

# Homework3

*Hwasoo Shin*

*2019 9 13*

## Problem 3

For the analysis, we will be using tidyverse package. Within tidyverse package, we will especially use readr, dplyr, and ggplot package. In order to make better plots using ggplot, we will also import gridExtra package.

### Sensory Data

```
setwd('C:/Users/pc/Desktop/HWASOO/STUDY/StatPackage')
library(tidyverse)
```

```
## Registered S3 method overwritten by 'rvest':
##   method          from
##   read_xml.response xml2
```

```
## -- Attaching packages -----
```

```
## v ggplot2 3.2.1    v purrr   0.3.2
## v tibble  2.1.1    v dplyr   0.8.3
## v tidyr   0.8.3    v stringr 1.4.0
## v readr   1.3.1    v forcats 0.4.0
```

```
## -- Conflicts -----
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##   combine
```

```
#We will use tidyverse package to do the analysis
setwd('C:/Users/pc/Desktop/HWASOO/STUDY/StatPackage')
#To retrieve the file, we have to set the file path we will be using
```

```
Sensory<-read_table("Sensory.txt",skip=1)
```

```
## Parsed with column specification:
## cols(
##   `Item 1 2 3 4 5` = col_character()
## )
```

```
#read_table is in readr package, which is included in tidyverse package.
#Since the table we are reading has delimiters of white spaces and has one row to skip, we will use read_table2
Sensory<-as.data.frame(Sensory)
#Change the data type into data.frame
class(Sensory)
```

```
## [1] "data.frame"
```

```
#Check the type of data
dim(Sensory)
```

```
## [1] 30 1
```

```
#Check the dimension of data. Keep in mind that the data has only one column
SensoryName<-word(colnames(Sensory),sep=' ',1:6)
#We will fetch the names of data. We use word function to get the variables.
Sensory<-Sensory %>% separate(colnames(Sensory),SensoryName,sep=" ")
#Pipe operators (%>%). separate function will distribute the items in one column into multiple columns.
idx<-c(1:30)[-seq(1,30,by=3)]
#Since there are values where there isn't a item number, we will get rows without item numbers
Sensoryidx<-Sensory[idx,]
#Getting rows without item numbers
Sensory[,1]<-rep(1:10,each=3)
#Put item numbers on every rows
Sensory[idx,2:6]<-Sensoryidx
#This is our modified data
#We can also make a data which columns are order of items.
Sensory2<-matrix(0,nrow=15,ncol=10)
for(i in 1:10){
Sensory2[,i]<-as.numeric(as.matrix(Sensory %>% filter(Item==i) %>% select(2:6)))
}
#We will arrange observations by item number
Sensory2Col<-paste('Item',1:10,sep='')
#Make column names for matrix Sensory2
Sensory2<-data.frame(Sensory2)
#Convert Sensory2 data type from matrix to data.frame
colnames(Sensory2)<-Sensory2Col
```

Through these steps we can successfully import and clean the data. We used pipe operations, dplyr, and readr package for effective data munging. Below is some syntax to help us glimpse the information implied in the data.

```
summary(Sensory)
```

```
##           Item           1           2           3
##  Min.      : 1.0   Length:30   Length:30   Length:30
```

```
## 1st Qu.: 3.0    Class :character    Class :character    Class :character
## Median : 5.5    Mode  :character    Mode  :character    Mode  :character
## Mean   : 5.5
## 3rd Qu.: 8.0
## Max.   :10.0
##      4          5
## Length:30      Length:30
## Class :character    Class :character
## Mode  :character    Mode  :character
##
##
##
```

*#Summary of the first data file.*

```
summary(Sensory2)
```

```
##      Item1      Item2      Item3      Item4
## Min.   :3.300   Min.   :4.200   Min.   :1.300   Min.   :5.90
## 1st Qu.:4.050   1st Qu.:4.700   1st Qu.:2.350   1st Qu.:6.40
## Median :4.400   Median :5.300   Median :2.600   Median :6.90
## Mean   :4.467   Mean   :5.313   Mean   :2.773   Mean   :6.88
## 3rd Qu.:5.100   3rd Qu.:5.950   3rd Qu.:3.050   3rd Qu.:7.20
## Max.   :5.700   Max.   :6.300   Max.   :4.600   Max.   :8.20
##      Item5      Item6      Item7      Item8
## Min.   :4.90    Min.   :1.100   Min.   :0.700   Min.   :3.000
## 1st Qu.:5.70    1st Qu.:1.750   1st Qu.:1.000   1st Qu.:4.400
## Median :5.90    Median :2.100   Median :1.200   Median :4.600
## Mean   :5.92    Mean   :2.393   Mean   :1.407   Mean   :4.427
## 3rd Qu.:6.15    3rd Qu.:3.150   3rd Qu.:1.550   3rd Qu.:4.800
## Max.   :7.00    Max.   :4.000   Max.   :3.100   Max.   :4.900
##      Item9      Item10
## Min.   :6.700   Min.   :2.80
## 1st Qu.:7.950   1st Qu.:3.90
## Median :8.800   Median :4.80
## Mean   :8.467   Mean   :4.52
## 3rd Qu.:9.000   3rd Qu.:5.10
## Max.   :9.400   Max.   :5.50
```

