Bhujade_Siddhant

Siddhant Bhujade

2022-12-28

Problem 1

Does Confidence Interval work?

```
# Clear the Memory
rm(list =ls())

# set seed
set.seed(4424)

# Normal distribution problem with N = 2,500, Mean =180,  and Std dev = 30.
normal_Dist = rnorm(2500,mean = 180, sd =30)

# Rounded the normal distribution to 1 decimal
normal_Dist_Rounded = round(normal_Dist, digits = 1)

# mean of the population
mean_Pop = mean(normal_Dist)
```

The mean of the population is **180.3155038**

```
# set seed again
set.seed(4424)

# Random sample of size of n=30
sample_30 = sample(normal_Dist_Rounded,size =30)

# The mean of this sample.
mean_Sample = mean(sample_30)
```

The mean of the sample is **175.7333333**

```
# The std error of this sample.
stdError_Sample = sd(sample_30)/sqrt(30)
```

The standard error of the sample is **6.5306129**

```r
# T-score for a Confidence level of 84.65%.
t_score = qt(0.92325,df = 29, lower.tail = TRUE)
```

The T-score for a Confidence level of 84.65% is **1.4656614**

```r
# The lower limit of the Confidence interval.
Lower_limit = mean_Sample - t_score * stdError_Sample;
```

The lower limit of the Confidence interval is **166.1616659**

```r
# The upper limit of the Confidence interval.
Upper_limit = mean_Sample + t_score * stdError_Sample;
```

The upper limit of the Confidence interval is **185.3050008**

```r
# d.If statement to see if the population mean falls within the Confidence interval.
if(mean_Pop > Lower_limit & mean_Pop < Upper_limit){
  cat("Population mean falls within the confidence interval")
} else {
  cat("Population mean doesnot falls within the confidence interval")
}
```

```
## Population mean falls within the confidence interval
```

Problem 2 (Set-1)

