13. 이변량 자료 탐색

데이터 세트를 구성하고 있는 각 개별 변수들을 대상으로 이루어지는 통계분석이 11장과 12장의 주 내용이었다. 본 장에서는 두 변수를 동시에 고려하는 통계분석 방법을 살펴보고자 한다. 이것은 두 변수로 구성된 이변량(bivariate) 자료를 대상으로 이루어지는 분석을 의미하는 것으로, 일변량 자료에 대해서는 자료의 분포에 대한 정보만이 필요했지만, 이변량 자료에 대해서는 각 자료의 분포에 대한 정보뿐만이 아니라, 두 변수의 분포를 비교하는 것과 두 변수 간의 관계가 중요한 정보가 된다.

본 장에서는 이변량 범주형 자료를 분할표로 정리하는 방법 및 그래프로 나타내는 방법을 살펴볼 것이며, 이변량 연속형 자료의 비교 및 관계를 나타내는 데 적합한 그래프 및 요약 통계량에 관련된 R 함수들을 살펴볼 것이다. 연속형 자료의 비교는 이변량 이상의 자료에서도 적용될 수 있는 내용이 되며, 이변량 이상의 자료 탐색을 위한 그래프도 살펴보겠다.

13.1 이변량 범주형 자료의 정리

범주형 자료가 주어지면 가장 먼저 해야 할 작업은 도수분포표 혹은 분할표를 작성하는 것이다. 일변량 자료에 대한 1차원 도수분포표는 함수 `table()`을 이용하여 작성한 바 있다. 본 절에서는 1차원 도수분포표뿐만이 아니라 2차원 분할표의 작성에도 사용할 수 있는 R 함수를 살펴볼 것이다. 또한 2차원 분할표로 정리된 자료를 효과적으로 시각화할 수 있는 그래프 작성법도 살펴볼 것이다.

13.1.1 분할표 작성

R에는 도수분포표 및 분할표를 작성하는데 사용되는 몇 가지 함수가 있다. 그 중 많이 사용되는 함수들이 표 13.1에 정리되어 있다.

<표 13.1> 도수분포표 및 분할표 작성에 관련된 R 함수

함수	대략의 기능
table(var1, var2, . . . , varN)	N개의 범주형 변수로 N차원 분할표 작성
xtabs(formula, data)	data에 있는 범주형 변수를 formula에 정의된 방식으로 분할표 작성
prop.table(table, margins)	작성된 분할표의 결합분포 혹은 margins로 정의된 방향으로 조건분포 작성
margin.table(table, margins)	작성된 분할표의 한계분포를 margins로 정의된 방향으로 작성
addmargins(table, margins)	작성된 분할표에 margins로 정의된 방향의 한계분포를 추가

위 함수들의 사용법을 예제를 이용하여 살펴보자. 예제로 사용할 자료는 패키지 vcd에 있는 데이터 프레임 Arthritis이다. 이 자료는 류머티즘 관절염 환자들을 대상으로 새로운 치료법의 효과를 알아보기 위해 이루어진 임상실험 자료이다.

```
> data(Arthritis, package="vcd")
> head(Arthritis)
  ID Treatment  Sex Age Improved
1 57   Treated Male  27     Some
2 46   Treated Male  29     None
3 77   Treated Male  30     None
4 17   Treated Male  32   Marked
5 36   Treated Male  46   Marked
6 23   Treated Male  58   Marked
```

변수 Treatment("Placebo", "Treated")와 Sex("Male", "Female"), Improved("None", "Some", "Marked")는 모두 범주형 자료다. 이제 표 13.1에 정리된 함수들을 이용하여 다양한 형태의 도수분포표 및 분할표를 작성해보자.

- **1차원 도수분포표 작성**

함수 table()과 xtabs()로 변수 Improved의 1차원 도수분포표를 작성해보자.

```
> my_table1 <- with(Arthritis, table(Improved))
> my_table1
Improved
  None   Some Marked
    42     14     28

> xtabs(~Improved, data=Arthritis)
Improved
  None   Some Marked
    42     14     28
```

우선 두 명령문의 실행결과는 동일함을 알 수 있다. 사용법에서는 약간의 차이가 있는데, 함수 table() 안에는 데이터 프레임을 선언할 수 없기 때문에 함수 with()와 함께 사용하였다. 함수 xtabs() 안에는 R 공식으로 도수분포표 혹은 분할표의 형태를 정의하는데, 1차원 도수분포표의 경우는 '~' 표시 오른쪽에 해당 변수를 하나 입력하면 된다. 공식을 사용하여 모형을 정의하는 함수 안에는 데이터 프레임을 선언하여 사용할 수 있는 것이 일반적이다.

- **1차원 상대도수분포표 작성**

상대도수분포표는 함수 table() 또는 xtabs()로 작성된 도수분포표를 함수 prop.table()에 입력하면 된다.

```
> prop.table(my_table1)
Improved
     None      Some    Marked
0.5000000 0.1666667 0.3333333
```

위의 결과에서 우리는 소수점 자릿수가 너무 길다는 문제를 발견 할 수 있다. 소수점 자릿수의 조정은 함수 options()으로 할 수 있는데, 이 함수는 R의 전체적인 환경에 관련된 여러 옵션을 조정하는 가능을 갖고 있다. 이 중 출력되는 소수점 자릿수의 현재 값을 알아보는 방법과 수정하는 방법은 다음과 같다.

```
> options("digits")
$digits
[1] 7

> options("digits"=2)
```

원래의 값은 소수점 이하 7자리까지 출력하는 것인데, 이것을 2자리로 수정하고 다시 변수 Improved의 상대도수분포표를 출력해 보자.

```
> prop.table(my_table1)
Improved
  None   Some Marked
  0.50   0.17   0.33
```

- **2차원 분할표 작성**

1차원 도수분포표의 경우와 같이 2차원 분할표도 함수 table() 혹은 xtabs()로 작성할 수 있다. 변수 Improved와 Treatment의 2차원 분할표를 작성해보자.

```
> my_table2 <- with(Arthritis, table(Treatment, Improved))
> # 혹은
> my_table2 <- xtabs(~Treatment+Improved, data=Arthritis)

> my_table2
         Improved
Treatment None Some Marked
  Placebo   29    7      7
  Treated   13    7     21
```

함수 table()에서는 두 개의 변수가 콤마로 연결되고 함수 xtabs()에서는 두 변수가 + 기호로 연결되는데, 두 경우 모두 첫 번째 변수가 분할표의 행 변수가 되고, 두 번째 변수가 열 변수가 된다.

- **2차원 상대도수분할표(결합분포) 작성**

작성된 분할표를 함수 prop.table()에 입력하면 된다.

```
> prop.table(my_table2)
          Improved
Treatment  None  Some Marked
```

```
Placebo 0.345 0.083  0.083
Treated 0.155 0.083  0.250
```

위에서 options("digits"= 2)가 선언된 상태이지만 소수점 3자리까지 출력된 것을 볼 수 있다. 이것은 "digits"에 대한 함수 options()의 선언이 일종의 기대값이기 때문이다.

- **2차원 분할표의 한계분포(marginal distribution) 작성**

2차원 분할표에서 한계분포를 작성하는 방법은 행 또는 열에 대한 합계를 구하는 것으로 함수 margin.table()로 작성할 수 있다. 사용법은 margin.table(x, margin=NULL)이 되는데, x는 함수 table() 등으로 작성된 분할표나 행렬이 되고, margin은 1은 행, 2는 열을 지정하는 것으로 실질적으로 이 함수는 apply(x, 1, sum)과 apply(x, 2, sum)과 동일한 작업을 수행한다. 또한 margin에 아무것도 지정하지 않으면 sum(x)를 출력한다.

```
> my_table2
         Improved
Treatment None Some Marked
  Placebo    29    7      7
  Treated    13    7     21

> margin.table(my_table2, 1)
Treatment
Placebo Treated
     43      41

> margin.table(my_table2, 2)
Improved
  None   Some Marked
    42     14     28

> margin.table(my_table2)
[1] 84
```

margin.table(my_table2, 1)은 행 방향으로 합계를 구하는 것이므로 행 변수 Treatment의 도수분포표 즉, 한계분포표가 작성되었고, margin.table(my_table2, 2)는 열 방향의 합계를 구하는 것으로 열 변수 Improved의 한계분포표가 작성되었다. margin.table(my_table2)는 margin에 아무것도 지정하지 않은 것으로 이 경우에는 관찰값의 총 개수가 계산된다.

한계분포표를 상대도수로 표현하려면 함수 margin.table()의 결과를 prop.table()에 입력하면 된다.

```
> prop.table(margin.table(my_table2, 1))
Treatment
Placebo Treated
   0.51    0.49
```

2차원 분할표에 한계분포표를 추가하려면 함수 addmargins()를 사용하면 된다.

```
> addmargins(my_table2)
        Improved
Treatment None Some Marked  Sum
  Placebo    29    7      7   43
  Treated    13    7     21   41
  Sum        42   14     28   84

> addmargins(prop.table(my_table2))
        Improved
Treatment  None  Some Marked   Sum
  Placebo 0.345 0.083  0.083 0.512
  Treated 0.155 0.083  0.250 0.488
  Sum     0.500 0.167  0.333 1.000
```

행 또는 열 변수 중 한 변수의 한계분포만을 추가하려면 함수 addmargins()에 margin에 관련된 숫자를 지정하면 된다.

```
> addmargins(prop.table(my_table2), 1)
        Improved
Treatment  None  Some Marked
  Placebo 0.345 0.083  0.083
  Treated 0.155 0.083  0.250
  Sum     0.500 0.167  0.333

> addmargins(prop.table(my_table2), 2)
        Improved
Treatment  None  Some Marked   Sum
  Placebo 0.345 0.083  0.083 0.512
  Treated 0.155 0.083  0.250 0.488
```

함수 addmargins()에 margin=1이 입력되면 행에 한계분포가 추가되고, margin=2가 입력되면 열에 한계분포가 추가된다.

- **조건분포 작성**

조건분포는 함수 prop.table()에 margin을 지정함으로써 작성할 수 있다. 두 범주형 변수의 결합분포인 2차원 상대도수분할표보다 두 변수 사이의 관계를 규명하는데 더 도움이 된다. 행 변수를 조건을 할 것인지 혹은 열 변수를 조건으로 할 것인지는 함수 prop.table()의 옵션 margin에 1 또는 2 중 하나를 선택하면 된다.

```
> prop.table(my_table2, 1)
        Improved
Treatment None Some Marked
  Placebo 0.67 0.16   0.16
  Treated 0.32 0.17   0.51

> prop.table(my_table2, 2)
        Improved
Treatment None Some Marked
  Placebo 0.69 0.50   0.25
  Treated 0.31 0.50   0.75
```

옵션 margin에 1을 입력한 첫 번째 경우는 행 변수 Treatment가 조건변수가 되어, 위약 처리한 Placebo에서는 16%의 환자에게서 큰 진전이 있었지만 새로운 처리방법으로 처리된 Treated에서는 51%의 환자에게서 큰 진전이 있었다는 것을 알 수 있다. 또한 옵션 margin에 2를 입력한 두 번째 경우는 열 변수 Improved가 조건변수가 된 경우로 병세가 많이 호전된 Marked 그룹 중에 75%의 환자들이 Treated 그룹에 속한다는 것도 알 수 있다.

[유용한 정보]

1. 함수 table()은 자료에 포함된 NA 값을 무시하고 분할표를 작성하는 것이 디폴트이다. 만일 NA의 빈도수를 분할표에 나타내려면 옵션 useNA="ifany"를 추가하면 된다. 예를 들어 데이터 프레임 airquality에서 변수 Ozone의 값이 80이 넘는 날짜가 며칠이나 되는지 월별로 나타내는 분할표를 작성해 보자.

```
> with(airquality, table(OzHi=Ozone > 80, Month))
       Month
OzHi     5  6  7  8  9
  FALSE 25  9 20 19 27
  TRUE   1  0  6  7  2
```

위의 분할표를 보면 5월은 26일, 6월은 9일 밖에 없는 등 많은 날짜가 생략되어 있음을 알 수 있다. 이것은 자료에 NA가 있기 때문이며 이것을 모두 포함시키려면 다음과 같이 실행하면 된다.

```
> with(airquality,
     table(OzHi=Ozone > 80, Month, useNA="ifany"))
      Month
OzHi     5  6  7  8  9
  FALSE 25  9 20 19 27
  TRUE   1  0  6  7  2
  <NA>   5 21  5  5  1
```

2. SAS의 PROC FREQ나 SPSS의 CROSSTABS의 실행결과와 비슷한 형태의 분할표를 얻고자 한다면 패키지 gmodels의 함수 CrossTable()을 사용해야 한다.

```
> library(gmodels)
> with(Arthritis, CrossTable(Treatment, Improved))
```

```
   Cell Contents
|-------------------------|
|                       N |
| Chi-square contribution |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  84

                    | Arthritis$Improved
Arthritis$Treatment |       None |       Some |     Marked | Row Total |
--------------------|------------|------------|------------|-----------|
            Placebo |         29 |          7 |          7 |        43 |
                    |      2.616 |      0.004 |      3.752 |           |
                    |      0.674 |      0.163 |      0.163 |     0.512 |
                    |      0.690 |      0.500 |      0.250 |           |
                    |      0.345 |      0.083 |      0.083 |           |
--------------------|------------|------------|------------|-----------|
            Treated |         13 |          7 |         21 |        41 |
                    |      2.744 |      0.004 |      3.935 |           |
                    |      0.317 |      0.171 |      0.512 |     0.488 |
                    |      0.310 |      0.500 |      0.750 |           |
                    |      0.155 |      0.083 |      0.250 |           |
--------------------|------------|------------|------------|-----------|
       Column Total |         42 |         14 |         28 |        84 |
                    |      0.500 |      0.167 |      0.333 |           |
--------------------|------------|------------|------------|-----------|
```

위 결과는 SAS 결과물 형태로 출력된 것이며 SPSS 결과물 형태로 출력하고자 한다면 옵션 format="SPSS"을 추가하면 된다. 자료의 도수와 기대도수만을 나타내고, 두 변수의 독립성 검정을 하고자 한다면 옵션 prop.r, prop.c, prop.t, prop.chisq에 FALSE를 지정하고, expected=TRUE와 chisq=TRUE를 추가하면 된다.

```
> with(Arthritis, CrossTable(Treatment, Improved,
       prop.r=FALSE, prop.c=FALSE, prop.t=FALSE,
       prop.chisq=FALSE, expected=TRUE, chisq=TRUE))
```

```
             | Improved
   Treatment |       None |       Some |     Marked | Row Total |
-------------|------------|------------|------------|-----------|
     Placebo |         29 |          7 |          7 |        43 |
             |     21.500 |      7.167 |     14.333 |           |
-------------|------------|------------|------------|-----------|
     Treated |         13 |          7 |         21 |        41 |
             |     20.500 |      6.833 |     13.667 |           |
-------------|------------|------------|------------|-----------|
Column Total |         42 |         14 |         28 |        84 |
-------------|------------|------------|------------|-----------|

Statistics for All Table Factors

Pearson's Chi-squared test
------------------------------------------------------------
Chi^2 =  13     d.f. =  2     p =  0.0015
```

13.1.2 이변량 및 다변량 범주형 자료를 위한 그래프

이변량 범주형 자료를 위한 그래프를 작성하는 주된 목적은 두 범주형 변수의 관계를 탐색하는 것이다. 많이 사용되는 그래프는 막대 그래프이며, 이변량의 경우 쌓아 올린 형태의 막대 그래프와 옆으로 붙여 놓은 형태의 막대 그래프가 사용된다. 패키지 graphics에서는 함수

barplot()이, 패키지 ggplot2에서는 함수는 geom_bar()가 두 형태의 그래프를 작성할 수 있다. 또한 이변량 및 다변량 범주형 자료의 관계 탐색에 적합한 그래프로써 mosaic plot이 있는데, 이 그래프는 패키지 vcd의 함수 mosaic()으로 작성할 수 있다.

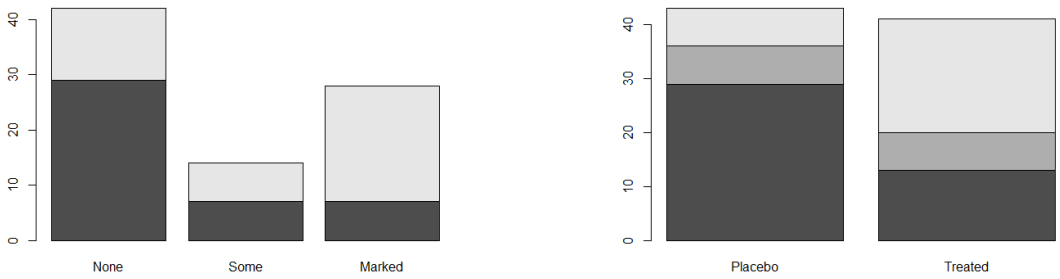
예제로 패키지 vcd의 데이터 프레임 Arthritis의 변수 Treatment와 Improved의 관계 탐색을 위한 그래프를 작성해 보자. 함수 barplot()으로 막대 그래프를 작성하기 위해 두 변수의 분할표를 함수 table()로 작성해 보자.

```
> data(Arthritis, package="vcd")
> my_table <- with(Arthritis, table(Treatment, Improved))
> my_table
         Improved
Treatment None Some Marked
  Placebo   29    7      7
  Treated   13    7     21
```

- 쌓아 올린 막대 그래프

쌓아 올린 막대 그래프는 작성된 두 변수의 분할표를 함수 barplot()에 입력하면 작성되는데, 주의할 점은 행 변수와 열 변수 중 어느 변수를 위주로 막대를 쌓아 올릴 것인지를 결정해야 한다는 것이다. 만일 두 변수가 설명변수와 반응변수의 성격을 갖고 있다면, 설명변수를 위주로 막대를 쌓는 것이 두 변수의 관계 탐색에 더 효과적일 것이다. 데이터 프레임 Arthritis의 경우에는 변수 Improved가 반응변수이고 나머지 변수는 모두 설명변수가 되기 때문에 변수 Treatment를 위주로 쌓아 올린 막대 그래프를 작성해 보자.

```
> # 그림 13.1 (a)
> barplot(my_table)
```



(a) (b)

<그림 13.1> 변수 Improved와 Treatment의 쌓아 올린 막대 그래프

그림 13.1 (a)의 막대 그래프는 열 변수인 Improved를 위주로 막대가 쌓아 올려졌음을 알 수 있다. 원하는 형태의 그래프를 작성하기 위해서는 두 변수 분할표의 열과 행을 서로 바꿔야 하는데, 이것은 전치 행렬(transposed matrix)을 만들어 주는 함수 t()를 객체 my_table에

적용하면 된다.

```
> # 그림 13.1 (b)
> barplot(t(my_table))
```

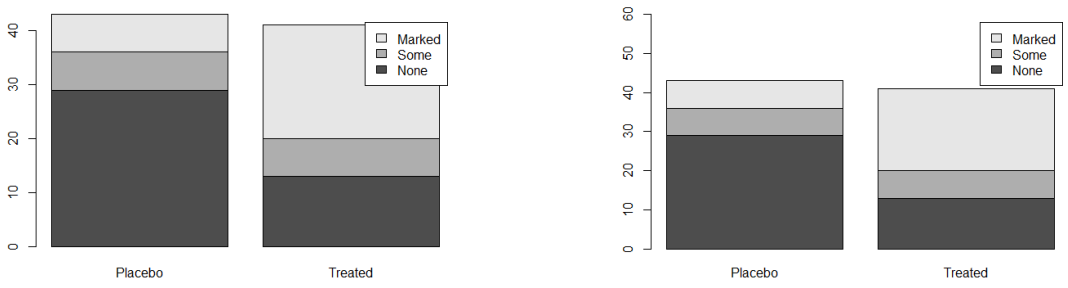
반응변수인 Treatment를 위주로 쌓아 올린 막대 그래프기 그림 13.1 (b)에 작성되어 있다. 위약 그룹과 실제 치료가 이루어진 그룹 안에서 병세 호전 정도의 분포를 확인할 수도 있으며, 두 그룹 사이에 병세 호전 정도의 차이를 확인 할 수 있는 그래프이다.

