Visualization

Seokjin Woo

시각화

개요

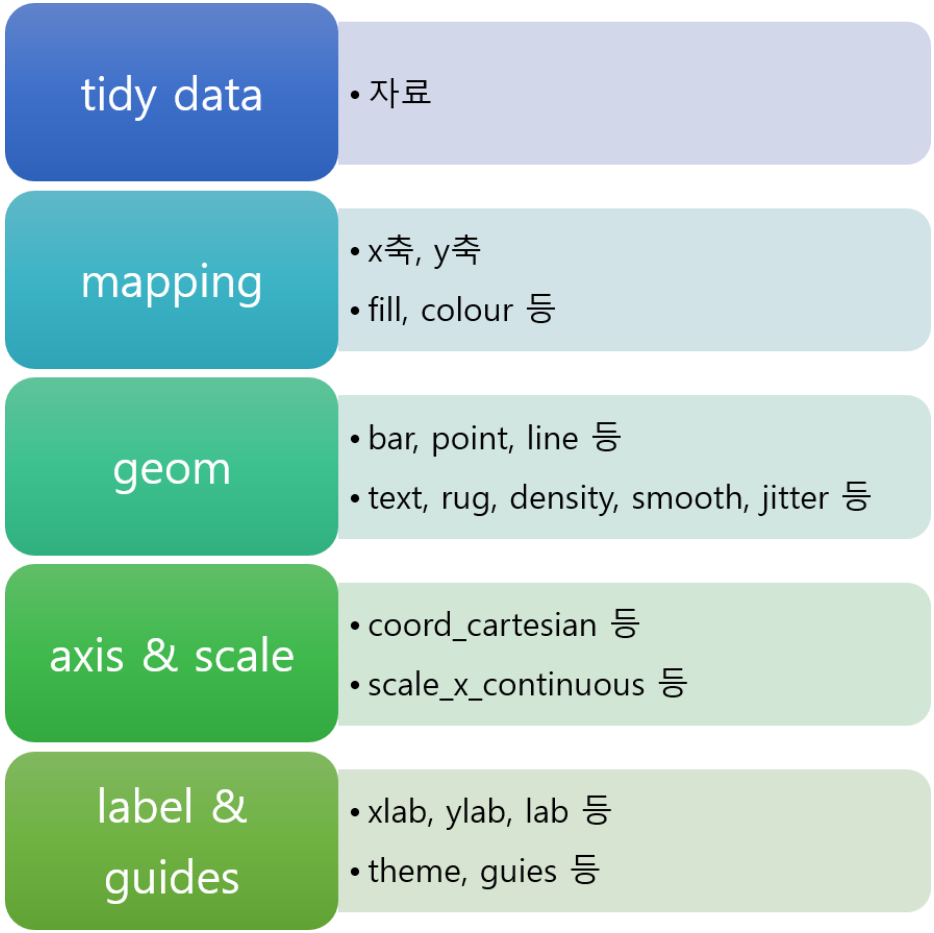
- R에 내장된 그래픽 기능을 사용하는 것도 좋지만
- ggplot2 패키지를 이용하는 것이 보통이다
- 따라서 중복 투자 없이 효율적으로 R을 사용하기 위해서는 ggplot2를 처음부터 사용하는 것이 좋다.

개요 2

- 기본적인 구조는 다음과 같다.

1. 먼저, 데이터가 있어야 한다.
2. 자료 중 어떤 변수를 사용할지를 시각화 시킬지 결정해야 한다
    - 이 과정을 mapping 이라고 한다
3. 어떤 모양을 통해 시각화를 할 것인지를 정해야 한다.
    - 이를 geom(etrics) 이라고 부른다.
4. 이렇게 그려진 그림에 축, 스케일, 색 팔레트, 범례 등을 설정해주면 된다

개요 3


tidy data
• 자료
mapping
• x축, y축
• fill, colour 등
geom
• bar, point, line 등
• text, rug, density, smooth, jitter 등
axis & scale
• coord_cartesian 등
• scale_x_continuous 등
label & guides
• xlab, ylab, lab 등
• theme, guies 등

tidy data

- ggplot으로 그림을 편하게 그리기 위해서는 R에서는 tidy data라고 부르는 형태의 자료가 필요하다.
- Stata에서는 자료의 형태를 wide-form 혹은 long-form 으로 부른다.
- wide-form은 가로로 뚱뚱한 자료 형태이다.

tidy data 2

region	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017
서울특별시	1.275	1.111	1.006	1.014	1.015	0.932	0.980	1.068	1.010	0.962	1.015	1.014	1.059	0.968	0.983	1.001	0.940	0.836
부산광역시	1.235	1.103	0.975	0.988	0.953	0.887	0.915	1.024	0.980	0.940	1.045	1.078	1.135	1.049	1.090	1.139	1.095	0.976
대구광역시	1.378	1.216	1.076	1.116	1.087	1.001	1.011	1.137	1.072	1.029	1.109	1.146	1.217	1.127	1.169	1.216	1.186	1.067
인천광역시	1.473	1.324	1.185	1.213	1.158	1.075	1.116	1.257	1.186	1.143	1.214	1.232	1.301	1.195	1.212	1.216	1.144	1.007
광주광역시	1.636	1.421	1.264	1.278	1.203	1.105	1.152	1.262	1.198	1.137	1.223	1.234	1.295	1.170	1.199	1.207	1.168	1.053
대전광역시	1.501	1.330	1.207	1.221	1.181	1.107	1.158	1.274	1.215	1.156	1.205	1.261	1.315	1.234	1.250	1.277	1.192	1.075
울산광역시	1.633	1.423	1.242	1.280	1.241	1.186	1.242	1.403	1.338	1.308	1.369	1.393	1.481	1.391	1.437	1.486	1.418	1.261
세종특별자치시													1.597	1.435	1.354	1.893	1.821	1.668
경기도	1.628	1.437	1.305	1.321	1.280	1.183	1.239	1.361	1.285	1.226	1.309	1.314	1.355	1.226	1.241	1.272	1.194	1.069
강원도	1.600	1.413	1.317	1.279	1.261	1.188	1.202	1.356	1.253	1.248	1.313	1.338	1.374	1.249	1.248	1.311	1.237	1.123
충청북도	1.583	1.426	1.294	1.270	1.272	1.195	1.233	1.398	1.319	1.317	1.402	1.428	1.485	1.365	1.363	1.414	1.358	1.235
충청남도	1.698	1.532	1.361	1.358	1.357	1.267	1.356	1.506	1.444	1.408	1.479	1.496	1.571	1.442	1.421	1.480	1.395	1.276
전라북도	1.595	1.426	1.275	1.274	1.239	1.184	1.213	1.380	1.305	1.279	1.374	1.405	1.440	1.320	1.329	1.352	1.251	1.151
전라남도	1.750	1.566	1.391	1.389	1.360	1.290	1.337	1.542	1.449	1.445	1.537	1.568	1.642	1.518	1.497	1.549	1.466	1.325
경상북도	1.578	1.402	1.232	1.253	1.203	1.173	1.208	1.369	1.313	1.274	1.377	1.434	1.489	1.379	1.408	1.464	1.396	1.256
경상남도	1.586	1.417	1.272	1.290	1.266	1.189	1.254	1.434	1.368	1.323	1.413	1.446	1.503	1.367	1.409	1.437	1.358	1.227
제주특별자치도	1.783	1.564	1.394	1.438	1.365	1.310	1.372	1.489	1.386	1.378	1.463	1.487	1.598	1.427	1.481	1.477	1.432	1.305

tidy data 3

- 아래 그림은 위 자료를 long-form, 이른바 tidy data로 전환한 것
- year 변수, region 변수, 옆에 합계출산율 fertility 의 값이 들어간다.

tidy data 4

year	region	fertility
2000	강원도	1.600
2001	강원도	1.413
2002	강원도	1.317
2003	강원도	1.279
2004	강원도	1.261
2005	강원도	1.188
2006	강원도	1.202
2007	강원도	1.356
2008	강원도	1.253
2009	강원도	1.248
2010	강원도	1.313
2011	강원도	1.338
2012	강원도	1.374
2013	강원도	1.249
2014	강원도	1.248
2015	강원도	1.311
2016	강원도	1.237
2017	강원도	1.123
2000	경기도	1.628
2001	경기도	1.437
2002	경기도	1.305
2003	경기도	1.321
2004	경기도	1.280

tidy data 만들기

- 필요한 라이브러리인 tidyverse를 장착하자

```
> rm(list = ls())
> library(tidyverse)
```

- 엑셀 자료인 fertility.xlsx 를 읽어 들이자

```
> fertility <-
readxl::read_xlsx("fertility.xlsx")
```

tidy data 만들기 2

- head()함수를 통해서 보면, fertility가 wide-form 자료임을 알 수 있다.

```
> head(fertility[,1:5], n = 3)
# A tibble: 3 x 5
  region     `2000` `2001` `2002` `2003`
  <chr>       <dbl>  <dbl>  <dbl>  <dbl>
1 서울특별시   1.27   1.11   1.01   1.01
2 부산광역시   1.24   1.10  0.975  0.988
3 대구광역시   1.38   1.22   1.08   1.12
```

tidy data 만들기 3

- ggplot으로 그림을 편하기 그리기 위해서는 tidy data로 전환을 해주어야 한다.

```
> fer.tidy <- fertility %>%
+   gather(year, fer, -region) %>%
+   arrange(region, year)
```

tidy data 만들기 4

- 그러면 아래와 같이 long-form, 즉 tidy data로 전환할 수 있다.

```
> head(fer.tidy, n = 5)
# A tibble: 5 x 3
  region year    fer
  <chr>  <chr> <dbl>
1 강원도 2000   1.6
2 강원도 2001   1.41
3 강원도 2002   1.32
4 강원도 2003   1.28
5 강원도 2004   1.26
```

GGPLOT2

gapminder

- tidy data를 가지고 그래프를 그려보자.
- 실습을 위해 gapminder 라이브러리를 장착하자.
- 그리고 gapminder 자료를 불러들이자.

```
> library(gapminder)
> data(gapminder)
```

str()

```
> str(gapminder, max.leve = 2, digits.d = 1)
Classes 'tbl_df', 'tbl' and 'data.frame':   1704 obs. of  6 variables:
 $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ year     : int  1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
 $ lifeExp  : num  29 30 32 34 36 ...
 $ pop      : int  8425333 9240934 10267083 11537966 13079460 14880372 12881816 13867957 16317921 22227415 ...
 $ gdpPercap: num  779 821 853 836 740 ...
```

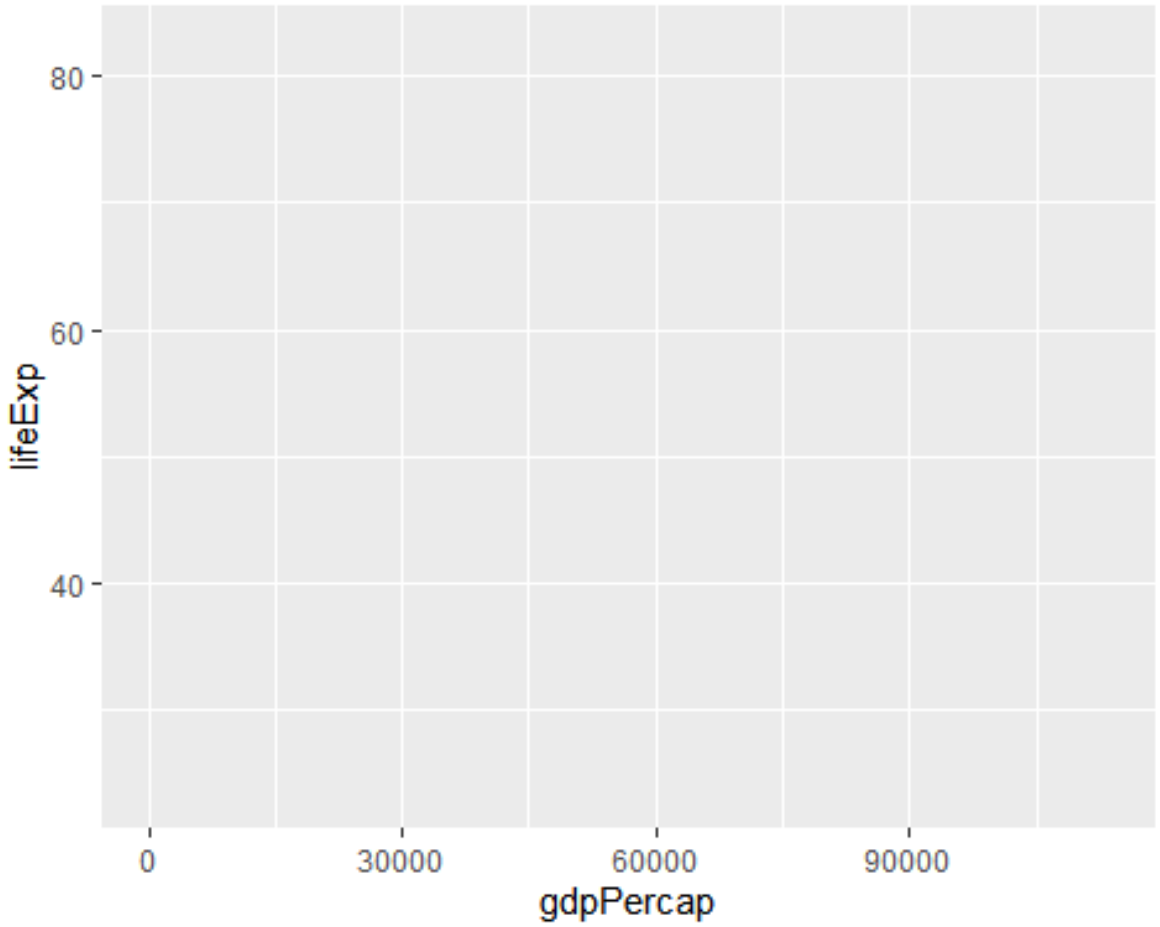
str() 2

- gapminder에는 6개의 변수에 1704개의 관측치가 포함되어 있다.
  - 국가명 country
  - 대륙명 continent
  - 연도 year
  - 기대수명 lifeExp
  - 인구 pop
  - 1인당 소득 gdpPercap

1인당 GDP와 수명간의 관계, aesthetics

- 시작은 데이터로부터 시작한다.
- ggplot()으로 시작한다.
- 변수들을 mapping 해준다.

```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp))
```


80
60
40
lifeExp
0
30000
60000
90000
gdpPercap

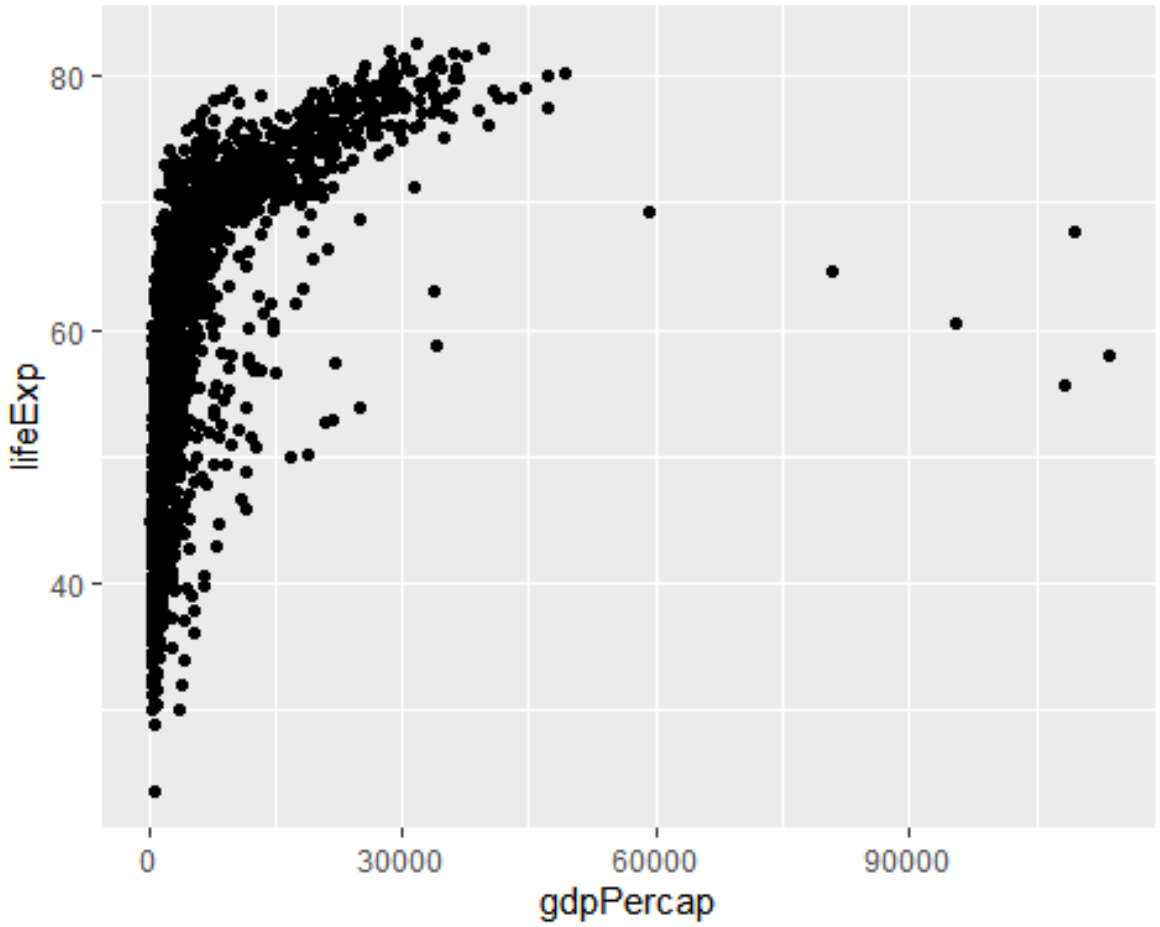
1인당 GDP와 수명간의 관계, aesthetics

- x축과 y축의 변수만을 보여주고 있다. aes() 함수는 어떤 변수를 그래프에 표시할 것인지를 지정해주는 역할을 한다.
- aes() 함수의 인자로서 x, y 외에도 색깔 colour, 모양 shape, 크기 size, 선의 타입 line type 등을 지정해줄 수 있다.