2- Way Annova

Three anti-bacteria creams were used on three age groups. The number of hours before the medicines started to show a noticeable effect are recorded in the table. Assume alpha = 0.05

```r
# Clear the Memory
rm(list =ls())

# Load readxl library
library(readxl)

# Read excel file
anova_data = read_excel("F22-6359-Test-3.xlsx", sheet="Set-1")

# Create individual vectors for each age column in the sheet
part1<-data.frame(Hours = anova_data[, 2], Medicine=anova_data[, 1], Age=rep("Young",30))
part2<-data.frame(Hours = anova_data[, 3], Medicine=anova_data[, 1], Age=rep("Middle_Age",30))
part3<-data.frame(Hours = anova_data[, 4], Medicine=anova_data[, 1], Age=rep("Senior",30))

# Rename columns
names(part1)[1] <- 'Hours'
names(part2)[1] <- names(part1)[1]
names(part3)[1] <- names(part1)[1]

# Combine everything and create a new dataset
```

```r
new_anova_data=rbind(part1, part2, part3);

# Performing the anova test and summarizing the result.
anova_test <- aov(new_anova_data$Hours ~ new_anova_data$Medicine +
                    new_anova_data$Age + new_anova_data$Medicine:new_anova_data$Age)
summary(anova_test)
```

```
##                                           Df Sum Sq Mean Sq F value  Pr(>F)
## new_anova_data$Medicine                    2   8414    4207   5.950 0.00388 **
## new_anova_data$Age                         2    661     331   0.468 0.62814
## new_anova_data$Medicine:new_anova_data$Age  4  10022    2506   3.543 0.01026 *
## Residuals                                 81  57280     707
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

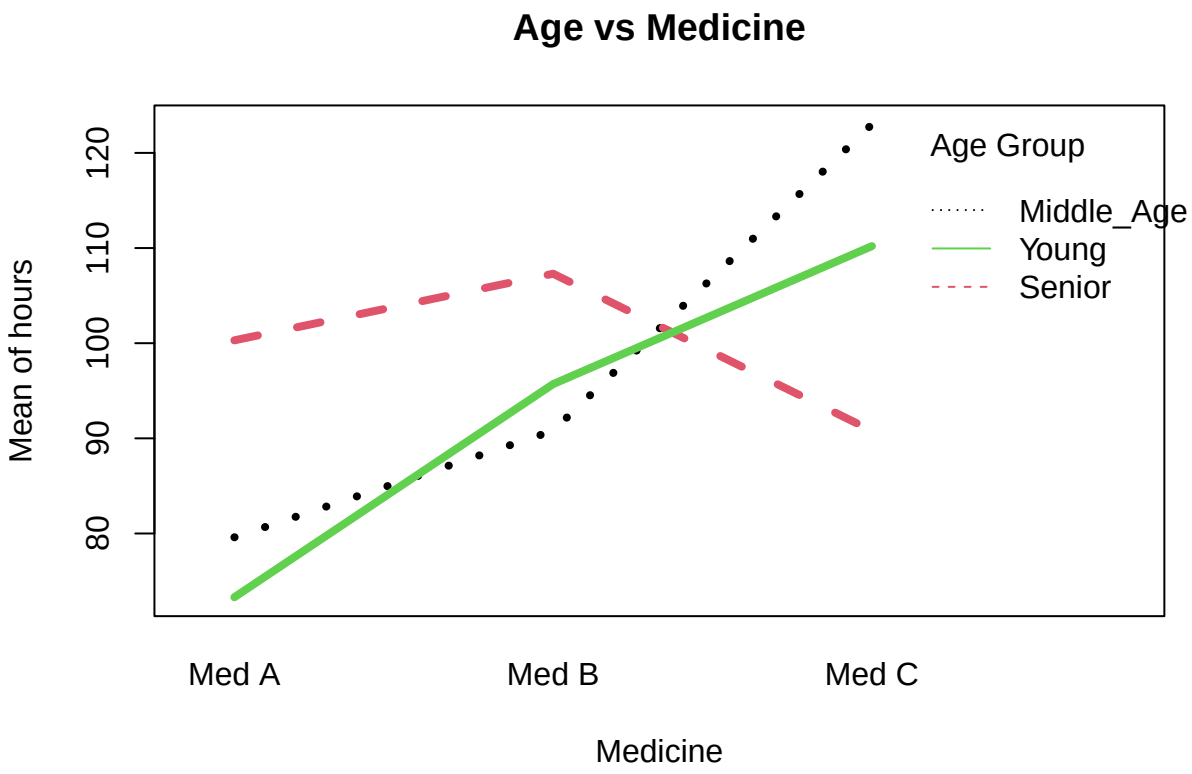
```r
# Creating mean table.
tapply(new_anova_data$Hours, list(new_anova_data$Medicine,new_anova_data$Age),mean)
```