그림 13.1 (b)의 문제는 쌓아 올려진 막대에 대한 범례가 없다는 점이다. 함수 barplot()으로 쌓아 올린 막대 그래프를 작성할 때 범례를 추가하려면 옵션 legend.text에 문자형 벡터를 지정하거나 TRUE를 지정하면 된다. TRUE가 지정되면 입력된 분할표의 행 이름이 범례로 사용된다.

```
> # 그림 13.2 (a)
> barplot(t(my_table), legend.text=TRUE)
```

옵션 legend.text로 범례를 추가할 때의 문제는 범례의 위치가 자동으로 선택되어서 가끔은 그래프와 겹쳐지는 경우가 있다는 점이다. 이런 경우에 Y축의 범위를 확대해 주면 겹치는 문제는 해결할 수 있다.

```
> # 그림 13.2 (b)
> barplot(t(my_table), legend.text=TRUE, ylim=c(0,60))
```

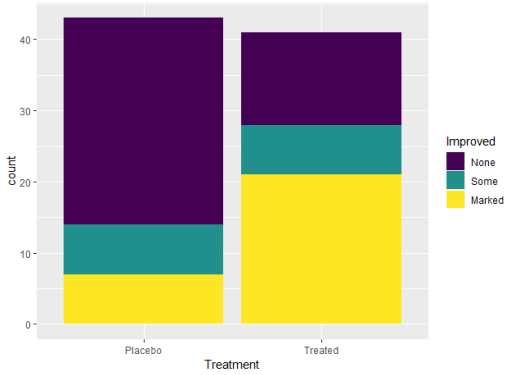


(a) (b)

<그림 13.2> 범례를 추가한 쌓아 올린 막대 그래프

패키지 ggplot2에서 쌓아 올린 막대 그래프는 함수 geom_bar()에서 시각적 요소 x에는 설명변수를, 시각적 요소 fill에는 반응변수를 연결하면, 설명변수를 위주로 쌓아 올린 막대 그래프를 작성할 수 있다.

```
> # 그림 13.3
> ggplot(Arthritis, aes(x=Treatment, fill=Improved)) +
    geom_bar()
```



<그림 13.3>패키지 ggplot2에서 작성된 쌓아 올린 막대 그래프

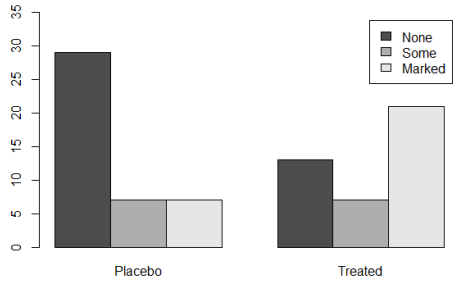
두 변수의 분할표를 데이터로 사용하여 쌓아 올린 막대 그래프를 작성해야 하는 경우에는 분할표를 함수 as.data.frame()에 입력하여 데이터 프레임으로 변환시키고, 설명변수인 Treatment를 x에, 빈도수인 Freq를 y에, 반응변수인 Improved를 fill에 매핑하고, 함수 geom_col() 또는 geom_bar(stat="identity")를 사용하면 된다.

```
> # 그림 13.3
> p <- ggplot(as.data.frame(my_table),
        aes(x=Treatment, y=Freq, fill=Improved))
> p + geom_col()
> p + geom_bar(stat="identity")
```

- 옆으로 붙여 놓은 막대 그래프

옆으로 붙여 놓은 막대 그래프를 작성해 보자. 함수 barplot()은 옵션 beside=TRUE를 추가하면 된다.

```
> # 그림 13.4
> barplot(t(my_table), beside=TRUE, legend.text=TRUE,
        ylim=c(0,35))
```

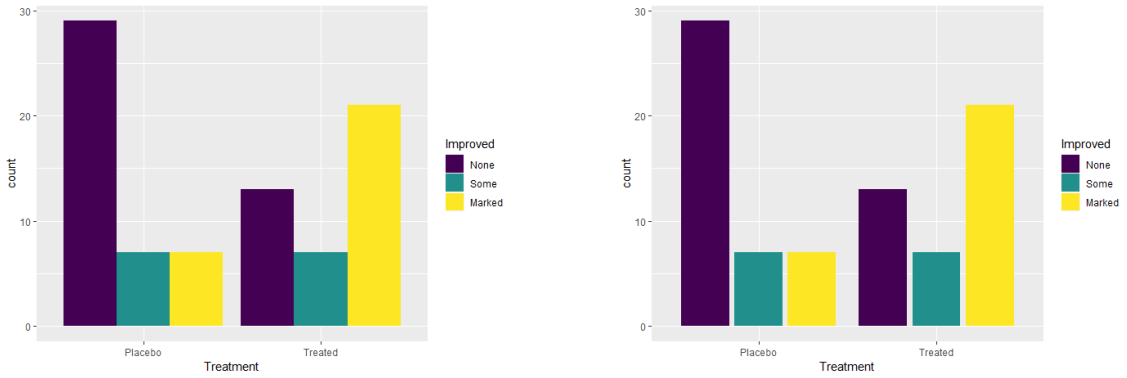


<그림 13.4> 옆으로 붙여 놓은 막대 그래프

함수 geom_bar()는 position에 "dodge" 혹은 "dodge2"를 지정하면 되는데, "dodge2"는 막대 폭을 조금 줄여서 막대 사이에 약간의 빈틈을 만들어 주는 차이가 있다.

```
> pp <- ggplot(Arthritis, aes(x=Treatment, fill=Improved))
```

```
> # 그림 13.5 (a)
> pp + geom_bar(position="dodge")

> # 그림 13.5 (b)
> pp + geom_bar(position="dodge2")
```

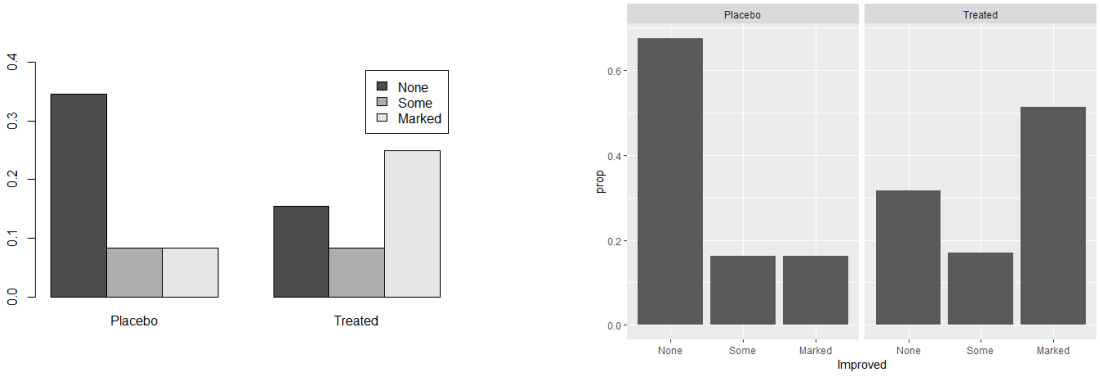


(a) (b)

<그림 13.5> 패키지 ggplot2에서 작성된 옆으로 붙여 놓은 막대 그래프

상대도수를 사용하여 옆으로 붙여 놓은 막대 그래프를 작성해 보자. 함수 barplot()의 경우에는 함수 prop.table()로 작성된 두 변수의 상대도수분할표를 입력하면 된다.

```
> # 그림 13.6 (a)
> barplot(t(prop.table(my_table)), beside=TRUE, legend.text=TRUE,
          ylim=c(0,0.4))
```



(a) (b)

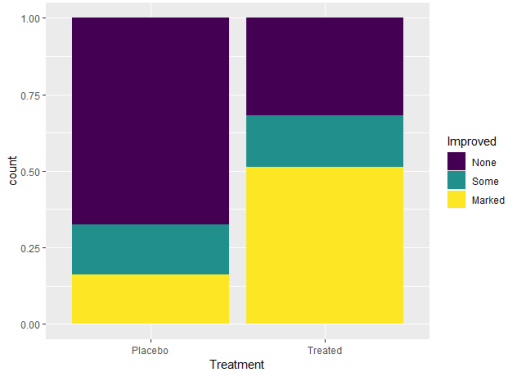
<그림 13.6> 상대도수에 의한 옆으로 붙여 놓은 막대 그래프

패키지 ggplot2에서는 함수 geom_bar()만으로는 원하는 그래프를 작성하기 어렵고, 함수 facet_wrap()을 추가로 사용해야 한다.

```
> # 그림 13.6 (b)
> ggplot(Arthritis, aes(x=Improved, y=..prop.., group=1)) +
    geom_bar() +
    facet_wrap(~Treatment)
```

함수 geom_bar()로 두 변수의 막대 그래프를 작성하는 경우, 시각적 요소 x와 fill에 각각 동일한 변수를 매핑하여도 position에 어떤 값을 지정하는가에 따라 다른 형태의 막대 그래프가 작성된다. 디폴트는 "stack"으로 쌓아 올린 형태가 되고, "dodge"는 옆으로 붙여 놓은 형태가 됨을 알 수 있었다. 가능한 다른 값으로 "fill"이 있는데, 이 경우에는 x 변수에 대한 fill 변수의 조건부 확률로 막대 그래프가 각각 작성된다. 따라서 각 막대의 높이는 1이 된다.

```
> # 그림 13.7
> ggplot(Arthritis, aes(x=Treatment, fill=Improved)) +
    geom_bar(position="fill")
```



<그림 13.7> position="fill"로 작성된 막대 그래프

그림 13.7에서 각 조각의 면적은 행 변수인 Treatment를 조건으로 하는 열 변수 Improved의 조건부 확률에 의하여 결정됨을 다음의 분할표와 비교해 보면 알 수 있다. 설명변수와 반응변수 사이의 관계를 파악하는데 더 적합하다고 할 수 있다.

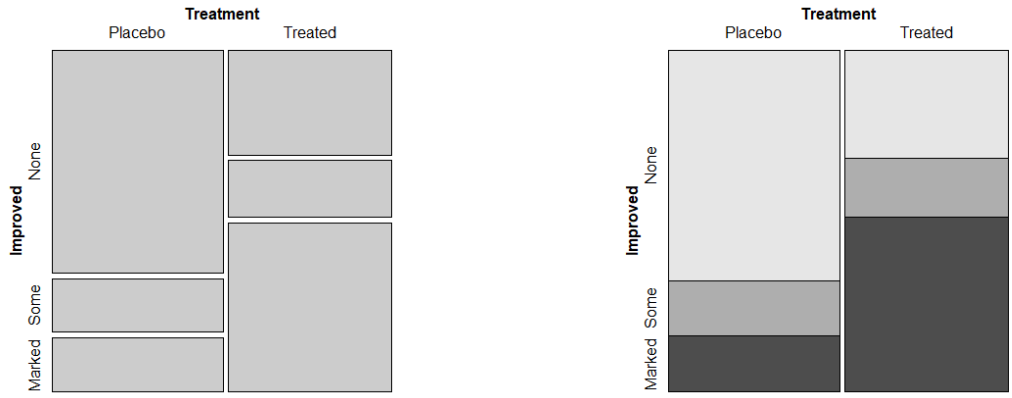
```
> prop.table(my_table, 1)
          Improved
Treatment  None  Some Marked
  Placebo 0.674 0.163  0.163
  Treated 0.317 0.171  0.512
```

- Mosaic plot

이변량 혹은 다변량 범주형 변수의 관계를 탐색할 때 유용하게 사용되는 그래프가 mosaic plot이다. 이 그래프는 쌓아 올린 막대 그래프를 훨씬 효과적으로 발전시킨 그래프라고 할 수 있으며, 패키지 vcd의 함수 mosaic()으로 작성할 수 있다. 먼저 두 변수의 분할표인 객체 my_table을 사용하여 작성해 보자.

```
> library(vcd)
```

```
> # 그림 13.8 (a)
> mosaic(my_table, direction="v")
```



(a) (b)

<그림 13.8> Mosaic plot

작성 방법은 두 변수의 분할표 행 변수(Treatment)의 상대도수(Placebo 0.512, Treated 0.488)의 비율에 따라 정사각형을 수직(direction="v")으로 분리한다. 이어서 수직으로 분리된 두 조각을 행 변수 Treatment를 조건으로 하는 열 변수 Improved의 조건부 확률에 비례하여 수평 방향으로 각각 분리하여 작성한다. 옵션 direction="v"가 생략되면 첫 번째 분할은 수평 방향(direction="h")으로 이루어지고, 따라서 두 번째 분할은 수직으로 이루어진다.

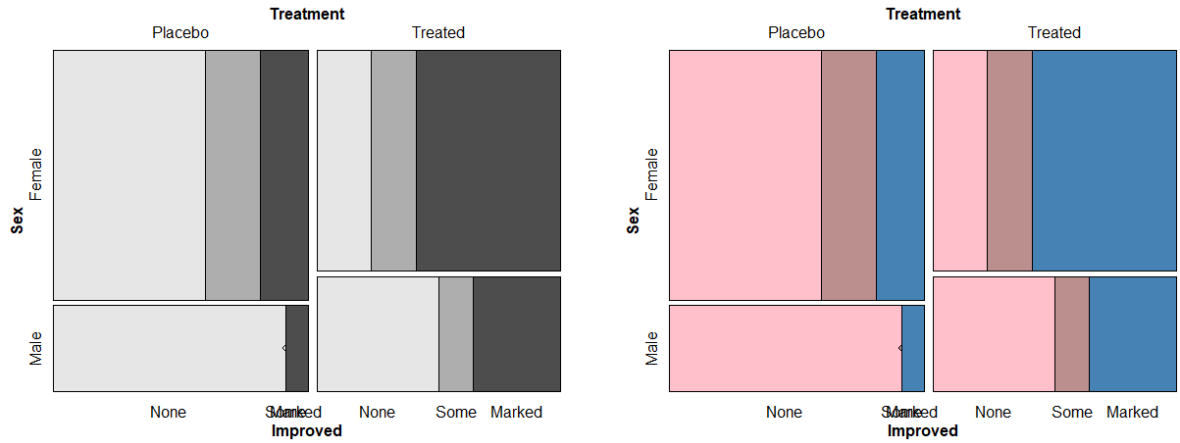
함수 mosaic()은 분할표가 아닌 R 공식으로도 그래프를 작성할 변수를 지정할 수 있다. 만일 '~' 기호의 오른쪽에만 mosaic(~var1+var2)의 형태로 변수를 입력을 하면 var1이 첫 번째 분할 변수, var2가 두 번째 분할 변수로 사용된다. 만일 mosaic(var1~var2)의 형태로 변수가 입력되면 var1이 반응변수, var2가 설명변수로 지정되는 것이며, 설명변수가 먼저 분할 변수로 사용되고 마지막으로 반응변수가 분할 변수로 사용된다. 또한 반응변수의 수준에 따라 각 조각이 다른 색으로 채워진다.

```
> # 그림 13.8 (a)
> mosaic(~Treatment+Improved, data=Arthritis, direction="v")

> # 그림 13.8 (b)
> mosaic(Improved~Treatment, data=Arthritis, direction="v")
```

이번에는 데이터 프레임 Arthritis에서 반응변수 Improved와 설명변수 Treatment, Sex의 관계를 mosaic plot으로 나타내 보자.

```
> # 그림 13.9 (a)
> mosaic(Improved~Treatment+Sex, data=Arthritis, direction="v")

> #그림 13.9 (b)
> mosaic(Improved~Treatment+Sex, data=Arthritis, direction="v",
```

```
gp=gpar(fill=c("pink","rosybrown","steelblue")))
```



(a) (b)

<그림 13.9> 세 변수의 mosaic plot

함수 mosaic()에서 선의 종류나 색 등의 그래픽 모수는 패키지 grid를 통해 조절이 된다. 그래픽 모수의 기본적인 설정은 gp=gpar(...)이 되는데, 다각형 도형을 색으로 채우는 옵션은 fill이 된다. 그림 13.9에서 확인할 수 있는 또 다른 사항은 Treatment가 Placebo이고, Sex가 Male인 그룹에서 Improved가 Some에 해당하는 부분에 점이 찍혀있는데, 그것은 이 그룹에 해당되는 자료가 없기 때문이다. 이때 사용되는 점의 크기는 옵션 zero_size로 조절할 수 있다.

Mosaic plot을 포함한 범주형 자료의 시각화와 관련된 다양한 방법에 대해서는 Michael Friendly의 홈페이지(http://datavis.ca)를 참조하기 바란다.

13.2 이변량 연속형 자료의 정리

연속형 변수가 두 개 혹은 그 이상 주어질 때 우리의 주된 관심은 변수들의 분포 비교와 변수 사이의 관계 탐색이라고 할 수 있다. 이 작업들은 그래프와 요약 통계를 함께 이용해야 효과적으로 이루어질 수 있다.

13.2.1 연속형 변수의 분포를 비교하기 위한 그래프

두 개 이상의 연속형 변수의 자료가 주어졌을 경우, 그 변수들의 분포를 비교하는 것은 가장 기본적인 작업이라고 할 수 있다. 만일 어느 한 연속형 변수의 분포가 어떤 요인으로 구분되는 그룹마다 다르다면, 그것은 곧 그룹변수가 연속형 변수에 미치는 영향이 통계적으로 유의하다고 해석될 수 있다.

일변량 연속형 변수의 분포를 나타내는 그래프라면 분포의 비교 목적으로도 사용될 수 있을 것이다. 그룹별로 분포를 비교하고자 할 때 먼저 시도할 수 있는 것은 facet일 것이다. 함수 facet_wrap() 혹은 facet_grid()를 이용하여 그룹별로 분포를 나타내는 그래프를 작성해서 비교하는 것을 생각해 볼 수 있다. 다른 방법으로는 그룹별 그래프를 겹치게 작성하는 것이다. 하나의 그래프에서 비교하는 것이므로 더 의미가 있는 비교가 될 수 있을 것이다.

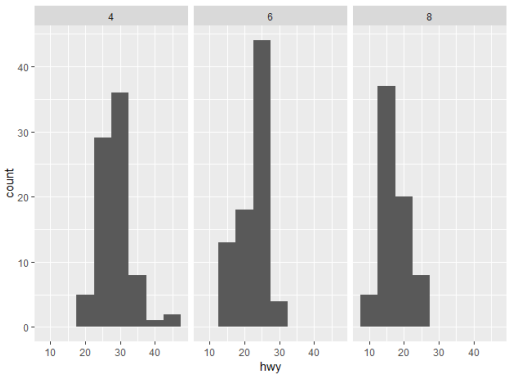
- 그룹 자료의 분포 비교를 위한 히스토그램

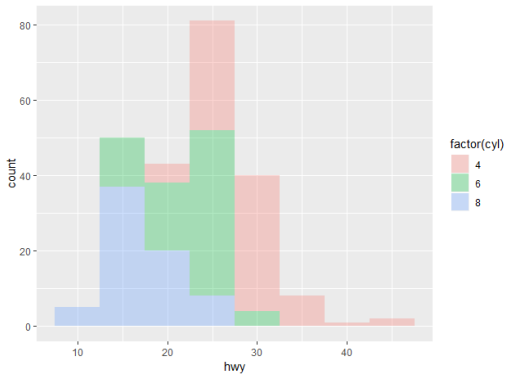
예제로 패키지 ggplot2에 있는 데이터 프레임 mpg의 변수 cyl에 따른 연속형 변수 hwy의 분포를 비교해 보자. 변수 cyl은 자동차 엔진의 실린더 개수를 의미하는 것으로 4, 5, 6, 8의 네 가지 값을 갖는 숫자형 변수이고, 변수 hwy는 자동차의 고속도로 연비를 나타내는 변수이다. 먼저 변수 cyl로 구분되는 그룹에 속한 자료의 개수를 구해보자.

```
> mpg %>% group_by(cyl) %>% summarize(n=n())
# A tibble: 4 x 2
    cyl     n
  <int> <int>
1     4    81
2     5     4
3     6    79
4     8    70
```

변수 cyl이 5가 되는 자료의 개수가 너무 작다는 것을 알 수 있다. 따라서 변수 cyl이 4, 6, 8인 그룹에 대해서만 변수 hwy의 분포를 비교해 보자.

```
> mpg_1 <- mpg %>% filter(cyl!=5)

> # 그림 13.10 (a)
> ggplot(mpg_1, aes(x=hwy)) +
    geom_histogram(binwidth=5) +
    facet_wrap(~cyl)

> # 그림 13.10 (b)
> ggplot(mpg_1, aes(x=hwy, fill=factor(cyl))) +
    geom_histogram(binwidth=5, alpha=0.3)
```





(a) (b)

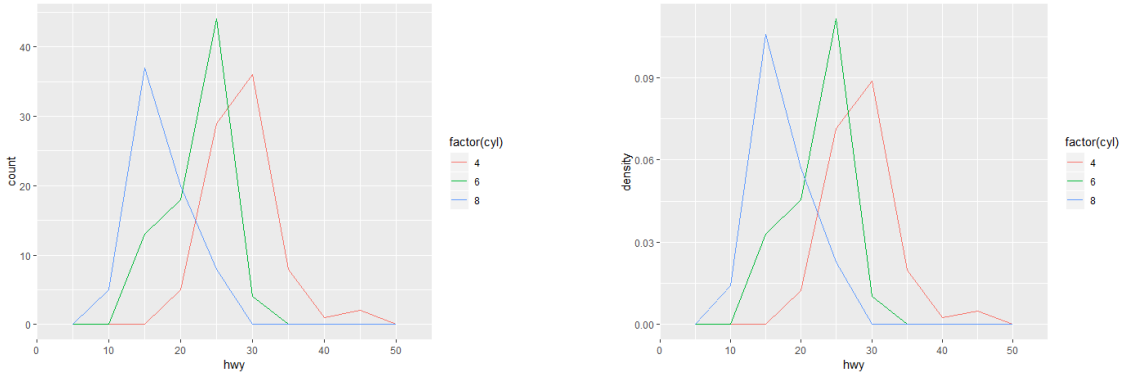
<그림 13.10> 그룹 자료의 분포 비교를 위한 히스토그램

겹쳐진 히스토그램을 작성하기 위해서는 시각적 요소 fill에 요인을 매핑해야 한다. 히스토그램으로는 그룹간 분포 비교가 쉽지 않음 보여주는 그래프라고 할 수 있다. 겹쳐 그리는 경우에는 투명도를 높이는 것이 큰 도움이 되지 않음을 알 수 있다.

