

EDA 5장 과제

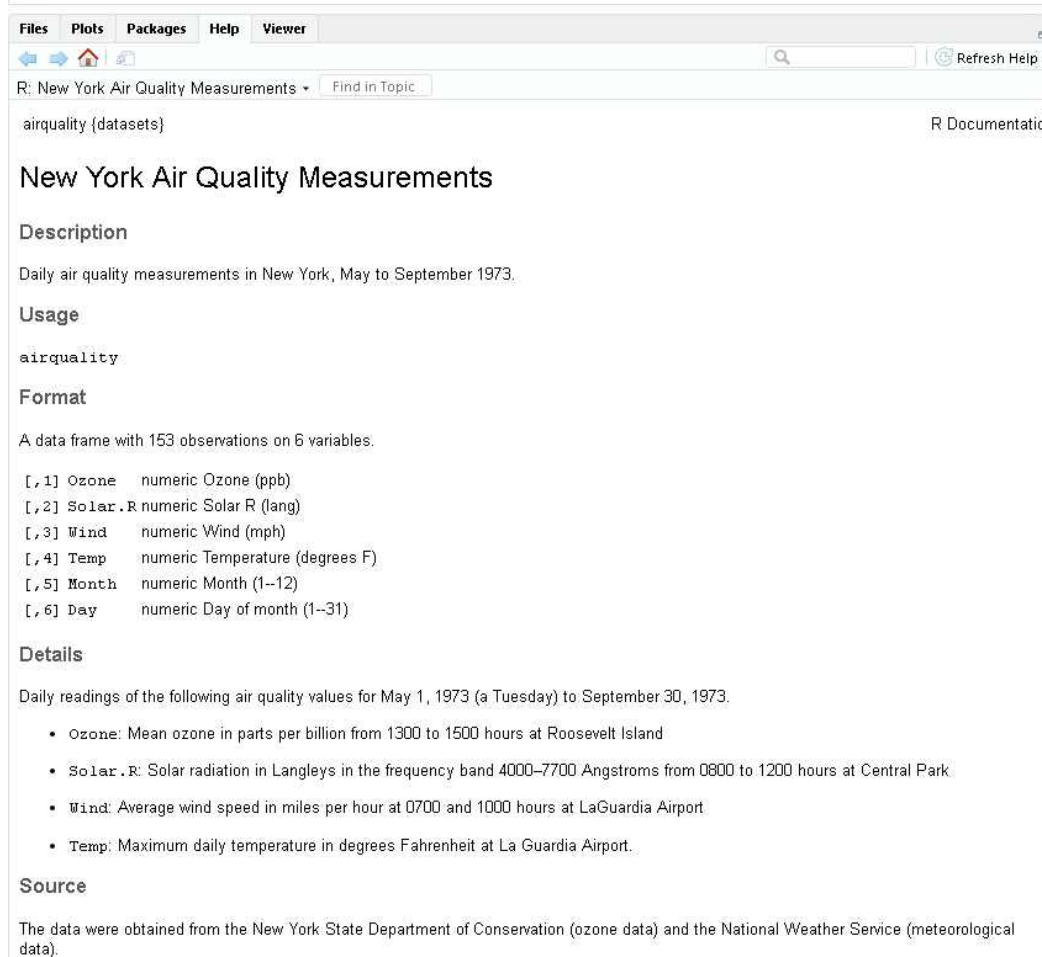
주의사항을 숙지하였고 모든 책임을 지겠습니다.

2019122041 송유진

1. airquality 자료에서 Ozone 자료를 대칭화하기 위한 변환 공식을 이용하여 변환 전과 변환 후를 비교하여라. 변환 전,후의 skewness 값을 비교하여라.

?airquality를 실행하여 데이터셋의 정보를 살펴보았다.

```
> ?airquality
> |
```



New York Air Quality Measurements

Description

Daily air quality measurements in New York, May to September 1973.

Usage

```
airquality
```

Format

A data frame with 153 observations on 6 variables.

[,1]	Ozone	numeric Ozone (ppb)
[,2]	Solar.R	numeric Solar R (lang)
[,3]	Wind	numeric Wind (mph)
[,4]	Temp	numeric Temperature (degrees F)
[,5]	Month	numeric Month (1-12)
[,6]	Day	numeric Day of month (1-31)

Details

Daily readings of the following air quality values for May 1, 1973 (a Tuesday) to September 30, 1973.

- **Ozone**: Mean ozone in parts per billion from 1300 to 1500 hours at Roosevelt Island
- **Solar.R**: Solar radiation in Langleys in the frequency band 4000-7700 Angstroms from 0800 to 1200 hours at Central Park
- **Wind**: Average wind speed in miles per hour at 0700 and 1000 hours at LaGuardia Airport
- **Temp**: Maximum daily temperature in degrees Fahrenheit at La Guardia Airport.

Source

The data were obtained from the New York State Department of Conservation (ozone data) and the National Weather Service (meteorological data).

airquality 데이터셋은 1973년 5월 1일부터 1973년 9월 30일까지 매일 뉴욕의 air quality를 측정한 것이고, Ozone, Solar.R, Wind, Temp, Month, Day 총 6개의 변수로 이루어져 있다.

```

> attach(airquality)
> Ozone
 [1] 41 36 12 18 NA 28 23 19 8 NA 7 16
[13] 11 14 18 14 34 6 30 11 1 11 4 32
[25] NA NA NA 23 45 115 37 NA NA NA NA NA
[37] NA 29 NA 71 39 NA NA 23 NA NA 21 37
[49] 20 12 13 NA NA NA NA NA NA NA NA NA
[61] NA 135 49 32 NA 64 40 77 97 97 85 NA
[73] 10 27 NA 7 48 35 61 79 63 16 NA NA
[85] 80 108 20 52 82 50 64 59 39 9 16 78
[97] 35 66 122 89 110 NA NA 44 28 65 NA 22
[109] 59 23 31 44 21 9 NA 45 168 73 NA 76
[121] 118 84 85 96 78 73 91 47 32 20 23 21
[133] 24 44 21 28 9 13 46 18 13 24 16 13
[145] 23 36 7 14 30 NA 14 18 20

> summary(Ozone)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00  18.00   31.50   42.13   63.25  168.00
   NA's
     37

```

summary() 함수를 사용하여 Ozone 데이터를 살펴보았다. NA값은 37개이고 Median은 31.5, 평균은 42.13, 그리고 최솟값은 1, 최댓값이 168임을 알 수 있다. 이때, 평균이 중앙값보다 크다. 168과 같은 극단적으로 큰 값에 의해 영향을 받았음을 알 수 있다.

다음으로, 줄기잎그림과 boxplot을 통해서 ozone 데이터의 분포를 살펴보았다.

```

> stem(Ozone)

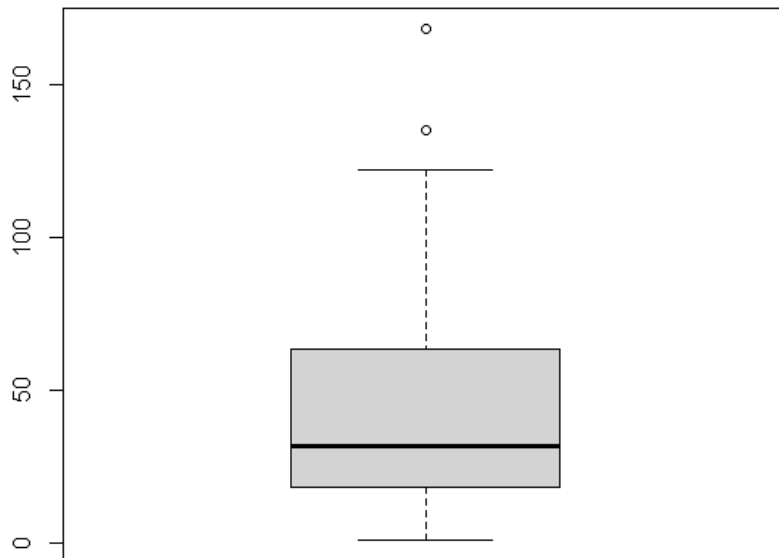
The decimal point is 1 digit(s) to the right of the |

 0 | 1467778999
 1 | 01112233334444666688889
 2 | 00001111233333334478889
 3 | 001222455667799
 4 | 01444556789
 5 | 0299
 6 | 134456
 7 | 13367889
 8 | 024559
 9 | 1677
10 | 8
11 | 058
12 | 2
13 | 5
14 |
15 |
16 | 8

```

줄기잎그림을 통해서 Ozone 데이터의 값들은 대부분 10~20대에 많이 분포하고 있음을 알 수 있다. 또한, 값이 커질수록 데이터의 수가 줄어드는 양상을 확인할 수 있다. 봉우리는 1-20에서 하나 그리고 6-80에서 하나 총 2개를 볼 수 있다. 또한, 168이라는 outlier를 확인할 수 있었다.

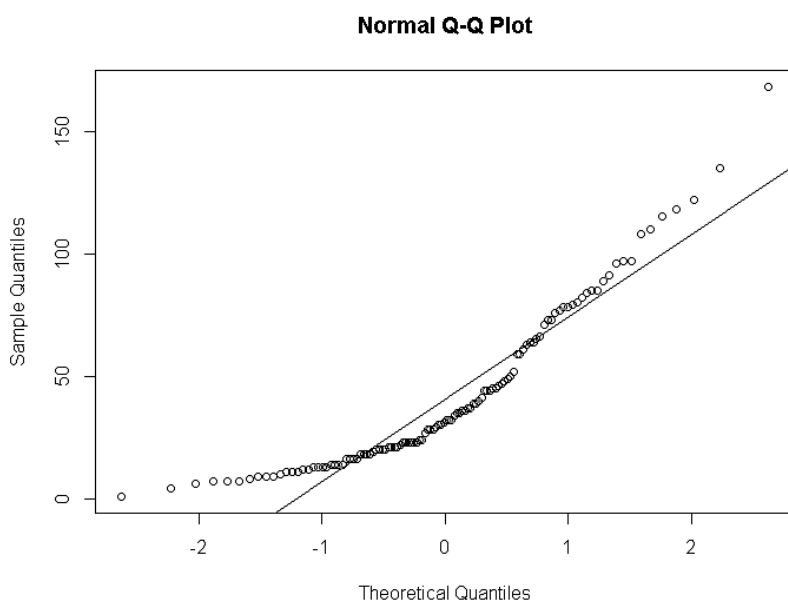
`boxplot(Ozone)`



상단의 결과를 통해 outlier 2개를 볼 수 있었고 median은 왼쪽에 위치함으로써 수염의 길이가 오른쪽으로 길게 뻗어있는 분포의 형태를 알 수 있었다. 또한, 자료는 symmetric하지 않고 오른쪽으로 꼬리가 길게 늘어져있는 양의 왜도를 가지고 있음을 알 수 있었다.

`qqnorm(Ozone)`

`qqline(Ozone)`



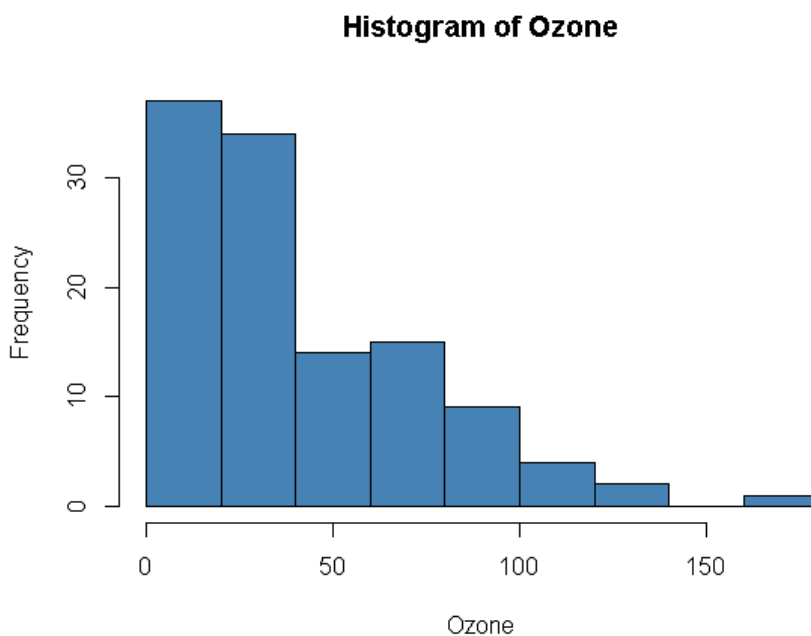
다음으로, Ozone 데이터로 qqplot을 그려보았다. 해당 결과를 통해 정규성에 위배됨을 알 수 있었다.

```

> hu <- fivenum(Ozone)[4]
> h1 <- fivenum(Ozone)[2]
> med <- fivenum(Ozone)[3]
> skew <- (((hu-med)-(med-h1))/((hu-med)+(med-h1)))
> skew
[1] 0.4065934

```

상단의 식으로 산출한 왜도는 0.41 정도로 양의 왜도를 수치로도 확인할 수 있었다.



histogram 으로 확인했을 때에도 양의 왜도의 양상을 뚜렷하게 확인해볼 수 있다.