1인당 GDP와 수명간의 관계, geom

- 여기에 두 번째 단계로서 geom 을 이용하여 어떤 그림을 그릴 것인지를 지정해줄 수 있다.
- geom_point()의 경우, 점을 찍을 수 있다.
- “+”로 연결해주면 된다.

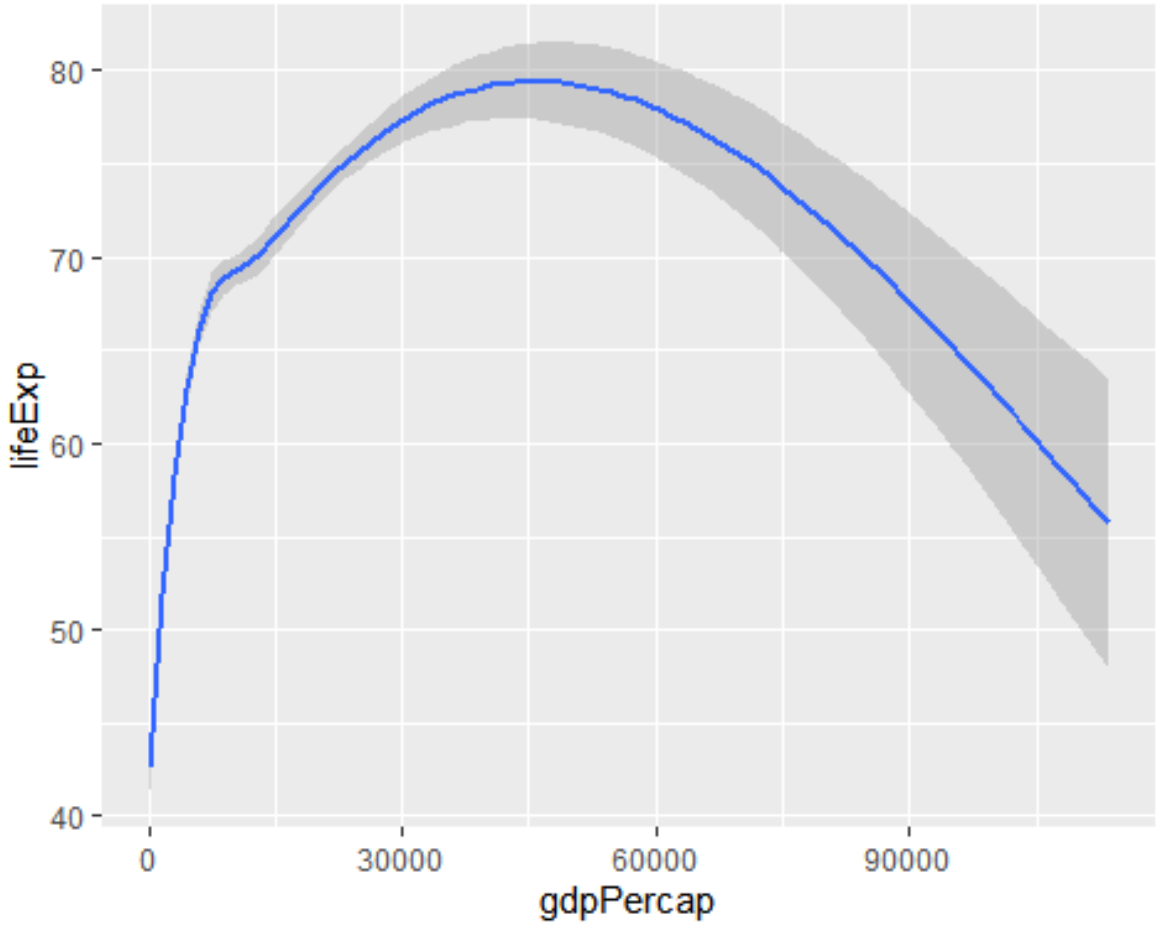
```
> gapminder %>%
+   ggplot(, mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point()
```


80
60
40
lifeExp
0
30000
60000
90000
gdpPercap

1인당 GDP와 수명간의 관계

- geom_smooth()를 이용하면 부드러운 회귀식을 그릴 수 있다.

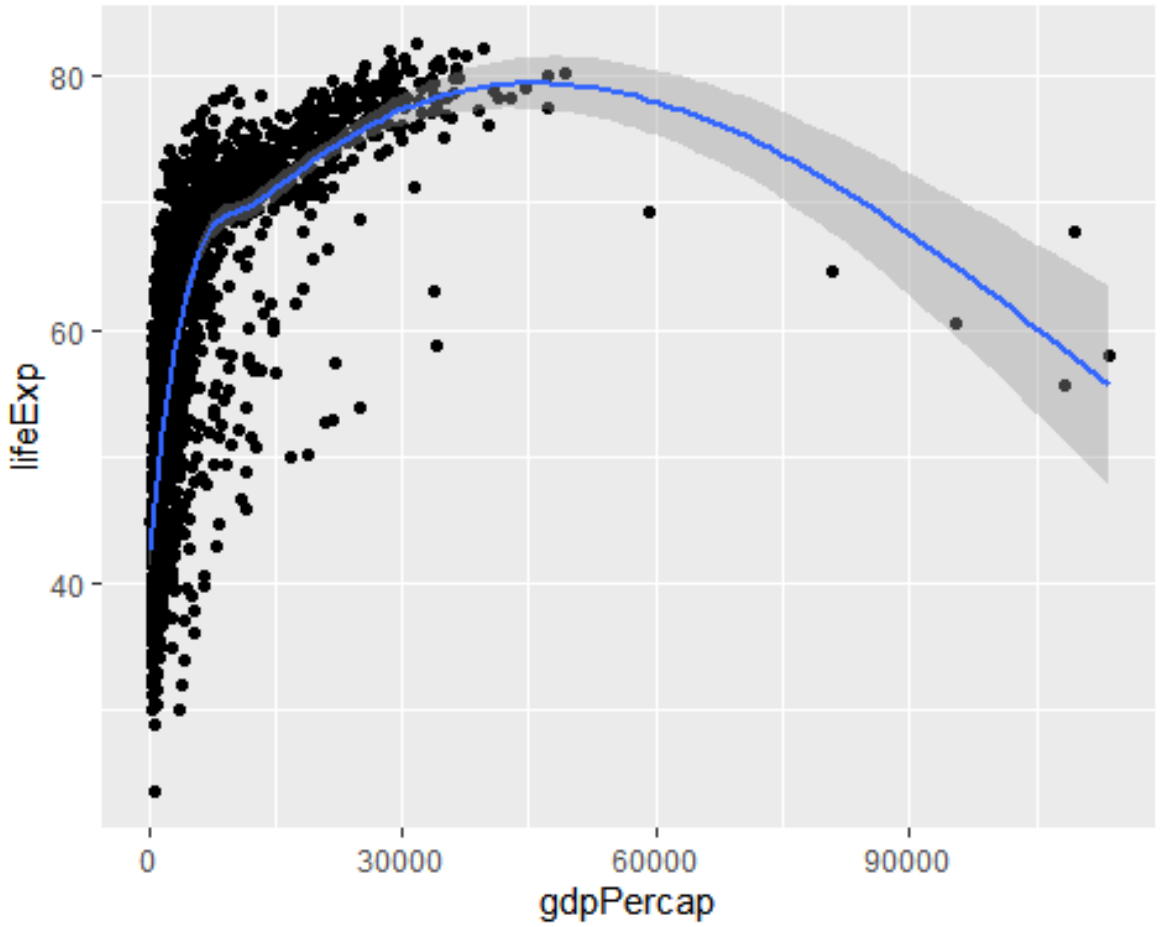
```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_smooth()
```


80
70
60
50
40
lifeExp
0
30000
60000
90000
gdpPercap

1인당 GDP와 수명간의 관계 2

- 위 두 그림을 합쳐서 그릴 수 있다.
- geom_point()와 geom_smooth()를 각각의 레이어(layer)로 보면 이해하기가 쉽다.
  - 산포도 레이어를 그리고
  - 그 위에 회귀선 레이어를 겹쳐 그리면 되기 때문이다.

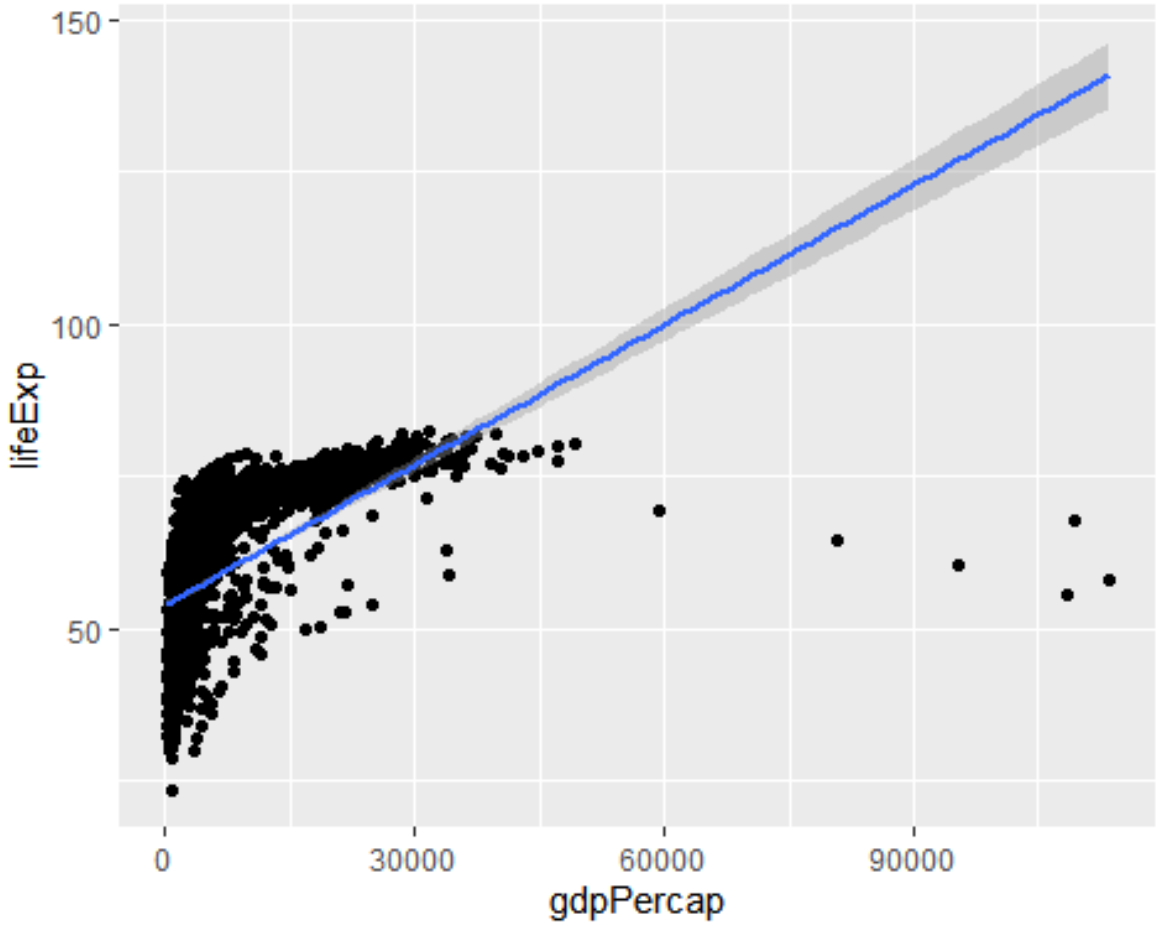
```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point() +
+   geom_smooth()
```


80
60
40
lifeExp
0
30000
60000
90000
gdpPercap

1인당 GDP와 수명간의 관계 3

- geom_smooth()함수에 몇 개의 파라미터를 조정해주면 다양한 회귀선을 그릴 수 있다.
- method 파라미터를 선형 lm 으로 지정해주면 선형 회귀식을 적합시킬 수 있다.

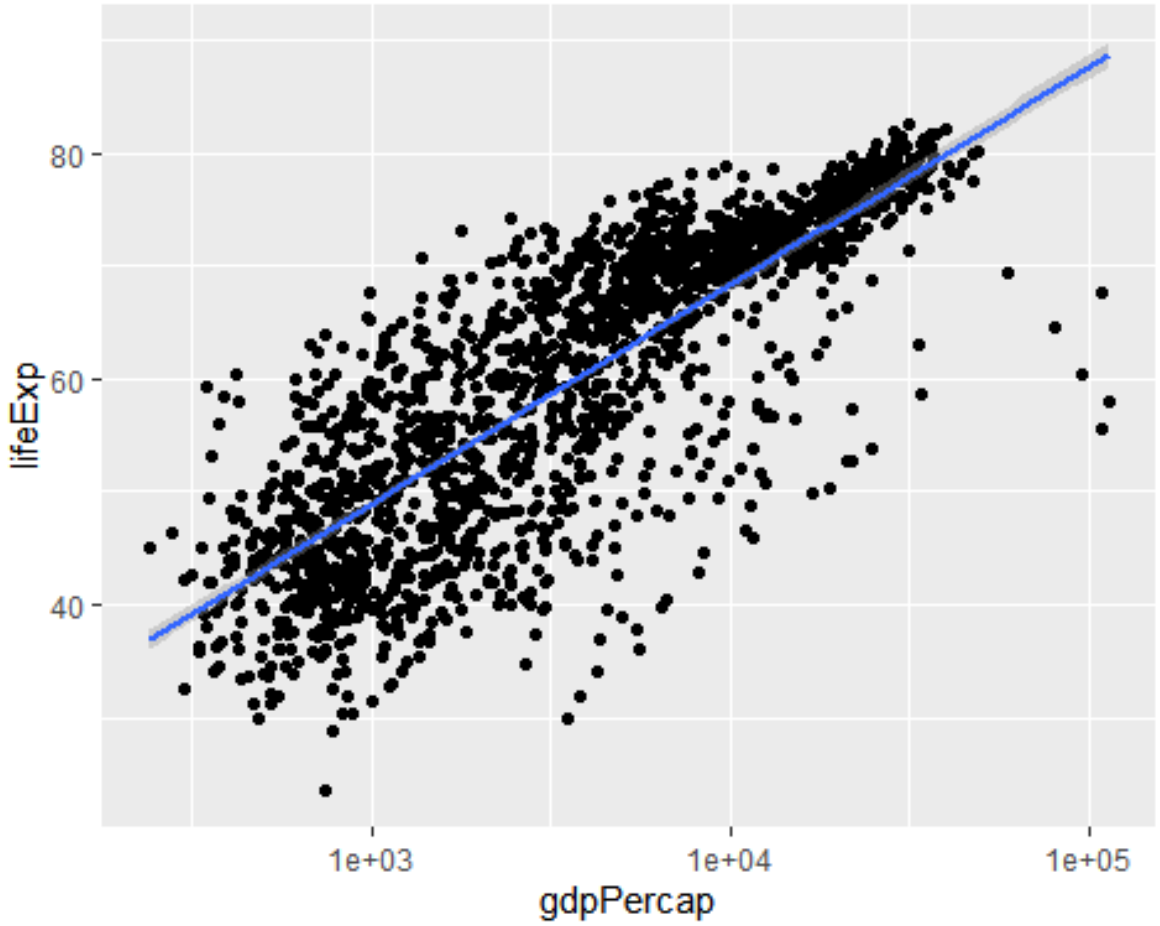
```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point() +
+   geom_smooth(method = "lm")
```


150
100
50
lifeExp
0
30000
60000
90000
gdpPercap

log-scale

- x축의 경우 작은 값에 많이 모여 있고, 큰 값은 듬성 듬성 있는 것을 눈으로 확인할 수 있다.
- 이런 경우 축의 스케일을 로그-스케일로 바꾸면 좀 더 보기 쉽고 이해하기 쉬운 그림이 된다.

```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point() +
+   geom_smooth(method = "lm") +
+   scale_x_log10()
```


80
60
40
lifeExp
1e+03
1e+04
1e+05
gdpPercap

단위 조정

- 여기에 scale 패키지를 이용하여 축의 스케일을 정해줄 수 있다.