- 그룹자료의 분포 비교를 위한 도수분포다각형

패키지 ggplot2의 데이터 프레임 mpg의 변수 cyl에 따른 hwy의 분포를 비교해 보자. 변수 cyl이 5인 자료를 제외한 mpg_1을 사용하자.

```
> p <- ggplot(mpg_1, aes(x=hwy, color=factor(cyl)))

> # 그림 13.11 (a)
> p + geom_freqpoly(binwidth=5)

> # 그림 13.11 (b)
> p + geom_freqpoly(aes(y=..density..), binwidth=5)
```



(a) (b)

<그림 13.11> 변수 cyl에 따른 hwy의 겹쳐 그린 도수분포다각형

함수 geom_freqpoly()에서 시각적 요소 color에는 요인을 연결해야 겹쳐진 도수분포다각형이 작성된다. 각 그룹에 속한 자료의 개수가 다르게 때문에 빈도수에 의한 도수분포다각형으로는 그룹간의 분포 비교가 부정확할 수 있다. 더 효과적인 비교를 하기 위해서는 시각적 요소 y에 변수 ..density..를 매핑하는 것이 바람직하다.

히스토그램과 도수분포다각형은 작성 과정에서 많은 부분을 공유하고 있는 그래프이며, 일변량 자료의 분포를 나타낼 때에는 모두 유용하게 사용되고 있다. 그러나 그룹간의 분포를 비교할 때는 히스토그램보다 도수분포다각형을 사용하는 것이 더 효과적인 비교가 가능하다고 본다.

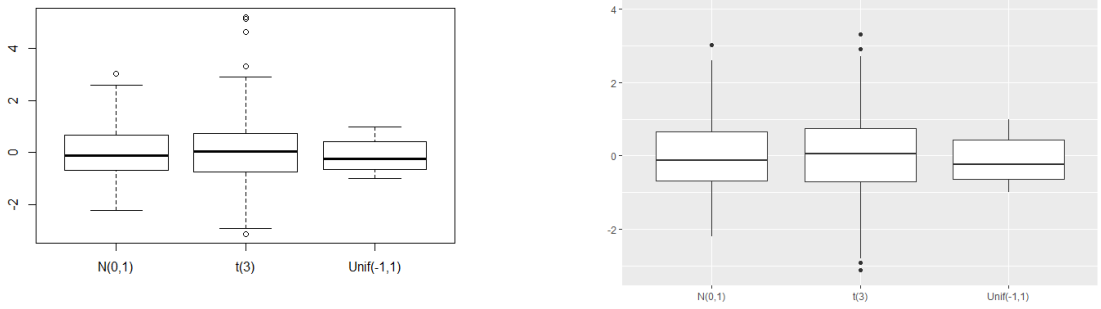
- 나란히 서 있는 상자그림

상자그림의 경우에는 facet을 이용하는 것보다 그룹별로 작성된 상자그림을 옆으로 나란히 세워

놓는 것이 더 효과적으로 분포를 비교하는 방법이 될 것이다. 자료의 형태는 비교 대상이 되는 변수가 각기 독립된 벡터에 입력되어 있는 경우와 하나의 연속형 벡터와 그룹을 구분하는 하나의 요인으로 입력되어 있는 경우로 구분할 수 있다.

먼저 비교할 대상이 되는 변수들이 x1, x2, x3 와 같이 개별적으로 구성된 경우를 살펴보자. 함수 boxplot()에서는 개별적으로 구성된 벡터들을 콤마로 구분해서 나열해 놓으면 된다. 패키지 ggplot2에서는 자료를 데이터 프레임으로 구성하되, 개별적으로 구성된 벡터들을 하나의 연속형 벡터와 하나의 요인으로 나타내야 한다. 예제로 정규분포, t 분포, 균등분포에서 각각 발생시킨 난수의 분포를 나란히 서 있는 상자그림으로 비교해보자.

```
> x1 <- rnorm(100)
> x2 <- rt(100, df=3)
> x3 <- runif(100, min=-1, max=1)

> # 그림 13.12 (a)
> boxplot(x1,x2,x3, names=c("N(0,1)", "t(3)", "Unif(-1,1)"))

> # 그림 13.12 (b)
> data.frame(x=c(rep(1:3,each=100)), y=c(x1,x2,x3)) %>%
    ggplot(aes(factor(x),y)) +
    geom_boxplot() +
    scale_x_discrete(labels=c("N(0,1)","t(3)","Unif(-1,1)")) +
    labs(x="", y="")
```



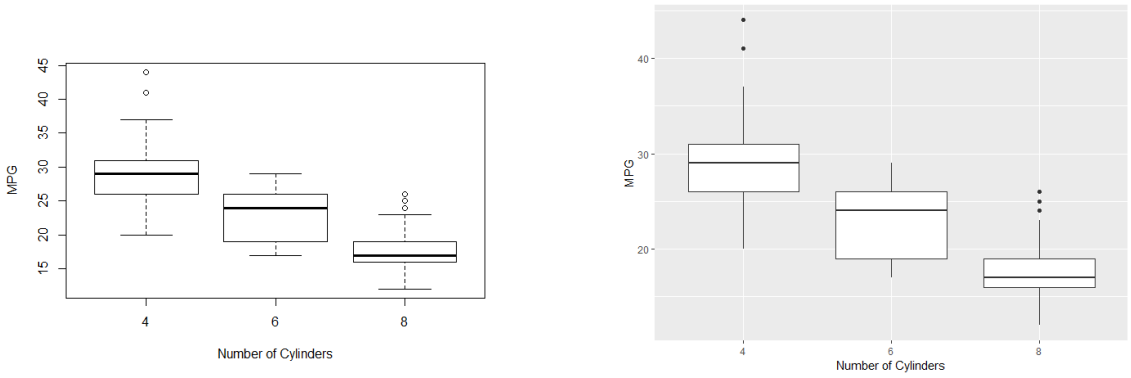
(a) (b)

<그림 13.12> 나란히 서 있는 상자그림

두 번째 경우로 하나의 연속형 변수가 하나 또는 그 이상의 요인에 의하여 몇 개의 그룹으로 구분되는 경우를 살펴보자. 함수 boxplot()의 경우에는 y~x의 형태로 x에 범주형 변수, y에 연속형 변수를 지정하면 되고, 함수 geom_boxplot()의 경우에 시각적 요소 x에는 요인을, y에는 연속형 변수를 매핑하면 된다. 예제로 패키지 ggplot2의 데이터 프레임 mpg의 변수 cyl에 따른 hwy의 분포를 비교해 보자. 변수 cyl이 5인 자료를 제외한 mpg_1을 사용하자.

```
> # 그림 13.13 (a)
> boxplot(hwy~cyl, data=mpg, subset=(cyl!=5),
          xlab="Number of Cylinders", ylab="MPG")
```

```
> # 그림 13.13 (b)
> ggplot(mpg_1, aes(x=factor(cyl), y=hwy)) +
    geom_boxplot() +
    labs(x="Number of Cylinders", y="MPG")
```



(a) (b)

<그림 13.13> 변수 cyl에 따른 mpg의 상자그림

함수 boxplot()에서 R 공식으로 변수를 지정한 경우에는 옵션 subset을 사용하여 자료의 일부분을 선택할 수 있음을 보여주기 위하여 데이터 프레임 mpg를 사용하였다. 함수 geom_boxplot()에서는 시각적 요소 x에 반드시 요인을 연결해야만 나란히 서 있는 상자그림을 작성할 수 있다.

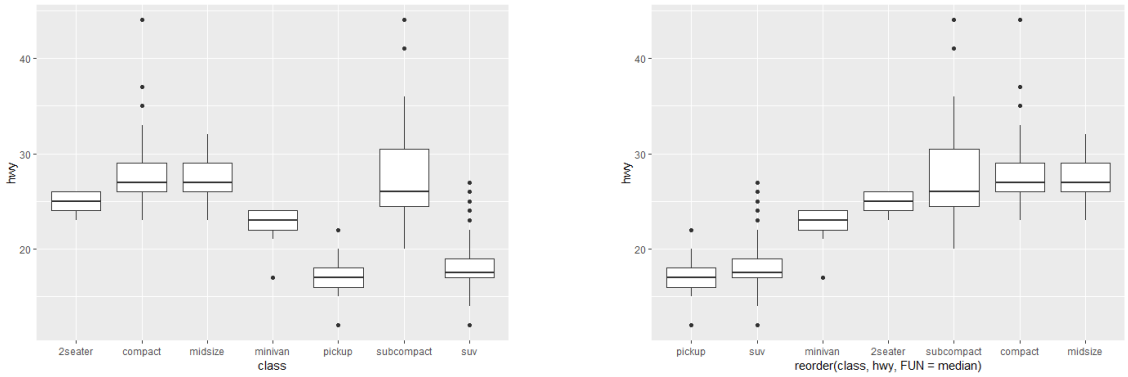
변수 cyl의 경우에는 실린더의 개수를 나타내는 4, 6, 8이라는 값이 나름의 순서가 있는 것이어서 그 순서대로 상자그림을 배치하는 것이 두 변수의 관계를 이해하는 데 도움이 된다. 그룹변수의 값들이 순서형 요인과 같이 자체적으로 순서가 있는 경우에는 그 순서대로 상자그림을 배치해도 괜찮겠지만, 명목형 요인과 같이 순서에 의미가 없는 경우에는 분석에 도움이 되도록 상자그림을 재배치하는 것이 더 좋을 것이다.

상자그림들을 재배치하려면 함수 reorder()를 사용하여 그룹변수로 사용되는 요인의 수준을 재정렬하면 된다. 먼저 데이터 프레임 mpg의 변수 hwy의 상자그림을 변수 class의 수준별로 작성해 보자.

```
> # 그림 13.14 (a)
> ggplot(mpg, aes(x=class, y=hwy)) +
    geom_boxplot()
```

상자그림이 변수 class의 알파벳 순서로 배치되어 있다. 이것을 변수 class의 각 그룹별 변수 hwy의 중앙값을 기준으로 재배치하면, 변수 class별로 hwy의 분포를 한 눈에 파악할 수 있게 된다. 함수 reorder()를 사용하여 요인 class의 수준을 변수 hwy의 중앙값을 기준으로 다시 배치하는 방법은 reorder(class, hwy, FUN=median)이 된다.

```
> # 그림 13.14 (b)
> ggplot(mpg,aes(x=reorder(class, hwy, FUN=median), y=hwy)) +
    geom_boxplot()
```



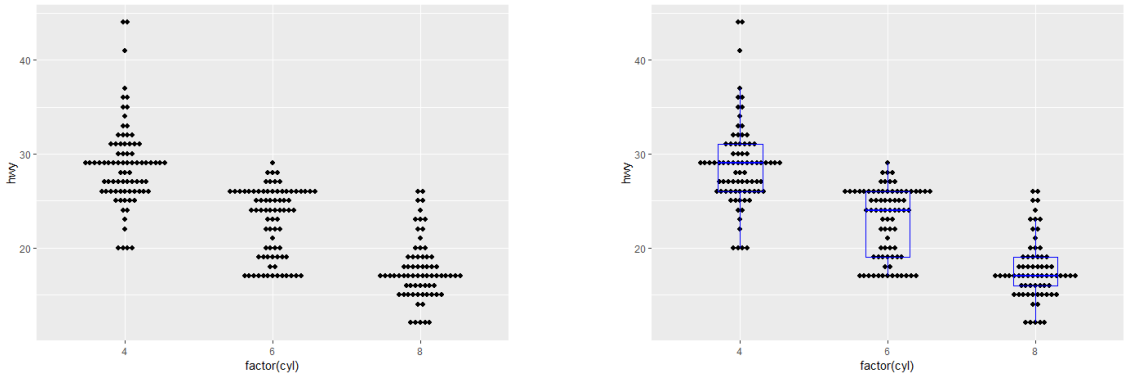
(a) (b)

<그림 13.14> 상자그림의 배치 순서 조정

- 그룹 자료에 대한 다중 점 그래프

다중 점 그래프도 그룹 자료의 분포를 비교하는 데 적합한 그래프라고 할 수 있다. 예제로 패키지 ggplot2의 데이터 프레임 mpg의 변수 cyl에 따른 hwy의 분포를 비교해 보자. 변수 cyl이 5인 자료를 제외한 mpg_1을 사용하자.

```
> mpg_1 <- mpg %>% filter(cyl!=5)

> # 그림 13.15 (a)
> ggplot(mpg_1, aes(x=factor(cyl), y=hwy)) +
    geom_dotplot(binaxis="y", binwidth=0.5, stackdir="center")
```



(a) (b)

<그림 13.15> 다중 점 그래프

수직 방향으로 점 그래프를 작성하는 것이므로 구간 설정은 Y축 변수를 대상으로 이루어져야 한다. 옵션 binaxis는 구간 설정 대상이 되는 축을 지정하는 것으로 디폴트는 "x"이다. 점을 쌓아 가는 방향을 결정하는 옵션은 stackdir이다. 가능한 값은 "up", "down", "center"와

“centerwhole”이 있다.

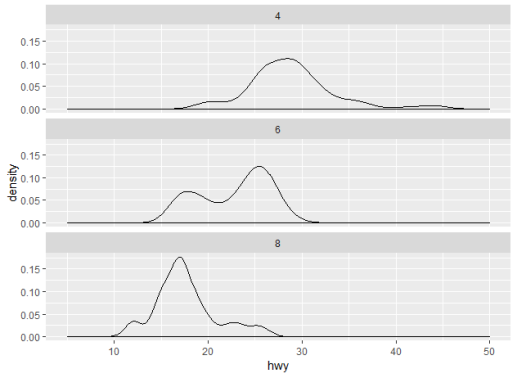
다중 점 그래프를 상자그림과 겹쳐서 작성하는 것도 의미가 있는 그래프가 된다. 이 경우에도 상자그림에 outlier.shape=NA를 포함시켜 이상값을 점으로 표시하지 않도록 해야 하며, 상자 내부에 색을 채우지 않도록 fill=NA를 지정할 필요가 있다. 또한 상자의 폭을 줄여 주는 것도 필요하다고 본다.

```
> # 그림 13.15 (b)
> ggplot(mpg_1, aes(x=factor(cyl), y=hwy)) +
    geom_dotplot(binaxis="y", binwidth=0.5, stackdir="center") +
    geom_boxplot(width=0.3, outlier.shape=NA, color="blue", fill=NA)
```

- 그룹 자료의 분포를 비교하기 위한 확률밀도함수 그래프

분포의 특징을 가장 세밀하게 표현할 수 있는 그래프가 확률밀도함수 그래프이다. Facet을 이용하여 그룹별 밀도함수를 비교하는 것도 의미가 있으며, 하나의 그래프에 겹치게 작성하는 것도 괜찮은 방법이 된다. 예제로 패키지 ggplot2의 데이터 프레임 mpg의 변수 cyl에 따른 hwy의 분포를 비교해 보자. 변수 cyl이 5인 자료를 제외한 mpg_1을 사용하자.

```
> mpg_1 <- mpg %>% filter(cyl!=5)

> # 그림 13.16
> ggplot(mpg_1, aes(x=hwy)) +
    geom_density() +
    xlim(5,50) +
    facet_wrap(~cyl, ncol=1)
```

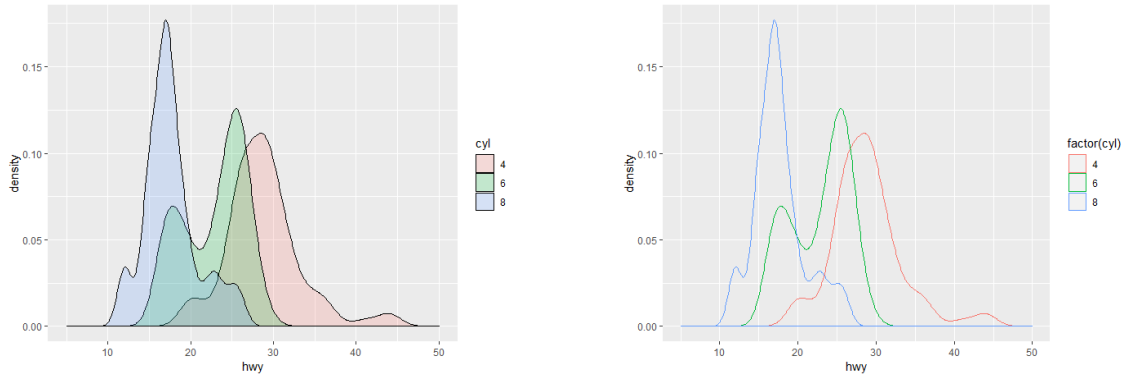


<그림 13.16> Facet에 의한 그룹별 확률밀도함수의 비교

하나의 그래프에 겹치게 작성하는 방법으로써 시각적 요소 fill을 사용하는 것과 시각적 요소 color를 사용하는 것을 생각해 볼 수 있다. 시각적 요소 fill을 사용하면, 확률밀도함수가 색으로 채워지기 때문에 서로 겹쳐져서 가려지는 부분이 있게 된다. 이 문제는 투명도를 높인 색을 사용하면 해결된다.

```
> # 그림 13.17 (a)
```

```
> ggplot(mpg_1, aes(x=hwy, fill=factor(cyl))) +
    geom_density(alpha=0.2) +
    xlim(5,50)
```



(a) (b)

<그림 13.17> 겹쳐 그린 확률밀도함수 그래프

시각적 요소 fill 대신 color를 사용하여 변수 factor(cyl)과 매핑하면 서로 다른 색의 선으로만 밀도함수가 그려지는 그래프가 작성된다.

```
> # 그림 13.17 (b)
> ggplot(mpg_1, aes(x=hwy, color=factor(cyl))) +
    geom_density() +
    xlim(5,50)
```

