PR11 - ggplot

조성우

2020 5월 28일

1. ggplot2 기초

- ggplot2 plot의 기본성분

1.1. ggplot2 기본 사용법

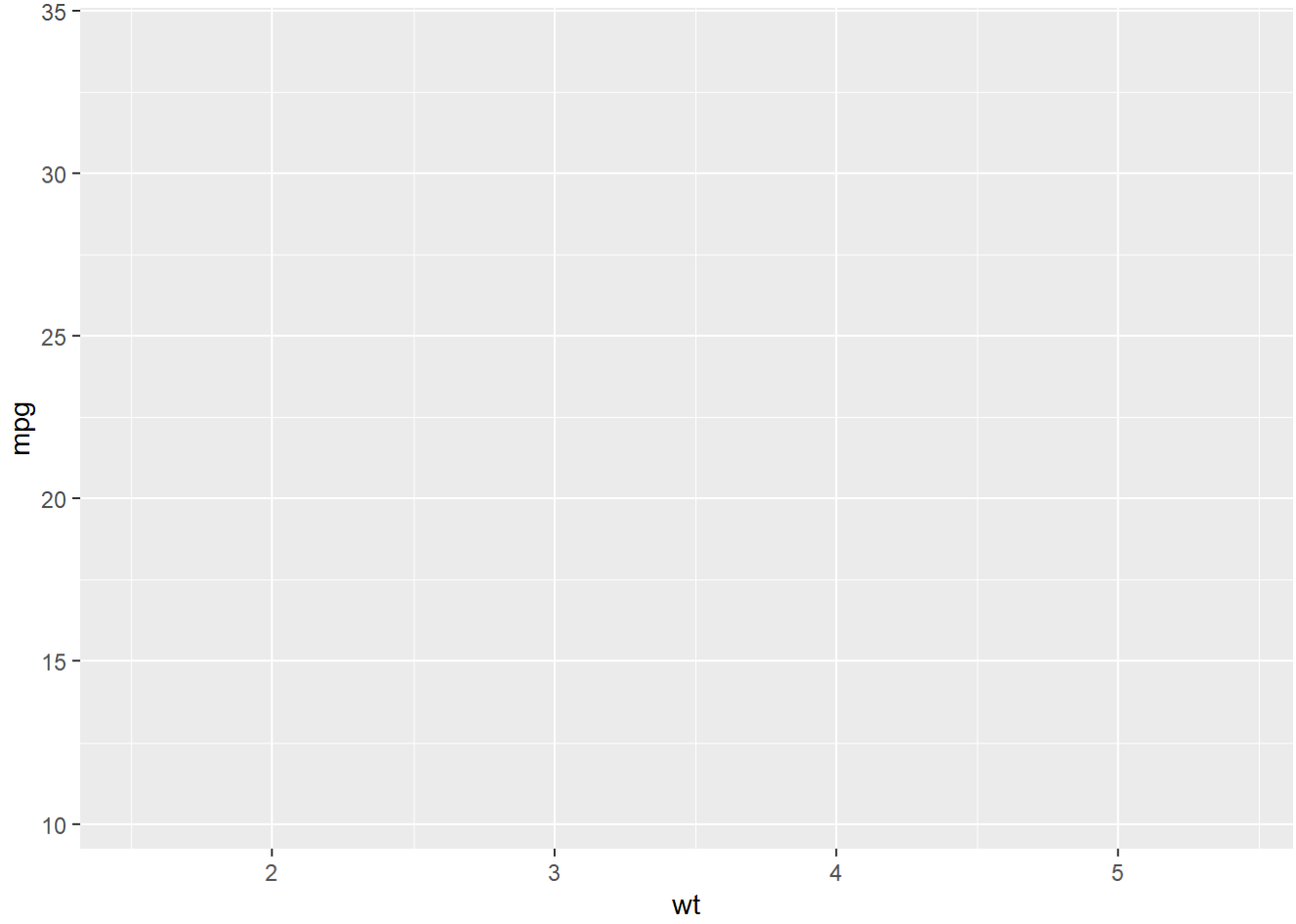
- mtcars를 활용: wt(중량) mpg(연비) cyl(실린더의 개수)

```
library(ggplot2)
head(mtcars)
```

	mpg <dbl>	cyl <dbl>	disp <dbl>	hp <dbl>	drat <dbl>	wt <dbl>	qsec <dbl>	vs <dbl>	am <dbl>
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0

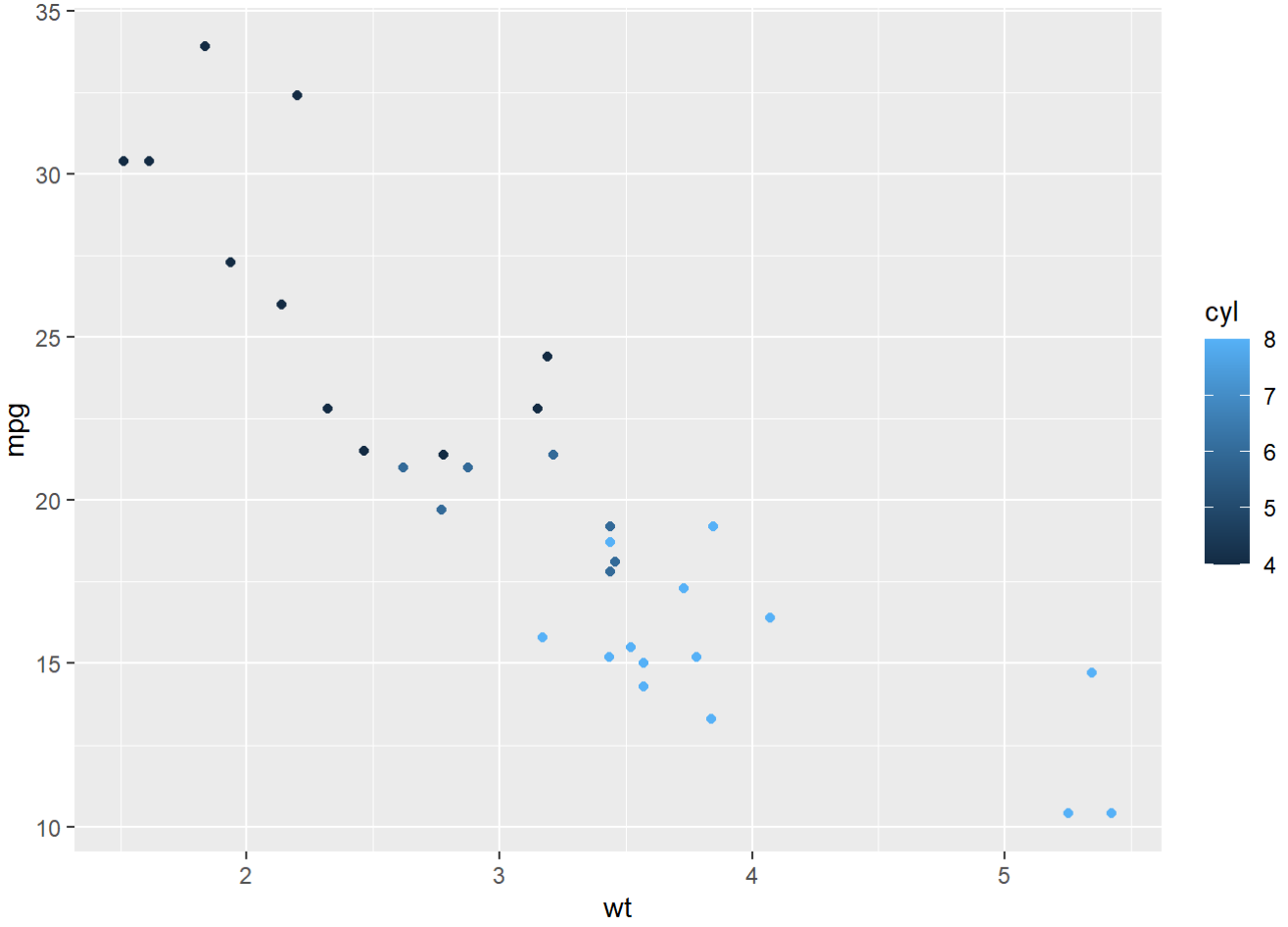
6 rows | 1-10 of 12 columns

```
p <- ggplot(data=mtcars,aes(x=wt,y=mpg,colour=cyl)) ; p
```



- geom_point()함수를 추가하여 산점도 표현

```
p <- p+geom_point() ; p
```



- ggplot 객체의 구조
  - summary를 보면 대략적인 그래프를 짐작할 수 있음
  - mapping을 보면 x,y축의 데이터와 색상을 결정짓는 변수를 확인 가능함
  - geom_point()는 산점도 그래프라는 의미
  - stat_identity는 통계변환이 identity, 즉 변환이 없는 상태의 데이터라는 것을 의미
  - position_identity도 데이터 위치가 어떠한 조정도 없었다는 것을 의미
  - na.rm=False는 결측값 제거를 하지 않았다는 것을 의미함

```
class(p)
```

```
## [1] "gg"     "ggplot"
```

```
attributes(p)
```

```
## $names
## [1] "data"        "layers"      "scales"      "mapping"     "theme"
## [6] "coordinates" "facet"       "plot_env"    "labels"
##
## $class
## [1] "gg"     "ggplot"
```

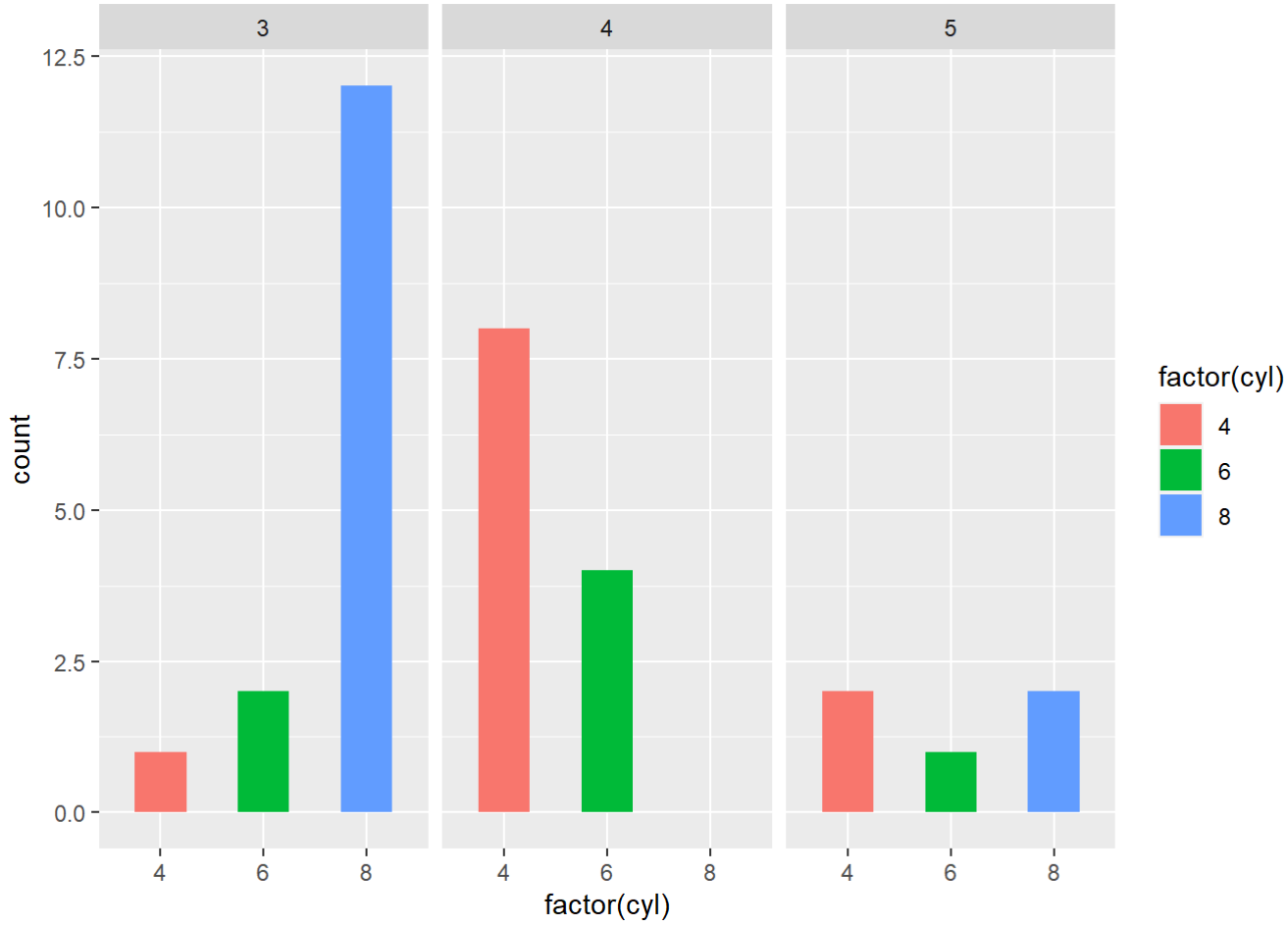
```
summary(p)
```

```
## data: mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, carb [32x11]
## mapping:  x = ~wt, y = ~mpg, colour = ~cyl
## faceting: <ggproto object: Class FacetNull, Facet, gg>
##     compute_layout: function
##     draw_back: function
##     draw_front: function
##     draw_labels: function
##     draw_panels: function
##     finish_data: function
##     init_scales: function
##     map_data: function
##     params: list
##     setup_data: function
##     setup_params: function
##     shrink: TRUE
##     train_scales: function
##     vars: function
##     super:  <ggproto object: Class FacetNull, Facet, gg>
## -----------------------------------
## geom_point: na.rm = FALSE
## stat_identity: na.rm = FALSE
## position_identity
```

1.2. ggplot2로 barplot 그리기

- 실린더 수에 대한 barplot을 그림
- 막대는 기어의 개수에 따라 서로 다른 facet에 출력
- facet이란 독립된 subplot이 그려지는 패널구조를 의미
- 결과값을 보면 3개의 subplot이 있는 것을 확인 할 수 있음
- 산점도 색상옵션으로 colour인수사용, barplot은 색상옵션으로 fill인수 사용

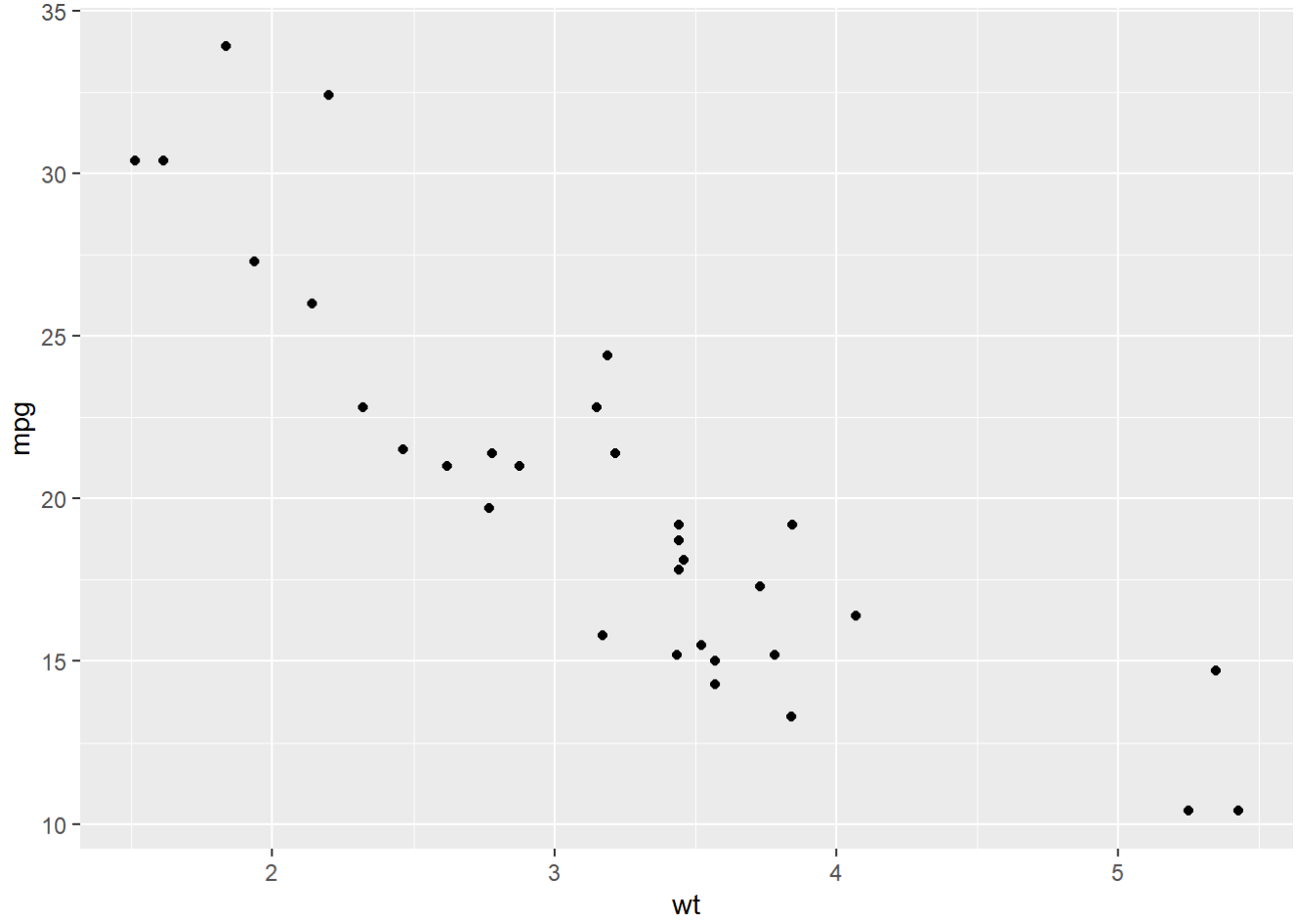
```
b <-ggplot(data=mtcars,aes(x=factor(cyl),fill=factor(cyl)))
b <- b + geom_bar(width=.5) #bar의 넓이를 정의
b <- b + facet_grid(.~gear) #기어 개수에 따른 facet 분할
b
```