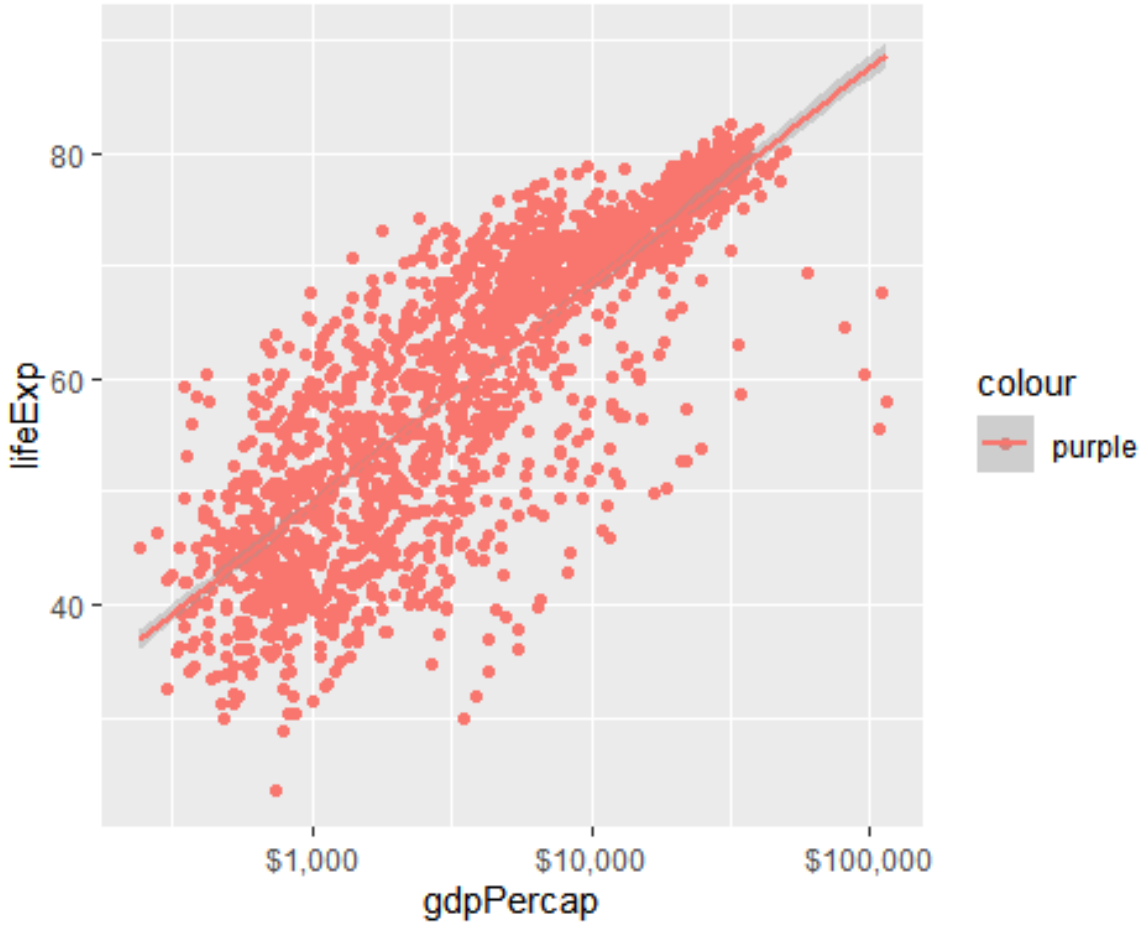
```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point() +
+   geom_smooth(method = "lm") +
+   scale_x_log10(labels = scales::dollar)
```


80
60
40
lifeExp
$1,000
$10,000
$100,000
gdpPercap

aesthetics

- aesthetics에 다양한 정보를 추가적으로 넣어줄 수 있다.
  - 예컨대, 점의 색을 다른 색으로 칠할 수 있다.

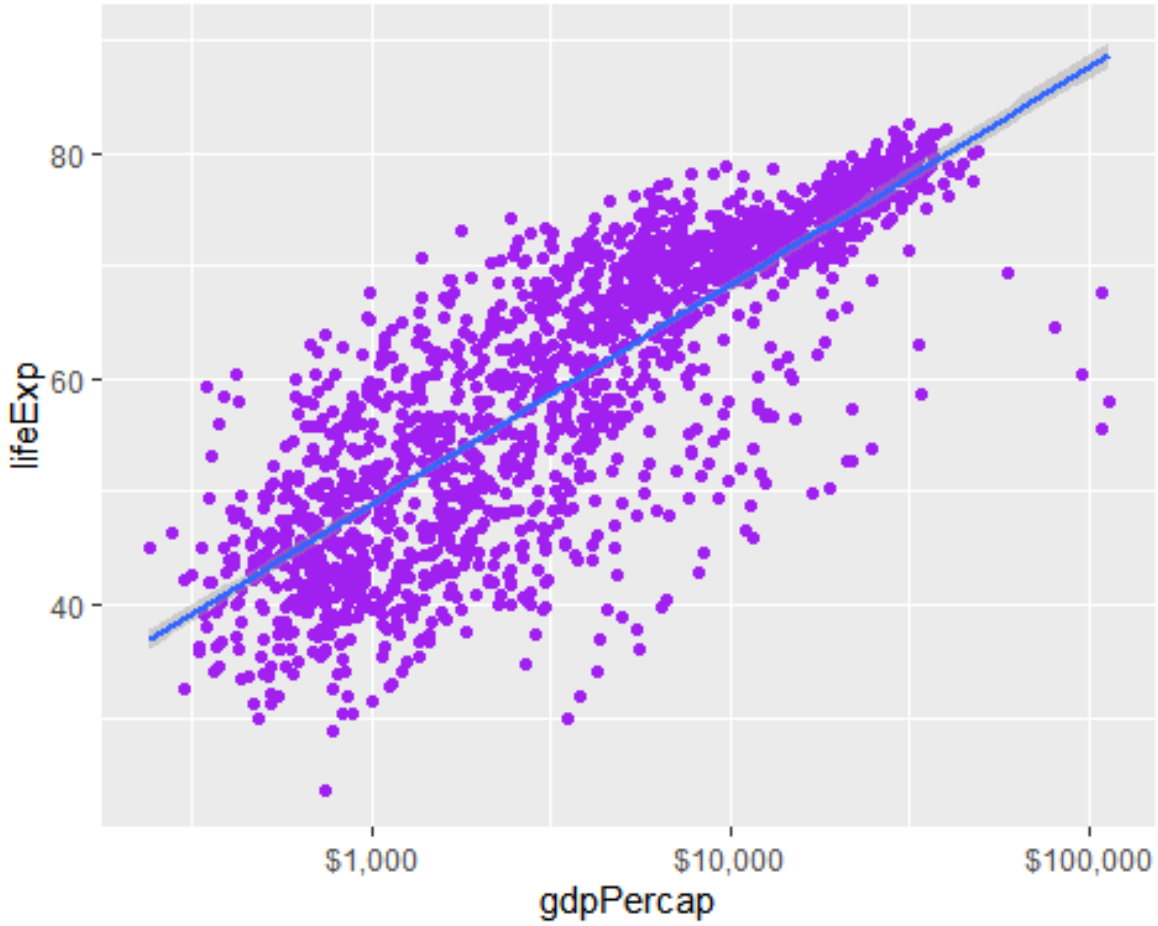
```
> ggplot(data = gapminder, mapping = aes(x =
gdpPercap, y = lifeExp, color = "purple")) +
+   geom_point() +
+   geom_smooth(method = "lm") +
+   scale_x_log10(labels = scales::dollar)
```


80
60
40
lifeExp
$1,000
$10,000
$100,000
gdpPercap
colour
purple

aesthetics 2

- 이번에는 aes() 대신에 geom_point()안에 color = “purple”을 넣어보자.
- 이러한 작업을 색을 셋팅(setting)한다고 하고, 보락색으로 점이 칠해진다.

```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point(color = "purple") +
+   geom_smooth(method = "lm") +
+   scale_x_log10(labels = scales::dollar)
```


80
60
40
lifeExp
$1,000
$10,000
$100,000
gdpPercap

투명도 alpha

- geom_point()안에 alpha 라는 파라미터를 넣을 수 있다.
- alapha는 0에서 1까지의 값을 갖고, 투명도를 나타낸다.
- 위 그림에서 alpha = 0.2 의 값을 갖도록 셋팅했다.

```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point(color = "purple", alpha = 0.2) +
+   geom_smooth(method = "lm") +
+   scale_x_log10(labels = scales::dollar)
```


80
60
40
lifeExp
$1,000
$10,000
$100,000
gdpPercap

투명도 alpha 2

- 이번에는 대륙 continent 별로 서로 다른 색깔을 가지도록 셋팅해보자.
- geom_point()안에 다시 aes()를 정해주었다. * 그리고 color = continent라고 정해주었다.

```
> gapminder %>%
+   ggplot(mapping = aes(x = gdpPercap, y =
lifeExp)) +
+   geom_point(mapping = aes(color = continent),
alpha   = 0.2) +
+   geom_smooth(method = "lm") +
+   scale_x_log10(labels = scales::dollar)
```


80
60
40
lifeExp
$1,000
$10,000
$100,000
gdpPercap
continent
Africa
Americas
Asia
Europe
Oceania

국가별로 그리는 방법은?

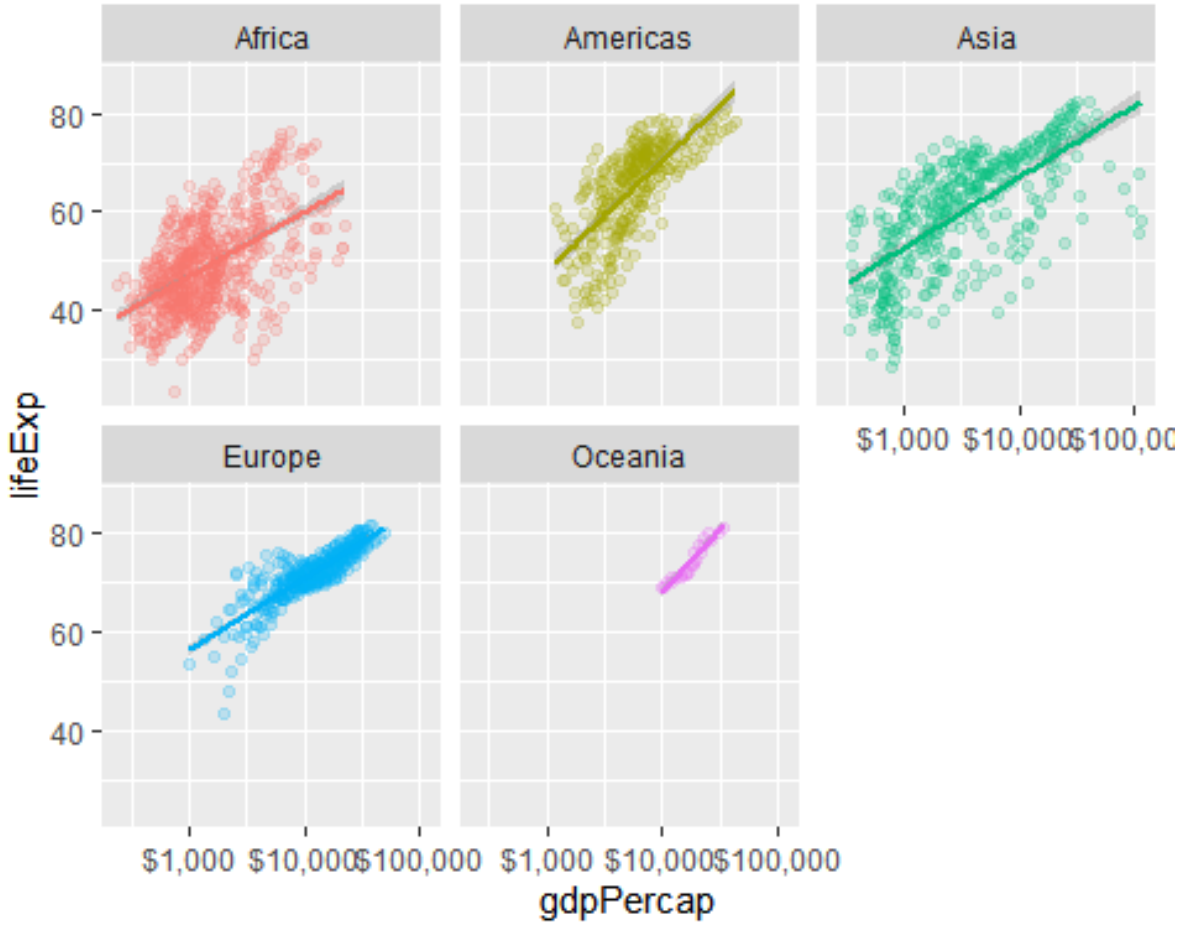
- 다음과 같이 aes에 color를 삽입

```
> gapminder %>%
+   ggplot( mapping = aes(x = gdpPercap, y =
lifeExp, color = continent)) +
+   geom_point(alpha    = 0.2) +
+   geom_smooth(method = "lm") +
+   scale_x_log10(labels = scales::dollar)
```


80
60
40
lifeExp
$1,000
$10,000
$100,000
gdpPercap
continent
Africa
Americas
Asia
Europe
Oceania

- 혹은 face_wrap()을 활용하면 된다.

```
> gapminder %>%
+   ggplot( mapping = aes(x = gdpPercap, y =
lifeExp, color = continent)) +
+   geom_point(alpha    = 0.2) +
+   geom_smooth(method = "lm") +
+   scale_x_log10(labels = scales::dollar) +
+   facet_wrap(~continent) +
+   theme(legend.position = "none")
```