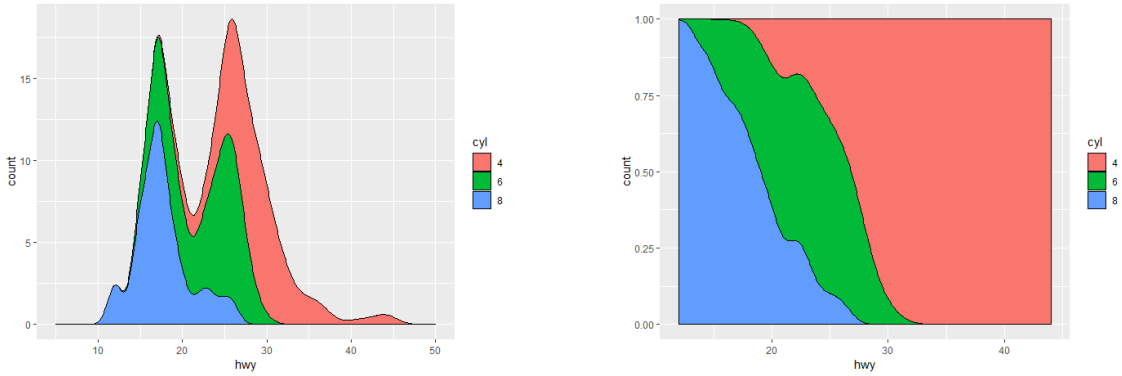
그룹별로 작성되는 여러 개의 확률밀도함수를 하나의 그래프에 나타내는 다른 방법으로 추정된 여러 개의 확률밀도함수를 위로 쌓아 올린 그래프와 x 변수의 주어진 값에 대한 각 그룹의 조건부 확률밀도함수를 그래프로 작성해 보자. 먼저 추정한 확률밀도함수를 위로 쌓아 올린 그래프를 작성해 보자. 이 경우에는 각 그룹마다 추정된 확률밀도에 자료의 수를 곱한 값을 사용하여 위로 쌓아 올리는 것이 각 그룹의 확률분포의 형태를 유지하며 통합하는 방법이 된다. 시각적 요소 y에 변수 ..count..를 연결하고, position에 "stack"을 지정하여 그래프를 작성해 보자.

```
> # 그림 13.18 (a)
> ggplot(mpg_1, aes(x=hwy, fill=factor(cyl))) +
    geom_density(position="stack", aes(y=..count..)) +
    xlim(5,50)
```

쌓아 올린 확률밀도함수 그래프와는 조금 다른 시각에서 분포를 비교할 수 있는 그래프로써 조건부 확률밀도함수 그래프를 작성해 보자. 이 그래프는 position에 "fill"을 지정하면 작성된다.

```
> # 그림 13.18 (b)
> ggplot(mpg_1, aes(x=hwy, fill=factor(cyl))) +
```

```
geom_density(position="fill", aes(y=..count..))
```



(a) (b)

<그림 13.18> 쌓아 올린 확률밀도함수 그래프와 조건부 확률밀도함수 그래프

- 평균 막대 그래프와 error bar 추가

여러 분포의 평균값을 비교하고자 할 때 단순히 숫자만을 보여주는 것보다는 그래프로 바꿔서 보여 주는 것이 훨씬 효과적으로 정보를 전달하는 방법이 될 것이다. 그런 면에서 막대 그래프는 그룹별 자료의 평균을 비교하는 그래프로 상당히 효과적으로 사용할 수 있다. Error bar는 분포의 변동을 그래프로 나타내는 기법이다. 따라서 그룹별 평균을 막대 그래프로 나타내고, 그 위에 error bar로 분포의 변동 혹은 신뢰구간을 함께 표시해 준다면 많은 정보를 매우 효과적으로 전달할 수 있을 것이다.

예제로 패키지 ggplot2의 데이터 프레임 mpg의 변수 cyl에 따른 hwy의 평균값을 막대 그래프로 나타내고, 그 위에 95% 신뢰구간을 error bar의 형태로 추가해 보자. 변수 cyl이 5인 자료는 분석에서 제외하기로 하자. 그래프 작성은 각 그룹 자료의 평균 및 신뢰구간이 이미 계산되어 있어서 그 결과를 이용하여 그래프를 작성하는 경우와 필요한 통계량 값을 직접 계산해서 그래프를 작성해야 하는 경우로 나누어 진행해 보자.

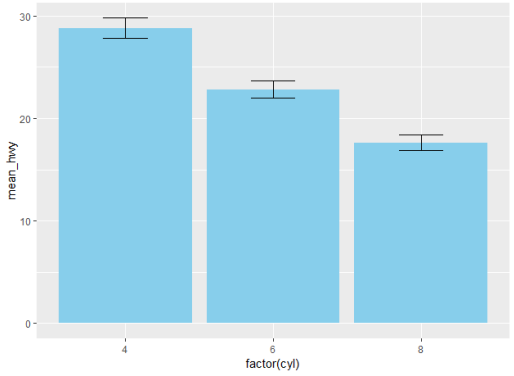
먼저 필요한 통계량 값인 변수 cyl로 구분되는 그룹별 hwy의 평균값과 95% 신뢰구간의 하한과 상한이 데이터 프레임으로 구성되어 있다고 하자.

```
> hwy_stat <- mpg %>%
    filter(cyl!=5) %>%
    group_by(cyl) %>%
    summarize(mean_hwy=mean(hwy), sd_hwy=sd(hwy),
              n_hwy=n(),
              ci_low=mean_hwy-qt(0.975,df=n_hwy-1)*sd_hwy/sqrt(n_hwy),
              ci_up=mean_hwy+qt(0.975,df=n_hwy-1)*sd_hwy/sqrt(n_hwy))

> hwy_stat
# A tibble: 3 x 6
    cyl mean_hwy sd_hwy n_hwy ci_low ci_up
  <int>    <dbl>  <dbl> <int>  <dbl> <dbl>
1     4     28.8   4.52     81   27.8  29.8
```

```
2     6     22.8    3.69     79   22.0   23.6
3     8     17.6    3.26     70   16.9   18.4
```

데이터 프레임 hwy_stat의 변수 mean_hwy로 막대 그래프를 작성하고, 변수 ci_low와 ci_up으로 error bar를 작성하면 된다. Error bar는 함수 geom_errorbar()로 작성되며, 시각적 요인 ymin에 변수 ci_low를, 시각적 요인 ymax에 변수 ci_up을 연결하면 된다.

```
> # 그림 13.19
> ggplot(hwy_stat, aes(x=factor(cyl), y=mean_hwy)) +
    geom_col(fill="skyblue") +
    geom_errorbar(aes(ymin=ci_low, ymax=ci_up), width=0.3)
```



<그림 13.19> 평균 막대 그래프와 error bar

이번에는 필요한 통계량을 직접 계산하고 그래프를 작성해야 하는 경우를 살펴보자. 이 경우 사용해야 할 함수는 stat_summary()이다. 그룹별 막대 그래프를 작성하기 위해서는 fun.y에 "mean"을, geom에는 "bar"를 각각 입력해야 하고, error bar를 작성하기 위해서는 fun.data에 "mean_cl_normal"을, geom에는 "errorbar"를 각각 입력해야 한다.

함수 mean_cl_normal()은 패키지 Hmisc의 함수 smean.cl.normal()와 동일한 것으로써 함수 stat_summary()에서 원활하게 사용될 수 있도록 불러주는 역할을 한다. 자료의 평균과 신뢰구간의 하한과 상한을 계산하는 함수이다.

```
> # 그림 13.19
> mpg %>% filter(cyl!=5) %>%
    ggplot(aes(x=factor(cyl), y=hwy)) +
    stat_summary(fun.y="mean", geom="bar", fill="skyblue") +
    stat_summary(fun.data="mean_cl_normal", geom="errorbar", width=0.3)
```

[유용한 정보]

1. R의 공식(formula) 객체

   변수들 사이의 관계를 설정하거나 수학적 모형을 설정하고자 하는 경우에 우리는 R에서

제공하는 공식(formula)를 사용할 수 있다. 공식의 예로는 y ~ x1 + x2 + x3를 들 수 있는데, 이것의 의미는 y는 종속변수 혹은 반응변수가 되고 x1, x2, x3는 독립변수 혹은 설명변수가 되는 것이며, 독립변수들은 서로 선형관계에 있음을 나타내고 있다. 위의 공식에서 사용된 '~'와 '+'의 두 가지 기호 외에도 공식을 설정하는데 사용되는 기호와 그 의미는 다음과 같다.

1) 물결표(~): 반응변수와 설명변수 사이의 관계를 설정한다.

2) 플러스(+): 변수들 사이의 선형관계를 표시한다.

3) 숫자 영(0): 절편이 없는 상태를 의미한다. 예: y ~ x1 + x2 + 0

4) 세로 줄(|): 조건 변수를 설정한다. 예: y ~ x1 | x2

5) 항등 함수(I( )): 항등 함수 안에 있는 표현은 수학 연산자로 해석된다. 예를 들어 y ~ x1 + x2는 y가 두 변수와 선형관계에 있음을 나타내지만, y ~ I(x1+x2)는 y가 x1과 x2를 더하여 생성된 새로운 하나의 변수와 선형관계에 있음을 나타낸다.

6) 별표(*): 두 변수 사이의 상호작용이 공식에 포함됨을 의미한다. 예를 들어 y ~ A*B는 y ~ A + B + I(A*B)를 의미한다.

7) 윗꺽쇠(^): 차수를 올리기 위해 사용되는 것으로, 예를 들어 y ~ (x1+x2)^2는 y ~ (x1+x2)*(x1+x2)를 의미한다.

13.2.2 연속형 변수의 분포를 비교하기 위한 요약통계

연속형 변수의 분포에 관련된 정보는 분포의 중심과 퍼짐 정도에 대한 요약통계량을 통해서 얻을 수 있다. 2개 이상의 연속형 변수에 대한 자료가 주어질 때 이러한 요약통계량의 값들을 일목요연하게 보여줄 수 있다면 연속형 변수들의 분포를 비교하는데 큰 도움이 될 것이다.

본 절에서는 연속형 변수의 기술통계량을 보기 좋게 정리하여 출력해주는 몇몇 함수들의 사용법을 살펴보겠다. 연속형 변수들의 병렬적 분포비교의 경우와 그룹별 분포비교의 경우에 각각 효과적으로 사용할 수 있는 함수들을 구분하여 소개하고자 한다.

1) 연속형 변수들의 병렬적 분포비교

여러 개의 변수들이 독립적으로 주어진 경우, 각 변수들의 기술통계량의 값을 일목요연하게 출력해주는 함수들이 표 13.2에 정리되어있다.

<표 13.2> 연속형 변수들의 병렬적 분포비교에 유용한 함수

함수	패키지
summary( )	base

describe( )	Hmisc
stat.desc( )	pastecs
describe( )	psych

예제를 통하여 표 13.2에 정리되어 있는 함수들의 사용법을 살펴보자. 예제로 사용할 자료는 데이터 프레임 mtcars로 변수 mpg와 disp, hp의 기술통계량을 계산해보자.

우선 함수 summary()는 이미 여러 번 사용된 함수로서 세 변수의 기술통계량이 다음과 같이 출력된다.

```
> mtcars_3 <- mtcars %>%
    select(mpg, disp, hp)

> summary(mtcars_3)
      mpg             disp             hp
 Min.   :10.40   Min.   : 71.1   Min.   : 52.0
 1st Qu.:15.43   1st Qu.:120.8   1st Qu.: 96.5
 Median :19.20   Median :196.3   Median :123.0
 Mean   :20.09   Mean   :230.7   Mean   :146.7
 3rd Qu.:22.80   3rd Qu.:326.0   3rd Qu.:180.0
 Max.   :33.90   Max.   :472.0   Max.   :335.0
```

다음으로 패키지 Hmisc에 있는 함수 describe()의 사용법 및 결과 출력 방식을 살펴보자. 패키지 Hmisc는 자료 분석 및 그래픽스 등의 분야에서 매우 유용한 함수들이 있는 패키지이며 함수 describe()에 데이터 프레임이 입력되면 변수의 개수, 관찰값의 개수와 각 변수의 평균값 및 분위수 등이 출력된다.

```
> Hmisc::describe(mtcars_3)
mtcars_3

 3  Variables      32  Observations
---------------------------------------------------------------------------
mpg
       n  missing distinct     Info     Mean      Gmd      .05
      32        0       25    0.999    20.09    6.796    12.00
     .10      .25      .50      .75      .90      .95
   14.34    15.43    19.20    22.80    30.09    31.30

lowest : 10.4 13.3 14.3 14.7 15.0, highest: 26.0 27.3 30.4 32.4 33.9
---------------------------------------------------------------------------
disp
       n  missing distinct     Info     Mean      Gmd      .05
      32        0       27    0.999    230.7    142.5    77.35
     .10      .25      .50      .75      .90      .95
   80.61   120.83   196.30   326.00   396.00   449.00

lowest :  71.1  75.7  78.7  79.0  95.1, highest: 360.0 400.0 440.0 460.0
472.0
---------------------------------------------------------------------------
hp
```

```
      n  missing distinct     Info     Mean      Gmd      .05
     32        0       22    0.997    146.7    77.04    63.65
    .10      .25      .50      .75      .90      .95
  66.00    96.50   123.00   180.00   243.50   253.55

lowest :  52  62  65  66  91, highest: 215 230 245 264 335
------------------------------------------------------------------------
```

패키지 pastecs에 있는 함수 stat.desc()도 다양한 기술통계량을 계산하여 출력해주는 기능을 갖고 있다. 기본적인 사용법은 stat.desc(x, basic = TRUE, desc = TRUE, norm = FALSE, p = 0.95)가 되는데, x는 데이터 프레임이고, 옵션 basic=TRUE이면 자료의 개수, 최소값 등의 기초 통계량의 값이 출력되고, 옵션 desc=TRUE이면 중앙값, 평균, 표준편차, 모평균에 대한 신뢰구간 등이 출력된다. 또한 옵션 norm이 TRUE가 되면 정규분포와 관련된 통계량이 출력되는데, 여기에는 왜도, 첨도 그리고 정규성 검정 결과 등이 출력된다. 옵션 p는 신뢰구간의 신뢰수준을 지정하는 역할을 하고 있다.

```
> pastecs::stat.desc(mtcars_3)
                  mpg    disp       hp
nbr.val        32.0   3.2e+01   32.00
nbr.null        0.0   0.0e+00    0.00
nbr.na          0.0   0.0e+00    0.00
min            10.4   7.1e+01   52.00
max            33.9   4.7e+02  335.00
range          23.5   4.0e+02  283.00
sum           642.9   7.4e+03 4694.00
median         19.2   2.0e+02  123.00
mean           20.1   2.3e+02  146.69
SE.mean         1.1   2.2e+01   12.12
CI.mean.0.95    2.2   4.5e+01   24.72
var            36.3   1.5e+04 4700.87
std.dev         6.0   1.2e+02   68.56
coef.var        0.3   5.4e-01    0.47
```

위 결과물에서 모평균의 95% 신뢰구간은 mean에 CI.mean.0.95을 빼고 더하는 것으로 계산되는데, 예컨대 변수 mpg의 경우에 95% 신뢰구간은 20.1 ± 2.2로 계산된다.

마지막으로 패키지 psych에 있는 함수 describe()의 사용법을 살펴보자. 기본적인 사용법은 describe(x, na.rm = TRUE, interp = FALSE, skew = TRUE, ranges = TRUE, trim = 0.1)가 되는데, x는 데이터 프레임이고, 옵션 na.rm은 결측값의 제거여부, interp는 중앙값의 계산방식, skew는 왜도와 첨도의 출력여부, range는 범위의 출력여부를 각각 묻는 것이고 trim은 절삭평균(trimmed mean)을 계산할 때 제거되는 자료의 비율을 정하게 된다.

```
> psych::describe(mtcars_3)
     vars  n    mean     sd median trimmed    mad    min   max  range skew
mpg     1 32   20.09   6.03   19.2    19.70    5.41 10.4  33.9   23.5 0.61
disp    2 32 230.72 123.94  196.3  222.52 140.48  71.1 472.0 400.9 0.38
hp      3 32  146.69  68.56  123.0  141.19   77.10 52.0 335.0 283.0 0.73
     kurtosis    se
mpg     -0.37  1.07
```

```
disp    -1.21 21.91
hp      -0.14 12.12
```

2) 연속형 변수의 그룹별 분포비교

연속형 변수의 그룹별 분포비교에 유용하게 사용되는 함수들이 표 13.3에 정리되어 있다.

<표 13.3> 연속형 변수의 그룹별 분포비교에 유용한 함수

함수	패키지
aggregate( )	stats
by( )	base
summaryBy( )	doBy
describe.by( )	psych

예제를 통하여 표 13.3에 정리되어 있는 함수들의 사용법을 살펴보자. 데이터 프레임 mtcars의 변수 mpg와 disp, hp의 기술통계량을 변수 cyl에 의해서 구분되는 세 그룹별로 계산해보자

함수 aggregate()의 사용법은 3.3.6절에서 이미 살펴보았다. 기본적인 사용법은 aggregate(x, by, FUN, ...)이다.

```
> mtcars_3 <- mtcars %>%
    select(mpg, disp, hp)

> aggregate(mtcars_3, by=list(cyl=mtcars$cyl), mean)
  cyl  mpg disp   hp
1   4 26.7  105  82.6
2   6 19.7  183 122.3
3   8 15.1  353 209.2

> aggregate(mtcars_3, by=list(cyl=mtcars$cyl), sd)
  cyl  mpg disp   hp
1   4 4.51 26.9 20.9
2   6 1.45 41.6 24.3
3   8 2.56 67.8 51.0
```

각 그룹별로 함수 mean()과 sd()를 따로 적용했는데, 만일 적용하고자 하는 함수가 하나 이상인 경우에는 사용자가 함수를 정의해서 사용해야 한다.

```
> my_stat <- function(x) c(Mean=mean(x), SD=sd(x))
> aggregate(mtcars_3, by=list(cyl=mtcars$cyl), my_stat)
  cyl mpg.Mean mpg.SD disp.Mean disp.SD hp.Mean hp.SD
1   4    26.66   4.51     105.1    26.9    82.6  20.9
2   6    19.74   1.45     183.3    41.6   122.3  24.3
3   8    15.10   2.56     353.1    67.8   209.2  51.0
```

다음으로 함수 by()의 사용법을 살펴보자. 이 함수의 기본적인 사용법은 by(data, INDICES,

FUN, ...)이며, data는 데이터 프레임, INDICES는 그룹을 구분하는 범주형 변수이며, FUN은 그룹별로 구분된 데이터 프레임에 적용할 함수가 된다. 또한 옵션 ... 에는 FUN에 적용되는 추가 옵션을 지정할 수 있다. 여기서 주의할 점은 FUN에는 데이터 프레임에 적용되는 함수를 지정해야 한다는 것이다. 따라서 평균을 계산하고자 한다면 함수 mean이 아니라 함수 colMeans를 사용해야 한다.

```
> by(mtcars_3, mtcars$cyl, colMeans)
mtcars$cyl: 4
  mpg   disp    hp
 26.7 105.1  82.6
------------------------------------------------------
mtcars$cyl: 6
  mpg   disp    hp
 19.7 183.3 122.3
------------------------------------------------------
mtcars$cyl: 8
  mpg   disp    hp
 15.1 353.1 209.2
```

표준편차의 경우에는 함수 sapply()를 사용하여 사용자가 함수를 정의해야 한다.

```
> by(mtcars_3, mtcars$cyl, function(x) sapply(x, sd))
mtcars$cyl: 4
  mpg   disp    hp
 4.51 26.87 20.93
------------------------------------------------------
mtcars$cyl: 6
  mpg   disp    hp
 1.45 41.56 24.26
------------------------------------------------------
mtcars$cyl: 8
  mpg   disp    hp
 2.56 67.77 50.98
```

한 개 이상의 기술통계량을 계산하고자 한다면 원하는 통계량을 계산하도록 하는 사용자가 함수를 미리 정의한 후 함수 sapply()에서 사용해야 한다.

```
> my_stat <- function(x) c(Mean=mean(x), SD=sd(x))

> by(mtcars_3, mtcars$cyl, function(x) sapply(x, my_stat))
mtcars$cyl: 4
       mpg   disp    hp
Mean 26.66 105.1  82.6
SD    4.51   26.9 20.9
------------------------------------------------------
mtcars$cyl: 6
        mpg  disp    hp
Mean 19.74 183.3 122.3
SD    1.45   41.6  24.3
------------------------------------------------------
mtcars$cyl: 8
        mpg  disp  hp
```

```
Mean 15.10 353.1 209
SD    2.56  67.8   51
```