다음으로, skewed되어있는 Ozone 데이터를 대칭 변환을 수행해보았다.

(1) p 산출 공식 활용

```

> boxplot(Ozone)
> ?airquality
> re_exp = function(x){
+   H1 <- fivenum(x)[2]
+   M <- fivenum(x)[3]
+   Hu <- fivenum(x)[4]
+   return(1-2*M*(Hu-M+H1-M)/((H1-M)^2+(Hu-M)^2))
+ }
>
> re_exp(Ozone)
[1] 0.03378238

```

상단의 코드를 통해 p의 값은 0.03378238임을 알 수 있었다. 해당 값을 적용하여 대칭 변환을

수행해보았다.

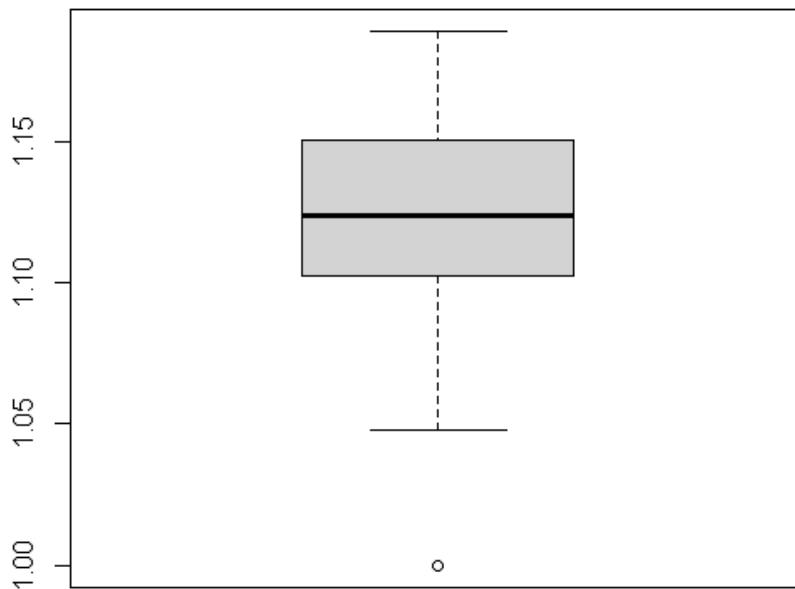
`stem(Ozone^0.03378238)`

```
> stem(Ozone^0.03378238)

The decimal point is 2 digit(s) to the left of the |

100 | 0
101 |
102 |
103 |
104 | 8
105 |
106 | 2888
107 | 3777
108 | 144488
109 | 111133338888
110 | 3333577778888
111 | 0222222338999
112 | 022344478899
113 | 0022346667789
114 | 0113889
115 | 0111256688999
116 | 0112245777
117 | 12456
118 | 09
```

`boxplot(Ozone^0.03378238)`



값들이 대부분 오른쪽으로 이동했음을 확인할 수 있다. 즉, 변환 후에는 왼쪽으로 꼬리가 길게 늘어져 음의 왜도를 갖는 것을 확인할 수 있다. Boxplot을 통해 1개의 outlier를 살펴볼 수 있는데 원래 데이터로 boxplot을 그렸을 때와 비교해보았을 때 변환 이전의 outlier 2개는 box 안으로 들어갔음을 유추할 수 있다.

다음으로 왜도의 정확한 수치를 산출해보았다.

```
[1] 0.1228674
> hu <- fivenum(Ozone^0.03378238)[4]
> hl <- fivenum(Ozone^0.03378238)[2]
> med <- fivenum(Ozone^0.03378238)[3]
> skew <- (((hu-med)-(med-hl))/((hu-med)+(med-hl)))
> skew
[1] 0.1228674
> |
```

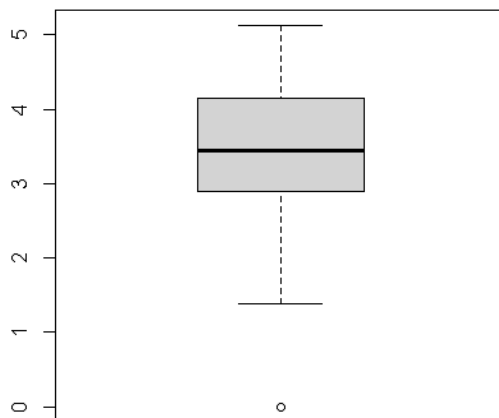
0.1228674로 변환 전보다 skewness값이 줄었음을 알 수 있다.

수업 시간에 배운 공식을 활용하여 산출한 값인 $p=0.03378238$ 으로 변환을 했지만 0.033이라는 값은 로그변환, 제곱근변환 등과 같이 어떠한 변환인지 쉽게 알 수 없다. 따라서, 0.033의 근사 값인 0으로 로그변환을 그리고 제곱근 변환을 수행하여 변환을 추가적으로 수행해보았다. 로 자료를 변환해보았다

(2) 로그변환

```
> stem(log(Ozone))
```

The decimal point is at the |

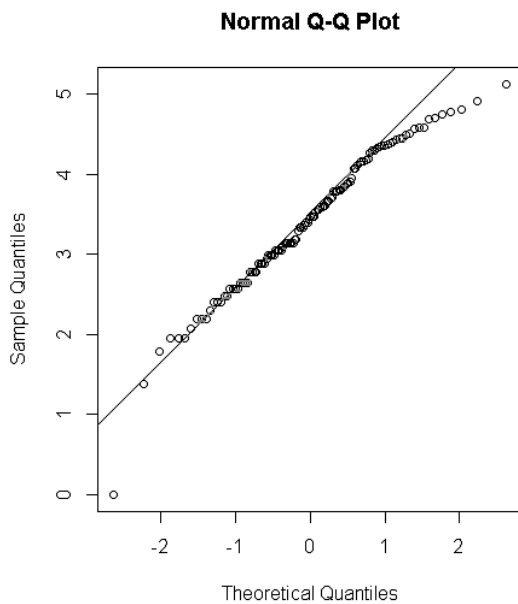


```
0 | 0
0 |
1 | 4
1 | 8999
2 | 12223444
2 | 5566666666888899999
3 | 0000000011111122233334444
3 | 555566666677778888889999
4 | 0111122223333344444444
4 | 55666777889
5 | 1
```

변환 전 raw data일 때와 $p=0.03378238$ 으로 변환했을 때보다 더 symmetric해졌음을 확인할 수 있다. 또한, 분포의 양상 또한 종 모양으로 더욱 대칭성을 가지게 되었음을 확인할 수 있었다.

```
qqnorm(log(Ozone))
```

```
qqline(log(Ozone))
```



qqplot을 그려봤을 때도 raw data일 때 그리고 $p=0.03378238$ 으로 변환했을 때에 비해 정규분포의 qqplot에 거의 근접한 양상을 살펴볼 수 있다. 즉, 변환 후의 정규성이 이전의 경우에 비해 만족하고 있음을 알 수 있다.

```
> hu <- fivenum(log(Ozone))[4]
> hl <- fivenum(log(Ozone))[2]
> med <- fivenum(log(Ozone))[3]
> skew <- (((hu-med)-(med-hl))/((hu-med)+(med-hl)))
> skew
[1] 0.1123698
```

skewness 또한 변환 전에 비해 줄어들었음을 확인할 수 있었다.

(3) 제곱근변환

다음으로, 제곱근변환을 수행해보았다.

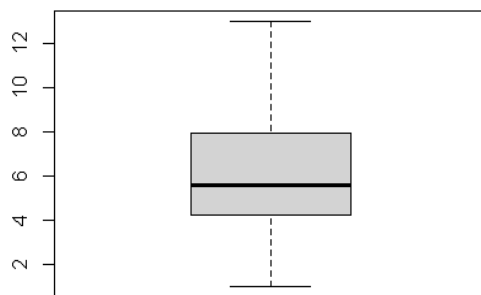
```
[1] 0.1123090
> stem(Ozone^0.5)

The decimal point is at the |

 1 | 0
 2 | 046668
 3 | 00023335566667777
 4 | 00002222455556666788888899
 5 | 23334556777899
 6 | 0011223466677899
 7 | 0127789
 8 | 0011455788899
 9 | 122245888
10 | 4579
11 | 06
12 |
13 | 0
```

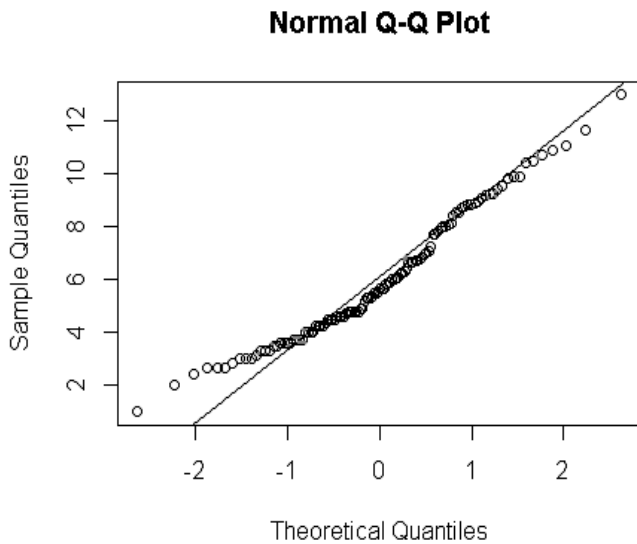
줄기잎그림을 통해 raw data일 때에 비해 더 대칭의 형태를 띠고 있음을 확인할 수 있다.

`boxplot(Ozone^0.5)`



boxplot에서는 이전에 존재하던 outlier 값이 없어졌음을 확인할 수 있다.


```
qqnorm(Ozone^0.5)
qqline(Ozone^0.5)
```



상단의 결과는 제곱근 변환 후의 qqplot이다.

변환 전보다 정규분포의 qqplot에 근접함을 알 수 있다. 즉, 변환 후에 정규성의 성질을 가지게 되었고 skewed된 정도도 감소했음을 알 수 있다.

```
> hu <- fivenum(Ozone^0.5)[4]
> hl <- fivenum(Ozone^0.5)[2]
> med <- fivenum(Ozone^0.5)[3]
> skew <- (((hu-med)-(med-hl))/((hu-med)+(med-hl)))
> skew
[1] 0.2648021
```

제곱근 변환에서도 skewness는 줄어들었지만 로그변환에 비해서는 skewed된 정도가 크다는 것을 알 수 있었다. 즉, Ozone 데이터의 경우 로그변환으로 변환을 수행하는 것이 적절해보인다는 결론을 내릴 수 있었다.

2. 2021년 삼성전자 주식 가격을 인터넷에서 찾아 줄기있 전시와 상자그림을 그려 대칭성을 점검하고 필요하다면 대칭화하는 변환을 시행착오방법으로 찾아라. 어떤 시행 착오로 찾아 갔는 지 중간 과정을 전부 보여라. 변환 후의 줄기있 전시와 상자그림을 그려 전,후를 비교하여라. 변환 전,후의 skewness 값을 비교하여라.

데이터는 아래의 웹사이트를 통해 다운로드 한 후 분석을 수행했음을 밝힙니다.