Africa
Americas
Asia
Europe
Oceania
lifeExp
80
60
40
$1,000
$10,000
$100,000
gdpPercap

group

- 새로운 자료를 하나 로딩하자.
- 연도별, 지역별 합계출산율, 경제활동참가율 자료임

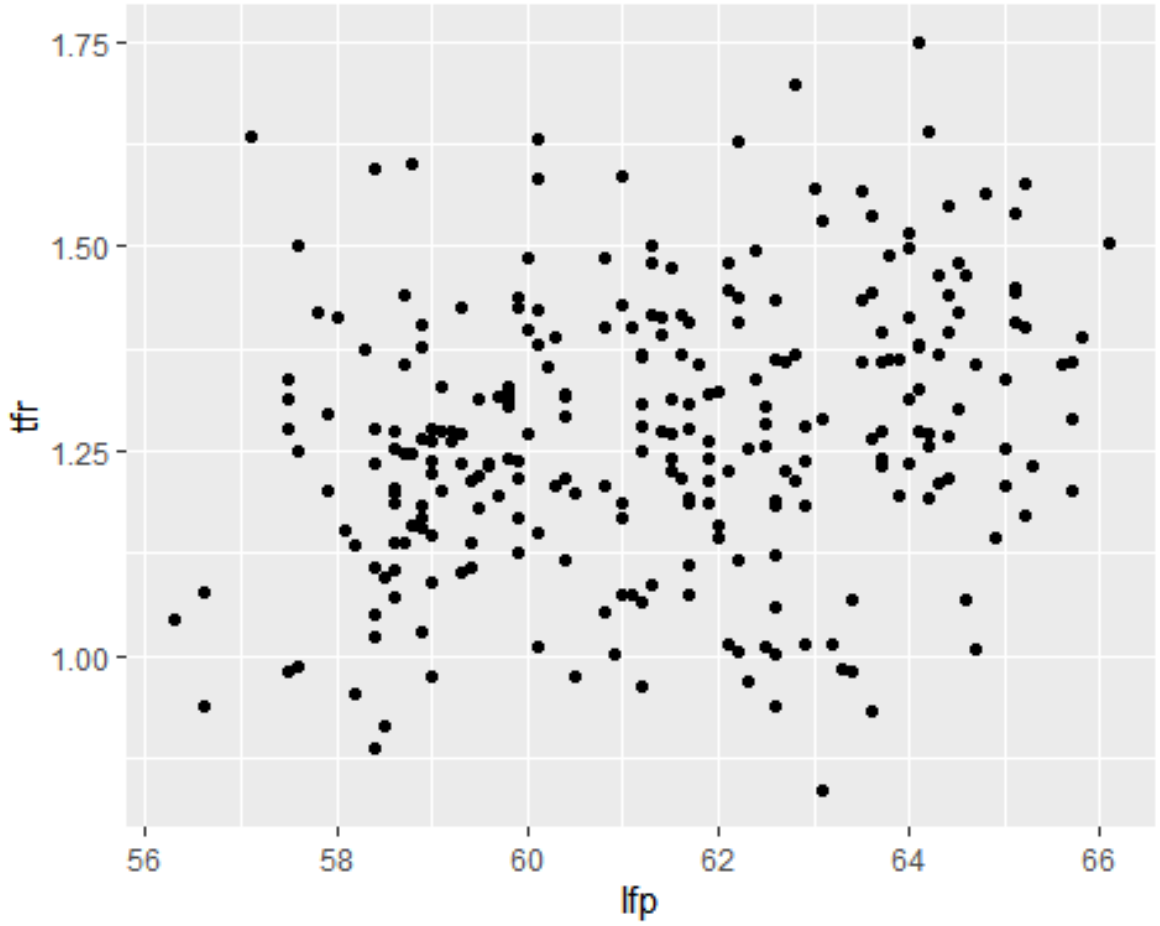
```
> tfr <- readxl::read_excel("tfr_lfp.xlsx")
```

- 처음 3개의 관측치는 다음과 같다.

```
> head(tfr, n = 3)
# A tibble: 3 x 4
   year region   lfp   tfr
  <dbl> <chr>  <dbl> <dbl>
1  2000 강원도  58.8  1.6
2  2001 강원도  58    1.41
3  2002 강원도  59.7  1.32
```

산포도

```
> tfr %>%
+   ggplot(aes(x = lfp, y = tfr)) +
+   geom_point()
```


1.75
1.50
1.25
1.00
tfr
56
58
60
62
64
66
lfp

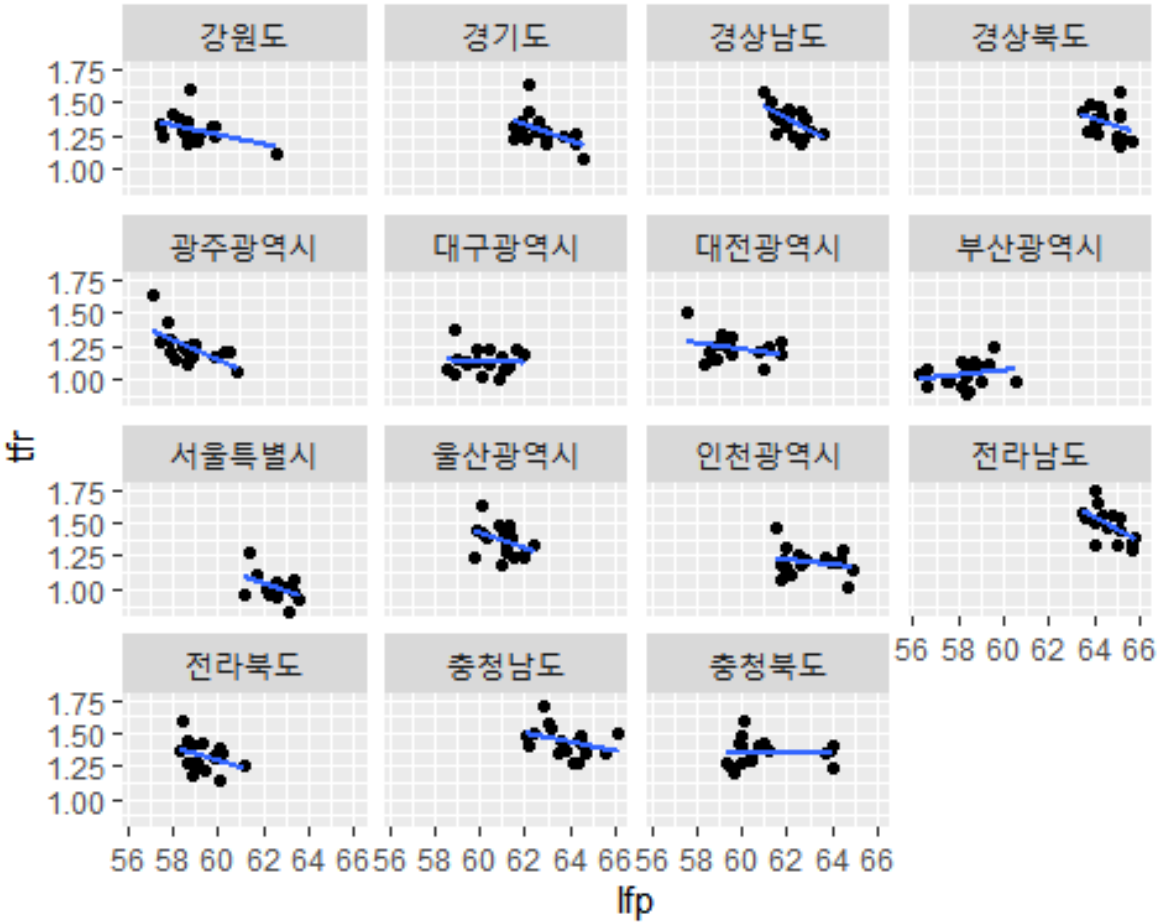
산포도 2(지역별)

```
> tfr %>%
+   ggplot(aes(x = lfp, y = tfr, color = region))
+
+   geom_point() +
+   theme(legend.position = "none")
```


1.75
1.50
1.25
1.00
tfr
56
58
60
62
64
66
lfp

산포도 3(지역별)

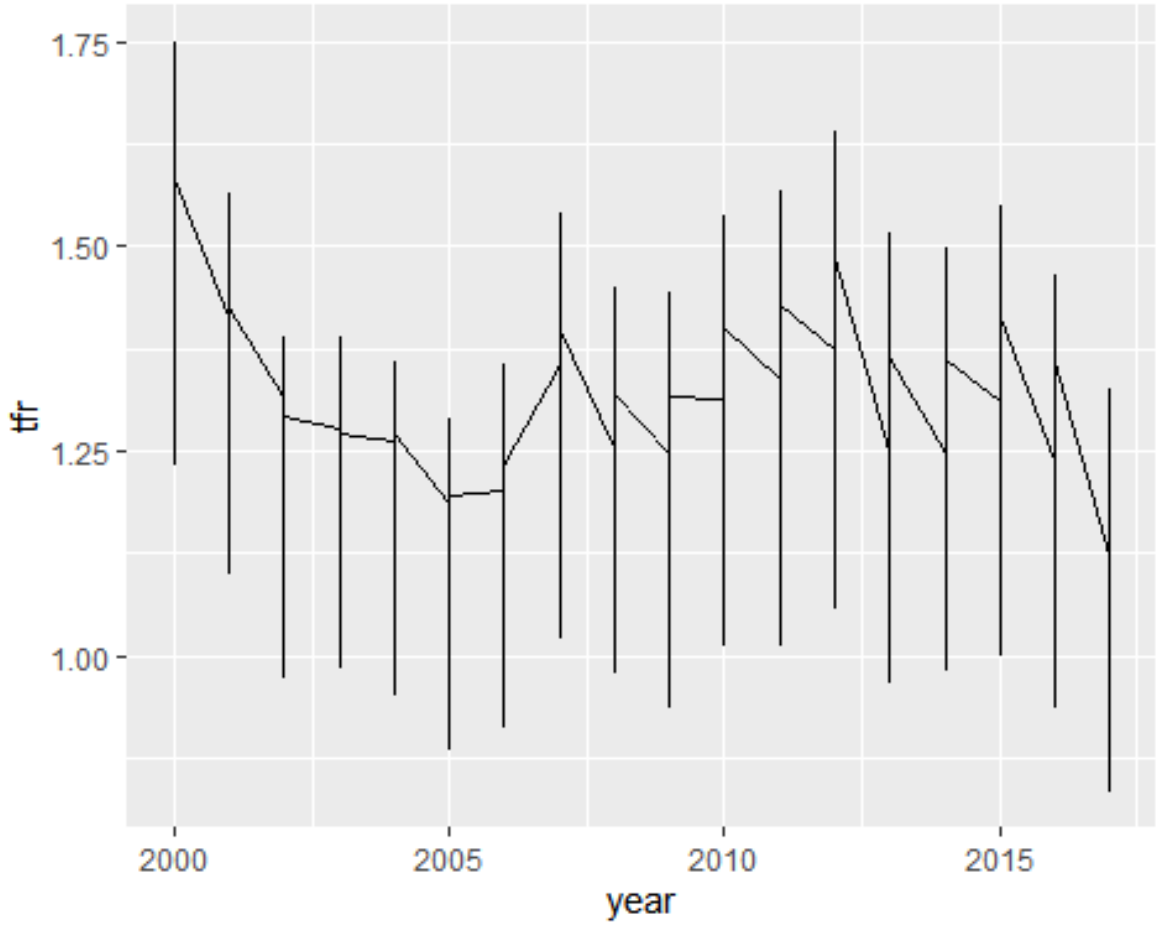
```
> tfr %>%
+   ggplot(aes(x = lfp, y = tfr)) +
+   geom_point() +
+   geom_smooth(method = "lm",se = FALSE) +
+   facet_wrap(~region)
```


강원도
경기도
경상남도
경상북도
광주광역시
대구광역시
대전광역시
부산광역시
서울특별시
울산광역시
인천광역시
전라남도
전라북도
충청남도
충청북도
1.75
1.50
1.25
1.00
56 58 60 62 64 66
tfr
lfp

추세선

- 우리가 x축에는 연도, y축에는 합계출산율을 나타내려고 했는데,
- 결과물은 우리가 상상했던 그런 결과물이 아니다

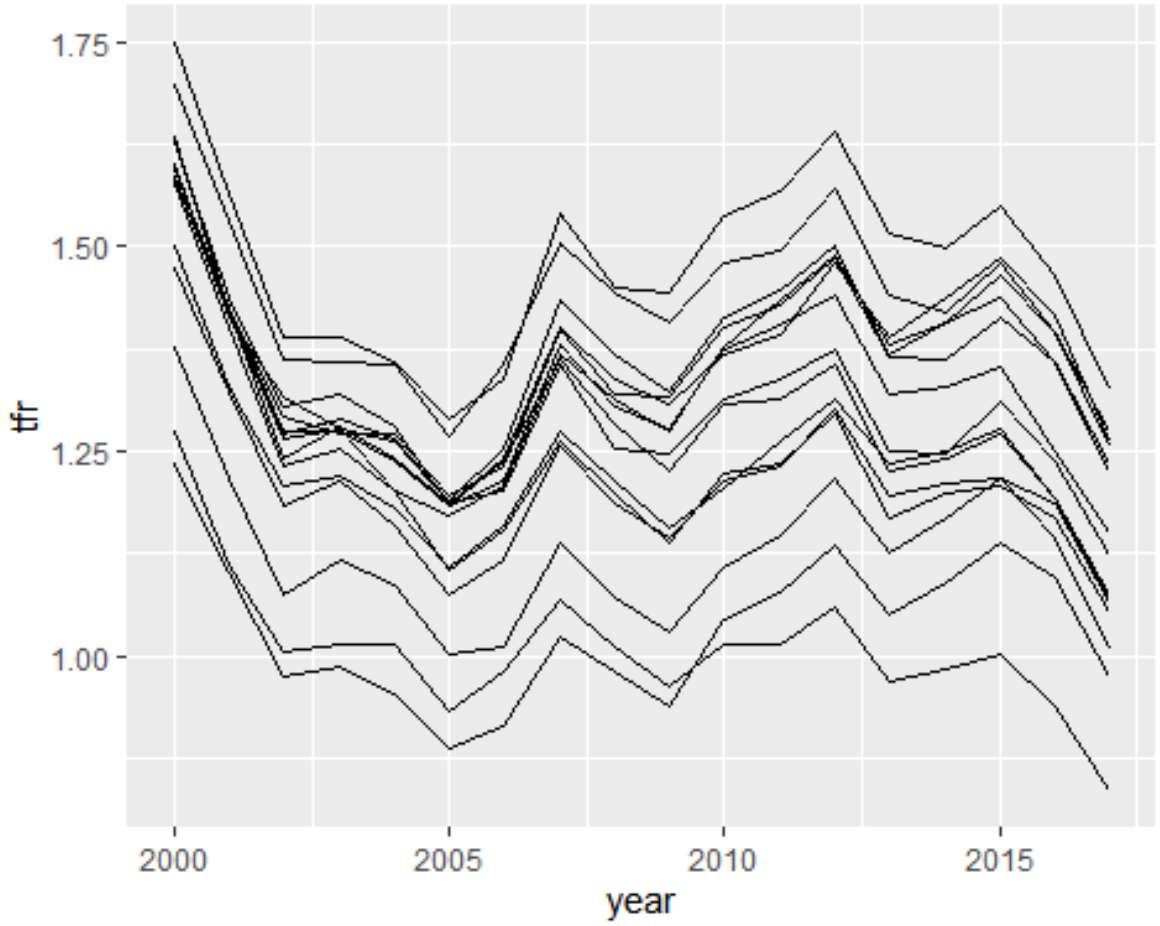
```
> tfr %>%
+   ggplot(mapping = aes(x = year, y = tfr))  +
+   geom_line()
```