*#Summary of the second data file.*

```
Sensory2Mean<-apply(Sensory2,2,mean)
```

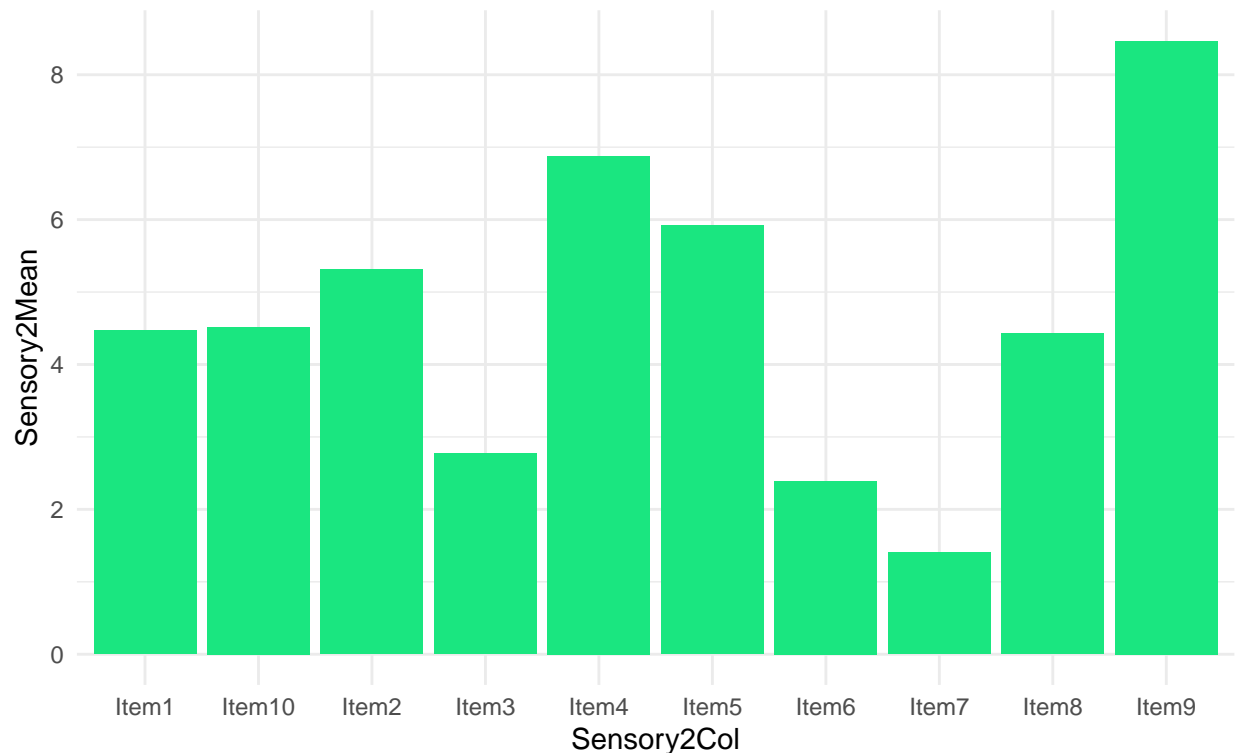
*#Mean by items.*

```
SensoryM<-data.frame(Sensory2Col,Sensory2Mean)
```

*#Make two columns into variables into data frame*

```
ggplot(SensoryM,aes(x=Sensory2Col,y=Sensory2Mean))+geom_bar(fill=rgb(0.1,0.9,0.5), stat = "identity")+
theme_minimal()+theme(plot.title = element_text(size=30,face="bold"))
```

# Mean by Items



*#Make a bar plot with the means by item. We can see that the Item9 has the biggest mean and Item7 has the smallest mean.*

By modifying the data we are able to see some information in each item. In this barplot, Item9 has the biggest mean and Item7 has the smallest mean.

