V. 상관(Correlation)의 시각화

상관관계(Correlation)의 시각화는 X 축과 Y 축으로 표현된 데이터가 전반적으로 어떠한 관계를 가지는지를 표현한 시각화이다. 관계의 시각화는 단순히 X 축 데이터의 분포에 따라 Y 축 데이터의 분포가 어떻게 분포하는지를 위주로 표현되는데 관계의 시각화에 따라 나타나는 상관관계는 부가적인 통계 처리 정보가 보완되지 않는다면 단순 관계에 불과할 뿐 인과관계를 표현하는 것은 아니다. 상관관계의 시각화를 설명하기 위해 앞으로 졸업자수에 따른 취업자 수의 시각화를 볼 것이다. 당연히 졸업자가 많은 학과는 상대적으로 취업자 수도 많게 될 것이다. 하지만 이러한 시각화 결과 만으로 단순히 졸업자가 많은 과는 취업자도 많다는 결론에 도달해서는 안된다는 것이다. 이를 증명하기 위해 다양한 통계적 검증 과정이 필요하고 이 검증 과정을 통해 독립변수가 종속 변수를 변화시키는 원인이라는 결론에 도달해야 한다. 그렇다면 관계의 시각화는 필요없는 것일까? 그렇지 않다. 어떤 독립 변수가 어떤 종속 변수와 밀접하게 관계가 있는지를 먼저 관계의 시각화를 통해 확인하여야 통계적 검증을 수행할 수 있다. 그저 데이터가 존재한다는 이유만으로 모든 변수간의 인과 관계를 검증할 수는 없는 것이다. 과거 컴퓨터 시스템이 발달하지 못했던 때는 여러가지 통계적 방법론을 사용하여 이 상관관계를 확인하였지만 1 장에서 본 바와 같이 통계적 방법론이 동일해도 시각화의 결과 큰 관계가 없는 경우도 있다. 따라서 상관관계를 확인하기 위한 통계치와 시각화 결과를 반드시 같이 확인해야 한다.

상관관계는 양의 상관관계와 음의 상관관계로 나뉜다. 양의 상관관계는 독립변수(보통 X 축 변수)가 증가할수록 종속변수(보통 Y 축변수)가 증가하는 관계이고 음의 상관관계는 독립변수가 증가할수록 독립변수가 감소하는 관계를 말한다. 이러한 상관관계는 상관계수라는 수치로 얼마나 강한 상관관계를 가지는지를 표현한다. 보통 0.7 이상의 상관계수는 매우 강한 상관관계가 있다고 간주된다.[1]