1.3. ggplot Layer

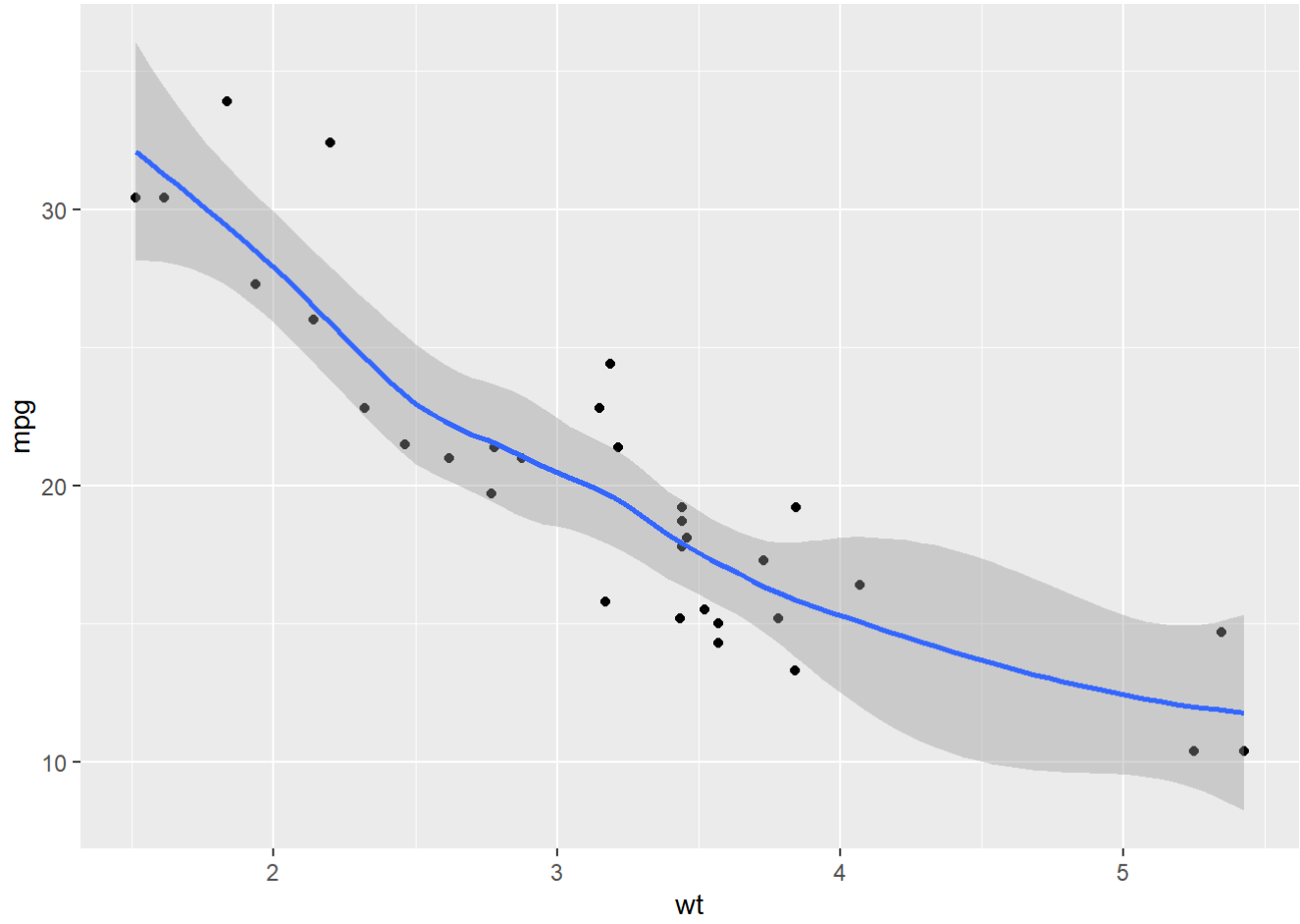
- ggplot은 Layer를 쌓아가는 방식으로 사용
- ggplot = layers + scales + coordinate system
- layers = data + mapping + geom + stat + position
- scales와 coordinate system은 그림을 그릴 캔버스의 개념
- layers가 실제 그리는 그림
- data, mapping, geom 등등으로 하나씩 중첩해가면서 plot을 그림
- geom의 요소 또한 중첩 가능

```
p <- ggplot(mtcars,aes(wt,mpg))
p <- p+geom_point() ; p
```




```
p <- p+geom_smooth(method="loess") #라인 그래프 smooth 그리기
p
```

```
## `geom_smooth()` using formula 'y ~ x'
```



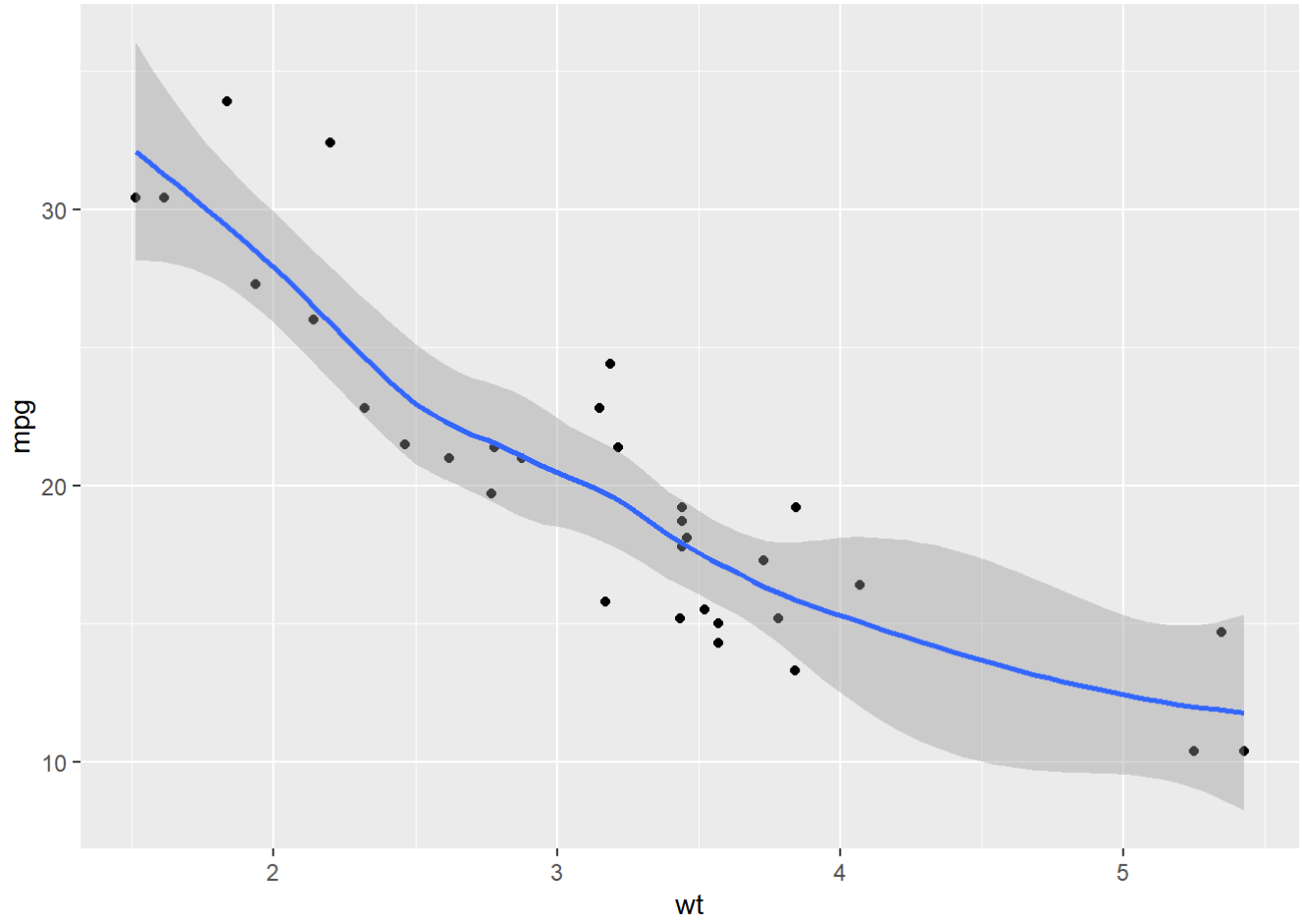

```
summary(p)
```

```
## data: mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, carb [32x11]
## mapping:  x = ~wt, y = ~mpg
## faceting: <ggproto object: Class FacetNull, Facet, gg>
##     compute_layout: function
##     draw_back: function
##     draw_front: function
##     draw_labels: function
##     draw_panels: function
##     finish_data: function
##     init_scales: function
##     map_data: function
##     params: list
##     setup_data: function
##     setup_params: function
##     shrink: TRUE
##     train_scales: function
##     vars: function
##     super:  <ggproto object: Class FacetNull, Facet, gg>
## -----------------------------------
## geom_point: na.rm = FALSE
## stat_identity: na.rm = FALSE
## position_identity
##
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE, flipped_aes = FALSE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, method = loess
## position_identity
```

- 동일한 결과를 다르게 표현

```
p <- ggplot(mtcars,aes(wt,mpg)) +
  geom_point() +
  geom_smooth(method="loess") #라인 그래프 smooth 그리기
p
```

```
## `geom_smooth()` using formula 'y ~ x'
```



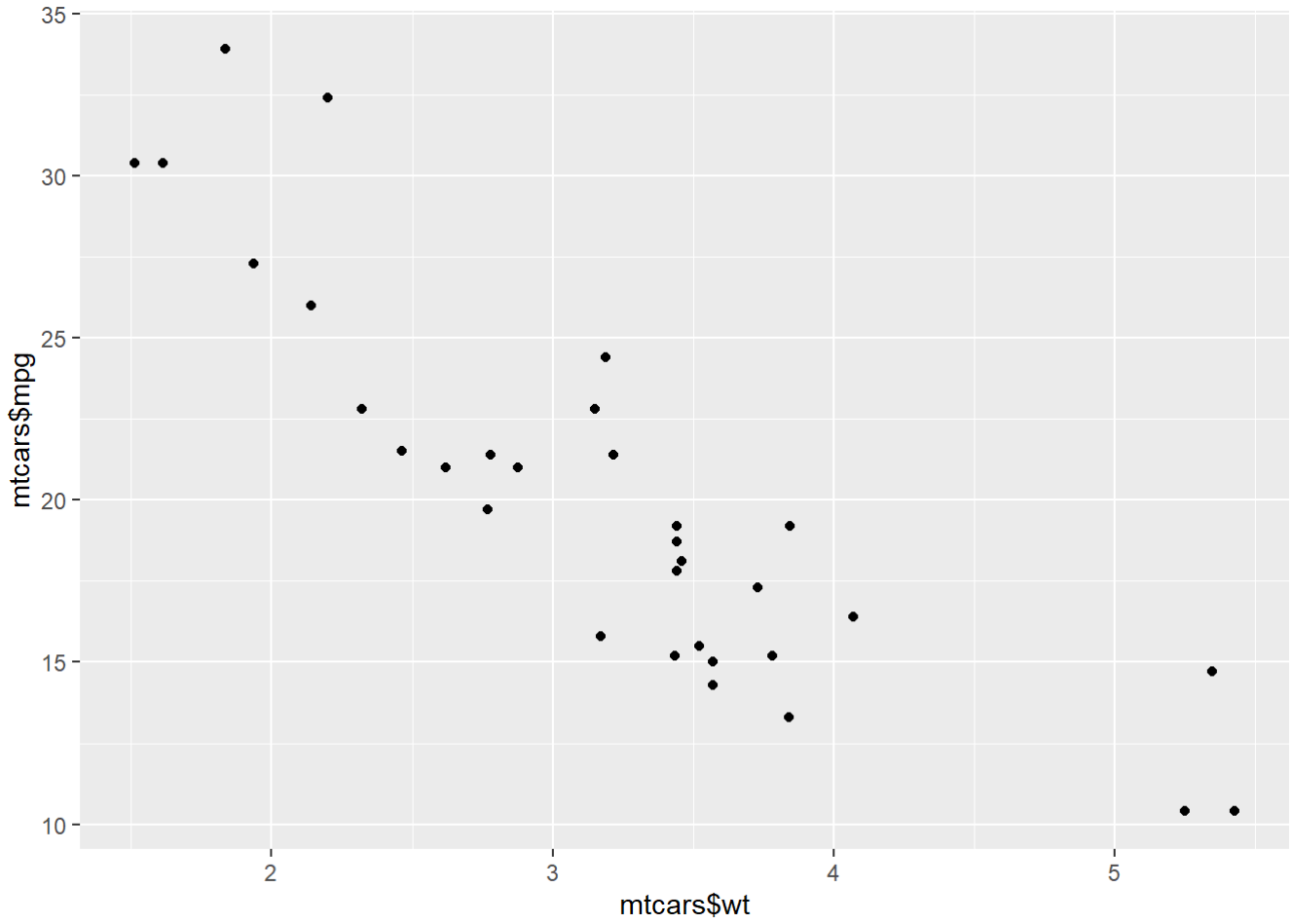
1.4. ggplot 요약

2. ggplot2 함수군

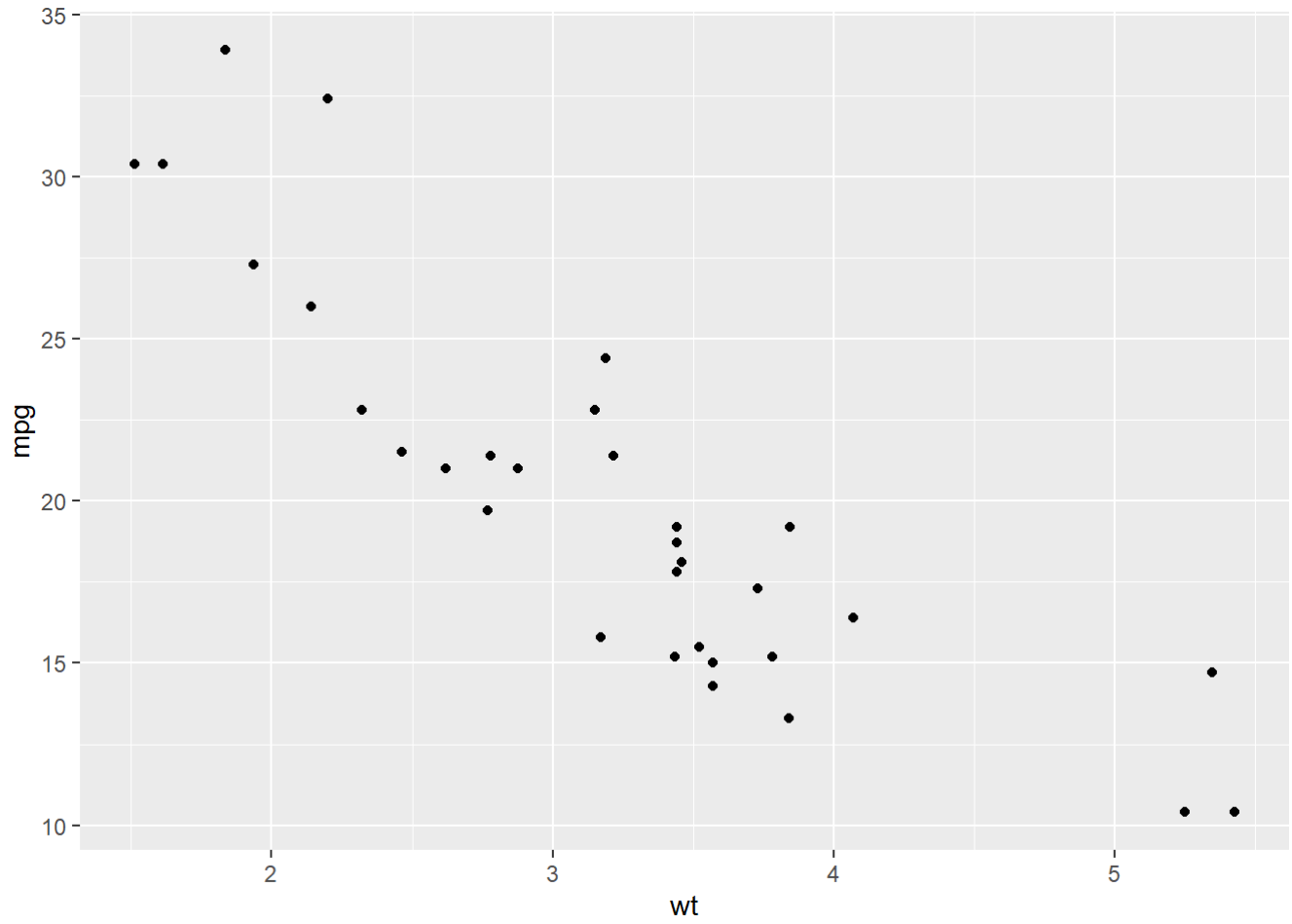
2.1. Plot creating 함수군

- 기초 플롯인 ggplot 클래스 객체를 생성
- 가장 많이 사용하는 함수는 ggplot()과 qplot()
  - ggplot() 함수는 dataframe 객체로 플롯을 그릴 때 사용
  - qplot() 함수는 각 변수가 독립적인 객체로 존재할 때 사용

```
qplot(mtcars$wt,mtcars$mpg)
```