다음으로 패키지 doBy에 있는 함수 summaryBy()의 사용법을 살펴보자. 기본적인 사용법은 summaryBy(formula, data, FUN)이 되는데, formula의 형식은 var1 + . . . + varN ~ groupvar1 + . . . + groupvarN이 되고, data는 데이터 프레임, FUN에는 사용할 함수를 나열하면 된다.

```
> doBy::summaryBy(mpg+disp+hp~cyl, data=mtcars, FUN=c(mean, sd))
  cyl mpg.mean disp.mean hp.mean mpg.sd disp.sd hp.sd
1   4     26.7       105    82.6   4.51   26.9   20.9
2   6     19.7       183   122.3   1.45   41.6   24.3
3   8     15.1       353   209.2   2.56   67.8   51.0
```

마지막으로 패키지 psych에는 있는 함수 describeBy()의 사용법을 살펴보자. 기본적인 사용법은 describeBy(x, group=NULL, mat=FALSE, ...)이 되는데, x는 데이터 프레임, group은 그룹을 구분하는 범주형 변수가 되며, 옵션 mat는 결과를 리스트 형태로 출력할 것인지(디폴트), 아니면 행렬의 형태로 출력할 것인지(TRUE)를 정하게 된다. 또한 옵션 ...는 패키지 psych에 있는 함수 describe()에서 사용되는 옵션을 추가적으로 사용할 수 있음을 의미한다. 함수 describeBy()의 경우에는 계산되는 요약 통계량의 종류를 사용자가 임의로 선택할 수 없게 되어 있다.

```
> psych::describeBy(mtcars_3, mtcars$cyl)

 Descriptive statistics by group
group: 4
     vars  n   mean    sd median trimmed   mad  min    max range skew
mpg     1 11   26.7  4.51     26    26.4  6.52 21.4   33.9  12.5 0.26
disp    2 11 105.1 26.87    108   104.3 43.00 71.1 146.7  75.6 0.12
hp      3 11   82.6 20.93     91    82.7 32.62 52.0 113.0  61.0 0.01
     kurtosis   se
mpg     -1.65 1.36
disp    -1.64 8.10
hp      -1.71 6.31
------------------------------------------------------------
group: 6
     vars n   mean    sd median trimmed   mad   min   max  range  skew
mpg     1 7   19.7  1.45   19.7    19.7   1.93  17.8  21.4    3.6 -0.16
disp    2 7 183.3 41.56  167.6   183.3 11.27 145.0 258.0 113.0  0.80
hp      3 7 122.3 24.26  110.0   122.3  7.41 105.0 175.0  70.0  1.36
     kurtosis    se
mpg     -1.91  0.55
disp    -1.23 15.71
hp       0.25  9.17
------------------------------------------------------------
group: 8
     vars  n  mean    sd median trimmed    mad   min   max  range  skew
mpg     1 14  15.1  2.56   15.2    15.2   1.56  10.4  19.2    8.8 -0.36
disp    2 14 353.1 67.77  350.5   349.6 73.39 275.8 472.0 196.2  0.45
hp      3 14  209.2 50.98  192.5   203.7 44.48 150.0 335.0 185.0  0.91
```

```
     kurtosis    se
mpg     -0.57  0.68
disp    -1.26 18.11
hp       0.09 13.62
```

다른 예제 자료를 이용하여 표 13.3에 정리되어 있는 함수들의 활용도를 비교해보자. 사용할 예제 자료는 데이터 프레임 airquality로 변수 Ozone과 Solar.R의 평균, 표준편차, 그리고 결측값을 포함한 총 표본크기와 결측값을 제외한 표본크기를 월별로 출력해보자. 원하는 통계량의 값을 계산하기 위한 함수는 다음과 같이 정의할 수 있다.

```
> my_stat2 <- function(x, ...) {
      c(mean=mean(x, ...), sd=sd(x, ...),
        total_n=length(x), n=sum(!is.na(x)))
    }
```

표 13.3에 있는 함수 중 사용자가 정의한 함수를 사용할 수 있는 함수 aggregate()와 by(), 그리고 summaryBy( )에 함수 my_stat2를 적용시켜 결과를 출력해보자.

우선 함수 aggregate()에 의한 결과는 다음과 같다.

```
> aggregate(airquality[c("Ozone","Solar.R")],
            by=list(month=airquality$Month), my_stat2, na.rm=TRUE)
  month Ozone.mean Ozone.sd Ozone.total_n Ozone.n Solar.R.mean
1     5        23.6     22.2          31.0    26.0        181.3
2     6        29.4     18.2          30.0     9.0        190.2
3     7        59.1     31.6          31.0    26.0        216.5
4     8        60.0     39.7          31.0    26.0        171.9
5     9        31.4     24.1          30.0    29.0        167.4
  Solar.R.sd Solar.R.total_n Solar.R.n
1      115.1            31.0       27.0
2       92.9            30.0       30.0
3       80.6            31.0       31.0
4       76.8            31.0       28.0
5       79.1            30.0       30.0
```

다음으로 함수 summaryBy()에 의한 결과는 다음과 같다.

```
> doBy::summaryBy(Ozone+Solar.R~Month, data=airquality,
                  FUN=my_stat2, na.rm=TRUE)
  Month Ozone.mean Ozone.sd Ozone.total_n Ozone.n Solar.R.mean
1     5        23.6     22.2            31       26          181
2     6        29.4     18.2            30        9          190
3     7        59.1     31.6            31       26          216
4     8        60.0     39.7            31       26          172
5     9        31.4     24.1            30       29          167
  Solar.R.sd Solar.R.total_n Solar.R.n
1      115.1              31       27
2       92.9              30       30
3       80.6              31       31
4       76.8              31       28
5       79.1              30       30
```

출력결과로만 보면 함수 aggregate()와 summaryBy()는 거의 동일하다고 볼 수 있으나, 사용하는 측면에서는 summaryBy()가 조금 더 편리하다고 할 수 있다.

다음으로 함수 by()에 의한 결과를 살펴보자.

```
> by(airquality[c("Ozone","Solar.R")], airquality$Month,
      function(x) sapply(x, my_stat2, na.rm=TRUE) )
airquality$Month: 5
        Ozone Solar.R
mean     23.6     181
sd       22.2     115
total_n  31.0      31
n         26.0      27
------------------------------------------------------
airquality$Month: 6
        Ozone Solar.R
mean     29.4   190.2
sd       18.2    92.9
total_n  30.0    30.0
n         9.0    30.0
------------------------------------------------------
airquality$Month: 7
        Ozone Solar.R
mean     59.1   216.5
sd       31.6    80.6
total_n  31.0    31.0
n         26.0    31.0
------------------------------------------------------
airquality$Month: 8
        Ozone Solar.R
mean     60.0   171.9
sd        39.7    76.8
total_n  31.0    31.0
n         26.0    28.0
------------------------------------------------------
airquality$Month: 9
        Ozone Solar.R
mean     31.4   167.4
sd       24.1    79.1
total_n  30.0    30.0
n         29.0    30.0
```

함수 by()의 경우에는 비교되는 변수가 많을 때 조금 더 선호되는 방식으로 결과물이 출력됨을 알 수 있다. 각각의 방법이 약간의 장단점이 있기 때문에 상황에 따라 가장 효과적인 방법을 선택해서 원하는 결과를 얻으면 될 것으로 보인다.

13.2.3 연속형 변수의 관계 탐색을 위한 그래프

두 연속형 변수의 관계를 탐색할 때 가장 많이 사용되는 그래프는 산점도일 것이다. 산점도를

확인해보면 두 변수의 관계를 시각적으로 파악할 수 있는데, 산점도에 두 변수의 회귀직선이나 비모수 회귀곡선 등을 추가하면 관계 파악에 도움이 된다. 대규모 자료의 산점도를 작성하면 점들이 서로 겹쳐서 관계 탐색이 어려울 때가 있다. 이런 경우 대안으로 X축 자료를 구간으로 구분하여 Y축 자료의 상자그림을 작성할 수 있다. 두 변수의 결합확률밀도함수를 추정하는 것도 두 변수 관계 파악에 유용하게 사용될 수 있다. 산점도 행렬은 많은 연속형 변수들의 관계를 파악하는데 도움이 많이 되는 그래프이다.

1) 다양한 유형의 산점도 작성

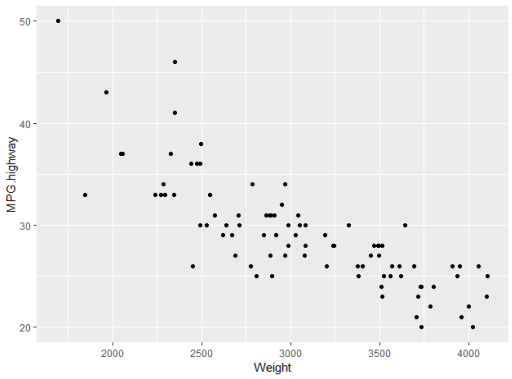
- 기본적인 형태의 산점도

산점도 작성의 예제 자료로 패키지 MASS의 데이터 프레임 Cars93을 사용해 보자. 먼저 변수 Weight와 MPG.highway의 산점도를 작성해 보자.

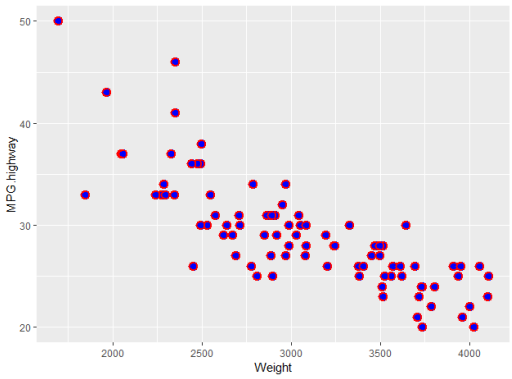
```
> data(Cars93, package="MASS")

> # 그림 13.20 (a)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point()
```

가장 기본적인 형태의 산점도에 어렵지 않게 변화를 줄 수 있다. 점의 모양(shape=21)을 내부와 외곽선의 색을 다르게 줄 수 있는 형태로 바꾸고, 크기도 늘리고(size=3), 점의 내부 색은 파란색(fill="blue"), 점의 외곽선 색은 빨간색(color="red"), 그리고 점의 외곽선 두께를 늘려서(stroke=1.5) 산점도를 작성해 보자. 이 경우 시각적 요소 shape, size, fill, color 등에 특정 변수를 매핑하는 것이 아니고 사용자가 값을 지정하는 것이므로 함수 aes() 밖에서 시각적 요소에 값을 지정해야 한다.

```
> # 그림 13.20 (b)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point(shape=21, color="red", fill="blue",
               stroke=1.5, size=3)
```



(a)



(b)

<그림 13.20> 기본적인 형태의 산점도

- 시각적 요소에 세 번째 변수 매핑

산점도는 두 변수 사이의 관계를 나타내는 그래프이지만, 세 변수 사이의 관계를 나타내는 것도 가능하다. 방법은 두 변수는 시각적 요소 x와 y에 매핑을 하고, 세 번째 변수는 점의 색이나 모양 등에 매핑을 하는 것이다. 시각적 요소인 점의 색(color)이나 모양(shape) 등을 함수 aes() 안에서 변수와 연결을 하면, 변수에 따라 점의 색이나 모양 등이 구분되어 산점도가 작성된다. 이러한 산점도를 작성하면 세 번째 변수가 두 변수의 관계에 미치는 영향력을 파악할 수 있다. 이 경우 color나 shape와 같은 시각적 요소에 매핑을 할 변수가 요인인 경우와 여러 개의 다른 값을 갖는 숫자형 변수인 경우의 차이점을 살펴보는 것도 중요하다. 변수 Weight와 MPG.highway의 산점도를 작성하되 점 색 및 점의 모양을 요인 Origin에 따라 구분이 되도록 작성해 보자.

```
> # 그림 13.21 (a)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway, color=Origin)) +
    geom_point()

> # 그림 13.21 (b)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway, shape=Origin,
                     color=Origin)) +
    geom_point()
```

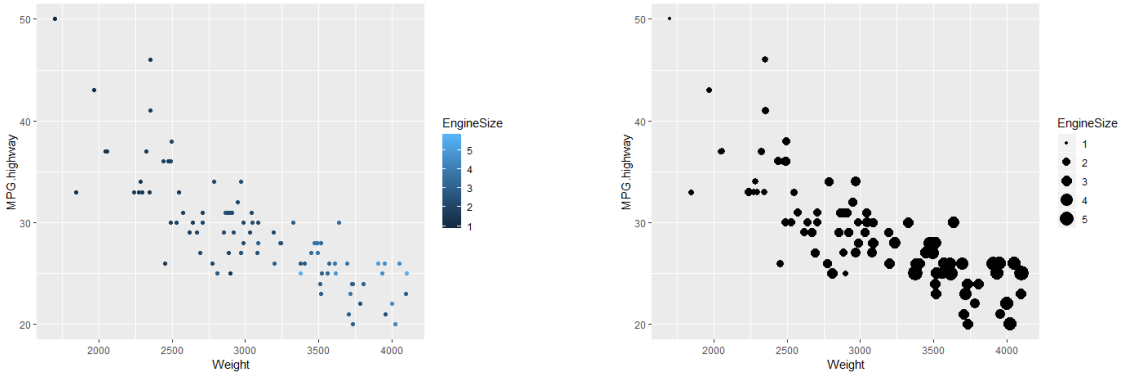
(a) (b)

<그림 13.21> 요인에 따라 점의 색 및 모양을 구분되게 작성한 산점도

이번에는 변수 Weight와 MPG.highway의 산점도를 작성하되 점의 색 및 크기를 숫자형 변수 EngineSize의 값에 따라 구분이 되도록 작성해 보자.

```
> # 그림 13.22 (a)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway, color=EngineSize)) +
    geom_point()

> # 그림 13.22 (b)
```

```
> ggplot(Cars93, aes(x=Weight, y=MPG.highway, size=EngineSize)) +
    geom_point()
```



(a) (b)

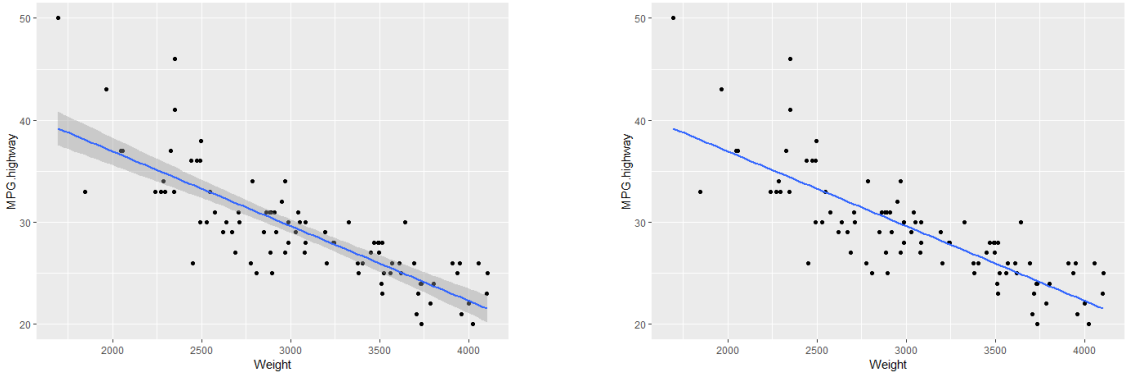
<그림 13.22> 숫자형 변수를 color나 size에 매핑한 산점도

시각적 요소 color나 size에 숫자형 변수의 매핑으로 작성된 그래프에서 세 번째 변수의 영향력은 정확하게 파악되지 않는 것으로 보인다. 점의 색이나 크기에서의 미세한 차이가 정확하게 구분이 되지 않기 때문이다.

- 산점도에 회귀직선 추가

산점도만으로 두 변수의 관계를 파악하는 것보다 회귀직선이나 비모수 회귀곡선 등을 추가하는 것이 훨씬 더 효과적일 수 있다. 회귀직선의 추가는 함수 geom_smooth()에 선형회귀모형을 의미하는 method="lm"을 지정하면 된다. 추정된 회귀직선에 신뢰구간이 함께 표현되는 것이 디폴트인데, 회귀직선만 나타내려면 se=FALSE를 입력해야 한다.

```
> # 그림 13.23 (a)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point() +
    geom_smooth(method="lm")

> # 그림 13.23 (b)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point() +
    geom_smooth(method="lm", se=FALSE)
```



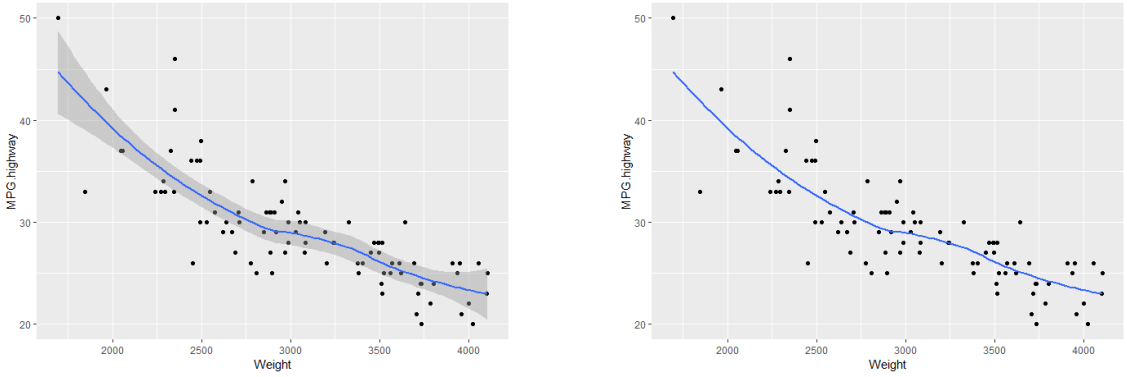
(a) (b)

<그림 13.23> 회귀직선을 산점도에 함께 표시

두 변수의 관계를 나타내는 회귀모형으로 가장 많이 사용되는 것이 loess(local regression)이라고 불리는 국소다항회귀모형이다. 비모수 회귀모형으로 분류되고 있으며, 자료의 개수가 너무 크지 않는 경우에 가장 무난하게 사용할 수 있는 분석 도구이다. 함수 geom_smooth()에서 디폴트로 적용되는 회귀모형이며, 신뢰구간을 포함하여 표시되는 것이 디폴트이다.

```
> # 그림 13.24 (a)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point() +
    geom_smooth()
`geom_smooth()` using method = 'loess' and formula 'y ~ x'

> # 그림 13.24 (b)
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point() +
    geom_smooth(se=FALSE)
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

선형회귀모형은 두 변수 X와 Y의 관계를 $Y = \beta_0 + \beta_1 X + \varepsilon$과 같이 미리 설정하고, 자료를 근거로 회귀함수의 계수를 추정하여 X 변수의 전체 범위에서 두 변수의 관계를 수식으로 표현하는 방식이다. 함수 lm()으로 추정된다. 이에 반하여 국소다항회귀모형은 두 변수의 관계를 미리 설정하지 않으며, X 변수의 개별 값에 대하여 회귀함수의 값을 각각 추정하는 방식이다. 회귀함수의 값을 추정하는 방법은 X 변수의 개별 값마다 가까운 자료 값만을 대상으로 가중 다항회귀모형을 적합하는 것이다. 함수 loess()로 추정된다.

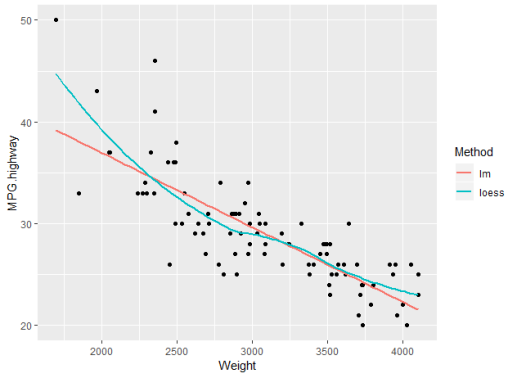


(a) (b)

<그림 13.24> 국소회귀곡선을 산점도에 함께 표시

이번에는 선형회귀직선과 국소회귀곡선을 산점도에 함께 나타내 보자. 두 종류의 선을 다른 색으로 표시하고 범례를 추가해야 하는데, 패키지 graphics의 함수 legend()와 같은 기능을 갖고 있는 함수가 패키지 ggplot2에는 없다. 대신 시각적 요소에 문자열을 매핑하는 방법이 유효하게 사용된다.