[1] https://www.reneshbedre.com/blog/correlation-analysis-r.html

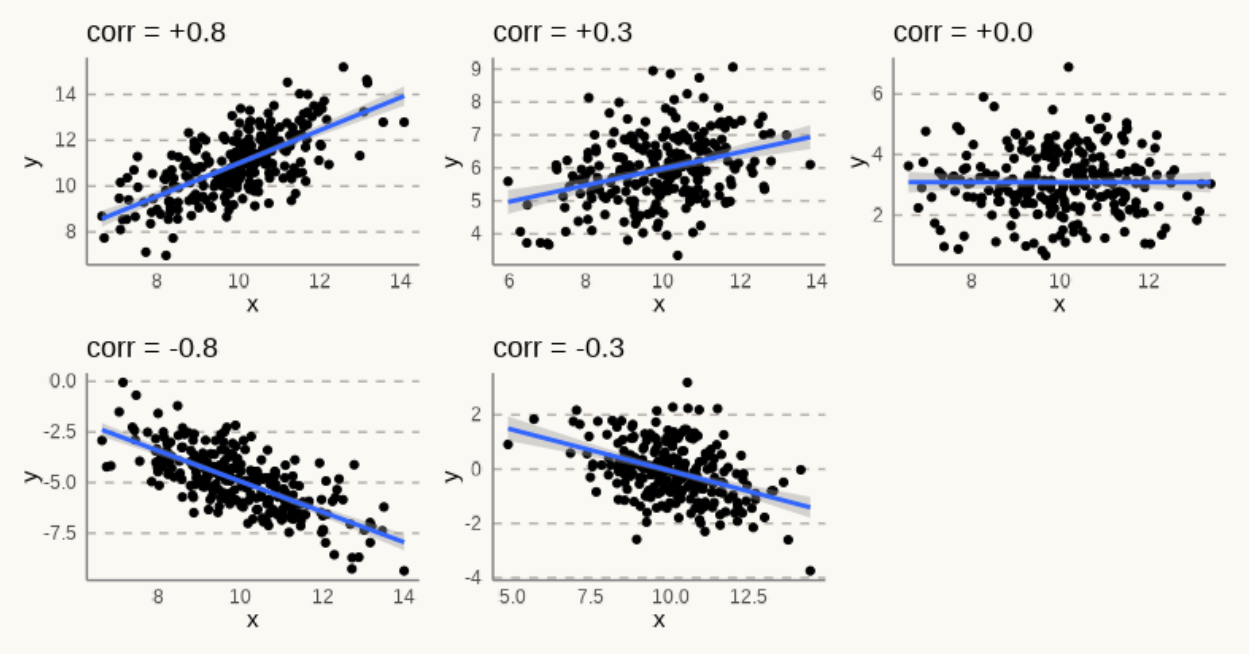


그림 5-1 상관관계와 상관계수

데이터의 상관관계를 확인하기 위한 시각화를 살펴보기 전에 이번 장에서 사용할 데이터를 다음과 같이 준비한다.

```
## 과정구분의 순서를 맞추기 위해 과정구분을 팩터로 설정하고 레벨의 순서를 설정
df_취업통계$과정구분 <- fct_relevel(df_취업통계$과정구분, '전문대학과정', '대학과정',
 '대학원과정')

## 대계열의 순서를 맞추기 위해 대계열을 팩터로 설정하고 레벨의 순서를 설정
df_취업통계$대계열 <- fct_relevel(df_취업통계$대계열, '인문계열', '사회계열', '교육계
열', '자연계열', '공학계열', '의약계열', '예체능계열')

## 시각화에 사용할 전체 테마를 설정
theme_set(
  theme(
    axis.ticks = element_blank(),
    axis.line = element_line(colour = "grey50"),
    panel.grid = element_line(color = "#b4aea9"),
    panel.grid.minor = element_blank(),
    panel.grid.major.x = element_blank(),
    panel.grid.major.y = element_line(linetype = "dashed"),
    panel.background = element_rect(fill = "#fbf9f4", color = "#fbf9f4"),
    plot.background = element_rect(fill = "#fbf9f4", color = "#fbf9f4"),
    legend.background = element_rect(fill = "#fbf9f4", color = "#fbf9f4"),
    legend.key = element_rect(fill = "#fbf9f4", color = NA)
```

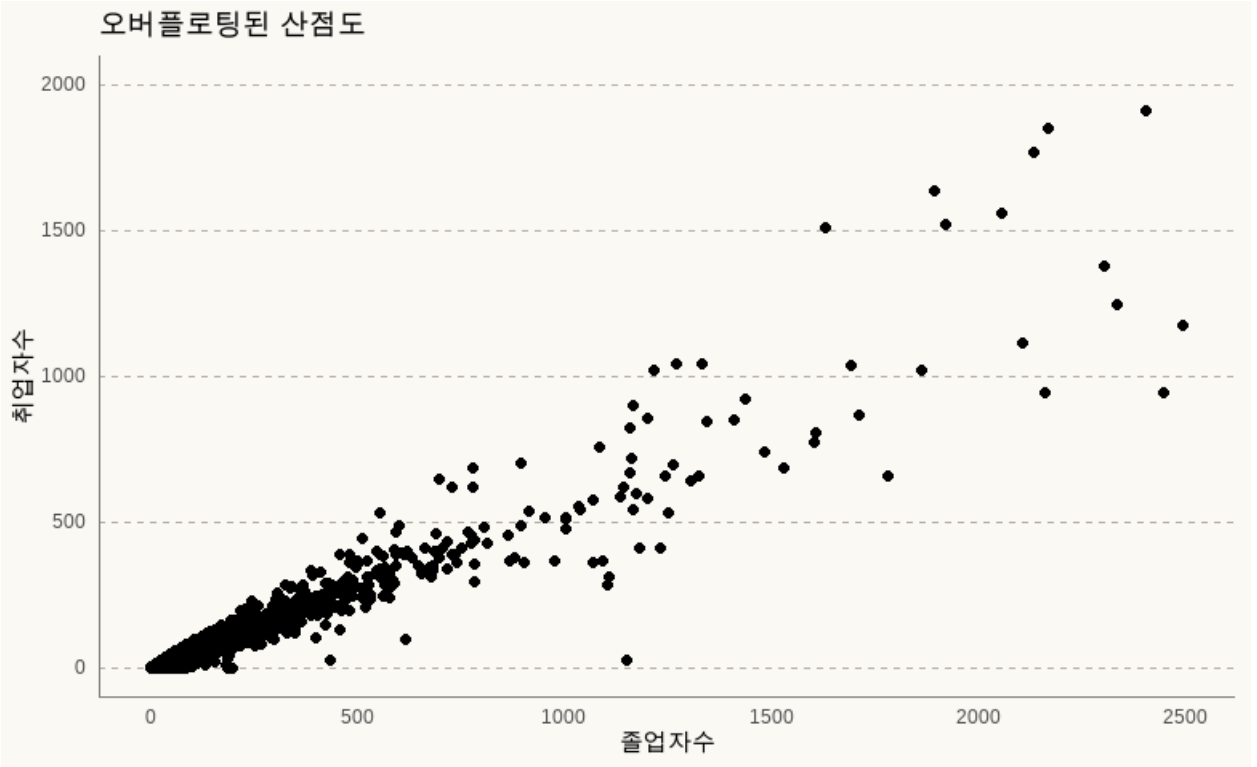
```
    )
  )
```

1. 산점도(Scatter Plot)

산점도는 x, y 두개의 변수에 따른 데이터의 위치를 점으로 표현한 시각화 방법이다. 산점도를 통해 데이터의 전반적 분포와 X 축의 독립변수에 따른 Y 축의 종속변수가 어떻게 분포하는지를 한눈에 확인할 수 있고 그 분포 모양에 따라 상관관계와 상관강도를 알아볼 수 있다.