<https://finance.yahoo.com/quote/005930.KS/history?period1=1609459200&period2=1640908800&interval=1d&filter=history&frequency=1d&includeAdjustedClose=true>

```
> setwd("D:\\2022-1(3-2)\\2022-01_탐자분\\Data")
> stock <- read.csv("samsung_stockprice.csv", header=T)
> head(stock)
      Date Close
1 2021-01-04 83000
2 2021-01-05 83900
3 2021-01-06 82200
4 2021-01-07 82900
5 2021-01-08 88800
6 2021-01-11 91000
> tail(stock)
      Date Close
243 2021-12-23 79900
244 2021-12-24 80500
245 2021-12-27 80200
246 2021-12-28 80300
247 2021-12-29 78800
248 2021-12-30 78300
```

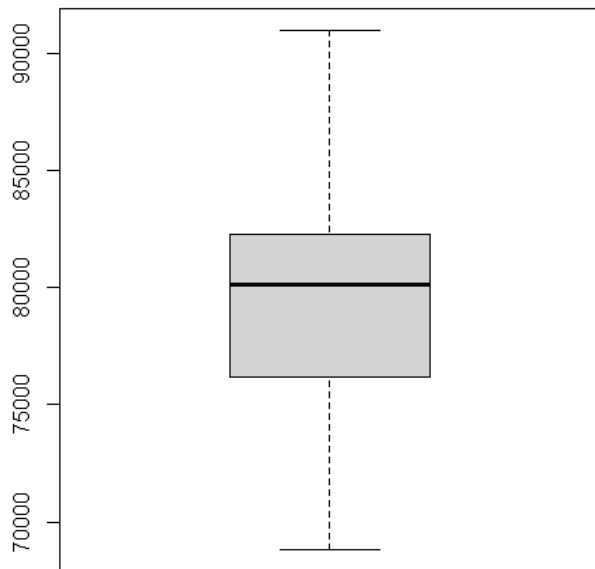
stock 데이터는 Date 즉, 기준일자 그리고 Close 즉, 종가로 2개의 column으로 이루어져 있다.

```
attach(stock)
stock_close <- as.numeric(close)
stock_close

summary(stock_close)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
68800  76250   80100   79156   82225   91000
length(stock_close)
1] 248
```

stock_close의 길이는 248개로서 248일치의 삼성전자 종가 데이터임을 알 수 있다.

`boxplot(stock_close)`



boxplot 결과를 통해 주식 가격은 왼쪽으로 꼬리가 긴 형태를 띠고 있음을 알 수 있었다. 즉, 왼쪽으로 skewed 된 데이터임을 알 수 있었다. outlier는 볼 수 없었다.

`stem(stock_close)`

```
> stem(stock_close)

The decimal point is 3 digit(s) to the right of the |

68 | 8
69 | 04899
70 | 112222223445666677
71 | 1233334556
72 | 2337
73 | 12379
74 | 1123446689
75 | 3336678
76 | 0113333667889
77 | 0001233444678
78 | 01235555588
79 | 00023344556667778889999
80 | 000111112234555566788999
81 | 00001222244455556667788899999
82 | 000000111122223344556667888899999
83 | 000222355667999
84 | 0001246789
85 | 03466
86 | 078
87 | 02
88 | 018
89 | 477
90 | 6
91 | 0
```

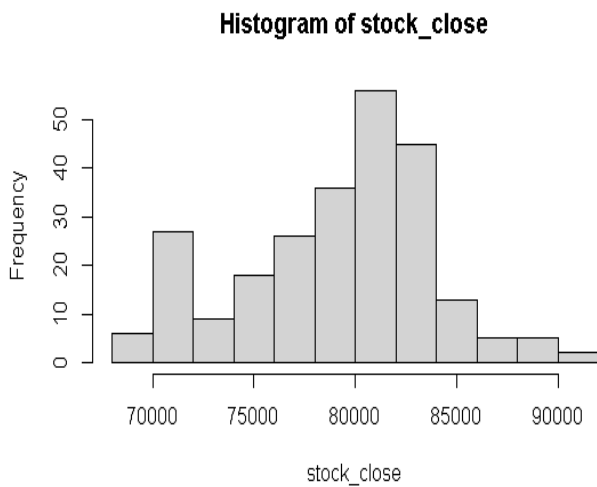
줄기잎그림을 통해 봉우리가 2개인 이봉분포의 형태를 볼 수 있었고 7만원대 그리고 82000원대

에서 봉우리를 가짐을 알 수 있었다. 줄기잎그림에서도 왼쪽으로 skewed되어있는 양상을 살펴볼 수 있었다.

```
> hu <- fivenum(stock_close)[4]
> h1 <- fivenum(stock_close)[2]
> med <- fivenum(stock_close)[3]
> skew <- (((hu-med)-(med-h1))/((hu-med)+(med-h1)))
> skew
[1] -0.2892562
```

skewness를 산출해본 결과 -0.289 정도로 음의 왜도를 가지고 있음을 알 수 있었다.

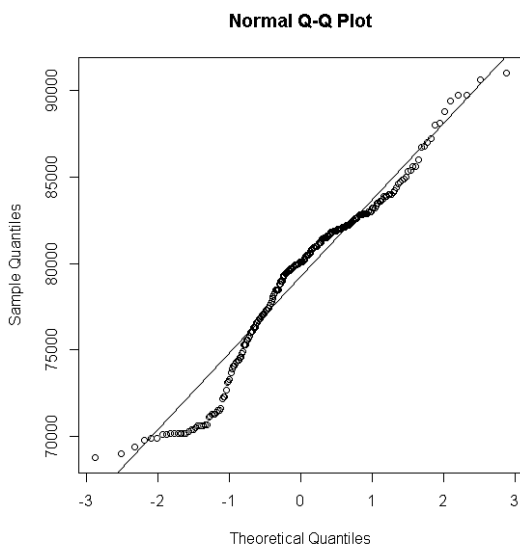
`hist(stock_close)`



히스토그램을 그려본 결과 왼쪽으로 치우친 분포의 양상을 다시 한 번 확인해볼 수 있었다.

`qqnorm(stock_close)`

`qqline(stock_close)`



qqplot을 그려보았을 때 정규성을 벗어나고 있음을 확인할 수 있었다.

(1) Symmetrizing re-expression with p

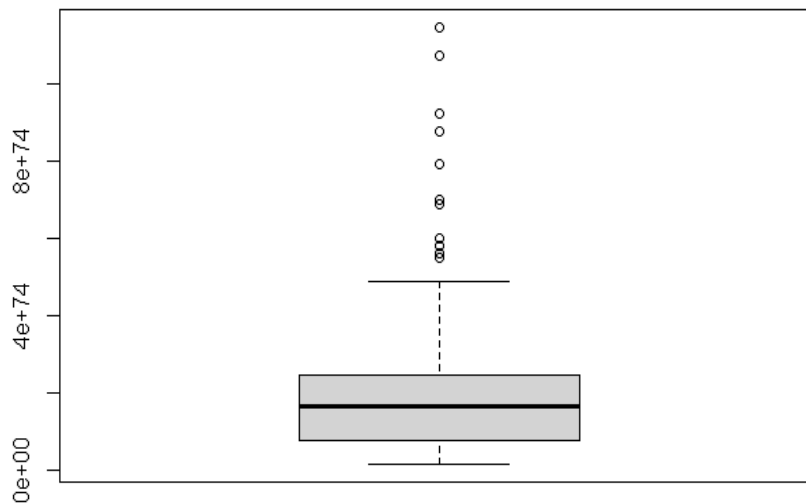
여러 가지 변환을 수행하기 전, 우선적으로 p의 값을 찾고 이를 적용해보았다.

```
> re_exp = function(x){
+   H1 <- fivenum(x)[2]
+   M <- fivenum(x)[3]
+   Hu <- fivenum(x)[4]
+   return(1-2*M*(Hu-M+H1-M)/((H1-M)^2+(Hu-M)^2))
+ }
>
> re_exp(stock_close)
[1] 15.13589
```

p=15.13589 라는 값이 도출되었다.

다음으로, 아래의 코드를 활용하여 boxplot과 줄기잎그림을 살펴보았다.

```
p = 15.13589
boxplot(stock_close^p)
stem(stock_close^p)
```

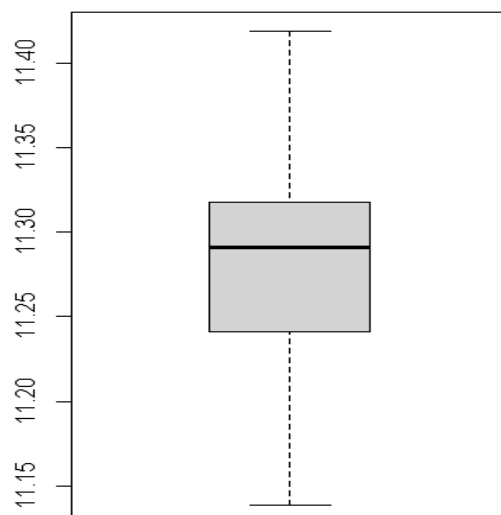


The decimal point is 74 digit(s) to the right of the |

눈에 띄는 변화는 outlier가 변환 이전에 비해 매우 많이 생겼다는 점이다. 수업 시간을 통해 re-expression의 결과로서 outlier의 생성을 알 수 있었는데 해당 결과를 통해 이를 확인할 수 있었다.

(2)로그변환

```
boxplot(log(stock_close))
```



boxplot 결과를 살펴보면 로그변환을 수행했을 때 median은 upper hinge 부근에 위치하고 있지만 로그 변환 수행 이후 수염의 길이가 조금 더 대칭의 형태를 띠고 있음을 확인할 수 있다.

```
stem(log(stock_close))
```

The decimal point is 2 digit(s) to the left of the |

```
1112 | 9
1114 | 2835588999999
1116 | 1223555566235556779
1118 | 7994
1120 | 01280335677
1122 | 003499933568
1124 | 00222266899022234557779
1126 | 1246781111155777
1128 | 011224455566677799990001111122456666778
1130 | 0011122223555577788880001122233333444444666677778899
1132 | 112223444455555777999033445777999
1134 | 013678904577
1136 | 20146
1138 | 564
1140 | 14449
```

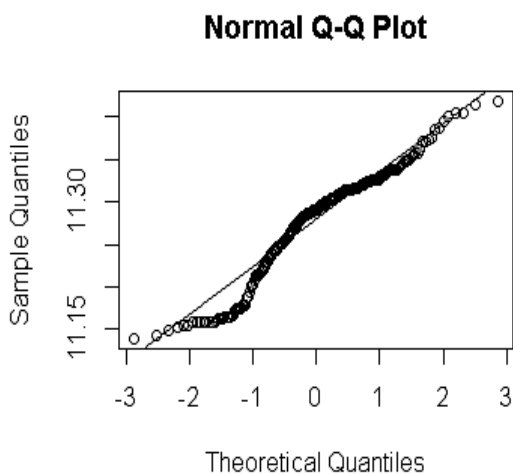
로그 변환을 수행한 이후 줄기잎그림을 그려본 결과, 여전히 left skewed 되어 있는 모습을 볼 수 있었다.

```
> hu <- fivenum(log(stock_close))[4]
> h1 <- fivenum(log(stock_close))[2]
> med <- fivenum(log(stock_close))[3]
> skew <- (((hu-med)-(med-h1))/((hu-med)+(med-h1)))
> skew
[1] -0.3066375
```

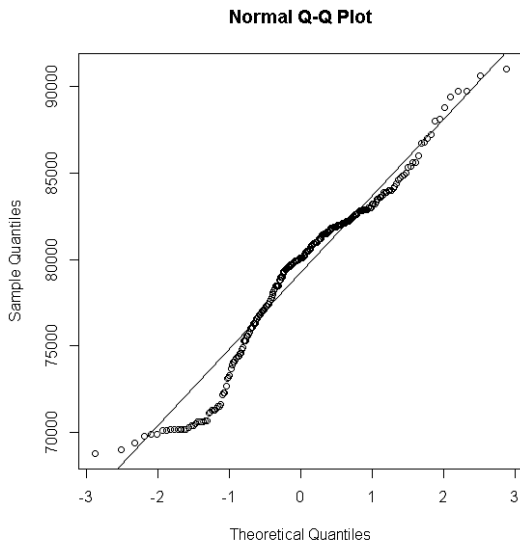
변환 이전의 skewness는 -0.289 정도였는데 변환 후에는 -0.3 정도로 오히려 skewed의 정도가 커졌음을 알 수 있었다.

```
qqnorm(log(stock_close))
```

```
qqline(log(stock_close))
```



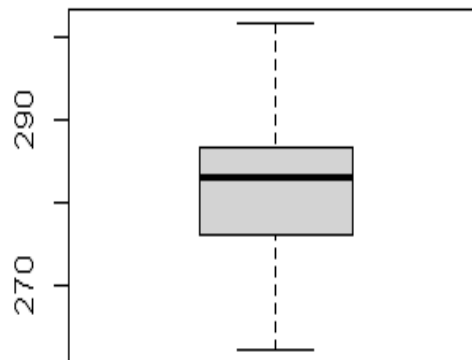
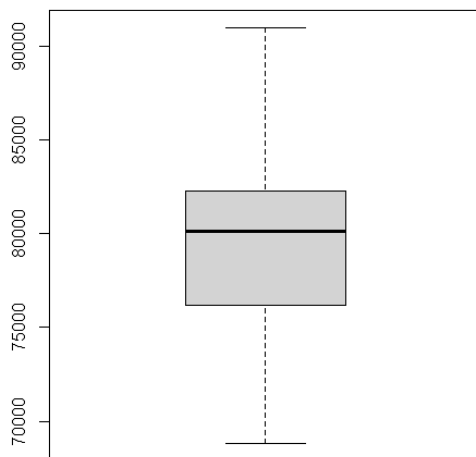
qqplot을 그려본 결과, 아래의 변환 이전 qqplot과 비교해보았을 때 여전히 정규성을 띠고 있다고 보기에는 어려웠다.