## Long Jump Data

We can similarly follow the steps as above;

```
##### Long Jump Data #####
```

```
LongJump<-read_table('https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat',skip=1)
```

```
## Parsed with column specification:
## cols(
##   ` -4 249.75 24 293.13 56 308.25 80 336.25` = col_character()
## )
```

```
LongJump<-as.data.frame(LongJump)
dim(LongJump)
```

```
## [1] 5 1
```

```

#We can see that the LongJump data only has 5 rows and one column. We will combine columns into Year and
LongJump<-LongJump %>% separate(colnames(LongJump),LETTERS[1:8],sep=" ")
#Separate items into 8 arbitrary columns
Year<-Long_Jump<-numeric()
#Make empty vectors of Year and Long_Jump
for(i in 1:4){
  Year<-c(Year,LongJump[,2*i-1])
  Long_Jump<-c(Long_Jump,LongJump[,2*i])
}
#Put multiple columns into each column
Year<-as.numeric(Year);Long_Jump<-as.numeric(Long_Jump)
#Convert each variable into numeric ones
Year<-Year+1900
#Since 0 means year 1900, we will add 1900 for each value
LongJump<-data.frame(Year,Long_Jump)
#Make it into a data frame
tail(LongJump)

```

```

##      Year Long_Jump
## 15 1976      328.50
## 16 1984      336.25
## 17 1988      343.25
## 18 1992      342.50
## 19   NA         NA
## 20   NA         NA

```

```

#We can see that there are NAs in last two rows.
LongJump<-LongJump[-c(19,20),]
#Remove last two rows

```

Through these steps we can import and clean the data. Unlike the first data, this data had repeated columns. So we can first assign these repeating columns as independent columns at first, and then merging the columns which should belong the same variable.

Following is some syntax to finish our analysis.

```
summary(lm(data=LongJump,Long_Jump~Year))
```

```

##
## Call:
## lm(formula = Long_Jump ~ Year, data = LongJump)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.4752  -4.1751  -0.6478   3.9988  24.0050
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -905.9381   150.7102  -6.011 1.81e-05 ***
## Year          0.6262     0.0774   8.091 4.79e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##

```

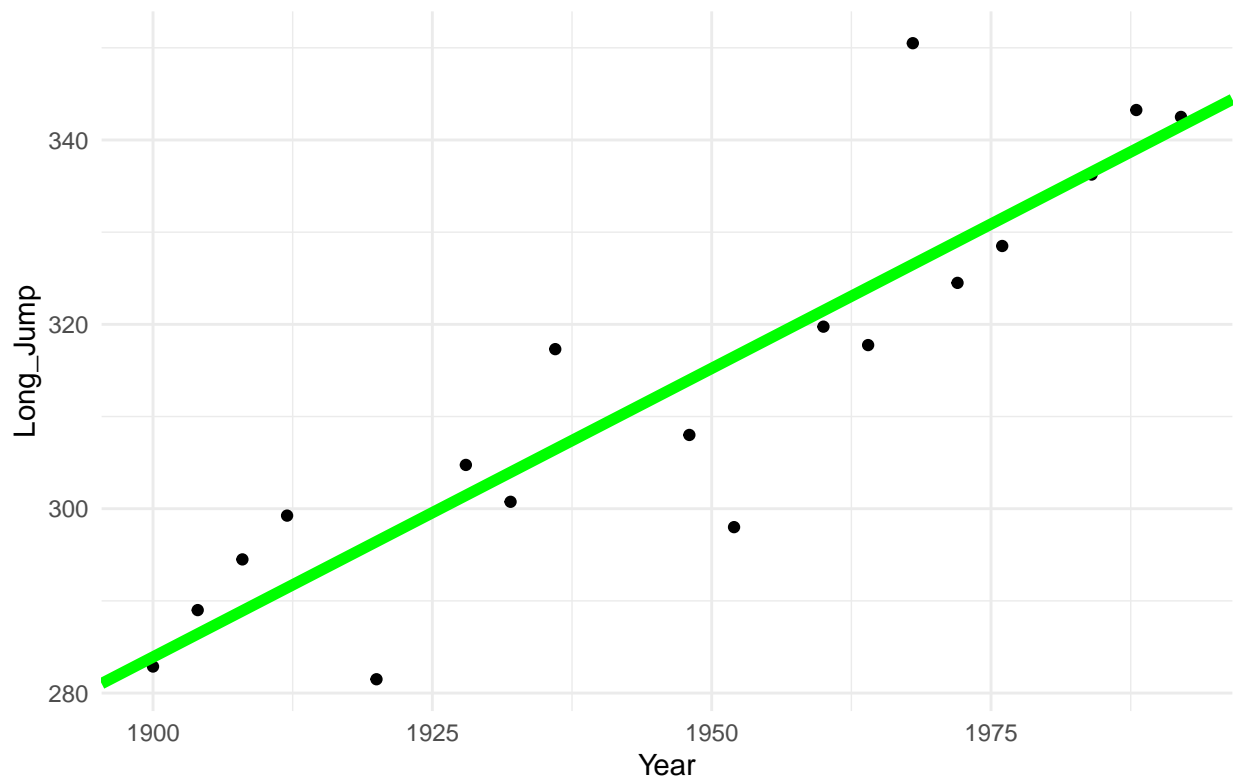
```
## Residual standard error: 9.7 on 16 degrees of freedom
## Multiple R-squared:  0.8036, Adjusted R-squared:  0.7913
## F-statistic: 65.46 on 1 and 16 DF,  p-value: 4.789e-07
```

```
#Summary of linear model in LongJump data. The independent variable will be Year and the target
#variable will be Long_Jump. We can see that the slope will be 0.6262 and both coefficients
#are significant.
cor(LongJump$Long_Jump, LongJump$Year)
```

```
## [1] 0.8964282
```

```
#Correlation between Long_Jump and Year variable. It is highly correlated, which is, it is likely the d
LMLongJump<-lm(data=LongJump, Long_Jump~Year)
#Assign the information about linear model of LongJump
ggplot(LongJump, aes(x=Year, y=Long_Jump))+geom_point()+ggtitle('Scatterplot of LongJump Data')+
geom_abline(slope=LMLongJump$coefficients[2], intercept=LMLongJump$coefficients[1], color='green', size=2)+
theme_minimal()+theme(plot.title = element_text(size=20, face="bold"))
```

