빅데이터분석입문(2)

신봉균 20191624

2023-03-29

02 단일변수 범주형 자료의 탐색

코드 5-1

```
favorite = c('WINTER','SUMMER','SPRING','SUMMER','SUMMER','FALL','FALL','SUMMER','SPRING','SPRING')
favorite        #favorite 의 내용 출력

##  [1] "WINTER" "SUMMER" "SPRING" "SUMMER" "SUMMER" "FALL"   "FALL"   "SUMMER"
##  [9] "SPRING" "SPRING"

table(favorite)  #도수분포표 계산

## favorite
##   FALL SPRING SUMMER WINTER
##      2      3      4      1

table(favorite)/length(favorite) #비율 출력

## favorite
##   FALL SPRING SUMMER WINTER
##    0.2    0.3    0.4    0.1
```

코드 5-1 의 실행 결과를 나누어 설명하면 다음과 같다. 자료를 분석하기 위해 **favorite** 이라는 이름의 벡터에 자료를 저장한다. 이와 같이 단일변순 자료는 벡터에 저장하여 분석하는 것이 일반적이다. 10 개의 자료값이 favorite 에 저장되어 있을을 알 수 있다.

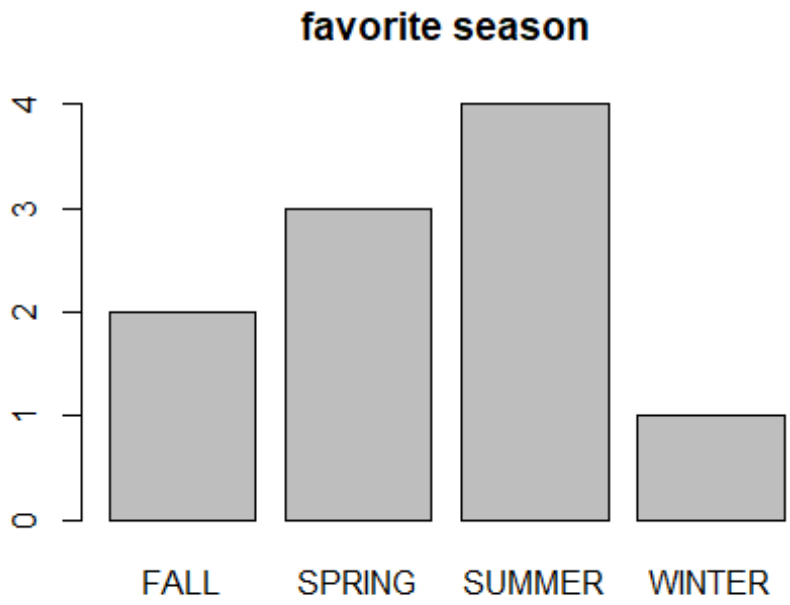
table()함수는 벡터에 저장된 범주형 자료에 대해 자료값의 종류별로 도수분포표를 계산해주는 함수이다. 실행 결과에서 **favorite** 은 자료가 저장된 벡터의 이름이고, 아랫부분은 4 계 절에 대해 선호 빈도를 계산해서 나타낸다. 선호 빈도를 보면 **SUMMER** 가

4 명으로 가장 많고, **WINTER** 가 1 명으로 가장 적다. 이와 같이 도수분포는 각 자료의 종류별로 빈도를 파악할 수 있도록 해준다.

table(favorite)/length(favorite) 이 명령어는 각 관측값의 종류별 비율을 계산하는 작업을 수행한다. 도수분포표에 있는 각 빈 도를 자료의 전체 개수 **length(favorite)**로 나누면 비율을 계산할 수 있다. 실행 결과에서 **0.2 는 20%,** 0.4 **는 40%와 같이 백분율을 의미한다.** SUMMER **를 좋아하는 사람은 전체의 40%,** WINTER**를 좋아하는 사람은 전체의 10%임을 알수 있다.