1.75
1.50
1.25
1.00
tfr
2000
2005
2010
2015
year

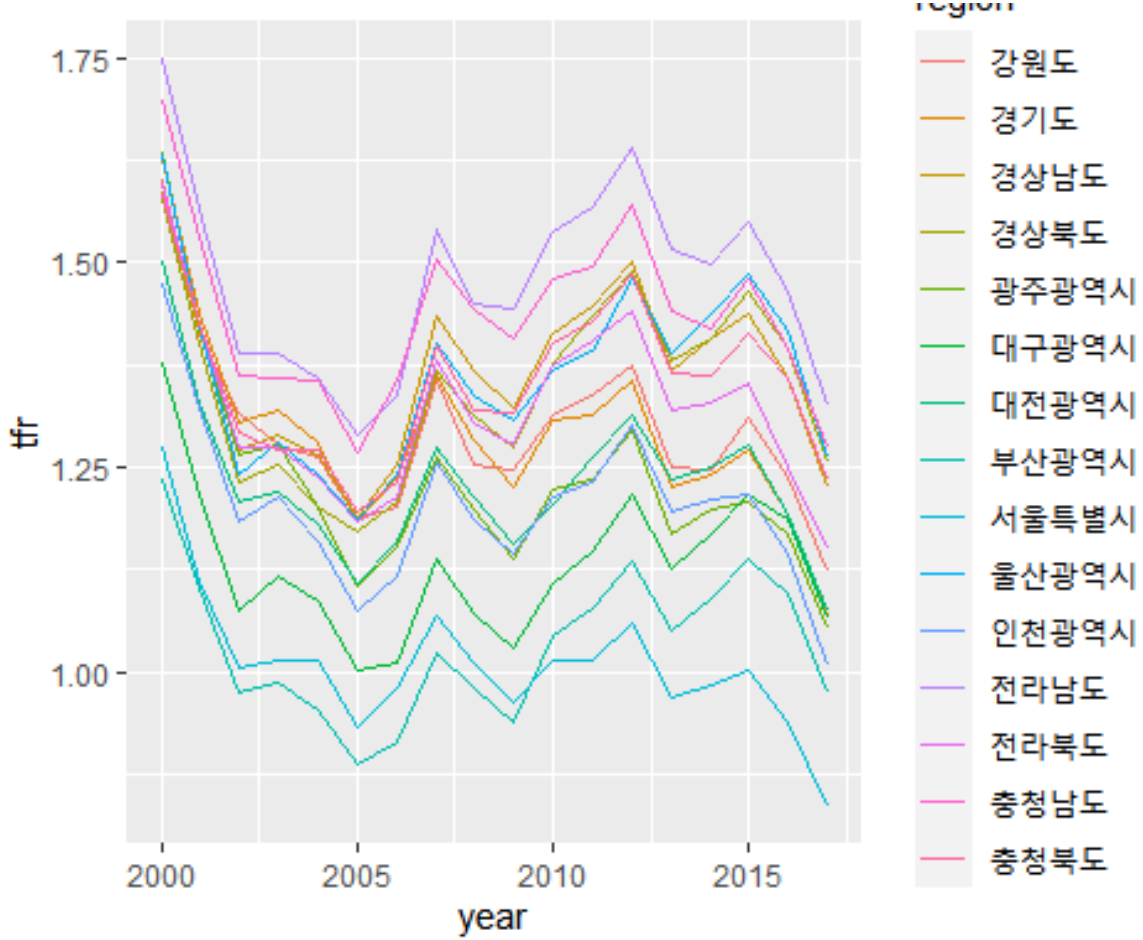
추세선 2

- 한 해에 많은 지역이 대응되고 있기 때문

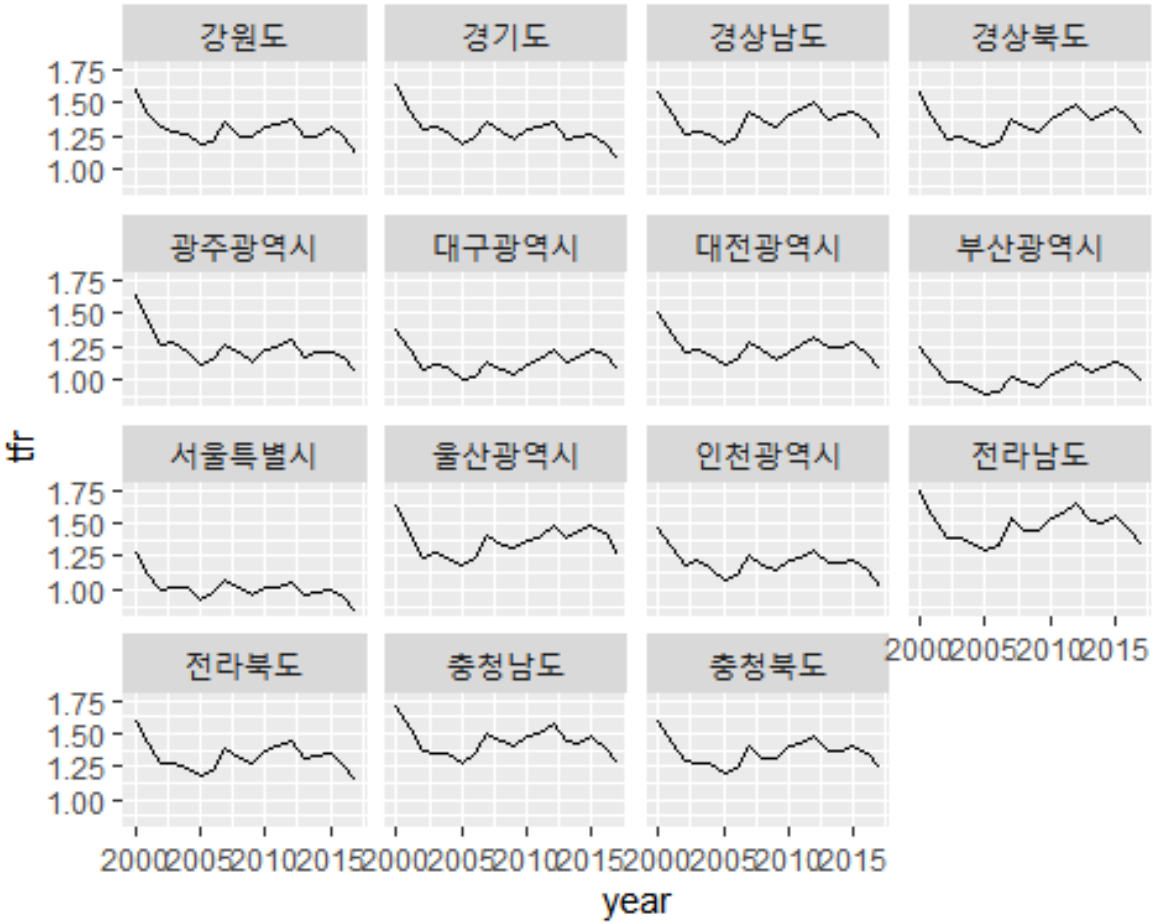
```
> tfr %>%
+   ggplot(mapping = aes(x = year, y = tfr, group
= region))  +
+   geom_line()
```


1.75
1.50
1.25
1.00
tfr
2000
2005
2010
2015
year

```
> tfr %>%
+   ggplot(mapping = aes(x = year, y = tfr, group
= region, color = region))  +
+   geom_line()
```


1.75
1.50
1.25
1.00
tfr
2000
2005
2010
2015
year
region
강원도
경기도
경상남도
경상북도
광주광역시
대구광역시
대전광역시
부산광역시
서울특별시
울산광역시
인천광역시
전라남도
전라북도
충청남도
충청북도

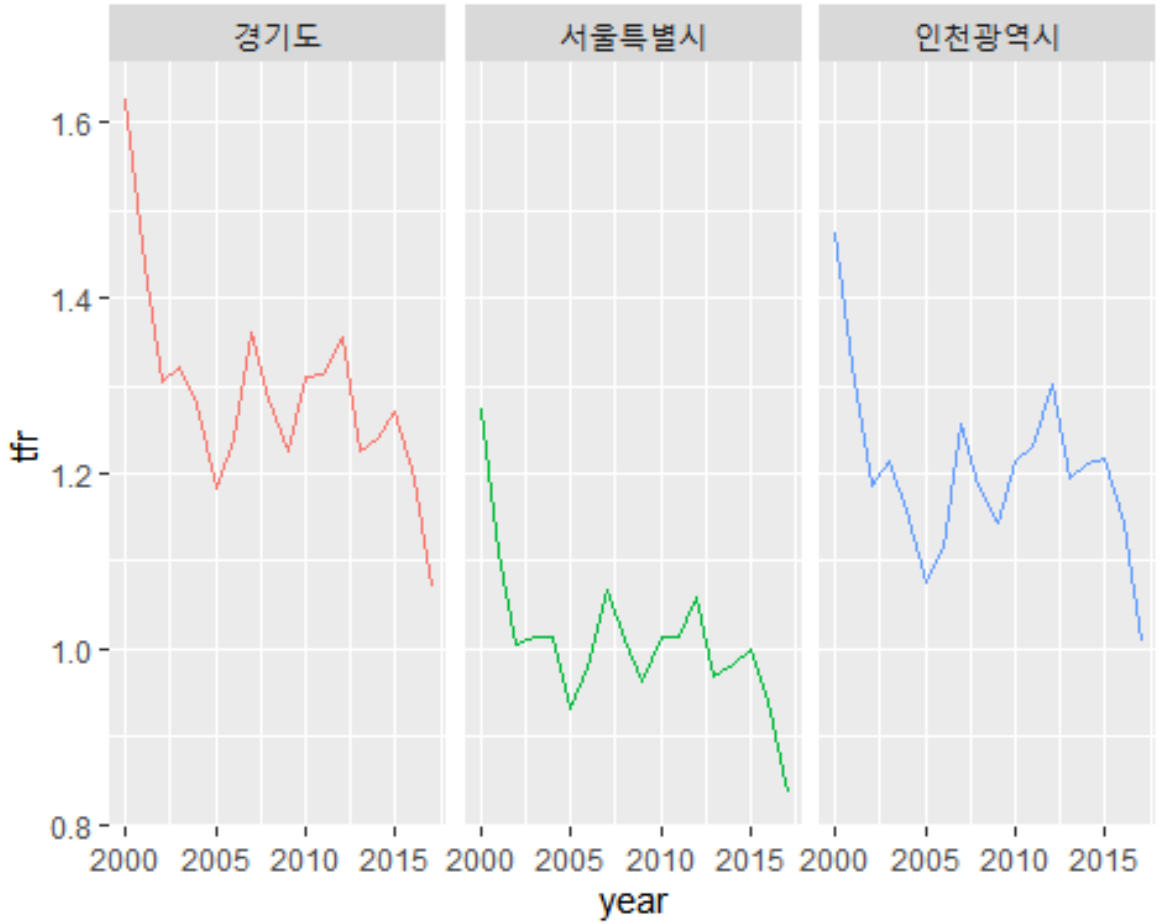
```
> tfr %>%
+   ggplot(mapping = aes(x = year, y = tfr))  +
+   geom_line()  +
+     facet_wrap(~region)
```


강원도
경기도
경상남도
경상북도
광주광역시
대구광역시
대전광역시
부산광역시
서울특별시
울산광역시
인천광역시
전라남도
전라북도
충청남도
충청북도
1.75
1.50
1.25
1.00
2000
2005
2010
2015
year

추세선 3

- 서울, 경기, 인천 만 뽑아서 그리는 경우

```
> seoulM = c("서울특별시", "경기도", "인천광역시")
>
> tfr %>%
+   filter(region %in% seoulM) %>%
+   ggplot(mapping = aes(x = year, y = tfr))  +
+   geom_line(aes(color = region))  +
+   facet_wrap(~region) +
+   theme(legend.position = "none")
```


경기도
서울특별시
인천광역시
1.6
1.4
1.2
1.0
0.8
tfr
2000
2005
2010
2015
year

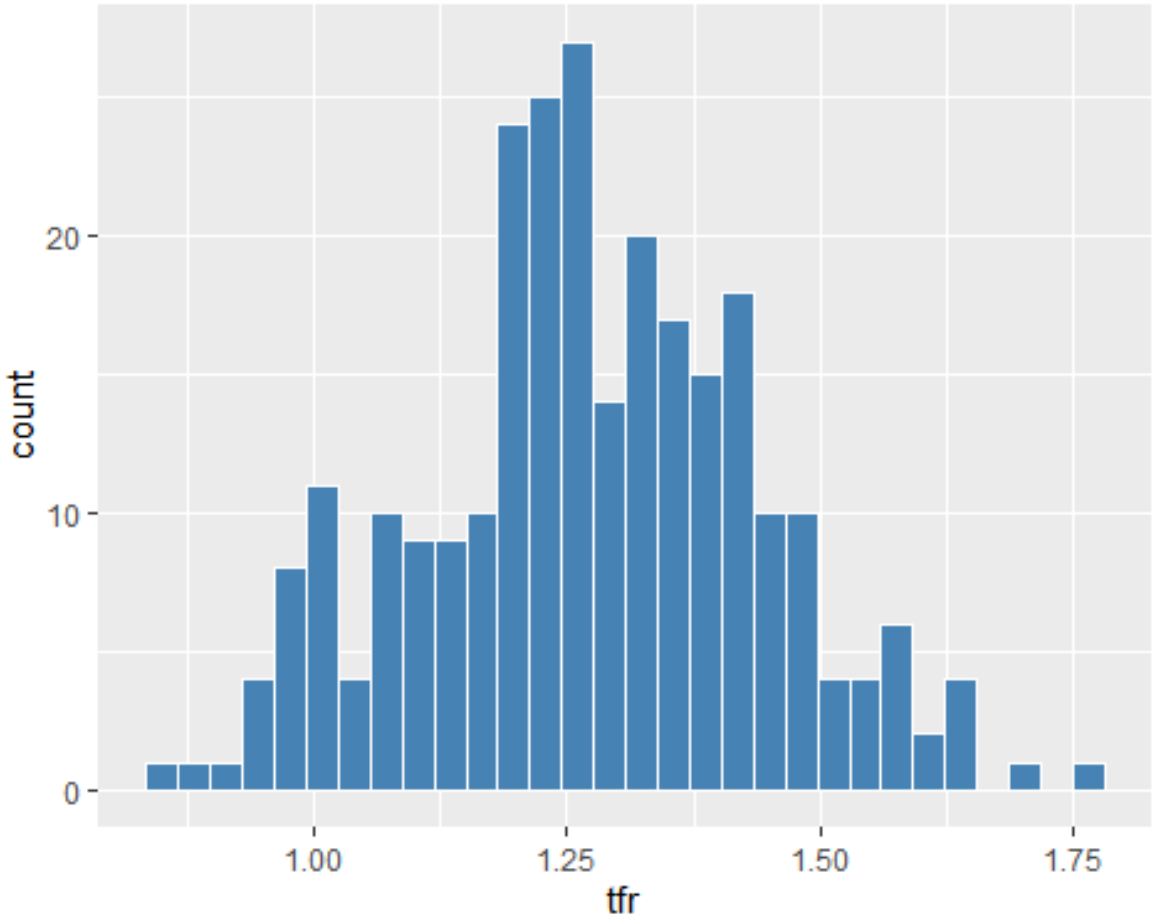
히스토그램

히스토그램

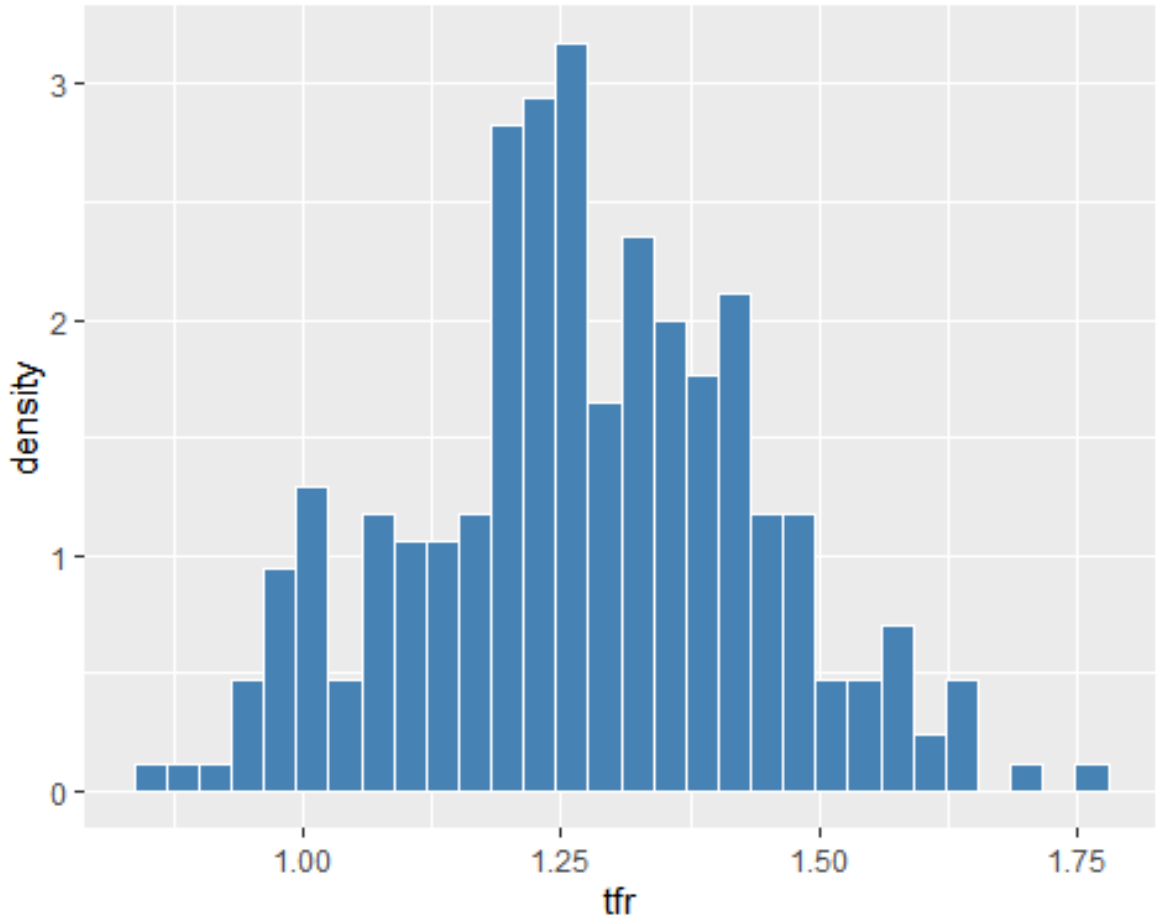
- 확률변수의 분포를 살펴보는 가장 편한 방법은 히스토그램을 그리는 것이다.

