Quetelet's Body Mass Index with rn96 data

coop711

2019-03-03

Data

`rn96.txt` 자료 읽어들이기

- `"../data/rn96.txt"` 는 파일 경로이므로 `rn96.txt` 를 다운로드받은 폴더로 지정하면 되나 가급적 `R` source 를 저장하는 폴더(예를 들어서, 폴더 `R`)와 원시 자료를 저장하는 폴더(예를 들어서, 폴더 `data`)를 구분해 두는 것이 효과적이다. 원 자료를 보고 변수명에 해당하는 `header` 매개변수를 `TRUE` 로 설정한다. `<-` 는 assignment. 아래 주어진 세 가지 표현은 모두 동일한 결과를 가져온다.

```
rn96 <- read.table("../data/rn96.txt", header = TRUE)
head(rn96, n = 10)
```

```
##    height weight
## 1     161     50
## 2     155     49
## 3     158     42
## 4     170     65
## 5     160     60
## 6     156     52
## 7     162     58
## 8     158     46
## 9     158     45
## 10    167     51
```

```
(rn96_2 <- read.table("../data/rn96.txt", header = TRUE))
```

```
##    height weight
## 1     161     50
## 2     155     49
## 3     158     42
## 4     170     65
## 5     160     60
## 6     156     52
## 7     162     58
## 8     158     46
## 9     158     45
## 10    167     51
## 11    160     50
## 12    155     42
## 13    154     53
## 14    155     52
## 15    157     48
## 16    157     48
## 17    160     49
## 18    158     52
## 19    160     51
## 20    160     53
## 21    152     44
## 22    154     56
## 23    150     63
## 24    161     52
## 25    162     57
## 26    164     49
## 27    161     52
## 28    155     54
## 29    159     46
## 30    163     50
## 31    159     61
## 32    160     55
## 33    158     45
## 34    165     63
## 35    156     60
## 36    163     56
## 37    155     52
## 38    164     47
## 39    163     52
## 40    168     55
## 41    157     48
```

```
assign("rn96_3", read.table("../data/rn96.txt", header = TRUE))
tail(rn96_3, n = 10)
```

```
##    height weight
## 32    160     55
## 33    158     45
## 34    165     63
## 35    156     60
## 36    163     56
## 37    155     52
## 38    164     47
## 39    163     52
## 40    168     55
## 41    157     48
```

- `str()` 은 자료의 구조를 살피는 함수이며, value는 없다. `rn96` 의 자료구조를 살펴보면, `int` (integer) class가 나온다. 정수이지만 사칙 연산에는 아무런 영향이 없다.
- `sapply()` 는 `rn96` 의 각 변수에 `mode()` 또는 `class()` 를 적용한다.

```
str(rn96)
```

```
## 'data.frame':    41 obs. of  2 variables:
##  $ height: int  161 155 158 170 160 156 162 158 158 167 ...
##  $ weight: int  50 49 42 65 60 52 58 46 45 51 ...
```

```
sapply(rn96, mode)
```

```
##    height    weight
## "numeric" "numeric"
```

```
sapply(rn96, class)
```

```
##    height    weight
## "integer" "integer"
```

Summary Statistics

- `height` 와 `weight` 의 기초통계 살피기

```
summary(rn96)
```

```
##      height          weight
##  Min.   :150.0   Min.   :42.00
##  1st Qu.:156.0   1st Qu.:48.00
##  Median :159.0   Median :52.00
##  Mean   :159.3   Mean   :52.02
##  3rd Qu.:162.0   3rd Qu.:55.00
##  Max.   :170.0   Max.   :65.00
```

- 평균과 표준편차를 각 변수별로 살펴보려면 `apply()` 를 이용한다. 이 때 `options(digits = 2)` 가 없으면 출력결과가 어떻게 달라지는지 살펴보자. 출력 결과에 이름을 붙이는 방법을 보여주고 있다. `height` 와 `weight` 를 불러들이는 방법에 유의하자. 이는 `str(rn96)` 의 결과로부터 생각할 수 있는 것으로서 `rn96` 을 `list` 로도 볼 수 있기 때문이다.

```
options("digits")
```

```
## $digits
## [1] 7
```

```
options(digits = 2)
apply(rn96, 2, mean)
```

```
## height weight
##    159     52
```

```
apply(rn96, 2, sd)
```

```
## height weight
##    4.3    5.7
```

```
c(mean(rn96$height), sd(rn96$height))
```

```
## [1] 159.3   4.3
```

```
c(mean(rn96$weight), sd(rn96$weight))
```

```
## [1] 52.0  5.7
```

```
c(Mean = mean(rn96$height), SD = sd(rn96$height))
```

```
##  Mean    SD
## 159.3   4.3
```

```
c(Mean = mean(rn96$weight), SD = sd(rn96$weight))
```

```
## Mean   SD
## 52.0  5.7
```

