소프트웨어시스템실습

5강: 데이터 시각화 (그래프)

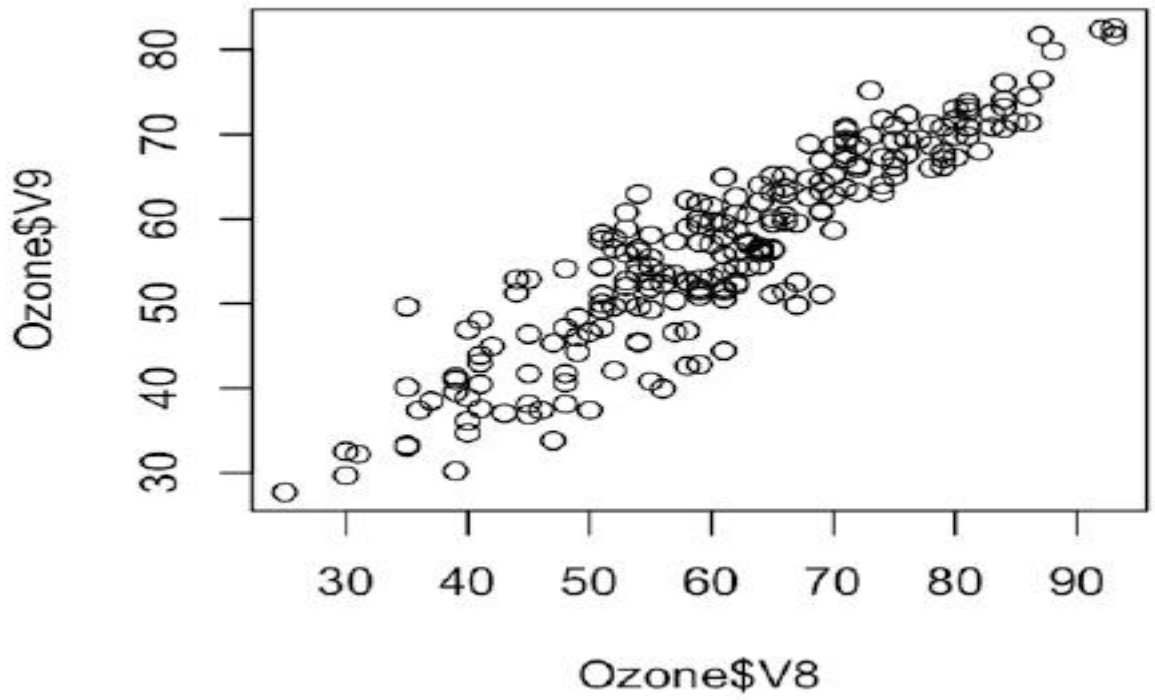
산점도(scatter plot)

- plot(x, y)
- 예)

```
> install.packages("mlbench")
> library(mlbench)
> data(Ozone)
> plot(Ozone$V8, Ozone$V9)
```

Mlbench패키지에 저장된 Ozone데이터 loading

각각 캘리포니아 Sandburg와 El Monte에서 매일 측정한 온도

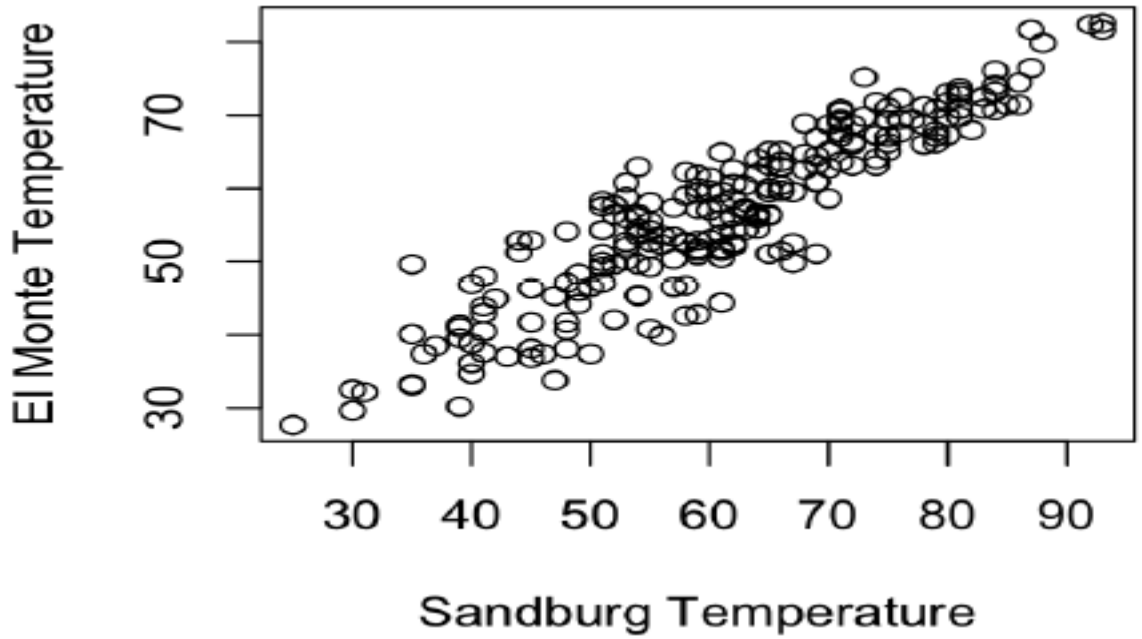


Sandburg(V8)와 El Monte(V9)지역의 온도

그래픽 옵션

- 축 이름(xlab, ylab)

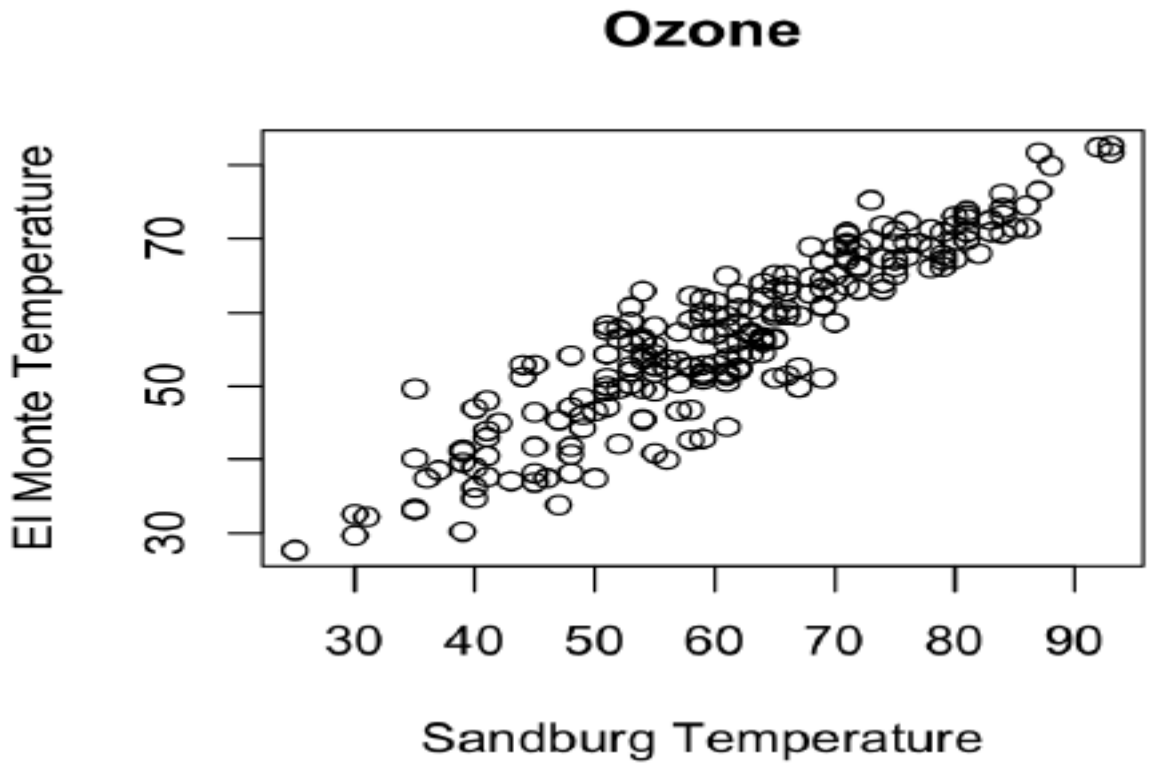
```
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+       ylab="El Monte Temperature")
```



그래픽 옵션

- 그래프 제목(main)

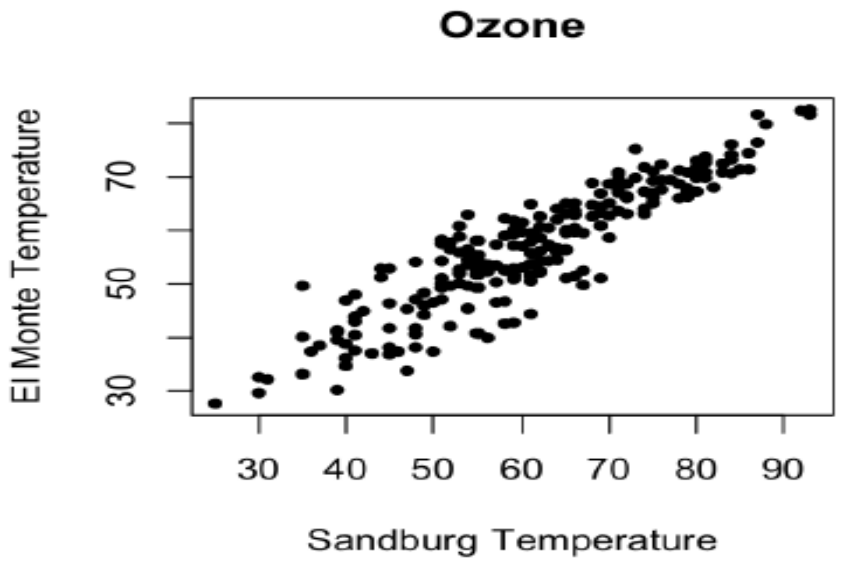
```
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+       ylab="El Monte Temperature", main="Ozone")
```



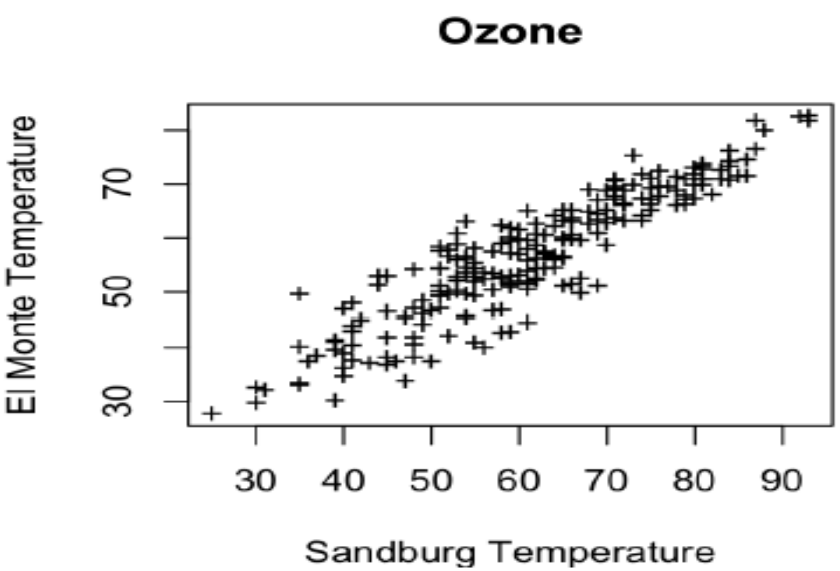
그래프 옵션

- **점의 종류(pch)**
  - 숫자를 지정하면 미리 지정된 심볼 (구글에서 'r pch symbols' 로 검색)
  - 문자(예를들어 '+')를 지정하면 그 문자를 사용

```
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+       ylab="El Monte Temperature", main="Ozone", pch=20)
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+       ylab="El Monte Temperature", main="Ozone", pch="+")
```



(a) pch=20

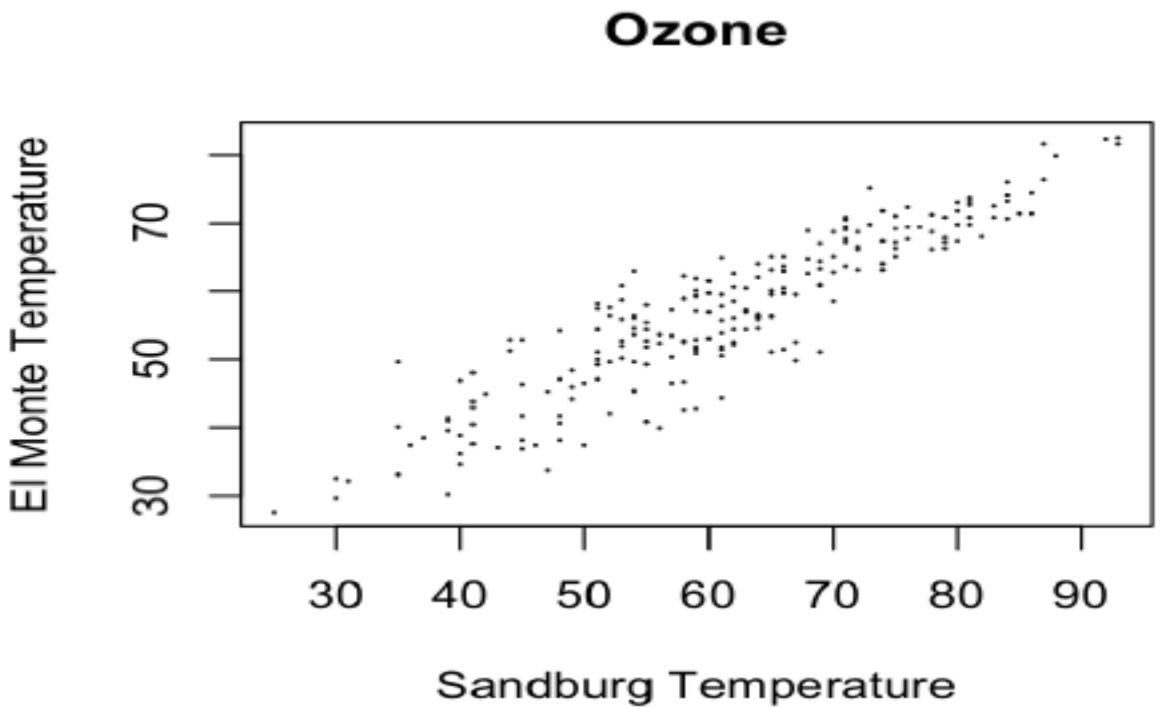


(b) pch='+'

그래프 옵션

- 점의 크기(cex)

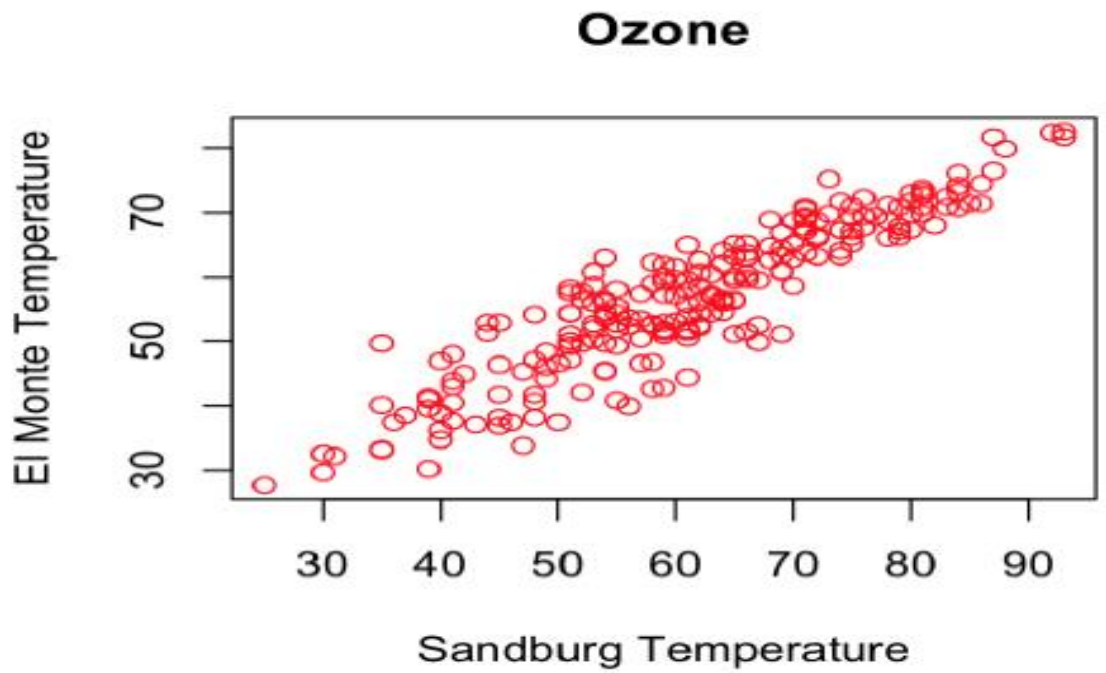
```
>   plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+         ylab="El Monte Temperature", main="Ozone", cex=.1)
```