코드 5-2

```
ds = table(favorite)
ds

## favorite
##   FALL SPRING SUMMER WINTER
##      2      3      4      1

barplot(ds, main='favorite season')
```

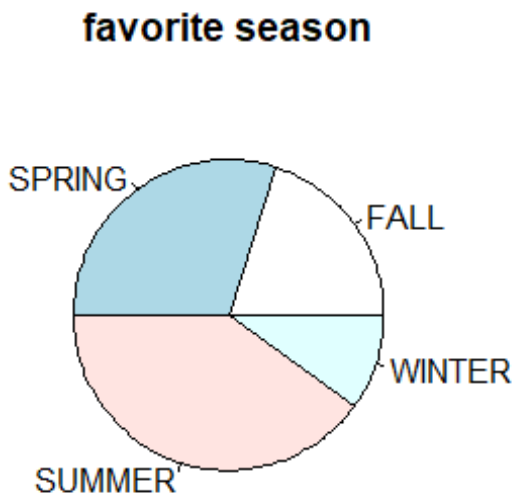


table() 함수를 통해 계산한 도수분포표를 **ds** 라는 이름의 변수에 저장하는 명령어이다. **ds** 의 내용을 확인해보면 도수분포표가 들어 있는 것을 알 수 있다.

barplot() 함수는 막대그래프를 작성하는 함수로 도수분포표를 기본적인 입력 매개변수로 받아들인다. **main** 은 막대그래프 상단의 타이틀을 지정하는 매개변수로, *코드 5-2 의 경우* 'favorite season'* 문자열을 입력값으로 하고있다. 이 함수의 실행 결과로 출력되는 막대 그래프는 *R* 스튜디오 우측 하단의 플롯 창에서 확일할 수 있다. 그래프를 보면 x 축은 관 측값의 종류를 나타내는 4 계절의 이름을 보여주고, y 축은 선호 빈도를 나타낸다. **barplot()** 함수는 다양한 매개변수를 가질 수 있으며, 매개변수값에 따라 막대의 색을 다르게하거나 X 축, y 축의 레이블을 지정할 수도 있다.

코드 5-3

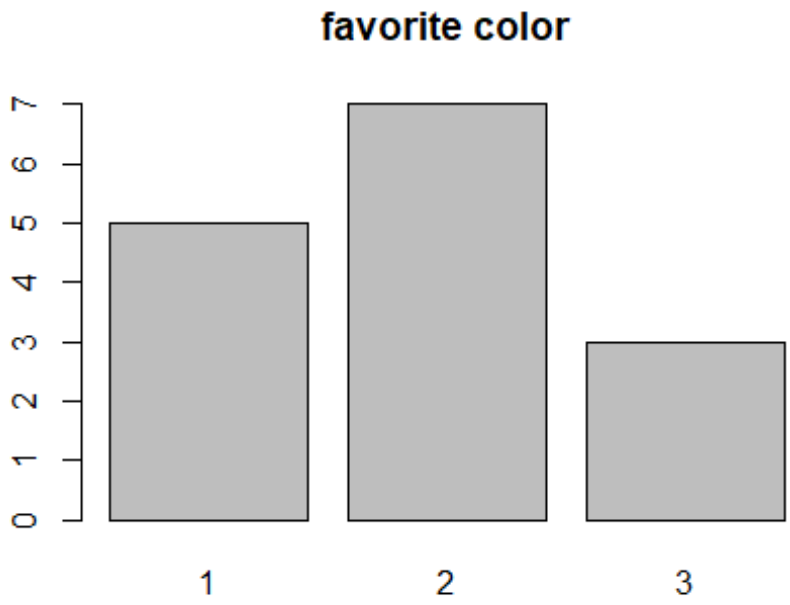
```
ds = table(favorite)
ds

## favorite
##   FALL SPRING SUMMER WINTER
##      2      3      4      1

pie(ds, main='favorite season')
```