(3)제곱근변환

`boxplot(stock_close)`

`boxplot(stock_close^0.5)`



boxplot 결과를 살펴보면 제곱근 변환을 수행했을 때 median은 여전히 upper hinge 부근에 위치하고 있다. 그러나 제곱근 변환 수행 이후 수염의 길이가 조금 더 대칭의 형태를 띠고 있음을 확인할 수 있다.


```
> stem(stock_close^0.5)
```

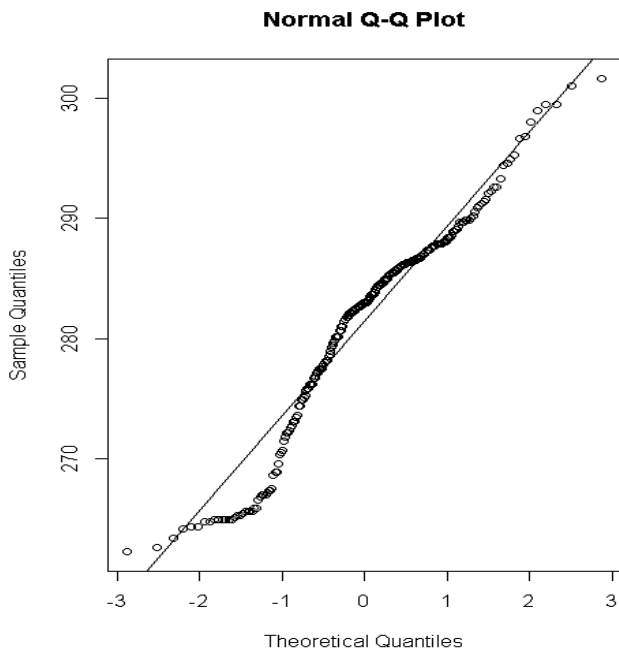
The decimal point is at the |

```
262 | 374
264 | 244880000001335777799
266 | 680002446
268 | 7996
270 | 46758
272 | 2246881157
274 | 4440013799
276 | 222288911355578
278 | 002226793568
280 | 222227711146688
282 | 001113335557777888000002245777799
284 | 133444666680000333555577788
286 | 0002222444444555577779911224446777799999
288 | 111444600113777888
290 | 02590245
292 | 12663
294 | 4603
296 | 68
298 | 0055
300 | 07
```

```
> hu <- fivenum(stock_close^0.5)[4]
> h1 <- fivenum(stock_close^0.5)[2]
> med <- fivenum(stock_close^0.5)[3]
> skew <- (((hu-med)-(med-h1))/((hu-med)+(med-h1)))
> skew
[1] -0.297963
```

변환 이전의 skewness는 -0.289 정도였는데 제곱근 변환 후에는 -0.297963으로 skewed된 정도가 조금 늘어났음을 확인할 수 있었다.

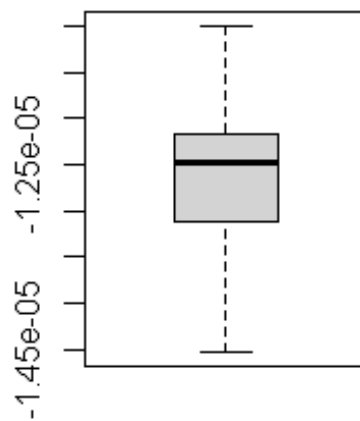
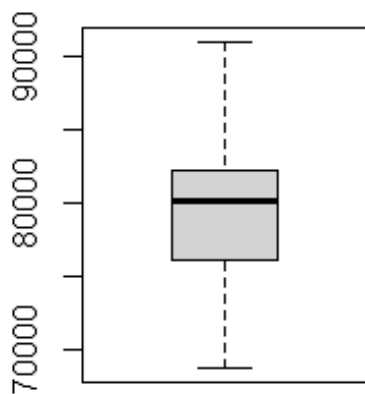
```
qqnorm(stock_close^0.5)
qqline(stock_close^0.5)
```



제곱근 변환 또한 qqplot으로 확인해봤을 때 정규성을 만족한다고 보기에는 어려움이 있었다.

(4) $-\frac{1}{x}$ 변환

```
par(mfrow=c(1,2))
boxplot(stock_close)
boxplot(-1/(stock_close))
```



다음으로, 역수 변환을 수행해보았다.

(역수 변환의 경우 데이터의 순서가 바뀌기 때문에 (-)를 붙여주었다.)

boxplot을 그려본 결과 여전히 upper hinge에 median이 위치하고 있었다.

또한, 변환 이전의 경우 위쪽 수염이 아래쪽 수염에 비해 조금 더 길었는데 변환 후에는 아래쪽 수염이 위쪽 수염에 비해 조금 더 길어보임을 확인했다.

```
> stem(-1/(stock_close))

The decimal point is 7 digit(s) to the left of the |

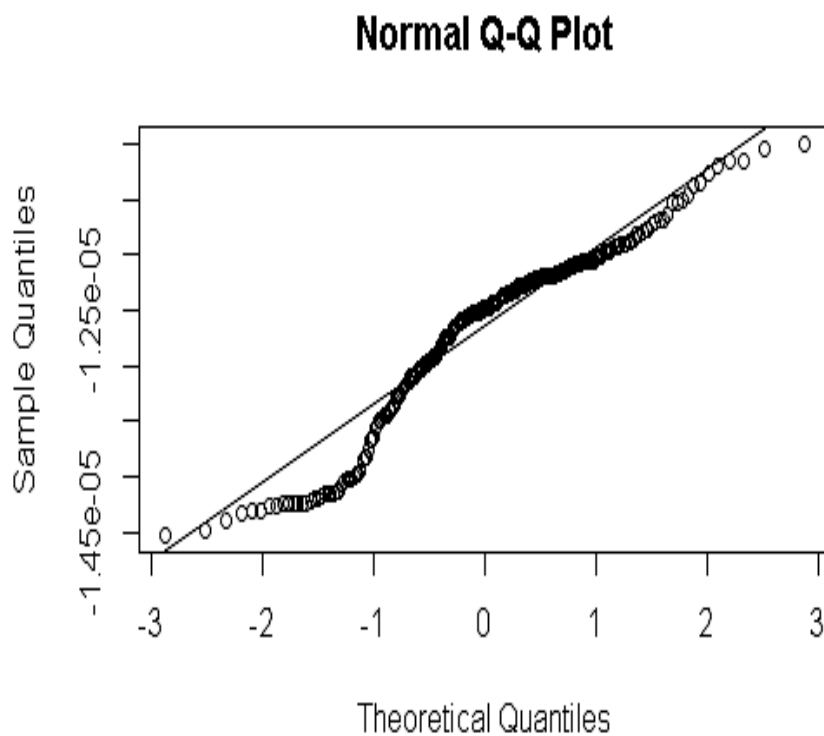
-144 | 391
-142 | 3117755555200
-140 | 8666644643331
-138 | 997533
-136 | 6864
-134 | 7300864400
-132 | 75888331
-130 | 96441111554220
-128 | 999754422297520
-126 | 974444499666311
-124 | 9988666555332222000888887754222211
-122 | 98866655553222299977775554422211111000000
-120 | 8888777755442211198888666665552220
-118 | 8866522200098521
-116 | 98621883
-114 | 3297
-112 | 656
-110 | 9554
-108 | 9
```

줄기잎그림을 그려본 결과이다. 여전히 left skewed 되어있음을 확인했다.

```
skewness <- function(x){
  hu <- fivenum(x)[4]
  hl <- fivenum(x)[2]
  med <- fivenum(x)[3]
  skew <- (((hu-med)-(med-hl))/((hu-med)+(med-hl)))
  return(skew)
}
skewness(-1/(stock_close))
1] -0.3238837
skewness(stock_close)
1] -0.2892562
```

다음으로 skewness를 수치로 확인해보았다. 변환 후 오히려 skewed된 정도가 커졌음을 알 수 있었다.

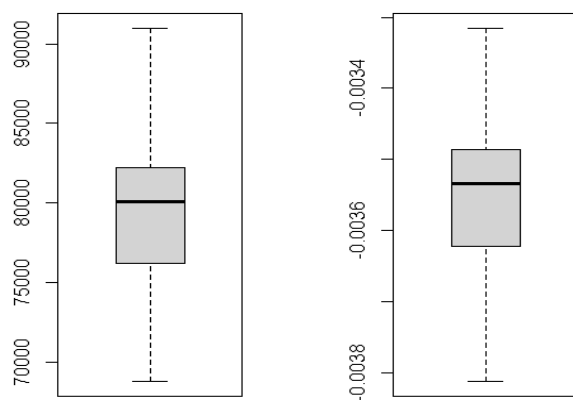
```
qqnorm(-1/(stock_close))
qqline(-1/(stock_close))
```



qqplot으로 확인해보았을 때도 정규성은 여전히 없음을 알 수 있었다.

(5) - $\frac{1}{\sqrt{(x)}}$ 변환

```
par(mfrow=c(1,2))
boxplot(stock_close)
boxplot(-1/sqrt(stock_close))
```



다음으로 $-\frac{1}{\sqrt{(x)}}$ 변환을 수행해보았다. boxplot을 확인해보았을 때 여전히 대칭의 형태를 보기엔 어려웠다.

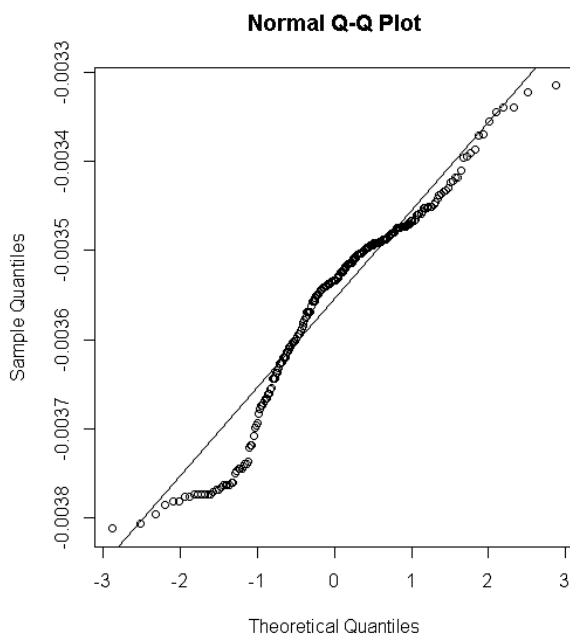
```
> skewness(-1/sqrt(stock_close))
[1] -0.3152782
> stem(-1/sqrt(stock_close))

The decimal point is 5 digit(s) to the left of the |

-380 | 27
-378 | 6522
-376 | 774444442996444411
-374 | 08555200
-372 | 72
-370 | 999
-368 | 9644
-366 | 944196611
-364 | 64444
-362 | 77527550000
-360 | 3318864441
-358 | 9774440751
-356 | 8649999922
-354 | 8883119977444222000
-352 | 8888666333331197555220
-350 | 886664444199995553333111
-348 | 99666444442222200008888664422
-346 | 999755533333111777511
-344 | 997222000862
-342 | 8642042
-340 | 880
-338 | 6406
-336 | 19
-334 | 65
-332 | 992
-330 | 5
```

줄기잎그림에서도 여전히 left skewed의 형태를 볼 수 있었고 수치로 확인해 본 결과 변환 이전 보다 skewed 정도가 커졌음을 확인해볼 수 있었다.

```
qqnorm(-1/sqrt(stock_close))
qqline(-1/sqrt(stock_close))
```



qqplot을 통해 확인해본 결과 여전히 정규성과는 거리가 멀어보인다.