그래프 옵션

- **색상(col)**
  - col="#FF0000" 또는 col="red
  - 목록은 colors()로 확인

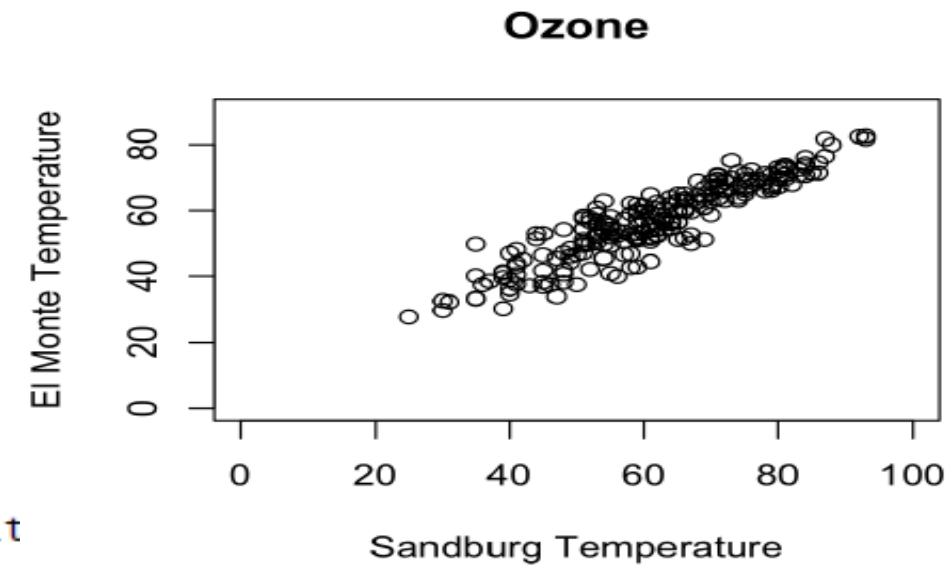
```
>  plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+       ylab="El Monte Temperature", main="Ozone", col="#FF0000")
```



그래프 옵션

- 좌표축 값의 범위(xlim, ylim)

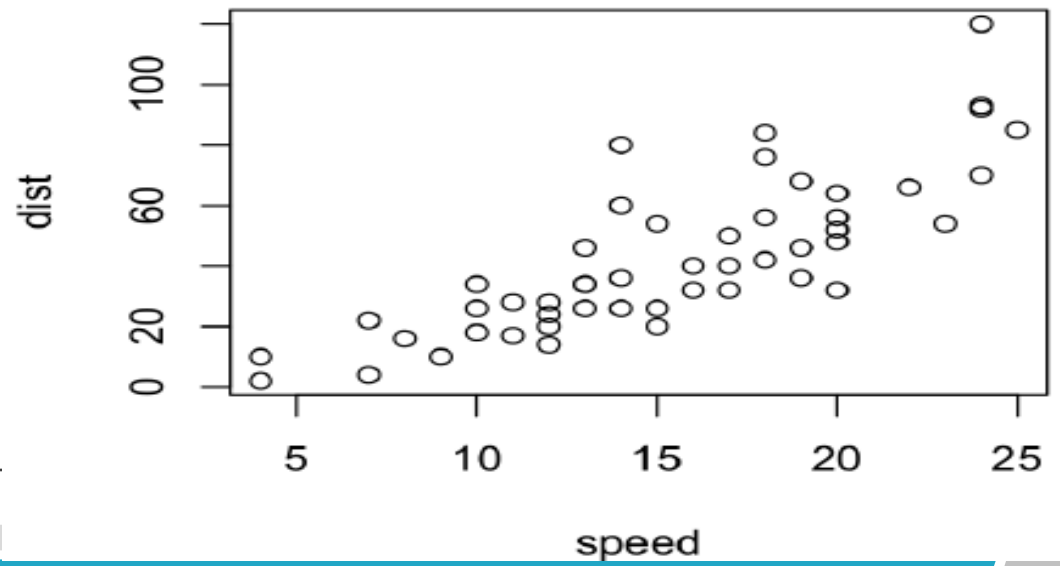
```
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+      ylab="El Monte Temperature", main="Ozone")
> max(Ozone$V8)
[1] NA
> max(Ozone$V8, na.rm=TRUE)
[1] 93
> max(Ozone$V9, na.rm=TRUE)
[1] 82.58
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperat
+      ylab="El Monte Temperature", main="Ozone",
+      xlim=c(0, 100), ylim=c(0, 90))
```



그래프 옵션

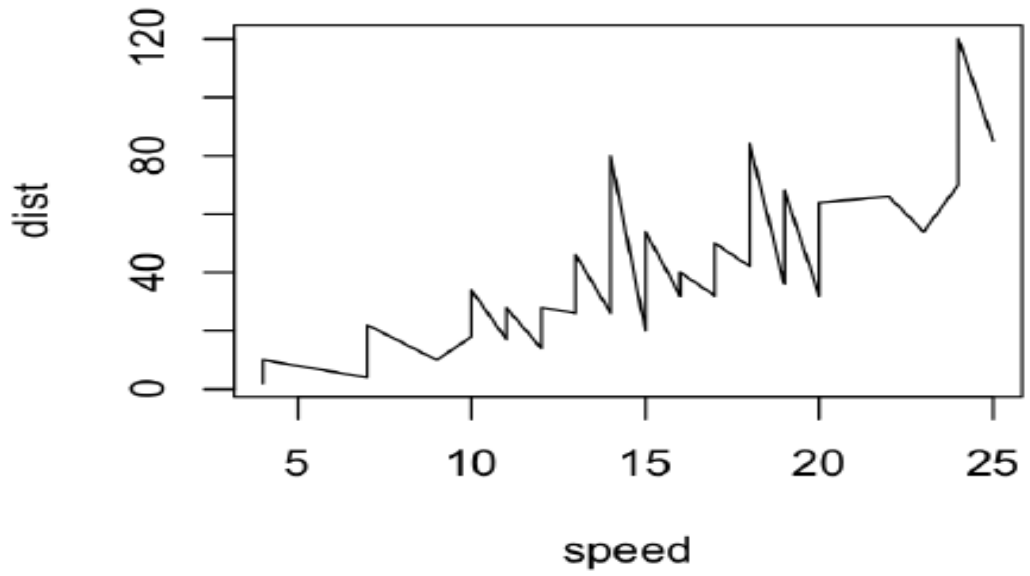
- **type**
  - 예) cars 데이터: 속도에서 브레이크를 잡았을 때 제동거리를 측정

```
> data(cars)
> str(cars)
'data.frame': 50 obs. of  2 variables:
 $ speed: num  4 4 7 7 8 9 10 10 10 11 ...
 $ dist : num  2 10 4 22 16 10 18 26 34 17 ...
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
> plot(cars)
```

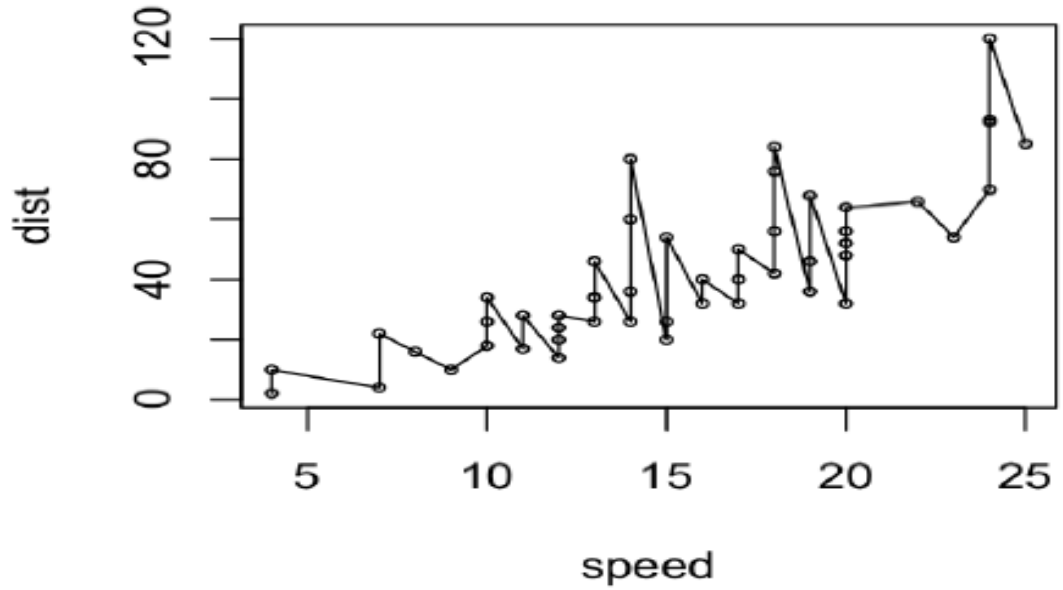


그래프 옵션

```
> plot(cars, type="l")
```

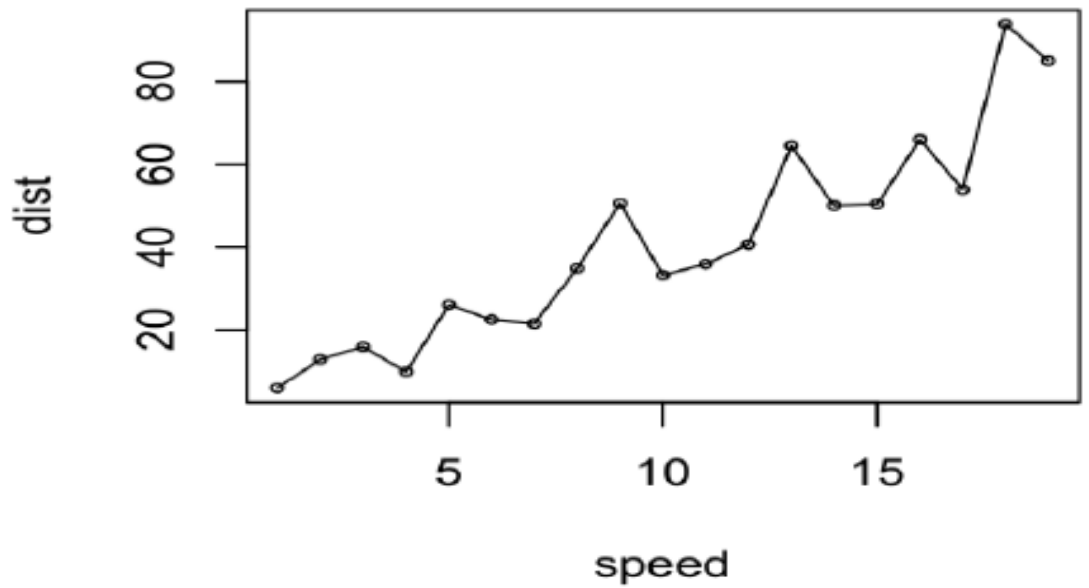



```
> plot(cars, type="o", cex=0.5)
```



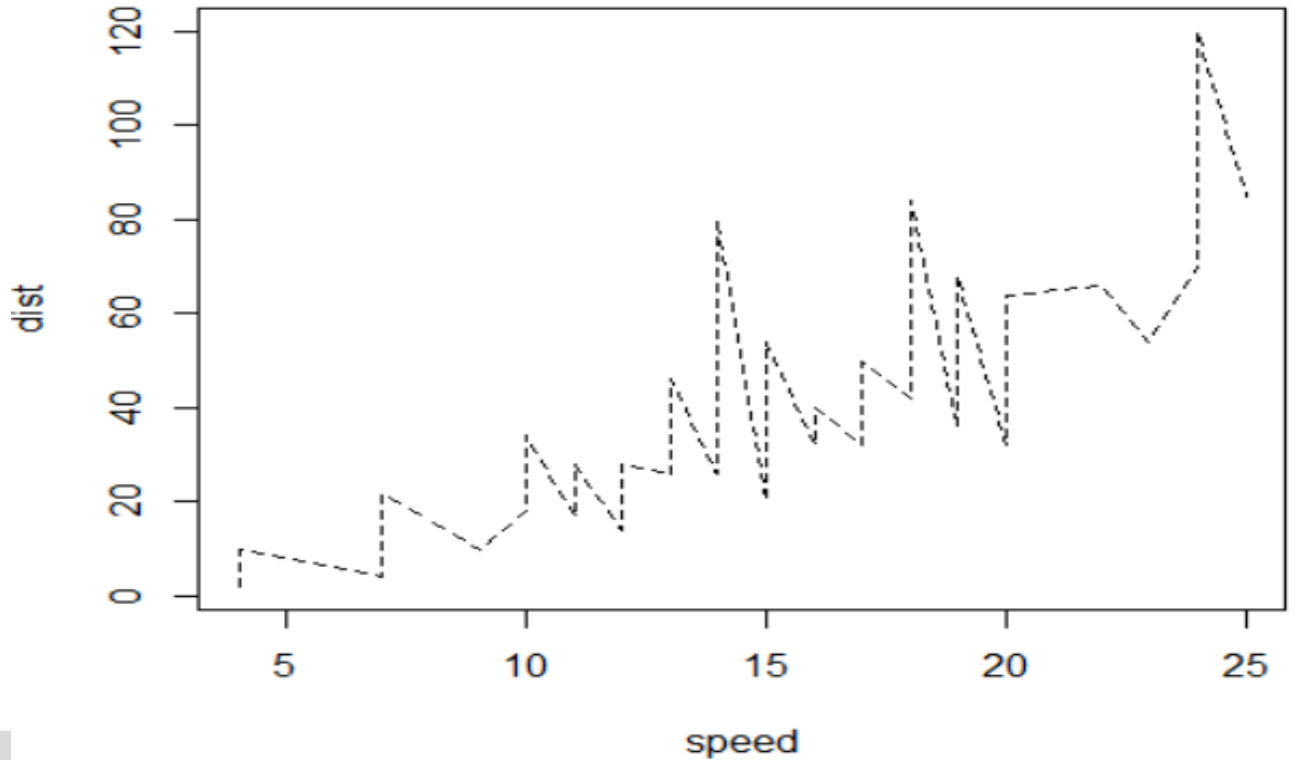
그래프 옵션

```
> tapply(cars$dist, cars$speed, mean)
       4         7         8         9        10        11        12
 6.00000  13.00000  16.00000  10.00000  26.00000  22.50000  21.50000
      13        14        15        16        17        18        19
35.00000  50.50000  33.33333  36.00000  40.66667  64.50000  50.00000
      20        22        23        24        25
50.40000  66.00000  54.00000  93.75000  85.00000
> plot(tapply(cars$dist, cars$speed, mean), type="o", cex=0.5,
+      xlab="speed", ylab="dist")
```



그래프 옵션

- 선유형(lty)
- 예) plot(card, type="l", lty="dashed")