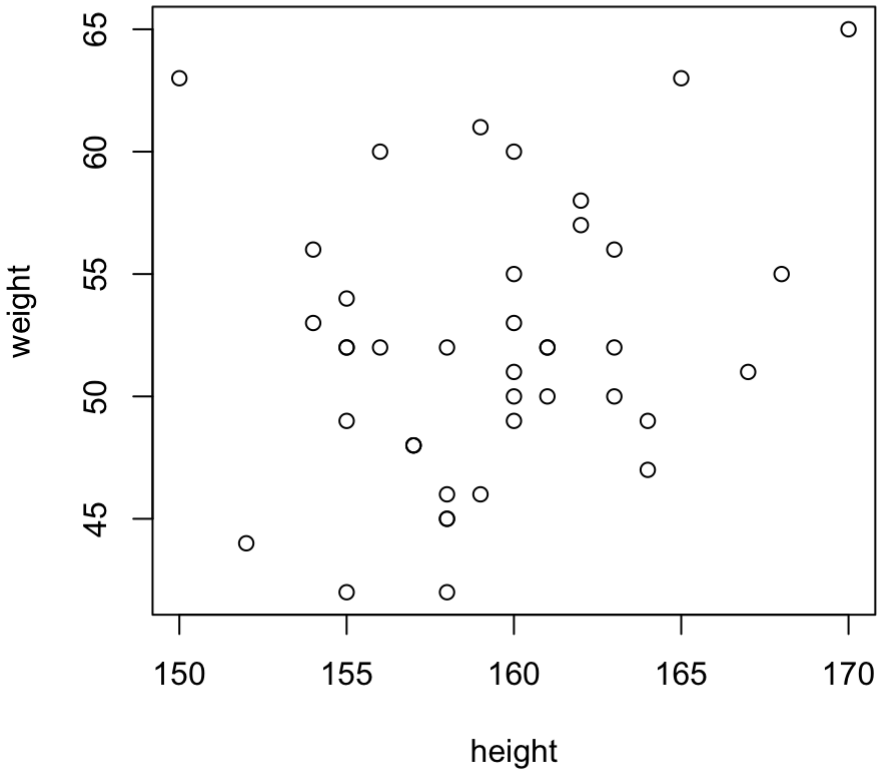
```
options(digits = 7)
```

Base Graphics

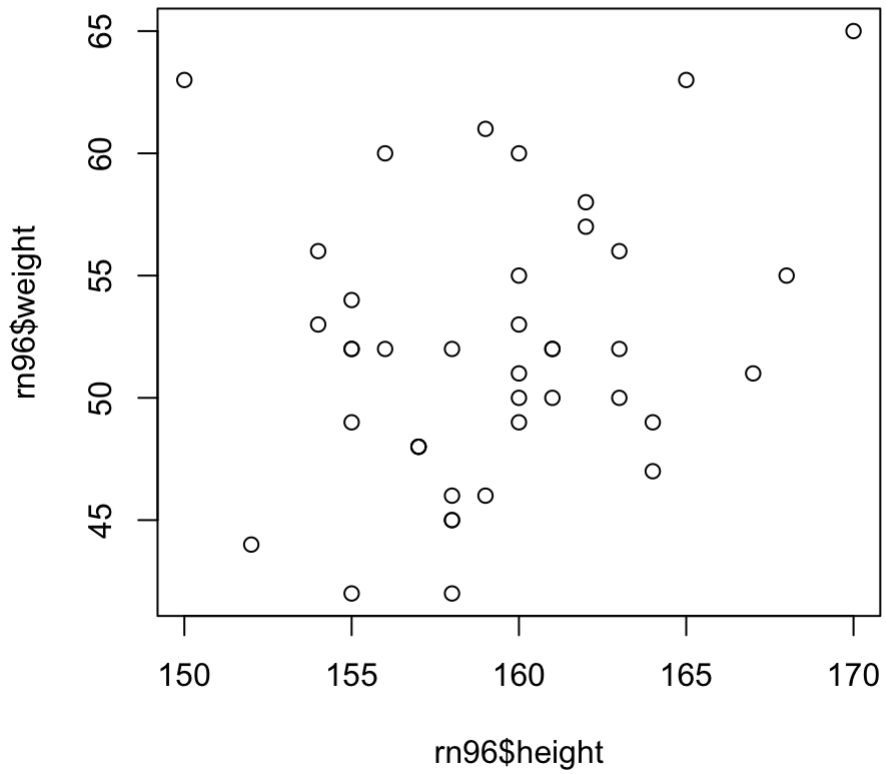
`height` 와 `weight` 의 산점도 그리기.

- `x` 축에 독립변수로 들어가는 변수와 `y` 축에 들어가는 종속변수를 설정하는 여러가지 방법이 있음을 알 수 있다. 동일한 `plot()` 이지만 출력 결과에서 default로 나오는 각 축의 label 값이 서로 다름을 알 수 있다. 맨 마지막 방법에 유의하라.
- 출력결과에서 도표의 크기는 `par()` 에서 조정할 수도 있고, R Studio의 R markdown 에서 조정할 수도 있다. Graphic 수업에서 보다 자세히 다룰 예정이다.