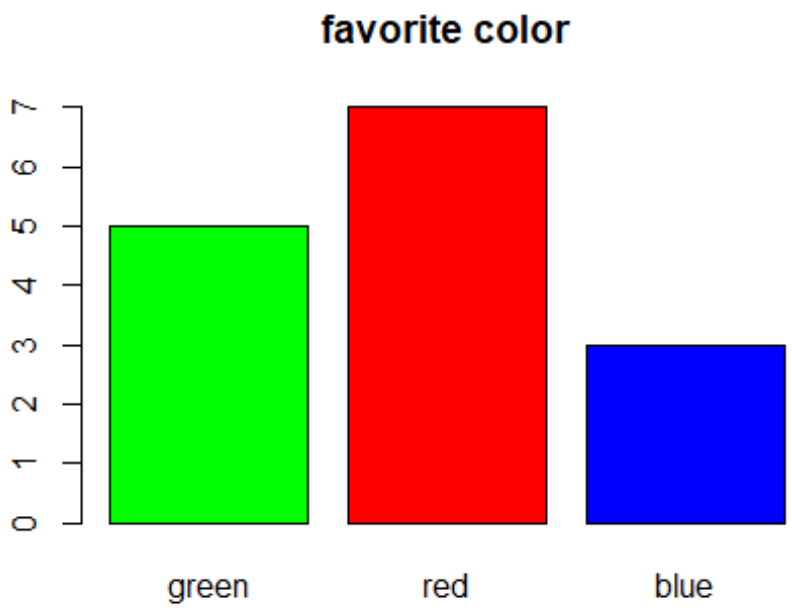
원그래프를 그리는 함수 이름은 **pie()**이며 실행방법은 **barplot()** 함수와 유사하다. 이함수의 실행 결과로 출력되는 원그래프는 *R* 스튜디오 우측 하단의 플롯 창에서 확인할 수 있다.

코드 5-4

```
favorite.color = c(2,3,2,1,1,2,2,1,3,2,1,3,2,1,2)
ds= table(favorite.color)
ds

## favorite.color
## 1 2 3
## 5 7 3

barplot(ds, main='favorite color')
```




```
colors = c('green','red','blue')
names(ds) = colors    #자료 값 1,2,3 을 grenn, red, blue 로 변경
ds

## green   red  blue
##     5     7     3

barplot(ds, main = 'favorite color', col= colors) #색 지정 막대그래프
```




```
pie(ds, main='favorite color', col=colors)
```

2 에서는 단일변수 범주형 자료를 분석하는 방법을 살표보았다. 범주형 자료의 값은 크기를 갖지 않기 때문에 도수분포를 계산한 다음 이를 막대그래프나 원그래프로 시각화하여 자료의 내용을 파악할 수 있다.

03 단일변수 연속형 자료의 탐색

코드 5-5

```
weight = c(60,62,64,65,68,69)
weight.heavy = c(weight, 120)
weight

## [1] 60 62 64 65 68 69

weight.heavy

## [1]  60  62  64  65  68  69 120

mean(weight)          #평균

## [1] 64.66667

mean(weight.heavy)    #평균

## [1] 72.57143

median(weight)   #중앙값

## [1] 64.5

median(weight.heavy) #중앙값

## [1] 65

mean(weight, trim=0.2)         #절사평균(상하위 20%제외)

## [1] 64.75

mean(weight.heavy, trim=0.2)   #절사평균(상하위 20%제외)

## [1] 65.6
```

코드 5-5 의 실행 결과를 나누어 설명하면 다음과 같다.

```
weight = c(60,62,64,65,68,69)
weight.heavy = c(weight, 120)
print(weight)
```

```
## [1] 60 62 64 65 68 69
```

```
weight.heavy
```

```
## [1]  60  62  64  65  68  69 120
```

비슷한 값들이 저장되어 있는 벡터 **weight** 와 **weight.heavy** 를 생성하고 내용을 출력한다. **weight.heavy** 에 특이값 120 이 포함되어 있는 것을 제외하면 **weight** 와 **weight.heavy** 는 동일한 값들이 저장되어 있다.