## Scatterplot of LongJump Data



```
#We can see that the points are on ascending order, which is, those two variables are in a positive rel.
```

Since we had only two continuous variables for this data, we can use scatterplot to see the general relationship of the data. The points are aligned in a line, which we can assume that those two variables will have high correlation.

## Brain Body Data

As previous data, the columns are repeated. We can go similar steps as we have just before done on the data.

```
BrainBody<-read_table('https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat',s
```

```
## Parsed with column specification:
## cols(
##   X1 = col_character()
## )
```

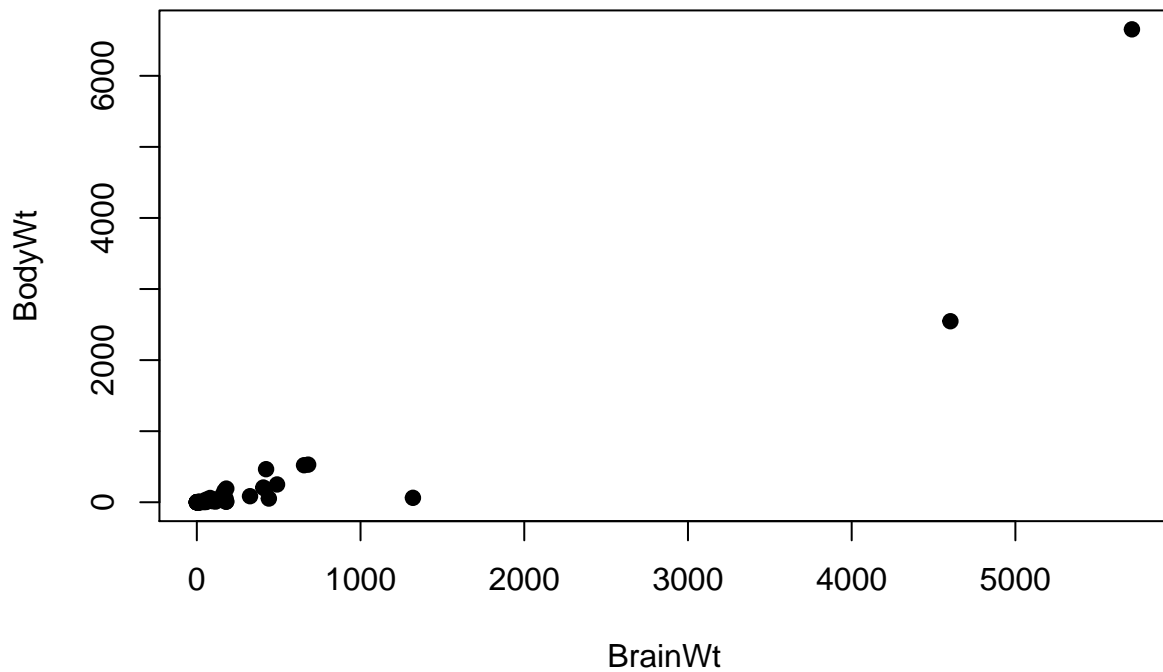
```
#Read data from Internet. For convenience, we will remove the first row and don't get the column names.
BrainBody<-as.data.frame(BrainBody)
#Convert format of the read data into data.frame
BrainBody<-BrainBody %>% separate(colnames(BrainBody),LETTERS[1:6],sep=' ')
#Separate items into individual row. We will combine the variables into BrainWt and BodyWt variables.
BrainWt<-BodyWt<-numeric()
for(i in 1:3){
  BodyWt<-c(BodyWt,BrainBody[,2*i-1])
  BrainWt<-c(BrainWt,BrainBody[,2*i])
}
#We will put the geneated variables into two variables
BodyWt<-as.numeric(BodyWt);BrainWt<-as.numeric(BrainWt)
#Make the two variable types to numeric
BrainBody<-data.frame(BodyWt,BrainWt)
#Put the variables into data.frame
tail(BrainBody)
```

```
##      BodyWt BrainWt
## 58 160.000   169.0
## 59   0.900     2.6
## 60   1.620    11.4
## 61   0.104     2.5
## 62   4.235    50.4
## 63      NA      NA
```

```
#We can check that there are NA values on the last row.
BrainBody<-BrainBody[-63,]
#Remove the last row of the data frame
```