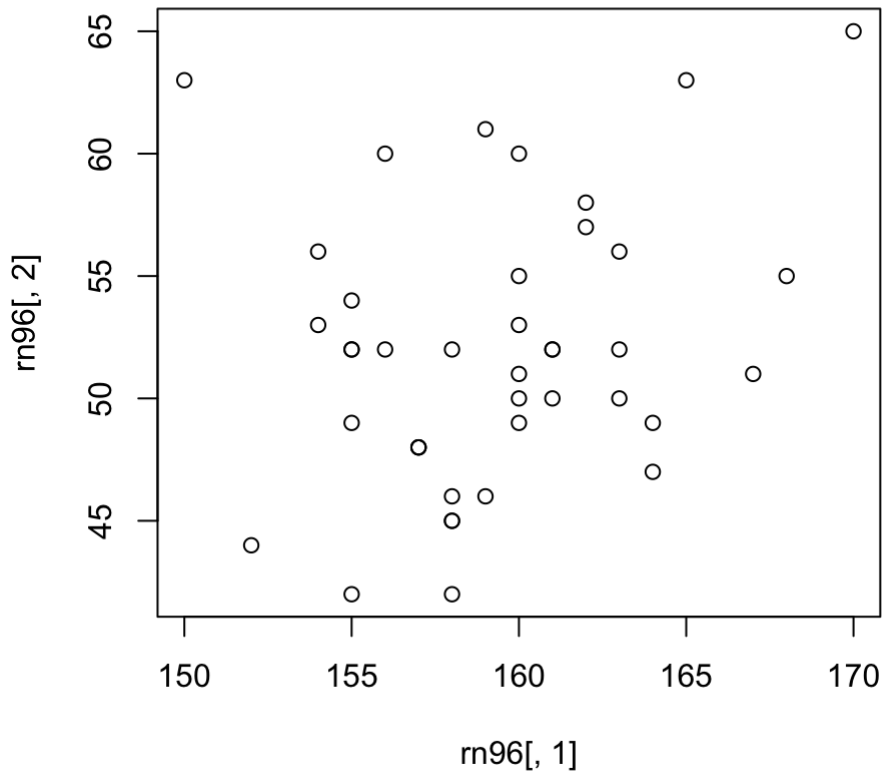
```
plot(weight ~ height, data = rn96)
```



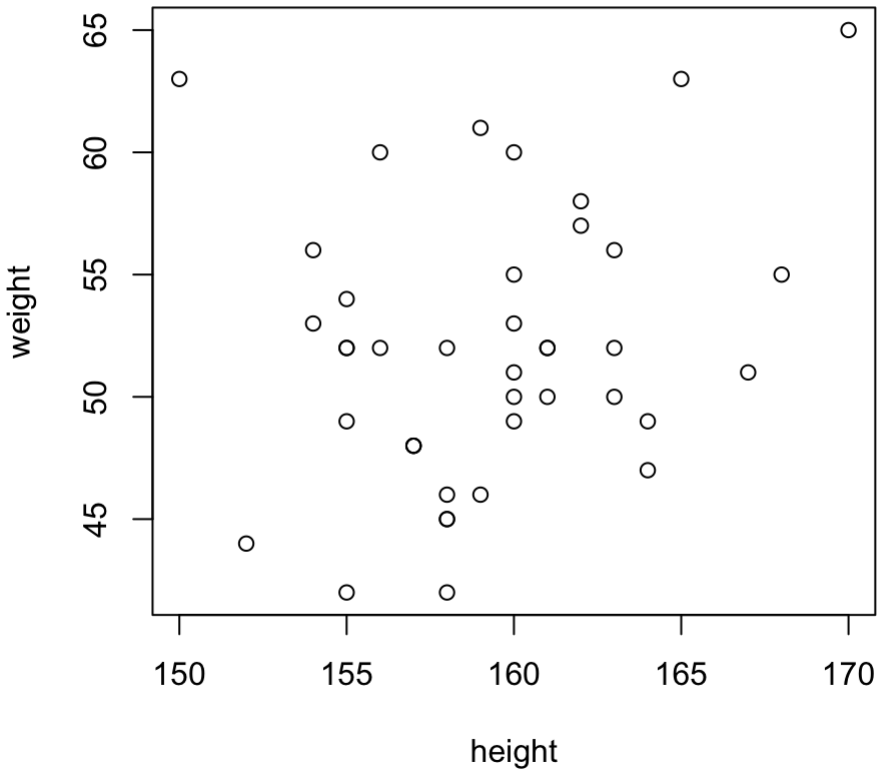

```
plot(rn96$height, rn96$weight)
```




```
plot(rn96[, 1], rn96[, 2])
```