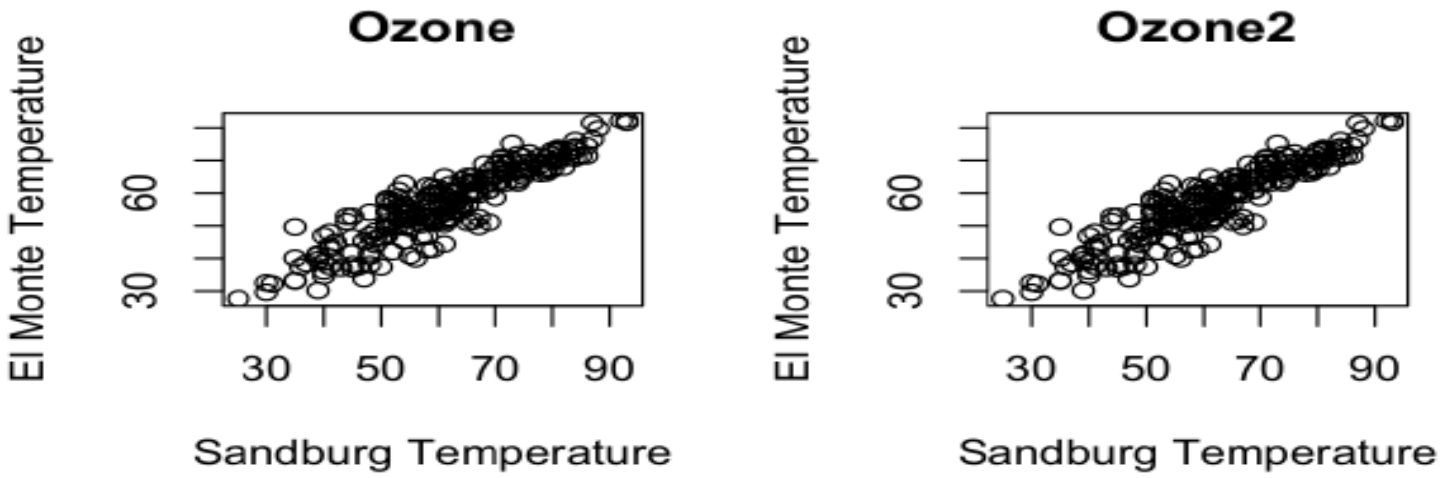


그래프 옵션(그래프의 배열(mfrow))

- **한 창에 여러 개의 그래프를 나열**
  - par(mfrow = c(nr, nc)) #행과 열의 개수 지정

```
> opar <- par(mfrow=c(1, 2))
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+      ylab="El Monte Temperature", main="Ozone")
> plot(Ozone$V8, Ozone$V9, xlab="Sandburg Temperature",
+      ylab="El Monte Temperature", main="Ozone2")
> par(opar)
```

Par를 호출하면 이전 설정을 반환함.
이를 기억했다가 나중에 돌려놈



그래픽 옵션

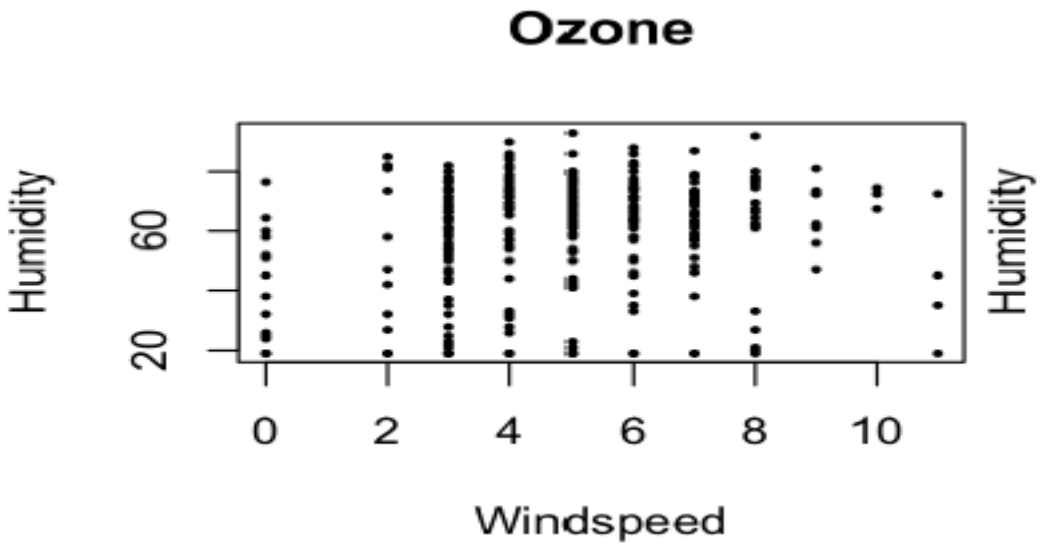
- **지터(jitter) – 중복 표현 방지를 위해 약간의 노이즈 추가**
  - 예) Ozone데이터의 V6와 V7은 각각 LAX에서의 풍속과 습도

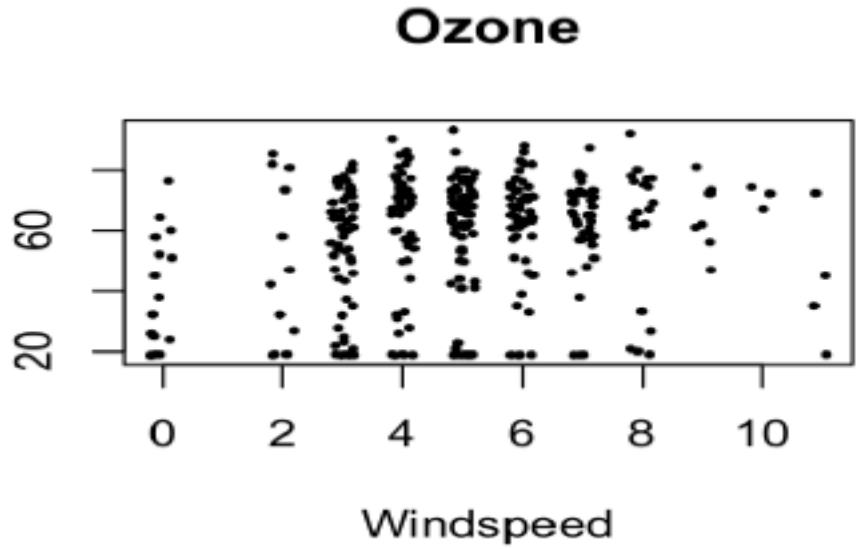
```
> head(Ozone)
  V1 V2 V3 V4   V5 V6 V7 V8    V9   V10 V11   V12 V13
1  1  1  4  3 5480  8 20 NA    NA  5000 -15 30.56 200
2  1  2  5  3 5660  6 NA 38    NA    NA -14    NA 300
3  1  3  6  3 5710  4 28 40    NA  2693 -25 47.66 250
4  1  4  7  5 5700  3 37 45    NA   590 -24 55.04 100
5  1  5  1  5 5760  3 51 54 45.32  1450  25 57.02  60
6  1  6  2  6 5720  4 69 35 49.64  1568  15 53.78  60
```

많은 중복

그래픽 옵션

```
> plot(Ozone$V6, Ozone$V7, xlab="Windspeed", ylab="Humidity",
+       main="Ozone", pch=20, cex=.5)
> plot(jitter(Ozone$V6), jitter(Ozone$V7),
+       xlab="Windspeed", ylab="Humidity", main="Ozone",
+       pch=20, cex=.5)
```

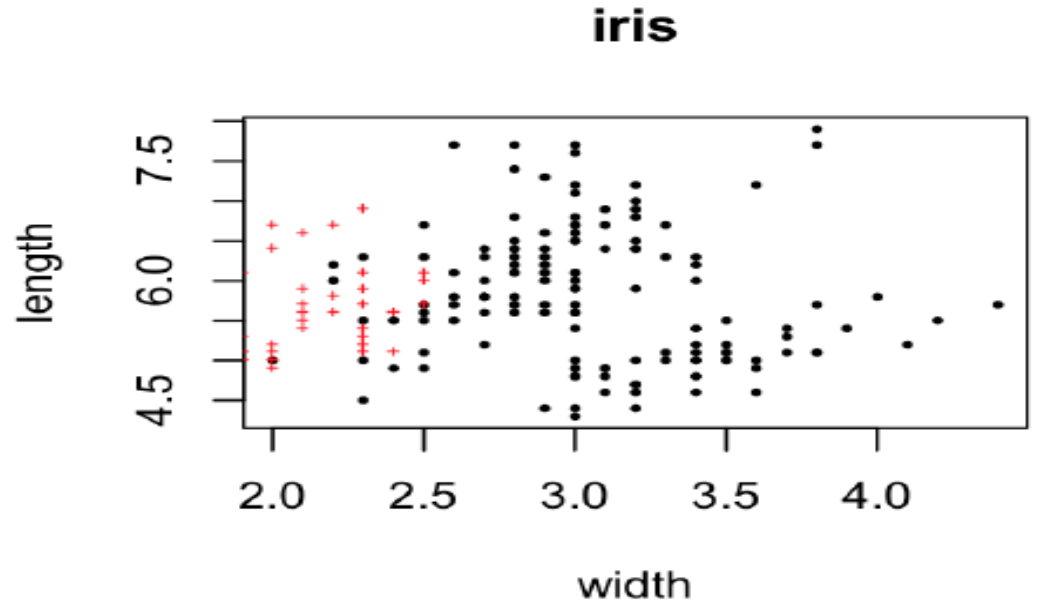




기본 그래프

- **점(points)**
  - point() : 이미 생성된 plot에 점을 추가

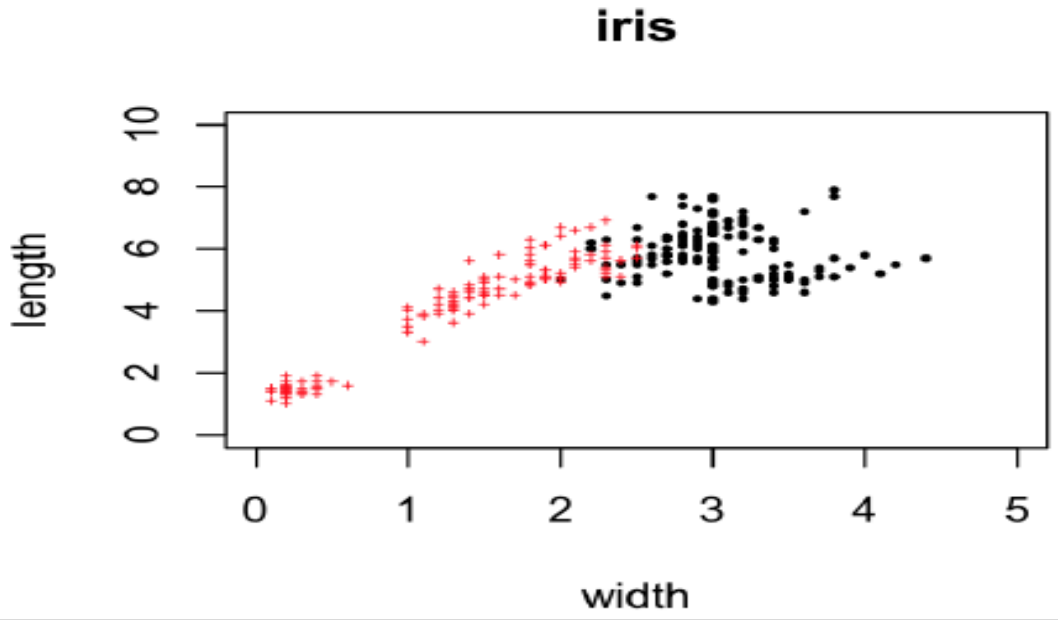
```
>   plot(iris$Sepal.Width, iris$Sepal.Length, cex=.5, pch=20,
+        xlab="width", ylab="length", main="iris")
>   points(iris$Petal.Width, iris$Petal.Length, cex=.5,
+          pch="+", col="#FF0000")
```



기본 그래프

- 빈 그래프 먼저 그리고 나중에 추가하기 (type = n)

```
> with(iris, {
+   plot(NULL, xlim=c(0, 5), ylim=c(0, 10),
+        xlab="width", ylab="length", main="iris", type="n")
+   points(Sepal.Width, Sepal.Length, cex=.5, pch=20)
+   points(Petal.Width, Petal.Length, cex=.5, pch="+", col="#FF0000")
+ })
```

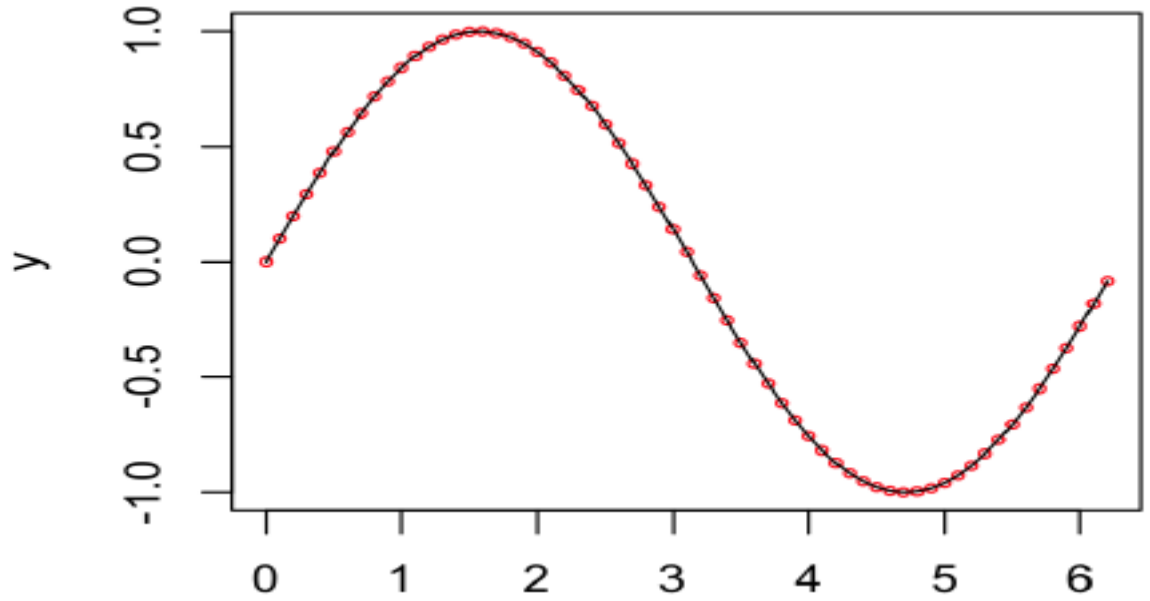


xlim과 ylim을 생략하면 초기 빈 그래프가 만들어지지 않음

기본 그래프

- **선(lines)**
  - 예) [0, 2π]까지 sin 그래프

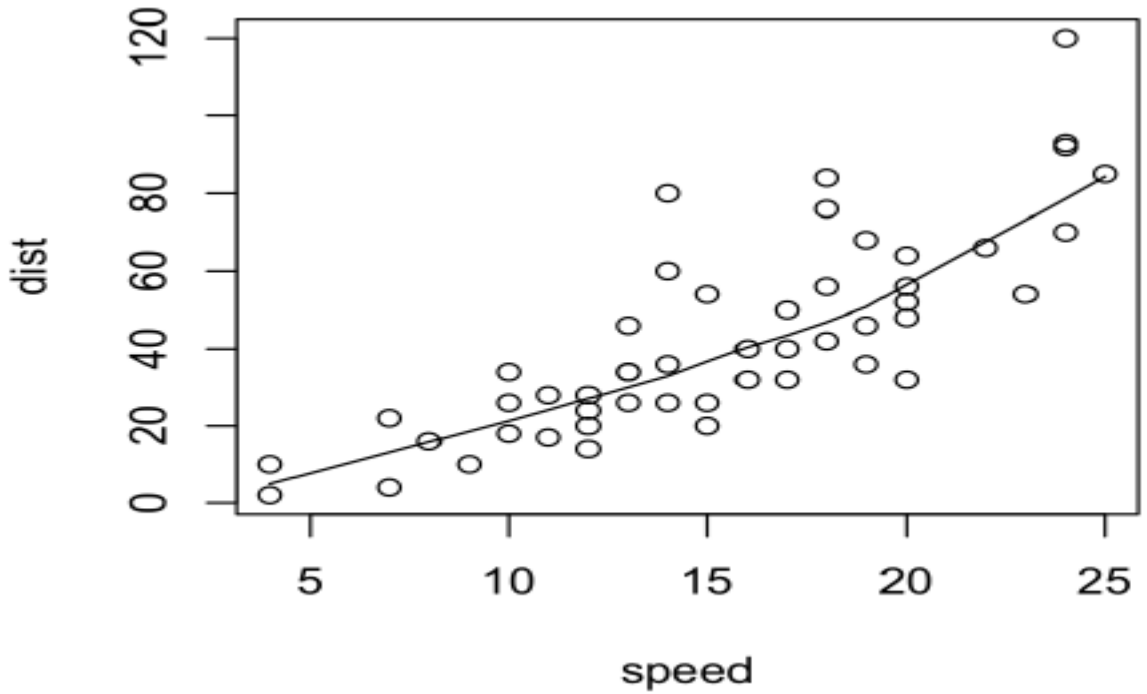
```
> x <- seq(0, 2*pi, 0.1)
> y <- sin(x)
> plot(x, y, cex=.5, col="red")
> lines(x, y)
```