- ggplot에서는 geom_histogram을 사용하게 된다.

```
> hist <- tfr %>%
+   ggplot(mapping = aes(x = tfr))
>
> hist + geom_histogram(color = "white", fill =
"steelblue")
```

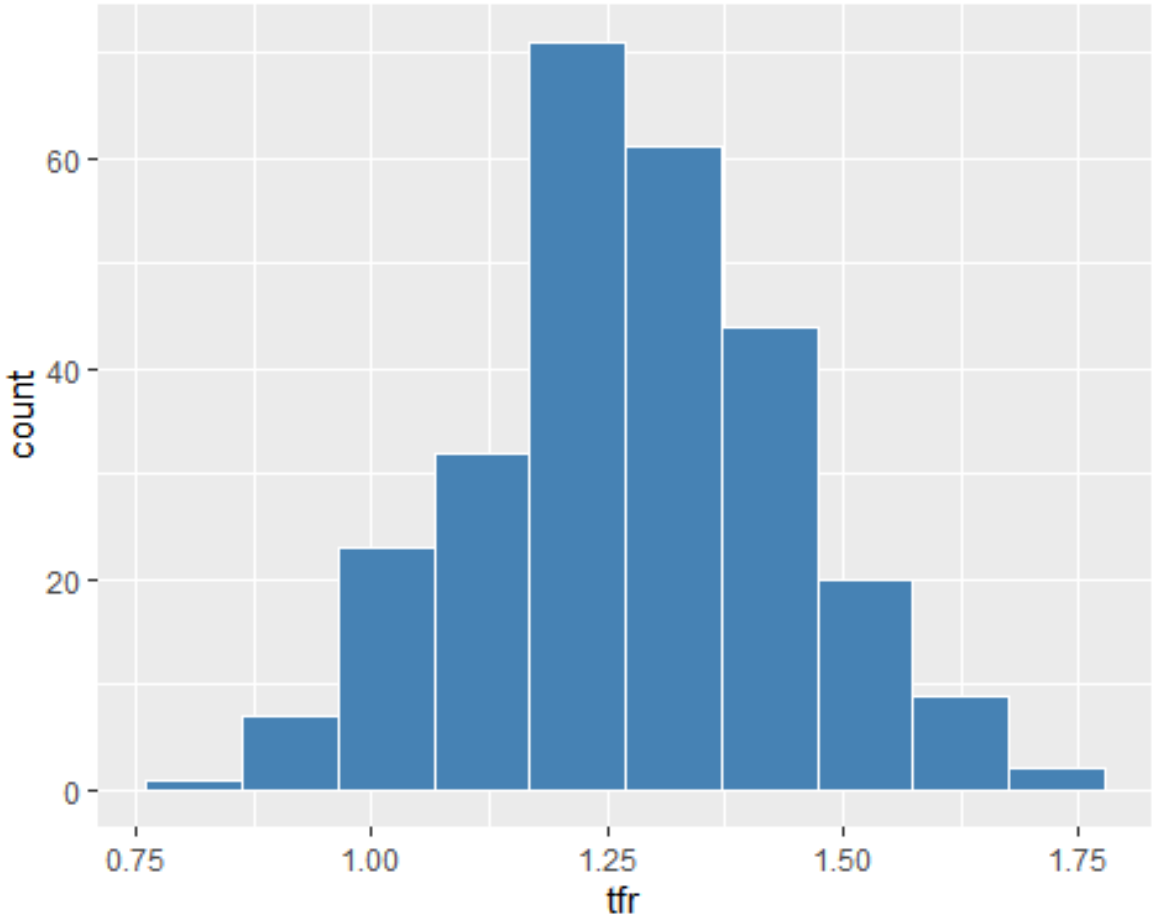

count
20
10
0
1.00
1.25
1.50
1.75
tfr

```
> hist + geom_histogram(aes(y = ..density..),
color = "white", fill = "steelblue")
```


density
3
2
1
0
1.00
1.25
1.50
1.75
tfr

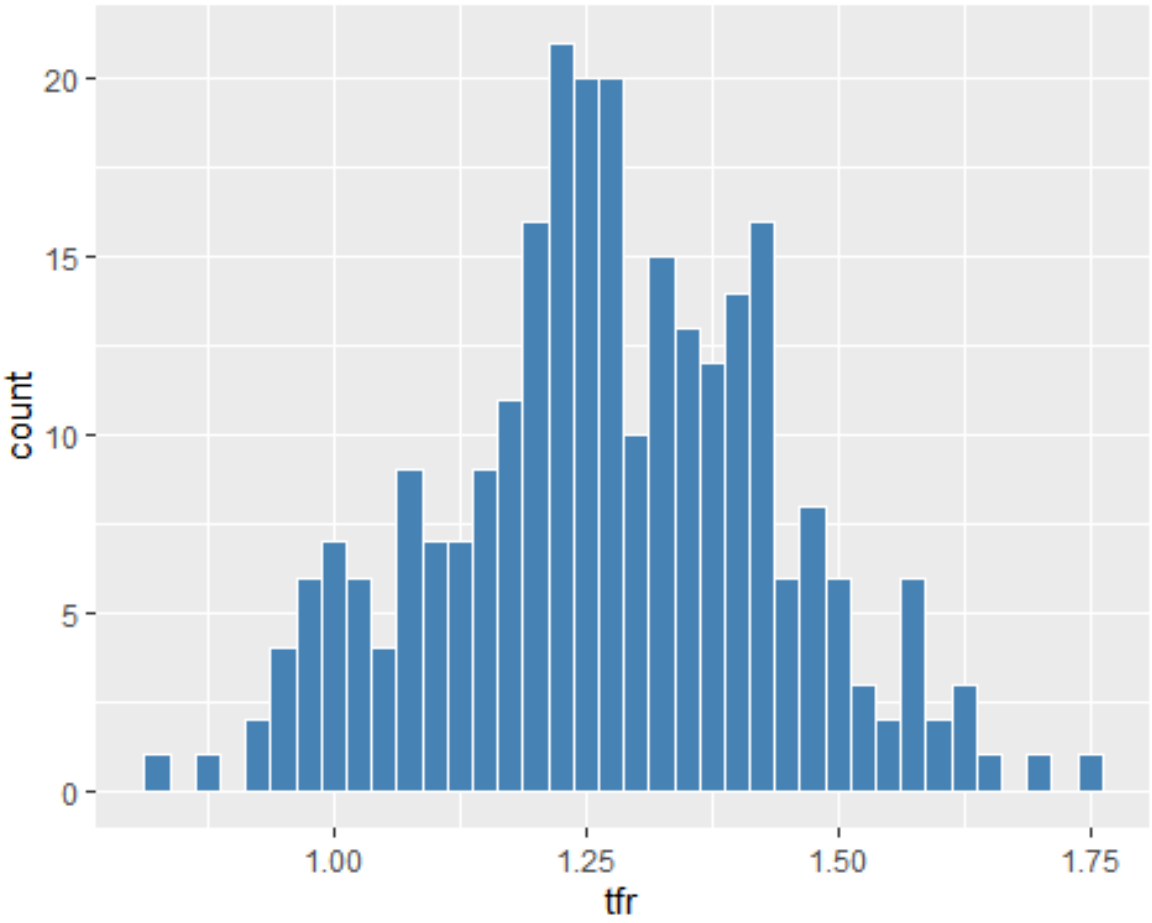
빈의 갯수 조절

```
> hist + geom_histogram(bins = 10, color =
"white", fill = "steelblue")
```


60
40
20
0
count
0.75
1.00
1.25
1.50
1.75
tfr

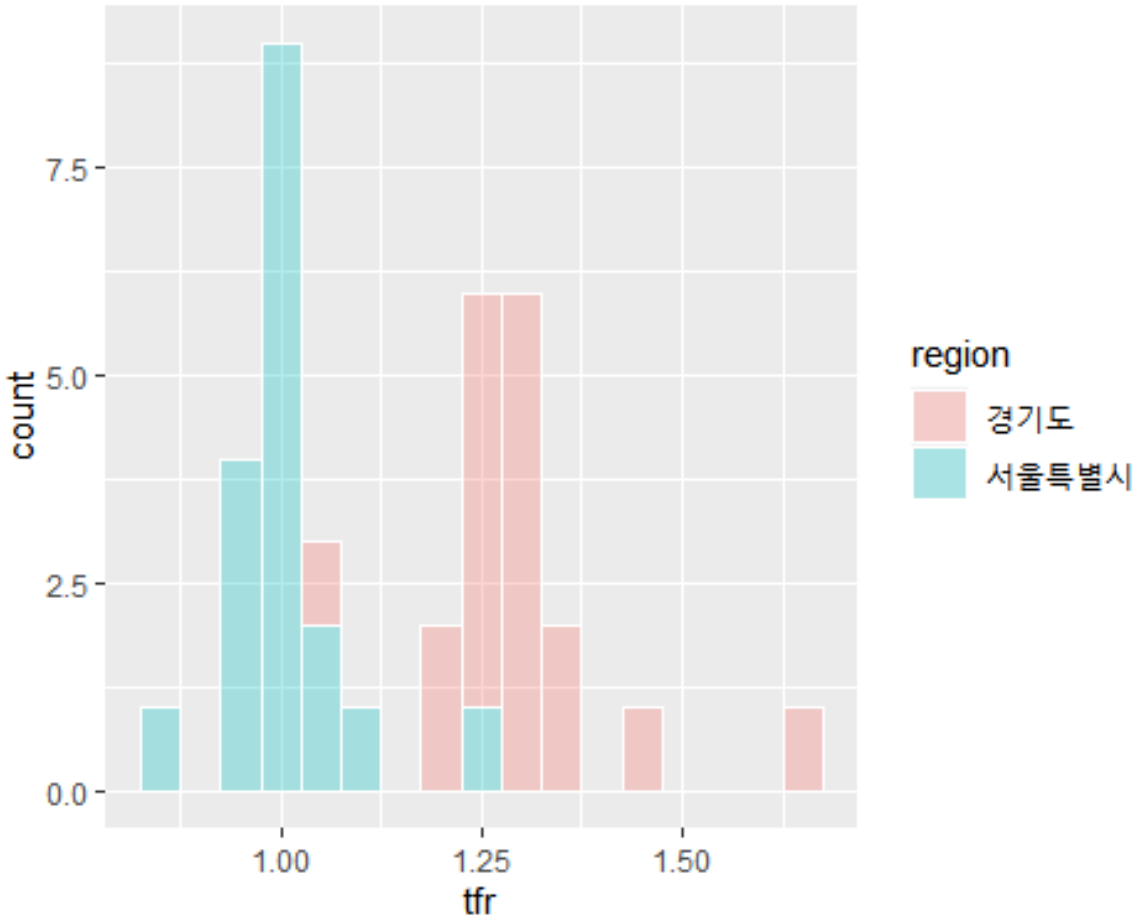
bin의 크기

```
> hist +
+   geom_histogram(binwidth = 0.025, color =
"white", fill = "steelblue")
```


20
15
10
5
0
count
1.00
1.25
1.50
1.75
tfr

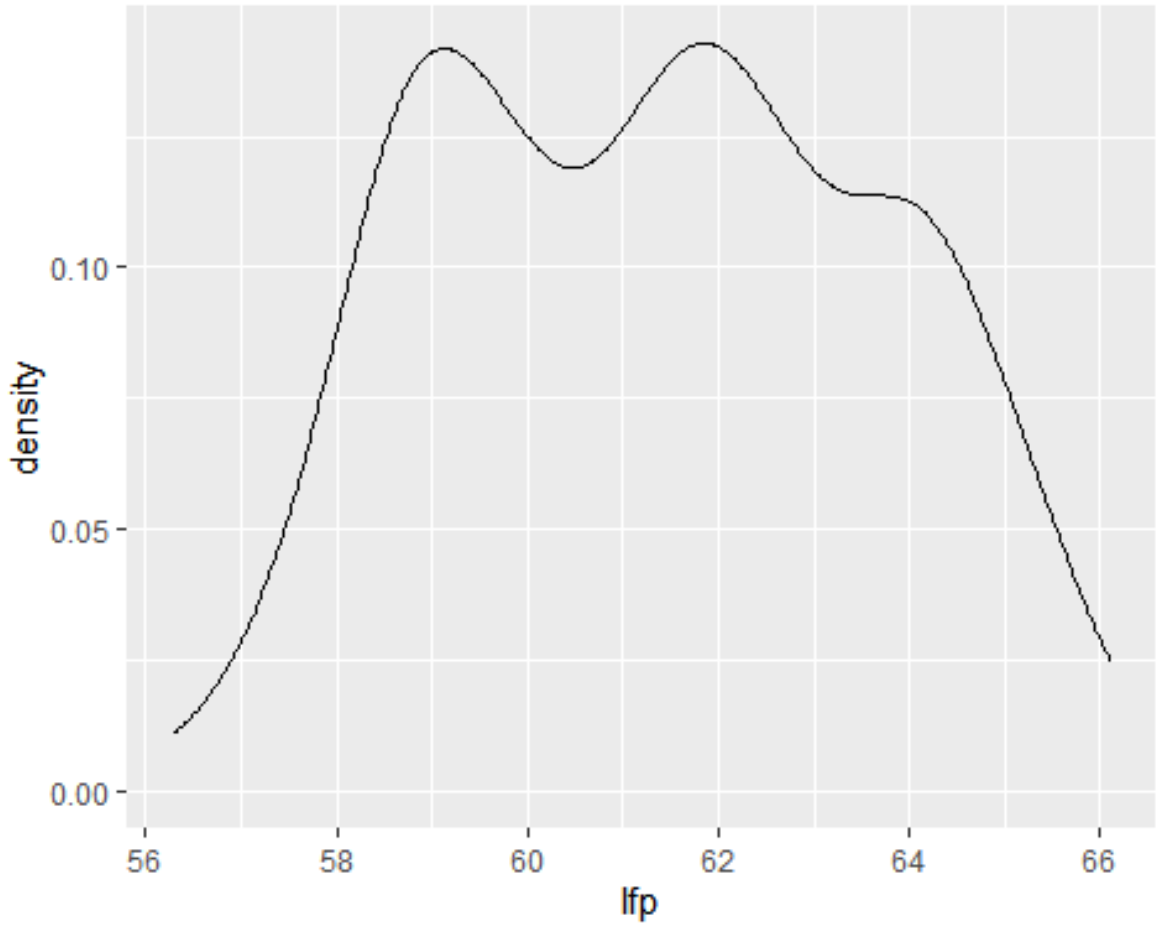
특정지역의 히스토그램

```
> metro <- c("서울특별시","경기도")
>
> p <- tfr %>%
+   filter(region %in% metro) %>%
+   ggplot(mapping = aes(x = tfr, fill = region))
>
> p + geom_histogram(alpha = 0.3, binwidth =
0.05, color = "white")
```


count
7.5
5.0
2.5
0.0
1.00
1.25
1.50
tfr
region
경기도
서울특별시

밀도함수 density 추가

```
> p <- ggplot(data = tfr, mapping = aes(x = lfp))
> p +
+   geom_density()
```


density
0.10
0.05
0.00
56
58
60
62
64
66
lfp

요약통계량

요약통계량

- ggplot의 편리한 점 중에 하나는 dplyr 패키지의 pipe 연산과 연계해서 요약통계량을 그릴 수 있다는 점
- 실습을 위해 AER 패키지의 CPSSW8 자료를 불러 들이자.

```
> library(dplyr)
> library(AER)
> data("CPSSW8")
```