`ggplot2` 에서 산점도를 시각화하기 위해서는 `geom_point()`를 사용한다.

```
df_취업통계 |>
  ggplot() +
  ## X 축이 졸업자_계, Y 축이 취업자_합계_계에 매핑된 geom_point 레이어 생성
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계)) +
  ## X 축과 Y 축의 범위를 설정
  lims(x = c(0, 2500), y = c(0, 2000)) +
  labs(title = '오버플로팅된 산점도', x = '졸업자수', y = '취업자수')
```



실행결과5-1. 오버플로팅된 산점도

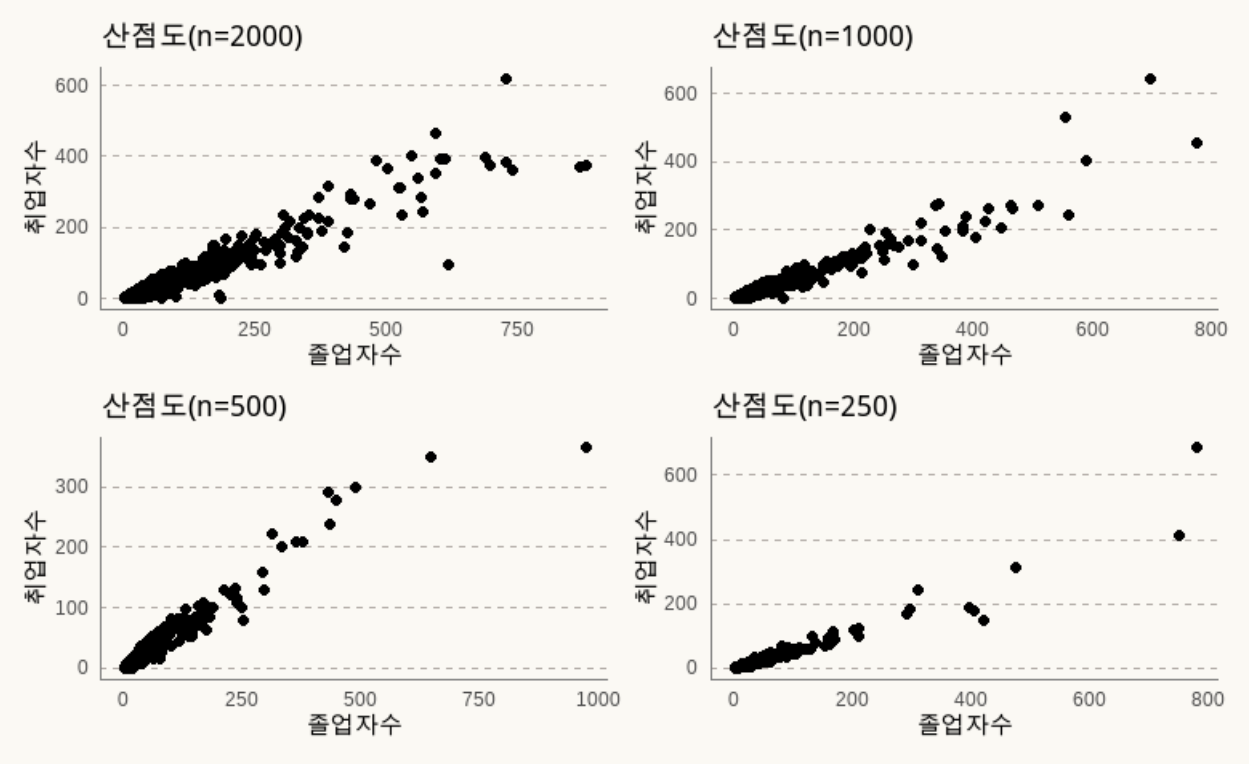
산점도를 만들때 주의해야 할 것이 산점도에 너무 많은 점이 한곳에 표현되는 오버플로팅(Over-Ploting)이 일어나지 않도록 해야한다는 점이다. 위의 산점도를 보면 다행이 선형 상관관계가 눈에 보이지만 좌측 하단 구간은 데이터가 집중되다보니 검정색 지역만 눈에 보인다. 그 지역에서도 데이터가 특히 밀집된 지역이 존재하겠지만 데이터가 오버플로팅되어 이러한 데이터의 밀집을 전혀 확인할 수 없다. 이런 오버플로팅을 해결하는 방법은 다음과 같다.