기본 그래프

- 회귀 분석(추세선 구하기)

```
> plot(cars)
> lines(lowess(cars))
```



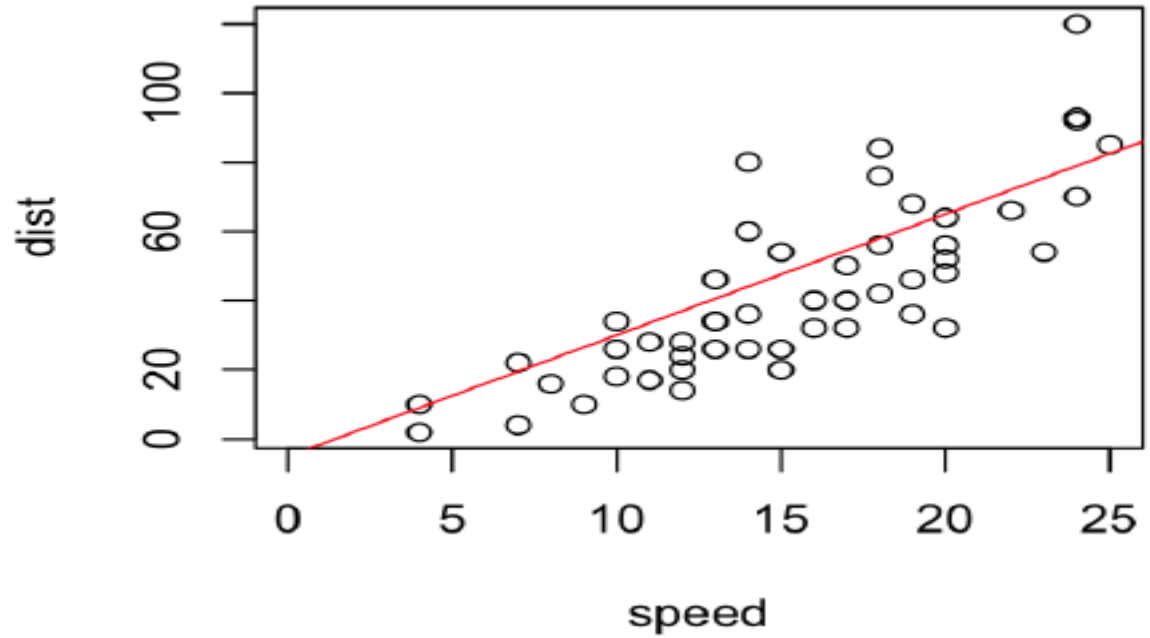

```
> lowess(cars)
$x
 [1]  4  4  7  7  8  9 10 10 10 11 11 12 12 12 12 13 13 13 13 14 14 14 1
4
[24] 15 15 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 2
4
[47] 24 24 24 25

$y
 [1]  4.965459  4.965459 13.124495 13.124495 15.858633 18.579691 21.2803
13
 3 21.280313 24.129277 24.129277 27.119549 27.119549 27.1195
 9 30.027276 30.027276 30.027276 30.027276 32.962506 32.9625
 6 32.962506 36.757728 36.757728 36.757728 40.435075 40.4350
```

회귀분석(x에 대항하는 y값 추정)
plot(lowess(cars)) 실행해볼 것

기본 그래프

- **직선(abline)**
  - y = a + bx, y = h, x = v 형태의 직성 그리기
  - 예) cars 데이터에서 dist = −5 + 3.5 × speed와 같은 직선 그리기

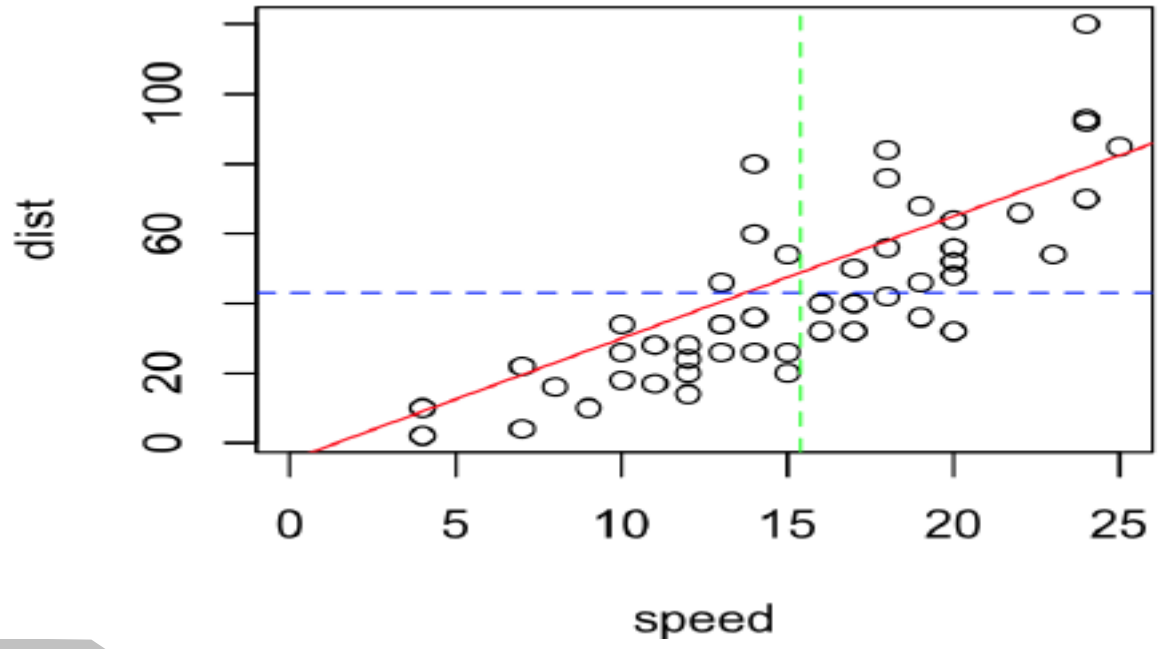
```
> plot(cars, xlim=c(0, 25))
> abline(a=-5, b=3.5, col="red")
```



기본 그래프

- 그래프에 speed와 dist의 평균 직선 그리기

```
> plot(cars, xlim=c(0, 25))
> abline(a=-5, b=3.5, col="red")
> abline(h=mean(cars$dist), lty=2, col="blue")
> abline(v=mean(cars$speed), lty=2, col="green")
```

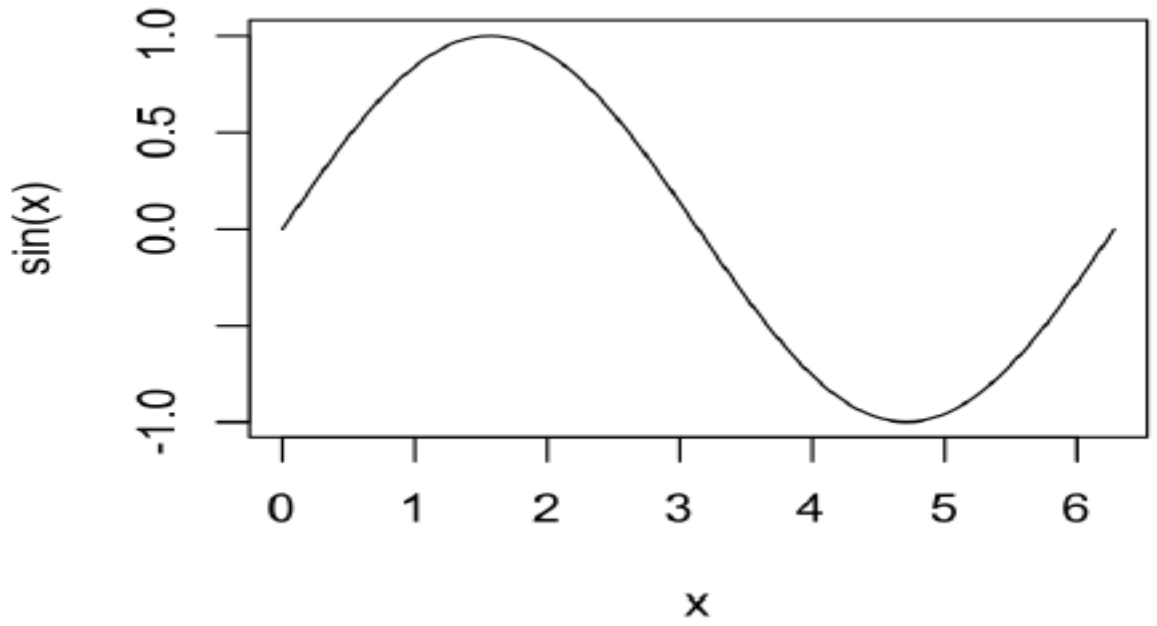


a, b, h, v는 고정

기본 그래프

- **곡선(curve)**
  - curve(expr, from, to)

```
> curve(sin, 0, 2*pi)
```

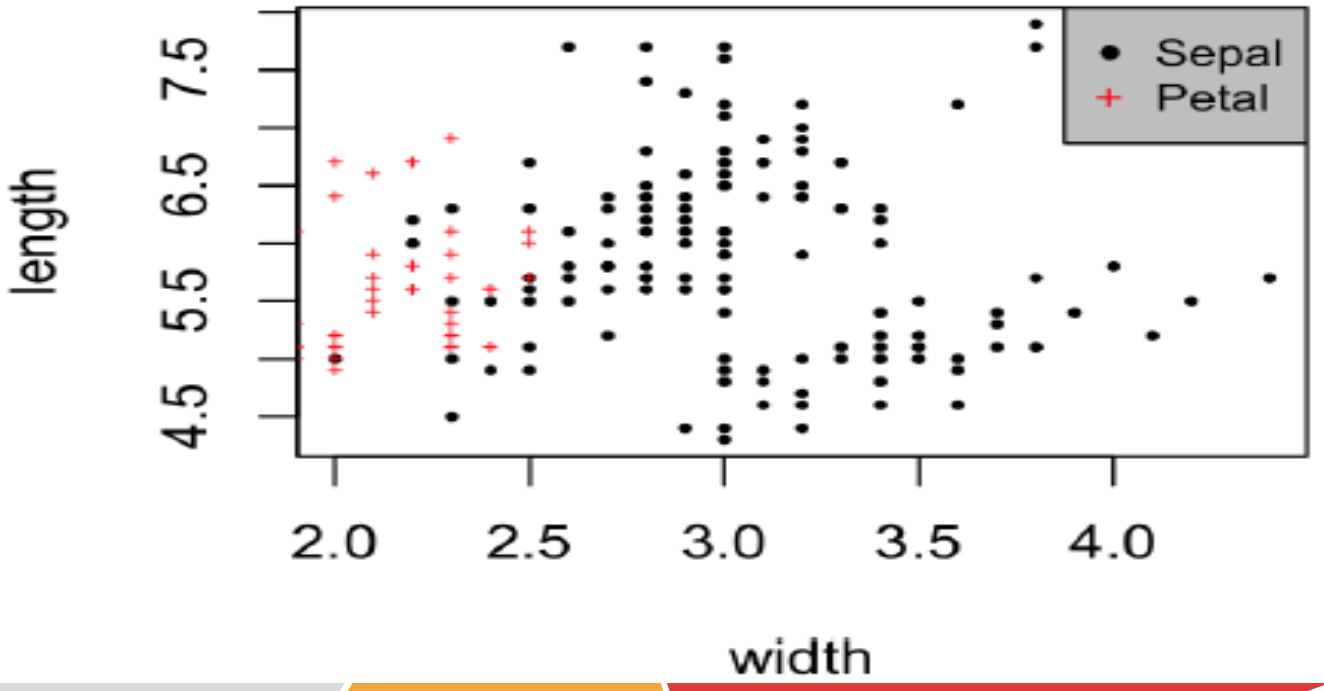


범례(legend)

```
> plot(iris$Sepal.Width, iris$Sepal.Length, cex=.5, pch=20,
+ xlab="width", ylab="length")
> points(iris$Petal.Width, iris$Petal.Length, cex=.5,
+ pch="+", col="#FF0000")
> legend("topright", legend=c("Sepal", "Petal"),
+ pch=c(20, 43), cex=.8, col=c("black", "red"), bg="gray")
```

벡터형태이므로
자료형을 통일

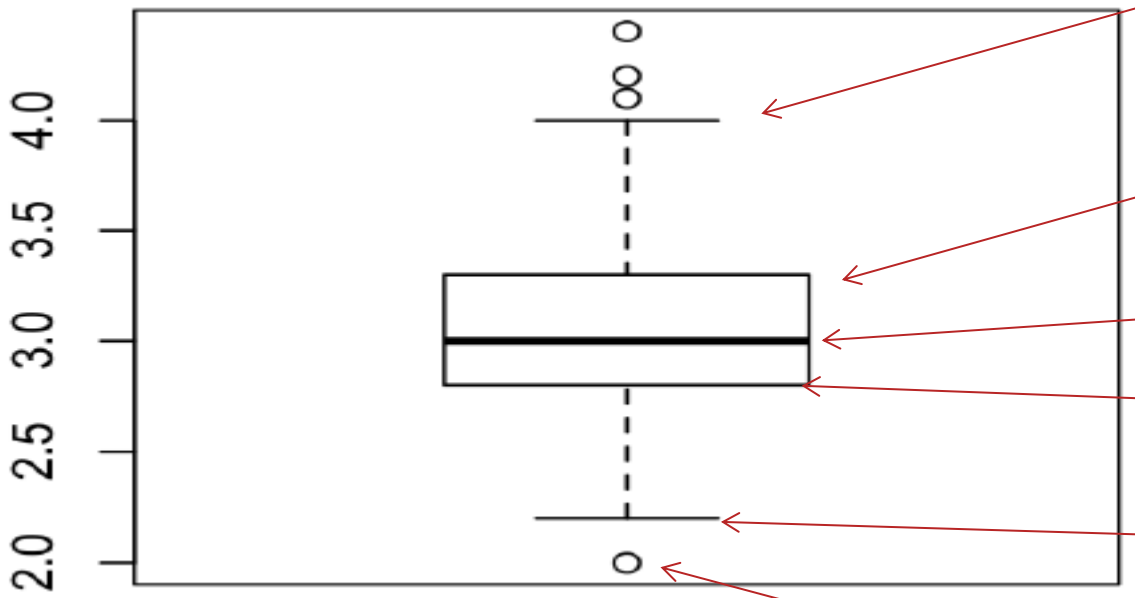
x, y좌표를 지정해도 됨



상자 그림(boxplot)

- 제1사분위수, 중앙값, 제3사분위수를 보여준다.

```
> boxplot(iris$Sepal.Width)
```



upper whisker
중앙값 + 1.5 * IQR 보다 작은 데이터 중 가장 큰 값(IQR = 제3사분위수 - 제1사분위수)

제3사분위수

중앙값

제1사분위수

lower whisker
중앙값 + 1.5 * IQR 보다 작은 데이터 중 가장 큰 값(IQR = 제3사분위수 - 제1사분위수)

Outlier(이상치)

상자 그림(boxplot)

```
> boxstats <- boxplot(iris$Sepal.Width)
> boxstats
$stats
     [,1]
[1,]  2.2
[2,]  2.8
[3,]  3.0
[4,]  3.3
[5,]  4.0

$n
[1] 150

$conf
          [,1]
[1,] 2.935497
[2,] 3.064503

$out
[1] 4.4 4.1 4.2 2.0

$group
[1] 1 1 1 1

$names
[1] "1"
```

신뢰구간

상자 그림(boxplot)