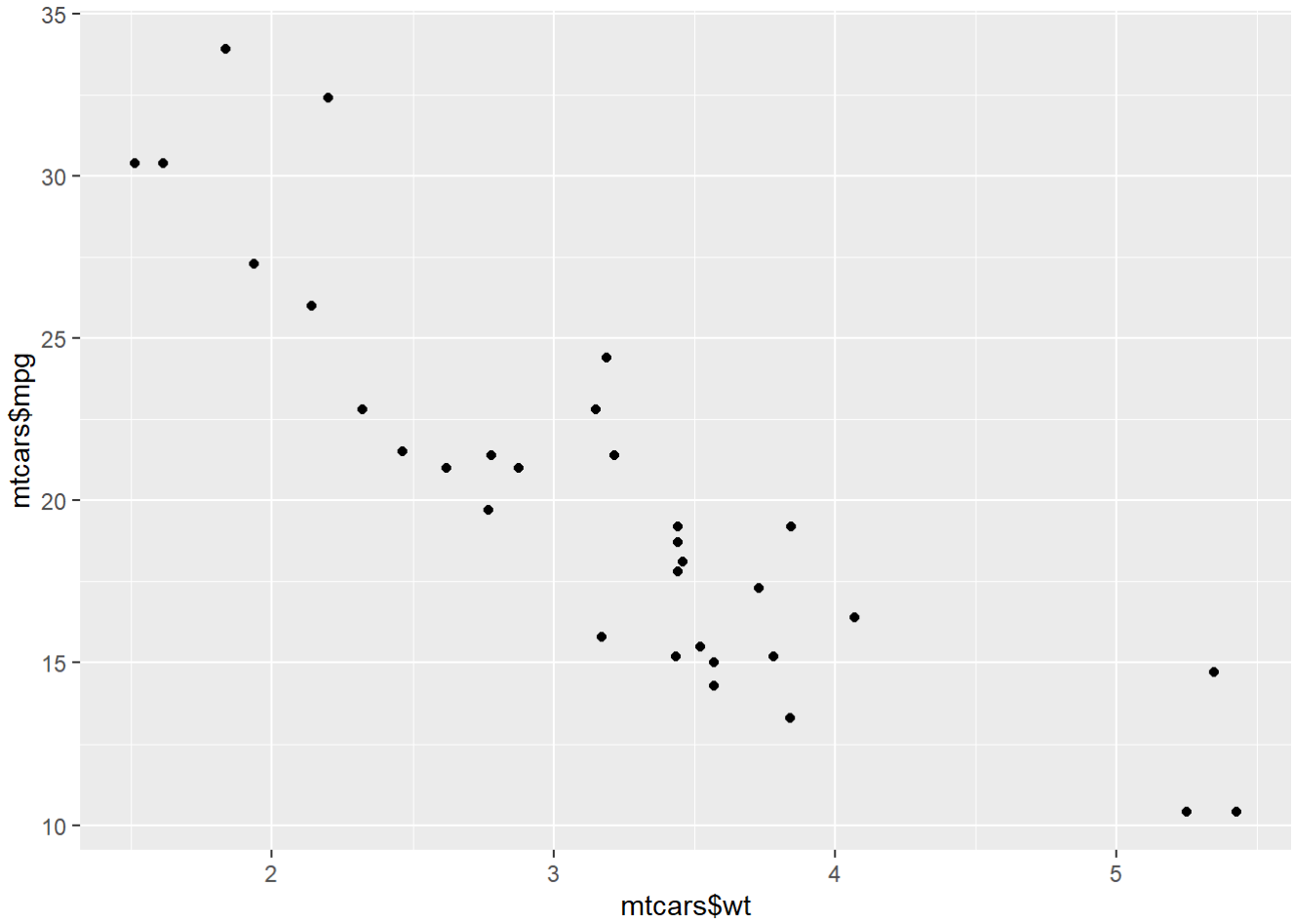



```
ggplot(mtcars,aes(wt,mpg)) + geom_point()
```

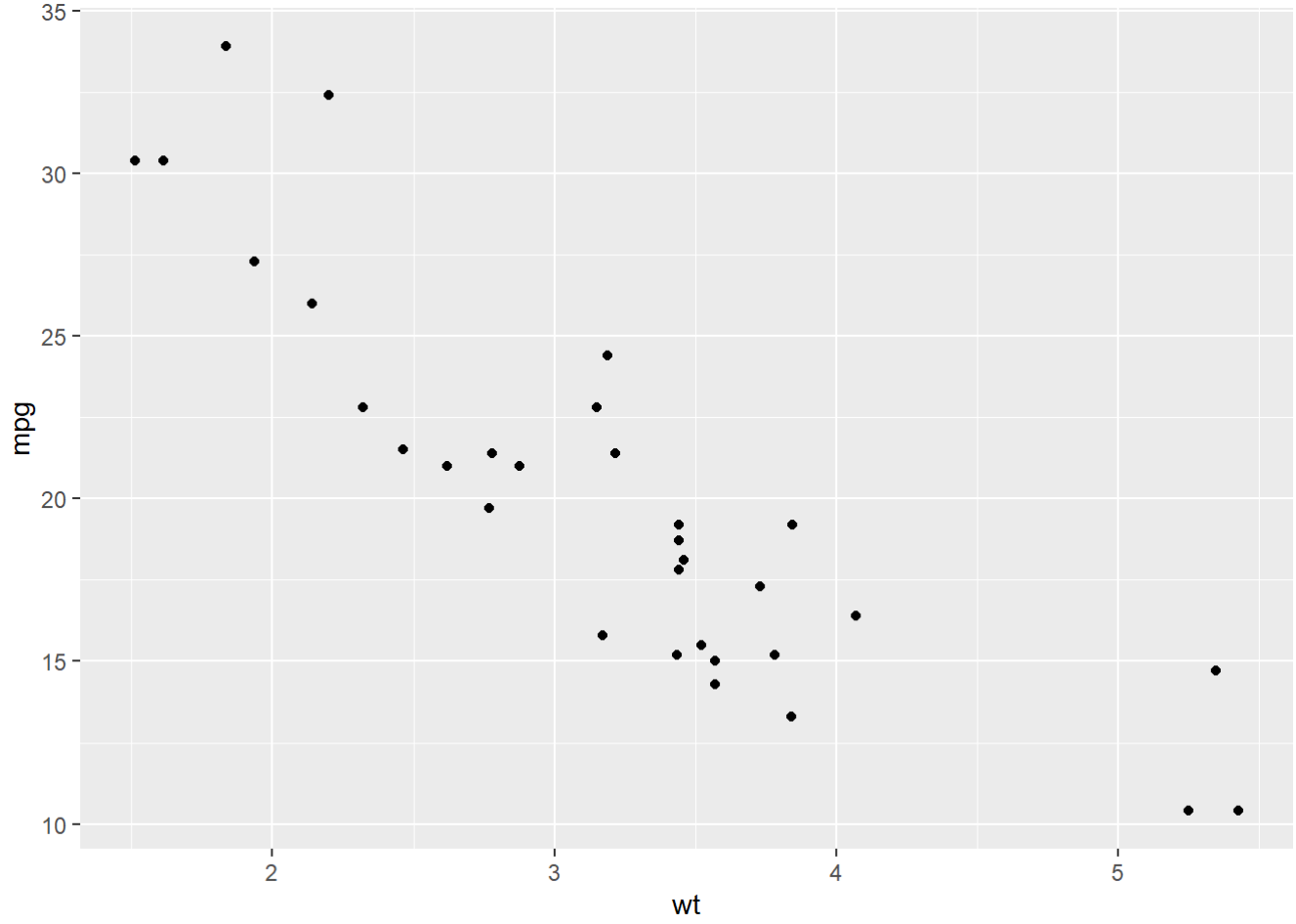


- qplot() 함수는 geo을 지정하지 않을 경우 point로 적용됨
- ggplot은 모든 그래픽 인수들을 구체적으로 지정해야함

```
qplot(mtcars$wt, mtcars$mpg, geom="point")
```




```
qplot(wt,mpg, data=mtcars, geom="point") #그래프는 위와 동일함
```

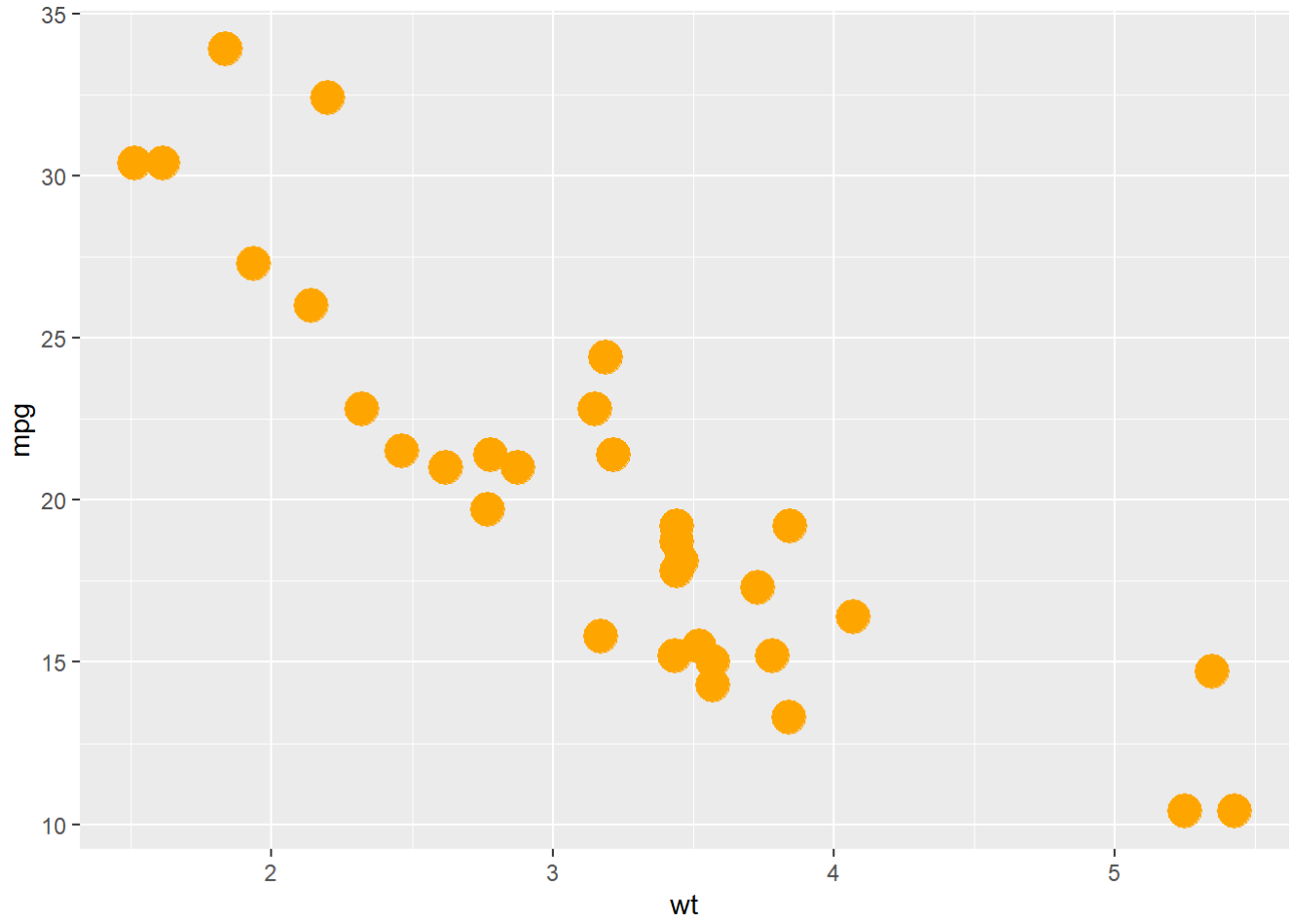


2.2. Geoms 함수군

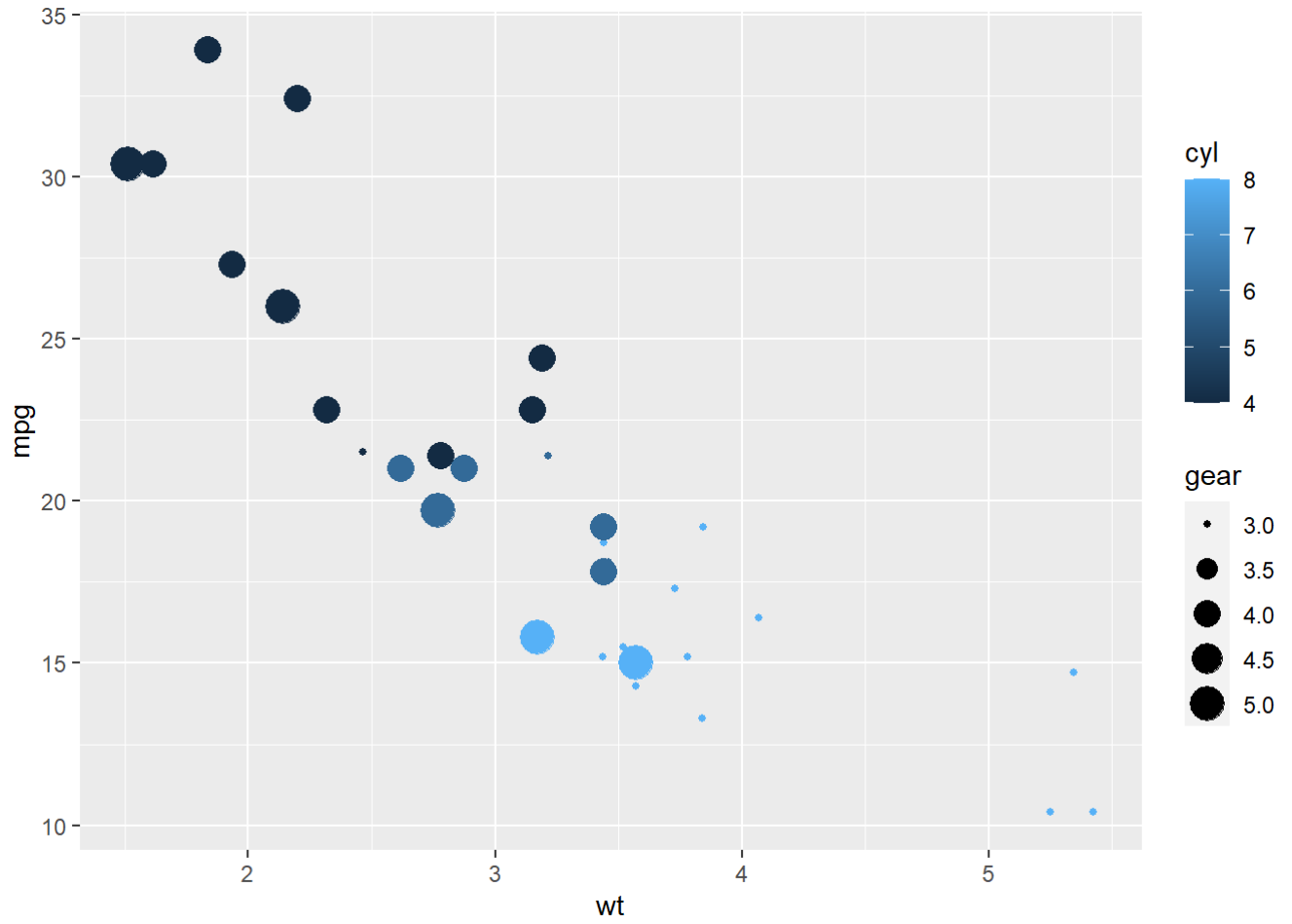
- geometric 요소를 지정하기 위한 함수군
- 선,점,막대,박스,파이 등을 생각하면 됨

2.2.1. geom_point() 산점도

```
p <- ggplot(data = mtcars, aes(wt, y=mpg))
p + geom_point(colour="orange",size=6)
```