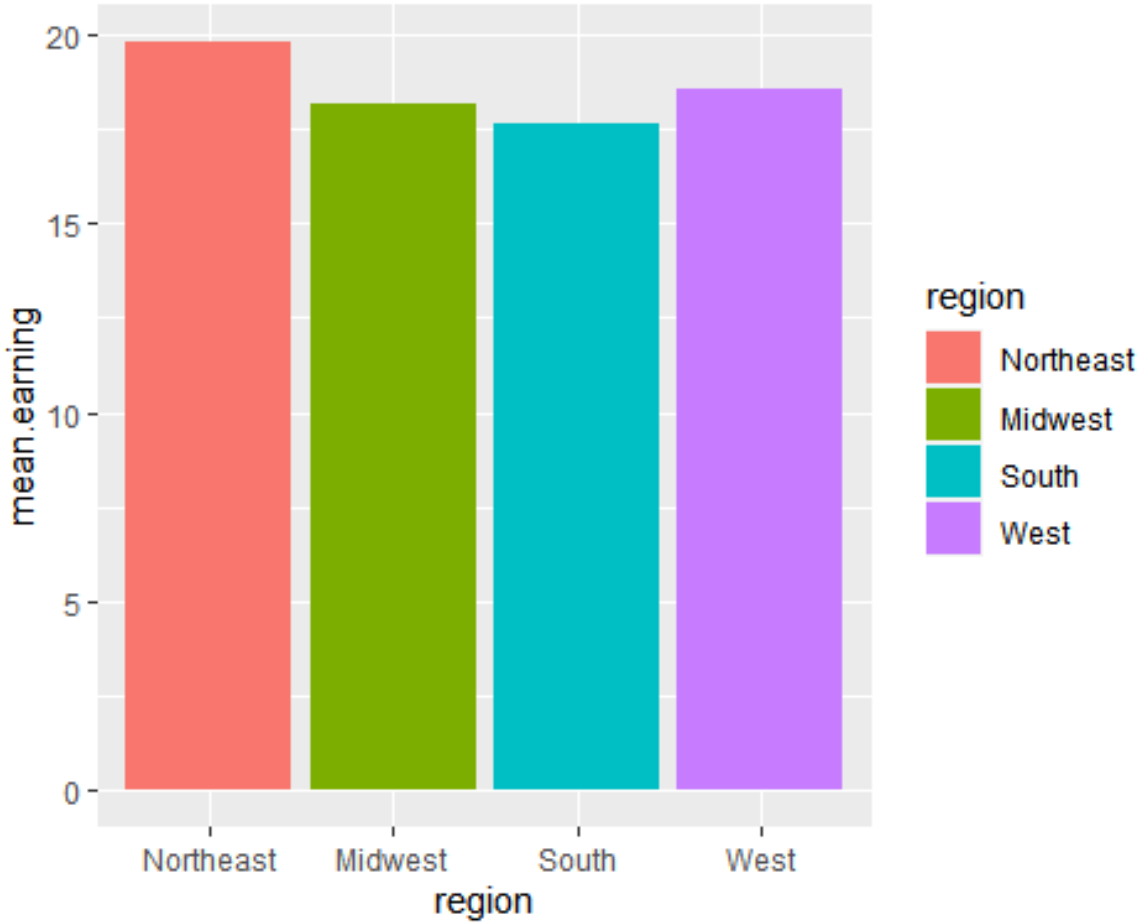
요약통계량 2

- 지역 region별 소득 수준을 살펴보자.

```
> earning.region <- CPSSW8 %>%
+   group_by(region) %>%
+   summarise(N = n(),
+             mean.earning = mean(earnings),
+             sd.earning = sd(earnings))
>
> head(earning.region, n= 3)
# A tibble: 3 x 4
  region         N mean.earning sd.earning
  <fct>      <int>        <dbl>      <dbl>
1 Northeast 12371         19.8       10.6
2 Midwest   15136         18.1        9.59
3 South     18963         17.6       10.0
```

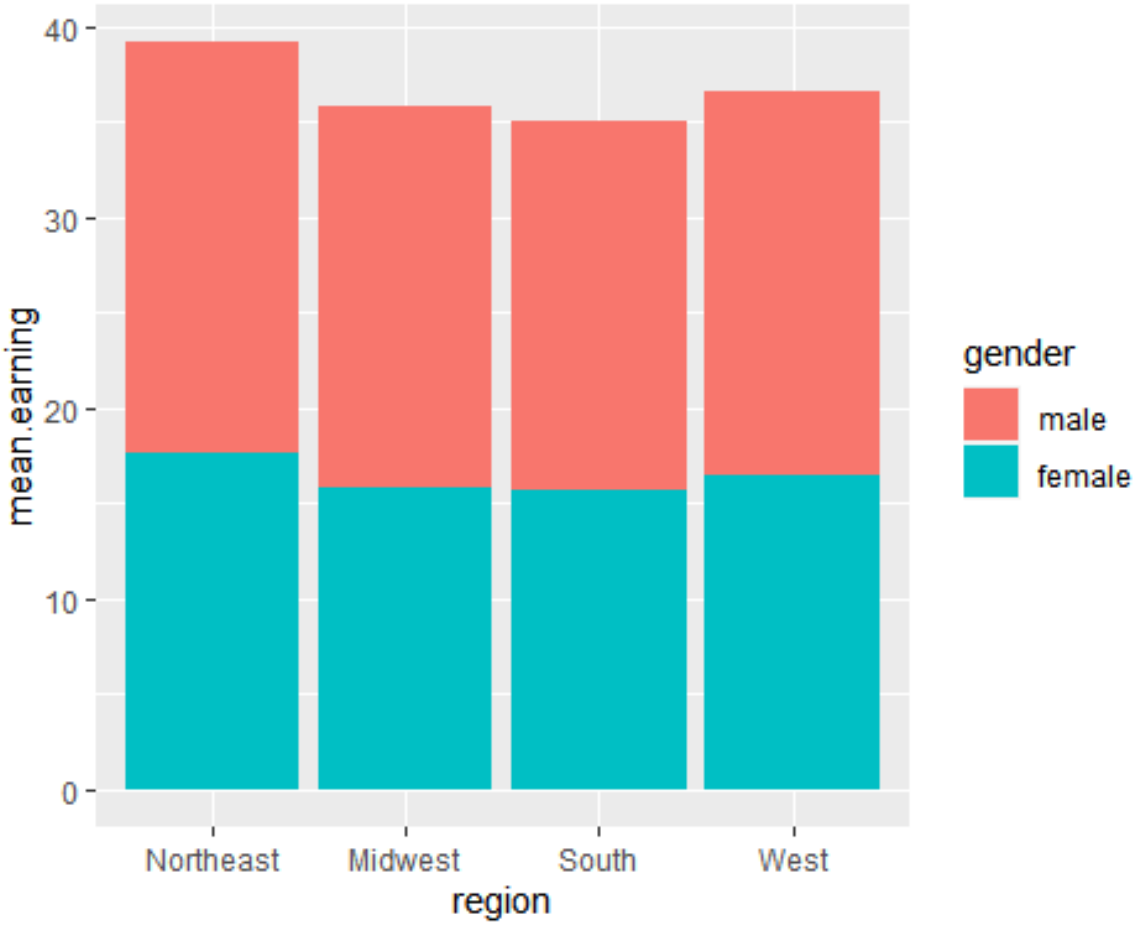
요약통계량 (bar graph,지역별)

```
> CPSSW8 %>%
+   group_by(region) %>%
+   summarise(N = n(),
+             mean.earning = mean(earnings),
+             sd.earning = sd(earnings)) %>%
+   ggplot(aes(x = region, y = mean.earning, fill
= region)) +
+   geom_bar(stat = "identity")
```


20
15
10
5
0
mean.earning
Northeast
Midwest
South
West
region
region
Northeast
Midwest
South
West

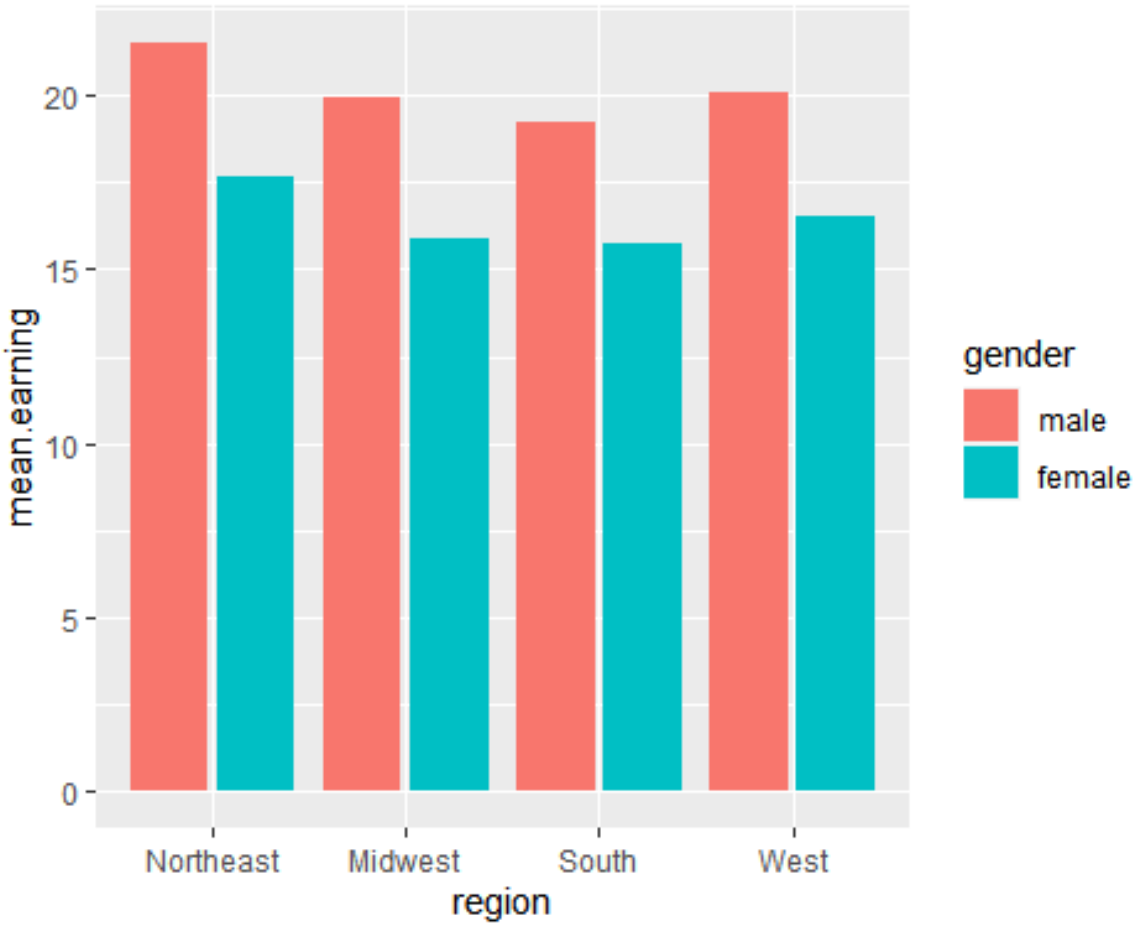
요약통계량 (bar graph,지역x성 별)

```
> CPSSW8 %>%
+   group_by(region, gender) %>%
+   summarise(N = n(),
+             mean.earning = mean(earnings),
+             sd.earning = sd(earnings)) %>%
+   ggplot(aes(x = region, y = mean.earning, fill
= gender)) +
+   geom_col()
```


40
30
20
10
0
mean.earning
Northeast
Midwest
South
West
region
gender
male
female

요약통계량 (bar graph,지역x성 별) 2

```
> CPSSW8 %>%
+   group_by(region, gender) %>%
+   summarise(N = n(),
+             mean.earning = mean(earnings),
+             sd.earning = sd(earnings)) %>%
+   ggplot(aes(x = region, y = mean.earning, fill
= gender)) +
+   geom_col(position = "dodge2")
```


20
15
10
5
0
mean.earning
Northeast
Midwest
South
West
region
gender
male
female

요약통계량 (facet_wrap,지역x성 별)

```
> CPSSW8 %>%
+   group_by(region, gender) %>%
+   summarise(N = n(),
+             mean.earning = mean(earnings),
+             sd.earning = sd(earnings)) %>%
+   ggplot(aes(x = gender, y = mean.earning, fill
= gender)) +
+   geom_col(position = "dodge2") +
+   coord_flip() +
+   facet_wrap(~region)
```


Northeast
Midwest
South
West
female
male
gender
0
5
10
15
20
mean.earning
gender
male
female

ggcharts

- 필요한 라이브러리 로딩

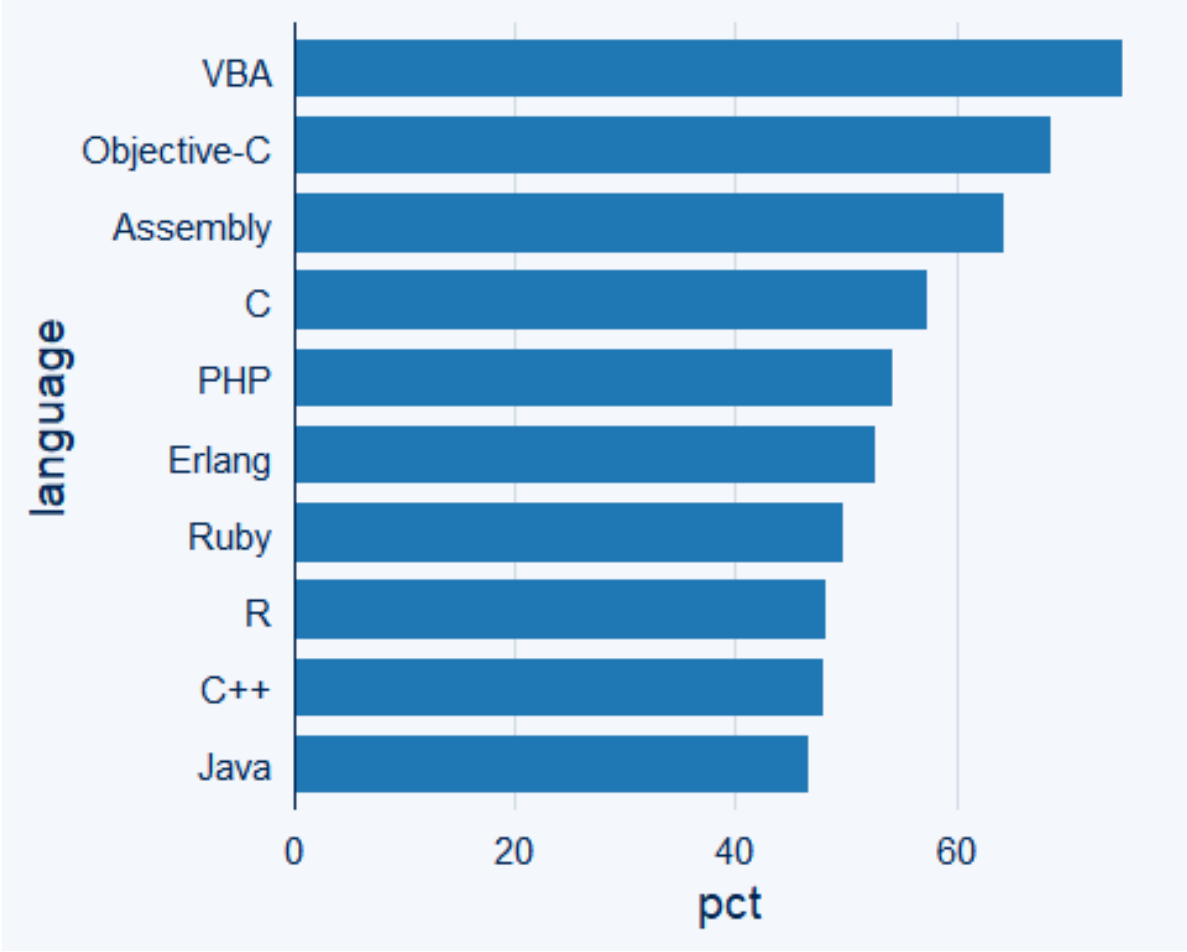
```
> library(dplyr)
> library(ggplot2)
> library(ggcharts)
```

데이터 입력

```
> dreaded_lang <- tibble::tribble(
+   ~language, ~pct,
+   "VBA", 75.2,
+   "Objective-C", 68.7,
+   "Assembly", 64.4,
+   "C", 57.5,
+   "PHP", 54.2,
+   "Erlang", 52.6,
+   "Ruby", 49.7,
+   "R", 48.3,
+   "C++", 48.0,
+   "Java", 46.6
+ )
```