```
##       Middle_Age Senior Young
## Med A       79.6  100.3  73.3
## Med B       90.8  107.3  95.7
## Med C      123.0   90.8 110.2
```

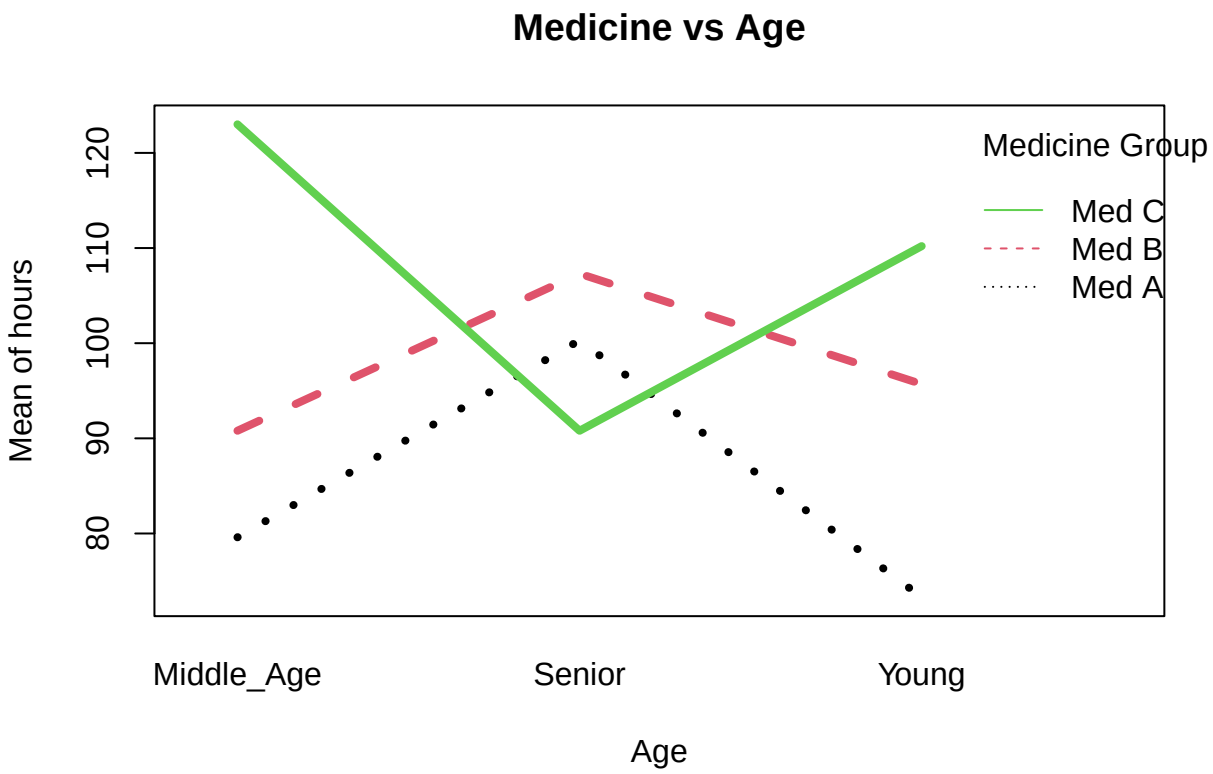
```r
tapply(new_anova_data$Hours, list(new_anova_data$Age,new_anova_data$Medicine),mean)
```

```
##            Med A Med B Med C
## Middle_Age  79.6  90.8 123.0
## Senior     100.3 107.3  90.8
## Young       73.3  95.7 110.2
```

```r
# Ploting interaction plots.
interaction.plot(new_anova_data$Medicine, new_anova_data$Age, new_anova_data$Hours,
                 trace.label = "Age Group",
                 lwd=4, col=1:3,xlab="Medicine", ylab ="Mean of hours",main="Age vs Medicine")
```




```
interaction.plot(new_anova_data$Age, new_anova_data$Medicine, new_anova_data$Hours,
                 trace.label = "Medicine Group",
                 lwd=4,col=1:3,xlab="Age", ylab ="Mean of hours", main="Medicine vs Age")
```



Problem 3 (Set-2)

Two sample t-test

Automobile Insurance companies consider many factors including the miles driven by a driver and the gender. The dataset consists of the reported miles (in thousands) driven by young drivers (25 years or less) in the previous year. One insurance company wants to know if there are any difference between the two genders.

```
# Clear the Memory
rm(list =ls())

# Load readxl library
library(readxl)

# Read excel file
data_ttest = read_excel("F22-6359-Test-3.xlsx", sheet="Set-2")

# Variance test to see if the two variances are equal
var.test(Distance ~ Gender, data = data_ttest)
```

```
##
##  F test to compare two variances
##
## data:  Distance by Gender
```

```
## F = 1.0246, num df = 99, denom df = 99, p-value = 0.904
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.6893942 1.5227966
## sample estimates:
## ratio of variances
##           1.024601
```

Since the p-value is **0.904** we can say variance are equal.

```
# t-test at alpha = 5%.
t.test(Distance ~ Gender,var.equal=TRUE, data = data_ttest)
```

```
##
##  Two Sample t-test
##
## data:  Distance by Gender
## t = -1.4193, df = 198, p-value = 0.1574
## alternative hypothesis: true difference in means between group Female and group Male is not equal to
## 95 percent confidence interval:
##  -1.3810709  0.2250709
## sample estimates:
## mean in group Female   mean in group Male
##                9.677               10.255
```

Problem 4 (Set-3)

Logistic Regression

A bank has collected a sample and is trying to see how various factors impact it's loan approvals. Divide Crdeit Scores by 10 and incomes by 1000 (in R) and perform Logistics Regression.

