

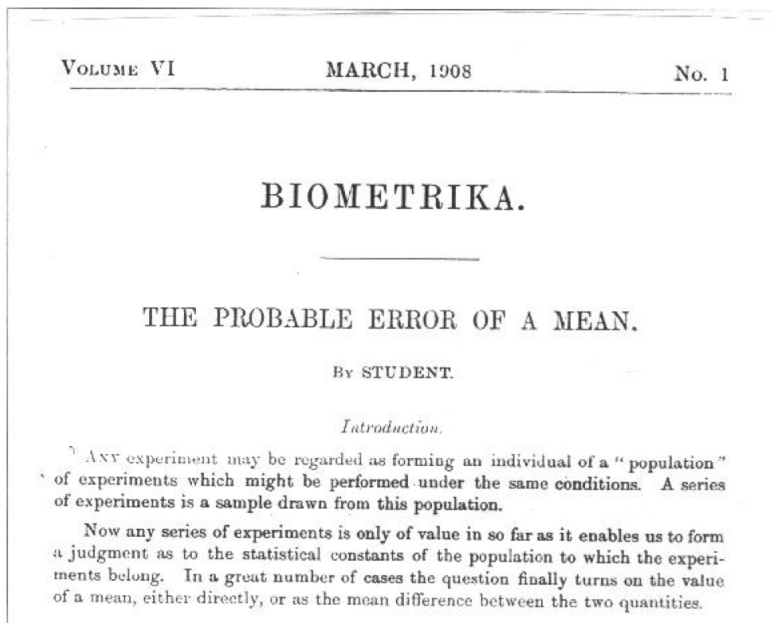
# Student 3000 Criminal Data

coop711

2025-10-08

## Biometrika 1908

```
include_graphics("../pics/biometrika_1908.png")
```



## Structure of Data

“Student”: William. S. Gosset(1876 ~ 1937)이 t-분포를 유도하기 위하여 모의실험에 활용한 자료는 다음과 같다.

```
options(width = 80)
crimtab
```

```
##      142.24 144.78 147.32 149.86 152.4 154.94 157.48 160.02 162.56 165.1 167.64
## 9.4      0      0      0      0      0      0      0      0      0      0      0
## 9.5      0      0      0      0      0      1      0      0      0      0      0
## 9.6      0      0      0      0      0      0      0      0      0      0      0
## 9.7      0      0      0      0      0      0      0      0      0      0      0
```