<분석 결과 및 결론>

p를 산출하여 변환을 수행해보았고 4가지의 변환을 추가로 수행해보았다. 그러나 skewness는 오히려 개선되지 않았고 정규성을 만족하지 못했다. 따라서, 변환 이전의 상태로 그대로 분석을 수행하는 것이 바람직하다는 결론을 내렸다.

3. DISTRESS.DAT는 severe idiopathic respiratory distress syndrome이 있는 신생아의 태어날 때 몸무게(Kg)이다. *표는 죽은 아이이다.

```
> setwd("D:\\2022-1(3-2)\\2022-01_탐자분\\Data")
> data = read.table("DISTRESS.dat", header=FALSE)
> data
      v1      v2      v3      v4
1  1.050*  2.500*  1.890*  1.760
2  1.175*  1.030*  1.940*  1.930
3  1.230*  1.100*  2.200*  2.015
4  1.310*  1.185*  2.270*  2.090
5  1.500*  1.225*  2.440*  2.600
6  1.600*  1.262*  2.560*  2.700
7  1.720*  1.295*  2.730*  2.950
8  1.750*  1.300*  1.130  3.160
9  1.770*  1.550*  1.575  3.400
10 2.275*  1.820*  1.680  3.640
      v5
1  2.830
2  1.410
3  1.715
4  1.720
5  2.040
6  2.200
7  2.400
8  2.550
9  2.570
10 3.005

> data = as.vector(as.matrix(data))
> data
 [1] "1.050*" "1.175*" "1.230*" "1.310*" "1.500*"
 [6] "1.600*" "1.720*" "1.750*" "1.770*" "2.275*"
[11] "2.500*" "1.030*" "1.100*" "1.185*" "1.225*"
[16] "1.262*" "1.295*" "1.300*" "1.550*" "1.820*"
[21] "1.890*" "1.940*" "2.200*" "2.270*" "2.440*"
[26] "2.560*" "2.730*" "1.130"  "1.575"  "1.680"
[31] "1.760"  "1.930"  "2.015"  "2.090"  "2.600"
[36] "2.700"  "2.950"  "3.160"  "3.400"  "3.640"
[41] "2.830"  "1.410"  "1.715"  "1.720"  "2.040"
[46] "2.200"  "2.400"  "2.550"  "2.570"  "3.005"
```

해당 데이터를 불러온 결과 “,*와 같은 str 이 있었기 때문에 아래의 코드를 실행하여 전처리해 주었다.

```
> library(stringr)
> data2 = str_replace_all(data, "\\*", "")
> data2 = as.numeric(data2)
> data2
 [1] 1.050 1.175 1.230 1.310 1.500 1.600 1.720 1.750
 [9] 1.770 2.275 2.500 1.030 1.100 1.185 1.225 1.262
[17] 1.295 1.300 1.550 1.820 1.890 1.940 2.200 2.270
[25] 2.440 2.560 2.730 1.130 1.575 1.680 1.760 1.930
[33] 2.015 2.090 2.600 2.700 2.950 3.160 3.400 3.640
[41] 2.830 1.410 1.715 1.720 2.040 2.200 2.400 2.550
[49] 2.570 3.005
```

또한, as.numeric() 코드를 실행하여 숫자형태로 변환해주었다.

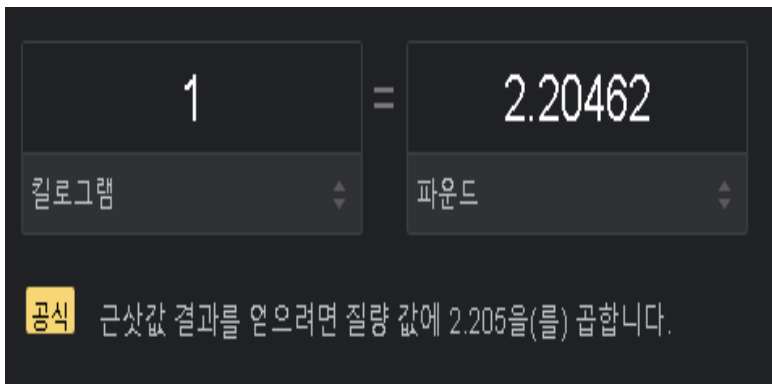
가. 생존 여부에 관계 없이 전체 몸무게의 자료로 상자그림과 줄기그림을 그려라. 몸무게를 파운드로 변환하여 상자그림과 줄기그림을 그리고 선형변환이 분포의 형태를 바꾸지 않는 것을 확인하여라.

전체 몸무게의 줄기그림은 아래와 같다.

```
> stem(data2)

The decimal point is at the |

1 | 0111222233334
1 | 566677778888999
2 | 001223344
2 | 56666778
3 | 0024
3 | 6
```



1kg을 파운드로 변환하기 위해서는 2.20462를 곱해주면 된다.
따라서, data2에 2.20462를 곱한 후 줄기잎그림을 그려보았다.

```
> stem(data2*2.20462)

The decimal point is at the |

 2 | 334566778999
 3 | 134557888999
 4 | 023345699
 5 | 003456677
 6 | 00256
 7 | 05
 8 | 0
```

또한, 변환 전과 후의 boxplot을 그려보았다.

```
par(mfrow=c(1,2))
boxplot(data2)
title("몸무게")
boxplot(data2*2.20462)
title("파운드 변환 후 몸무게")
```

boxplot과 줄기잎그림을 통해 분석해본 결과, 선형 변환은 분포의 변화에 영향을 끼치지 않는다는 것을 확인할 수 있었다.

나. 생존 여부에 관계 없이 전체 몸무게의 자료로 대칭화하여 전과 후를 비교하여라.

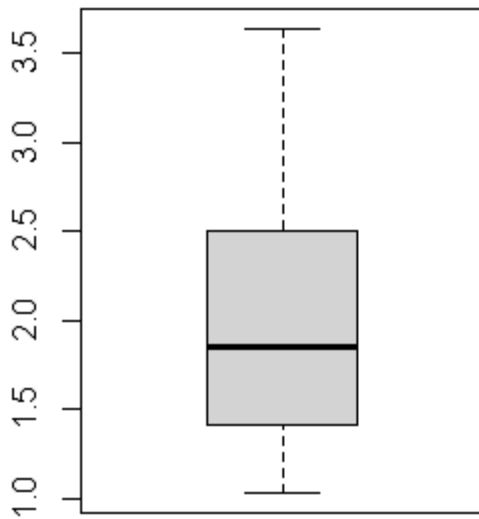
p를 구하기 위해 fivenum()을 수행했다.

$$p = 1 - (2 * M * (Hu - M + Hl - M) / ((Hl - M)^2 + (Hu - M)^2))$$

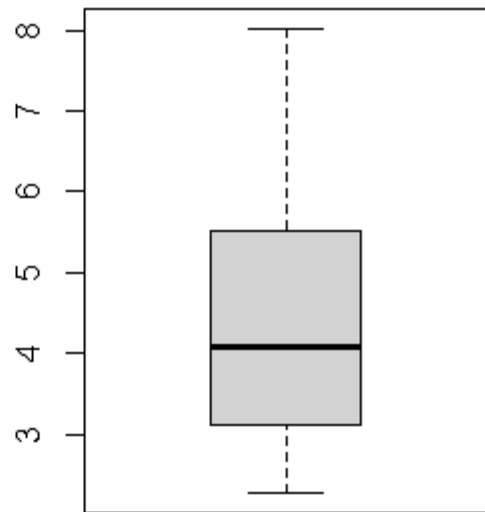
```
> fivenum(data2)
[1] 1.030 1.410 1.855 2.500
[5] 3.640
```

Upper Hinge는 2.5, Lower Hinge는 1.41임을 확인했다.

몸무게



파운드 변환 후 몸무게



```
> re_exp = function(x){
+   Hl <- fivenum(x)[2]
+   M <- fivenum(x)[3]
+   Hu <- fivenum(x)[4]
+   return(1-2*M*(Hu-M+Hl-M)/((Hl-M)^2+(Hu-M)^2))
+ }
> re_exp(data2)
[1] -0.2083707
```

re_exp 결과 -0.2083707라는 수치가 산출되었다. 계산된 p를 그대로 쓰기보다는 이를 표준화된 숫자로 바꿔주는 것이 분석의 용이성 측면에서 더 나은 방향이라고 결론 내렸다. 따라서 $p=0$ 이라고 가정하여 신생아 몸무게에 대한 재표현으로 로그 변환을 채택 후 수행해보았다.

```
> stem(data2)
```

```
The decimal point is at the |
```

```
1 | 0111222233334
1 | 566677778888999
2 | 001223344
2 | 56666778
3 | 0024
3 | 6
```

```
> stem(log(data2))
```

```
The decimal point is 1 digit(s) to the left of the |
```

```
0 | 350267
2 | 0136674
4 | 14572444677
6 | 046601499
8 | 2289244469
10 | 04805
12 | 29
```

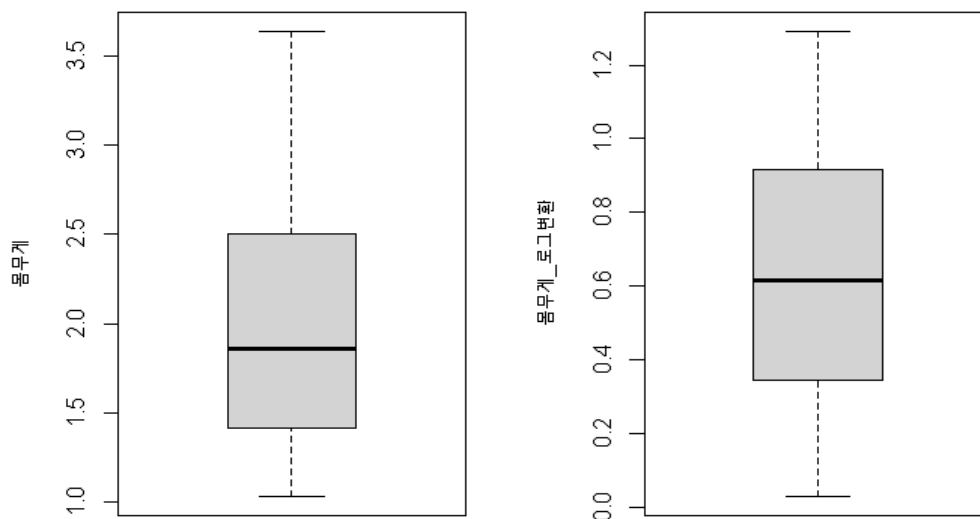
줄기잎그림을 그려본 결과, 대칭성이 생겼음을 확인할 수 있었다. 분포가 종모양을 따르고 있음을 통해 근사적으로 정규분포에 가까운 모양임을 확인했다.

