

M23 - ggplot2

learningSpoonsR@gmail.com



시작하기

I. Scatterplot (산점도)

II. Faceting

III. 중첩

IV. Discussion

V. Grammar of Graphics

VI. More Plots

VII. Summary

VIII. 연습 문제

시작하기

시각화

‘간단한 그래프는 다른 어떤 장치보다 데이터 분석가의 마음에 더 많은 정보를 제공합니다.’
- John Tukey

효과적인 차트란 어떤 것일까요?

- ▶ 사실을 왜곡하지 않고 올바른 정보를 전달하세요
- ▶ 단순하지만 우아합니다. 많은 생각을 하지 않아도 이해할 수 있습니다.
- ▶ 미학은 정보를 가리기보다는 정보를 지원합니다.
- ▶ 너무 많은 정보를 담지 않습니다.

데이터셋 불러오기 mpg

- ▶ **ggplot2** 패키지에 내장된 데이터셋 (`library(ggplot2)`를 실행하면 사용가능)
- ▶ 1999년 부터 2008년 까지의 38개 차종 연비 데이터
- ▶ 데이터셋 **diamond**, **iris**와 함께 R에서 가장 많이 예제로 쓰임

변수이름	설명
manufacturer	
model	차종
displ	엔진크기
year	
cyl	기통

변수이름	설명
drv	전륜(f)/후륜(r)/사륜(4)
cty	도심 마일리지
hwy	고속도로 마일리지
fl	연료종류 (fuel)
class	클래스 (중형, 트럭, SUV,...)

```
library(ggplot2)
? mpg # getting help from web
help(mpg) # getting help from help panel
```

```
class(mpg)
## [1] "tbl_df"      "tbl"        "data.frame"
```

- ▶ mpg는 여러개의 class의 특징을 가지고 있습니다.
- ▶ tbl_df는 data.frame의 업그레이드 버전입니다.
 - ▶ data.frame의 모든 기능 사용
 - ▶ 몇 가지 기능 추가

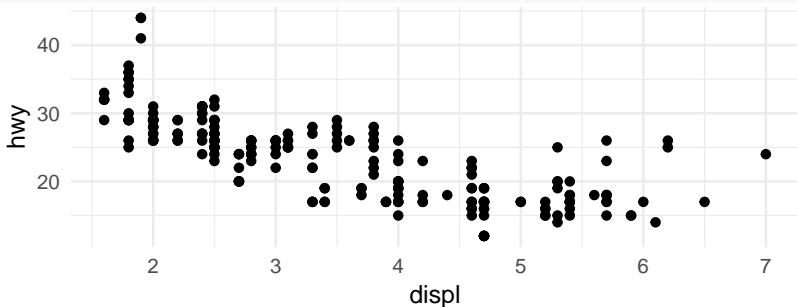
```
str(mpg)

## Classes 'tbl_df', 'tbl' and 'data.frame':   234 obs. of  11 variables:
## $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
## $ model       : chr  "a4" "a4" "a4" "a4" ...
## $ displ      : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
## $ year       : int  1999 1999 2008 2008 1999 1999 2008 1999 2008 ...
## $ cyl        : int  4 4 4 4 6 6 6 4 4 4 ...
## $ trans      : chr  "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
## $ drv        : chr  "f" "f" "f" "f" ...
## $ cty        : int  18 21 20 21 16 18 18 18 16 20 ...
## $ hwy        : int  29 29 31 30 26 26 27 26 25 28 ...
## $ fl         : chr  "p" "p" "p" "p" ...
## $ class      : chr  "compact" "compact" "compact" "compact" ...
```

I. Scatterplot (산점도)

- ▶ 엔진이 크면 연비가 안 좋을까요?
- ▶ 엔진 크기(displ)를 x축, 고속도로 마일리지(hwy)를 y축으로 하는 산점도 (scatterplot) 그리기

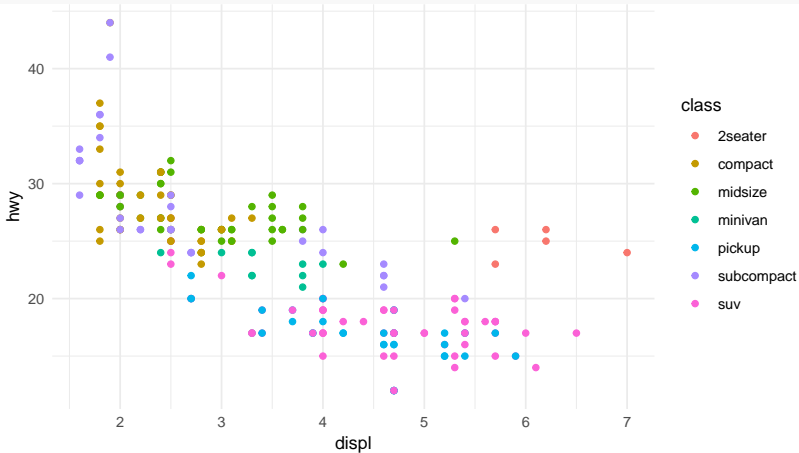
```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy))
```



```
# basic ggplot command  
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPING>))
```



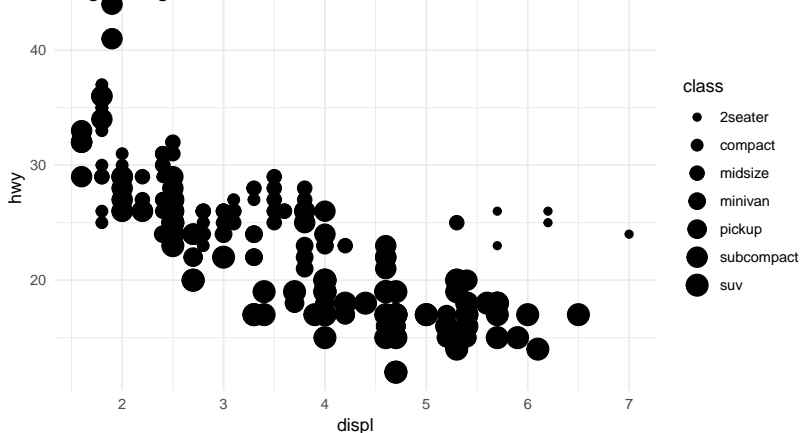
```
a <- ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, color = class))  
a
```