두 개의 벡터에 저장된 값들의 평균이 큰 차이가 나는데, 이는 **weight.heavy** 에 있는 값 120 의 영향 때문이다.

두 벡터의 중앙값에는 큰 차이가 없음을 알 수 있다. 즉, 중앙값은 특이값의 영향을 덜 받는 것이다.

매개변수 **trim** 은 상하위 값들을 몇% 정도 제외하고 평균을 구할 것인지를 지정하는 역할을 한다. **trim=0.2** 는 상하위 각 20%를 제외하고 평균을 구하라는 의미이다. 절사평균은 특이값을 제외하고 나머지 값들로평균을 구하는 효과가 있기 때문에 두 벡터의 **절사평균값**은 큰 차이가 없다.

코드 5-6

```
mydata = c(60,62,64,65,68,69,120)
quantile(mydata)
```

```
##    0%   25%   50%   75%  100%
##  60.0  63.0  65.0  68.5 120.0
```

```
quantile(mydata, (0:10)/10)   #10% 단위로 구간을 나누어 계산
```

```
##    0%   10%   20%   30%   40%   50%   60%   70%   80%   90%  100%
##  60.0  61.2  62.4  63.6  64.4  65.0  66.8  68.2  68.8  89.4 120.0
```

```
summary(mydata)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   60.00   63.00   65.00   72.57   68.50  120.00
```

코드 5-6 의 실행 결과를 설명하면 다음과 같다.

mydata 에 7 개의 값을 저장하고 **quantile()** 함수를 통해 사분위수를 구한다. **25%,50%,75%** 에 해당하는 값이 사분위수이고, **0%**는 mydata 의 최솟값, **100%**는 mydata 의 최댓값을 나타낸다.

quantile() 함수의 매개변수 중 **(0:10)/10** 부분이 구간을 몇 개로 나눌지를 지정한다. 사분위수는 전체 자료를 네 개의 구간으로 나누지만 경웨 따라 더 세분화하여 나눌 필요가 있을 수 있다. **(0:10)/10** 의 의미는 0~10 의 정수를 10 으로 나누라는 것이다. 백분율은 10% ~ 100%가 된다. 그렇기 때문에 앞의 결과와 같이 10% 단위로 구간을 나누어 결과값이 출력된다.

사분위수를 구할때 가장 일반적으로 사용되는 함수가 **summary()**함수이다. **summary()**함수는 사분위수에 최댓값과 최솟값과 평균을 함께 출력하기 때문에 편리하다.

코드 5-7

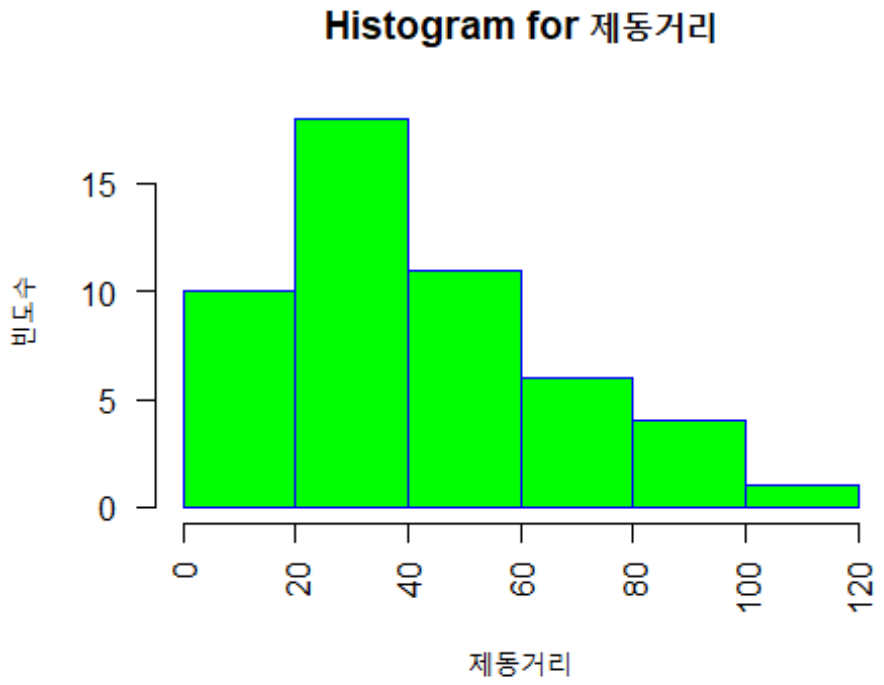
```
mydata = c(60,62,64,65,68,69,120)
var(mydata)    #분산

## [1] 447.2857

sd(mydata)   #표준편차

## [1] 21.14913

range(mydata)     #값의 범위

## [1]  60 120

diff(range(mydata))   #최댓값 과 최솟값 차이

## [1] 60
```