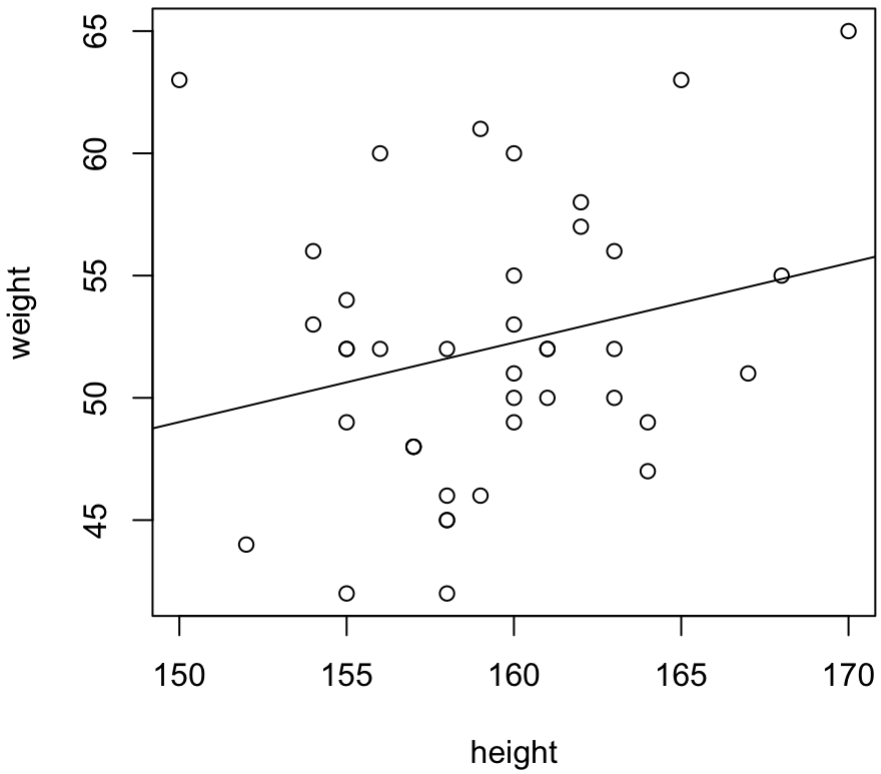
```
plot(rn96)
```



선형회귀선 추가하기.

- 선형모형으로 적합하는 `lm()` 의 결과물을 활용하고 있다.

```
plot(weight~height, data = rn96)
abline(lm(weight ~ height, data = rn96)$coefficient)
```



- 선형모형으로 분석하기 위하여 별도의 `R` 오브젝트로 저장한다.

```
rn96_lm <- lm(weight ~ height, data = rn96)
```

회귀계수와 관련 통계량 살피기.

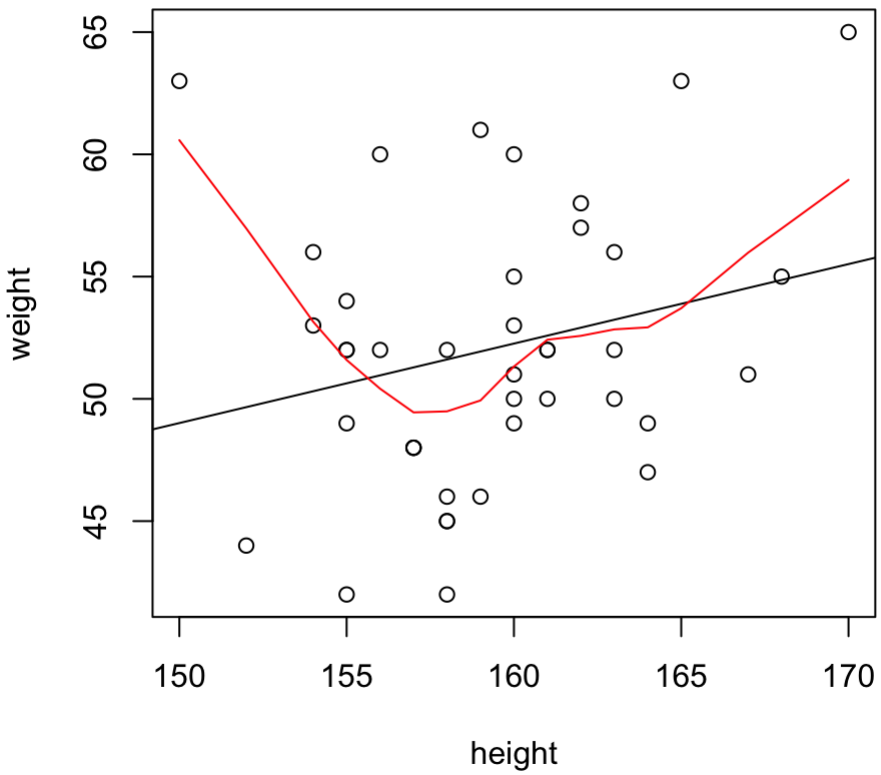
- `summary()` 를 이용하여 선형모형 분석에 등장하는 각종 통계를 살펴볼 수 있다.

```
summary(rn96_lm)
```

```
##
## Call:
## lm(formula = weight ~ height, data = rn96)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.6120 -3.2868 -0.5875  2.7622 13.9893
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.2369    32.8626   0.007    0.994
## height        0.3252     0.2063   1.576    0.123
##
## Residual standard error: 5.565 on 39 degrees of freedom
## Multiple R-squared:  0.0599, Adjusted R-squared:  0.0358
## F-statistic: 2.485 on 1 and 39 DF,  p-value: 0.123
```

- 1차 회귀식으로는 살피기 힘든 국소적인 변화를 살피기 위하여 `lowess()` 를 이용한 local smoother를 추가한다.

```
plot(weight ~ height, data = rn96)
abline(lm(weight ~ height, data = rn96)$coefficient)
lines(lowess(rn96$height, rn96$weight), col = "red")
```