## 9.8	0	0	0	0	0	0	1	0	0	0	0
## 9.9	0	0	1	0	1	0	1	0	0	0	0
## 10	1	0	0	1	2	0	2	0	0	1	0
## 10.1	0	0	0	1	3	1	0	1	1	0	0
## 10.2	0	0	2	2	2	1	0	2	0	1	0
## 10.3	0	1	1	3	2	2	3	5	0	0	0
## 10.4	0	0	1	1	2	3	3	4	3	3	0
## 10.5	0	0	0	1	3	7	6	4	3	1	3
## 10.6	0	0	0	1	4	5	9	14	6	3	1
## 10.7	0	0	1	2	4	9	14	16	15	7	3
## 10.8	0	0	0	2	5	6	14	27	10	7	1
## 10.9	0	0	0	0	2	6	14	24	27	14	10
## 11	0	0	0	2	6	12	15	31	37	27	17
## 11.1	0	0	0	3	3	12	22	26	24	26	24
## 11.2	0	0	0	3	2	7	21	30	38	29	27
## 11.3	0	0	0	1	0	5	10	24	26	39	26
## 11.4	0	0	0	0	3	4	9	29	56	58	26
## 11.5	0	0	0	0	0	5	11	17	33	57	38
## 11.6	0	0	0	0	2	1	4	13	37	39	48
## 11.7	0	0	0	0	0	2	9	17	30	37	48
## 11.8	0	0	0	0	1	0	2	11	15	35	41
## 11.9	0	0	0	0	1	1	2	12	10	27	32
## 12	0	0	0	0	0	0	1	4	8	19	42
## 12.1	0	0	0	0	0	0	0	2	4	13	22
## 12.2	0	0	0	0	0	0	1	2	5	6	23
## 12.3	0	0	0	0	0	0	0	0	4	8	10
## 12.4	0	0	0	0	0	0	1	1	1	2	7
## 12.5	0	0	0	0	0	0	0	1	0	1	3
## 12.6	0	0	0	0	0	0	0	0	0	1	0
## 12.7	0	0	0	0	0	0	0	0	0	1	1
## 12.8	0	0	0	0	0	0	0	0	0	0	1
## 12.9	0	0	0	0	0	0	0	0	0	0	0
## 13	0	0	0	0	0	0	0	0	0	0	3
## 13.1	0	0	0	0	0	0	0	0	0	0	0
## 13.2	0	0	0	0	0	0	0	0	0	0	1
## 13.3	0	0	0	0	0	0	0	0	0	0	0
## 13.4	0	0	0	0	0	0	0	0	0	0	0
## 13.5	0	0	0	0	0	0	0	0	0	0	0
##	170.18	172.72	175.26	177.8	180.34	182.88	185.42	187.96	190.5	193.04	195.58
## 9.4	0	0	0	0	0	0	0	0	0	0	0
## 9.5	0	0	0	0	0	0	0	0	0	0	0
## 9.6	0	0	0	0	0	0	0	0	0	0	0
## 9.7	0	0	0	0	0	0	0	0	0	0	0
## 9.8	0	0	0	0	0	0	0	0	0	0	0
## 9.9	0	0	0	0	0	0	0	0	0	0	0
## 10	0	0	0	0	0	0	0	0	0	0	0
## 10.1	0	0	0	0	0	0	0	0	0	0	0
## 10.2	0	0	0	0	0	0	0	0	0	0	0
## 10.3	0	0	0	0	0	0	0	0	0	0	0
## 10.4	0	0	0	0	0	0	0	0	0	0	0
## 10.5	1	0	1	0	0	0	0	0	0	0	0
## 10.6	0	0	1	0	0	0	0	0	0	0	0
## 10.7	1	2	0	0	0	0	0	0	0	0	0
## 10.8	2	1	0	0	0	0	0	0	0	0	0

```
## 10.9      4      1      0      0      0      0      0      0      0      0      0
## 11       10      6      0      0      0      0      0      0      0      0      0
## 11.1      7      4      1      0      0      0      0      0      0      0      0
## 11.2     20      4      1      0      0      0      0      0      0      0      1
## 11.3     24      7      2      0      0      0      0      0      0      0      0
## 11.4     22     10     11      0      0      0      0      0      0      0      0
## 11.5     34     25     11      2      0      0      0      0      0      0      0
## 11.6     38     27     12      2      2      0      1      0      0      0      0
## 11.7     45     24      9      9      2      0      0      0      0      0      0
## 11.8     34     29     10      5      1      0      0      0      0      0      0
## 11.9     35     19     10      9      3      1      0      0      0      0      0
## 12       39     22     16      8      2      2      0      0      0      0      0
## 12.1     28     15     27     10      4      1      0      0      0      0      0
## 12.2     17     16     11      8      1      1      0      0      0      0      0
## 12.3     13     20     23      6      5      0      0      0      0      0      0
## 12.4     12      4      7      7      1      0      0      1      0      0      0
## 12.5     12     11      8      6      8      0      2      0      0      0      0
## 12.6      3      5      7      8      6      3      1      1      0      0      0
## 12.7      7      5      5      8      2      2      0      0      0      0      0
## 12.8      2      3      1      8      5      3      1      1      0      0      0
## 12.9      1      2      2      0      1      1      0      0      0      0      0
## 13       0      1      0      1      0      2      1      0      0      0      0
## 13.1      1      1      0      0      0      0      0      0      0      0      0
## 13.2      1      0      1      0      3      0      0      0      0      0      0
## 13.3      0      0      0      0      0      1      0      1      0      0      0
## 13.4      0      0      0      0      0      0      0      0      0      0      0
## 13.5      0      0      0      0      0      0      1      0      0      0      0
```

```
str(crimtab)
```

```
## 'table' int [1:42, 1:22] 0 0 0 0 0 0 1 0 0 0 ...
## - attr(*, "dimnames")=List of 2
## ..$ : chr [1:42] "9.4" "9.5" "9.6" "9.7" ...
## ..$ : chr [1:22] "142.24" "144.78" "147.32" "149.86" ...
```

자료구조에서 살필 수 있다시피 crimtab은 (왼손 가운데)손가락 길이와 키를 계급으로 나누고 손가락 길이와 키의 조합을 이름으로 갖는 table이다.

각 cell의 값은 길이와 키의 조합이 나타나는 빈도이다.