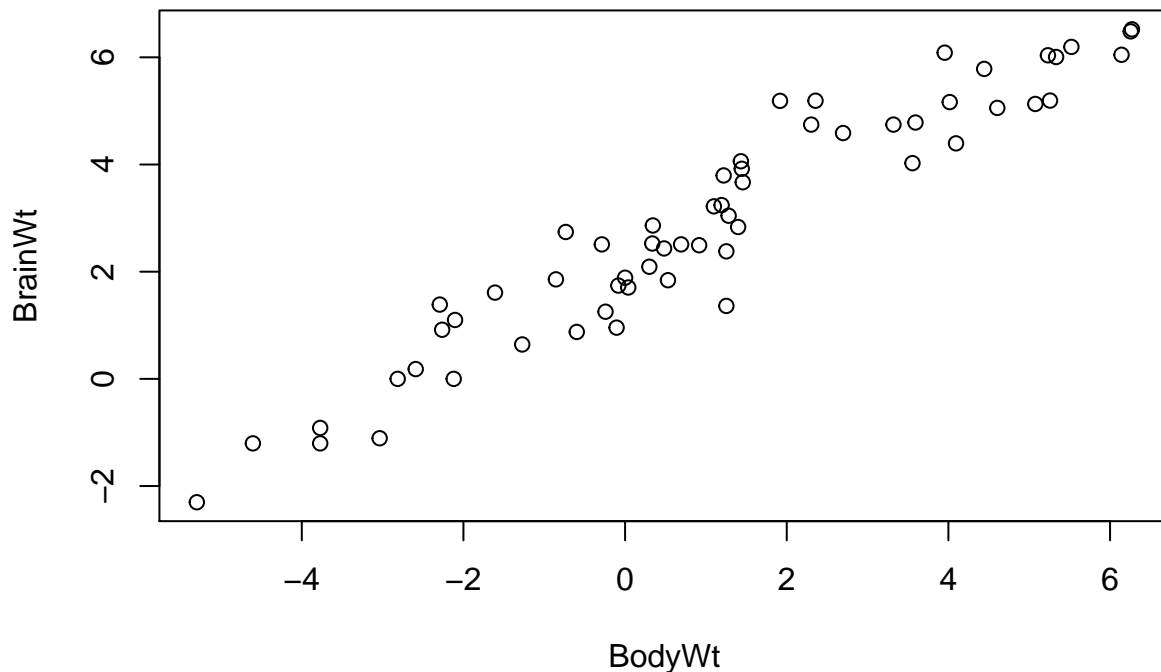
Through these steps we are able to import the data.

```
plot(data=BrainBody,BodyWt~BrainWt,pch=19)
```



```
#The scatter plot of Brain Weight and Body Weight. It is hard to see how the variables are related because
#of some extreme values. For analysis, we will only select values that are smaller than 1000 for each variable.
BrainBody2<-BrainBody %>% filter(BrainWt<1000&BodyWt<1000)
BrainBody2<-log(BrainBody2)
#Since the data has large numbers with very small numbers, we will put log on our data
plot(BrainBody2) #Since the extreme data are now modified, we can see the relationship between variables.
```