- 데이터 사이즈의 축소

플로팅되는 데이터를 랜덤 샘플링과 같은 방법을 통해 축소시키는 방법이다. 다음은 오버플로팅 된 데이터를 랜덤 샘플링을 통해 2000, 1000, 500 개로 축소한 산점도이다.

```
## 랜덤 샘플링을 위한 난수 설정
set.seed(123)

## df_취업통계에서 일부 데이터를 필터링하고 2000 개 데이터 샘플링
df_취업통계_sample <- df_취업통계 |>
  filter(졸업자_계 <= 1000, 취업자_합계_계 <= 1000) |>
  sample_n(2000)

## df_취업통계에서 일부 데이터를 필터링하고 1000 개 데이터 샘플링
df_취업통계_sample_1000 <- df_취업통계 |>
  filter(졸업자_계 <= 1000, 취업자_합계_계 <= 1000) |>
  sample_n(1000)

## df_취업통계에서 일부 데이터를 필터링하고 500 개 데이터 샘플링
df_취업통계_sample_500 <- df_취업통계 |>
  filter(졸업자_계 <= 1000, 취업자_합계_계 <= 1000) |>
  sample_n(500)

## df_취업통계에서 일부 데이터를 필터링하고 500 개 데이터 샘플링
df_취업통계_sample_250 <- df_취업통계 |>
  filter(졸업자_계 <= 1000, 취업자_합계_계 <= 1000) |>
  sample_n(250)

df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계)) +
  labs(title = '산점도(n=2000)', x = '졸업자수', y = '취업자수')
```

```r
df_취업통계_sample_1000 |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계)) +
  labs(title = '산점도(n=1000)', x = '졸업자수', y = '취업자수')

df_취업통계_sample_500 |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계)) +
  labs(title = '산점도(n=500)', x = '졸업자수', y = '취업자수')

df_취업통계_sample_250 |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계)) +
  labs(title = '산점도(n=250)', x = '졸업자수', y = '취업자수')
```

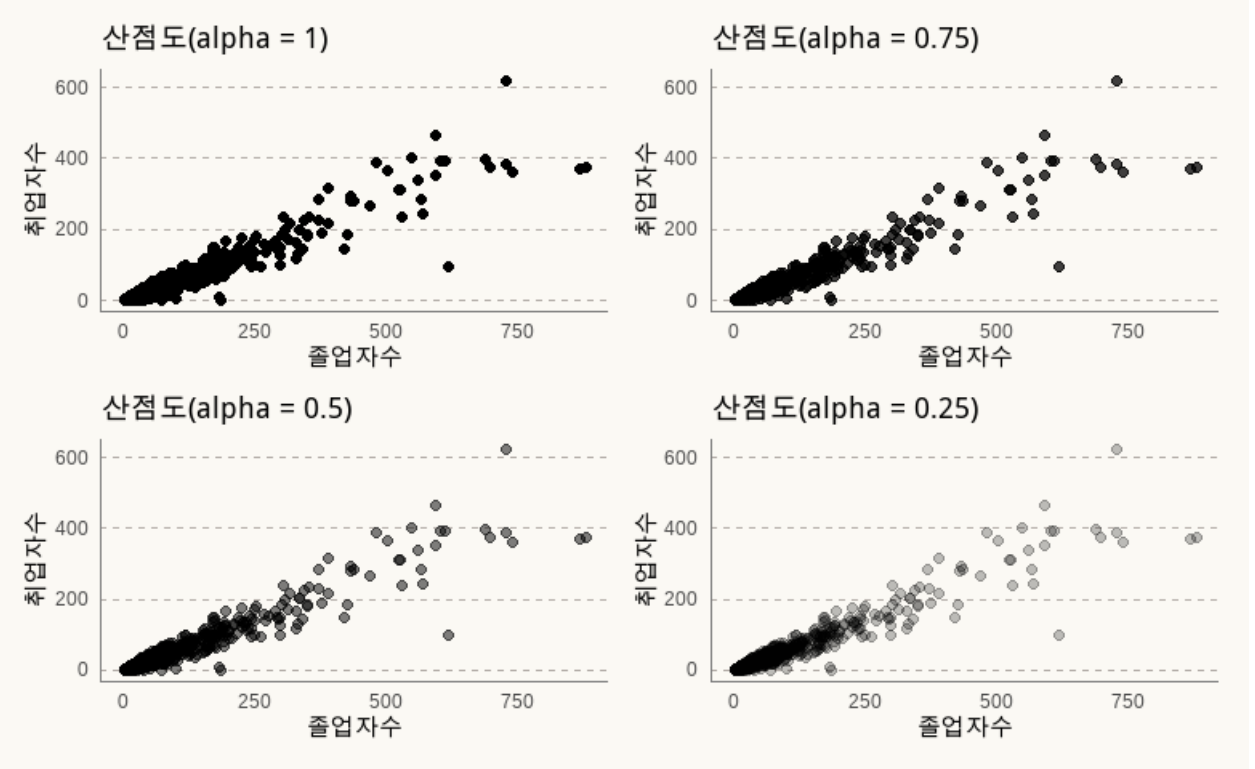


실행결과5-2. 샘플수 조절들을 통한 오버플로팅 제거

- 표시점의 투명도(alpha) 조절

위의 실행결과를 보면 샘플수 조절만으로는 오버플로팅이 완전히 해결되지는 않아 보인다. 이때 사용할 수 있는 또 하나의 방법이 표시점의 투명도를 조절하는 방법이다. 샘플수 축소와 투명도 조절을 잘 사용하면 효과적으로 오버플로팅을 제거할 수 있다.

```
df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), alpha = 1) +
  labs(title = '산점도(alpha = 1)', x = '졸업자수', y = '취업자수')

df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), alpha = 0.75) +
  labs(title = '산점도(alpha = 0.75)', x = '졸업자수', y = '취업자수')

df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), alpha = 0.5) +
  labs(title = '산점도(alpha = 0.5)', x = '졸업자수', y = '취업자수')

df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), alpha = 0.25) +
  labs(title = '산점도(alpha = 0.25)', x = '졸업자수', y = '취업자수')
```