```
> # 그림 13.25
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point() +
    geom_smooth(aes(color="lm"), method="lm", se=FALSE) +
    geom_smooth(aes(color="loess"), se=FALSE) +
    labs(color="Method")
```



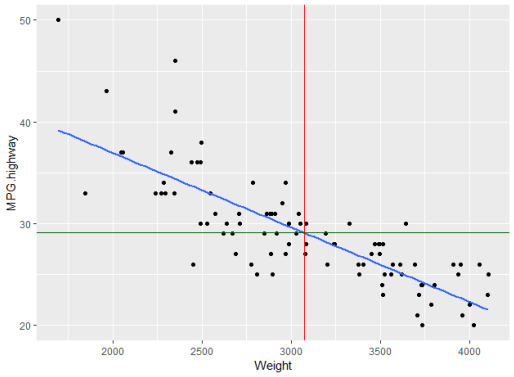
<그림 13.25> 선형회귀직선과 국소회귀곡선을 함께 표시

- 산점도에 수평선, 수직선 추가

산점도에는 회귀직선뿐만 아니라 수평선과 수직선도 추가할 수 있다. 수평선의 추가는 함수 geom_hline(yintercept)으로, 수직선의 추가는 함수 geom_vline(xintercept)으로, 직선의 추가는 함수 geom_abline(slope, intercept)으로 할 수 있다. 변수 Weight와 MPG.highway의 산점도에 두 변수의 선형회귀직선을 추가하고, 두 변수의 평균값에 해당하는

위치에 수직선과 수평선을 각각 추가해 보자.

```
> # 그림 13.26
> ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point() +
    geom_smooth(method="lm", se=FALSE) +
    geom_vline(aes(xintercept=mean(Weight)), color="red") +
    geom_hline(aes(yintercept=mean(MPG.highway)), color="darkgreen" )
```

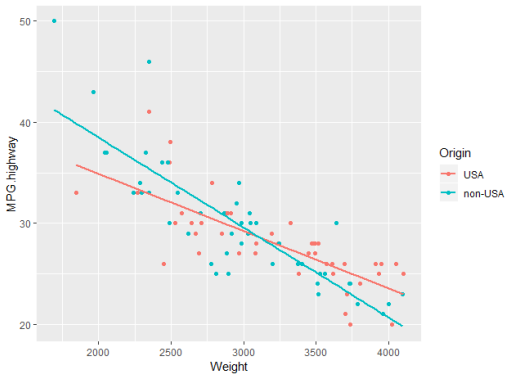


<그림 13.26> 산점도에 수직선과 수평선 추가

- 산점도에 그룹별 회귀직선 추가

산점도의 점이 그룹변수의 값에 따라 색이나 모양이 구분되어 있는 경우, 각 그룹별 자료만을 대상으로 추정된 회귀직선을 추가해 보자. 변수 Weight와 MPG.highway의 산점도를 요인 Origin에 따라 점의 색을 구분하고, 각 그룹별 회귀직선을 추정하여 추가해 보자.

```
> # 그림 13.27
> ggplot(Cars93, aes(x=Weight, y=MPG.highway, color=Origin)) +
    geom_point() +
    geom_smooth(method="lm", se=FALSE)
```



<그림 13.27> 그룹별 회귀직선의 추가

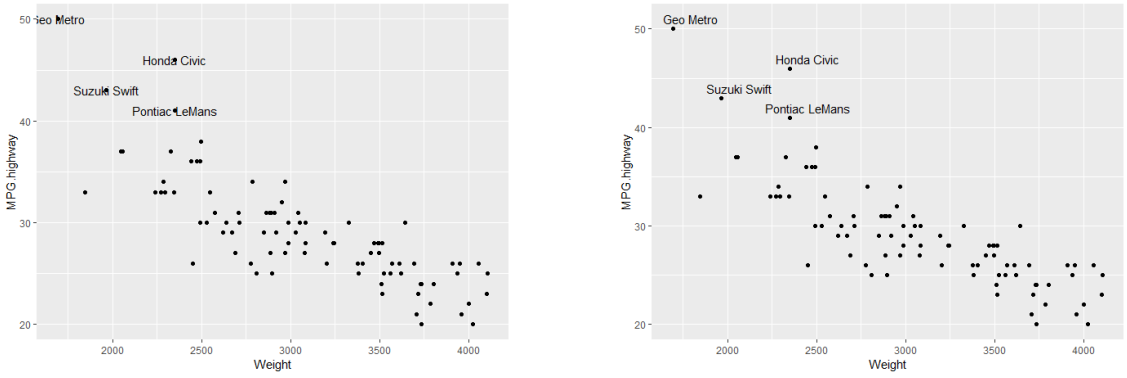
● 산점도의 점에 라벨 추가

산점도에서 특정 조건을 만족하는 점에 라벨을 붙이게 되면 자료의 특성을 파악하는데 중요한 그래프가 될 수 있다. 산점도의 점에 라벨을 붙이는 작업은 함수 geom_text()로 할 수 있는데, 시각적 요소 label에 라벨의 내용이 되는 문자형 변수를 연결하면 된다. 변수 Weight와 MPG.highway의 산점도에서 변수 MPG.highway가 40을 초과하는 점에 라벨을 붙여 보자. 라벨의 내용은 변수 Manufacturer와 Model을 연결한 것으로 하자.

```
> p <- ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point()

> # 그림 13.28 (a)
> p + geom_text(data=filter(Cars93, MPG.highway>40),
              aes(label=paste(Manufacturer, Model)))
```

추가한 라벨이 점과 겹쳐 있으며, 가장 위에 있는 점의 라벨은 첫 글자가 가려져 있는 것을 볼 수 있다. 라벨의 위치를 조정하는 작업은 시각적 요소 vjust와 hjust로도 할 수 있지만, 옵션 nudge_x와 nudge_y에 라벨의 위치를 이동시킬 값을 직접 지정하는 것이 조금 더 세밀한 정렬이 가능한 것으로 보인다.

```
> # 그림 13.28 (b)
> p + geom_text(data=filter(Cars93, MPG.highway>40),
            aes(label=paste(Manufacturer, Model)),
            nudge_y=1, nudge_x=100)
```



(a) (b)

<그림 13.28> 산점도의 점에 라벨 추가

위의 예제는 geom 함수마다 서로 다른 데이터 프레임을 사용할 수 있음을 보여주고 있다. 함수 geom_point()에서는 자료를 모두 사용하여 산점도를 작성했지만, 함수 geom_text()에서는 함수 filter()로 축소된 데이터 프레임을 사용하여 라벨을 붙이는 작업을 진행했다.

● 산점도에 주석 추가

산점도의 점에 대한 라벨이 아닌 그래프 자체에 주석을 추가하는 것도 함수 geom_text()로 할 수 있다. 만일 R의 수학기호 표현식을 사용하여 주석을 달고 싶다면 옵션 parse=TRUE를 추가해야 한다. 변수 Weight와 MPG.highway의 산점도에 회귀직선을 그리고, 결정계수를 주석으로 추가해 보자. 결정계수는 다음과 같이 구할 수 있다.

```
> fit <- lm(MPG.highway~Weight, Cars93)
> r2 <- summary(fit)$r.squared
```

그래프 작성 및 주석의 추가는 다음과 같이 할 수 있다.

```
> pp <- ggplot(Cars93, aes(x=Weight, y=MPG.highway)) +
    geom_point() +
    geom_smooth(method="lm", se=FALSE)

> # 그림 13.29
> pp +  geom_text(x=3500, y=45, size=7,
           label=paste("R^2==", r2), parse=TRUE)
```

함수 geom_text()에서 x, y와 size는 주석의 위치 및 크기를 지정하는 것이며, label에 주석의 내용을 입력하면 된다. 옵션 parse=TRUE가 지정되어서 수학기호 표현식을 인식할 수 있게 된다. 따라서 'R^2=='는 '$R^2 =$' 받아들여져 객체 r2에 할당된 값과 연결된다.

<그림 13.29> 산점도에 수학기호 주석 추가

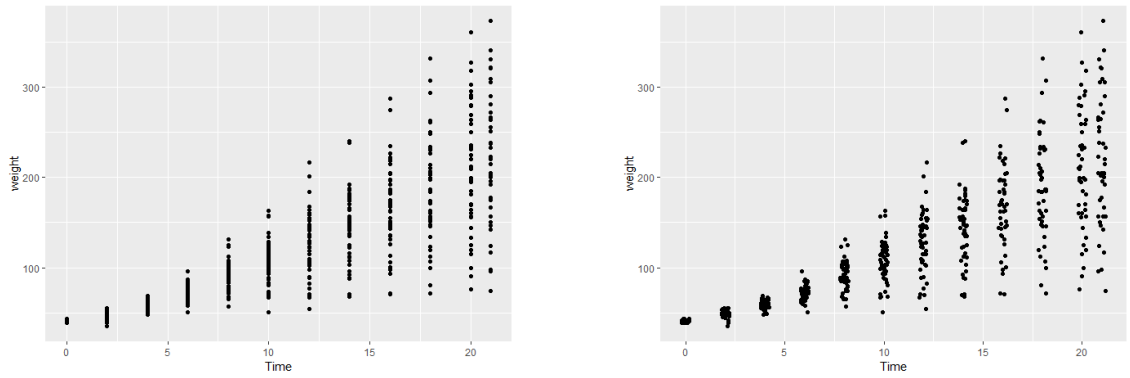
2) 산점도에서 점이 겹쳐지는 문제

산점도의 점들이 서로 겹쳐지면 자료의 분포나 관계 등을 확인하는 것이 어렵게 된다. 대규모의 자료를 대상으로 산점도를 작성하는 경우 종종 일어나는 현상이며, 두 변수 중 한 변수가 이산형인 경우에도 흔히 볼 수 있는 현상이다. 원자료를 반올림하여 정수로 만들었다면 이산형

자료가 아니어도 겹치는 현상이 발생할 수 있다. 점이 겹쳐지는 문제의 해결 방안은 상황에 따라 달라질 수 밖에 없으며, 마땅한 해결 방안이 없는 상황도 있을 수 있다.

데이터 프레임 ChickWeight는 578마리의 병아리를 대상으로 측정한 몸무게(weight)와 몸무게 측정이 생후 며칠에 이루어졌는지(Time)를 조사한 자료이다. 두 변수의 산점도를 작성해 보자.

```
> p1 <- ggplot(ChickWeight, aes(x=Time, y=weight))

> # 그림 13.30 (a)
> p1 + geom_point()
```



(a) (b)

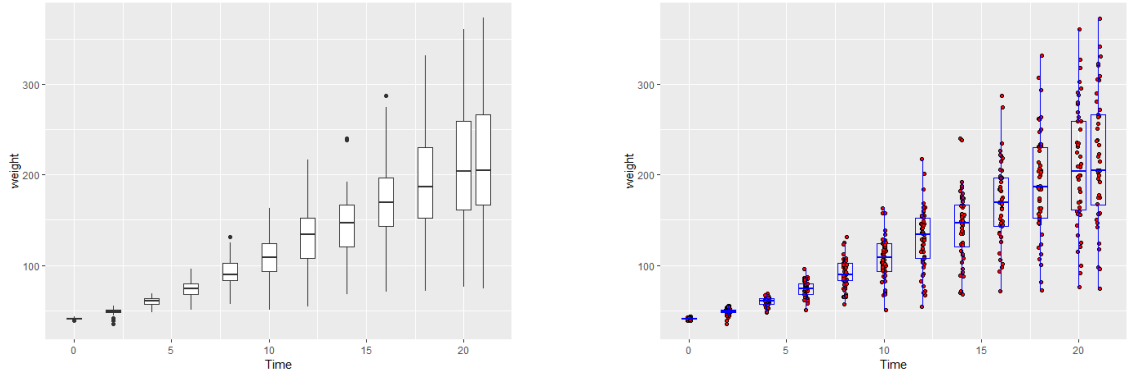
<그림 13.30> 산점도 점 겹치는 문제 해결 방안: 점 흩트리기

변수 Time은 12개의 값만을 갖고 있는 변수이기 때문에 많은 점들이 겹치게 될 수밖에 없다. 이런 경우에는 함수 geom_jitter()을 사용하여 점을 약간 흩트리면 점이 겹치는 정도를 약간 완화할 수 있다.

```
> # 그림 13.30 (b)
> p1 + geom_jitter(width=0.2, height=0)
```

두 변수 중 한 변수가 이산형 변수의 특성을 지니고 있다면 나란히 서 있는 상자그림을 산점도 대신 작성하는 것도 두 변수의 관계 탐색에 도움이 될 수 있다.

```
> # 그림 13.31 (a)
> p1 + geom_boxplot(aes(group=Time))
```



(a) (b)

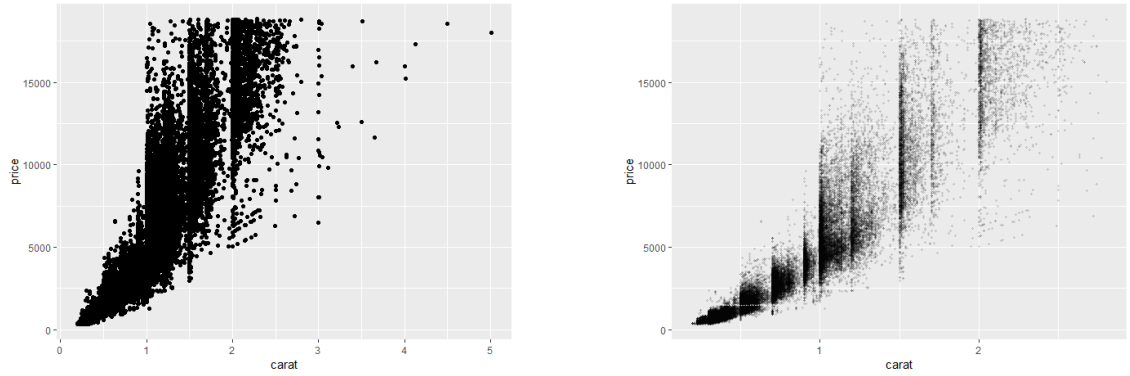
<그림 13.31> 산점도 점 겹치는 문제 해결 방안: 상자그림 추가

함수 geom_boxplot()으로 그룹별 상자그림을 작성할 때 x 변수인 Time이 숫자형 변수이기 때문에 반드시 시각적 요소 group에 추가로 매핑해야 한다. 상자그림의 문제는 각각의 상자그림 작성에 사용된 자료의 개수를 파악할 수 없다는 것이다. 이 문제는 함수 geom_jitter()를 추가하여 흩트린 점을 함께 나타내면 된다. 이 경우, 상자그림에서 이상값을 표시하지 않도록 해야 하며, 상자그림의 내부에 채워진 흰색도 없애야 한다.

```
> # 그림 13.31 (b)
> p1 + geom_jitter(width=0.1, fill="red", shape=21) +
    geom_boxplot(aes(group=Time), outlier.shape=NA, fill=NA, color="blue")
```

이번에는 대규모 자료를 대상으로 작성된 산점도에서 발생하는 점이 겹쳐지는 현상을 살펴보자. 패키지 ggplot2의 데이터 프레임 diamonds는 53,940개 다이아몬드를 대상으로 가격(price), 무게(carat), 컷의 등급(cut), 색상(color), 투명도(clarity) 등을 조사한 자료이다. 변수 carat과 price의 산점도를 작성해 보자.

```
> # 그림 13.32 (a)
> ggplot(diamonds, aes(x=carat, y=price)) +
    geom_point()
```



(a) (b)

<그림 13.32> 변수 carat과 price의 산점도

점들이 너무 심하게 겹쳐져서 자료의 분포를 알아보기 어려운 상황이다. 변수 carat이 3을

초과하는 자료가 많지 않으므로, carat<3인 자료만을 대상으로 산점도를 작성해 보자. 함수 geom_point()에서 점 모양은 shape=16이 디폴트인데, shape=20을 사용하면 점의 크기가 조금 줄어든다. 점의 투명도를 높이는 것도 해결 방안이 될 수 있다.

```
> p2 <- ggplot(filter(diamonds, carat<3), aes(x=carat, y=price))

> # 그림 13.32 (b)
> p2 + geom_point(alpha=0.1, shape=20)
```

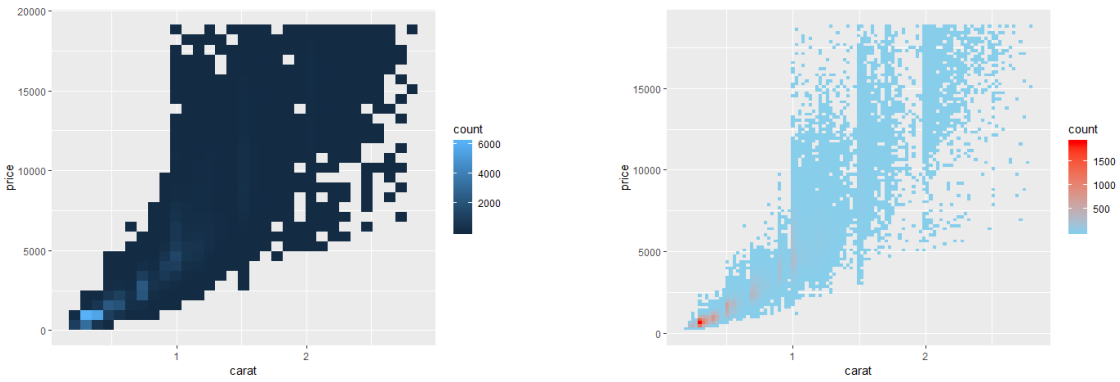
점의 투명도를 높이는 것으로는 문제가 해결되지 않는 것으로 보인다. 다만 특정 carat의 값에서 점들이 형성하고 있는 수직의 띠 개수가 더 늘어난 것이 보인다.

대안으로 제시할 수 있는 방안은 함수 geom_bin2d()를 사용하는 것인데, 이것은 XY축으로 형성된 2차원 공간을 직사각형의 2차원 구간(2D bin)으로 나누고, 그 구간에 속한 자료의 개수를 색으로 나타내는 그래프를 작성하는 것이다. 히스토그램을 2차원으로 확장한 그래프라고 할 수 있다. X축과 Y축을 30개의 구간으로 나누어 전체 공간을 900개의 직사각형 구간으로 나누는 것이 디폴트이다.

```
> # 그림 13.33 (a)
> p2 + geom_bin2d()
```

빈도수를 나타내는 색깔에 큰 변화가 없어서 구분이 잘 안되고 있고, 구간의 개수를 더 늘릴 필요가 있는 것으로 보인다. 디폴트 구간 개수인 30개를 원하는 개수로 조정하는 옵션은 bins이다. 색깔의 조정은 함수 scale_fill_gradient()에서 원하는 색깔을 low와 high에 각각 지정하면 된다.