또한, boxplot을 그려서 확인해보았다.

```
par(mfrow=c(1,2))
```

```
boxplot(data2, ylab="몸무게")
```

```
boxplot(log(data2), ylab="몸무게_로그변환")
```



median의 위치가 box의 중간에 왔음을 확인할 수 있었다. 즉, 대칭적인 분포로 변화했음을 알 수 있다.

```
> skewness = function(x){
+   H1 <- fivenum(x)[2]
+   M <- fivenum(x)[3]
+   Hu <- fivenum(x)[4]
+   return((Hu-M-M+H1)/(Hu-M+M-H1))
+ }
> skewness(data2)
[1] 0.1834862
> skewness(log(data2))
[1] 0.0427223
```

skewness를 수치로 확인해보았다. 0.18 정도에서 0.04로 거의 0에 근접함을 확인할 수 있었다.

다. 사망과 생존 집단으로 나누어 상자 그림을 그리고 산포(spread)가 같도록 하는 재표현을 찾아 상자 그림을 그리고 두 집단을 비교하여라. 위의 병이 있는 신생아 중 몸무게가 몇 Kg이면 사망할 가능성이 많다고 예측할 수 있겠는가?

(자료가 5개 칸으로 되어 있는 것은 특별한 의미 없다. 5개 집단 아님)

(사망, 생존 여부를 변수로 만들어 편집 한 후 R로 읽는 것이 편하다.)

```
> data = as.vector(as.matrix(data))
> data
 [1] "1.050*" "1.175*" "1.230*" "1.310*" "1.500*"
 [6] "1.600*" "1.720*" "1.750*" "1.770*" "2.275*"
[11] "2.500*" "1.030*" "1.100*" "1.185*" "1.225*"
[16] "1.262*" "1.295*" "1.300*" "1.550*" "1.820*"
[21] "1.890*" "1.940*" "2.200*" "2.270*" "2.440*"
[26] "2.560*" "2.730*" "1.130"  "1.575"  "1.680"
[31] "1.760"  "1.930"  "2.015"  "2.090"  "2.600"
[36] "2.700"  "2.950"  "3.160"  "3.400"  "3.640"
[41] "2.830"  "1.410"  "1.715"  "1.720"  "2.040"
[46] "2.200"  "2.400"  "2.550"  "2.570"  "3.005"
```

raw data를 다시 불러와보았다.

```
> length(data)
[1] 50
> label = c(rep("사망", 27), rep("생존", 23))
```

data는 총 50개로 이루어져있고 앞의 27개는 * 표시로 사망을 뜻한다. 따라서 상단의 코드를 통해 label을 붙여주었다.

```

> df = data.frame(data2, label)
> colnames(df) = c("몸무게", "생존여부" )
> head(df)
  몸무게 생존여부
1  1.050      사망
2  1.175      사망
3  1.230      사망
4  1.310      사망
5  1.500      사망
6  1.600      사망
>
> re_exp(df[df$생존여부=="생존",]$몸무게)
[1] 0.1296567
> re_exp(df[df$생존여부=="사망",]$몸무게)
[1] -0.07216304
~ |

```

다음으로, 이전에 정의내렸던 re_exp() 함수를 통해 p 값을 찾아보았다. 생존 집단의 경우 0.129 그리고 사망 집단의 경우 -0.07 정도가 나왔음을 확인했다. 두 집단 모두 모두 표준화된 숫자인 0으로 가정하여 로그를 취하는 것이 바람직하다는 결론을 내렸다.

다음으로, 산포를 같게 하는 재표현을 찾아보도록 하자.

산포를 같게 하기 위해서는 $\log(H\text{-spread}) = k + b * \log(M)$ 을 그래프로 그려 기울기 b를 찾아야한다. 회귀식을 통해 기울기를 계산해서 1-b를 계산하고 최적의 p를 찾도록 하겠다.

```

> HsprA <- fivenum(df[df$생존여부=="사망",]$몸무게)[4] - fivenum(df[df$생존여부=="사망",]$몸무게)[2]
> HsprB <- fivenum(df[df$생존여부=="생존",]$몸무게)[4] - fivenum(df[df$생존여부=="생존",]$몸무게)[2]
> M <- c(median(df[df$생존여부=="사망",]$몸무게), median(df[df$생존여부=="생존",]$몸무게))
> Hspr = c(HsprA, HsprB)
>
> (RegrLine <- lm(log(Hspr) ~ log(M)))

```

Call:

```
lm(formula = log(Hspr) ~ log(M))
```

Coefficients:

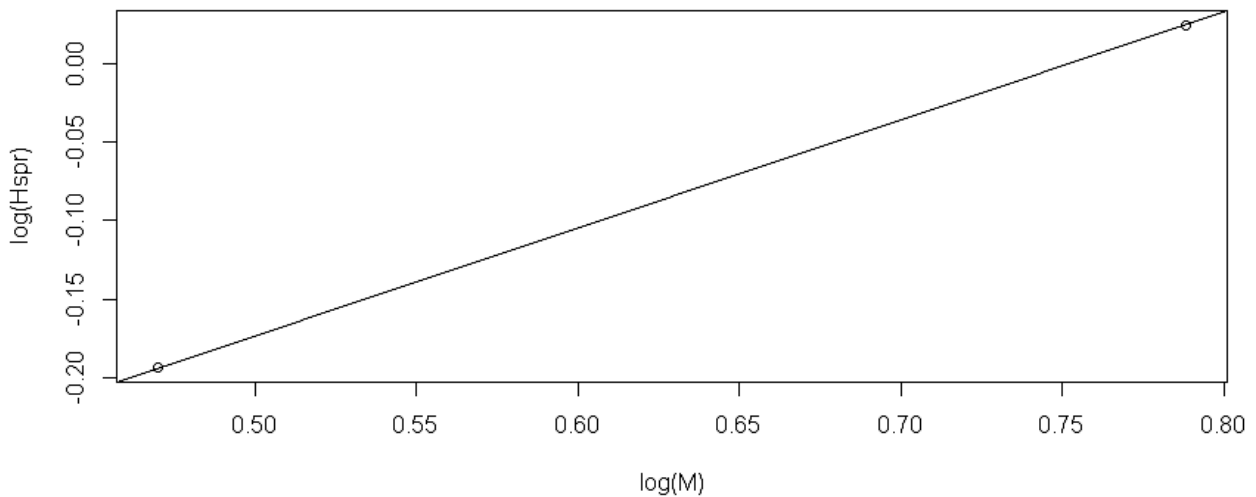
(Intercept)	log(M)
-0.5157	0.6854

```
plot(log(M), log(Hspr))
```

```
abline(coef(RegrLine))
```

회귀식을 통해 확인한 b는 0.6854이므로 $p=1-b=0.3146$ 이다.

0.3146을 그대로 사용하기보다는 반올림한 0.5를 사용하도록 하겠다. 0.5는 Square Root 변환이다. 이를 사용하여 산포(H-spread)의 차이가 줄었는지를 확인해보았다.



```
> HsprA <- fivenum(df[df$생존여부=="사망",]$몸무게)[4] - fivenum(df[df$생존여부=="사망",]$몸무게)[2]
> HsprB <- fivenum(df[df$생존여부=="생존",]$몸무게)[4] - fivenum(df[df$생존여부=="생존",]$몸무게)[2]
>
> HsprA - HsprB
[1] -0.201
> HsprC <- fivenum(df[df$생존여부=="사망",]$몸무게^0.5)[4] - fivenum(df[df$생존여부=="사망",]$몸무게^0.5)[2]
> HsprD <- fivenum(df[df$생존여부=="생존",]$몸무게^0.5)[4] - fivenum(df[df$생존여부=="생존",]$몸무게^0.5)[2]
>
> HsprC - HsprD
[1] -0.02182665
```

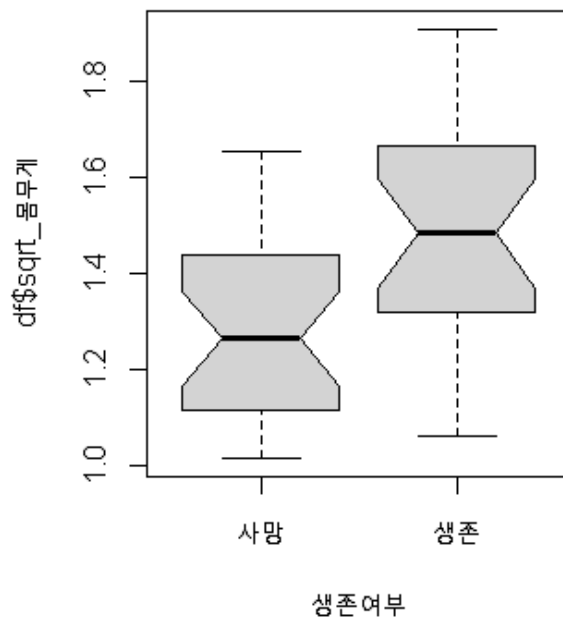
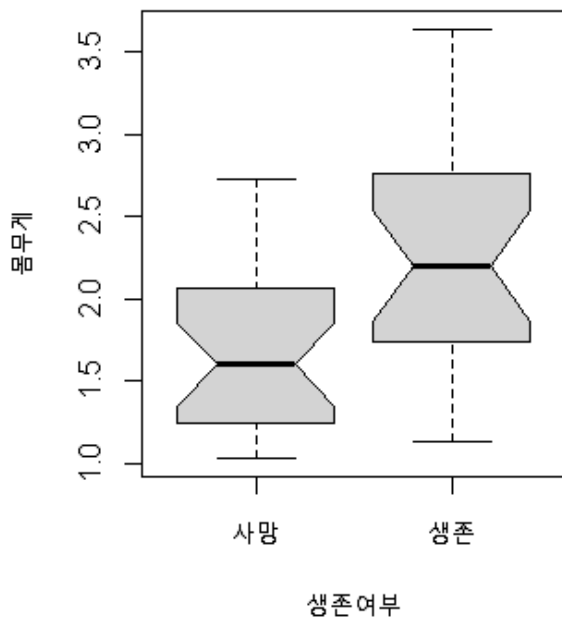
변환 이전엔 -0.201에서 변환 이후엔 -0.02 정도로 산포의 차이가 줄었음을 확인할 수 있었다. 다음으로, boxplot을 그려 확인해보았다.

```
> par(mfrow=c(1,2))
> boxplot(몸무게 ~ 생존여부, data=df, notch=T)
> df$sqrt_몸무게 = sqrt(df$몸무게)
> boxplot(df$sqrt_몸무게 ~ 생존여부, data=df, notch=T)
```

정확한 비교를 위해 notch=T 인자를 포함하여 boxplot을 그렸다. 변환 이전에 비해 변환 이후에 산포가 거의 유사해졌음을 확인할 수 있다. median이 box의 중앙으로 이동했으며 box의 수염의 길이가 비슷해졌음을 확인했다.

notch는 EDA에서 사용하는 robust한 신뢰구간으로 중앙값에 대한 95% 신뢰구간을 계산해준다. 즉, 비교하는 대상의 notch의 범위가 중첩하지 않는다면 약 95%의 유의수준에서 두 데이터는 유의미한 차이가 있는 것이다.

변환 후의 boxplot을 통해 ‘위의 병이 있는 신생아 중 몸무게가 몇 Kg이하면 사망할 가능성이 많다고 예측할 수 있겠는가?’에 대한 답을 찾아보았다.



사망한 신생아의 루트 변환 후 몸무게 median은 1.25정도이다. 생존한 신생아의 루트 변환 후 몸무게 median은 1.5정도이다. notch를 통해 약 95%의 유의 수준에서 사망한 신생아의 몸무게 median은 생존한 신생아의 몸무게 median보다 낮다는 것을 알 수 있다. 몸무게가 낮을수록 사망할 가능성이 높다는 결론을 내릴 수 있다.