- ▶ a는 ggplot 그래픽 객체
- ▶ color=class: class 변수의 값에 따라 color가 다르게

```
b <- ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy, size = class))
b
```

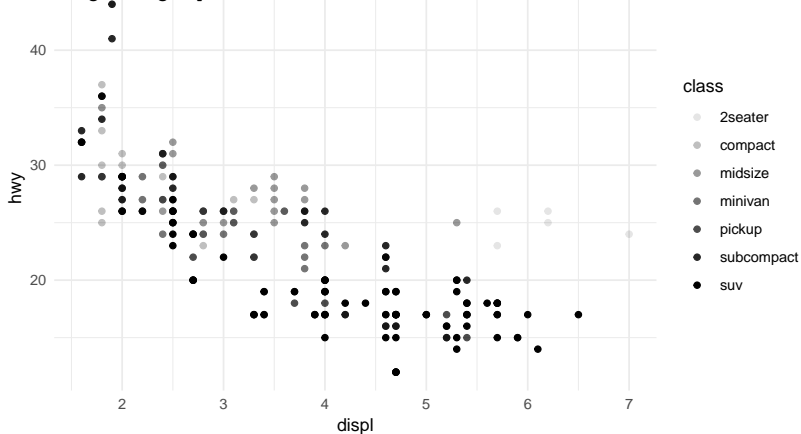
```
## Warning: Using size for a discrete variable is not advised.
```



▶ size=class: class변수의 값에 따라 size가 다르게

```
c <- ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy, alpha = class))
c
```

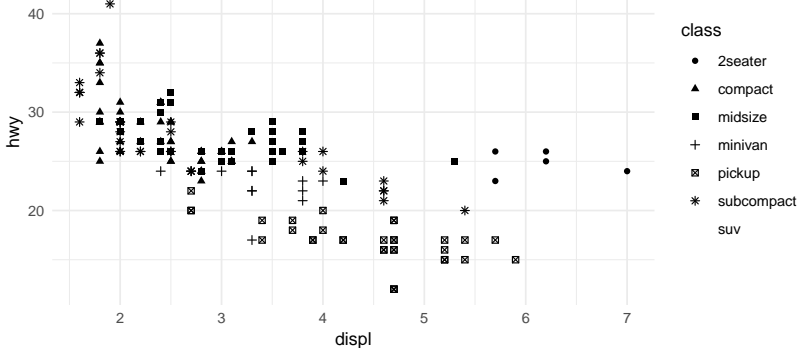
```
## Warning: Using alpha for a discrete variable is not advised.
```



▶ `alpha=class`: class 변수의 값에 따라 `alpha`(진하기)가 다르게

```
d <- ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, shape = class))  
d
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values  
## because more than 6 becomes difficult to discriminate; you have 7.  
## Consider specifying shapes manually if you must have them.  
## Warning: Removed 62 rows containing missing values (geom_point).
```



▶ `shape=class`: class변수의 값에 따라 shape이 다르게

Discussion

a, b, c, d 중에 어느 그림이 가장 효과적인가요?

- ▶ **size, alpha, color, shape**의 사용 기준은 무엇인가요?
- ▶ 일반화 시키실 수 있나요?

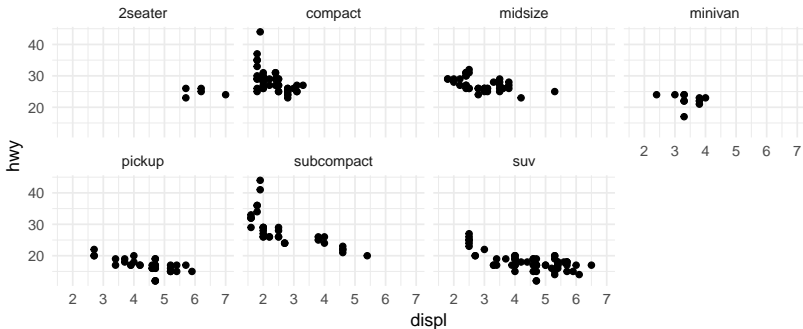
Aesthetics (**aes**) - size, alpha, color, shape

- ▶ Numeric 변수
 - ▶ 수치 변수, 정량적 변수, numeric, quantitative
 - ▶ 크고 작음이 있음, 더할 수 있음
 - ▶ **size**
 - ▶ **alpha**: 상한/하한이 있는 경우에 적합 (eg. 농도, 빈도, 확률)
- ▶ Categorical 변수
 - ▶ 범주형 변수, 정성적 변수, categorical, factor, qualitative
 - ▶ 집합. A,B,C로 치환 가능, 더할 수 없음, 크고 작음이 없음
 - ▶ **color**: 10개 미만의 범주에서 사용
 - ▶ **shape**: 7개 미만의 범주에서 사용, 컬러프린트 없을 때.