```
# Clear the Memory
rm(list =ls())

# Load readxl library
library(readxl)

# Read excel file
data_lr = read_excel("F22-6359-Test-3.xlsx", sheet="Set-3")

# Divide Credit_Scores by 10
Credit_Score = data_lr$`Credit scores`/10

# Divide income by 1000
Income = data_lr$Income/1000

# Divide Neighborhood Income by 1000
Neighborhood_Income = data_lr$`Neighborhood income`/1000

# Perform Logistic Regression Model
Logistic_Reg = glm(data_lr$`Loan Approved`~ Credit_Score+Income+Neighborhood_Income,
```

```
                   family = "binomial")

# Summarize the result of Logistic Regression
summary(Logistic_Reg)
```

```
##
## Call:
## glm(formula = data_lr$`Loan Approved` ~ Credit_Score + Income +
##     Neighborhood_Income, family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5276  -0.9104  -0.6602   1.1599   2.1914
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -9.23814    1.54545  -5.978 2.26e-09 ***
## Credit_Score         0.03141    0.01088   2.888  0.00387 **
## Income               0.04892    0.02028   2.412  0.01589 *
## Neighborhood_Income  0.09551    0.02615   3.653  0.00026 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 510.13  on 399  degrees of freedom
## Residual deviance: 468.78  on 396  degrees of freedom
## AIC: 476.78
##
## Number of Fisher Scoring iterations: 4
```