BMI 계산하고 줄기-잎 그리기

- 조금 편하게 작업하기 위해서 `height` 와 `weight` 를 별도의 R object로 복사하여 사용하자. 우선, `height` 와 `weight` 를 불러내는 방법이 다음과 같이 다양하다는 점을 기억해 두자.

```
rn96$height
```

```
##  [1] 161 155 158 170 160 156 162 158 158 167 160 155 154 155 157 157 160
## [18] 158 160 160 152 154 150 161 162 164 161 155 159 163 159 160 158 165
## [35] 156 163 155 164 163 168 157
```

```
str(rn96$height)
```

```
##  int [1:41] 161 155 158 170 160 156 162 158 158 167 ...
```

```
rn96[, 1]
```

```
##  [1] 161 155 158 170 160 156 162 158 158 167 160 155 154 155 157 157 160
## [18] 158 160 160 152 154 150 161 162 164 161 155 159 163 159 160 158 165
## [35] 156 163 155 164 163 168 157
```

```
str(rn96[, 1])
```

```
##  int [1:41] 161 155 158 170 160 156 162 158 158 167 ...
```

```
rn96[, "height"]
```

```
##  [1] 161 155 158 170 160 156 162 158 158 167 160 155 154 155 157 157 160
## [18] 158 160 160 152 154 150 161 162 164 161 155 159 163 159 160 158 165
## [35] 156 163 155 164 163 168 157
```

```
str(rn96[, "height"])
```

```
##  int [1:41] 161 155 158 170 160 156 162 158 158 167 ...
```

```
rn96["height"]
```

```
##    height
## 1     161
## 2     155
## 3     158
## 4     170
## 5     160
## 6     156
## 7     162
## 8     158
## 9     158
## 10    167
## 11    160
## 12    155
## 13    154
## 14    155
## 15    157
## 16    157
## 17    160
## 18    158
## 19    160
## 20    160
## 21    152
## 22    154
## 23    150
## 24    161
## 25    162
## 26    164
## 27    161
## 28    155
## 29    159
## 30    163
## 31    159
## 32    160
## 33    158
## 34    165
## 35    156
## 36    163
## 37    155
## 38    164
## 39    163
## 40    168
## 41    157
```

```
str(rn96["height"])
```

```
## 'data.frame':    41 obs. of  1 variable:
##  $ height: int  161 155 158 170 160 156 162 158 158 167 ...
```

```
rn96[1]
```

```
##    height
## 1     161
## 2     155
## 3     158
## 4     170
## 5     160
## 6     156
## 7     162
## 8     158
## 9     158
## 10    167
## 11    160
## 12    155
## 13    154
## 14    155
## 15    157
## 16    157
## 17    160
## 18    158
## 19    160
## 20    160
## 21    152
## 22    154
## 23    150
## 24    161
## 25    162
## 26    164
## 27    161
## 28    155
## 29    159
## 30    163
## 31    159
## 32    160
## 33    158
## 34    165
## 35    156
## 36    163
## 37    155
## 38    164
## 39    163
## 40    168
## 41    157
```

```
str(rn96[1])
```

```
## 'data.frame':    41 obs. of  1 variable:
##  $ height: int  161 155 158 170 160 156 162 158 158 167 ...
```

```
height <- rn96$height
weight <- rn96$weight
```

BMI 계산

- 체질량지수라고 알려져 있는 BMI(Body Mass Index) 공식은 $\frac{\text{몸무게}(kg)}{\text{키}^2(m)}$로 주어진다. 이는 벨기에의 수학자, 천문학자 이자 사회통계학자로 알려져 있는 아돌프 케틀레의 업적 중 하나이다. 아래 계산에서 `round()` 를 씌우지 않으면 어떤 출력이 나오는지 살펴 보고, `digits =` 를 바꿔 가며 결과를 비교해 보자. 위에서 `options(digits = 2)` 라고 설정했을 때 와의 차이를 생각해 보자.
- `rn96` 에 BMI 계산 결과를 합쳐 보기 위해서 `cbind()` (column끼리 묶는다)를 사용하였다. 키, 몸무게, BMI가 모두 숫자변수이기 때문에 가능하다.
- `head()` , `tail()` 은 괄호 안에 들어가는 자료의 첫 6개와 끝 6개를 보여준다. 갯수를 조정하려면 `n =` 매개변수를 사용한다.

```
(BMI <- weight / (height / 100) ^ 2)
```

```
##  [1] 19.28938 20.39542 16.82423 22.49135 23.43750 21.36752 22.10029
##  [8] 18.42653 18.02596 18.28678 19.53125 17.48179 22.34778 21.64412
## [15] 19.47341 19.47341 19.14062 20.83000 19.92187 20.70312 19.04432
## [22] 23.61275 28.00000 20.06095 21.71925 18.21832 20.06095 22.47659
## [29] 18.19548 18.81892 24.12879 21.48437 18.02596 23.14050 24.65483
## [36] 21.07720 21.64412 17.47472 19.57168 19.48696 19.47341
```

```
(BMI <- round(weight / (height / 100) ^ 2, digits = 1))
```

```
##  [1] 19.3 20.4 16.8 22.5 23.4 21.4 22.1 18.4 18.0 18.3 19.5 17.5 22.3 21.6
## [15] 19.5 19.5 19.1 20.8 19.9 20.7 19.0 23.6 28.0 20.1 21.7 18.2 20.1 22.5
## [29] 18.2 18.8 24.1 21.5 18.0 23.1 24.7 21.1 21.6 17.5 19.6 19.5 19.5
```

```
head(cbind(rn96, BMI))
```

```
##   height weight  BMI
## 1    161     50 19.3
## 2    155     49 20.4
## 3    158     42 16.8
## 4    170     65 22.5
## 5    160     60 23.4
## 6    156     52 21.4
```

```
tail(cbind(rn96, BMI), n = 10)
```

```
##    height weight  BMI
## 32    160     55 21.5
## 33    158     45 18.0
## 34    165     63 23.1
## 35    156     60 24.7
## 36    163     56 21.1
## 37    155     52 21.6
## 38    164     47 17.5
## 39    163     52 19.6
## 40    168     55 19.5
## 41    157     48 19.5
```