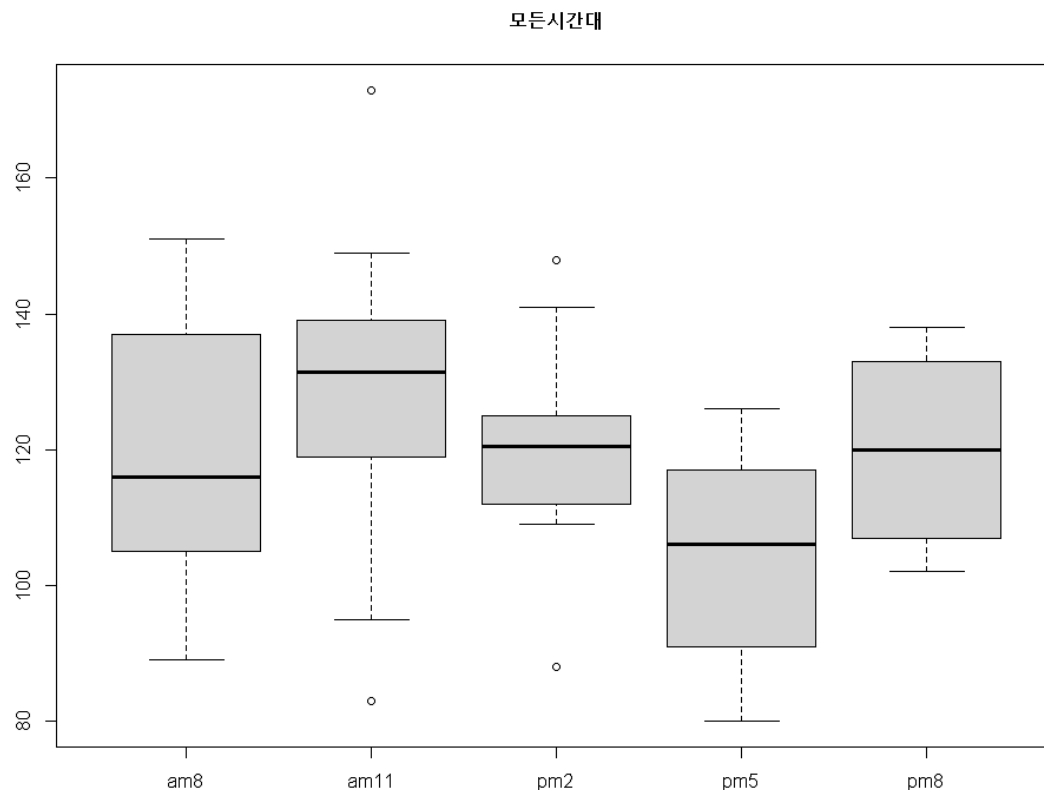
해당 데이터의 경우 각각의 notch가 약 1.35 정도에서 위아래로 나뉘고 있음을 알 수 있다. 따라서 기준을 1.35로 정하는 방향으로 결정했다. 루트 변환 후의 몸무게가 1.35이므로 raw data의 기준으로 환산하기 위해서는 2 를 해주어야 한다. 즉, $1.35^2=1.8225$ 이다. 결론적으로 태어날 때의 몸무게가 1.8225kg 미만이라면 사망할 가능성이 높다.

4. PLASMA.DAT 자료는 오전 8시, 11시, 오후 2시, 5시, 8시 시간대별 10명의 plasma citrate concentration (혈장 농도)를 측정한 결과이다. 상자그림을 그리고 재표현이 필요한지 판단하여 시간대별 어떠한 변화가 있는지 분석하여라.

```
> setwd("D:\\2022-1(3-2)\\2022-01_탐자분\\Data")
>
> data = read.table("PLASMA.DAT", header=FALSE)
> colnames(data) <- c("8am", "11am", "2pm", "5pm", "8pm")
> data
   8am 11am 2pm 5pm 8pm
1   93  121 112 117 121
2  116  135 114  98 135
3  125  137 119 105 102
4  144  173 148 124 122
5  105  119 125  91 133
6  109   83 109  80 104
7   89   95  88  91 116
8  116  128 122 107 119
9  151  149 141 126 138
10 137  139 125 109 107
```

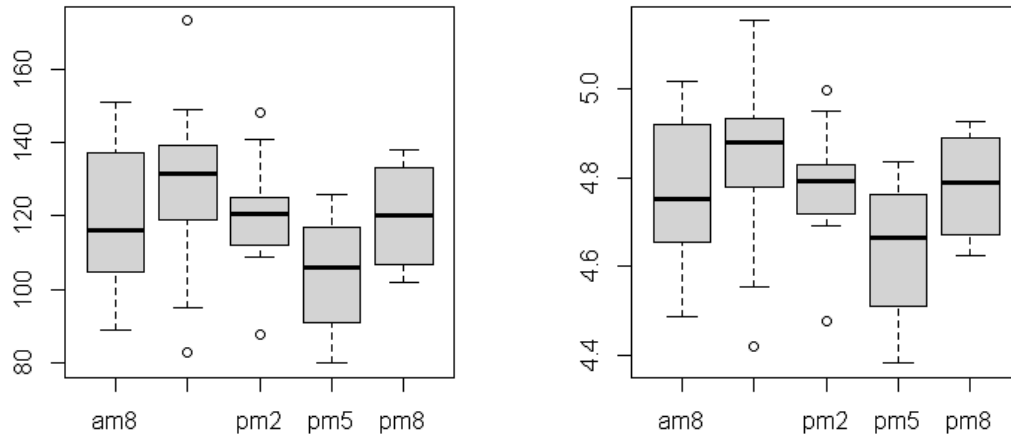
data를 불러온 이후, colnames를 시간대에 맞추어 지정해주었다.

다음으로, boxplot을 그려 시간대별 분포를 확인해보았다.



그 결과, 시간대별로 spread 정도가 다른 것을 알 수 있었다.

가장 먼저, 로그 변환을 수행해서 boxplot을 그려보고 결과를 분석해보았다.



로그 변환을 수행한 후에도 집단 별로 spread 정도가 많이 다름을 확인할 수 있었다.

그룹 간의 spread가 같으면 그룹 간의 비교를 할 때 spread에 상관없이 location만 확인해보면 된다. 따라서 시간대별 인구 분포를 symmetric하게 만듦으로써 분포들의 spread를 비슷하게 맞춰주는 방향으로 변환을 시도해 보았다. (자료를 symmetric하게 만들면 여러 group들의 spread도 같이 맞춰지게 되기 때문이다.)

variance-stabilizing transformation

Model의 spread는 $Median^b$ 에 비례하는 값으로 많이 나온다. 따라서 Spread가 같게 되는 re-expression을 찾을 때, median값에 proportional한 model을 사용한다.

$$\begin{aligned}
 Hspr &= HU - HL = cMb \\
 \log Hspr &= \log c + b * \log M \\
 &= k + b * \log M
 \end{aligned}$$

Hspr와 median에 로그를 취함으로써 선형관계로 만들어주었다.

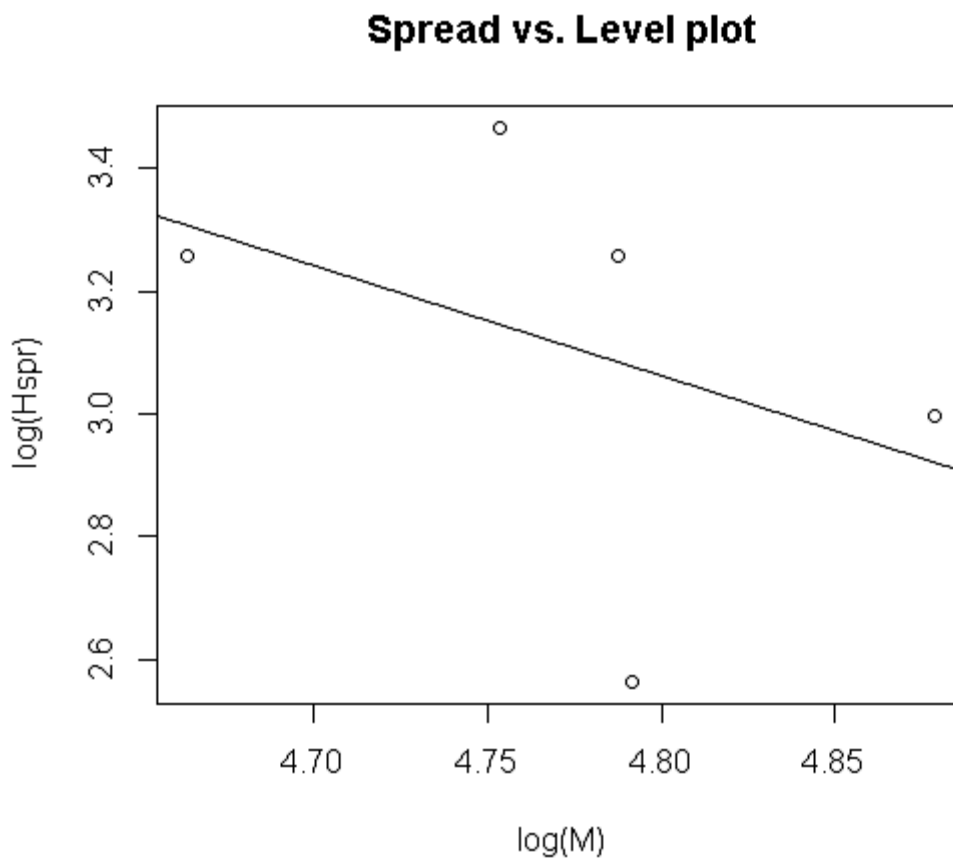
log(Hspr)와 log(M)의 관계는 보기 위해 아래의 코드를 실행했다.

```
Hspr_am8 <- fivenum(data$am8)[4] - fivenum(data$am8)[2]
Hspr_am11 <- fivenum(data$am11)[4] - fivenum(data$am11)[2]
Hspr_pm2 <- fivenum(data$pm2)[4] - fivenum(data$pm2)[2]
Hspr_pm5 <- fivenum(data$pm5)[4] - fivenum(data$pm5)[2]
Hspr_pm8 <- fivenum(data$pm8)[4] - fivenum(data$pm8)[2]

Hspr <- c(Hspr_am8, Hspr_am11, Hspr_pm2, Hspr_pm5, Hspr_pm8)
Hspr
11 32 20 13 26 26

> M <- c(median(data$am8),
+        median(data$am11),
+        median(data$pm2),
+        median(data$pm5),
+        median(data$pm8))
> M
[1] 116.0 131.5 120.5 106.0 120.0
```

```
plot(log(M), log(Hspr), main="Spread vs. Level plot")
(RegrLine <- lm(log(Hspr) ~ log(M)))
abline(coef(RegrLine))
```



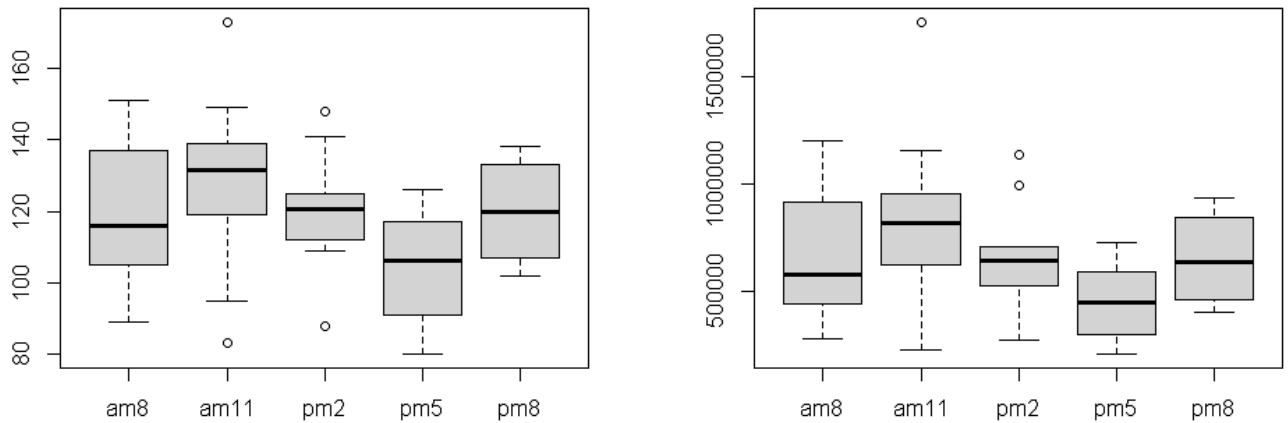
```
> (RegrLine <- lm(log(Hspr) ~ log(M)))
Call:
lm(formula = log(Hspr) ~ log(M))

Coefficients:
(Intercept)      log(M)
      11.66         -1.79

>
> abline(coef(RegrLine))
> 1 - coef(RegrLine)[2]
      log(M)
 2.790418
```

계산한 coefficient는 $p = -1.79$ 로, $b = 1 - p = 2.79$ 로 나왔다. 이를 활용해서 데이터를 변환해 보았다.

```
par(mfrow=c(1,2))
boxplot(data, data = data)
boxplot(data^2.79, data = data)
```



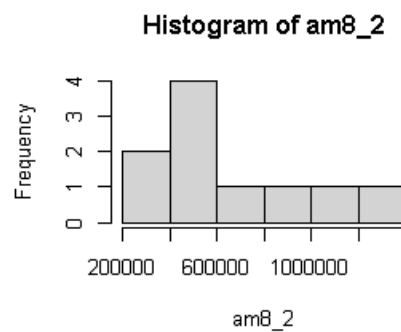
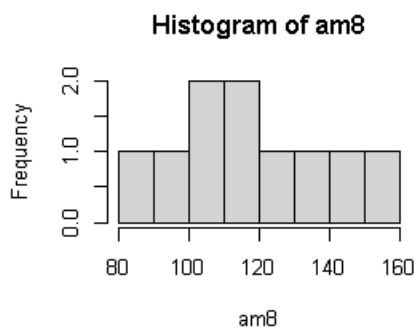
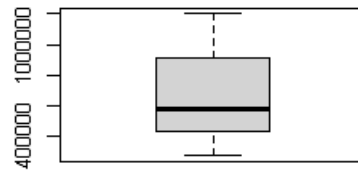
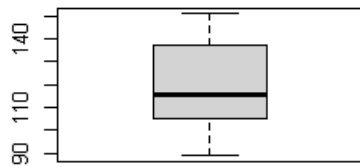
상단의 코드를 활용해 변환 이전과 이후를 비교해보았다. 한 눈에 봐도 시간대별 box의 길이가 변환 이전에 비해 비슷해졌음을 확인해볼 수 있었다.