여기서 키를 인치 단위로 환원하면 어떤 모양이 드러나는지 살펴보자.

cm 단위로 되어 있는 키의 계급을 인치 단위로 변환하면 숨어있던 자료구조가 드러난다. (1인치 = 2.54cm)

이러한 table구조의 행과 열에 이름을 붙이려면, list를 사용한다.dimnames설정 과정에서 list(finger = ..., height = ...) 같은 방법으로 이름을 부여할 경우와 그렇지 않을 때 차이가 무엇인지 알아 두자.

```
crimtab_2 <- crimtab
colnames(crimtab_2) <- as.numeric(colnames(crimtab_2))/ 2.54
dimnames(crimtab_2) <- list(finger = rownames(crimtab_2),
                             height = colnames(crimtab_2))
crimtab_2
```

```
##          height
## finger 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77
```

```

## 9.4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 9.5 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 9.6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 9.7 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 9.8 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 9.9 0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 10 1 0 0 1 2 0 2 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
## 10.1 0 0 0 1 3 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
## 10.2 0 0 2 2 2 1 0 2 0 1 0 0 0 0 0 0 0 0 0 0 0 0
## 10.3 0 1 1 3 2 2 3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 10.4 0 0 1 1 2 3 3 4 3 3 0 0 0 0 0 0 0 0 0 0 0 0
## 10.5 0 0 0 1 3 7 6 4 3 1 3 1 0 1 0 0 0 0 0 0 0 0
## 10.6 0 0 0 1 4 5 9 14 6 3 1 0 0 1 0 0 0 0 0 0 0 0
## 10.7 0 0 1 2 4 9 14 16 15 7 3 1 2 0 0 0 0 0 0 0 0 0
## 10.8 0 0 0 2 5 6 14 27 10 7 1 2 1 0 0 0 0 0 0 0 0 0
## 10.9 0 0 0 0 2 6 14 24 27 14 10 4 1 0 0 0 0 0 0 0 0 0
## 11 0 0 0 2 6 12 15 31 37 27 17 10 6 0 0 0 0 0 0 0 0 0
## 11.1 0 0 0 3 3 12 22 26 24 26 24 7 4 1 0 0 0 0 0 0 0 0
## 11.2 0 0 0 3 2 7 21 30 38 29 27 20 4 1 0 0 0 0 0 0 0 1
## 11.3 0 0 0 1 0 5 10 24 26 39 26 24 7 2 0 0 0 0 0 0 0 0
## 11.4 0 0 0 0 3 4 9 29 56 58 26 22 10 11 0 0 0 0 0 0 0 0
## 11.5 0 0 0 0 0 5 11 17 33 57 38 34 25 11 2 0 0 0 0 0 0 0
## 11.6 0 0 0 0 2 1 4 13 37 39 48 38 27 12 2 2 0 1 0 0 0 0
## 11.7 0 0 0 0 0 2 9 17 30 37 48 45 24 9 9 2 0 0 0 0 0 0
## 11.8 0 0 0 0 1 0 2 11 15 35 41 34 29 10 5 1 0 0 0 0 0 0
## 11.9 0 0 0 0 1 1 2 12 10 27 32 35 19 10 9 3 1 0 0 0 0 0
## 12 0 0 0 0 0 0 1 4 8 19 42 39 22 16 8 2 2 0 0 0 0 0
## 12.1 0 0 0 0 0 0 0 2 4 13 22 28 15 27 10 4 1 0 0 0 0 0
## 12.2 0 0 0 0 0 0 1 2 5 6 23 17 16 11 8 1 1 0 0 0 0 0
## 12.3 0 0 0 0 0 0 0 0 4 8 10 13 20 23 6 5 0 0 0 0 0 0
## 12.4 0 0 0 0 0 0 1 1 1 2 7 12 4 7 7 1 0 0 1 0 0 0
## 12.5 0 0 0 0 0 0 0 1 0 1 3 12 11 8 6 8 0 2 0 0 0 0
## 12.6 0 0 0 0 0 0 0 0 0 1 0 3 5 7 8 6 3 1 1 0 0 0
## 12.7 0 0 0 0 0 0 0 0 0 1 1 7 5 5 8 2 2 0 0 0 0 0
## 12.8 0 0 0 0 0 0 0 0 0 0 1 2 3 1 8 5 3 1 1 0 0 0
## 12.9 0 0 0 0 0 0 0 0 0 0 0 1 2 2 0 1 1 0 0 0 0 0
## 13 0 0 0 0 0 0 0 0 0 0 0 3 0 1 0 1 0 2 1 0 0 0
## 13.1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0
## 13.2 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 3 0 0 0 0 0
## 13.3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
## 13.4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## 13.5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

```

```
str(crimtab_2)
```

```

## 'table' int [1:42, 1:22] 0 0 0 0 0 0 1 0 0 0 ...
## - attr(*, "dimnames")=List of 2
## ..$ finger: chr [1:42] "9.4" "9.5" "9.6" "9.7" ...
## ..$ height: chr [1:22] "56" "57" "58" "59" ...