```
# Coefficient of Logistic Regression
coef(Logistic_Reg)
```

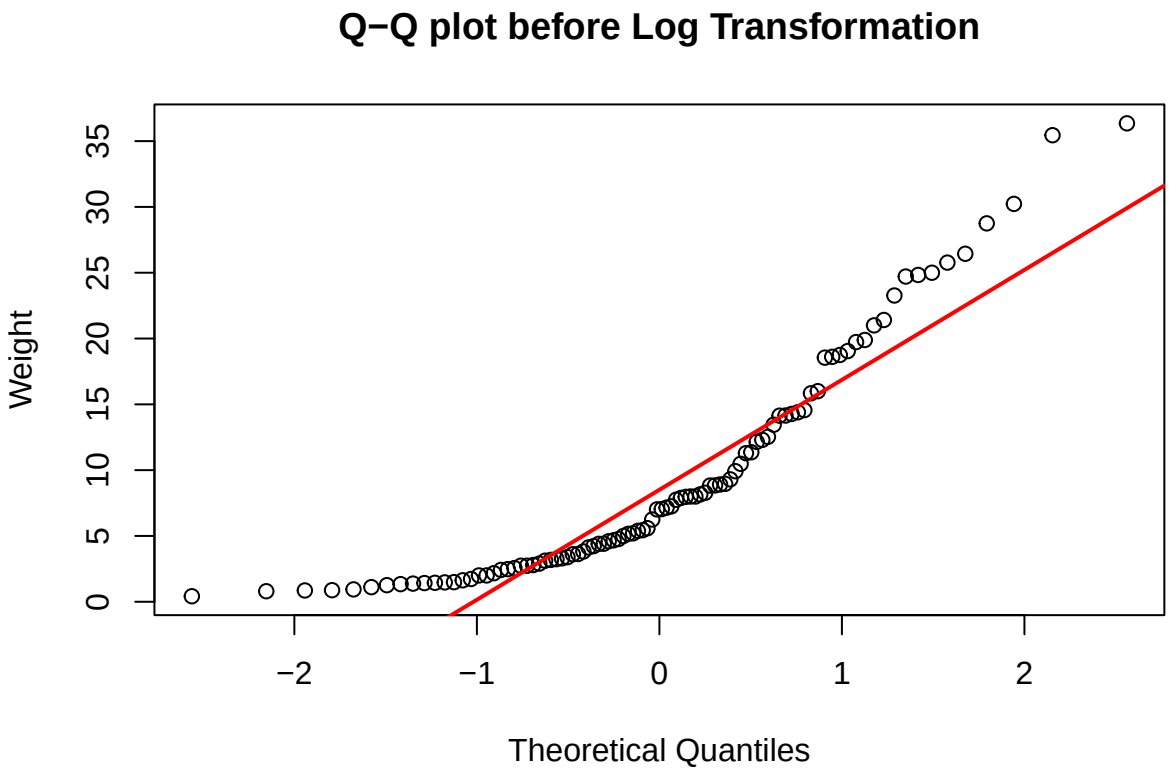
```
##         (Intercept)        Credit_Score              Income Neighborhood_Income
##         -9.23813634          0.03141410          0.04891621          0.09551381
```

Problem 5 (Set-4)

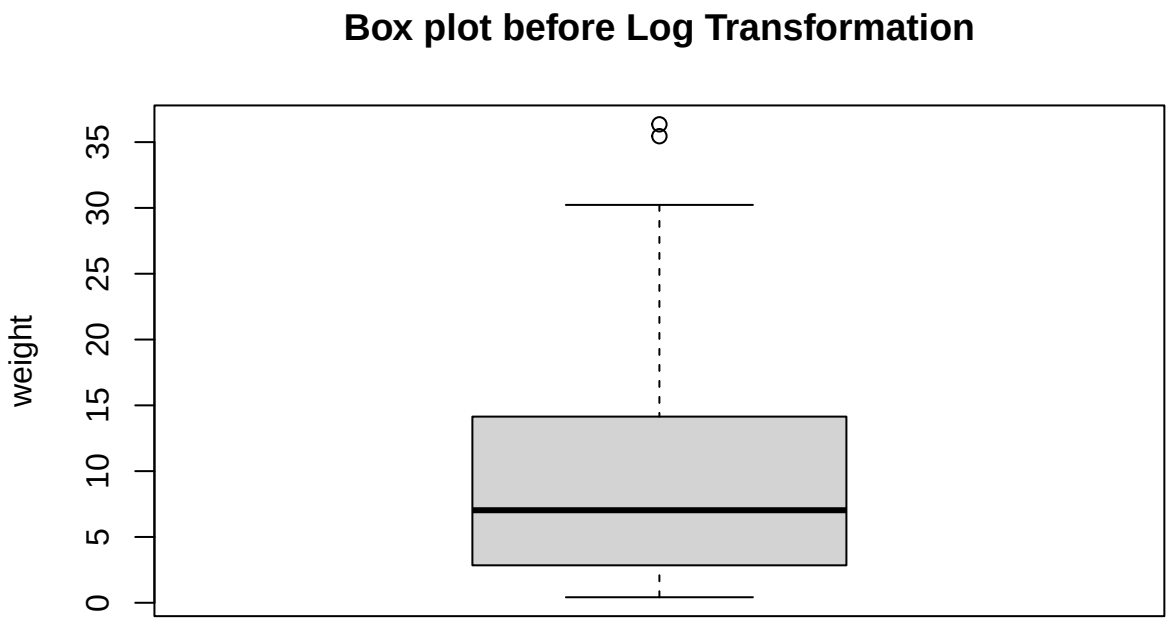
You've picked up a bunch of rocks from a rocky beach and want to estimate the weight of all the rocksat the beach with a Confidence level of 93.47%.

```
# Clear the Memory
rm(list =ls())

# Load readxl library
library(readxl)

# Read excel file
rock_data = read_excel("F22-6359-Test-3.xlsx", sheet="Set-4")
```

```r
# qqline plot before transformation
qqnorm(rock_data$Weight, main = "Q-Q plot before Log Transformation", ylab = "Weight" )
qqline(rock_data$Weight, col = "red", lwd = 2)
```