BMI 값들의 줄기-잎 그림 그리기

- John W. Tukey의 수많은 업적 중의 하나인 줄기-잎 그림은 자료의 윤곽 뿐 아니라 개별 값도 함께 파악할 수 있는 유용한 도구이다. R에서는 `stem()` 이라는 함수로 계산한다. 많이 쓰이는 매개변수로는 `scale =` 이 있다.

```
stem(BMI)
```

```
##
##   The decimal point is at the |
##
##   16 | 855
##   18 | 00223480135555569
##   20 | 11478145667
##   22 | 1355146
##   24 | 17
##   26 |
##   28 | 0
```

```
stem(BMI, scale = 2)
```

```
##
##   The decimal point is at the |
##
##   16 | 8
##   17 | 55
##   18 | 0022348
##   19 | 0135555569
##   20 | 11478
##   21 | 145667
##   22 | 1355
##   23 | 146
##   24 | 17
##   25 |
##   26 |
##   27 |
##   28 | 0
```

- `weight` 와 `height` 의 줄기-잎 그림

```
stem(height)
```

```
##
##   The decimal point is at the |
##
##   150 | 0
##   152 | 0
##   154 | 0000000
##   156 | 00000
##   158 | 0000000
##   160 | 000000000
##   162 | 00000
##   164 | 000
##   166 | 0
##   168 | 0
##   170 | 0
```

```
stem(weight)
```

```
##
##   The decimal point is at the |
##
##   42 | 00
##   44 | 000
##   46 | 000
##   48 | 000000
##   50 | 00000
##   52 | 000000000
##   54 | 000
##   56 | 000
##   58 | 0
##   60 | 000
##   62 | 00
##   64 | 0
```

BMI를 토대로 한 비만도 판정

- 18.5 미만은 underweight, 18.5 ~ 24.9 는 Normal, 25 ~ 29.9 는 Overweight, 30 이상은 Obese 로 판정

```
rn96$BMI <- BMI
rn96$obesity <- ifelse(BMI < 18.5, "Underweight",
                       ifelse(BMI >= 18.5 & BMI < 24.9, "Normal",
                              ifelse(BMI >= 25 & BMI < 29.9, "Overweight", "Obese")))
rn96
```

```
##    height weight  BMI     obesity
## 1     161     50 19.3      Normal
## 2     155     49 20.4      Normal
## 3     158     42 16.8 Underweight
## 4     170     65 22.5      Normal
## 5     160     60 23.4      Normal
## 6     156     52 21.4      Normal
## 7     162     58 22.1      Normal
## 8     158     46 18.4 Underweight
## 9     158     45 18.0 Underweight
## 10    167     51 18.3 Underweight
## 11    160     50 19.5      Normal
## 12    155     42 17.5 Underweight
## 13    154     53 22.3      Normal
## 14    155     52 21.6      Normal
## 15    157     48 19.5      Normal
## 16    157     48 19.5      Normal
## 17    160     49 19.1      Normal
## 18    158     52 20.8      Normal
## 19    160     51 19.9      Normal
## 20    160     53 20.7      Normal
## 21    152     44 19.0      Normal
## 22    154     56 23.6      Normal
## 23    150     63 28.0  Overweight
## 24    161     52 20.1      Normal
## 25    162     57 21.7      Normal
## 26    164     49 18.2 Underweight
## 27    161     52 20.1      Normal
## 28    155     54 22.5      Normal
## 29    159     46 18.2 Underweight
## 30    163     50 18.8      Normal
## 31    159     61 24.1      Normal
## 32    160     55 21.5      Normal
## 33    158     45 18.0 Underweight
## 34    165     63 23.1      Normal
## 35    156     60 24.7      Normal
## 36    163     56 21.1      Normal
## 37    155     52 21.6      Normal
## 38    164     47 17.5 Underweight
## 39    163     52 19.6      Normal
## 40    168     55 19.5      Normal
## 41    157     48 19.5      Normal
```