II. Faceting

Facets (1 variable) – facet_wrap

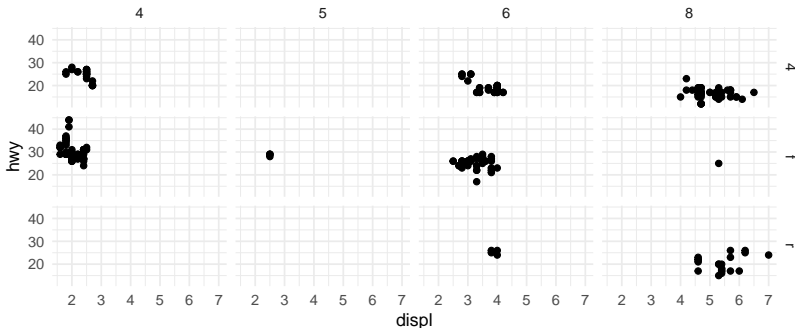
```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  facet_wrap(~ class, nrow = 2)
```



- ▶ `facet_wrap(~ class, nrow = 2)`
 - ▶ 이산변수인 `class`의 값에 따라서 분할하여 배열
 - ▶ `nrow=2`에 의해 2행으로 배열

Facets (2 variables) - facet_grid

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  facet_grid(drv ~ cyl)
```



- ▶ `facet_grid(drv ~ cyl)`
 - ▶ 이산변수인 `drv`와 `cyl`의 값에 따라 분할하여 배열
 - ▶ 2차원의 `grid` 형태로 배열하는데,
 - ▶ `drv`를 y축(세로)으로 `cyl`을 x축(가로)으로 배열

Discussion

▶ Do **facet**

1. Data가 충분히 많을 때
2. Categorical 변수별로 각각의 distribution을 보고 싶을 때

▶ Do Not **facet**

1. 근접 비교를 하고 싶을 때
2. Reader들의 사전 지식 수준이 높을때

▶ Do and Don't **facet**!

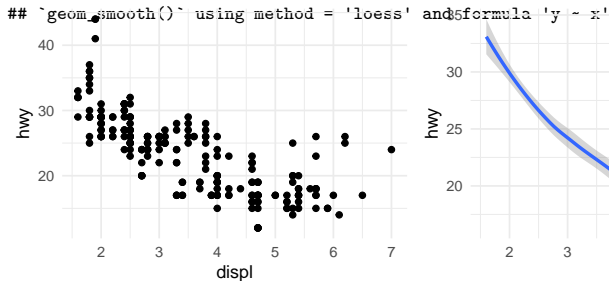
1. No facet → Yes facet 순서로
2. 전체를 보여준 후에 나누어서 보여주며 top-down approach로 표현

III. 중첩

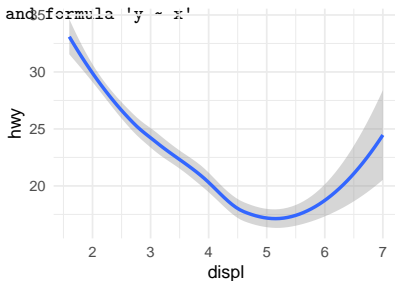
geom_point vs geom_smooth

```
e <- ggplot(data = mpg) + geom_point(mapping = aes(x = displ, y = hwy))
f <- ggplot(data = mpg) + geom_smooth(mapping = aes(x = displ, y = hwy))
```

e



f



▶ geom_point: 점으로 표시

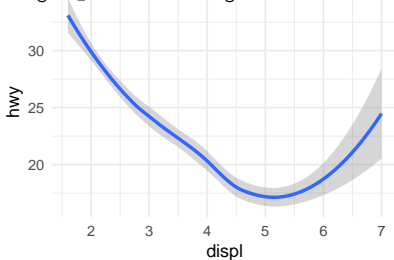
▶ geom_smooth: curve로 표시

mapping의 상속

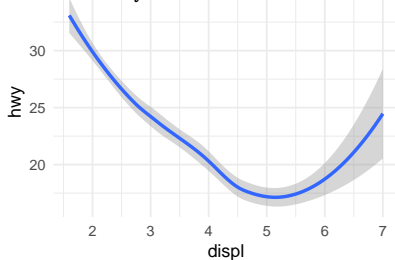
```
g <- ggplot(data = mpg) +
  geom_smooth(mapping = aes(x = displ, y = hwy))
h <- ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
  geom_smooth()
```

g

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



h



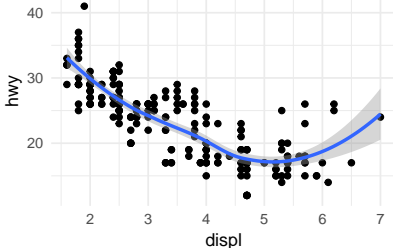
- ▶ g와 h의 명령어의 약간 variation이 있지만, 결과는 동일함
- ▶ h의 geom_smooth()에서 mapping이 없음 → 바로 앞의 ggplot()의 값을 사용

geom_point + geom_smooth

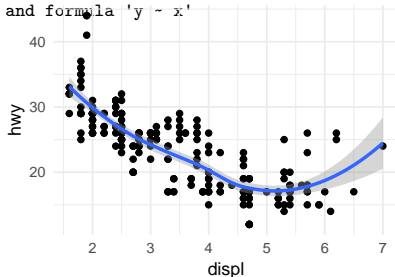
```
i <- ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  geom_smooth(mapping = aes(x = displ, y = hwy))
j <- ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
  geom_point() + geom_smooth()
```

i

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



j

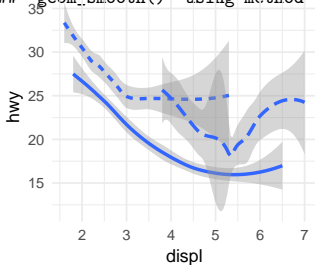


- ▶ i와 j는 동일한 결과를 만듦
- ▶ j의 `geom_point()`와 `geom_smooth()`는 앞의 `ggplot()`에서 `mapping`을 상속받음

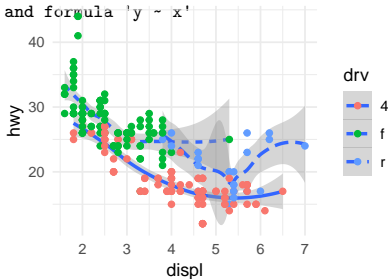
```
k <- ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
  geom_smooth(aes(linetype = drv))
l <- ggplot(data = mpg, mapping = aes(x = displ, y = hwy)) +
  geom_smooth(aes(linetype = drv)) +
  geom_point(aes(color = drv))
```

k

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



l



- ▶ `linetype=drv`는 `drv`값에 따라서 각각 다른 curve를 그림
- ▶ `k`와 `l`의 `geom_smooth()`에는 `x`, `y`의 값이 없지만 이는 `ggplot()`에서 상속 받음
- ▶ `l`의 `geom_point()`도 마찬가지

IV. Discussion

Summary so far