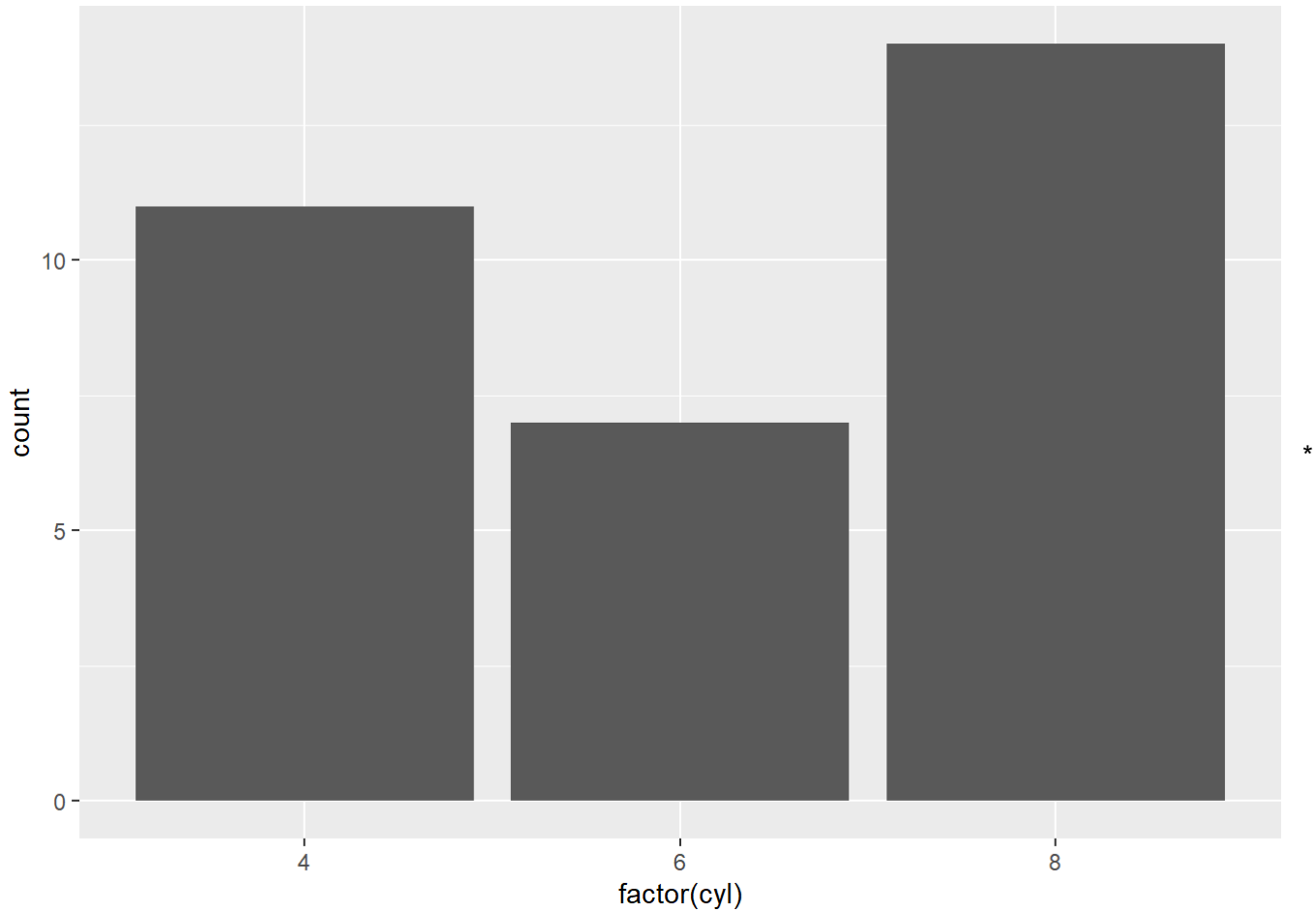



```
p + geom_point(aes(colour=cyl, size=gear))
```



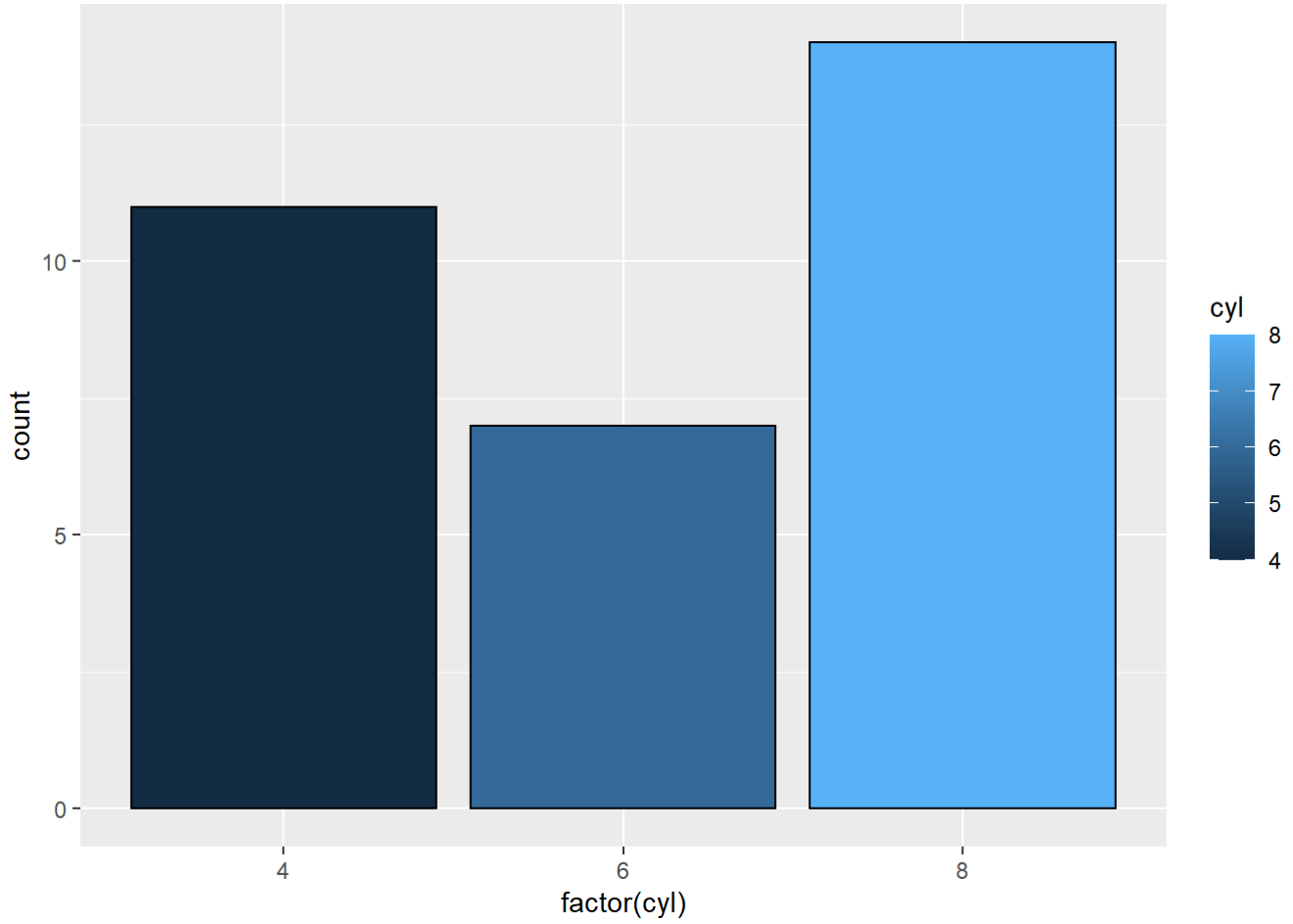
2.2.2 geom_bar() 막대 그래프

```
p <- ggplot(mtcars, aes(factor(cyl)))
p + geom_bar()
```

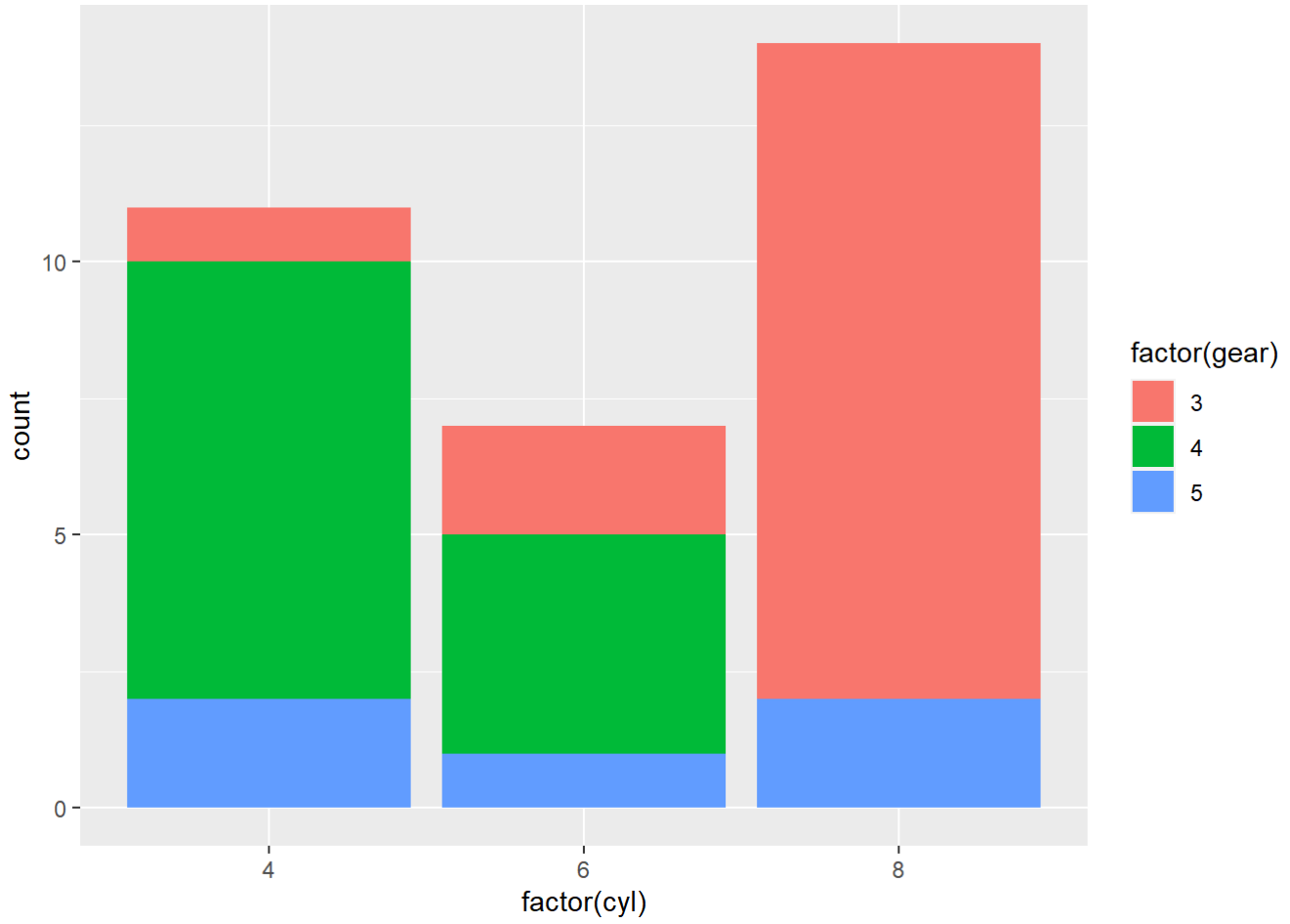


geom_bar를 aes함수로 꾸미기

```
p <- ggplot(mtcars, aes(factor(cyl)))
p + geom_bar(aes(fill=cyl),colour="black")
```

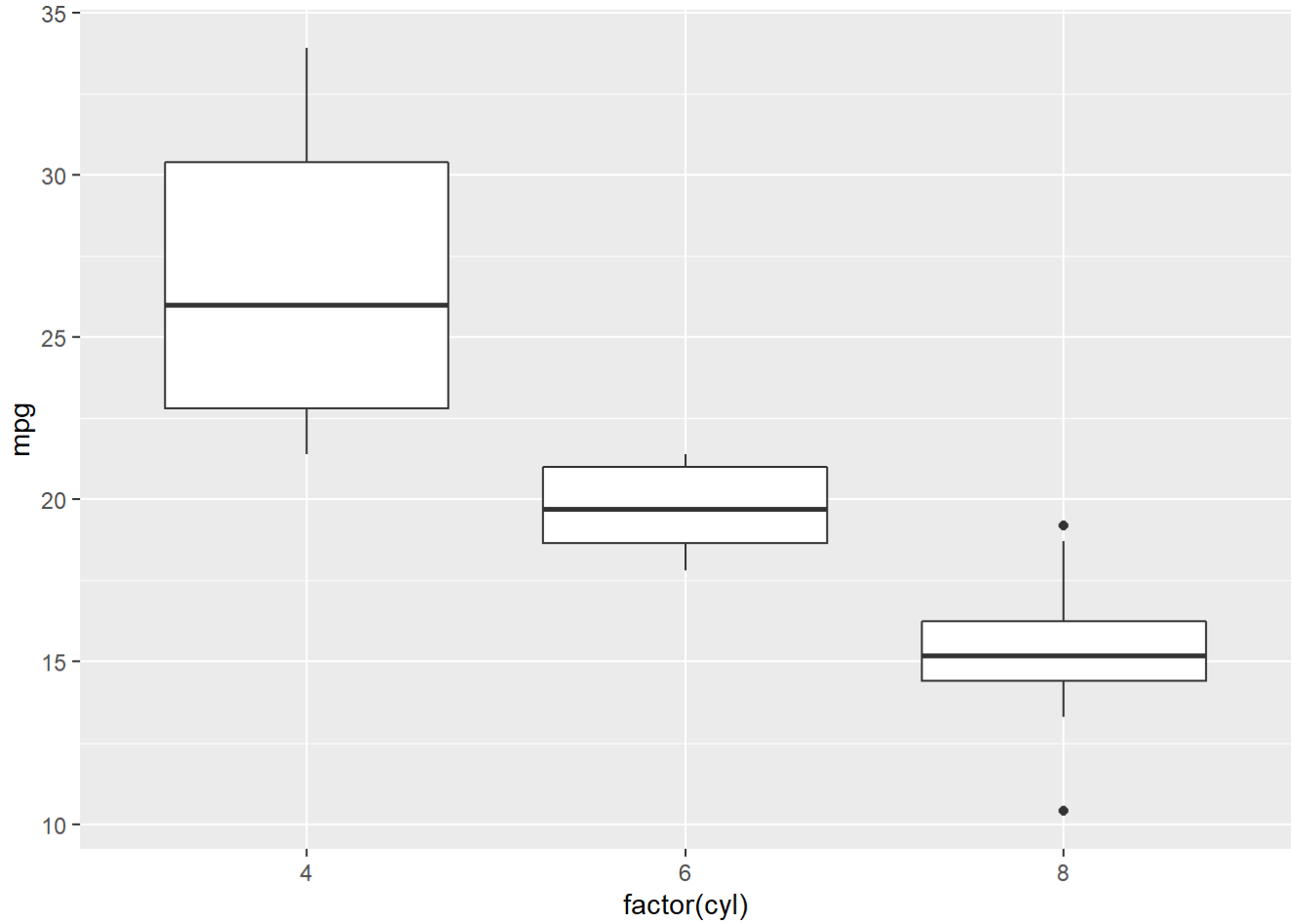



```
p <- ggplot(mtcars, aes(factor(cyl)))
p+geom_bar(aes(fill=factor(gear)))
```



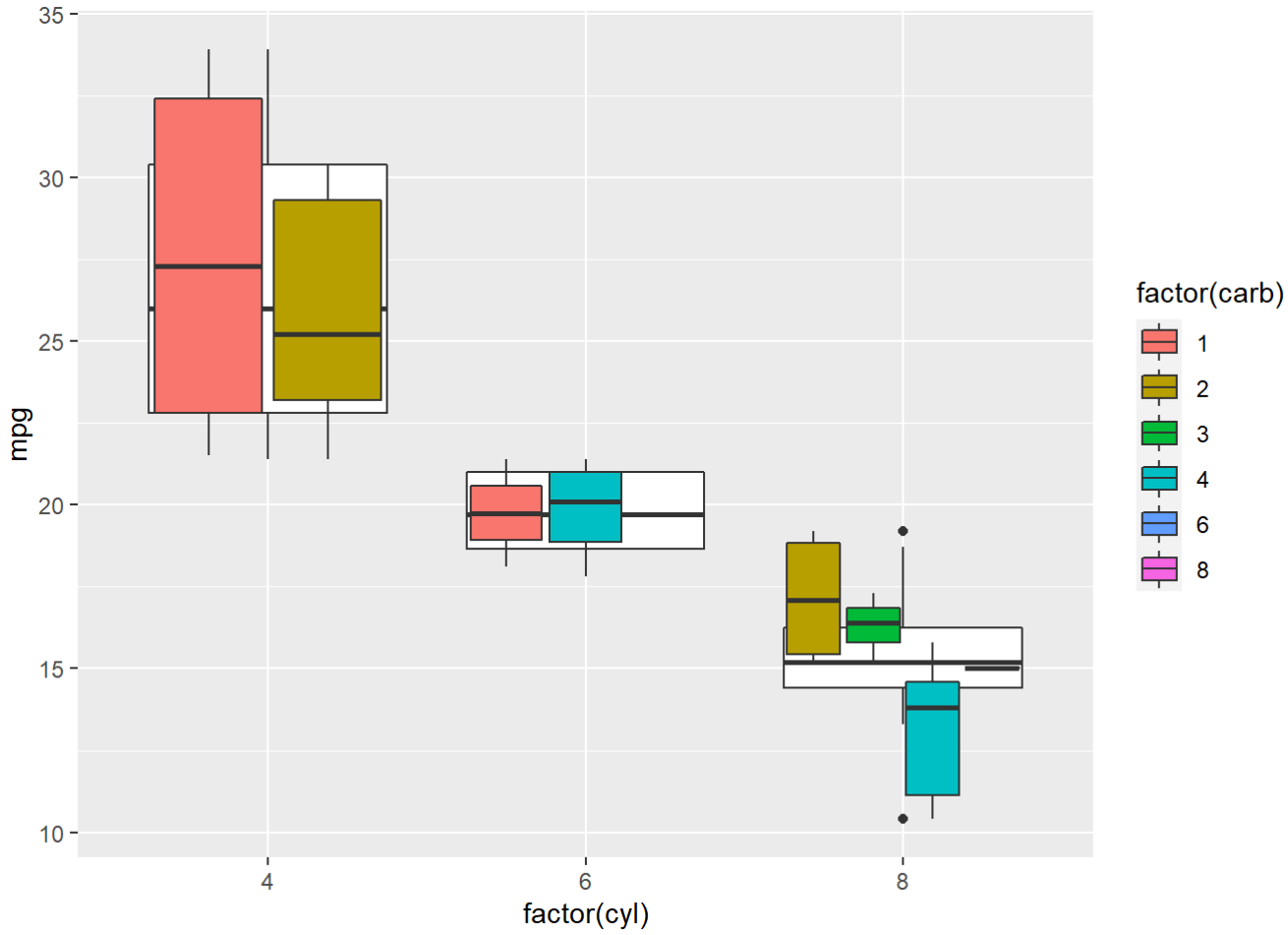
2.2.3. geom_boxplot() 박스플롯

```
p <- ggplot(mtcars, aes(factor(cyl),mpg)) #기본 박스플랏
p <- p + geom_boxplot()
p
```