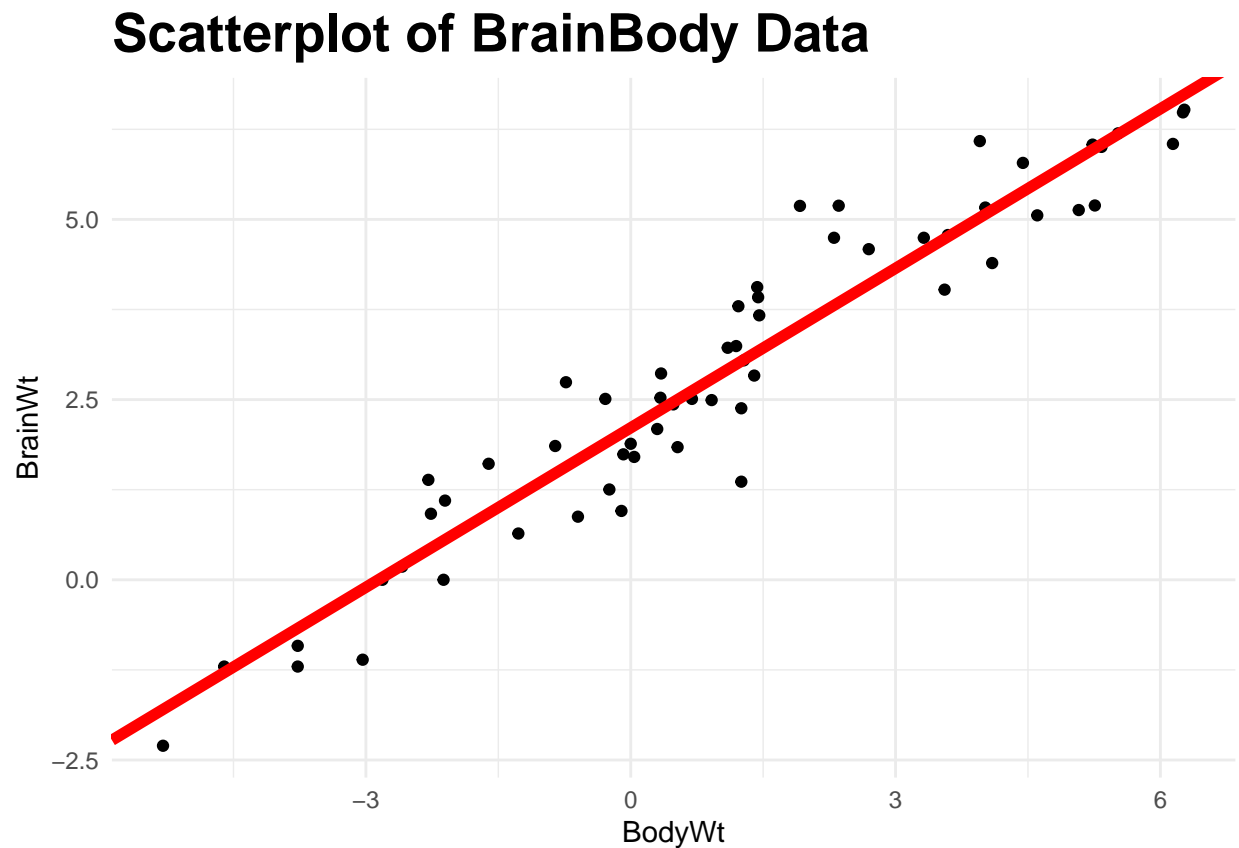
```
summary(lm(data=BrainBody2,BrainWt~BodyWt))
```

```
##
## Call:
## lm(formula = BrainWt ~ BodyWt, data = BrainBody2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.67191 -0.51590 -0.03111  0.48139  1.66464
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.10890    0.09179   22.97  <2e-16 ***
## BodyWt       0.73756    0.03007   24.53  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6618 on 57 degrees of freedom
## Multiple R-squared:  0.9135, Adjusted R-squared:  0.9119
## F-statistic: 601.7 on 1 and 57 DF,  p-value: < 2.2e-16
```

```
#This is the summary of simple linear model of the modified data. Although coefficient of intercept is
LMBrainBody2<-lm(data=BrainBody2,BrainWt~BodyWt)
#Assign the information about linear model of BrainBody2
ggplot(BrainBody2,aes(x=BodyWt,y=BrainWt))+geom_point()+ggtitle('Scatterplot of BrainBody Data')+