- iris의 setosa종과 versicolor종의 Sepal.Width에 대한 상자 그림을 그린 뒤 이 두 종의 중앙값이 다른지 비교

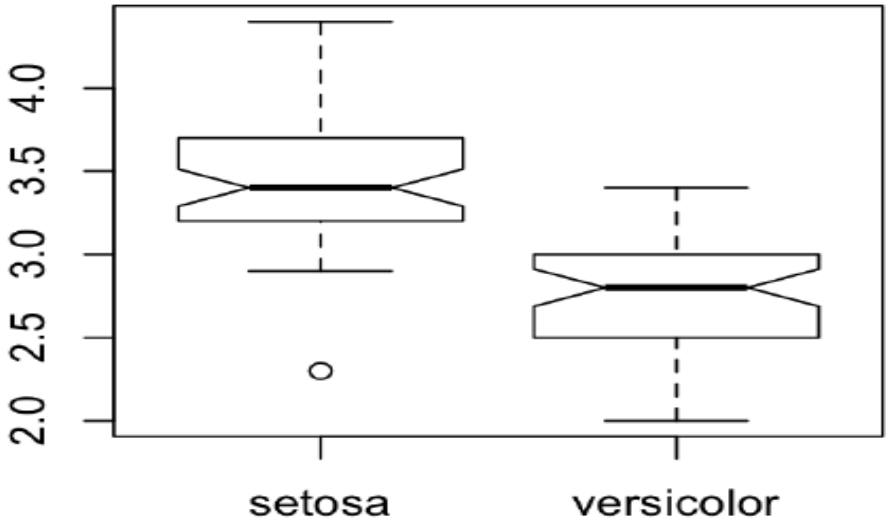
```
> sv <- subset(iris, Species=="setosa" | Species=="versicolor")
> sv$Species <- factor(sv$Species)
> boxplot(Sepal.Width ~ Species, data=sv, notch=TRUE)
```

||가 아니라|
(벡터간 비교)

원래 factor형 이지만
3개중 setosa와
versicolor만
남겨놓기 위해 다시 지정

Sepal.Width를 Species 마다 그리기

신뢰구간이 오목하게 그려짐

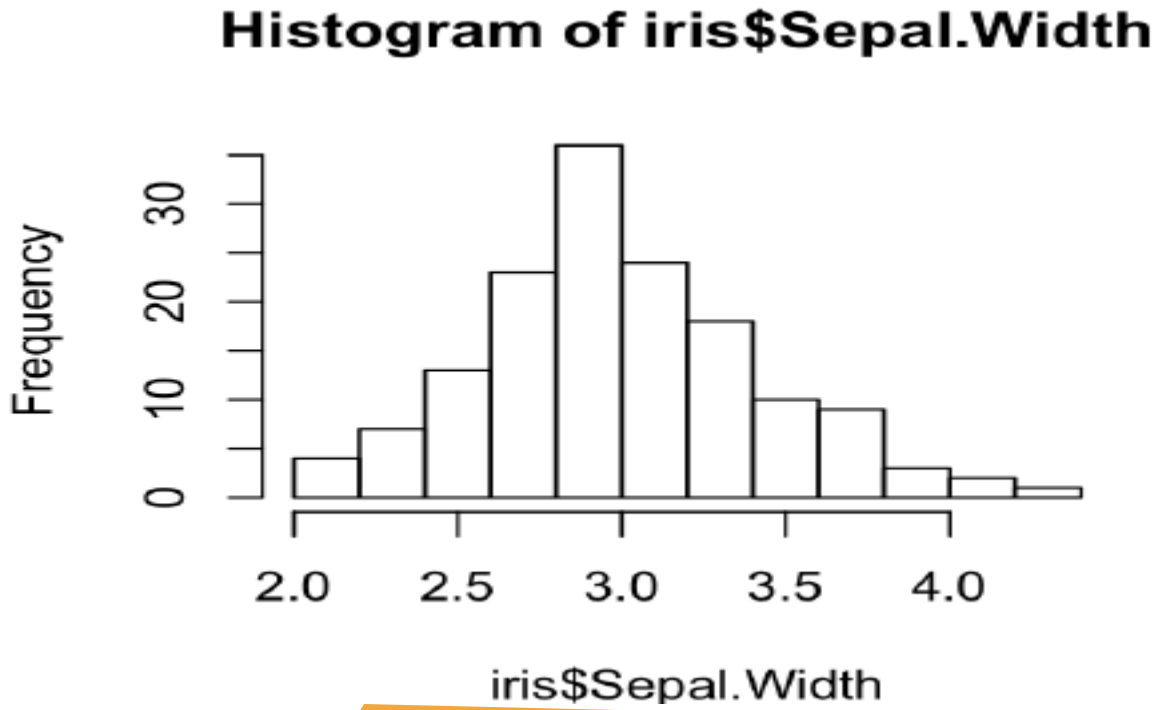


setosa와 versicolor의 신뢰구간이 겹치지
않으므로 이 두 그룹의
중앙값은 서로 다르다

히스토그램(hist)

```
hist(
    x,                      # 벡터 데이터
    breaks="Struge",  #막대 너비. 기본값은 Struge로  n개일 때 [log2(n) + 1]
                            #또는 데이터를 나눌 구분 값이 저장된 벡터 또는 함수
    freq=NULL,              # 기본은 빈도수, FALSE면 확률밀도
)
```

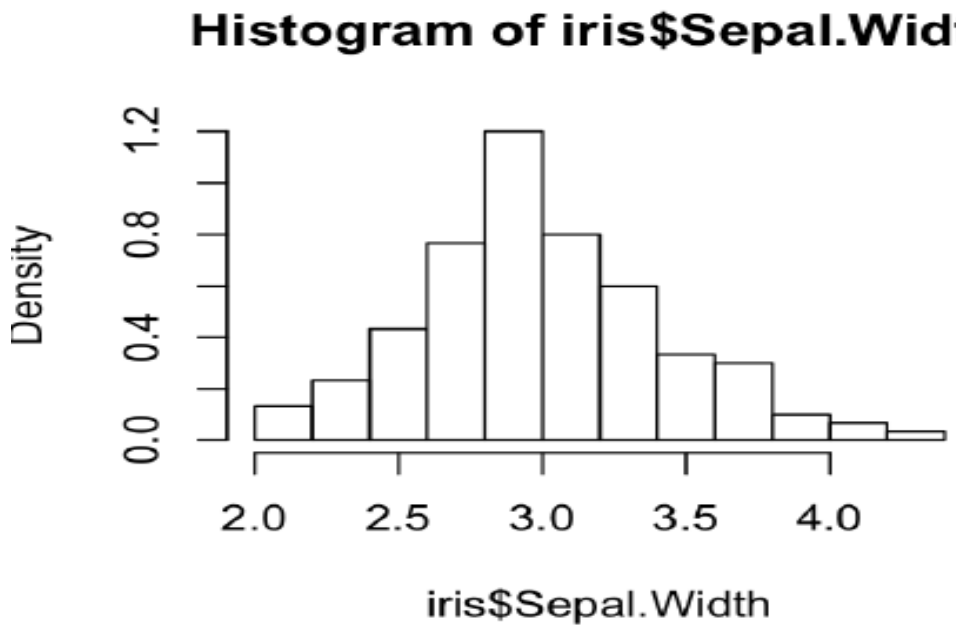
```
> hist(iris$Sepal.Width)
```



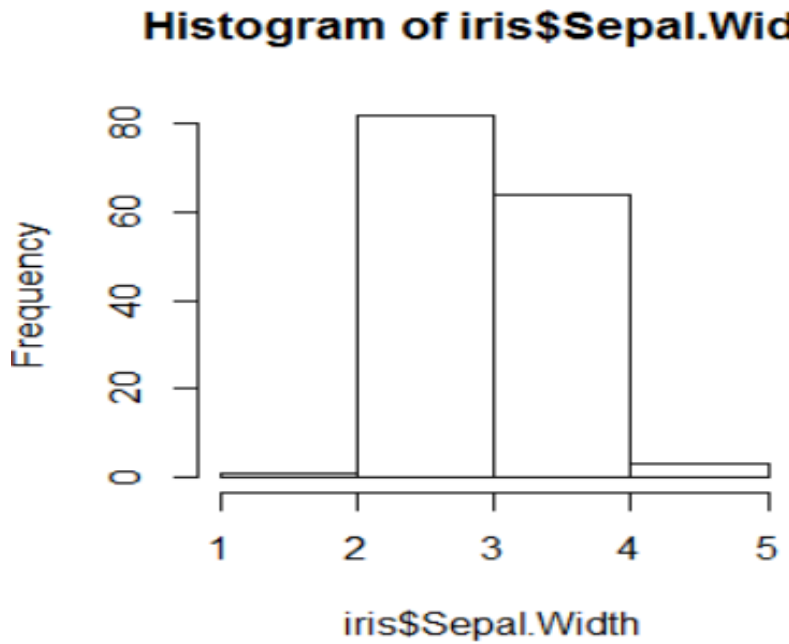
히스토그램(hist)

```
> hist(iris$Sepal.Width, freq=FALSE)
```

넓이가 1인 확률밀도




```
> hist(iris$Sepal.Width, breaks=c(1, 2, 3, 4, 5))
```



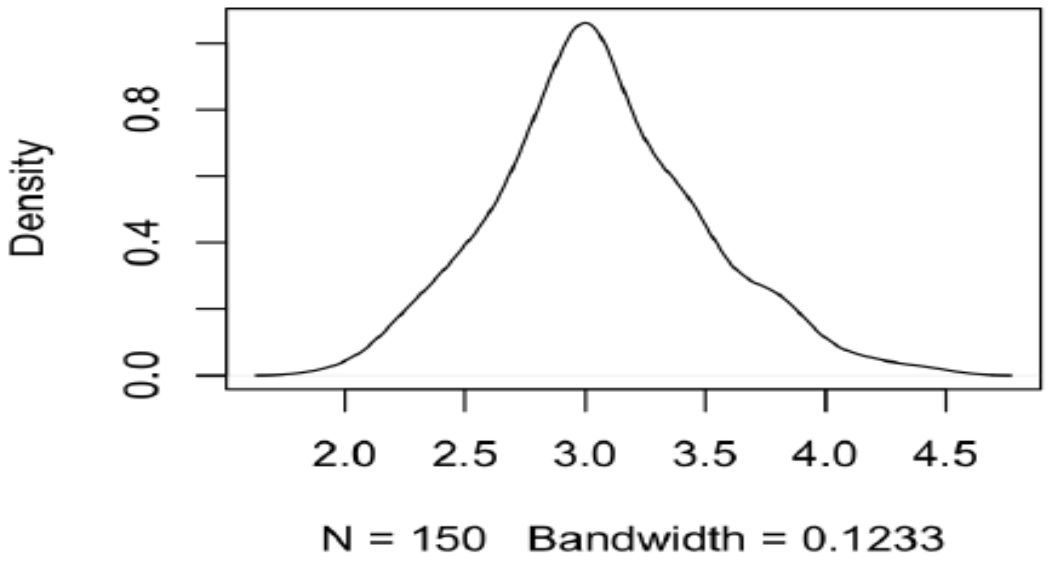
히스토그램(hist)

```
> x <- hist(iris$Sepal.Width, freq=FALSE)
> x
$breaks
 [1] 2.0 2.2 2.4 2.6 2.8 3.0 3.2 3.4 3.6 3.8 4.0 4.2 4.4

$counts
 [1]  4  7 13 23 36 24 18 10  9  3  2  1

$intensities
 [1] 0.13333333 0.23333333 0.43333333 0.76666667 1.20000000
 [6] 0.80000000 0.60000000 0.33333333 0.30000000 0.10000000
[11] 0.06666667 0.03333333

$density
 [1] 0.13333333 0.23333333 0.43333333 0.76666667 1.20000000
 [6] 0.80000000 0.60000000 0.33333333 0.30000000 0.10000000
[11] 0.06666667 0.03333333
```

밀도 그림(density)

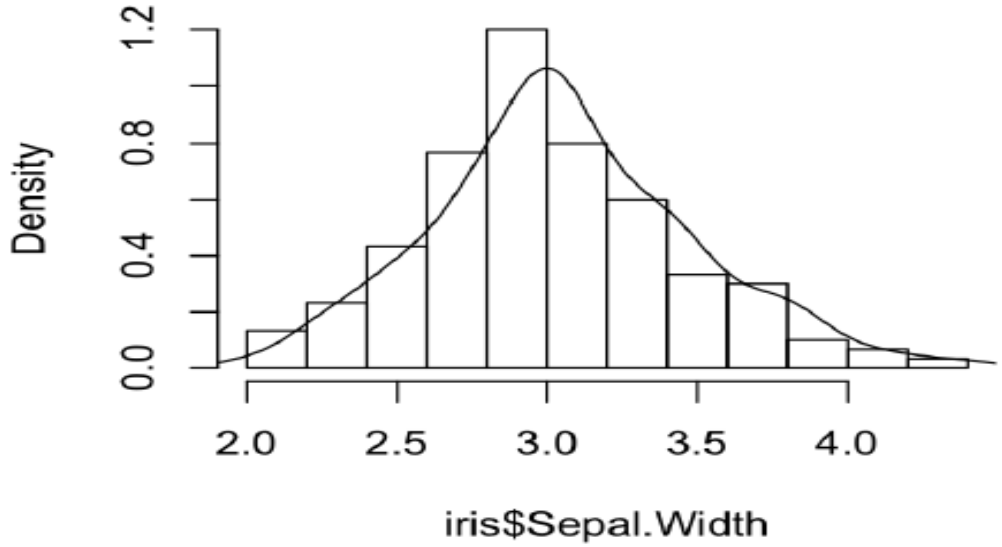
- 막대의 너비를 가정하지 않음
- 모든 점에서 밀도를 추정: 커널밀도추정방식

```
> plot(density(iris$Sepal.Width))
```



밀도그림과 히스토그램을 동시 표출

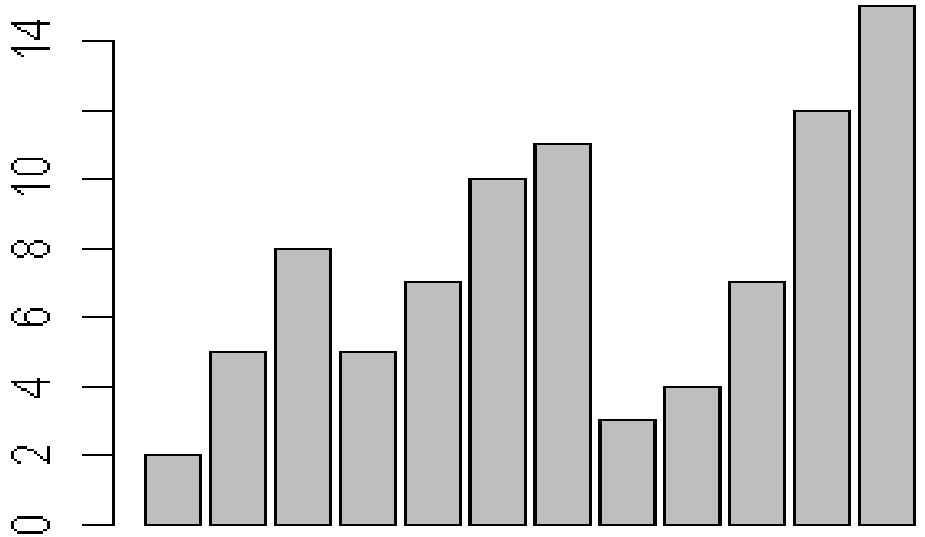
```
> hist(iris$Sepal.Width, freq=FALSE)
> lines(density(iris$Sepal.Width))
```



막대 그래프(barplot)

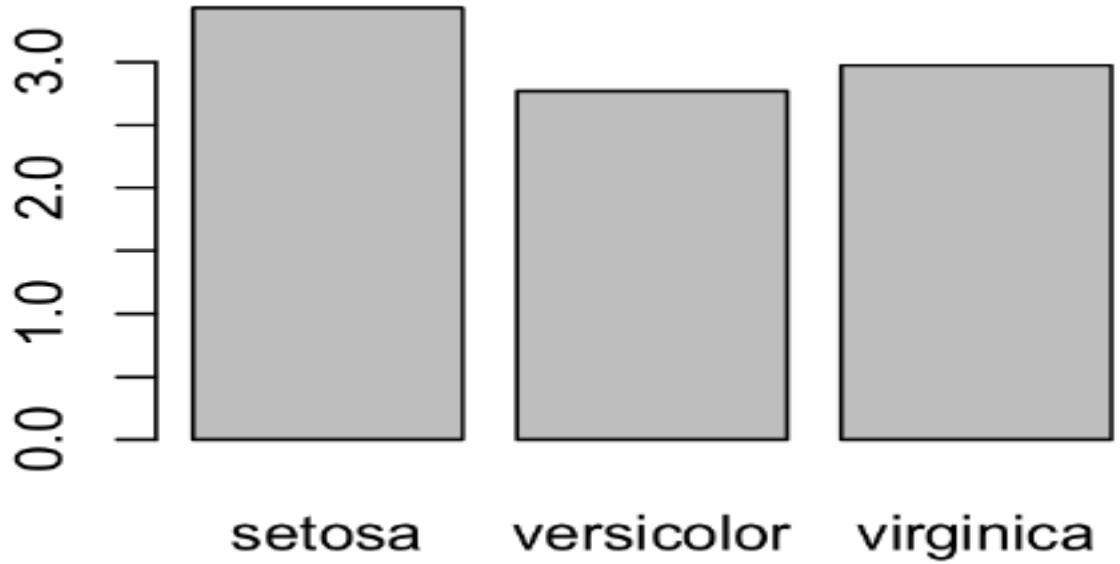
- 예)

```
> x <- c(2,5,8,5,7,10,11,3,4,7,12,15)
> barplot(x)
```