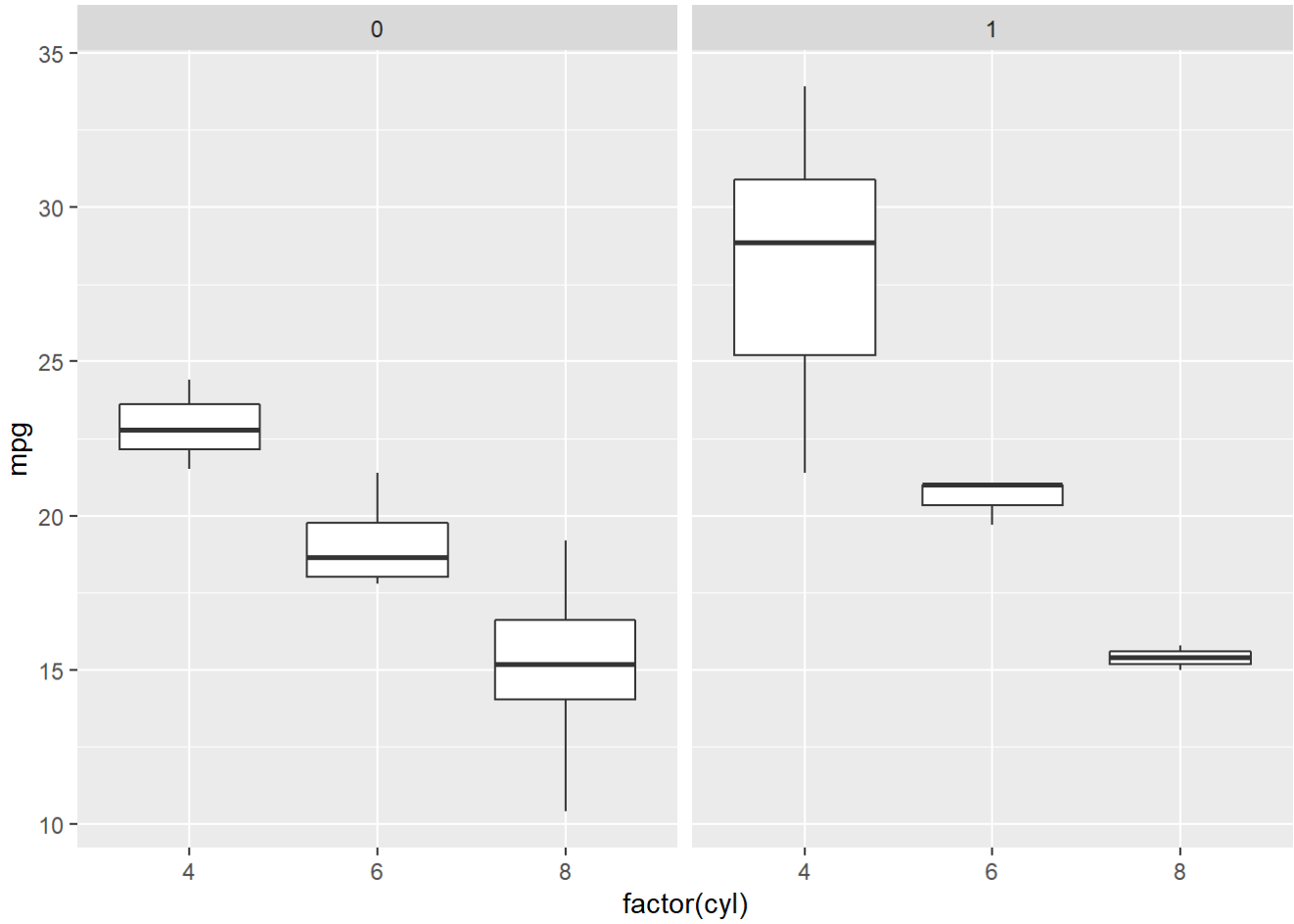


- box-plot을 aes함수로 꾸미기

```
p + geom_boxplot(aes(fill=factor(carb))) #박스 내부를 carb 팩터로 구분하여 표현
```



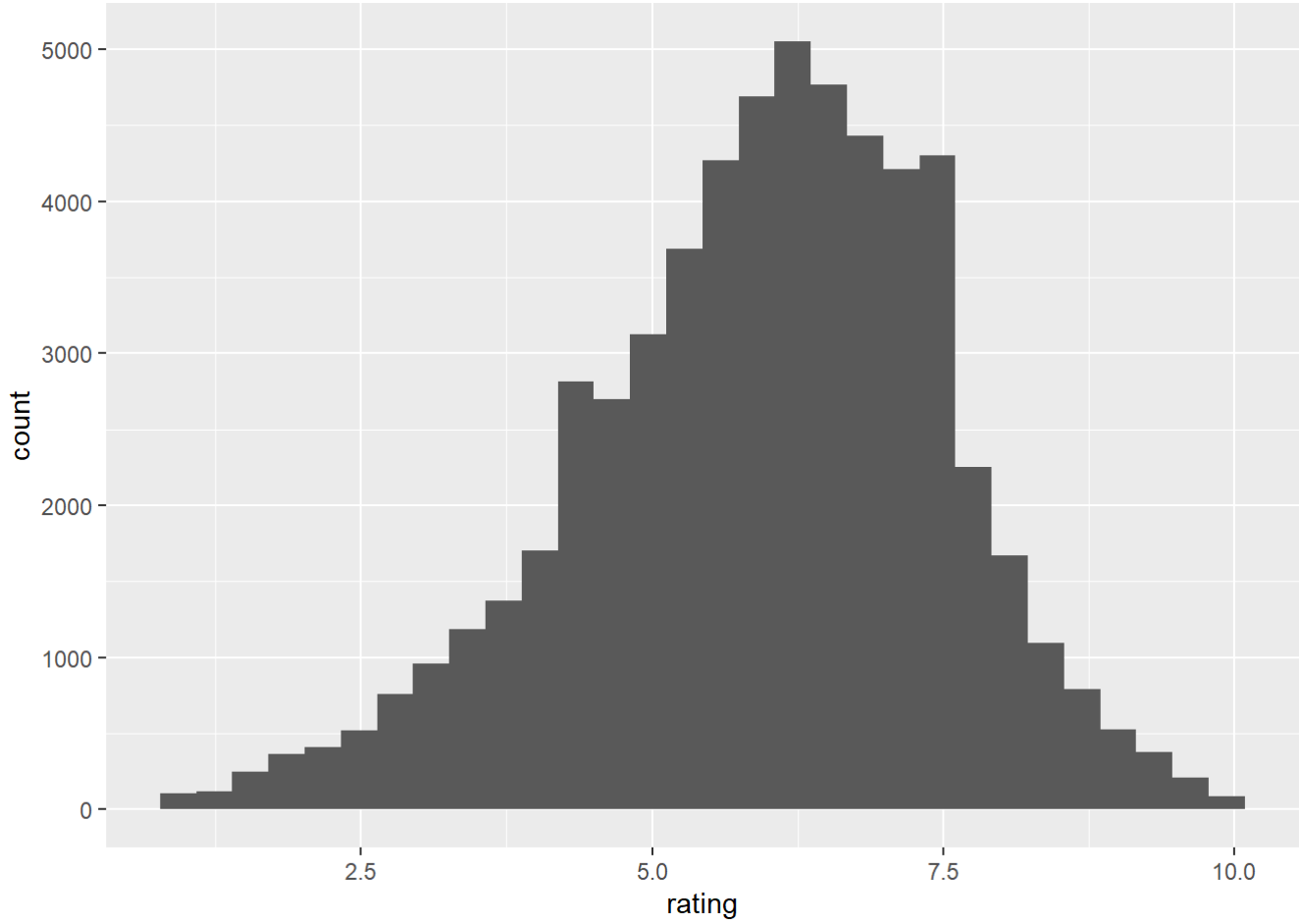

```
p + facet_grid(~am) #am을 기준으로 facet 분리
```



2.2.4. geom_histogram() 히스토그램

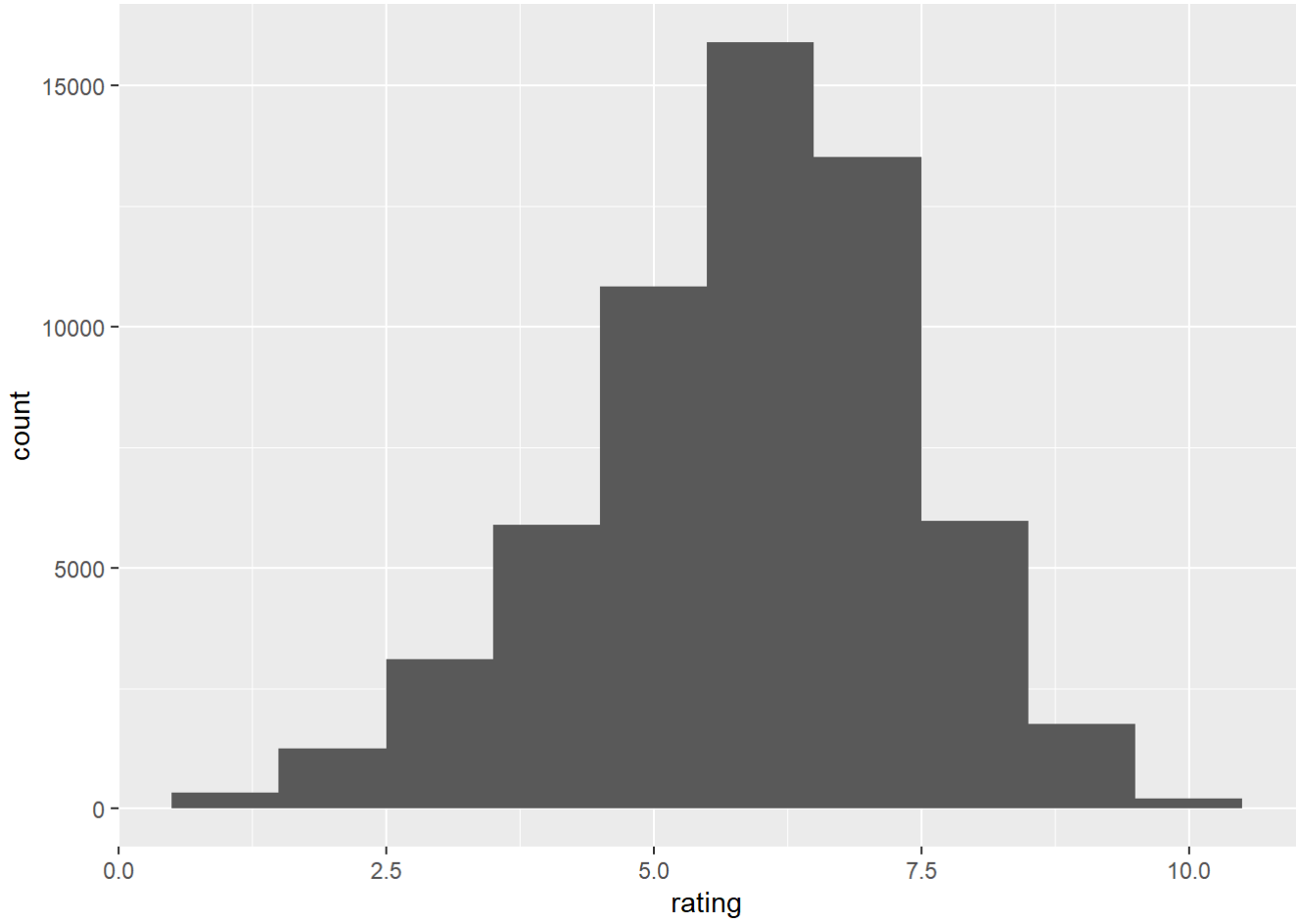
```
# install.packages("ggplot2movies")
library(ggplot2movies)
p <- ggplot(movies,aes(rating))
p <- p+geom_histogram()
p
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# binwidth: histogram의 막대의 넓이 조절
p <- p+geom_histogram(binwidth=1)
p
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



*Histogram에 확률밀도곡선 표현

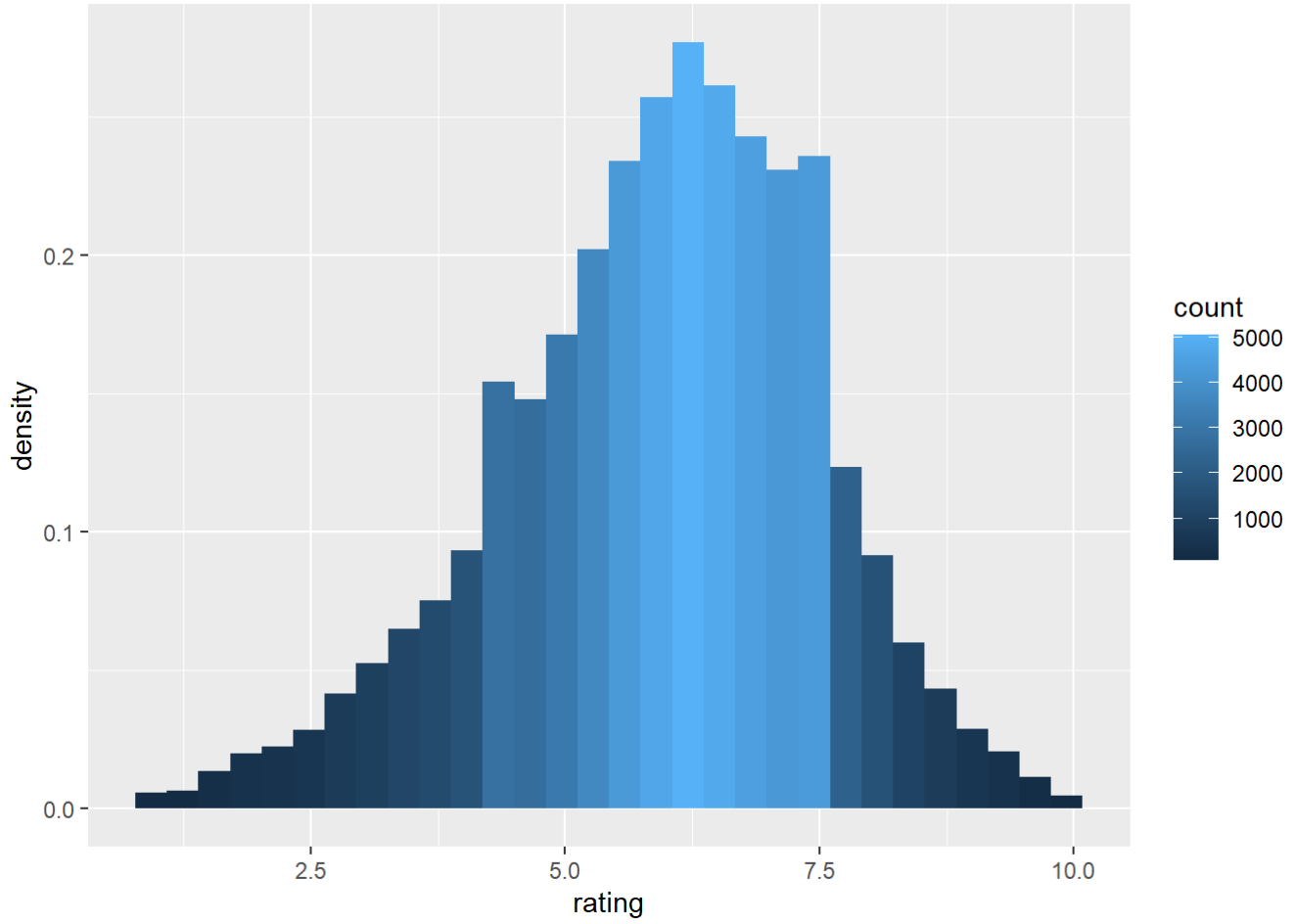
```
head(movies)
```

title <chr>	year <int>	length <int>	budget <int>	rating <dbl>	votes <int>	r1 <dbl>	r2 <dbl>	r3 <dbl>	r4 <dbl>
$	1971	121	*NA*	6.4	348	4.5	4.5	4.5	4.5
$1000 a Touchdown	1939	71	*NA*	6.0	20	0.0	14.5	4.5	24.5
$21 a Day Once a Month	1941	7	*NA*	8.2	5	0.0	0.0	0.0	0.0
$40,000	1996	70	*NA*	8.2	6	14.5	0.0	0.0	0.0
$50,000 Climax Show, The	1975	71	*NA*	3.4	17	24.5	4.5	0.0	14.5
$pent	2000	91	*NA*	4.3	45	4.5	4.5	4.5	14.5

6 rows | 1-10 of 24 columns

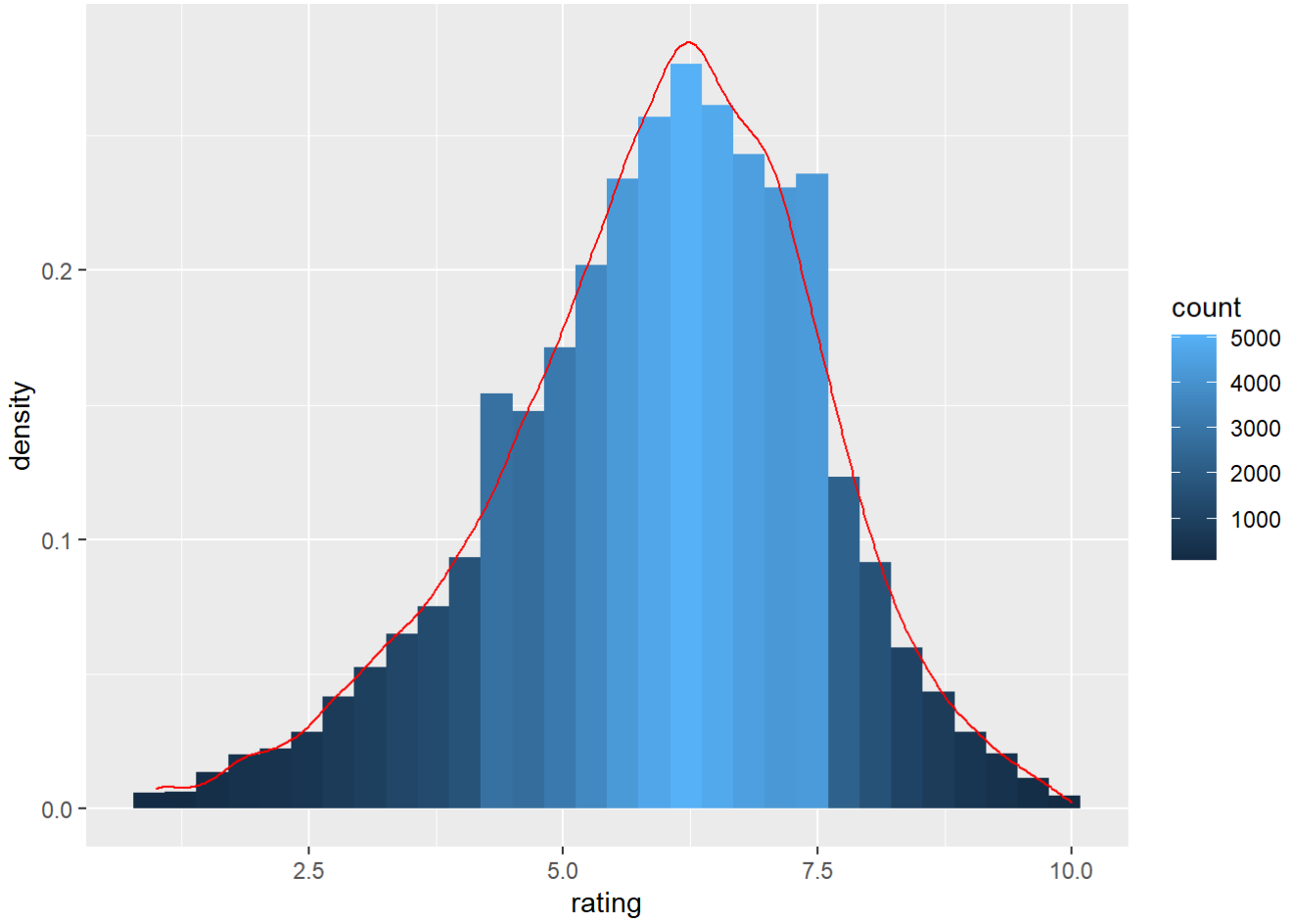
```
p <- ggplot(movies,aes(rating))
p <- p + geom_histogram(aes(y=..density..,fill=..count..)) ; p#히스토그램 그리기 =
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