var() 함수는 **variation** 의 약자임으로 분산을, **sd()**는 **standard deviation** 약자 임으로 표준편차를 **range()**는 최댓값과 최솟값의 범위를 **diff()**는 최댓값과 최솟값으 차이를 나타낸다.

코드 5-8

```
dist = cars[,2] #자동차 제동거리
hist(dist,            #자료
     main = 'Histogram for 제동거리',   #제목
```

```
    xlab= '제동거리',                    #x 축 레이블
    ylab= '빈도수',                       #y 축 레이블
    border = 'blue',                    #막대 테두리색
    col ='green',                       #막대 색
    las = 2,                            #x 축 글씨 방향(0~3)
    breaks=5                            #막대 개수 조절
    )
```

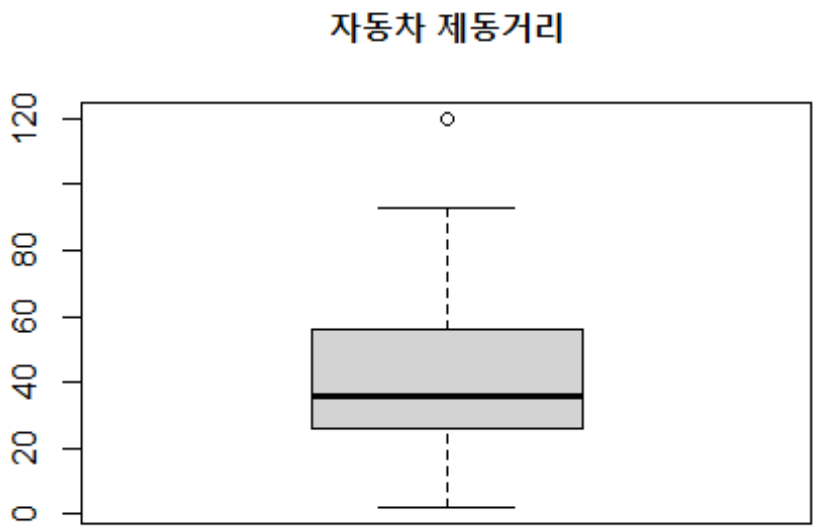


hist() 함수에서 매개변수 **las** 는 **x** 축에 표시되는 값들 (0,20,40,60,80,100,120)의 출력 방향을 조절하는 역할을 한다. 출력을 가로 방향, 세로 방향 등으로 조절할 수 있는데 **las=2** 일 때 글씨는 세로 방향으로 출력된다. 매개변수 **breaks** 눈 구간을 몇 개로 나눌지(막 대를 몇 개로 할지)를 조절하는 역할을 한다. **breaks** 의 값이 커지면 구간의 개수도 늘어나고, **breaks** 의 값이 작아지면 구간의 개수도 줄어든다.

hist() 함수의 결과를 보면 제동거리가 20~40 사이에 있는 것이 대략 18 개 정도로 가장 빈도수가 높은 것을 알 수 있다. 이와 같이 히스토그램은 관측값들이 어느 구간에 분포하는지를 쉽게 파악할 수 있도록 해준다.