```
> # 그림 13.33 (b)
> p2 + geom_bin2d(bins=100) +
    scale_fill_gradient(low="skyblue", high="red")
```



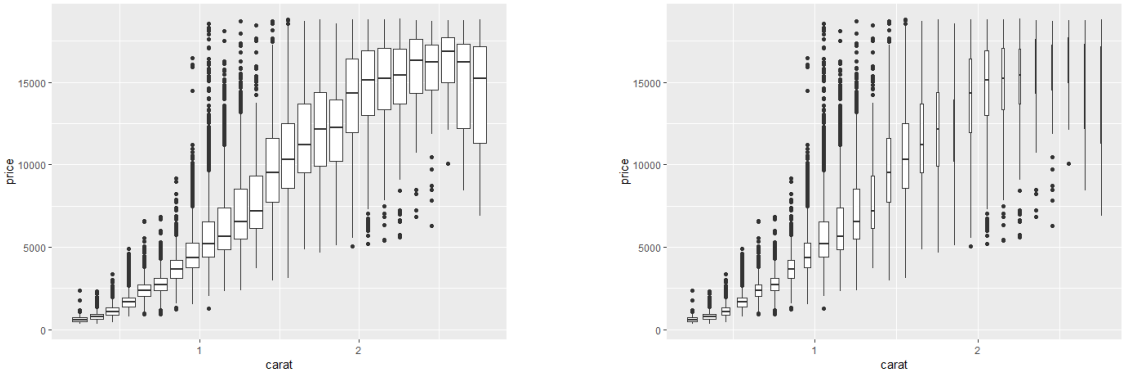
(a) (b)

<그림 13.33> 함수 geom_bin2d()로 작성된 그래프

대부분의 다이아몬드가 작고 가격이 상대적으로 싼 편이라는 것을 알 수 있다.

X축 변수인 carat이 이산형 변수가 아니지만, 이것을 구간으로 구분하여 나란히 서 있는 상자그림을 작성하는 것도 시도해 볼만하다. 숫자형 변수를 구간으로 구분하는데 유용한 함수가 cut_width()와 cut_number(), cut_interval()이다. 구간의 간격을 동일하게 한다면 cut_width(x, width, boundary)를 사용하면 된다. 옵션 boundary는 구간의 시작점을 지정할 때 사용한다. 반면에 자료를 n개 구간으로 구분하되, 각 구간에 속한 자료의 개수를 동일하게 한다면 cut_number(x, number=n)을 사용하면 되고, 같은 간격의 n개 그룹으로 구분한다면 cut_interval(x, n, length)를 사용하면 된다. 이 함수들이 결과는 각 구간을 수준으로 갖는 요인이다.

변수 carat을 0을 시작점으로 0.1의 간격을 갖는 구간으로 구분하고, 각 구간의 자료들을 대상으로 상자그림을 작성해 보자.



(a) (b)

<그림 13.34> 산점도의 대안으로 작성된 그룹별 상자그림

산점도에서는 인식하기 어려웠던 두 변수의 관계가 조금 드러나는 것을 볼 수 있다. 상자의 폭을 자료의 개수에 비례하도록 조절하면 각 구간에 속한 자료의 개수를 대략 비교할 수 있는데, 이것은 옵션 varwidth에 TRUE를 지정하면 된다.

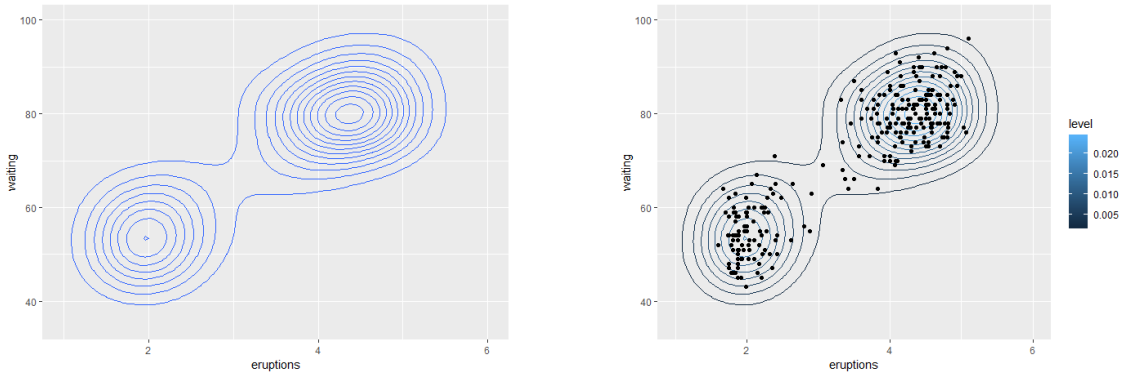
```
> # 그림 13.34 (a)
> p2 + geom_boxplot(aes(group=cut_width(carat, width=0.1,boundary=0)))

> # 그림 13.34 (b)
> p2 + geom_boxplot(aes(group=cut_width(carat, width=0.1,boundary=0)),
                    varwidth=TRUE)
```

3) 이차원 결합확률밀도 그래프

두 연속형 변수의 관계를 탐색할 때 두 변수의 결합확률밀도를 추정한 그래프가 큰 역할을 할 수 있다. 데이터 프레임 faithful의 두 변수 eruptions와 waiting의 결합확률밀도를 추정하여 그래프로 작성해 보자. 기본적인 그래프는 함수 geom_density_2d()로 작성할 수 있는데,

등고선 그래프가 작성된다.

```
> p3 <- ggplot(faithful, aes(x=eruptions, y=waiting)) +
    xlim(1,6) + ylim(35,100)

> # 그림 13.35 (a)
> p3 + geom_density_2d()
```



(a) (b)

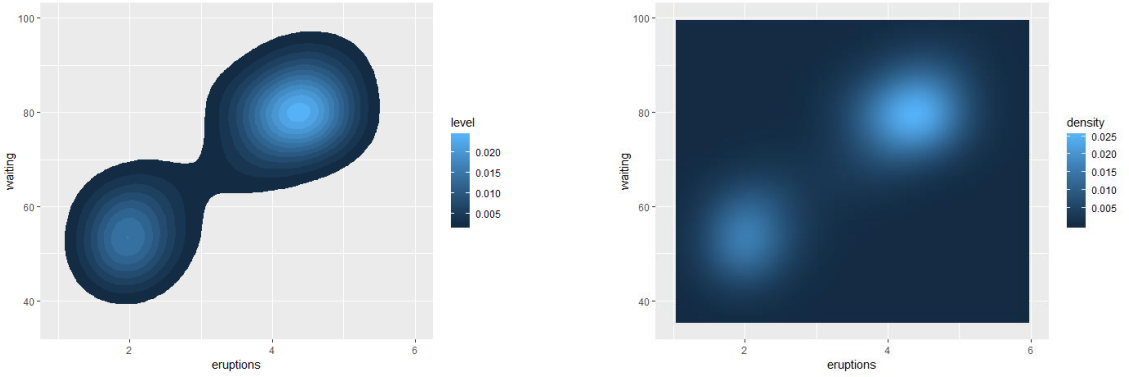
<그림 13.35> 결합확률밀도 등고선 그래프

등고선 그래프는 3차원 자료를 2차원 공간에 표시한 것으로 확률밀도가 같은 영역을 선으로 연결하여 그린 그래프다. 각 등고선에 적절한 라벨이 붙어 있어야 확률밀도가 높은 지역과 낮은 지역을 구분할 수 있지만, 라벨을 일일이 확인하여 높이를 구분하는 작업은 매우 번거롭고 부정확할 수밖에 없다. 대안으로 제시되는 방법은 등고선의 높이를 색으로 구분하는 것이다. 시각적 요소 color를 함수 stat_density_2d()에서 계산된 변수 ..level.. 또는 stat(level)에 매핑하여 선의 색깔을 구분시켜 보자. 최근에는 계산된 변수를 ..level..과 같이 점 두 개로 이름을 감싸는 방식보다 함수 stat()을 사용하는 것이 더 선호되는 방식이다. 두 변수의 산점도를 함께 작성하는 것도 좋을 것이다.

```
> # 그림 13.35 (b)
> p3 + geom_density_2d(aes(color=stat(level))) + geom_point()
```

등고선만을 색으로 구분하는 것보다는 높이가 같은 영역을 구분된 색으로 채우는 것이 더 효율적인 그래프의 작성이 될 것이다. 이 작업은 함수 stat_density_2d()를 사용하여 geom에 “polygon”을 지정하고, 시각적 요소 fill에 stat(level)을 매핑해야 한다.

```
> # 그림 13.36 (a)
> p3 + stat_density_2d(aes(fill=stat(level)), geom="polygon")
```



(a) (b)

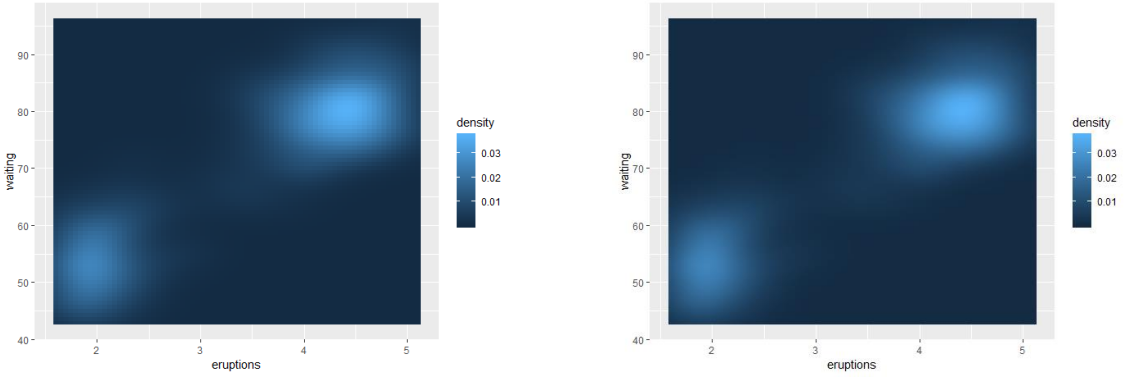
<그림 13.36> 색을 이용한 결합확률밀도 그래프

등고선 내부 영역만을 색으로 채우기보다 그래프 전체 영역을 타일로 구분하여 확률밀도의 높이에 따라 색을 다르게 하는 것이 훨씬 안정적인 느낌의 그래프를 작성하는 방법이 된다. 직사각형의 타일을 만드는 작업은 함수 geom_tile() 또는 geom_raster()를 사용해야 하는데, 두 변수에 대한 반응표면을 작성하는 경우에는 함수 geom_raster()를 사용하는 것이 좋다. 두 변수의 결합확률밀도 그래프도 일종의 반응표면을 작성하는 것이므로 geom_rater()를 사용해 보자. 타일의 색을 구분하기 위한 결합확률밀도의 추정은 함수 stat_density_2d()에서 stat(density)로 할 수 있다. 추정 결과를 시각적 요소 fill에 매핑하면 된다. 이 때 옵션 contour은 FALSE로 지정해야 한다.

```
> # 그림 13.36 (b)
> p3 + stat_density_2d(aes(fill=stat(density)), geom="raster",
                       contour=FALSE)
```

패키지 ggplot2에는 데이터 프레임 faithful의 두 변수인 eruptions와 waiting의 결합확률밀도 추정 결과가 변수(density)로 포함된 데이터 프레임 faithfuld가 있다. 변수 density에 할당된 결합확률밀도의 추정은 XY축에 75개의 격자점을 각각 구성하여 형성된 75×75=5625의 점에서 이루어졌다. 이와 같이 결합확률밀도의 높이가 데이터 프레임의 변수로 주어진 경우에는 함수 geom_raster()로 간단하게 그래프를 작성할 수 있다. 다만 이 경우에는 옵션 interpolate에 TRUE를 지정하는 것이 훨씬 안정적인 그래프를 작성하는 방법이다.

```
> # 그림 13.37 (a)
> ggplot(faithfuld, aes(x=eruptions, y=waiting)) +
    geom_raster(aes(fill=density))

> # 그림 13.37 (b)
> ggplot(faithfuld, aes(x=eruptions, y=waiting)) +
    geom_raster(aes(fill=density), interpolate=TRUE)
```



(a) (b)

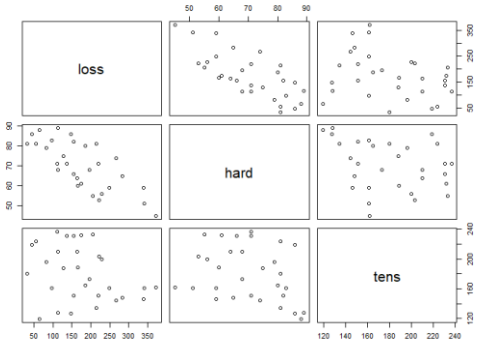
<그림 13.37> 색을 이용한 결합확률밀도 그래프

(4) 산점도 행렬(scatter plot matrix)

산점도 행렬은 여러 변수로 이루어진 자료에서 두 변수끼리 짝을 지어 작성된 산점도를 행렬의 행태로 배열하여 하나의 그래프에 함께 나타낸 그래프이다. 복잡하고 어려운 문제를 간단하고 명확하게 해결할 수 있도록 도와주는 뛰어난 그래프라고 할 수 있다. 패키지 graphics에서는 함수 pairs()가 산점도 행렬을 작성한다. 패키지 ggplot2에서는 산점도 행렬을 작성하는 함수가 없지만, 패키지 GGally의 함수 ggpairs()로 작성할 수 있다. 패키지 GGally는 ggplot2의 확장이라고 할 수 있는데, 산점도 행렬, 평행 좌표 그래프, 생존 그래프와 네트워크를 나타내는 그래프, 모형 검진을 위한 그래프 등등 유용한 그래프를 작성할 수 있다.

함수 pairs()의 기본적인 사용법은 pairs(x) 또는 pairs(formula)가 되는데, x는 행렬 혹은 데이터 프레임이고, formula는 ~x+y+z의 형태로 산점도 행렬을 구성하는 변수를 지정한다. 함수 pairs()에 데이터 프레임을 입력하면 데이터 프레임을 구성하는 모든 변수를 대상으로 산점도 행렬이 작성된다. 첫 번째 예제 자료로 패키지 MASS의 데이터 프레임 Rubber를 구성하고 있는 변수 loss, tens, hard의 산점도 행렬을 작성해 보자.

```
> data(Rubber, package="MASS")
> # 그림 13.38
> pairs(Rubber)
```

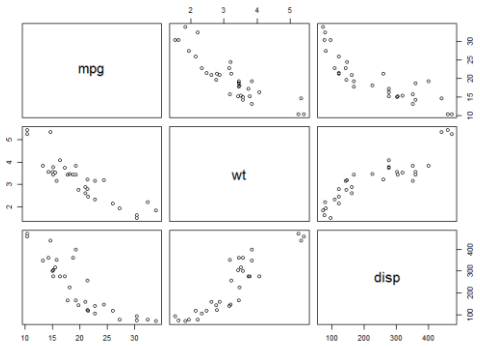


<그림 13.38> 데이터 프레임 Rubber의 산점도 행렬

3×3=9개의 패널로 구성된 행렬에서 대각선 패널에는 변수 이름이 나타나고 있으며, 나머지 패널에는 각각 짝지어진 두 변수의 산점도가 작성되어 있다. 대각선 패널의 위쪽에 있는 3개 패널과 대각선 패널의 아래쪽에 있는 3개 패널은 대각선을 중심으로 대칭이 된다.

데이터 프레임에 변수가 많은 경우, 필요한 변수를 선택하여 함수 pairs()에 입력해야 할 것이다. 예제 자료로 데이터 프레임 mtcars의 변수 mpg, wt, disp의 산점도 행렬을 작성해 보자. 다음의 두 가지 방법으로 변수를 선택하여 작성해 보자.

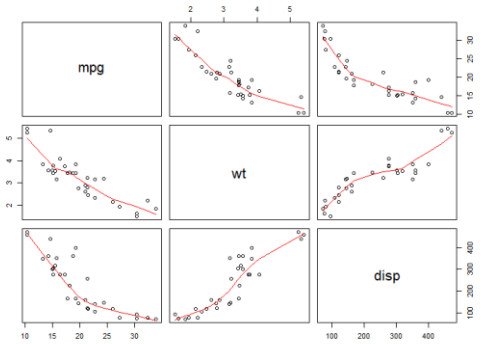
```
> # 그림 13.39
> mtcars %>%
    select(mpg, wt, disp) %>%
    pairs()
> # 그림 13.39
> pairs(~ mpg + wt + disp, data=mtcars)
```



<그림 13.39> 데이터 프레임 mtcars의 산점도 행렬

산점도 행렬의 각 패널에 그려지는 그래프는 옵션 panel에 원하는 그래프를 작성하는 함수를 지정함으로써 다양하게 변화시킬 수 있다. 많이 사용되는 패널 함수로는 panel.smooth()가 있다. 이 함수는 Cleveland가 제안한 로버스트 국소선형회귀 곡선을 작성하는데, 잔차의 절대값 크기에 반비례하도록 계산된 가중값을 이용하여 국소선형모형을 반복적으로 적합시킴으로 해서 극단값의 영향을 받지 않도록 만든 방법이다.

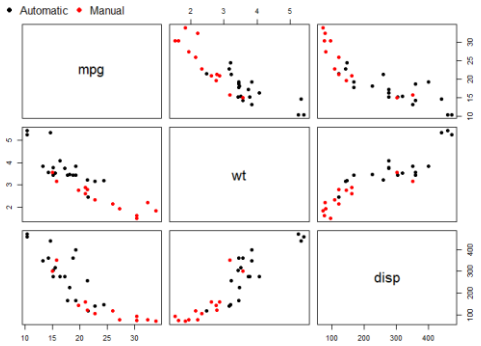
```
> # 그림 13.40
> pairs(~ mpg + wt + disp, data=mtcars, panel=panel.smooth)
```



<그림 13.40> 로버스트 국소선형회귀 곡선을 추가한 산점도 행렬

패널 함수는 사용자가 정의하여 사용할 수 있다. 예를 들어 mtcars의 변수 am은 변속기 종류를 나타내는 것으로 0=automatic이고 1=manual이다. 변수 mpg, wt, disp의 산점도 행렬을 작성하되 변수 am이 0인 자료는 검은 점으로, am이 1인 자료는 빨간 점으로 작성해 보자. 또한 적절한 범례로 추가해 보자.

```
> # 그림 13.41
> my_panel_1 <- function(x, y) points(x, y, col=mtcars$am+1, pch=16)
> pairs(~ mpg + wt + disp, data=mtcars, panel=my_panel_1)
> legend("topleft", c("Automatic", "Manual"), pch=16, col=c(1,2),
        xpd=TRUE, horiz=TRUE, bty="n", y.intersp=-0.01)
```



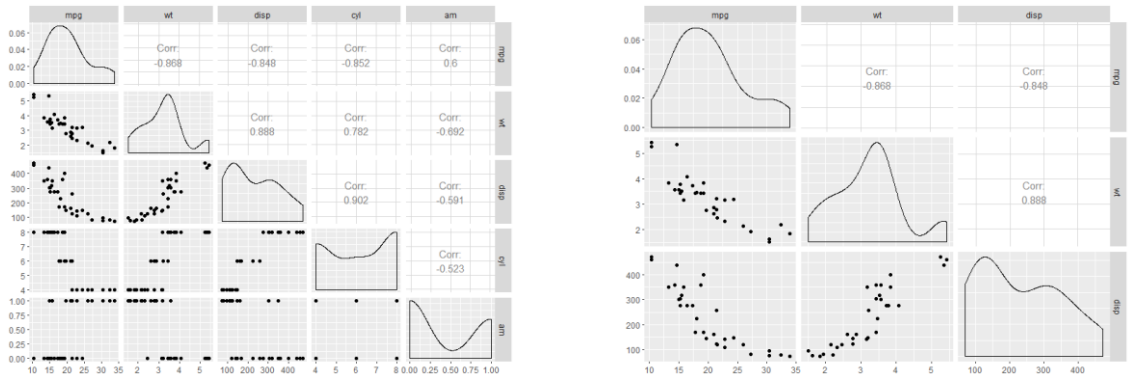
<그림 13.40> 사용자가 정의한 패널 함수에 의한 산점도 행렬

함수 my_panel_1()에서 am이 0인 경우에는 col=1, am이 1인 경우에는 col=2가 되도록 지정을 하였다. 산점도 행렬은 그래프 영역 전체를 사용하기 때문에 범례는 내부 마진에 위치해야 되므로 함수 legend()에 옵션 xpd에 TRUE을 지정했고, 범례의 작성 방향을 옆으로 하기 위해 옵션 horiz에 TRUE를 지정했다. 또한 사각 외곽선을 제거하기 위해 bty="n"을 지정했다. 옵션 y.intersp는 범례의 위치를 "topleft"와 같은 문자로 지정하는 경우 Y축 방향에서의 미세한 위치 조정을 위한 것이다. X축 방향에서의 위치 조정은 옵션 x.intersp로 할

수 있다.

이번에는 패키지 GGally의 함수 ggpairs()로 산점도 행렬을 작성해 보자. 데이터 프레임 mtcars에서 변수 mpg, wt, disp, cyl, am의 산점도 행렬을 작성해 보자. 함수 ggpairs()에는 사용될 변수만으로 이루어진 데이터 프레임을 입력하는 것이 좋다.