```
ggplot(data = <DATA>) + <GEOM_FUNCTION>(mapping = aes(<MAPPING>))
```

1. Y 변수 결정
 - ▶ 무엇을 설명하고 싶은가?
 - ▶ 목적 (target) 변수가 일반적으로 Y변수가 됨
2. X 변수 결정
 - ▶ 무엇으로 설명할 수 있을까?
 - ▶ 설명 (explanatory) 변수가 X변수가 됨
3. X변수와 Y변수의 성격이 각각 어떠한가?
 - ▶ X변수는 이산/연속?
 - ▶ Y변수는 이산/연속?
 - ▶ `geom_function`이 결정됨
4. X, Y 변수외에 다른 추가할 수 있는 변수가 있을까?
 - ▶ 특성을 추가하여 그림의 정보량을 늘림
 - ▶ `size`, `alpha`, `color`, `shape`, `fill`, `dodge`,...
5. 그래프를 장식
 - ▶ 축, 격자, 레이블, 색깔, 폰트 크기
 - ▶ 좌표 변환
 - ▶ M24의 예제들과 특히 `theme()` 섹션 참조

Carseats와 additivity

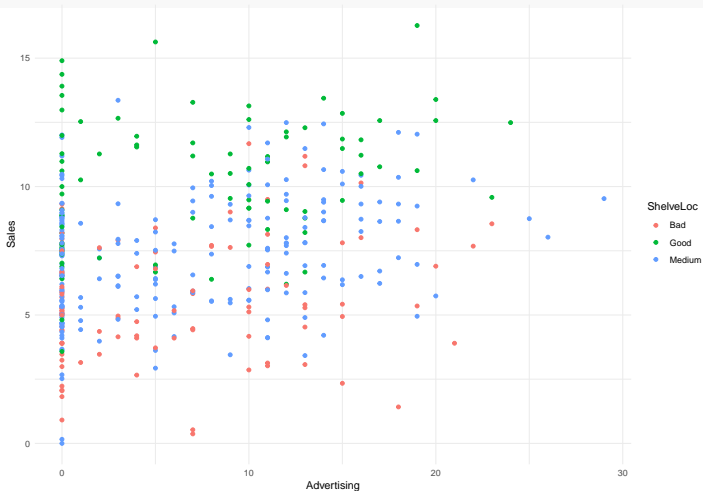
Dataset review

```
library(ISLR)
str(Carseats)
```

```
## 'data.frame':    400 obs. of  11 variables:
## $ Sales      : num  9.5 11.22 10.06 7.4 4.15 ...
## $ CompPrice  : num  138 111 113 117 141 124 115 136 132 132 ...
## $ Income     : num   73 48 35 100 64 113 105 81 110 113 ...
## $ Advertising: num   11 16 10 4 3 13 0 15 0 0 ...
## $ Population : num  276 260 269 466 340 501 45 425 108 131 ...
## $ Price      : num  120 83 80 97 128 72 108 120 124 124 ...
## $ ShelfLoc   : Factor w/ 3 levels "Bad","Good","Medium": 1 2 3 3 1 1 3 2 3 3 ...
## $ Age        : num   42 65 59 55 38 78 71 67 76 76 ...
## $ Education  : num   17 10 12 14 13 16 15 10 10 17 ...
## $ Urban      : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 2 2 1 1 ...
## $ US         : Factor w/ 2 levels "No","Yes": 2 2 2 2 1 2 1 2 1 2 ...
```

ggplot review

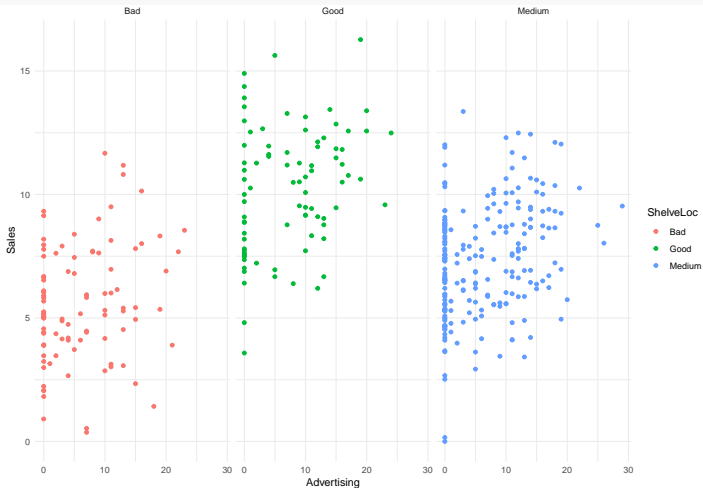
```
a <- ggplot(data = Carseats, aes(x = Advertising, y = Sales)) +  
  geom_point(aes(color = ShelfLoc))  
print(a)
```



Amazingly Additive!

Add features to existing ggplot object by "SIMPLY ADDING"