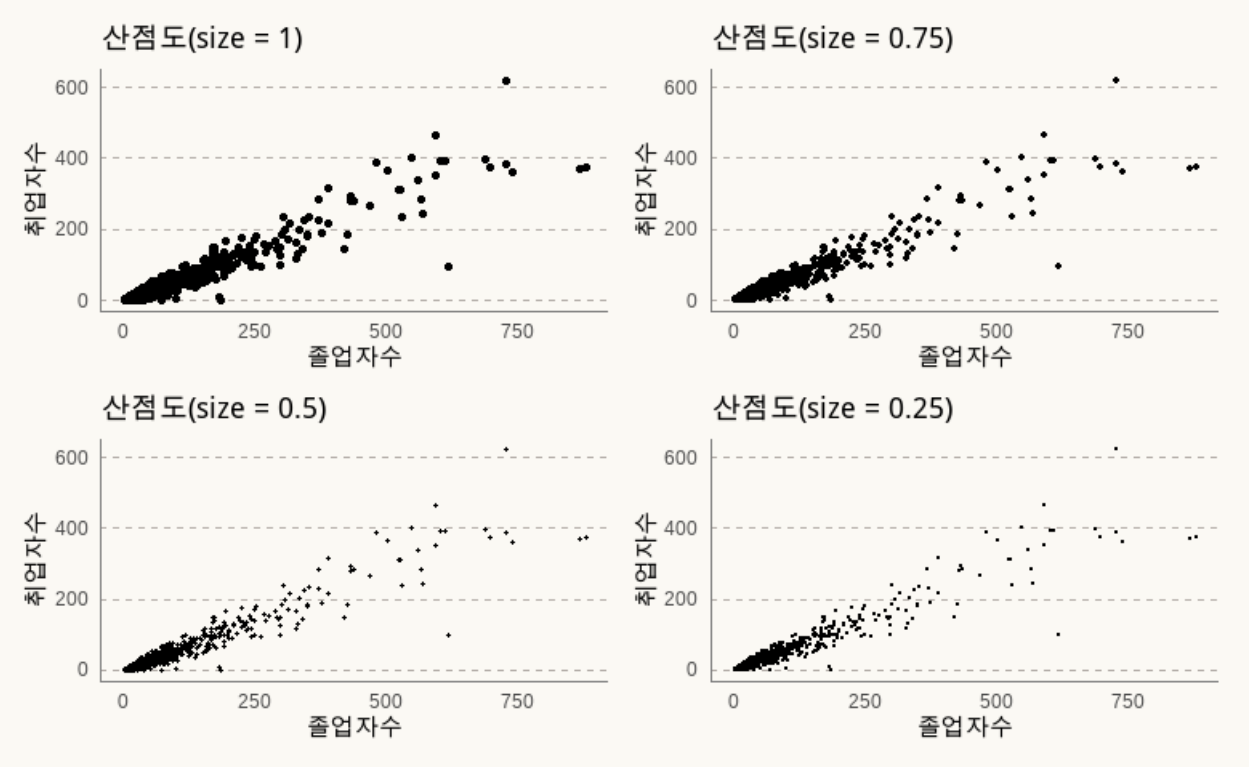
실행결과5-3. 투명도 조절을 통한 오버플로팅 제거

- 표시점의 크기(size) 조절

산점도의 오버플로팅을 제거하는데 사용할 수 있는 또 하나의 방법은 표시점의 크기를 조절하는 것이다.

```
df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), size = 1) +
  labs(title = '산점도(size = 1)', x = '졸업자수', y = '취업자수')

df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), size = 0.75) +
  labs(title = '산점도(size = 0.75)', x = '졸업자수', y = '취업자수')

df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), size = 0.5) +
  labs(title = '산점도(size = 0.5)', x = '졸업자수', y = '취업자수')
```