코드 5-9

```
dist = cars[,2]    #자동차 제동거리(단위: 피트)
boxplot(dist, main='자동차 제동거리')
```



dist 벡터에는 자동차의 제동거리 자료가 저장되어있다. 위의 상자그림은 y 축의 눈금이 20 단위, 제동거리 값이 0~120 범위에 있다. 이 중 정상범위의 제동거리는 대략 1~95 사이이다. 정상범위 내에서는 상자가 아래좍으로 치우쳐 있는데, 전체의 50%에 해당하는 값들이 아래족에 분포됨을 알려준다. 상자그림에서 상자 위쪽으로 특이값 1 개가 관찰되는데, 다른 경우에 비해 제동거리가 매우 긴 경우를 의미한다.

상자그림은 자료의 전반적인 분포를 이해하는 데 도움이 되지만 구체적으로 최솟값, 최댓값, 중앙값 등의 정확한 값을 알기는 어렵다. 이를 알기 위해서는 **boxplot.stats()** 함수를 사용하 며 실행 결과는 리스트 형태로 출력된다.

코드 5-10

```
boxplot.stats(dist)
```

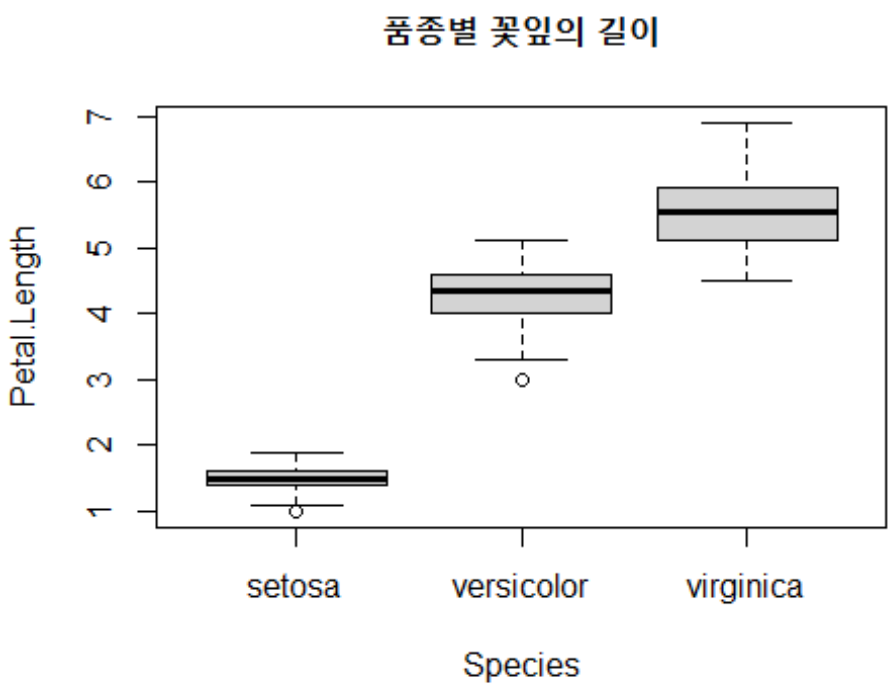
```
## $stats
## [1]  2 26 36 56 93
```

```
##
## $n
## [1] 50
##
## $conf
## [1] 29.29663 42.70337
##
## $out
## [1] 120
```

먼저 *stats* **
에는정상범위자료의 4 *분위수에해당하는값들이표시된다.*5 *개의값은차례로최솟값,*1 *사분위수, 중앙값*
∗**n** 의 값은 자료에 있는 관측 값들의 개수로, 총 50 개의 관측값을 저장하고 있다는 것을 의미한다. **conf* ∗ *는중앙값에관련된신뢰구간을의미,*out **은 특이값의 목록을 알려준다.**
만약 특이 값들을 출력시키려면 boxplot.stats(dist)$out**과 같은 명령문을 사용한다.