```
a <- a + facet_wrap(~ ShelfLoc)  
print(a)
```



기능을 더하면 plot에 더해집니다!

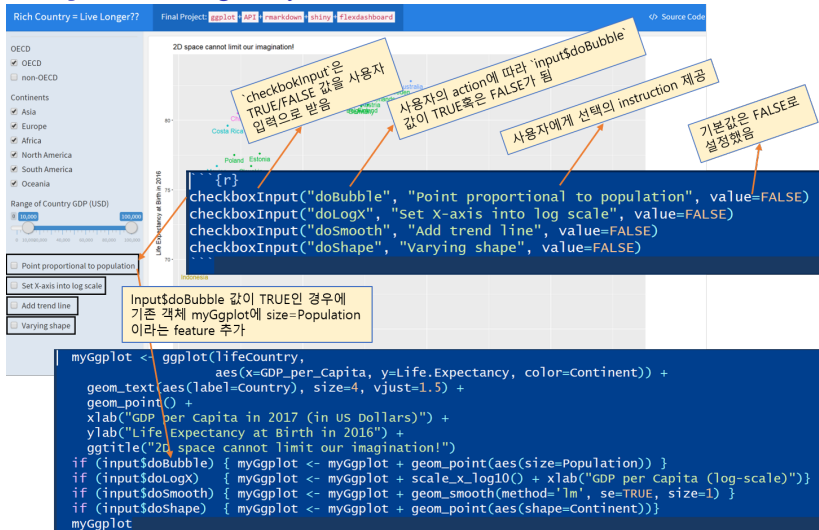
```
a <- ggplot(data = Carseats, aes(x = Advertising, y = Sales)) +  
  geom_point(aes(color = Urban))
```

```
a <- a + facet_wrap(~ Urban)
```

rmarkdown과 함께 사용하면 매우 강력합니다.

- ▶ 스토리텔링 기반 발표 및 보고서 작성
 - ▶ 점점 깊이 들어가면서 발표 (top-down)
 - ▶ 좌우 혹은 상하로 배열하여 보고서 작성
 - ▶ 좌우 혹은 상하로 배열하여 대시보드 작성 (M32)
 - ▶ 사용자가 선택할 수 있게 대시보드 작성 (M33)

Example: M41-longevity



실습과제 1

1. `C:/LS-DS/classProject`라는 폴더를 만드세요.
2. Rstudio를 열어서 rmarkdown의 html형식에 해당하는 `.Rmd`파일을 만드세요.
3. `classProject1.Rmd`로 위 폴더에 저장하세요.
4. Rstudio를 닫고 `classProject1.Rmd`를 더블클릭하여 실행하면 working directory가 자동으로 위의 폴더가 됩니다.
5. github `M41-longevity/data`에서 `lifeCountry.csv`파일을 다운받아서 위 폴더에 넣으세요.
6. 이제 분석을 위한 모든 준비가 끝났습니다.
7. Infile/Preprocessing을 해야합니다. `classProject1.Rmd`에서 `lifeCountry.csv`를 불러와서 `data.frame`으로 저장합니다.
8. 각 나라의 GDP와 기대수명에 대해서 산점도를 그리고 대륙에 따라서 점의 색깔이 달라지게 해보세요.
9. 자유롭게 분석을 시작해보세요.
10. html로 제작해 이메일을 보내주세요. `learningSpoonsR@gmail.com`

V. Grammar of Graphics

GG? - Grammar of Graphics

► Features

1. 독립적이고 더할 수 있는 구성 요소들로 그래픽을 표현
2. 개발과정에서 그래프의 특징을 한 가지 씩, 반복적으로 바꾸면서 그래프를 만들어 감
3. 생각의 흐름, 스토리텔링과 연계시켜 interactive graphics와 잘 조화됨

► Motivation

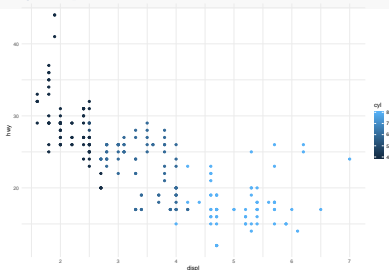
1. 그래픽 관련 패키지와 함수는 단지 특수 경우의 모음일 뿐??
2. 요리 백과사전을 다 읽는 것 vs. 물, 기름, 불을 익히고 백과사전을 보유하는 것

► Advantage

1. 새로운 graphics를 만들어 내는 아이디어
2. Data를 보는 직관과 상상력
3. 효과적인 Communication

구성 요소

```
library(ggplot2)
ggplot(mpg) +
  aes(x = displ, y = hwy, color = cyl) +
  geom_point()
```



► Aesthetics `aes()`

1. position
2. size
3. color
4. shape

► Geometric Object (`geom_()`)

1. Scatterplot - **point**
2. Bubblechart - **point** (size)
3. Barchart - **bar** (frequency)
4. Box-and-whisker plot - **boxplot** (distribution)
5. Line chart - **line**

Behind the scene

data

```
head(mpg[,c("displ", "hwy", "cyl")],10)
```

	displ	hwy	cyl
1	1.80	29	4
2	1.80	29	4
3	2.00	31	4
4	2.00	30	4
5	2.80	26	6
6	2.80	26	6
7	3.10	27	6
8	1.80	26	4
9	1.80	25	4
10	2.00	28	4

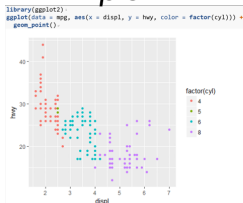
aes

x	y	colour
1.8	29	4
1.8	29	4
2.0	31	4
2.0	30	4
2.8	26	6
2.8	26	6
3.1	27	6
1.8	26	4

geom

x	y	colour	size	shape
0.037	0.531	#F8766D	1	19
0.037	0.531	#F8766D	1	19
0.074	0.594	#F8766D	1	19
0.074	0.562	#F8766D	1	19
0.222	0.438	#00BFC4	1	19
0.222	0.438	#00BFC4	1	19
0.278	0.469	#00BFC4	1	19
0.037	0.438	#F8766D	1	19

plot



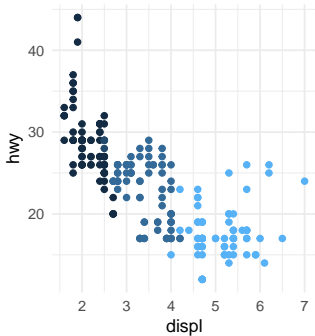
VI. More Plots

position = "jitter"

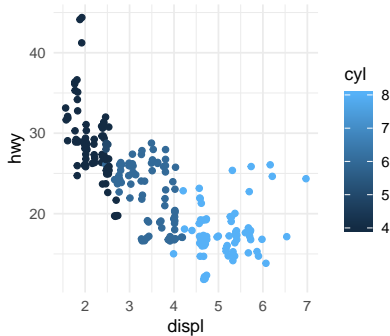
▶ 중첩된 관찰값에 노이즈를