```

```
geom_abline(slope=LMBrainBody2$coefficients[2], intercept=LMBrainBody2$coefficients[1], color='red', size=2)
theme_minimal()+theme(plot.title = element_text(size=20, face="bold"))
```



*#We can see that the points are on ascending order, which is, those two variables are in a positive relationship.*

We can see that the data is positively related as the previous data. But we have to keep that in mind that there are some extreme values, which have too small or large values compared to some other values. To successfully do the analysis we had to remove some data points and use logarithms to deal with some rest of the extreme values. We can see the variables with logarithms are usually distributed in a line. Therefore, we can conclude that  $\log(\text{Long\_Jump})$  and  $\log(\text{Year})$  are highly correlated.

### Tomato data

Unlike the data we have seen on the other steps, there are multiple observations in one cell in tomato data. We first have to separate these repeated measurements into respective entries in matrix. Then, we will use each tomato brand as our variables.

```
Tomato<-read_table('https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat', skip=1, col_names=
```

```
## Parsed with column specification:
## cols(
##   X1 = col_character(),
##   X2 = col_character(),
```

```
## X3 = col_character(),
## X4 = col_character()
## )
```

```
#Since the first row will be the message we will read the data from the second line.
#Also, to assign variables on our own we will not put column names at first.
Tomato<-data.frame(Tomato)
#Convert the data type into data.frame
Tomato2<-Tomato[2:3,2:4]
#Tomato2 is a data frame with observations only.
Tomato3<-matrix(0,nrow=2,ncol=9)
#Make an empty matrix to put values in Tomato2
for(j in 1:2){
  for(i in 1:3){
    Tomato3[j,c(3*i-2,3*i-1,3*i)]<-word(Tomato2[j,i],1:3,sep=',')
    #Extract each observations into a matrix
  }
}
Ife<-Tomato3[1,];PursaEarlyDwarf<-Tomato3[2,]
#Extract the rows to make columns, which will be the name of brand
Ife<-as.numeric(Ife);PursaEarlyDwarf<-as.numeric(PursaEarlyDwarf)
#Change the variable types to numeric variables
Tomato<-data.frame(Ife,PursaEarlyDwarf)
#Put two variables together to make a data.frame
TomatoRow<-paste(rep(c('10k','20k','30k'),each=3),'_',rep(1:3,3),sep='')
#Making row names for data frame 'Tomato'
rownames(Tomato)<-TomatoRow
#Put row names to the data
```

We can consider rows as columns in our raw data with each 3 trial. The names of rows will be '(Raw data column)\_\_(trial number)', and columns will be each tomato brand.

```
summary(Tomato)
```

```
##           Ife           PursaEarlyDwarf
##  Min.      :15.30   Min.       : 8.10
##  1st Qu.:16.60   1st Qu.:10.10
##  Median :18.00   Median :12.70
##  Mean    :18.11   Mean     :12.02
##  3rd Qu.:19.20   3rd Qu.:13.70
##  Max.    :21.00   Max.      :15.40
```