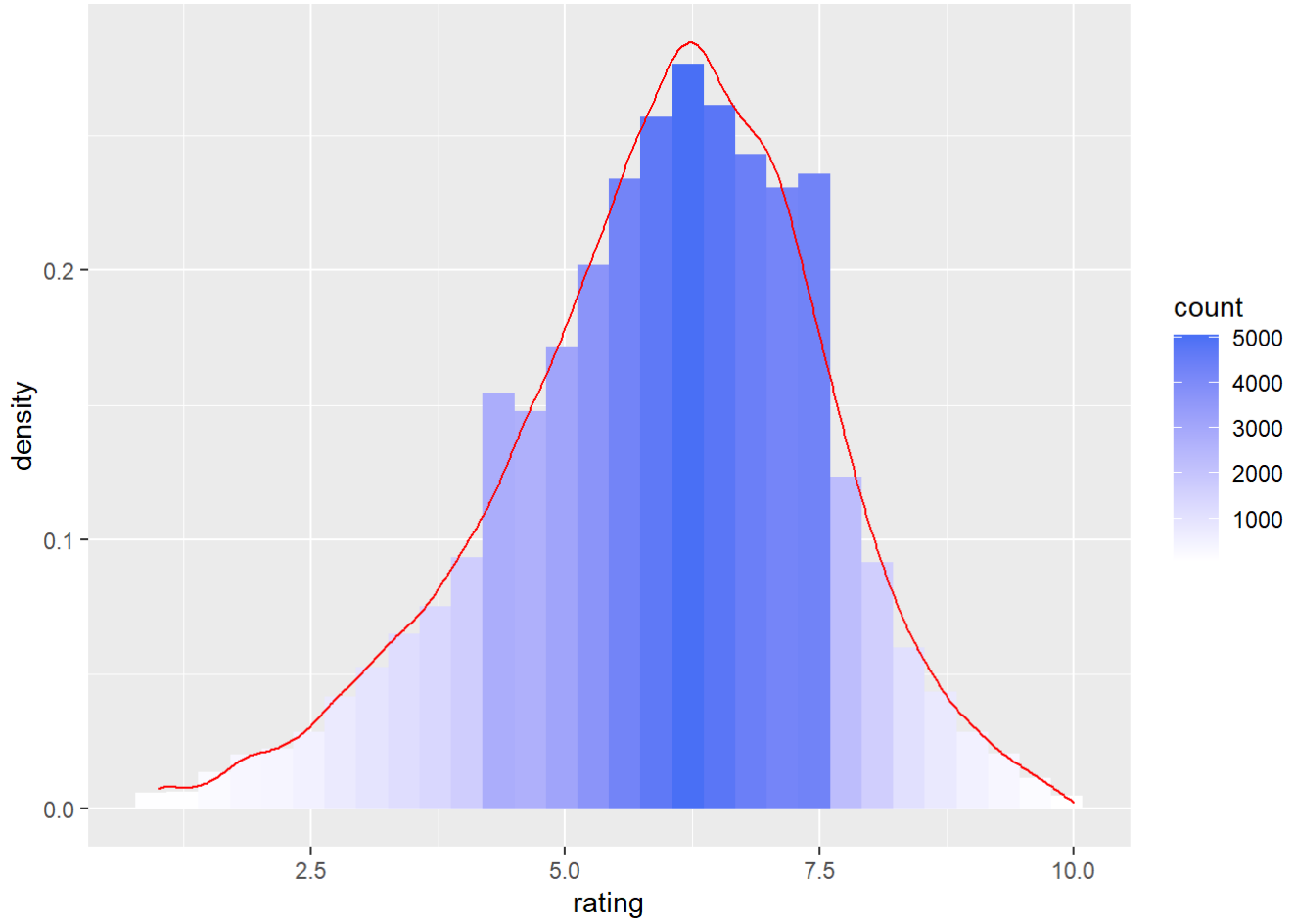

```
p <- p + geom_density(colour="red") ; p#확률밀도곡선 그리기
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



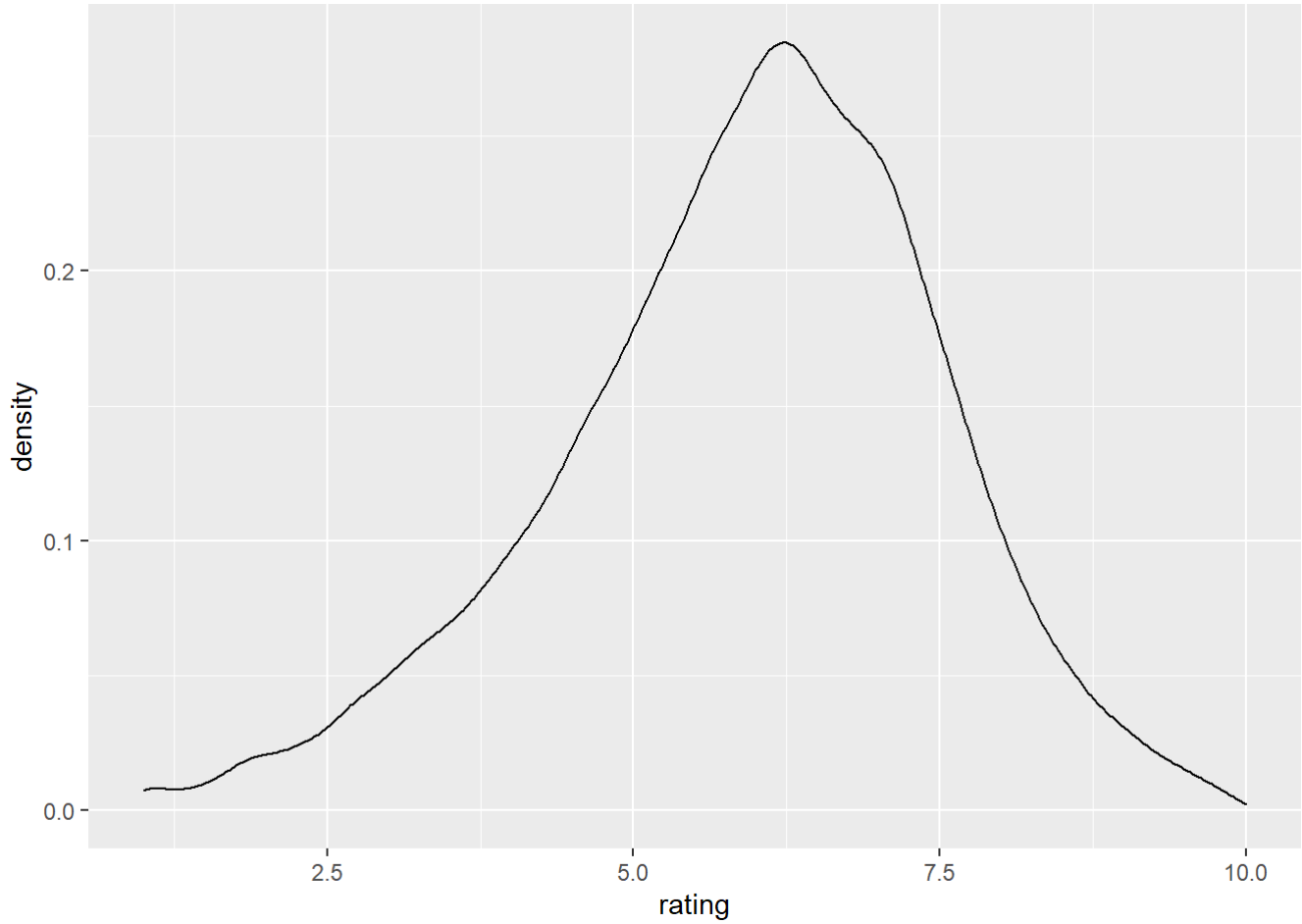

```
p <- p + scale_fill_gradient(low="white",high="#496ff5") ; p#그라데이션 추가
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



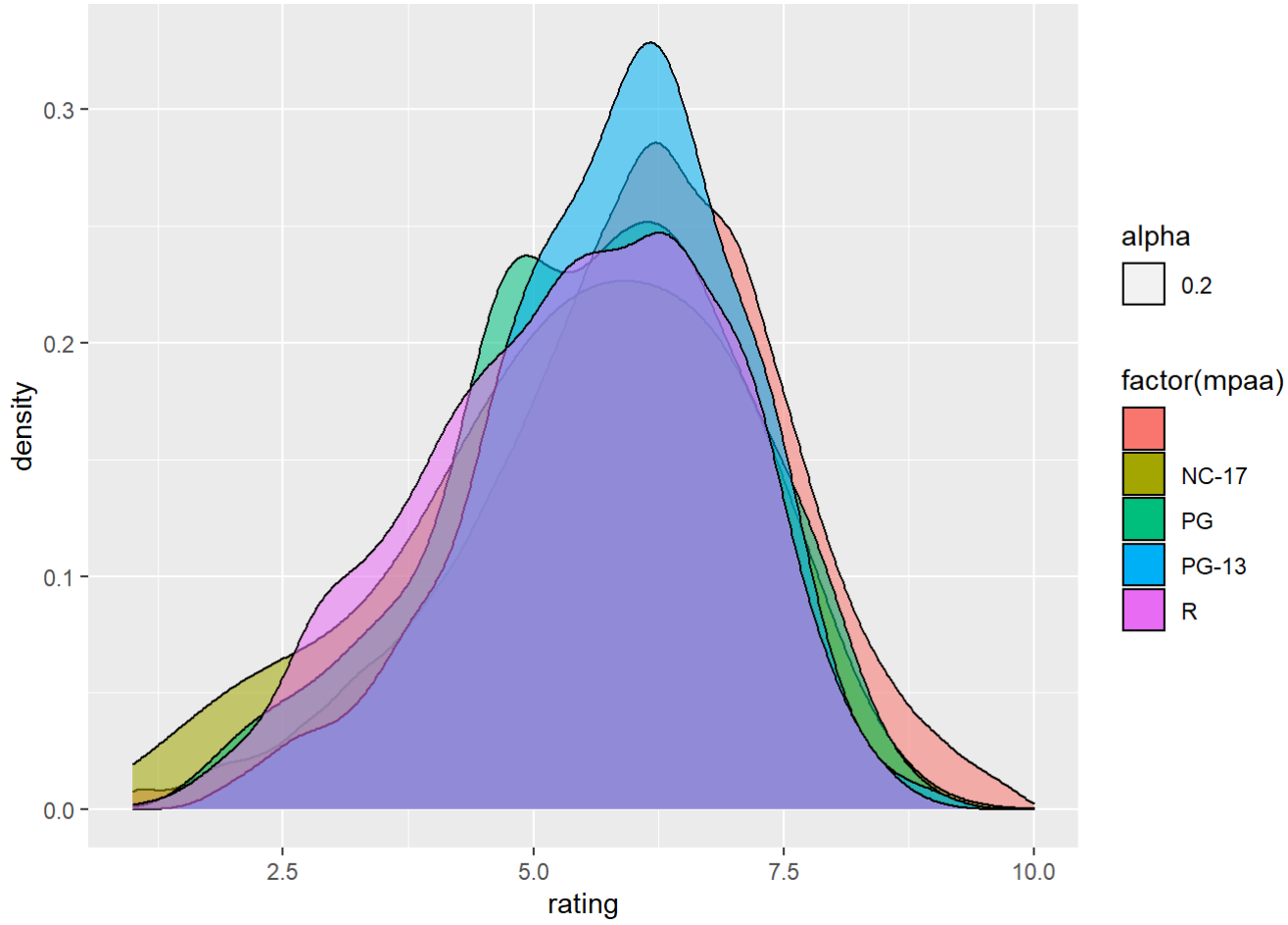
2.2.5. geom_density() 확률밀도곡선

```
p <- ggplot(movies,aes(rating))
p <- p +geom_density()
p
```



- 여러개의 확률밀도곡선 그리기
- aes(fill=factor(mpaa))인수로 mpaa를 분류한 rating의 확률밀도곡선 그리기

```
p <- ggplot(movies,aes(rating))
p <- p + geom_density(aes(fill=factor(mpaa),alpha=0.2))
p
```



3.leaflet

- leaflet 패키지는 interactive한 그래프를 그릴 수 있는 패키지
- 구글맵과 오픈스트리트 맵을 이용하여 공간데이터를 시각화 함

```
#install.packages(leaflet)
library(leaflet)
```

```
## Warning: package 'leaflet' was built under R version 3.6.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
m = leaflet() %>% addTiles()
m = m %>% setView(127.0462, 37.2830, zoom =15) #아주대 경도위 위도 설정
m %>% addPopups(127.0462, 37.2830, 'Here is Ajou University!')
```

leaflet 예제

- addTiles()함수와 addAwesomeMarkers함수 사용
- 서울시 교통 돌발상황 조회 서비스 데이터 중 일부를 사용하여 지도에 표시하기