```
a <- ggplot(mpg) + geom_point(aes(displ, hwy, color = cyl))  
b <- ggplot(mpg) + geom_point(aes(displ, hwy, color = cyl), position = "jitter")
```

a



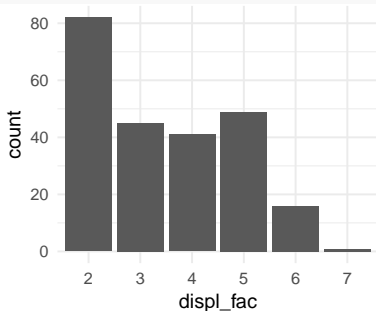
b



Barchart (count, density, distribution)

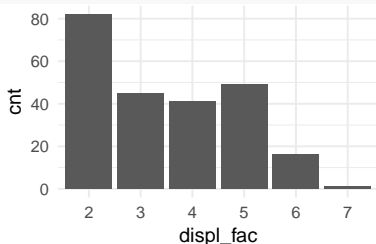
- ▶ x축에 Categorical 변수를 넣어서
- ▶ 각각의 값에 대해서 몇 개의 관찰값이 있는지를 보여줌.

```
mpg$displ_fac <-  
  as.factor(round(mpg$displ,0))  
a <- ggplot(mpg) +  
  geom_bar(aes(x = displ_fac))  
a
```



- ▶ y축에 해당하는 값이 계산되어 있다면 `stat = "identity"`를 `geom_bar()`에 넣어주어야 함.

```
library(dplyr)  
mpg_fac <- mpg %>%  
  group_by(displ_fac) %>%  
  summarise(cnt = length(displ_fac))  
b <- ggplot(mpg_fac) +  
  geom_bar(aes(x = displ_fac, y = cnt),  
           stat = "identity")  
b
```

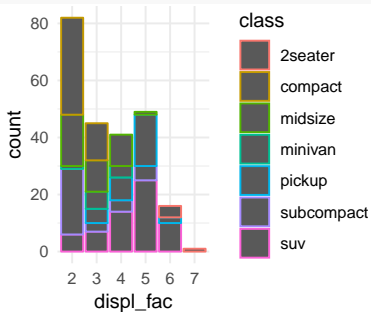


geom_bar()에 다른 Categorical 변수 추가

color and fill

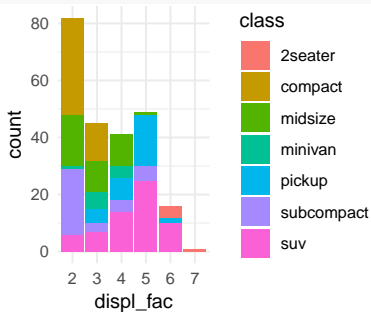
► color - 테두리만 됨

```
ggplot(mpg, aes(x = displ_fac)) +  
  geom_bar(aes(color = class))
```



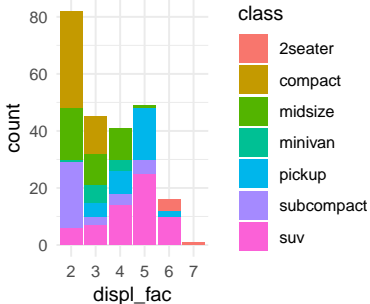
► fill - 채워짐

```
ggplot(mpg, aes(x = displ_fac)) +  
  geom_bar(aes(fill = class))
```

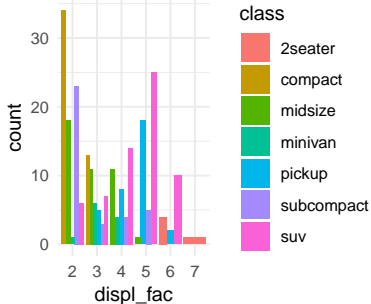


`position="dodge"`

```
ggplot(mpg, aes(x = displ_fac)) +  
  geom_bar(aes(fill = class))
```



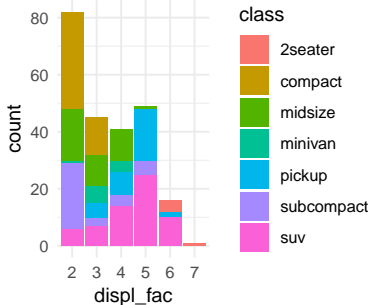
```
ggplot(mpg, aes(x = displ_fac)) +  
  geom_bar(aes(fill = class),  
           position = "dodge")
```



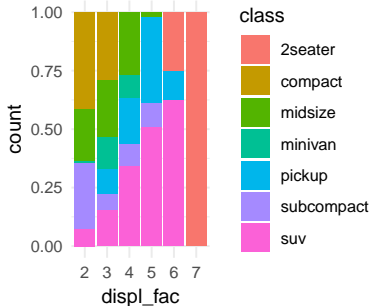
- ▶ discrete 변수를 잘 처리하는 법?
 - ▶ 쌓아서 배치 vs 비껴서 배치
 - ▶ 변수가 가지는 값의 갯수를 고려해야 함 (i.e. `unique()`)

Barchart (position = "fill")

```
ggplot(mpg, aes(x = displ_fac)) +  
  geom_bar(aes(fill = class))
```



```
ggplot(mpg, aes(x = displ_fac)) +  
  geom_bar(aes(fill = class),  
           position = "fill")
```

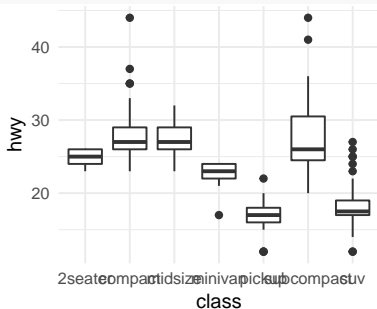


- ▶ class의 각각의 displ_fac 값에서의 분포?
- ▶ count vs proportion

Boxplot

- ▶ x는 이산 변수, y는 연속 변수
- ▶ x의 값에 따라 y의 분포를 보고 싶을 때