```
> mtcars_1 <- mtcars %>%
    select(mpg, wt, disp, cyl, am)

> library(GGally)
> # 그림 13.41 (a)
> ggpairs(mtcars_1)
```



(a) (b)

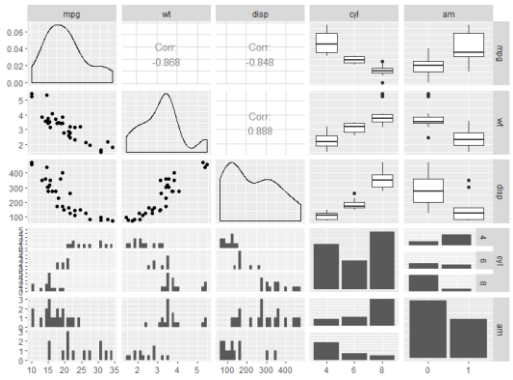
<그림 13.41> 숫자형 변수에 대한 함수 ggpairs()의 산점도 행렬

산점도 행렬에 포함시킬 변수를 옵션 columns으로 지정할 수 있다.

```
> # 그림 13.41 (b)
> ggpairs(mtcars_1, columns=1:3)
```

데이터 프레임 mtcars의 모든 변수는 숫자형 변수로 선언되어 있다. 변수 am과 cyl을 요인으로 전환시키고 다시 산점도 행렬을 작성해 보자.

```
> mtcars_2 <- mtcars_1 %>%
    mutate(am=factor(am), cyl=factor(cyl))
> # 그림 13.42
> ggpairs(mtcars_2)
```



<그림 13.42> 함수 ggpairs()로 작성된 그래프의 기본 형태

함수 pairs()로 작성된 산점도 행렬과는 많이 다른 모습의 그래프가 작성되는데, 단순히 산점도만을 작성하는 것이 아니라 변수의 유형에 따라 적합한 형태의 그래프를 각 패널에 작성한다. 또한 패널의 위치가 대각선인지, 혹은 대각선 위쪽인지, 혹은 대각선 아래쪽인지에 따라 다른 유형의 그래프가 작성되는 것이 디폴트이다. 변수를 유형별로 구분한다면 양적 변수와 질적 변수로 나눌 수 있다. 양적 변수는 숫자형 변수를 의미하며, 질적 변수는 범주형 변수를 의미하는 것으로 요인이 해당된다.

각 패널에 작성되는 디폴트 그래프를 살펴보자. 대각선 패널의 경우, 양적 변수는 확률밀도 그래프가 작성되고 질적 변수는 막대 그래프가 작성된다. 대각선 위쪽 패널의 경우, 두 변수가 모두 양적 변수이면 상관계수가 계산되고, 하나는 양적 변수, 다른 하나는 질적 변수이면 상자그림이 작성되며, 둘 다 질적 변수이면 facet 막대 그래프가 작성된다. 대각선 아래쪽 패널의 경우, 둘 다 양적 변수이면 산점도, 둘 다 질적 변수이면 facet 막대 그래프, 섞여 있으면 facet 히스토그램이 각각 작성된다.

각 패널에 작성되는 그래프를 바꾸는 방법을 살펴보자. 대각선 위 아래 패널의 경우에는 옵션 upper 혹은 lower를 이용해야 한다. 두 옵션은 변수 continuous와 combo, discrete로 이루어진 리스트로써 원하는 그래프에 해당하는 문자를 각 변수에 지정하면 된다. 양적 변수의 경우에는 변수 continuous에 “points”, “smooth”, “smooth_loess”, “density”, “cor” 중 하나를 지정하면 되고, 질적 변수의 경우에는 변수 discrete에 “facetbar”, “ratio” 중 하나를 지정하면 되며, 섞여 있을 경우에는 변수 combo에 “box”, “box_no_facet”, “dot”, “dot_no_facet”, “facethist”, “facetdensity”, “denstrip” 중 하나를 선택하면 된다. 만일 그래프 작성을 원하지 않는다면 “blank”를 지정하면 된다. 대각선 패널의 경우에는 옵션 diag에 변수 continuous와 discrete로 이루어진 리스트를 지정해야 한다. 변수 continuous에는 “densityDiag, “barDiag”, “blankDiag” 중 하나를 지정하고, 변수 discrete에는 “barDiag”, “blankDiag” 중 하나를 지정하면 된다.

각 그래프를 작성할 때 디폴트로 주어진 변수가 있다. 예를 들어 히스토그램을 작성할 때 구간은 bins=30으로 설정되는 것이 디폴트이다. 이러한 디폴트 값을 변경하고자 한다면 함수 wrap()을

사용하여 필요한 변수 값을 지정하면 된다.

이제 대각선 아래쪽 패널의 산점도에 회귀직선을 추가하고, 히스토그램의 구간 개수를 10개로 지정해 보자. 산점도에 회귀직선을 추가한 그래프 작성에 해당되는 문자는 "smooth"이다. 또한 시각적 요소 color에 요인 am을 매핑해 보자. 시각적 요소에 그룹 변수를 매핑하는 작업은 함수 aes() 안에서 하면 된다.

```
> # 그림 13.43
> ggpairs(mtcars_2, aes(color=am), columns=1:4,
          lower=list(continuous="smooth",
                     combo=wrap("facethist", bins=10)))
```

<그림 13.43> 함수 ggpairs()로 작성된 그래프

13.3 13장을 마치며

13장에서 우리는 이변량 자료가 갖고 있는 정보를 효과적으로 얻어내기 위한 그래프 작성 및 요약통계량을 계산하기 위한 R 함수의 사용법을 살펴보았다. 이변량 범주형 자료의 분할표 작성 및 그래프 작성을 할 수 있게 되었고, 이변량 이상의 연속형 변수들의 분포를 비교하는 그래프 및 기술통계량을 일목요연하게 정리할 수 있게 되었다. 또한 연속형 변수의 관계를 탐색하기 위한 다양한 그래프를 작성할 수 있게 되었다. 이제 다음 단계는 정리된 이변량 자료를 근거로 통계적 추론을 실시하는 것이다. 다음 장에서 우리는 이변량 자료와 관련된 통계적 추론에 사용할 수 있는 R 함수들의 사용법을 살펴보도록 하겠다.

13.4 연습문제

1. 패키지 UsingR에 있는 데이터 프레임 grades에는 122명의 학생들이 선수과목과 후수과목으로 연결된 어떤 강좌에서 받은 학점이 변수 prev와 grade에 각각 입력되어 있다. 두 변수는 요인으로 A부터 F까지 9개 수준으로 이루어져 있다.

   1) 두 범주형 변수 prev와 grade의 분할표를 다음과 같이 작성하라.

prev	grade A	A-	B+	B	B-	C+	C	D	F
A	15	3	1	4	0	0	3	2	0
A-	3	1	1	0	0	0	0	0	0
B+	0	2	2	1	2	0	0	1	1
B	0	1	1	4	3	1	3	0	2
B-	0	1	0	2	0	0	1	0	0
C+	1	1	0	0	0	0	1	0	0
C	1	0	0	1	1	3	5	9	7
D	0	0	0	1	0	0	4	3	1
F	1	0	0	1	1	1	3	4	11

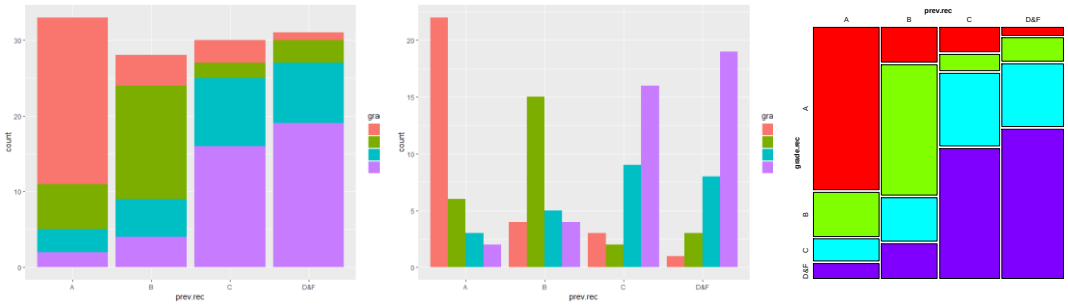
   2) 두 변수의 수준이 너무 세밀하게 구분되어 있어서 2차원 분할표에 빈도가 0인 칸이 다수 발생하였다. 이런 경우 인접한 범주를 합쳐 범주의 수를 줄이는 것이 한 방법이 될 수 있다. 이에 따라 변수 prev와 grade의 범주 (A, A-)를 A로, (B+, B, B-)를 B로, (C+, C)를 C로, 그리고 (D, F)를 D&F로 각각 합쳐 새로운 범주형 변수 prev.rec와 grade.rec를 생성시키고, 다음과 같이 두 변수의 분할표를 작성하라.

prev.rec	grade.rec A	B	C	D&F	Sum
A	22	6	3	2	33
B	4	15	5	4	28
C	3	2	9	16	30
D&F	1	3	8	19	31
Sum	30	26	25	41	122

   3) 선/후수 과목의 경우, 선수과목에서 받은 학점이 후수과목에서 받을 학점에 영향을 준다고 할 수 있는지 확인하고자 한다. 두 변수 prev.rec와 grade.rec를 이용하여 다음과 같은 조건분포 분할표를 작성하라.

prev.rec	grade.rec A	B	C	D&F
A	0.67	0.18	0.09	0.06
B	0.14	0.54	0.18	0.14
C	0.10	0.07	0.30	0.53
D&F	0.03	0.10	0.26	0.61

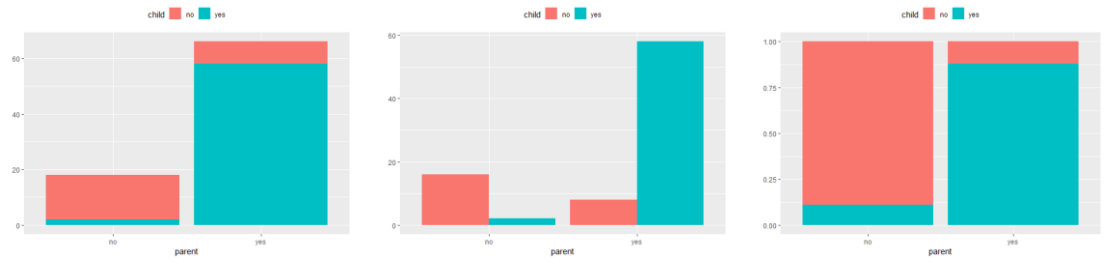
4) 두 범주형 변수의 관계를 나타내는 그래프로 쌓아 올린 막대그림, Spine plot, Mosaic plot이 있다. 두 변수 prev.rec와 grade.rec에 대하여 다음과 같이 세 종류의 그래프를 각각 작성하라.



쌓아 올린 막대 그래프 | 옆으로 붙여 놓은 막대 그래프 | Mosaic plot

2. 다음은 부모와 어린 자녀의 안전벨트 착용 여부에 대한 2차원 분할표로 비공식적인 조사로 얻어진 자료라고 한다. 부모의 안전벨트 착용 여부가 어린 자녀의 안전벨트 착용 여부에 미치는 영향을 살펴볼 수 있는 다음의 그래프를 작성해 보자.

	Child	
Parent	Yes	No
Yes	58	8
No	2	16

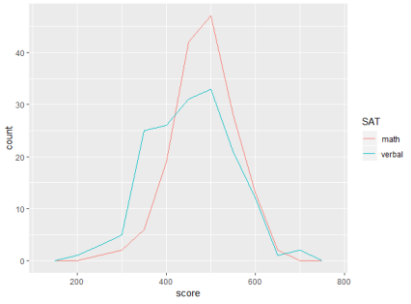


(a) (b) (c)

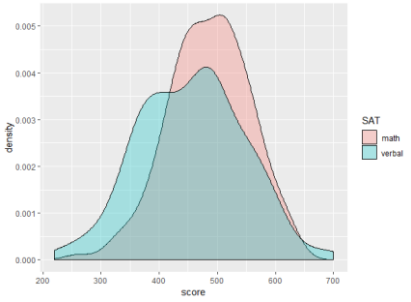
※범례 위치를 위로 옮기는 것은 `theme(legend.position="top")`으로 할 수 있다.

3. 패키지 UsingR에 있는 데이터 프레임 stud.recs에는 160명 학생들의 SAT 모의점수가 들어있다. 변수 sat.v는 verbal에 대한 점수, 변수 sat.m은 math에 대한 점수가 된다. 두 변수의 분포를 그래프를 이용하여 비교하고자 한다. 다음의 그래프를 각각 작성하라.

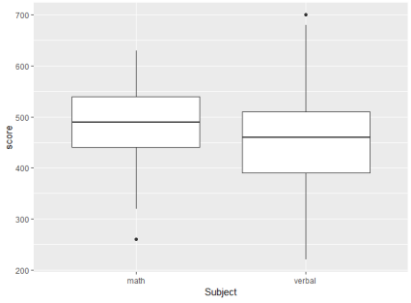
도수분포다각형



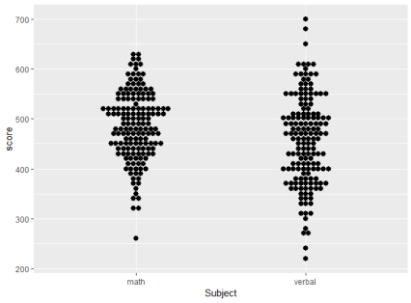
확률밀도 그래프



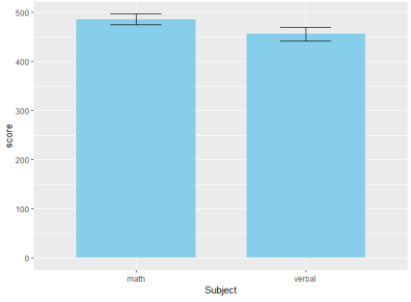
상자그림



다중 점 그래프

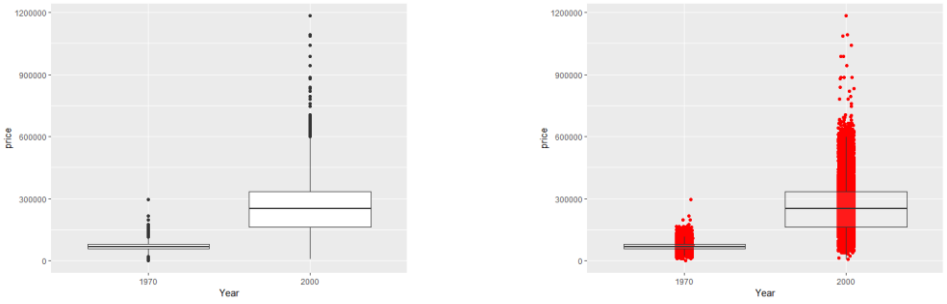


평균 막대 그래프와
95% 신뢰구간에 의한
error bar



4. 패키지 UsingR에 있는 데이터 프레임 homedata에는 미국 뉴저지 지역에서 임의로 선택한 6841채 주택의 1970년과 2000년 가격이 각각 변수 y1970과 y2000에 입력되어있다.

   1) 두 변수의 분포를 다음과 같이 상자그림을 작성하여 비교해 보자. 오른쪽 그래프는 상자그림에 자료의 위치를 점으로 함께 나타낸 것이다.

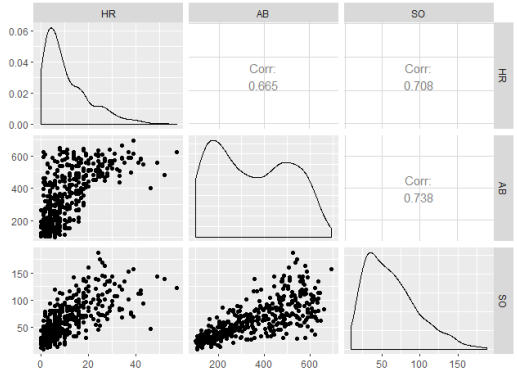


   2) 선택된 6841채의 주택 중 2000년의 주택가격이 1970년의 가격보다 상승한 주택과 하락한 주택의 빈도수를 구하고 다음과 같이 도수분포표로 정리해 보자.

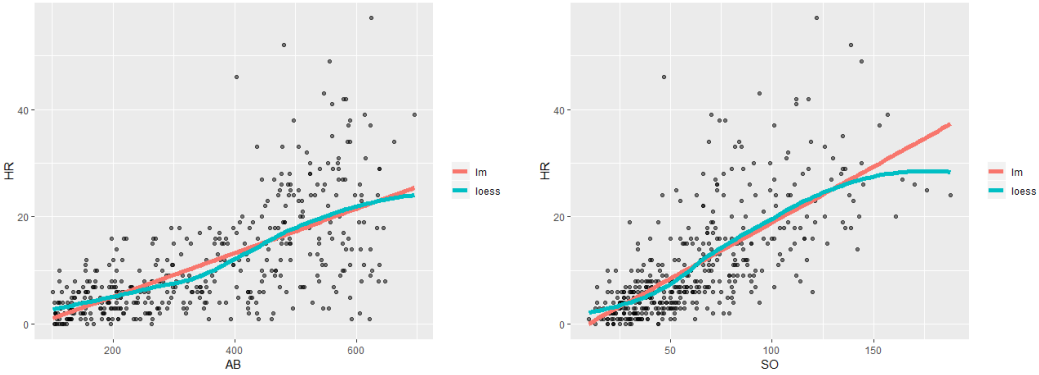
집값 하락	집값 상승
1	6840

5. 패키지 UsingR에 있는 데이터 프레임 batting에는 미국 MLB의 2002년 시즌에 참여했던 438명 선수들에 대한 통계가 22개의 변수에 입력되어있다. HR은 홈런 수, AB는 타석 수, SO는 삼진 수를 나타내는 변수이고, playerID로 선수를 구분할 수 있다.

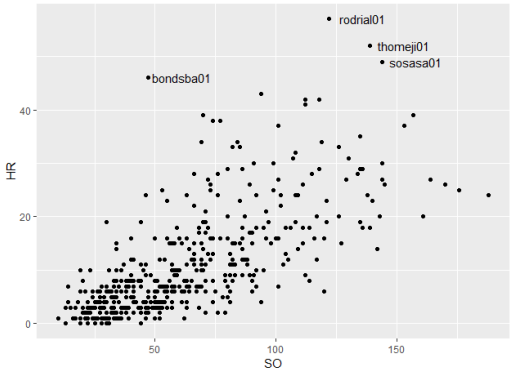
   1) 변수 HR과 AB, SO의 산점도 행렬을 다음과 같이 작성해 보자.



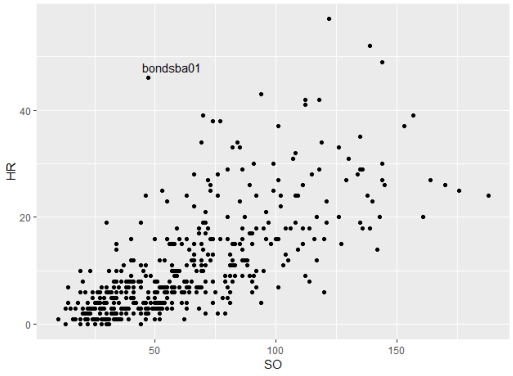
   2) 변수 HR과 AB, 그리고 HR과 SO의 산점도를 각각 작성하되, 회귀직선과 국소회귀곡선을 다음과 같이 추가해 보자.



3) 변수 HR과 SO의 산점도를 작성하되 HR이 45이상인 선수들의 playerID를 해당 점 옆에 다음과 같이 표시해 보자.



4) 변수 HR과 SO의 산점도를 작성하되 타석 당 홈런 수(HR/AB)가 최대인 선수의 playerID를 해당 점 옆에 다음과 같이 표시해 보자.



5) 변수 HR과 SO 사이에 비교적 강한 양의 상관관계가 관측되었다. 이것을 홈런을 많이 치기 위해서는 삼진을 많이 당해야 한다는 원인-결과의 방식으로 해석할 수 있는가?

6. 데이터 프레임 iris는 3종류 붓꽃 각 50송이의 꽃받침 조각의 길이(Sepal.Length)와 폭(Sepal.Width), 꽃잎의 길이(Petal.Length)와 폭(Petal.Width)을 측정한 자료이다.

변수 Species에는 붓꽃의 종류(setosa, versicolor, virginica)가 입력되어 있다. 측정된 변수를 이용하여 붓꽃의 종류를 구분할 법칙을 발견하고자 한다. 분석의 첫 단계로 측정된 네 변수의 산점도 행렬을 작성해 보자. 함수 pairs()와 GGally::ggpairs()를 사용하여 각각 다음과 같이 작성해 보자.