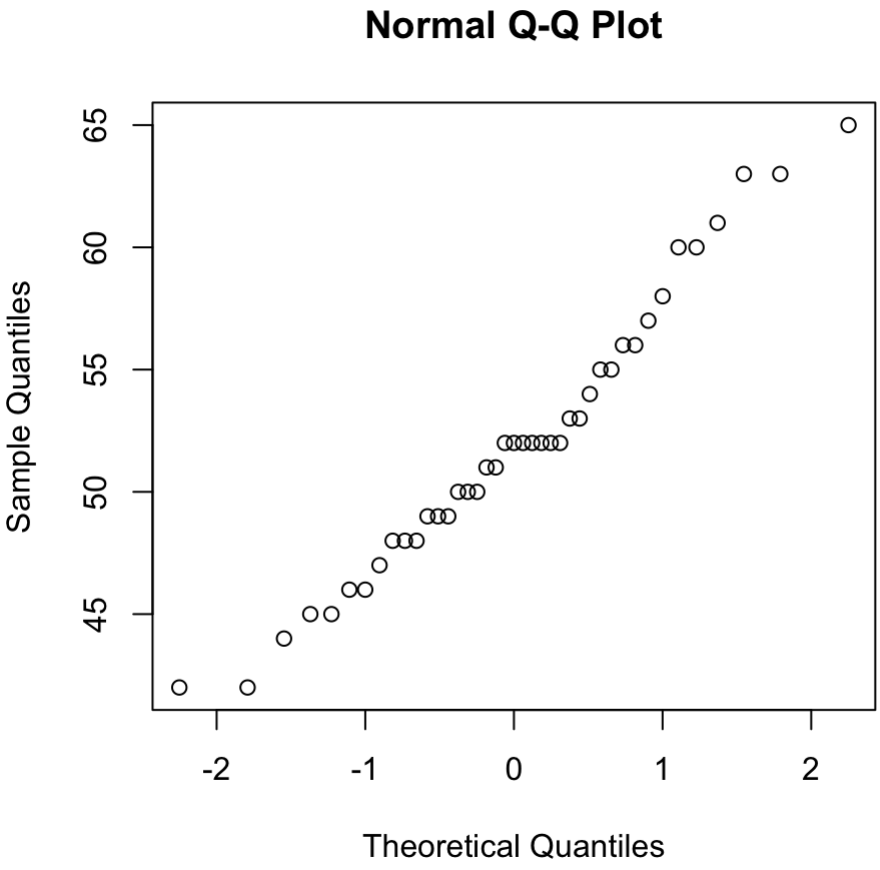
```
str(rn96)
```

```
## 'data.frame':    41 obs. of  4 variables:
##  $ height : int  161 155 158 170 160 156 162 158 158 167 ...
##  $ weight : int  50 49 42 65 60 52 58 46 45 51 ...
##  $ BMI    : num  19.3 20.4 16.8 22.5 23.4 21.4 22.1 18.4 18 18.3 ...
##  $ obesity: chr  "Normal" "Normal" "Underweight" "Normal" ...
```

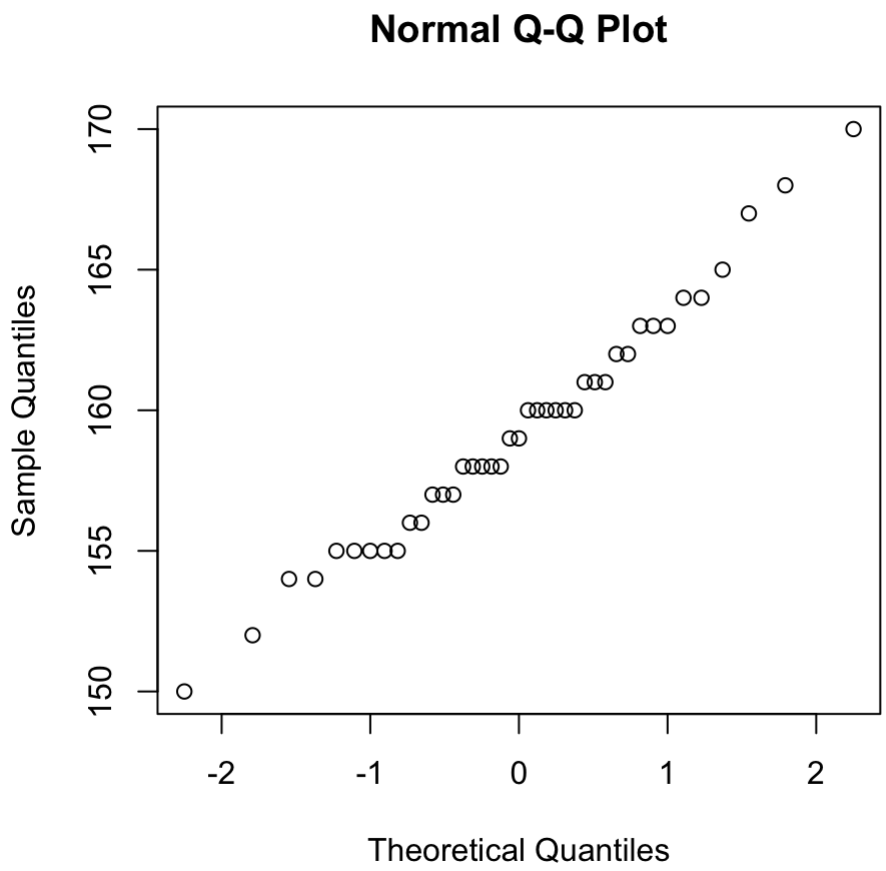
정규성(normality) 살펴보기

- `qqnorm()` 을 이용하여 각 변수가 정규분포에 가까운지 시각적으로 살펴보자.