```

## Data Frame 변환

글자 속성을 갖는 손가락 길이와 키의 조합에 따른 빈도를 또 다른 변수로 갖는 data frame으로 변환하고 이를 다시 long format 으로 전환하는 과정은 다음과 같음.

- 1차적으로 각 손가락 길이와 키의 조합을 갖는 인원수로 구성된 data frame으로 변환하고, case 단위의 long format data frame으로 2단계 변환.
  - 단순히 data.frame()으로 변환할 경우 Factor로 설정되어 numeric으로 변환하더라도 의미없는 숫자를 얻게 됨.
  - as.data.frame()의 결과물로 두 개의 character vector 와 counts를 나타내는 새로운 변수 Freq가 나오게 됨. 손가락 길이와 키를 나타내는 character를 numeric으로 전환하고 다음 작업 진행. Factor는 본질적으로 음이 아닌 정수이기 때문임.
- sapply를 이용하여 두 변수의 속성을 한번에 글자(character)에서 숫자(numeric)으로 변환함.

```
#> R 4.0.0      `stringsAsFactors = FALSE`  
crimtab_df <- as.data.frame(crimtab_2,  
                           stringsAsFactors = FALSE)  
str(crimtab_df)
```

```
## 'data.frame':   924 obs. of  3 variables:  
## $ finger: chr  "9.4" "9.5" "9.6" "9.7" ...  
## $ height: chr  "56" "56" "56" "56" ...  
## $ Freq : int  0 0 0 0 0 0 1 0 0 0 ...
```

```
crimtab_df[1:2] <- sapply(crimtab_df[1:2],  
                         FUN = as.numeric)  
# crimtab_df[1:2] %<>%  
# sapply(FUN = as.numeric)  
str(crimtab_df)
```

```
## 'data.frame':   924 obs. of  3 variables:  
## $ finger: num  9.4 9.5 9.6 9.7 9.8 9.9 10 10.1 10.2 10.3 ...  
## $ height: num  56 56 56 56 56 56 56 56 56 56 ...  
## $ Freq : int  0 0 0 0 0 0 1 0 0 0 ...
```

손가락 길이와 키의 조합이 나타나는 빈도만큼 그 조합을 반복하는 long format으로 변환하기 위하여 sapply 활용. apply를 사용하는 방법도 주석처리하여 보여 줌.

```
# crimtab_long <- apply(crimtab_df[, 1:2], MARGIN = 2, function(x) rep(x, crimtab_df[, 3]))  
# crimtab_long <- apply(crimtab_df[, c("finger", "height")], MARGIN = 2, function(x) rep(x, crimtab_df[, 3]))  
# crimtab_long <- apply(crimtab_df[, c("finger", "height")], MARGIN = 2, function(x) rep(x, crimtab_df[, 3]))  
# crimtab_long <- apply(crimtab_df[1:2], MARGIN = 2, function(x) rep(x, crimtab_df[, 3]))  
crimtab_long <- sapply(crimtab_df[, c("finger", "height")],  
                     function(x) rep(x, crimtab_df$Freq))  
# crimtab_long <- mapply(function(x) rep(x, crimtab_df$Freq), crimtab_df[, c("finger", "height")])  
str(crimtab_long)
```

```
## num [1:3000, 1:2] 10 10.3 9.9 10.2 10.2 10.3 10.4 10.7 10 10.1 ...  
## - attr(*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:2] "finger" "height"
```

long format 으로 변환하는 다른 방법 1

```
# finger_long <- rep(crimtab_df$finger, crimtab_df$Freq)
# height_long <- rep(crimtab_df$height, crimtab_df$Freq)
# crimtab_long_df <- data.frame(finger = finger_long, height = height_long)
# str(crimtab_long_df)
```

long format 으로 변환하는 다른 방법 2

```
# index_crimtab <- rep(1:nrow(crimtab_df), crimtab_df[, "Freq"])
# index_crimtab[1:10]
# crimtab_df_long <- crimtab_df[index_crimtab, c("finger", "height")]
# str(crimtab_df_long)
```

matrix 를 data frame으로 변환

```
crimtab_long_df <- data.frame(crimtab_long)
str(crimtab_long_df)
```

```
## 'data.frame': 3000 obs. of 2 variables:
## $ finger: num 10 10.3 9.9 10.2 10.2 10.3 10.4 10.7 10 10.1 ...
## $ height: num 56 57 58 58 58 58 58 58 59 59 ...
```

Save for future works

```
save.image("./crimtab.RData")
```