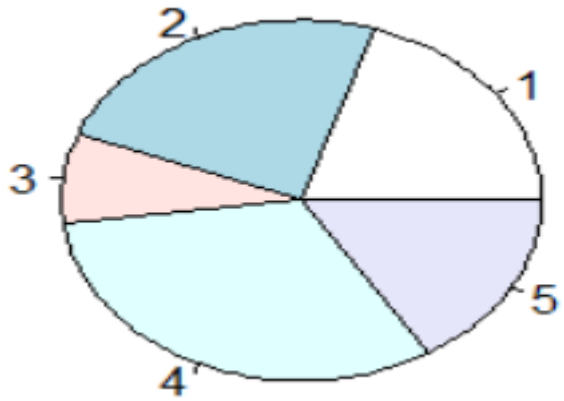
막대 그래프(barplot)

```
> barplot(tapply(iris$Sepal.Width, iris$Species, mean))
```

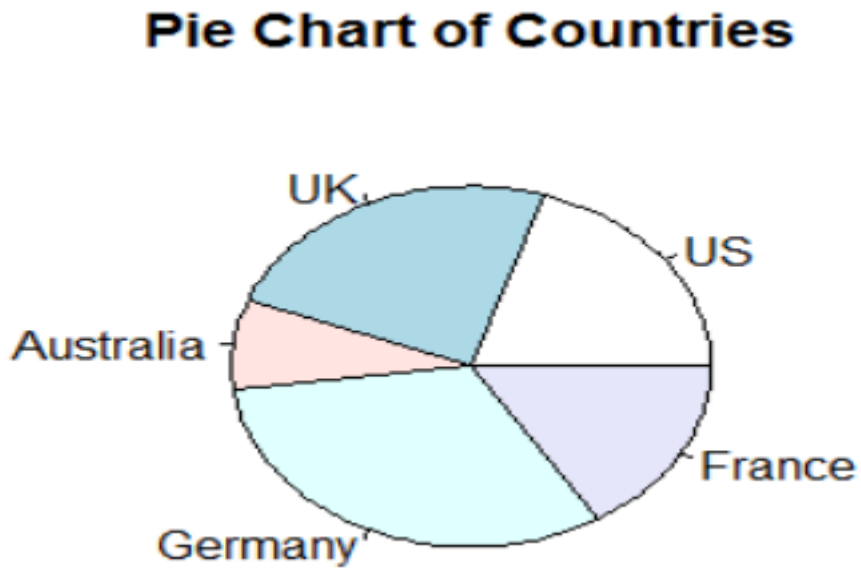


파이(pie) 그래프

```
> slices <- c(10, 12,4, 16, 8)
> pie(slices)
```




```
> slices <- c(10, 12,4, 16, 8)
> lbls <- c("US", "UK", "Australia", "Germany", "France")
> pie(slices, labels = lbls, main="Pie Chart of Countries")
```



모자이크 플롯(mosaicplot)

- **범주형 다변량 데이터를 표현(table형태의 분할표를 표현)**
- **분할표**
  - 명목형, 순서형 데이터의 돗수 표현
  - 예)

	테스트-양성	테스트-음성
실제 –양성	92	5 (위음성)
실제- 음성	7 (위양성)	42

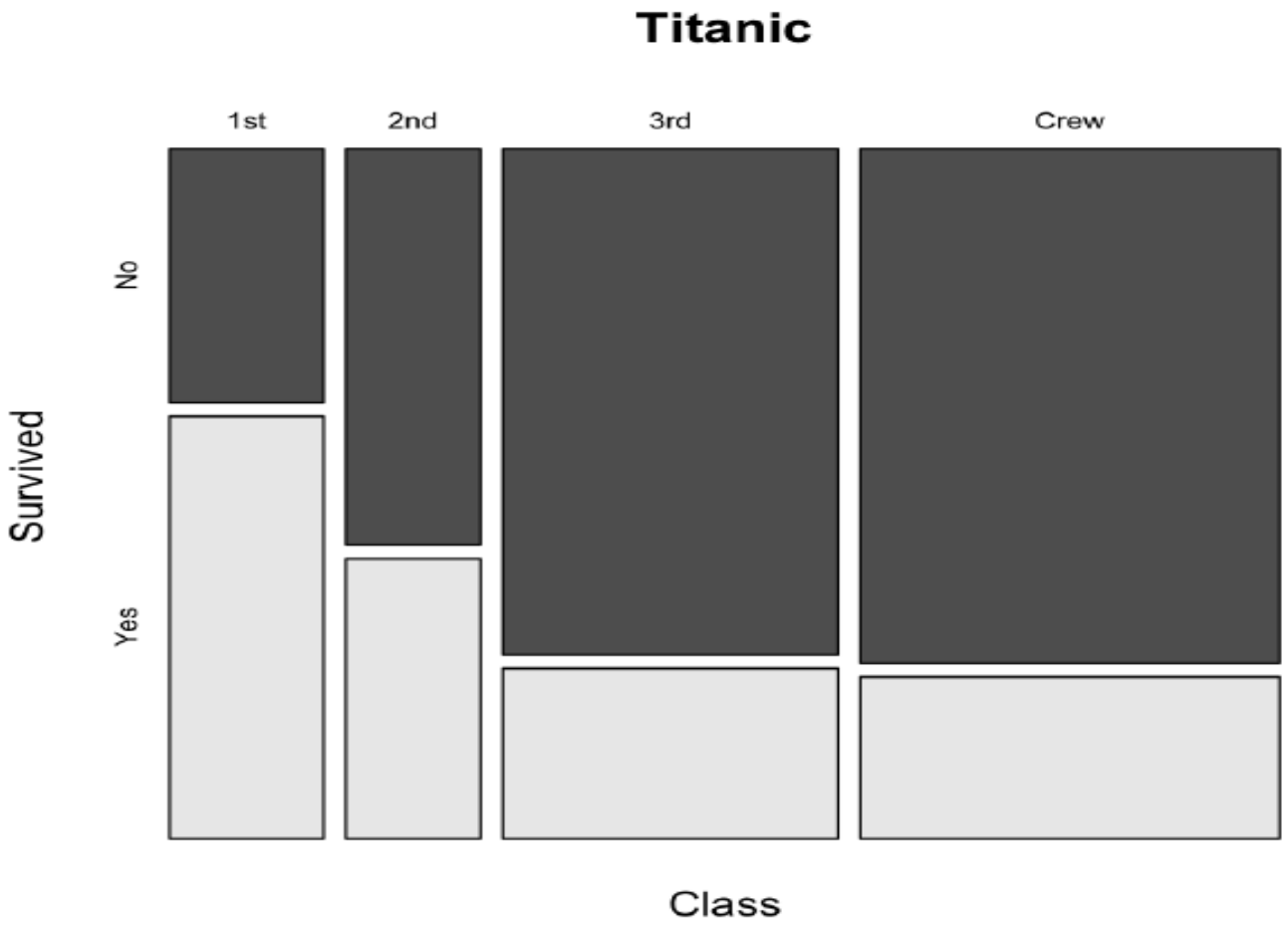
  - 예) Titanic data

```
> str(Titanic)
 table [1:4, 1:2, 1:2, 1:2] 0 0 35 0 0 0 17 0 118 154 ...
 - attr(*, "dimnames")=List of 4
  ..$ Class    : chr [1:4] "1st" "2nd" "3rd" "Crew"
  ..$ Sex      : chr [1:2] "Male" "Female"
  ..$ Age      : chr [1:2] "Child" "Adult"
  ..$ Survived: chr [1:2] "No" "Yes"
```

```
> Titanic
, , Age = Child, Survived = No

      Sex
Class  Male Female
  1st     0      0
  2nd     0      0
  3rd    35     17
  Crew    0      0

, , Age = Adult, Survived = No

      Sex
Class  Male Female
  1st   118      4
  2nd   154     13
  3rd   387     89
  Crew  670      3

, , Age = Child, Survived = Yes
```

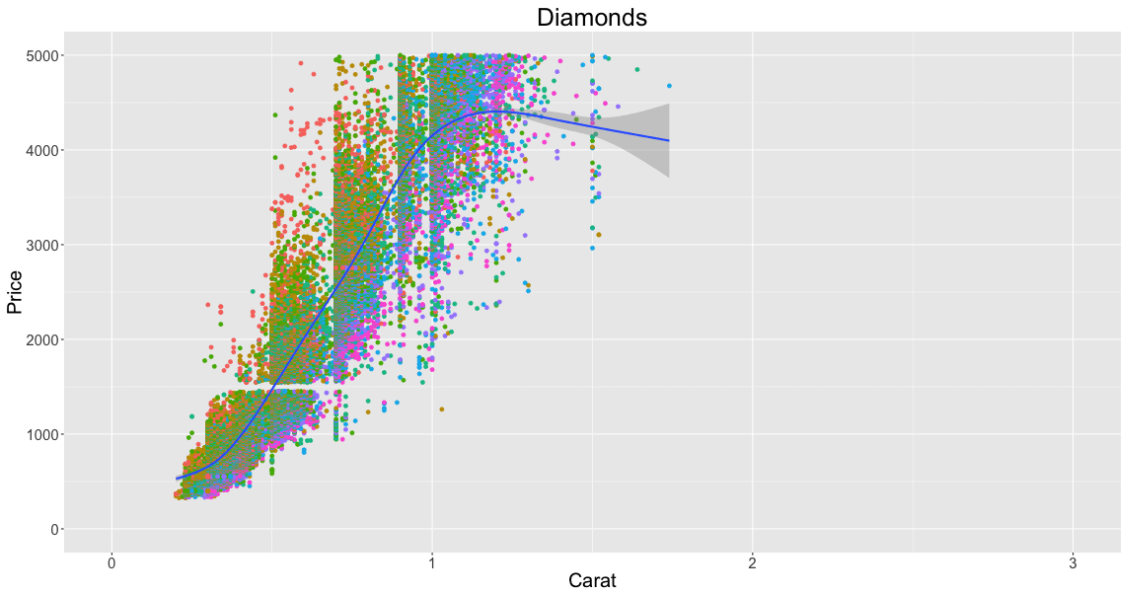
모자이크 플롯(mosaicplot)

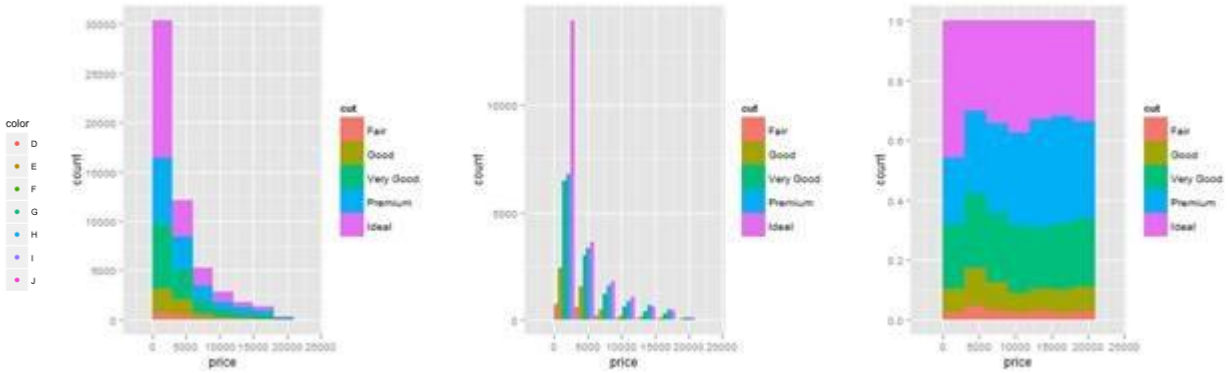
```
> mosaicplot(~ Class + Survived, data=Titanic, color=TRUE)
```



ggplot2 패키지

- **Elegant Graphics Tool for Data Analysis**





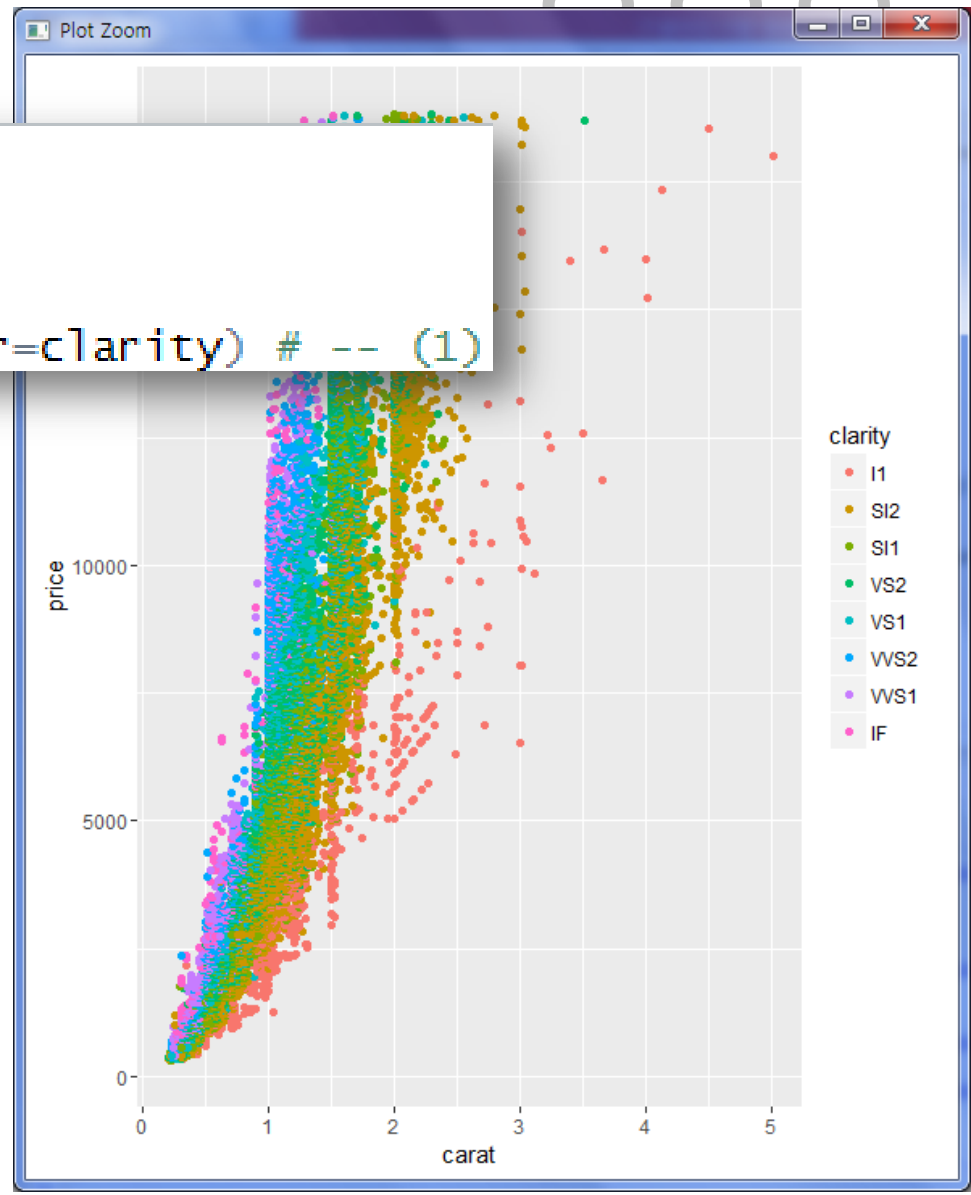
- ggplot2 도움말 사이트: http://docs.ggplot2.org/current/

