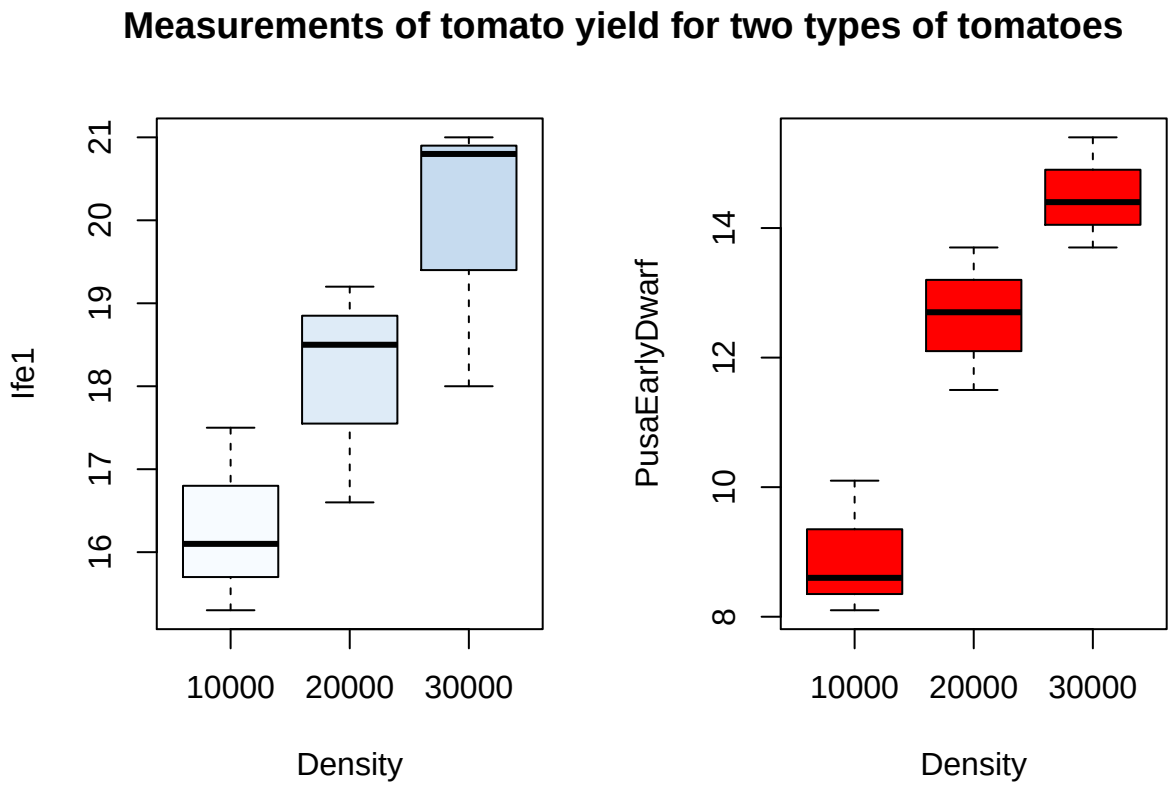
```
colnames(tomato_tidy_tv) <- c('Density', "M.number", "Ife1", "PusaEarlyDwarf")

head(tomato_tidy_tv)
```

```
##   Density M.number Ife1 PusaEarlyDwarf
## 1   10000       M1 16.1            8.1
## 2   10000       M2 15.3            8.6
## 3   10000       M3 17.5           10.1
## 4   20000       M1 16.6           12.7
## 5   20000       M2 19.2           13.7
## 6   20000       M3 18.5           11.5
```

We have converted the dataframes to tidy dataframes using the tidyverse functions.Here is a summary of the data:

Density	M.number	Ife1	PusaEarlyDwarf
Min. :10000	Length:9	Min. :15.30	Min. : 8.10
1st Qu.:10000	Class :character	1st Qu.:16.60	1st Qu.:10.10
Median :20000	Mode :character	Median :18.00	Median :12.70
Mean :20000	NA	Mean :18.11	Mean :12.02
3rd Qu.:30000	NA	3rd Qu.:19.20	3rd Qu.:13.70
Max. :30000	NA	Max. :21.00	Max. :15.40

Then our plot of density to measurements of tomato yield for two varieties of tomatoes:




```
## [1] "Measurements of tomato yield for two types of tomatoes"
```

Problem 5

Finish this homework by pushing your changes to your repo. In general, your workflow for this should be:

1. git pull – to make sure you have the most recent repo
2. In R: do some work
3. git add – this tells git to track new files
4. git commit – make message INFORMATIVE and USEFUL
5. git push – this pushes your local changes to the repo

If you have difficulty with steps 1-5, git is not correctly or completely setup. See me for help.