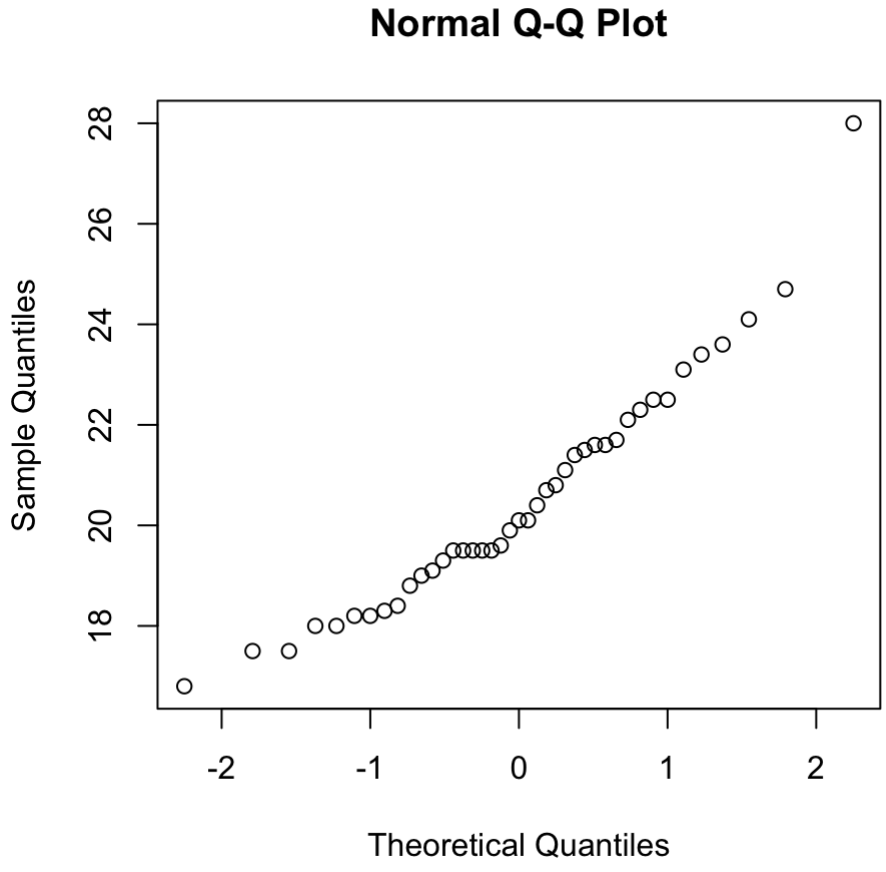
```
qqnorm(weight)
```




```
qqnorm(height)
```


Normal Q-Q Plot
Sample Quantiles
150
155
160
165
170
-2
-1
0
1
2
Theoretical Quantiles

```
qqnorm(BMI)
```


Normal Q-Q Plot
Sample Quantiles
18
20
22
24
26
28
-2
-1
0
1
2
Theoretical Quantiles

작업 폴더 정리하기

- `save()` 를 이용하면 작업 디렉토리에서 꼭 필요한 객체들만 모아서 저장해 놓을 수 있고, `save.image()` 를 이용하면 현재 작업 디렉토리에 있는 모든 객체를 저장하게 된다. 불러들일 때는 `load()` 를 이용한다. `rm()` 은 현재 디렉토리에 있는 객체 중에 삭제하고 싶은 것을 골라서 삭제하는 기능을 갖는다. 당연히 사용할 때 주의하여야 한다. 저장하는 다양한 방법을 살펴보자.
- 작업 history를 저장하고 나중에서 편집해서 다시 활용하려면 `savehistory()` 를 이용한다.

```
ls()
```

```
## [1] "BMI"     "height"  "rn96"    "rn96_2"  "rn96_3"  "rn96_lm" "weight"
```

```
save("rn96", "BMI", file = "./rn96_1.RData")
save(list = c("rn96", "BMI"), file = "./rn96_2.RData")
save(list = ls(), file = "./rn96_3.RData")
save.image(file = "./rn96_4.RData")
rm(list = ls())
ls()
```

```
## character(0)
```

```
load("./rn96_1.RData")
ls()
```

```
## [1] "BMI"  "rn96"
```

```
rm(list = ls())
ls()
```

```
## character(0)
```

```
load("./rn96_2.RData")
ls()
```

```
## [1] "BMI"  "rn96"
```

```
rm(list = ls())
ls()
```

```
## character(0)
```

```
load("./rn96_3.RData")
ls()
```

```
## [1] "BMI"     "height"  "rn96"    "rn96_2"  "rn96_3"  "rn96_lm" "weight"
```

```
rm(list = ls())
load("./rn96_4.RData")
ls()
```

```
## [1] "BMI"     "height"  "rn96"    "rn96_2"  "rn96_3"  "rn96_lm" "weight"
```

```
# savehistory(file = "./rn96.Rhistory")
```