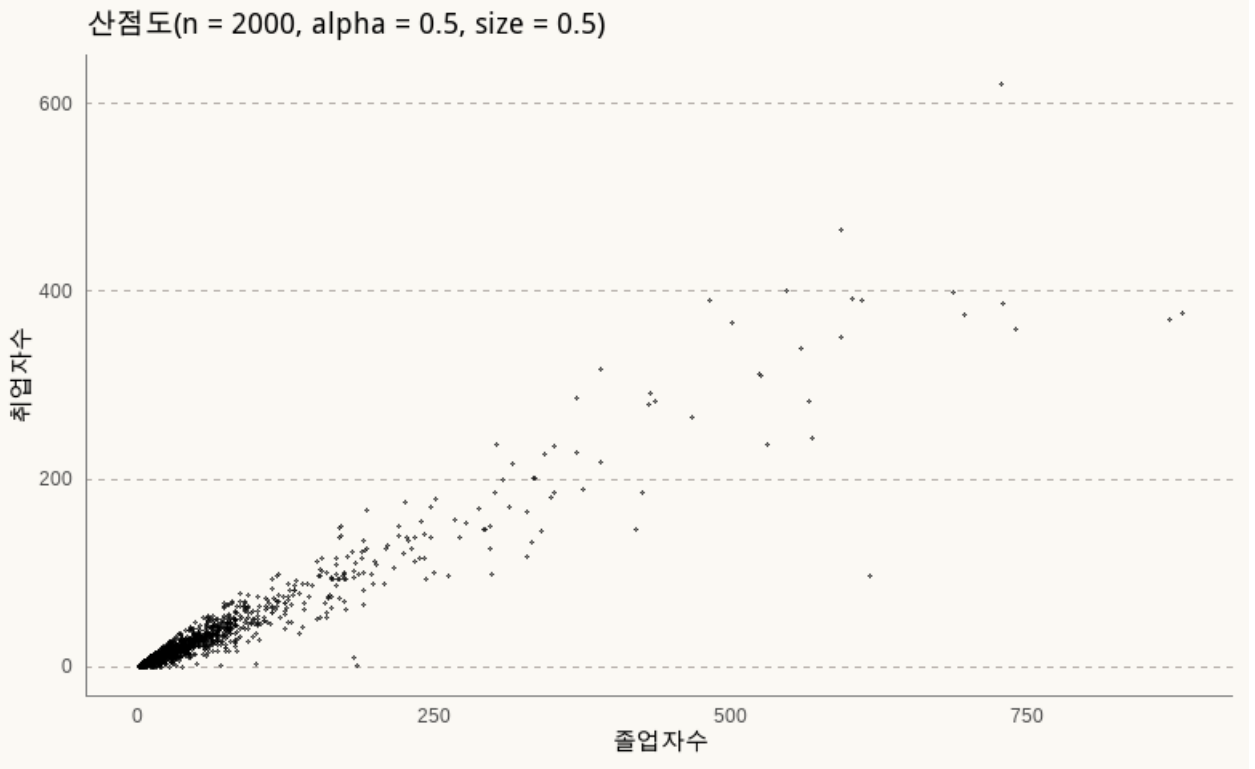
```
df_취업통계_sample |>
  ggplot() +
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), size = '.') +
  labs(title = '산점도(size = 0.25)', x = '졸업자수', y = '취업자수')
```



실행결과 5-4. 사이즈 조절을 통한 오버플로팅 제거

위의 세가지 방법을 모두 사용하여 오버플로팅을 다음과 같이 제거할 수 있다.

```
df_취업통계_sample |>
  ggplot() +
  ## x 축을 졸업자_계, y 축을 취업자_합계_계로 매핑하고 size 와 alpha 를 설정한 geom_point 레이어 생성
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계), size = 0.5, alpha = 0.5) +
  labs(title = '산점도(n = 2000, alpha = 0.5, size = 0.5)', x = '졸업자수', y = '취업자수')
```



실행결과 5-5. 세가지 미적요소 조절을 통한 오버플로팅 제거

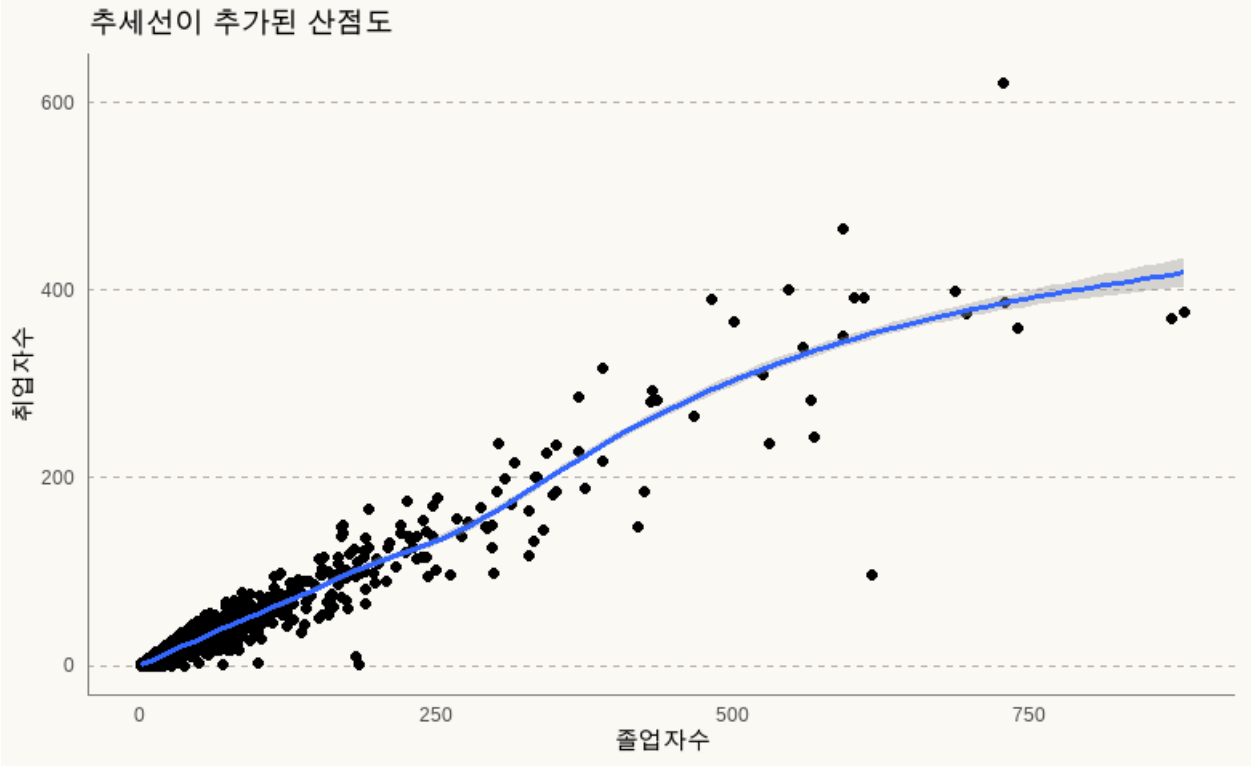
오버플로팅을 해결하는 방법으로 추가적으로 사용할 수 있는 방법은 밀도 분포 시각화를 사용하는 방법이 있다. 이는 다음 장에서 추가적으로 설명하겠다.