또한, outlier를 보았을 때 변환 이전에는 am11에서 2개, pm2에서 2개 볼 수 있었다. 변환 이후에는 am11에서 1개, pm2에서는 2개 볼 수 있었다.

해당 문제에서는 산포의 특성을 이용하여 변환을 수행해보았다. 해당 방법은 spread와 median의 1-p제곱의 proportional한 성질(데이터의 값이 커지면 spread도 커지는 경향이 있다는 경향)을 이용한 방법이었다. 결과적으로, 변환 이후 각 집단별 spread의 차이 정도가 줄어들었음을 확인할 수 있었다.

마지막으로, 변환 전과 이후의 각 시간대별 boxplot과 histogram 그리고 skewness 수치를 통해 결과를 확인해보았다.

```
> am8 = data$am8
> am11 = data$am11
> pm2 = data$pm2
> pm5 = data$pm5
> pm8 = data$pm8
>
> data_2 = data^2.79
> am8_2 = data_2$am8
> am11_2 = data_2$am11
> pm2_2 = data_2$pm2
> pm5_2 = data_2$pm5
> pm8_2 = data_2$pm8
>
> par(mfrow=c(2,2))
> boxplot(am8)
> boxplot(am8_2)
> hist(am8)
> hist(am8_2)
>
> skewness = function(x){
+   H1 <- fivenum(x)[2]
+   M <- fivenum(x)[3]
+   Hu <- fivenum(x)[4]
+   return((Hu-M-M+H1)/(Hu-M+M-H1))
+ }
>
> skewness(am8)
[1] 0.3125
> skewness(am8_2)
[1] 0.4176656
```

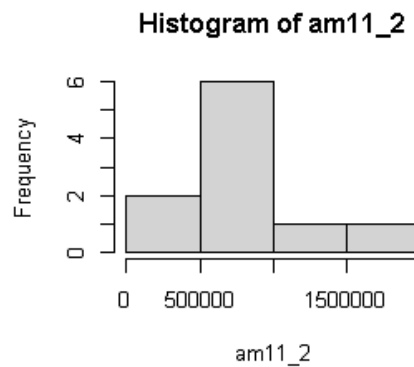
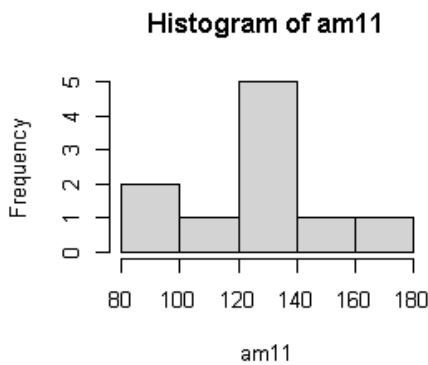
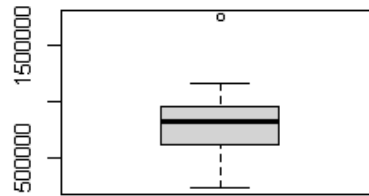
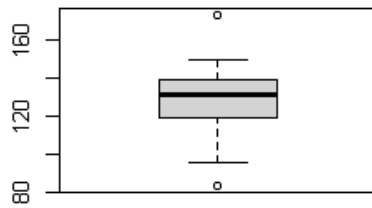


am8의 경우 변환 이후 skewness가 조금 커졌음을 확인할 수 있었다.

다음으로, am11의 경우를 살펴보았다.

```
> par(mfrow=c(2,2))
> boxplot(am11)
> boxplot(am11_2)
> hist(am11)
> hist(am11_2)
> skewness(am11)
[1] -0.25
> skewness(am11_2)
[1] -0.1933392
.
```

skewness의 정도가 줄어들었음을 확인할 수 있었다.

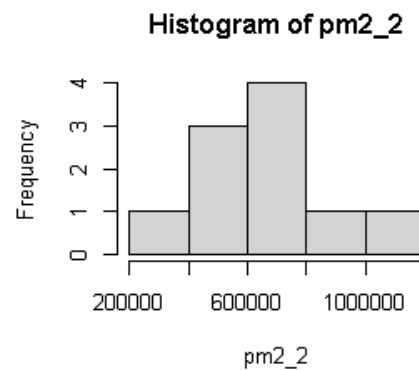
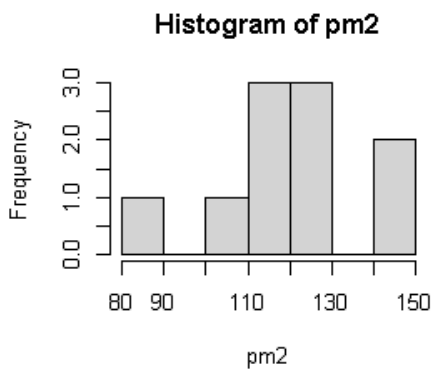
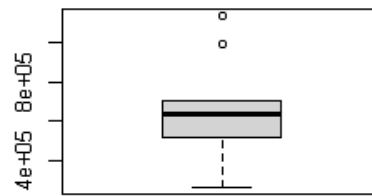
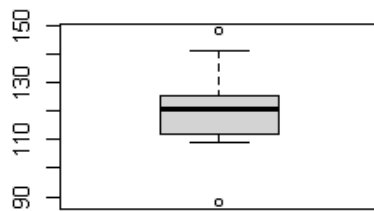


histogram과 boxplot을 통해서도 대칭성이 변환 이전에 비해 나타났음을 확인할 수 있었다.

다음으로, pm2의 경우를 살펴보았다.

```
> par(mfrow=c(2,2))
> boxplot(pm2)
> boxplot(pm2_2)
> hist(pm2)
> hist(pm2_2)
> skewness(pm2)
[1] -0.3076923
> skewness(pm2_2)
[1] -0.2657308
```

pm2 시간대도 마찬가지로 skewness의 정도가 줄어들었다.

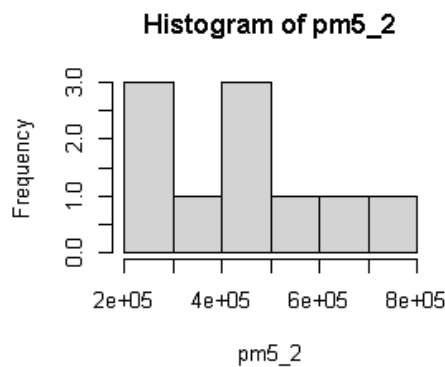
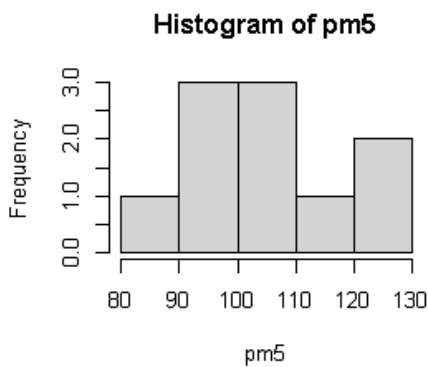
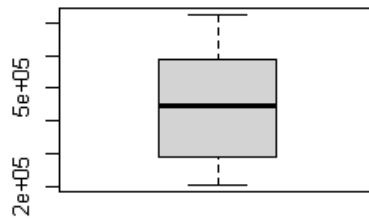
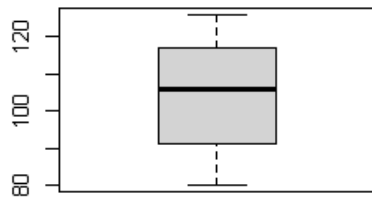


histogram과 boxplot을 통해서도 대칭성이 변환 이전에 비해 두드러지게 나타났음을 확인할 수 있었다.

다음으로, pm5 경우를 살펴보자.

```
> par(mfrow=c(2,2))
> boxplot(pm5)
> boxplot(pm5_2)
> hist(pm5)
> hist(pm5_2)
> skewness(pm5)
[1] -0.1538462
> skewness(pm5_2)
[1] -0.04516235
> |
```

pm2 시간대도 마찬가지로 skewness의 정도가 약 -0.15에서 약 -0.04로 줄어들었다.

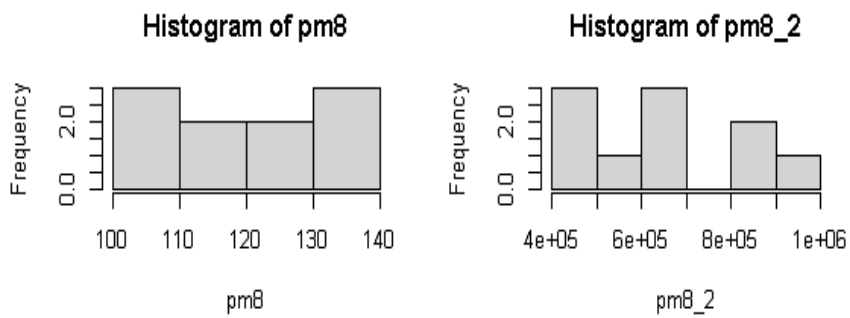
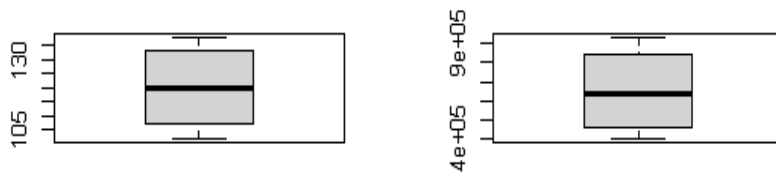


boxplot 결과를 통해 median 값이 box의 중간으로 이동했음을 확인할 수 있었다.

마지막으로, pm8의 경우 또한 확인해보았다.

```
> par(mfrow=c(2,2))
> boxplot(pm8)
> boxplot(pm8_2)
> hist(pm8)
> hist(pm8_2)
> skewness(pm8)
[1] 0
> skewness(pm8_2)
[1] 0.09610301
```

pm8의 경우 skewness가 0에서 0.09 정도로 살짝 증가했음을 확인할 수 있었다.



pm8의 경우 변환 이전의 skewness가 0이었기 때문에 변환 이전이 더 낫다는 결론을 내릴 수 있었다. pm8 집단의 경우는 그대로 둔 채로 am8, am11, pm2, pm5 4개의 시간대별로 회귀계수를 구해서 동일한 분석을 진행해보는 방향으로 수행해보는 것도 더 나은 결과를 볼 수 있을 것이라는 생각을 하게 되었다.