qplot() 함수

```
install.packages("ggplot2")
library(ggplot2)
qplot(diamonds$carat, diamonds$price)
qplot(carat, price, data = diamonds)
qplot(carat, price, data = diamonds, geom="point" , colour=clarity) # -- (1)
```

x축 y축

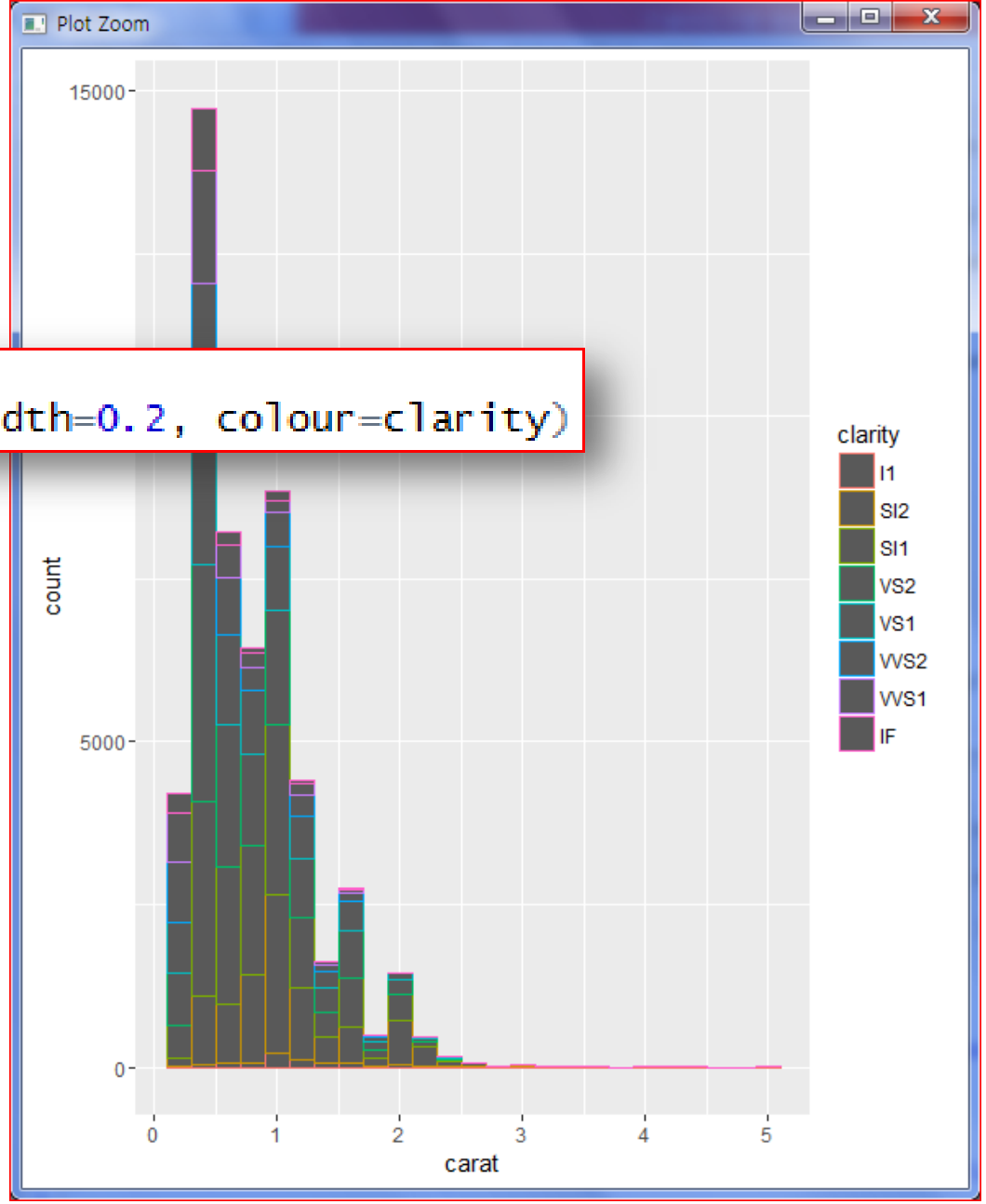
```
> str(diamonds)
Classes 'tbl_df', 'tbl' and 'data.frame':       53940 obs. of  10 variables:
 $ carat   : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
 $ cut     : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
 $ color   : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
 $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
 $ depth   : num  61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
 $ table   : num  55 61 65 58 58 57 57 55 61 61 ...
 $ price   : int  326 326 327 334 335 336 336 337 337 338 ...
 $ x       : num  3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
 $ y       : num  3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
 $ z       : num  2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```



Clarity가 좋을 수록,
Carat에 대한 price가 크게 반응함

qplot() 함수

```
qplot(carat, data = diamonds,geom="histogram")
qplot(carat, data = diamonds,geom="histogram", binwidth=0.2, colour=clarity)
```



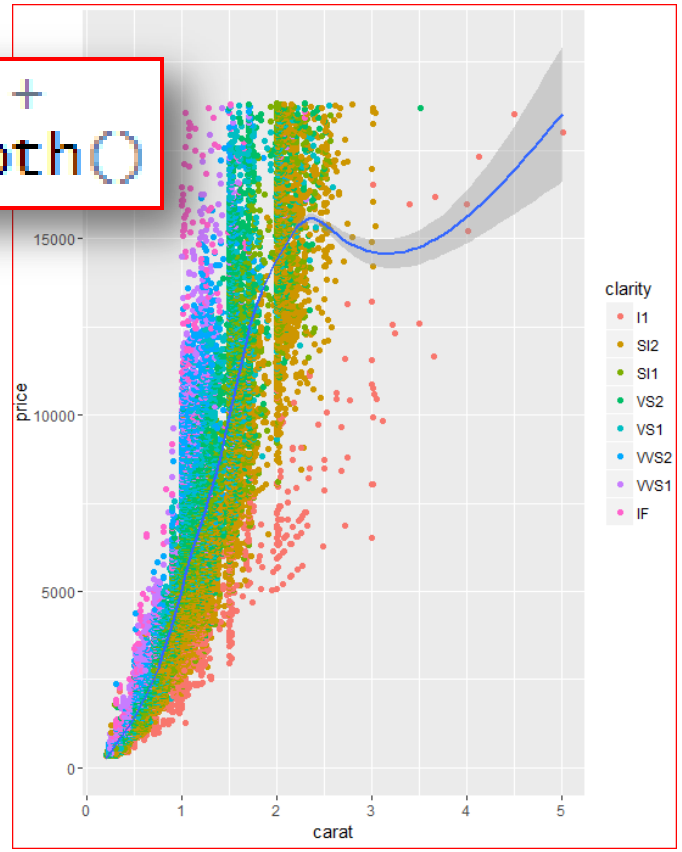
ggplot() 함수

aesthetic mapping

x축에 length 컬럼, y축에 width 컬럼을 매핑

- **미적매핑 이후에 필요한 레이어 계속 추가**

```
ggplot(data=diamonds, aes(x=carat, y=price)) +
  geom_point(aes(colour=clarity)) + geom_smooth()
```



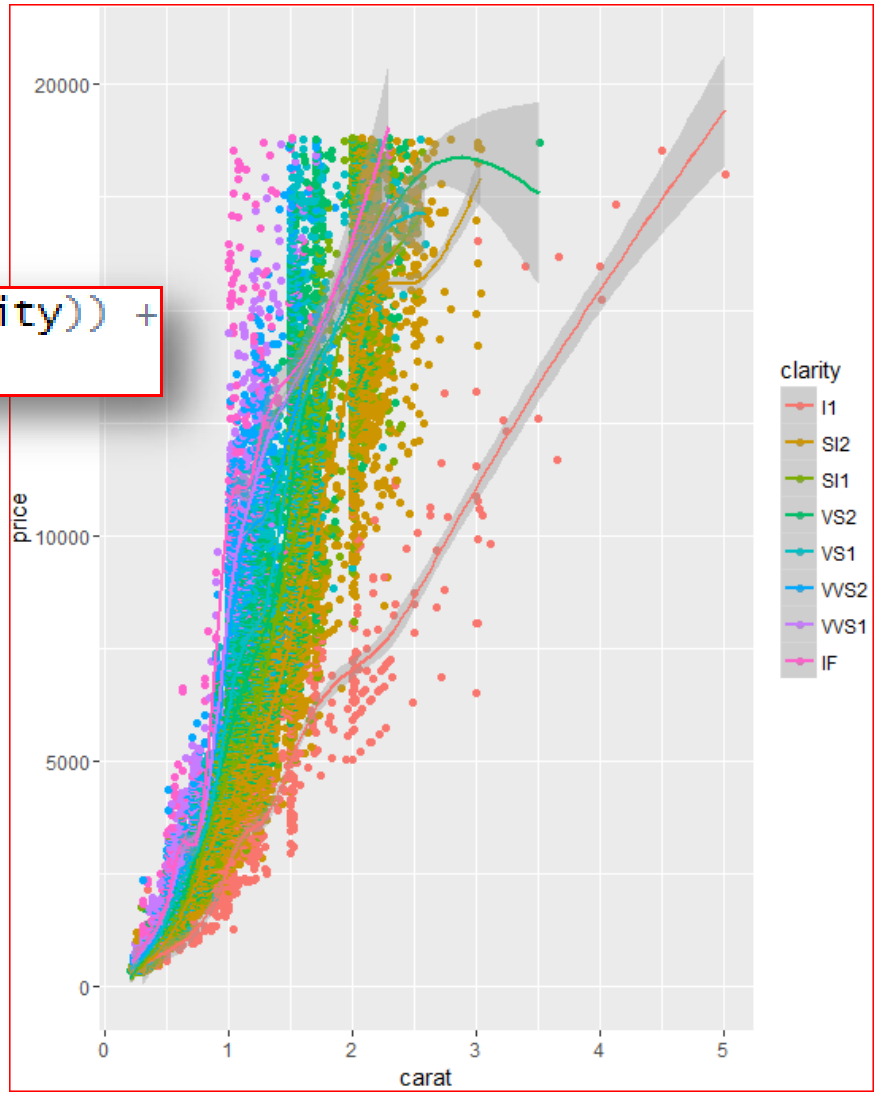
- 처음에 aes 매핑한 후 geom_point() 함수만 그 매핑정보를 승계하여 coloring
- geom_smooth() 함수는 초기 aes 매핑한 것에 대하여 회귀선을 그림

ggplot() 함수

- **앞의 예와 비교**

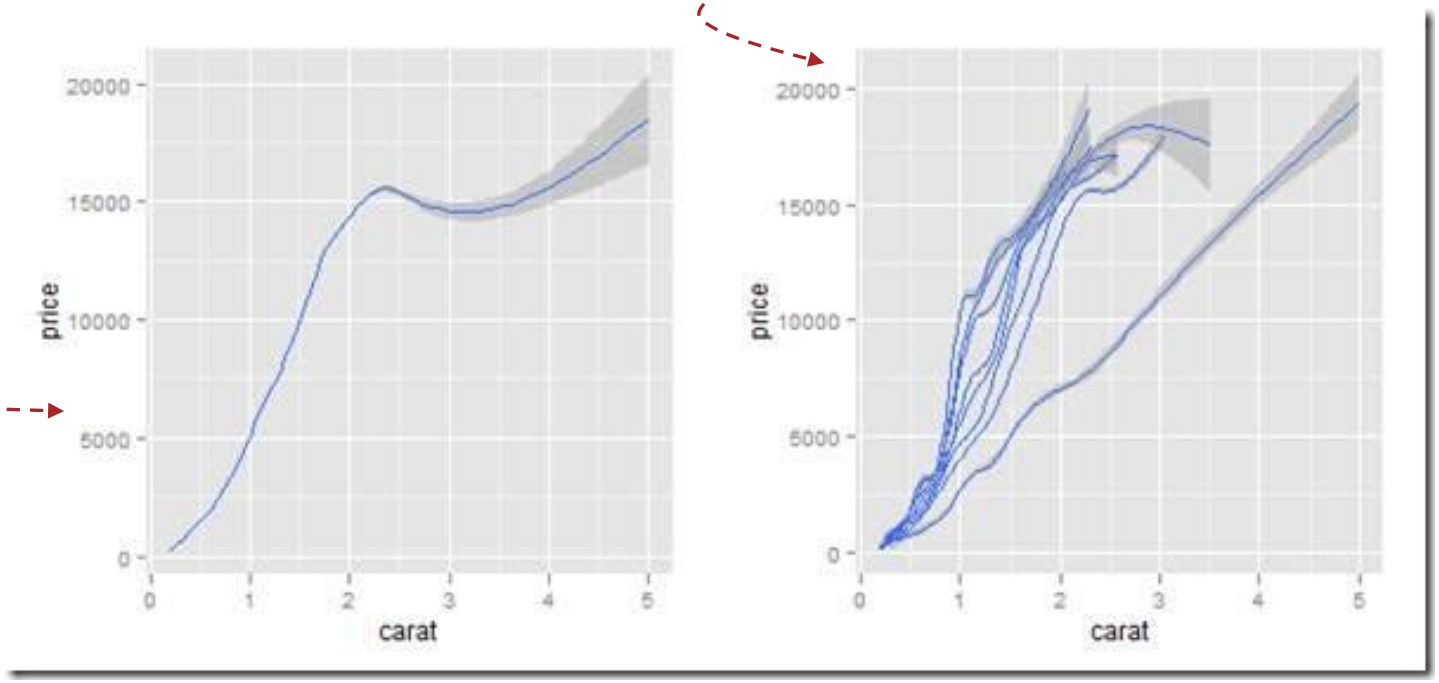
```
ggplot(data=diamonds, aes(x=carat, y=price, colour=clarity)) +
  geom_point() + geom_smooth()
```

  - 초기에 aes 매핑할 때 colour 옵션까지 넣었으므로, geom_smooth()함수는 각 colour값에 대한 8가지 회귀선을 그리게 됨



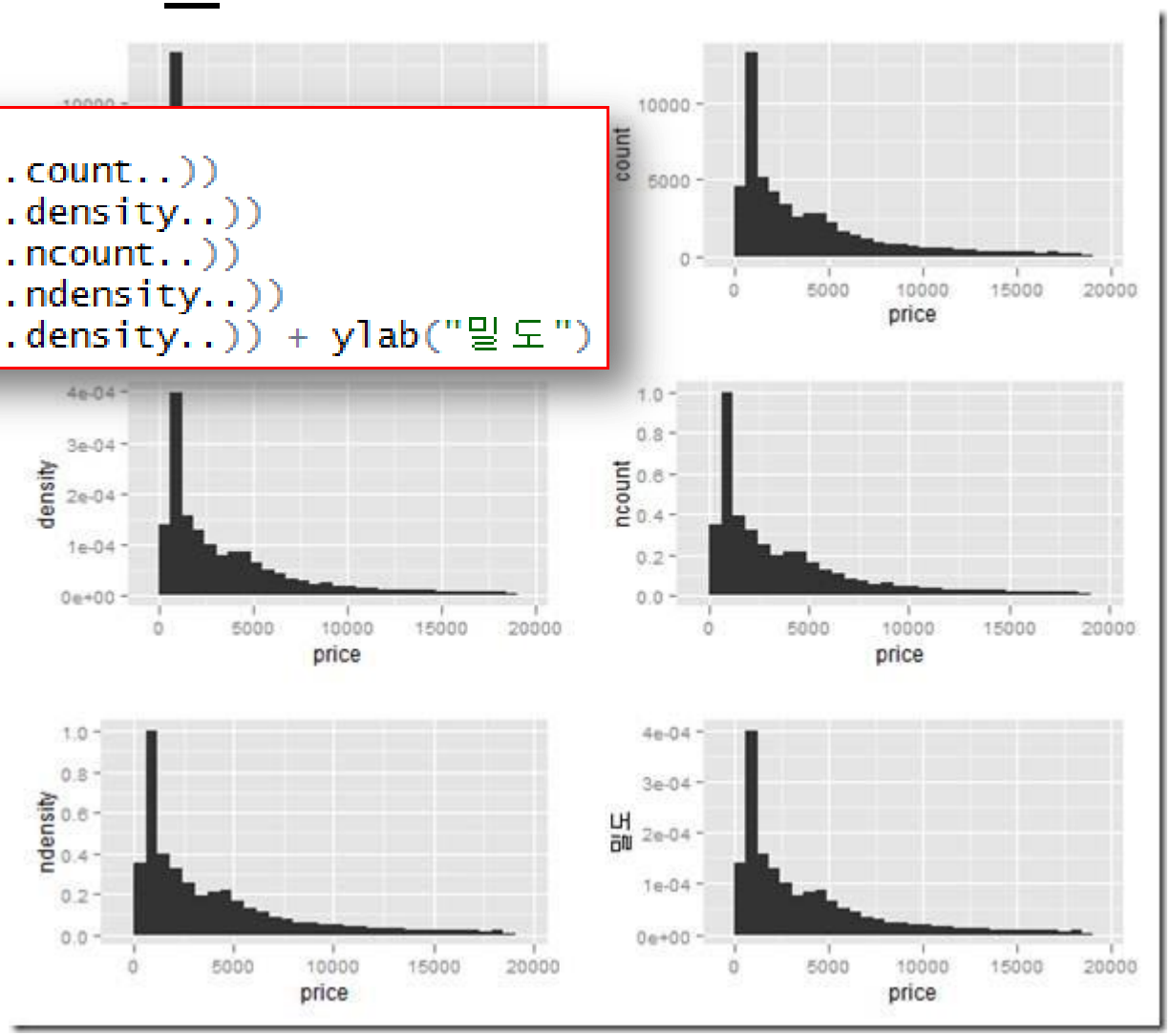
ggplot() 함수: geom_smooth