```
ggplot(mpg, aes(x = class, y = hwy)) +  
  geom_boxplot()
```

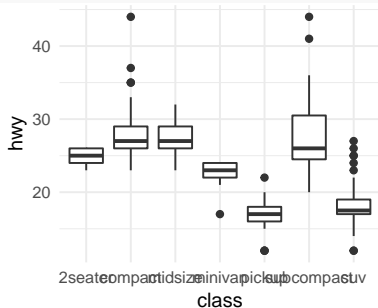


1. 박스의 가운데 직선은 중간값
2. 박스의 상단은 상위 25%, 하단은 하위 25%
3. 점으로 표현된 것은 이상치(outlier)로서 이상하게 높거나 낮은 값

Violin plot

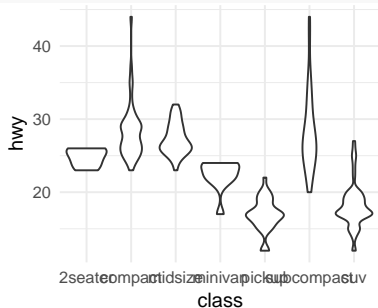
```
a <- ggplot(mpg, aes(x = class, y = hwy)) +  
  geom_boxplot()
```

a



```
b <- ggplot(mpg, aes(x = class, y = hwy)) +  
  geom_violin()
```

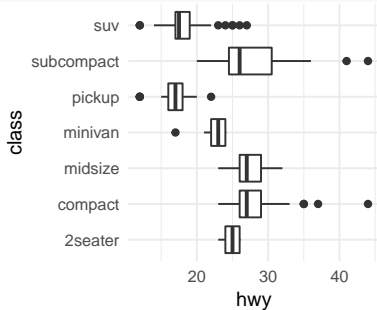
b



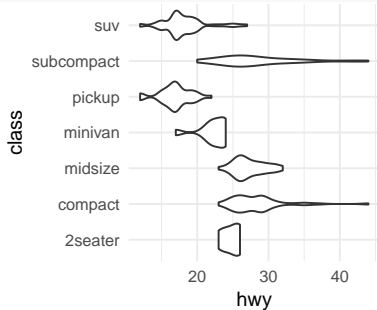
4. x변수의 값이 **character**라서 display가 복잡하네요.
5. 이럴때는 어떻게 할까요?

coord_flip()

a + coord_flip()

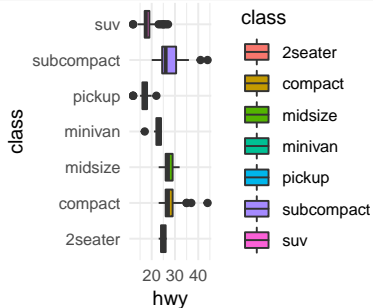


b + coord_flip()

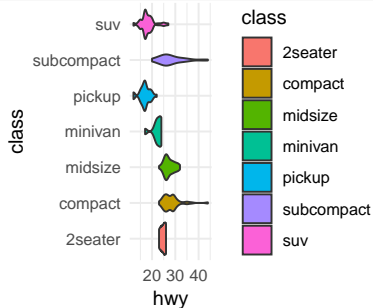


색깔을 넣어볼까요?

a + `coord_flip()` + `aes(fill=class)`



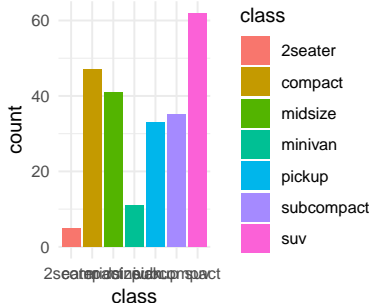
b + `coord_flip()` + `aes(fill=class)`



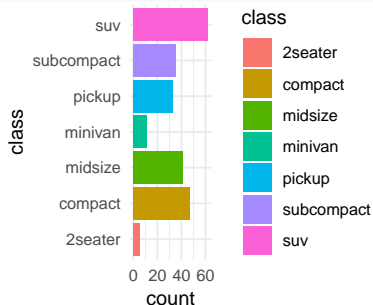
- ▶ 생각해보면 `aes()`도 `ggplot()`이나 `geom_XXX()`이후에도 추가가 가능해야 겠죠?

coord_flip() for Barplot

```
ggplot(mpg, aes(x = class, fill=class)) +  
  geom_bar()
```

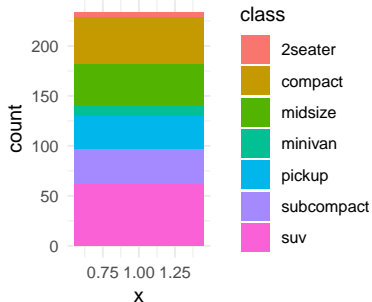


```
ggplot(mpg, aes(x = class, fill=class)) +  
  geom_bar() + coord_flip()
```

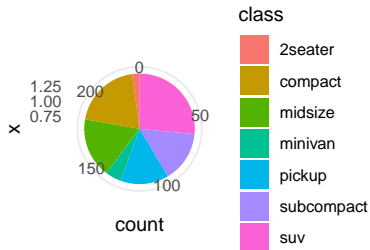


coord_polar() for Barplot