데이터로드 및 시각화

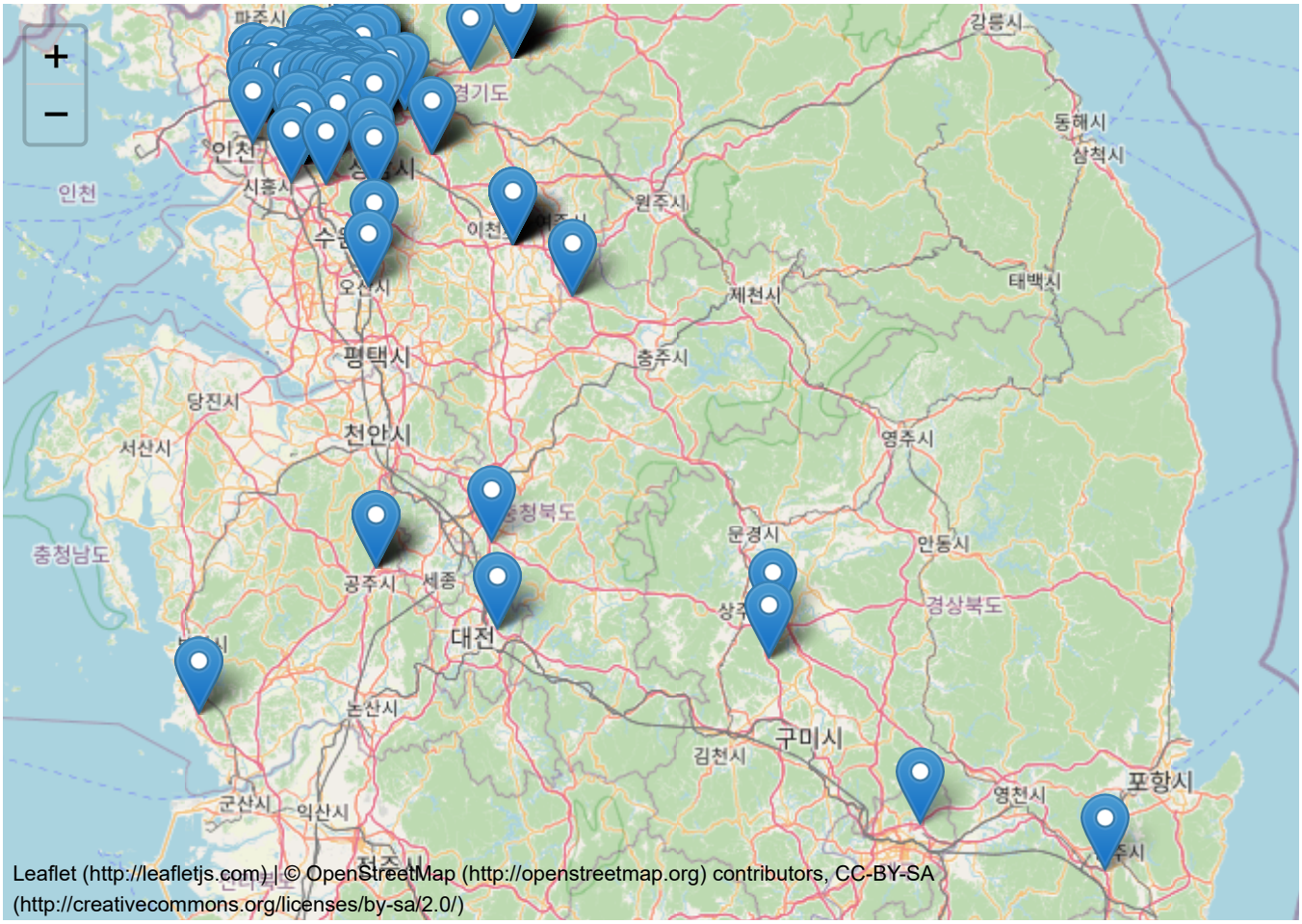
```
traffic <- read.csv("traffic.csv", fileEncoding="utf-8")
range(traffic$start.pos.x) #돌발상황 시작점 경도
```

```
## [1]   0.0000 129.1827
```

```
range(traffic$start.pos.y) #돌발상황 시작점 위도
```

```
## [1]  0.00000 37.69728
```

```
traffic1 <- traffic[traffic$start.pos.x!=0 & traffic$start.pos.y!=0,] #na값 제거(0인값)

leaflet(traffic1) %>% addTiles() %>%
addAwesomeMarkers(~start.pos.x, ~start.pos.y)
```



출처 ###### 1. R을 활용한 데이터 시각화 - 유충현,홍석학 ###### 2.ggplot2 - 김성근 ###### 3.R라뷰 - 서진수

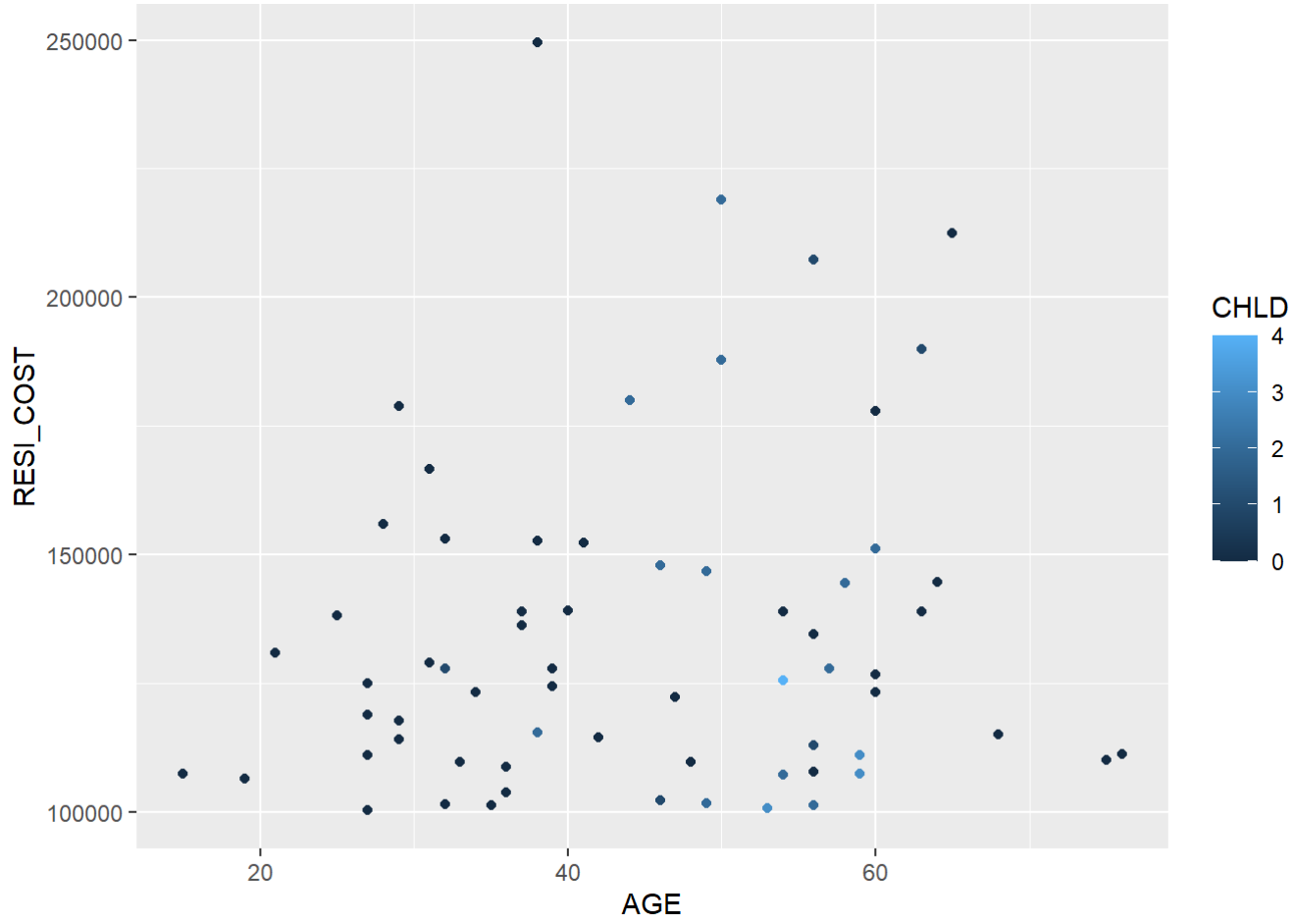
PR11 연습문제

- 주어진 BGCON_CUST_DATA.csv 파일을 사용하여 다음을 해결하시오.
- BGCON_CUST_DATA.csv는 2016년 빅콘테스트에서 제공된 보험사기여부 데이터를 수정 가공한 것 입니다.
- 각 열 이름의 의미는 다음과 같습니다.
    - ID : 고객을 구분하는 고유번호
    - SIU : Y의 경우 보험사기자, N의 경우 일반고객
    - GENDER : 성별 1은 남성, 성별 2는 여성
    - AGE : 고객 연령
    - RESI_COST : 고객의 거줒 주택가격 추정값 (단위:만원)
    - FP_CA : (당사 FP로서의) Y:경력 있음, N:경력 없음
    - RGST : 최초 당사의 고객으로서의 등록 년월
    - CTPR : 고객의 거주 시/도
    - WEDD : Y :결혼함,N: 결혼 안함(계약 당시에는 결호하지 않았던 상태 포함)
    - CHLD : 고객의 자녀 수
    - DMND_AMT : 청구금액
    - PAYM_AMT : 지불금액

문제 1

- 거주지 가겨이 100000 이상인 고객에 한하여 다음을 생성하시오.

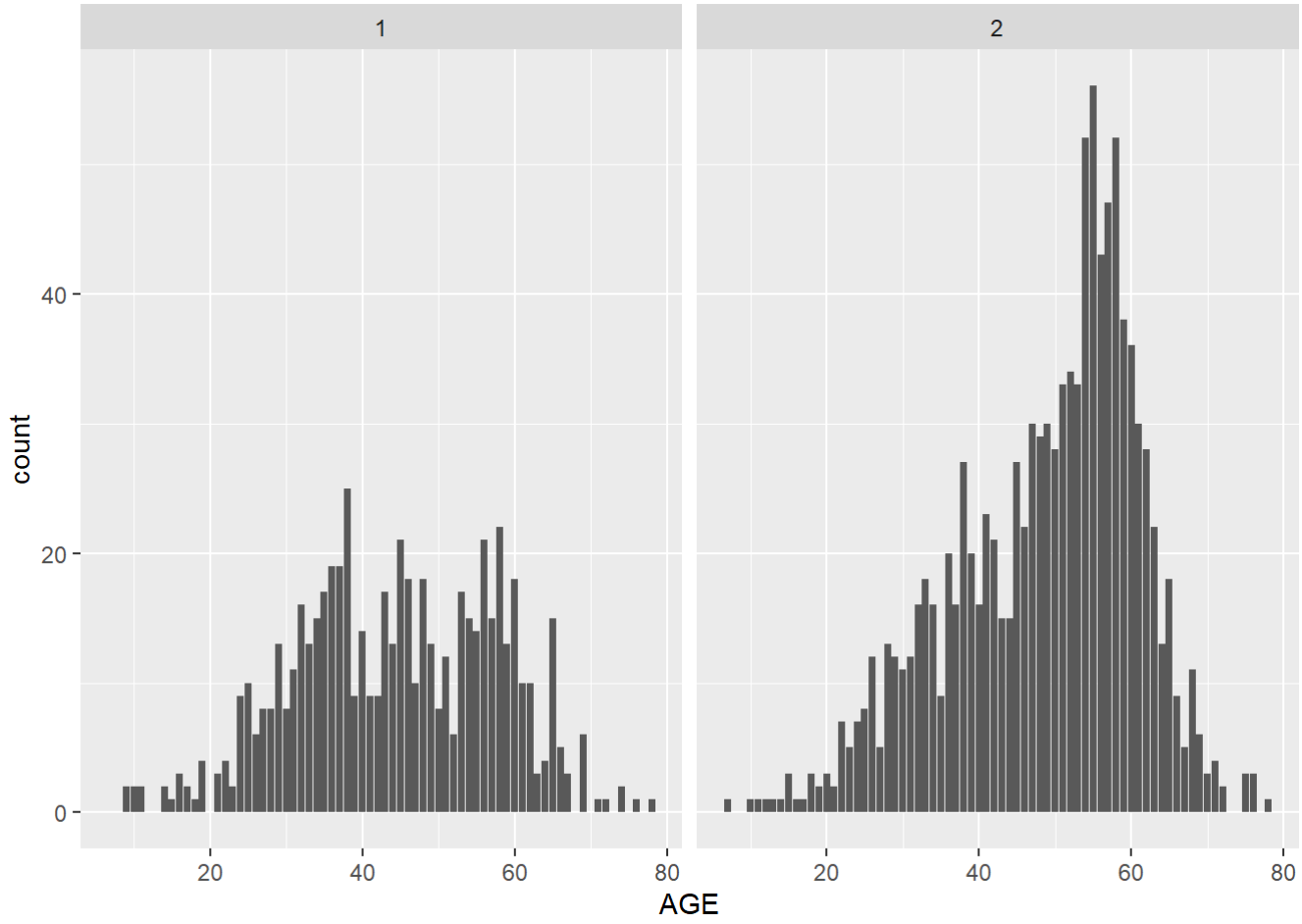
```
setwd("C:WWUsersWWJSWWWDesktopWW강의자료WWR프로그래밍WWR 실습 및 과제WWPR")
BC_cust<- read.csv("BGCON_CUST_DATA.csv")

BC_cust1 <- BC_cust[BC_cust$RESI_COST>100000,]

library(ggplot2)

p <- ggplot(data=BC_cust1,aes(x=AGE,y=RESI_COST,colour=CHLD))

p <- p + geom_point()

p
```



문제2

- 보험사기 고객에 대하여 다음을 생성하시오(성별에 따라 구분할 것)

```
BC_cust2 <- BC_cust[BC_cust$SIU=="Y",]

p <- ggplot(BC_cust2,aes(AGE))
p <- p + geom_bar()
p <- p + facet_grid(~GENDER) ; p
```



문제 3

- 보험사기 고객중 지불금액이 4000000 이상이고 14000000 이하인 고객에 대하여 다음을 생성하시오.

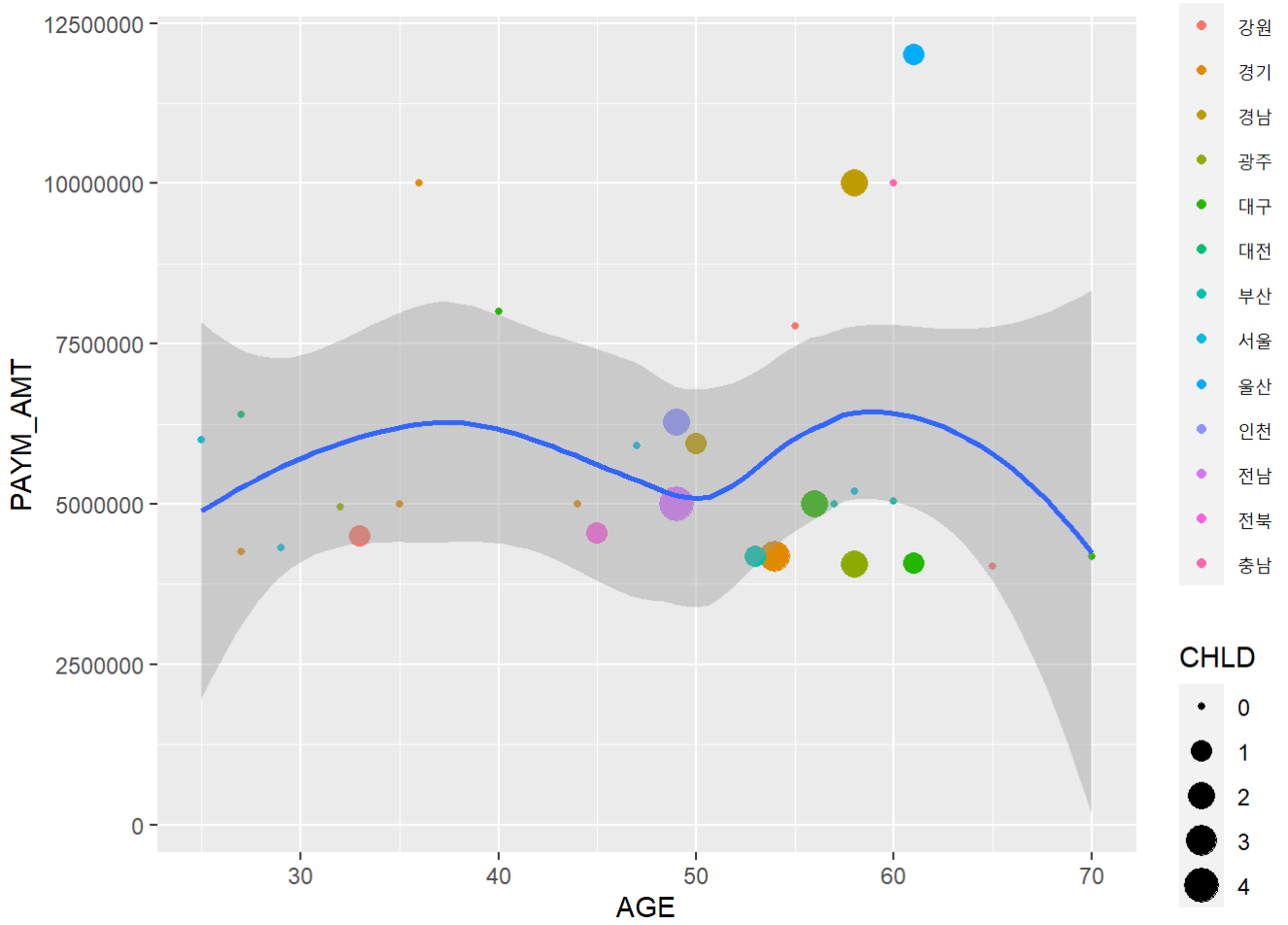
```
BC_cust3 <- BC_cust2[BC_cust2$PAYM_AMT > 4000000 & BC_cust2$PAYM_AMT < 14000000,]

BC_cust3
```