1.1. 추세선 산점도(scatter plot)

산점도를 사용하는 가장 큰 이유는 데이터의 전체적인 분포를 보면서 데이터의 관계성을 확인하는데 있다. 이 관계성을 확인하는 과정에서 산점도는 사실 노이즈가 많이 발생한다. 이 노이즈들 사이에서 데이터의 관계성 패턴을 시각화하는 것이 추세선이다. `ggplot2` 는 이 추세선을 `geom_smooth()`를 통해 제공한다. 다음은 `geom_smooth()`를 사용해 앞선 산점도에 추세선을 표현한 결과이다.

```
## 샘플수가 2000 개인 df_취업통계_sample 을 ggplot 객체로 생성
p_scatter <- df_취업통계_sample |>
  ggplot() +
  labs(x = '졸업자수', y = '취업자수')

p_scatter1 <- p_scatter +
  ## X 축을 졸업자_계, Y 축을 취업자_합계_계로 매핑한 geom_point 로 산점도 레이어 생성
```

```
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계)) +
  ## X 축을 졸업자_계, Y 축을 취업자_합계_계로 매핑한 geom_smooth 로 추세선 레이어 생성
  geom_smooth(aes(x = 졸업자_계, y = 취업자_합계_계)) +
  ## 제목, X 축 제목, Y 축 제목 설정
  labs(title = '추세선이 추가된 산점도')

p_scatter1
```