```
ggplot(mpg, aes(x = 1, fill = class)) +  
  geom_bar()
```



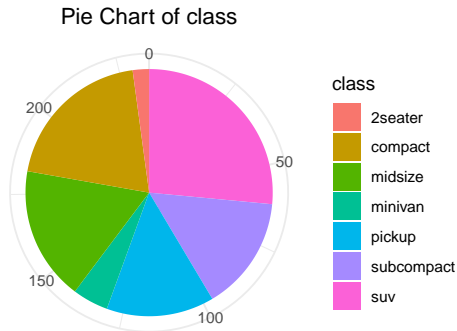
```
ggplot(mpg, aes(x = 1, fill = class)) +  
  geom_bar() + coord_polar(theta = "y")
```



- ▶ Bar chart vs Pie chart
- ▶ 둘의 차이는 ggplot2 개발의 motivation이 되었다고 합니다.

Pie Chart (from M24-ggplot2 Gallery)

```
ggplot(mpg, aes(x = "", fill = factor(class))) +  
  geom_bar(width = 1) +  
  theme(axis.line = element_blank(), plot.title = element_text(hjust=0.5)) +  
  labs(fill = "class", x = NULL, y = NULL,  
        title = "Pie Chart of class", caption = "Source: mpg") +  
  coord_polar(theta = "y", start=0)
```



Source: mpg

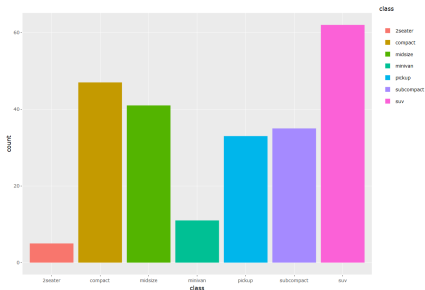
ggplot 기타 기능 (저장 & plotly)

- ▶ ggplot 객체를 png 파일로 저장

```
a <- ggplot(mpg) + geom_bar(aes(x=class)) + coord_flip()  
ggsave(filename = "out_file.png", plot = a, dpi = 300, width = 9, height = 7.5)
```

- ▶ plotly - html에서 각종 추가 기능 제공

```
library(plotly)  
a <- ggplot(mpg) + geom_bar(aes(x=class, fill=class))  
ggplotly(a) # better in html
```



VII. Summary

Summary

▶ 1 Variable

변수	이름	Name	geom_()
Continuous (연속 변수)	확률분포 히스토그램	density plot histogram	geom_density() geom_histogram()
Discrete (이산 변수)	막대차트 파이차트	bar plot pie chart	geom_bar() ("dodge", "jitter") geom_bar() + coord_polar()

▶ 2 Variables

x	y	이름	Name	geom_()
Discrete	Continuous	박스플랏	box plot	geom_boxplot()
Continuous	Continuous	산점도	scatter plot line chart	geom_point() "jitter" geom_line(), geom_smooth()

More References

1. **ggplot2** Cheatsheet
2. **M24-ggplot-Gallery** - 50개의 예제
3. **M6X References/books/ggplot2.pdf**
 - ▶ 무료 배포된 ggplot관련 영문 서적
 - ▶ 번역본도 출간되어 있음
4. google: 'ggplot gallery'
 - ▶ <https://www.r-graph-gallery.com/portfolio/ggplot2-package/>

‘Tantum videmus quantum scimus.’

‘아는 만큼 본다.’

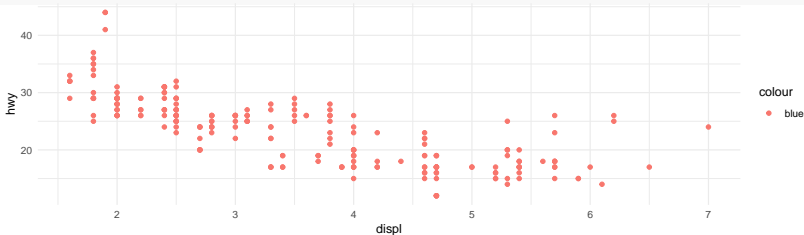
VIII. 연습 문제

1. `hwy`와 `cyl`을 이용해서 산점도(scatterplot)를 그려보세요. `cyl`과 같은 discrete variable을 이용해서 산점도를 그리면 어떤 일이 벌어지나요? 산점도가 어떤 그래프를 그려야 할까요?

```
ggplot(mpg, aes(x = cyl, y = hwy)) + geom_point()
ggplot(mpg, aes(x = cyl, y = hwy)) + geom_boxplot()
ggplot(mpg, aes(x = as.factor(cyl), y = hwy)) + geom_boxplot()
ggplot(mpg, aes(x = as.factor(cyl), y = hwy)) + geom_violin()
```

2. 아래 명령은 무엇이 잘못되었나요? 모든점을 파란색으로 그리려면 어떻게 해야하나요?

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, color = "blue"))
```



정답은 아래의 코드에 있습니다. 무엇이 다르나요?

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy), color = "blue")
```

이해되시나요? 즉, `aes()`에는 데이터마다 달라지는 특성, `geom_XXXXX`에는 데이터에 상관없이 공통적으로 적용되는 특성을 넣어줘야 합니다.

3. `mpg`의 어떤 변수가 categorical입니까? 어떤 변수가 continuous입니까? (Hint: `summary()`, `length(unique())`)
4. Continuous 변수로 `facet`을 하면 어떤 일이 벌어질까요?

5. 아래의 코드를 실행시켜 보세요. `facet_grid()`가 아니라 `facet_wrap()`을 사용한 것과 차이가 있나요?

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  facet_grid(drv ~ .)
```

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy)) +  
  facet_grid(. ~ cyl)
```

6. 아래 코드를 실행해보세요. 어떤 점을 새로 알게 되었나요?

```
ggplot(data = diamonds, mapping = aes(x = cut, fill = clarity)) +  
  geom_bar(alpha = 1/5)
```

```
ggplot(data = diamonds, mapping = aes(x = cut, colour = clarity)) +  
  geom_bar(fill = NA)
```

7. `labs()` 함수는 레이블을 만듭니다. ? `labs`를 콘솔에 쳐서 읽어보세요. 이를 이용해서 그래프의 제목, x축, y축에 글자를 넣어보세요.