	ID <int>	SIU <fctr>	GENDER <int>	A... <int>	RESI_COST <int>	FP_CA <fctr>	RGST <int>	CTPR <fctr>	WE... <fctr>
23	6849	Y	2	65	3793	Y	199809	강원	N
209	21885	Y	1	33	14274	N	200405	강원	Y
318	11797	Y	1	55	3129	N	200306	강원	N
1564	2209	Y	2	44	15555	N	201312	경기	Y
3106	6692	Y	1	35	20972	N	201202	경기	N
3195	1258	Y	1	36	9745	N	200306	경기	N
3663	7430	Y	2	54	9504	N	200306	경기	Y
4209	19867	Y	1	27	8703	N	200705	경기	N
5289	8229	Y	2	58	26111	N	200102	경남	Y
5558	14946	Y	1	50	12083	N	200104	경남	Y

1-10 of 29 rows | 1-10 of 13 columns Previous 1 2 3 Next

```
p <- ggplot(data=BC_cust3,aes(x=AGE,y=PAYM_AMT))
p <- p + geom_point(aes(colour=CTPR,size=CHLD))

p <- p+geom_smooth(method="loess")
p
```

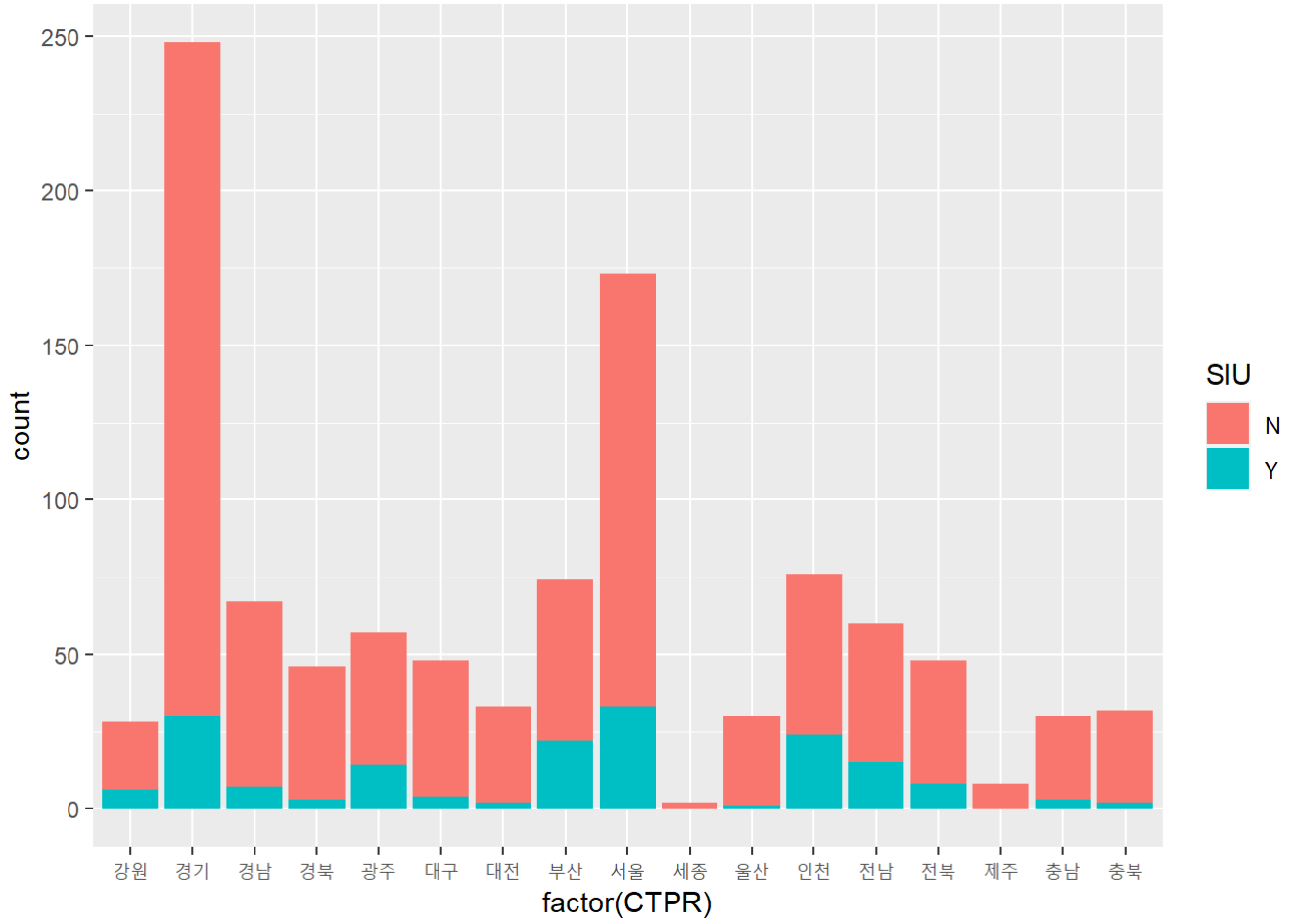
```
## `geom_smooth()` using formula 'y ~ x'
```



문제 4

- 당사 FP로서의 경력이 있는 고객에 대하여 다음을 생성하시오.

```
BC_cust4 <- BC_cust[BC_cust$FP_CA=="Y",]
p <- ggplot(BC_cust4,aes(factor(CTPR)))
p + geom_bar(aes(fill=SIU))
```

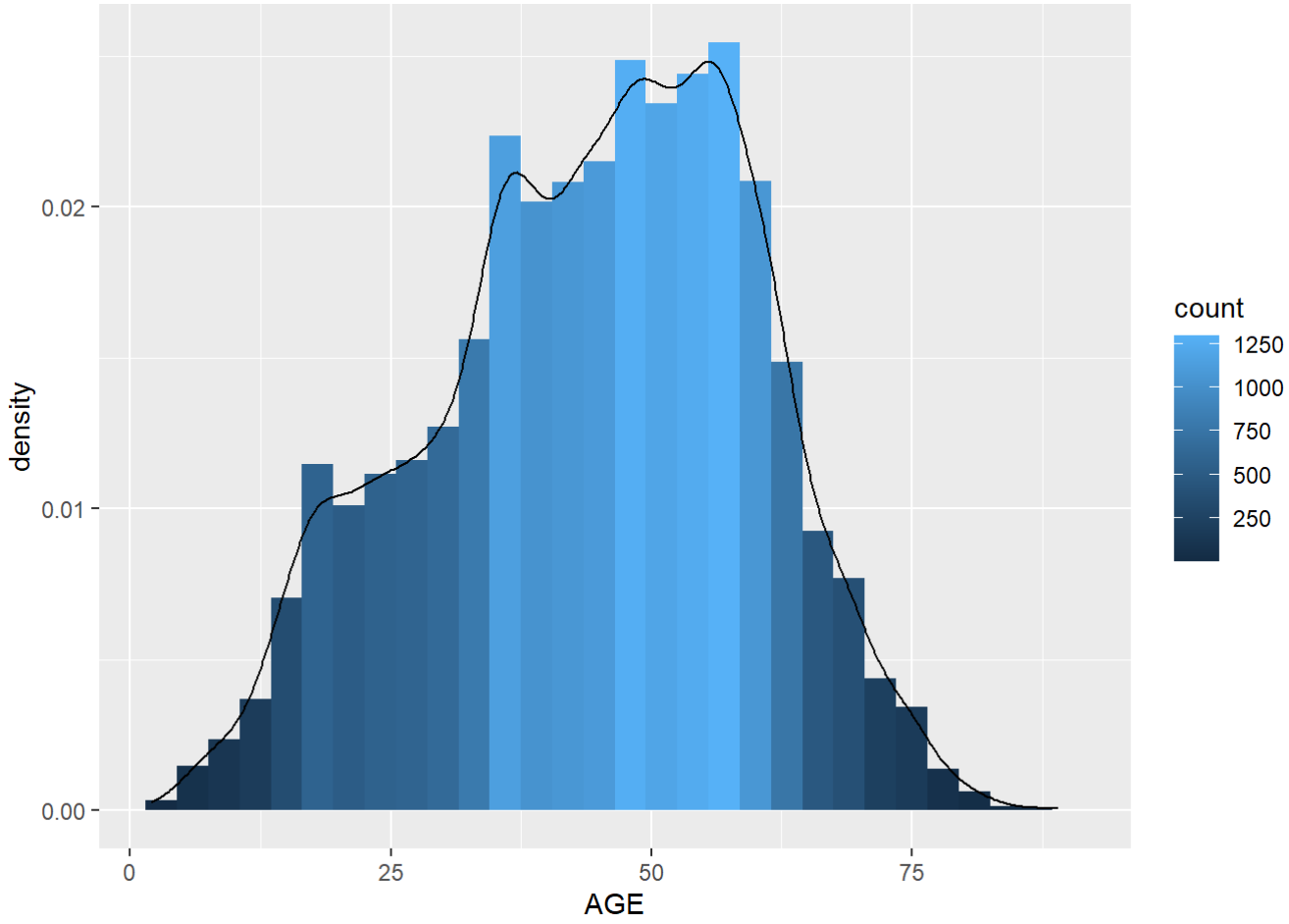


문제 5

- 보험 사기고객이 아닌 고객에 대하여 다음을 생성하시오

```
BC_cust5 <- BC_cust[BC_cust$SIU=="N",]

p <- ggplot(BC_cust5,aes(AGE))
p <- p + geom_histogram(aes(y=..density..,fill=..count..))
p <- p + geom_density(colour="black")
p
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



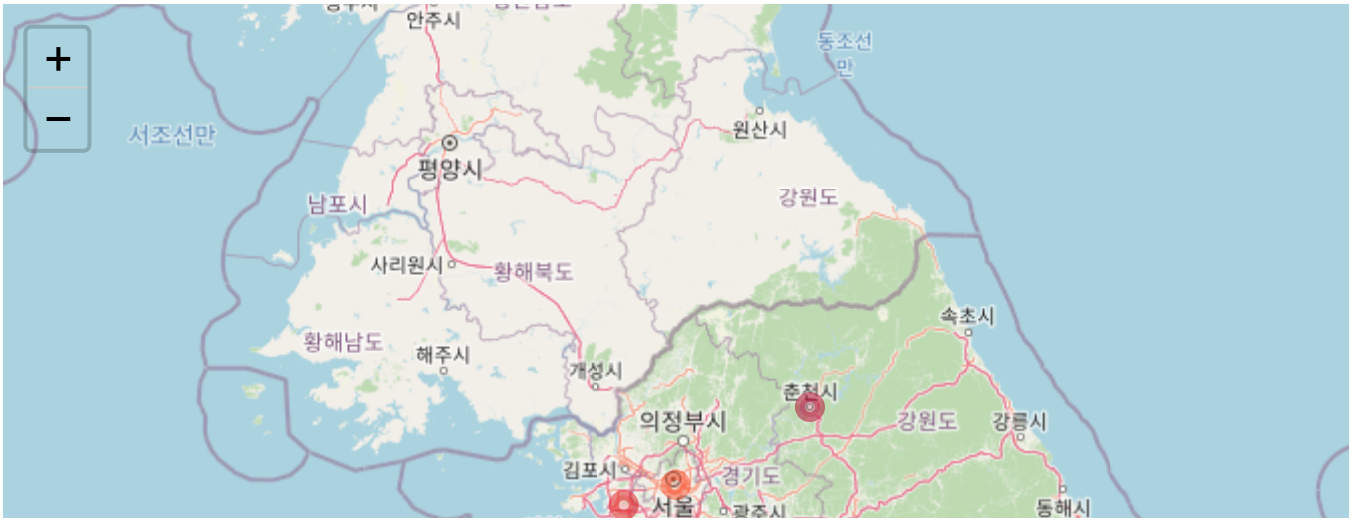
문제6

- 각 시도별 보험사기 고객 비율(per)를 구하여 다음을 생성하시오.

```
library(leaflet)
library(dplyr)

setwd("C:WWUsersWWJSWWWDesktopWW강의자료WWR프로그래밍WWR 실습 및 과제WWPR")

BC_cust<- read.csv("BGCON_CUST_DATA.csv") #읽어오기

BC_cust1 <- select(BC_cust,CTPR,SIU) # 지역과 보험사기여부만으로 데이터프레임을 재구성합니다




ratiofunc <- function(a,b) {  #각 시도별 보험사기 백분율을 계산하기위한 함수를 정의합니다.(a=데이터셋, b=지역명 )
  ratio1 = NULL
  ratio1.1 = NULL
  ratio1 <- select(subset(a,CTPR==b),SIU)
  ratio1.1 <- select(subset(ratio1,SIU=="Y"),SIU)
  c <- length(ratio1.1$SIU) / length(ratio1$SIU)
  return(c)

}


Region <-  c("강원","경기","경남","경북","광주","대구","대전","부산","서울","세종","울산","인천","전남","전북","제주","충남","충북") #지역명 변수 선언

ratioGroup <- 0     #앞서 정의했던 함수를 반복문으로 각 시도별 보험사기 백분율을 한번에 구해줍니다
for (i in 1:length(Region)){
  ratioGroup[i] <- ratiofunc(BC_cust1,Region[i])
}

FF <- data.frame(Region,ratioGroup)   #보험사기 백분율과 앞서 선언한 지역명변수를 FF 데이터프레임으로 구성합니다.
FF <-FF[order(FF$ratioGroup),] #보험사기백분율의 크기순으로 정렬합니다.
rownames(FF) <- NULL   #재정렬한 순서에 맞춰 행이름을 재설정해줍니다


PER<-0 #총 17지역이므로 17중 순위에맞게 백분위를 계산해줍니다 PER변수에 백분위가 차례로 들어갈 것입니다.
totalRG <- for (i in 1:17){
  PER[i] <- i/17 *100
}

FF<- data.frame(FF,PER) #보험사기백분율 FF데이터프레임과 PER함수를 묶어줍니다

#-- leaflet 활용 시작------------------/

pal3<-colorBin(palette="YlOrRd", domain=c(0,100), bins = 8, pretty=F, alpha = T) #0~100을 8클래스로 나눈 파레트를 만듭니다.
pal <-colorFactor(palette = "YlOrRd",domain=NULL)
#-----
class <- c("c1","c1","c2","c2","c3",'c3',"c4",'c4','c5','c5','c6','c6','c7','c7','c8','c8','c8') #기존 데이터프레임의 백분위를 범례의 구간별로 할당할 벡터 class를 선언해줍니다
# C1: 0 - 15
# C2: 15- 25
```

```
# C3: 25 - 38
# C4: 38 - 50
# C5: 50 - 62
# C6: 62 - 75
# C7: 75 - 88
# C8: 88 - 100


FF<- data.frame(FF,class) #FF와 class를 병합합니다
#-------------------------------------
L.L <- read.csv("KOR_LAT_LON.csv") #위.경도 데이터를 불러옵니다

colnames(L.L)<- c("Region","Lat","Lon") #위경도데이터의 열이름을 바꿔줍니다



GG<-NULL
HH<-NULL

for (i in 1:length(FF$Region)){  #앞서 FF에서 범죄백분율 크기 기준으로 정렬되었던 지역명의 순서대로 맞추어서 위경도데이터를 재구성합니다.
  GG<- L.L[L.L$Region==FF$Region[i],]
  H <- GG[,2:3]
  HH<-bind_rows(HH,H)
}

FF <- bind_cols(FF,HH)  #위의 반복문으로 FF순서에맞춰 재구성된 HH를 FF에 합쳐줍니다
#-----------------------------------
m=0
m = leaflet() %>% addTiles() %>%   # leaflet 활용부분입니다.
  setView(126.9860,37.54100,zoom=7) #snapshot에서 보여줄 경도와 위도 확대정도를 설정합니다

m = m %>% addLegend(pal = pal3,  #addLegend는 범례생성함수입니다.
                    values  = FF$ratioGroup,
                    position = "bottomright",
                    title = "ratio",
                    labFormat = labelFormat(suffix="%",between="%&ndash;")) #맨끝에 and '-'뒤에 %를 붙여주도록 설정합니다
m %>% addCircleMarkers(data=FF,radius=5,color= ~pal(class),)   #문제의 예시에 맞게 크기를 radius로 조절하고 FF 데이터프레임에 맞춘 circles를 찍어줍니다
```

```
## Assuming "Lon" and "Lat" are longitude and latitude, respectively
```




ratio
0.0%–12.5%
12.5%–25.0%
25.0%–37.5%
37.5%–50.0%
50.0%–62.5%
62.5%–75.0%
75.0%–87.5%
87.5%–100.0%
인천
수원시
이천시
원주시
제천시
태백시
당진시
천안시
충주시
영주시
서산시
안동시
충청남도
공주시
세종
상주시
경상북도
보령시
대전
논산시
구미시
익산시
김천시
포항시
대구
전라북도
대한민국
경주시
정읍시
밀양시
울산
Leaflet (http://leafletjs.com) | © OpenStreetMap (http://openstreetmap.org) contributors, CC-BY-SA (http://creativecommons.org/licenses/by-sa/2.0/)