```
p <- ggplot(data=diamonds, aes(x=carat,y=price))
p + geom_smooth()
p + geom_smooth(aes(group=clarity))
```



ggplot() 함수: geom_bar

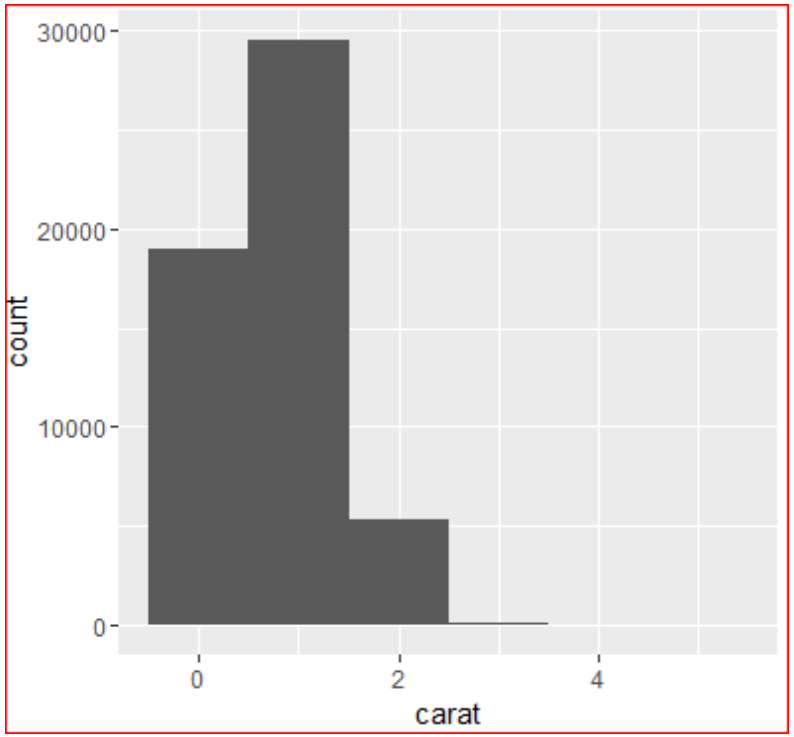
```
ggplot(data=diamonds, aes(x=price)) + geom_bar()
ggplot(data=diamonds, aes(x=price)) + geom_bar(aes(y=..count..))
ggplot(data=diamonds, aes(x=price)) + geom_bar(aes(y=..density..))
ggplot(data=diamonds, aes(x=price)) + geom_bar(aes(y=..ncount..))
ggplot(data=diamonds, aes(x=price)) + geom_bar(aes(y=..ndensity..))
ggplot(data=diamonds, aes(x=price)) + geom_bar(aes(y=..density..)) + ylab("밀도")
```

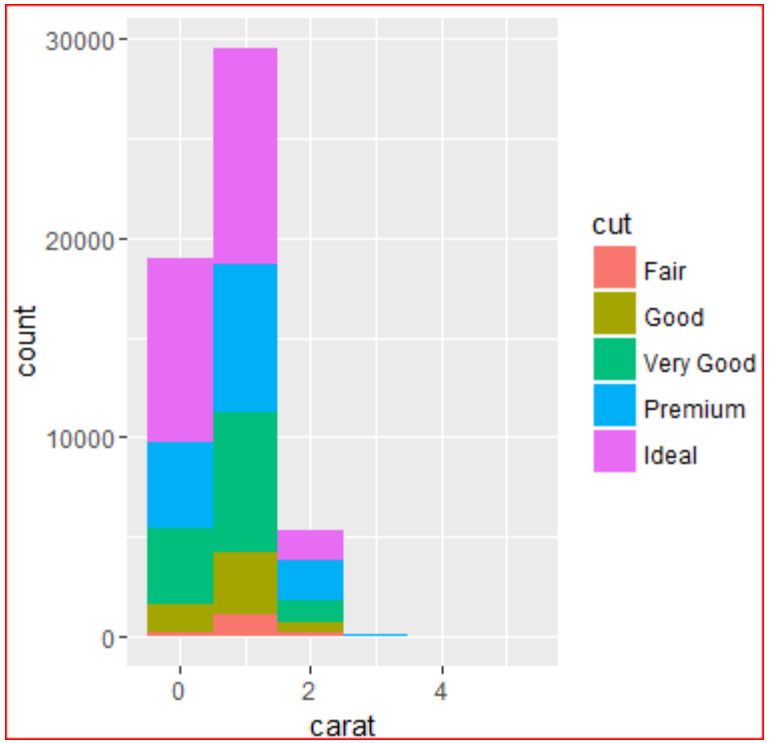


- count: 각 빈(bin)에 해당하는 관측값의 개수
- density: 각 빈(bin)의 밀도 (전체의 합이 1)
- ncount: count와 같으나 [0,1]로 스케일링
- ndensity: density와 같으나 [0,1]로 스케일링

ggplot() 함수: geom_histogram

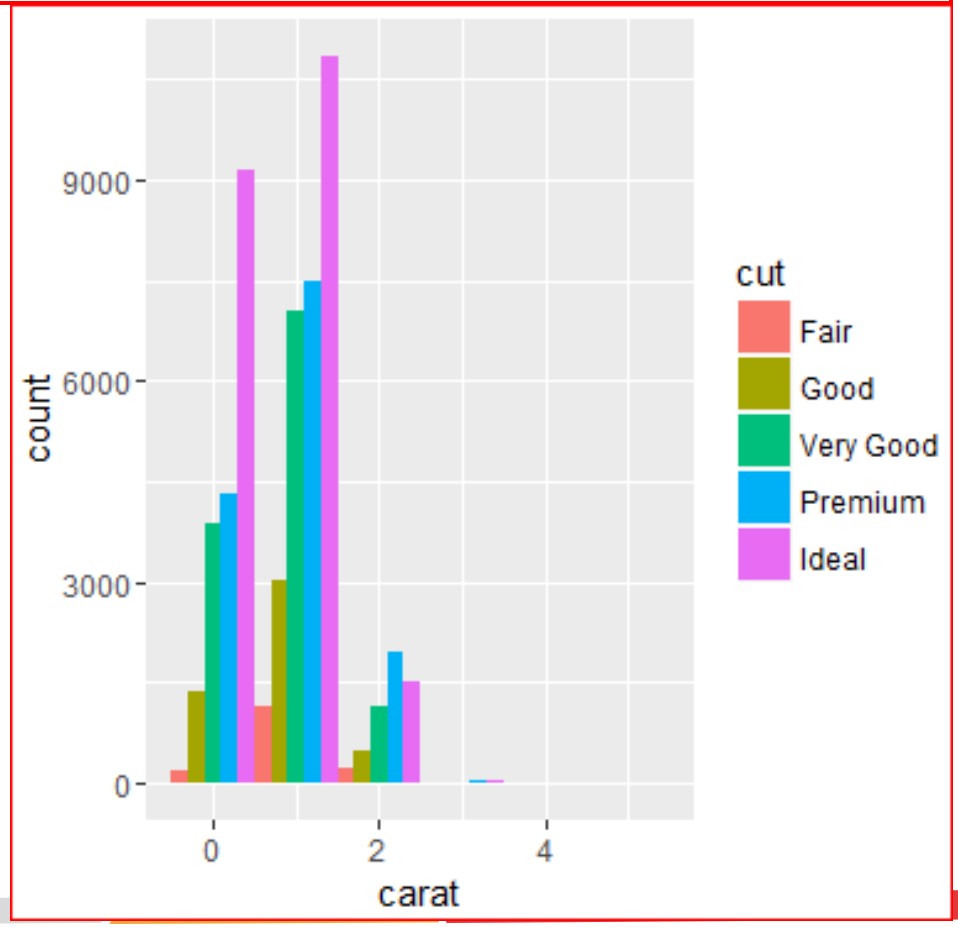
```
ggplot(diamonds, aes(carat)) + geom_histogram(binwidth = 1)

ggplot(diamonds, aes(carat)) + geom_histogram(aes(fill=cut), binwidth = 1)
```





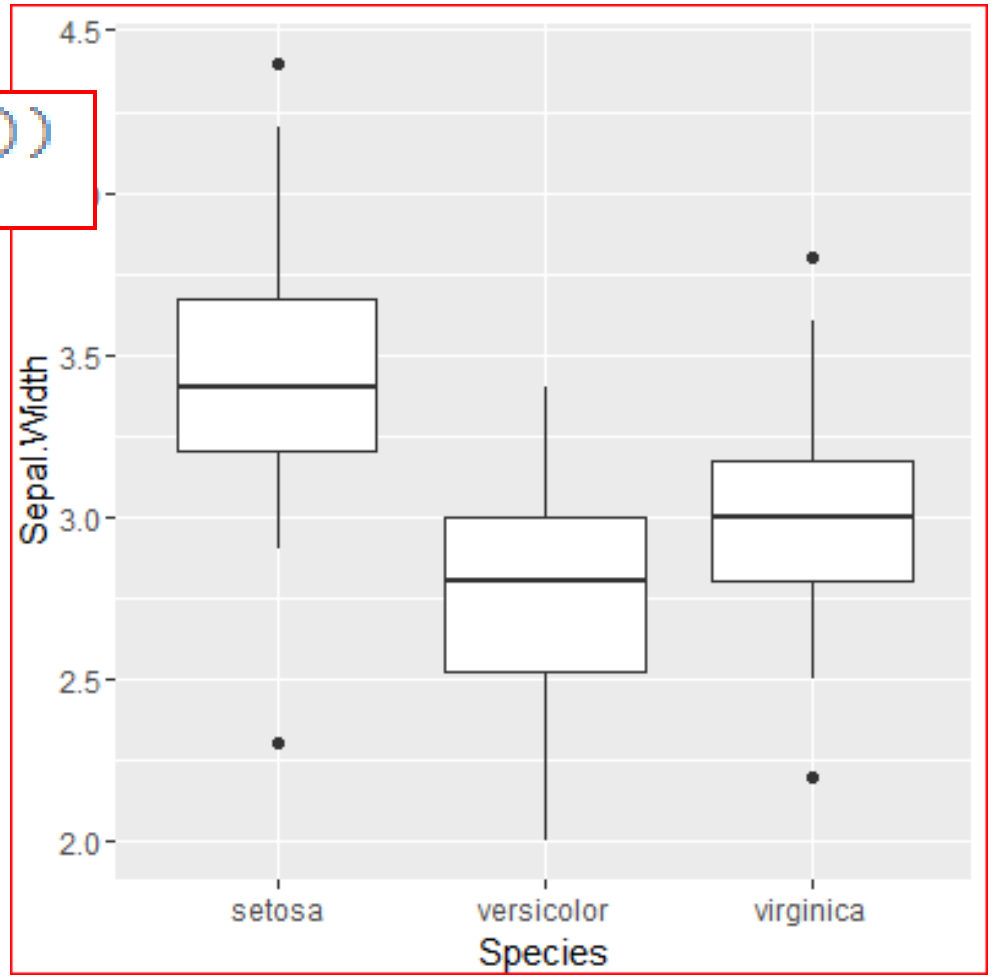
ggplot() 함수: geom_histogram

```
ggplot(diamonds, aes(carat)) +
  geom_histogram(aes(fill=cut), binwidth = 1, position="dodge")
```



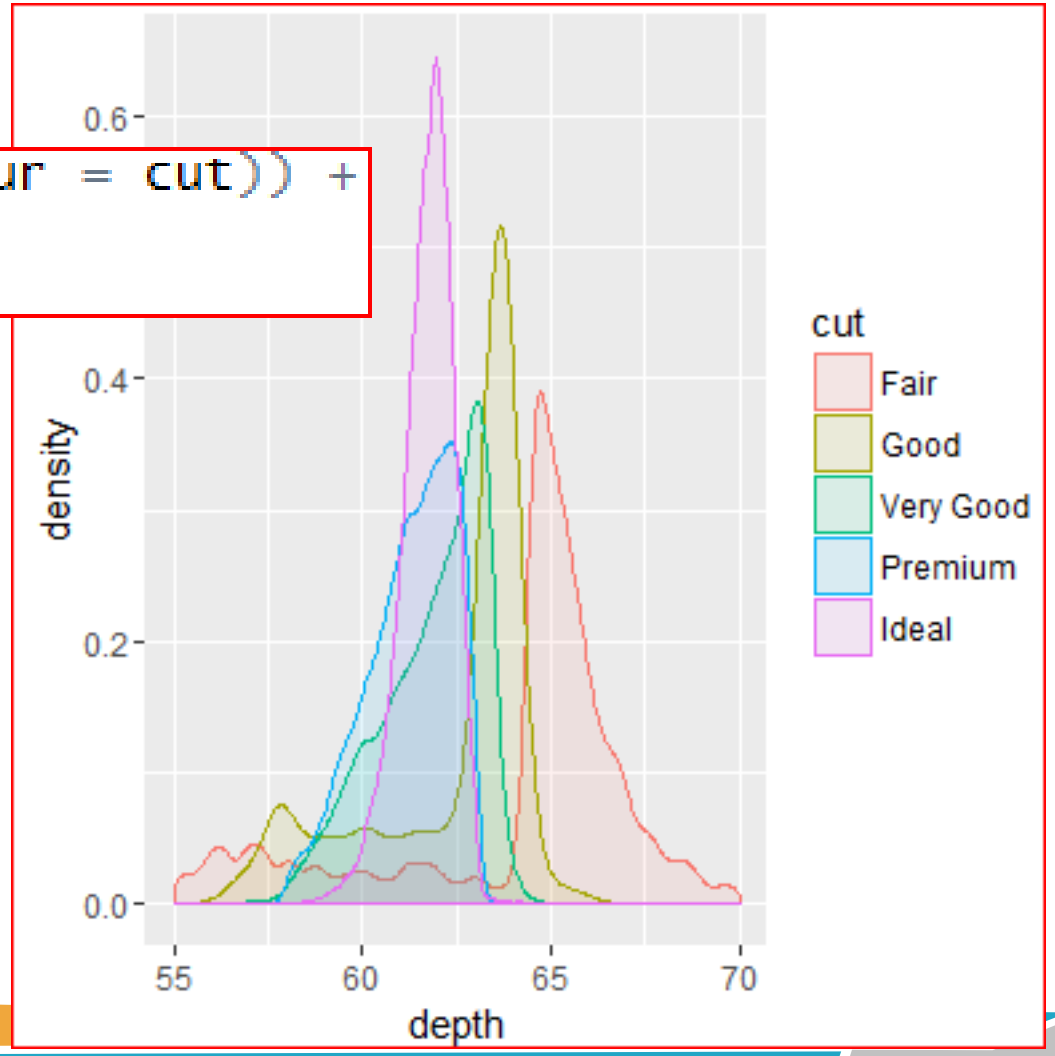
ggplot() 함수: geom_boxplot

```
p <- ggplot(iris, aes(Species, Sepal.Width))
p + geom_boxplot()
```



ggplot() 함수: geom_density

```
ggplot(diamonds, aes(depth, fill = cut, colour = cut)) +
  geom_density(alpha = 0.1) +
  xlim(55, 70)
```



ggplot() 함수: geom_density

```
ggplot(diamonds, aes(carat, fill = cut)) +
  geom_density(position = "stack")
```