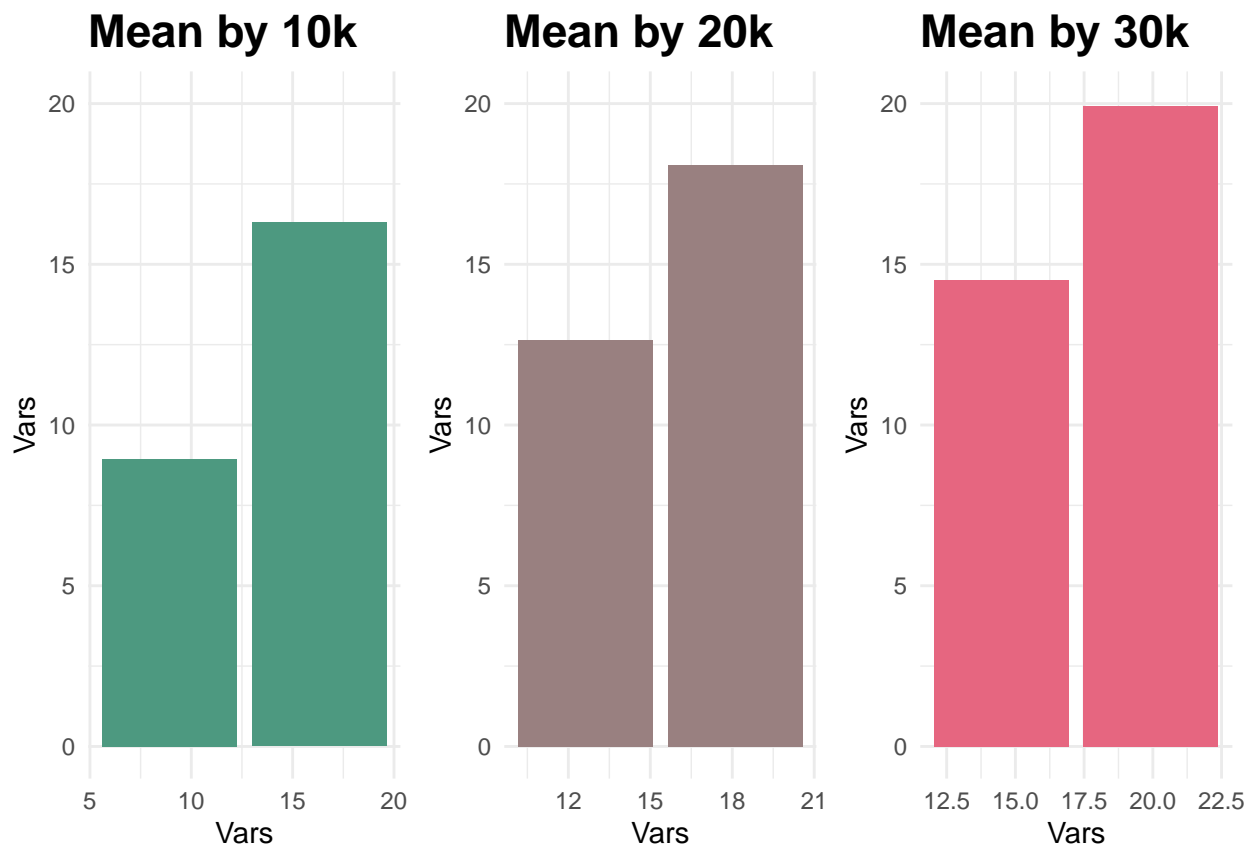
```
#Basic summary of variables in Tomato
t1<-Tomato[1:3,] %>% summarise(Ife_Mean=mean(Ife),Pursa_Mean=mean(PursaEarlyDwarf))
#Getting the mean of each brand from the first variable in our raw data.
t1<-data.frame(TName=c('Ife','Tomato'),Vars=as.numeric(as.matrix(t1)))
#Making into a 'ggplot barplot firendly' data frame. The first column will be the names of brand and se
#We will keep making these kinds of data frames on following process.
t2<-Tomato[4:6,] %>% summarise(Ife_Mean=mean(Ife),Pursa_Mean=mean(PursaEarlyDwarf))
t2<-data.frame(TName=c('Ife','Tomato'),Vars=as.numeric(as.matrix(t2)))
t3<-Tomato[7:9,] %>% summarise(Ife_Mean=mean(Ife),Pursa_Mean=mean(PursaEarlyDwarf))
t3<-data.frame(TName=c('Ife','Tomato'),Vars=as.numeric(as.matrix(t3)))
```

```

p1<-ggplot(t1,aes(x=Vars,y=Vars))+geom_bar(fill=rgb(0.3,0.6,0.5), stat = "identity")+ ggtitle('Mean by 10k')
theme_minimal()+theme(plot.title = element_text(size=17,face="bold"))+ylim(0,20)
#Make the data frame we made above into a ggplot barplot. This one will show the means in 10k variable.
#We will take the same process on other data frames as well.
p2<-ggplot(t2,aes(x=Vars,y=Vars))+geom_bar(fill=rgb(0.6,0.5,0.5), stat = "identity")+ ggtitle('Mean by 20k')
theme_minimal()+theme(plot.title = element_text(size=17,face="bold"))+ylim(0,20)
p3<-ggplot(t3,aes(x=Vars,y=Vars))+geom_bar(fill=rgb(0.9,0.4,0.5), stat = "identity")+ ggtitle('Mean by 30k')
theme_minimal()+theme(plot.title = element_text(size=17,face="bold"))+ylim(0,20)

grid.arrange(p1,p2,p3,layout_matrix=rbind(c(1,2,3))) #Put the bar plots in one window.

```



*#From the graph, we can learn that the bars get slightly higher for each trial groups. Also, PursaEarly*

Considering the variables in the raw data and each trials, we used mean of trials as the y-value for the barplot. From 10000, 20000, and 30000 variables in raw data the value became larger. Also, Pursa Early Dwarf brand had bigger values of observations than the ones from Ife brand.