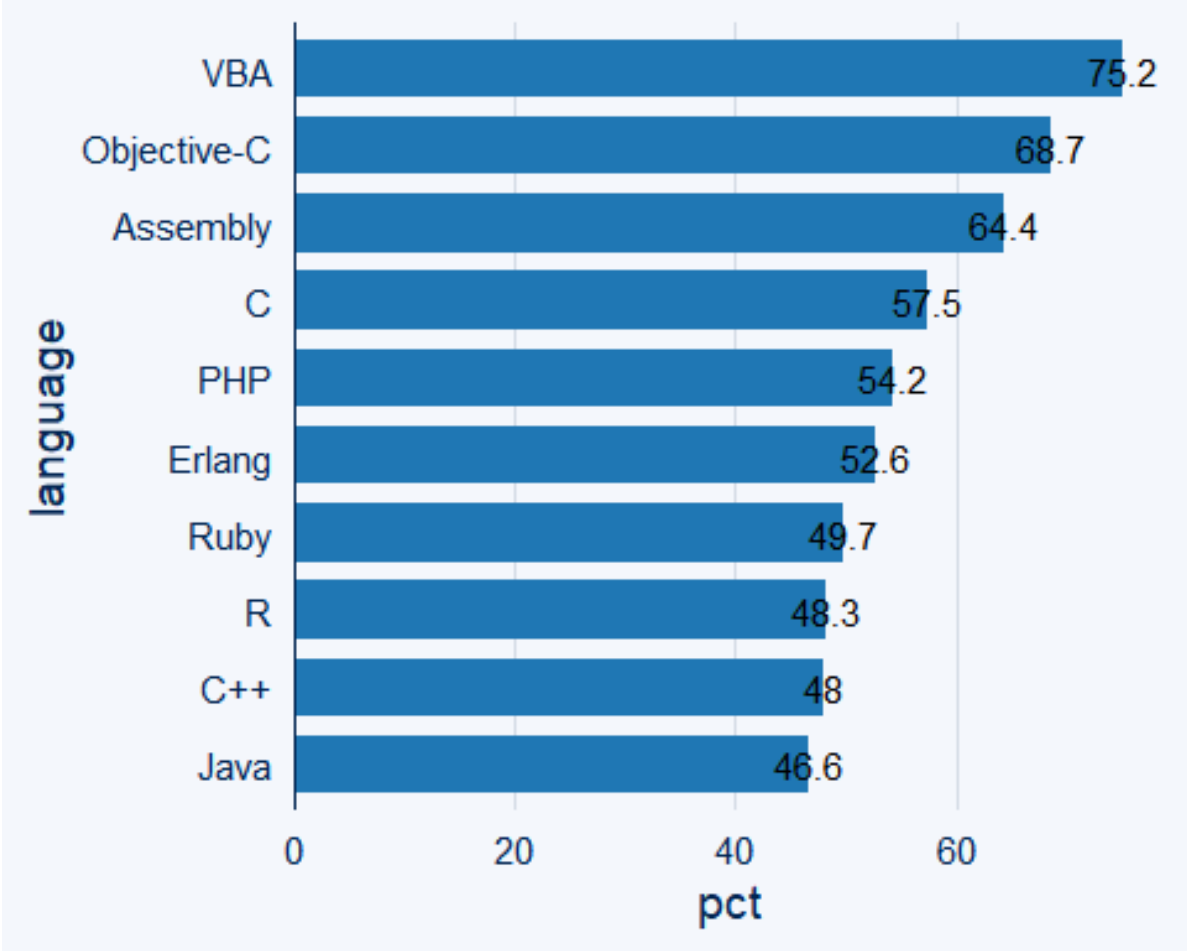
bar chart

```
> chart <- dreaded_lang %>%
+   bar_chart(language, pct) %>%
+   print()
```


VBA
Objective-C
Assembly
C
PHP
Erlang
Ruby
R
C++
Java
language
0
20
40
60
pct

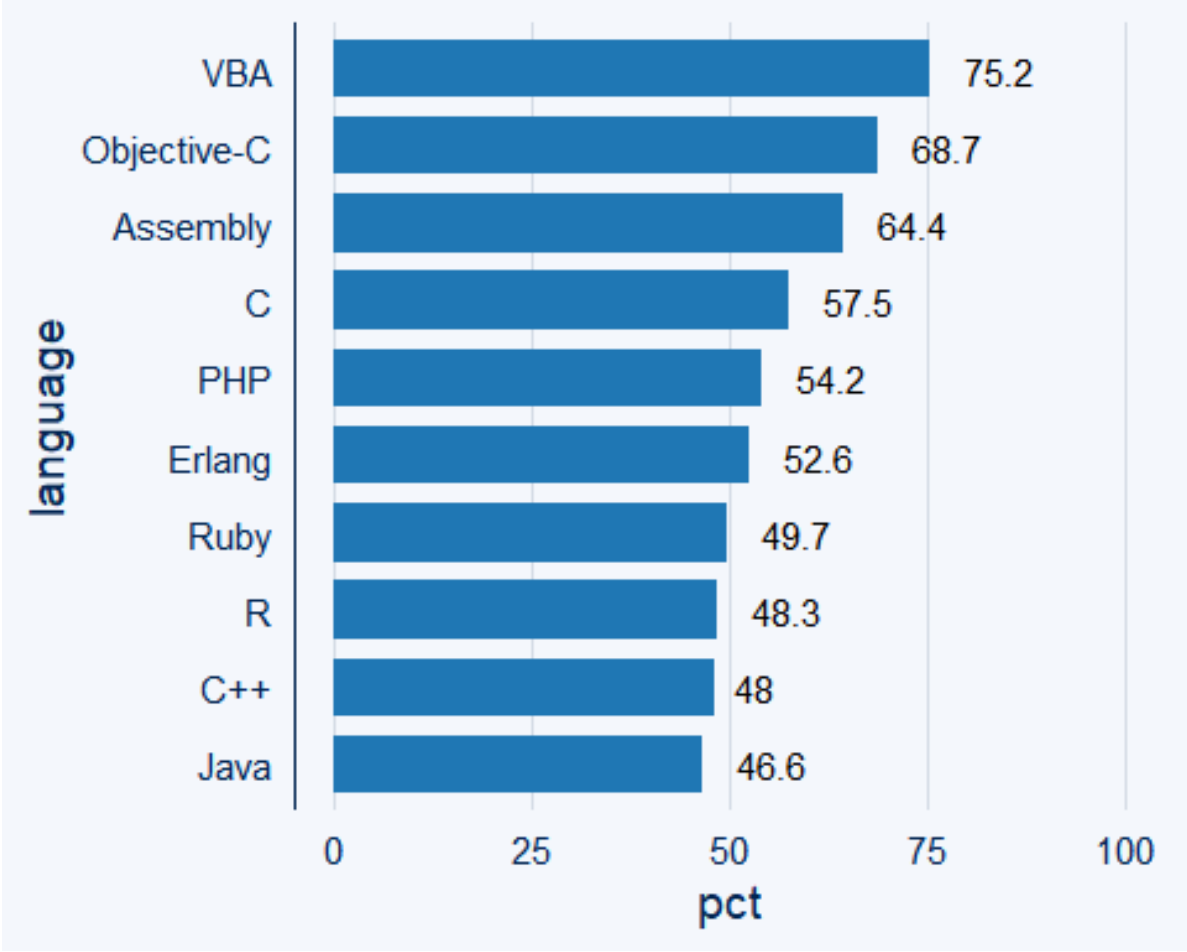
값-레이블 달기

```
> chart +
+   geom_text(aes(
+     x = language,
+     y = pct,
+     label = pct
+     ))
```


language
VBA
Objective-C
Assembly
C
PHP
Erlang
Ruby
R
C++
Java
75.2
68.7
64.4
57.5
54.2
52.6
49.7
48.3
48
46.6
0
20
40
60
pct

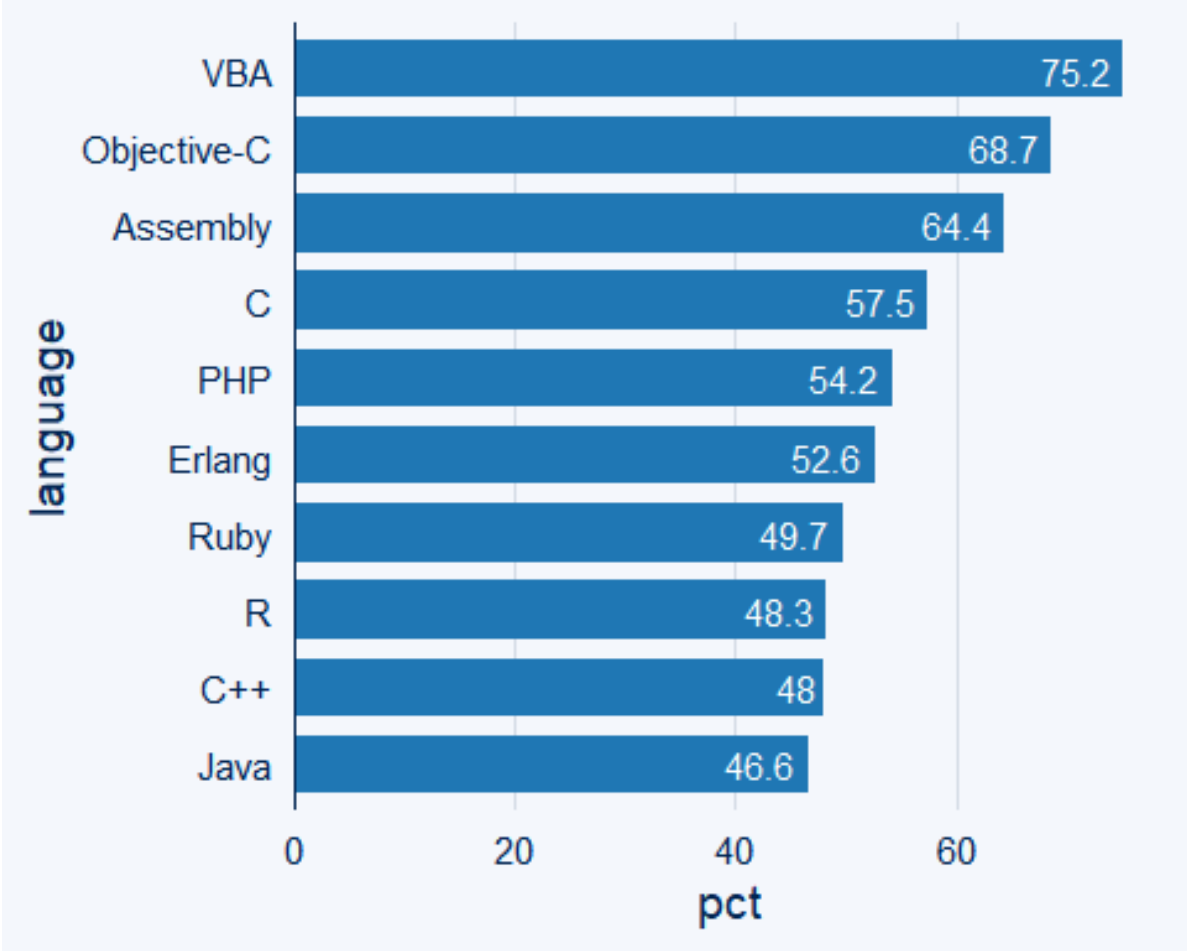
위치 조정

```
> chart +
+   geom_text(aes(label = pct, hjust = -0.5)) +
+   ylim(NA,100)
```


language
VBA
Objective-C
Assembly
C
PHP
Erlang
Ruby
R
C++
Java
75.2
68.7
64.4
57.5
54.2
52.6
49.7
48.3
48
46.6
0
25
50
75
100
pct

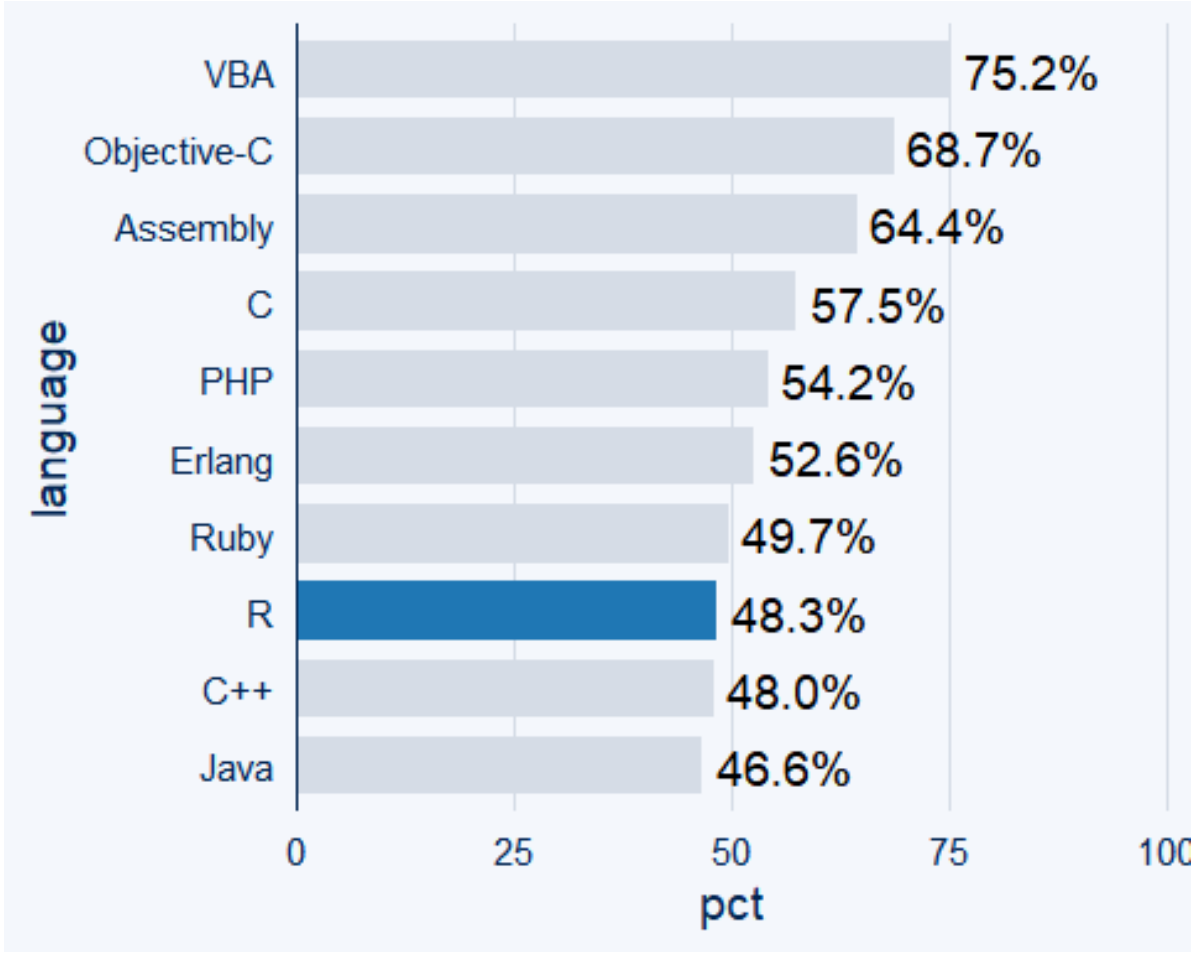
위치조정 2

```
> chart +
+   geom_text(aes(label = pct,
+                 hjust = 1.2),
+             color = "white"
+             )
```


language
VBA
Objective-C
Assembly
C
PHP
Erlang
Ruby
R
C++
Java
75.2
68.7
64.4
57.5
54.2
52.6
49.7
48.3
48
46.6
0
20
40
60
pct

완성된 챠트

```
> dreaded_lang %>%
+   mutate(label = sprintf("%1.1f%%", pct)) %>%
+   bar_chart(language, pct, highlight = "R",
bar_color = "black") +
+   geom_text(aes(label = label, hjust = -0.1),
size = 5) +
+   scale_y_continuous(
+     limits = c(0, 100),
+     expand = expansion()
+   )
```


language
VBA
Objective-C
Assembly
C
PHP
Erlang
Ruby
R
C++
Java
75.2%
68.7%
64.4%
57.5%
54.2%
52.6%
49.7%
48.3%
48.0%
46.6%
0
25
50
75
100
pct