코드 5-11

```
boxplot(Petal.Length~Species, data=iris, main='품종별 꽃잎의 길이')
```

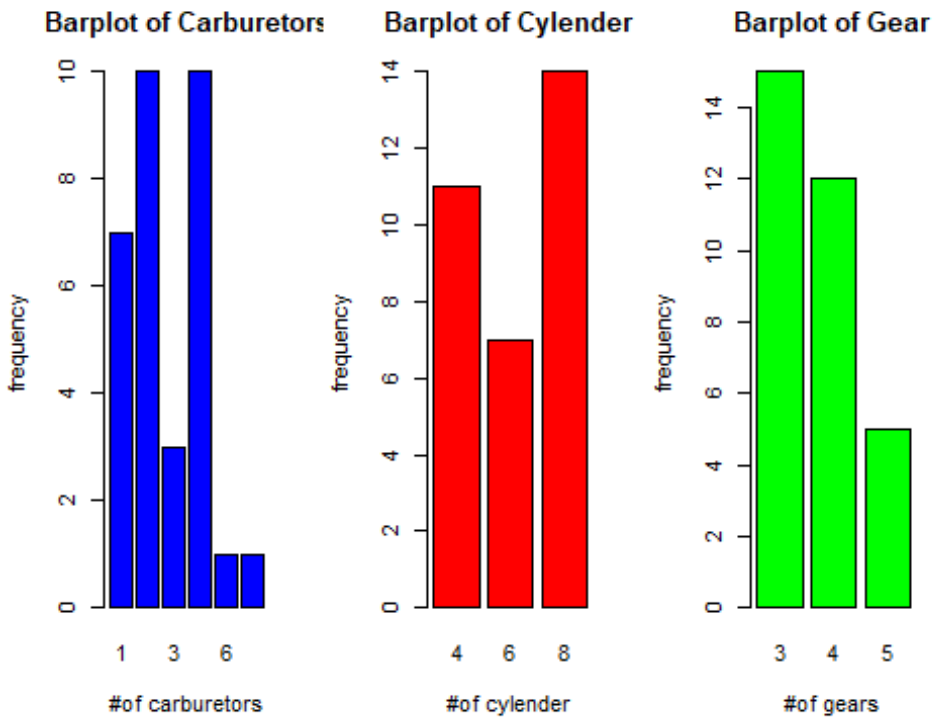


boxplot() 함수의 매개변수에서 **Petal.length~Species** 는 품종별 꽃잎의 길이를 의미한다, **data=iris** 는 데이터는 iris 에서 불러온다는 뜻이고 **main** 은 해당 **boxplot** 의 이름을 지정해주는 인자이다.

한 화면에 그래프 여러개 출력하기

R 을 이용하여 그래프를 그리다 보면 한 번에 하나씩만 그래프가 출력되는 것을 알 수 있다. R 에서는 화면을 여러 개로 가상 불한한 후, 각각의 분할된 화면에 여러 개의 그래프를 출력하는 기능을 제공한다. 다음 코드를 실행해보자.

```
par(mfrow=c(1,3))           #1x3 가상화면 분할
barplot(table(mtcars$carb),
        main='Barplot of Carburetors',
        xlab= '#of carburetors',
        ylab= 'frequency',
        col='blue')

barplot(table(mtcars$cyl),
        main= 'Barplot of Cylender',
        xlab='#of cylender',
        ylab='frequency',
        col='red')



barplot(table(mtcars$gear),
        main= 'Barplot of Gear',
        xlab='#of gears',
        ylab='frequency',
        col='green')
```




```
par(mfrow=c(1,1))    #가상화면 분할 해제
```

가상화면을 분활 해주는 부분은 **par(mfrow=c(1,3))**이다. 여기서 1 은 가상화면의 행의수 3 은 가상화면의 열의 수를 나타낸다.