```r
# box plot before transformation
boxplot(rock_data$Weight,main = "Box plot before Log Transformation", ylab ="weight")
```



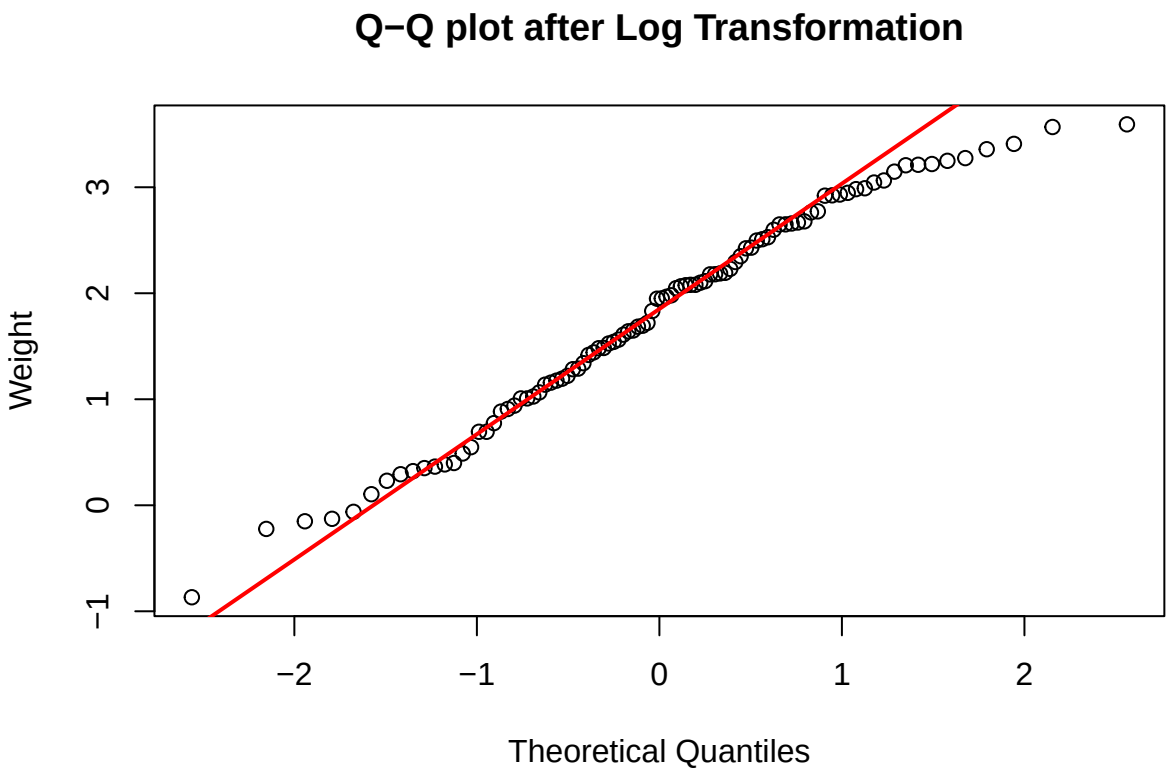

```
# skewness
library(moments)
data_skewness = skewness(rock_data$Weight)
```

The skewness of the data before transformation is **1.2339372**

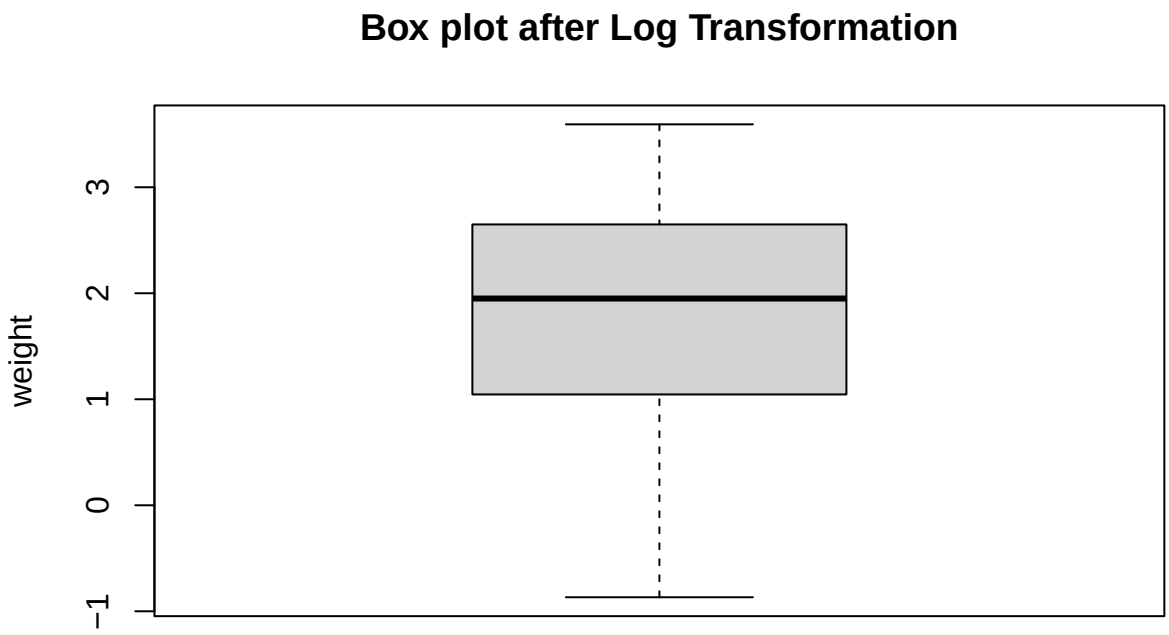
```
#conclusion before transformation
```

Looking at the Q-Q plot, Box plot and the skewness of the data it can be concluded that the distribution of the data is not normal

```
# Log transformation base e
 data_transformed = log(rock_data$Weight)

 # qqline plot after transformation
 qqnorm(data_transformed, main = "Q-Q plot after Log Transformation", ylab = "Weight" )
 qqline(data_transformed, col = "red", lwd = 2)
```




```
# box plot after transformation
boxplot(data_transformed,main = "Box plot after Log Transformation", ylab ="weight")
```




```
# skewness
skewness_transformed = skewness(data_transformed)
```

The skewness of the data after transformation is **-0.2867998**

```
# conclusion after transformation
```

Looking at the Q-Q plot, Box plot and the skewness of the data after transformation it can be concluded that the distribution of the data is normal

```
# mean of the transformed data
mean_weight = mean(data_transformed)
```

The mean of the transformed weight data is **1.7919505**

```
# standard deviation of the transformed data
sd_weight = sd(data_transformed)
```

The standard devation of the transformed weight data is **1.0267557**

```
# sample size
sample_size = length(data_transformed)
```

The sample size of the transformed weight data is **96**

```
# standard error
standard_error = sd_weight/sqrt(sample_size)
```

The standard error of the transformed weight data is **0.1047928**

```
# T-score for the 93.47% confidence interval
t_score = qt(0.96735,df = sample_size -1, lower.tail = TRUE)
```

The t-score of the transformed weight data is **1.8647753**

```
# upper limit
upper_limit = mean_weight + t_score * standard_error
```

The upper limit of the transformed weight data is **1.9873655**

```
# lower limit
lower_limit = mean_weight - t_score * standard_error
```

The lower limit of the transformed weight data is **1.5965354**

```
# Reverse transformation to get the Confidence Interval in Ounces.
reverse_upper_limit = exp(upper_limit)
```

The upper limit of the reverse transformed weight data is **7.2962865**

```
reverse_lower_limit = exp(lower_limit)
```

The lower limit of the reverse transformed weight data is **4.935902**

Problem 6 (Set-5)

Chi Square test

A random sample of 1100 U.S. adults were questioned regarding their political affiliation and opinion on a tax reform bill.Perform a test to see if the political affiliation and their opinion on a tax reform bill are independent.Get ChiSq Stats, P-values, etc. as required by the Online test.

```
# Clear the Memory
rm(list =ls())

# Load readxl library
library(readxl)

# Read excel file
chi_data = read_excel("F22-6359-Test-3.xlsx", sheet="Set-5")
```

```
## New names:
## * `` -> `...1`
```

```r
# chi square
chi_sq_matrix  = data.matrix(chi_data)
chi_sq_matrix = chi_sq_matrix[-4,-5]
chi_sq_matrix = chi_sq_matrix[,-1]
chisq.test(chi_sq_matrix)
```

```
##
##  Pearson's Chi-squared test
##
## data:  chi_sq_matrix
## X-squared = 8.9437, df = 4, p-value = 0.06252
```