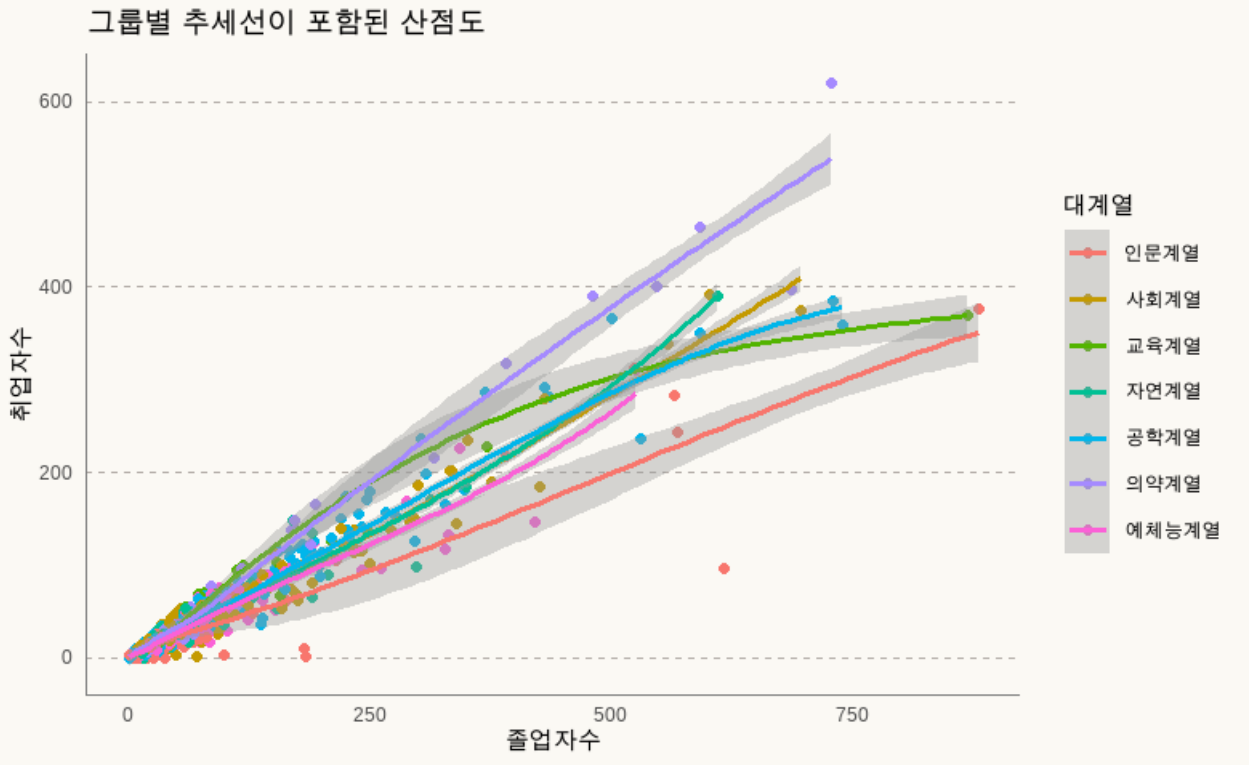


실행결과 5-6. 추세선이 추가된 산점도

앞의 실행결과 같이 추세선은 전체를 대상으로 생성할 수도 있지만 그룹별로 추세선을 생성할 수도 있다. 다음은 앞선 산점도를 대계열별로 그룹화하고 각각의 그룹에 대한 추세선을 추가한 산점도이다.

```
p_scatter2 <- p_scatter +
  ## X 축을 졸업자_계, Y 축을 취업자_합계_계, color 를 대계열로 매핑한 geom_point 로 산점도 레이어 생성
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계, color = 대계열)) +
  ## X 축을 졸업자_계, Y 축을 취업자_합계_계, color 를 대계열로 매핑한 geom_smooth 로 추세선 레이어 생성
  geom_smooth(aes(x = 졸업자_계, y = 취업자_합계_계, color = 대계열)) +
```

```
  labs(title = '그룹별 추세선이 포함된 산점도')

p_scatter2
```



실행결과 5-7. 그룹별 추세선이 포함된 산점도

1.2. 러그 산점도(rug scatter plot)

사실 산점도는 아무리 오버플로팅을 제거하려고 노력해도 산점도 자체가 가진 한계로 인해 완전히 제거하기란 매우 힘들다. 그래서 앞서 언급한 바와 같이 샘플수를 조절한다던지 투명도를 조절한다던지 하는 방법이 사용되지만 이로 충분치는 않다. 결국 중첩되어 표현된 부분에 얼마나 많은 데이터가 중첩되는지를 알아내는 방법이 필요한데 러그 산점도가 이에 대한 대안이 될 수 있다.

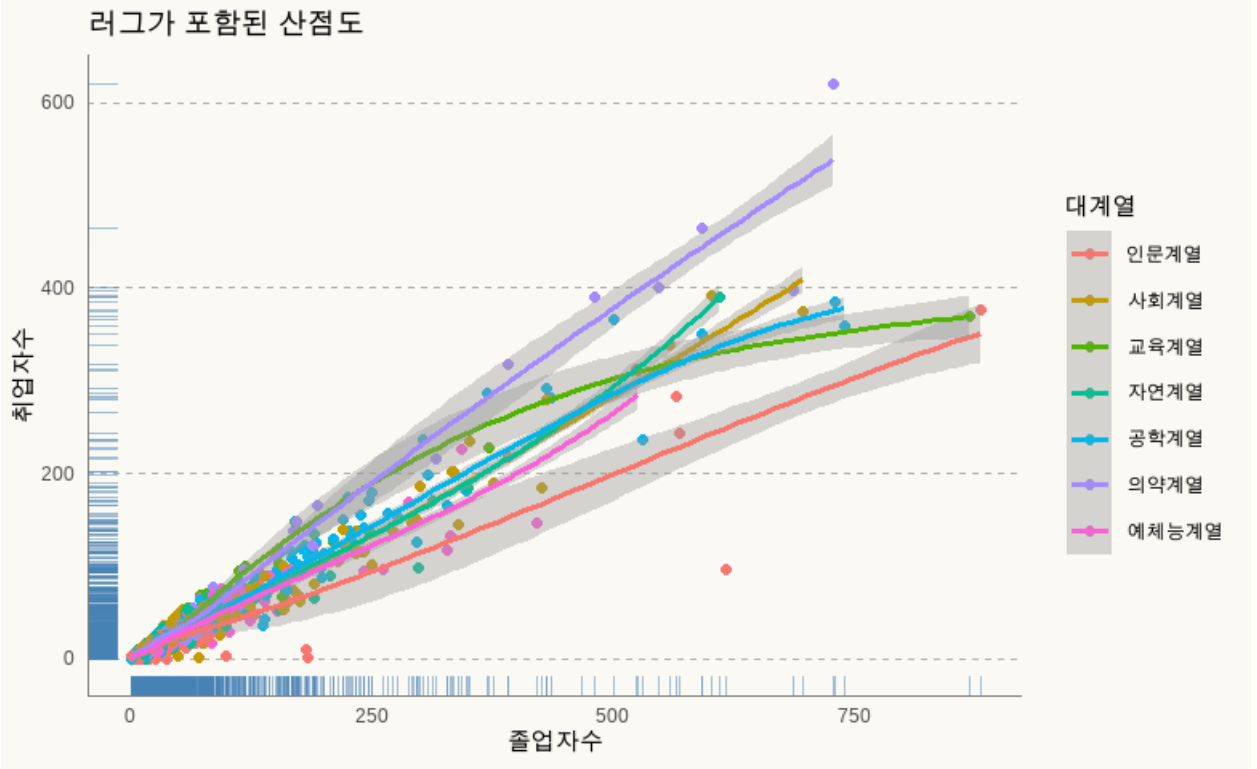
러그(rug)는 '바닥에 까는 발판'으로 X 축과 Y 축의 바닥에 막대를 표현함으로써 데이터의 밀도를 표현하는 방법이다. `ggplot2` 에서 러그를 표현하기 위해서는 `geom_rug()`를 사용할 수 있다.

```
geom_rug(mapping = NULL, data = NULL, stat = "identity", position = "identity",
 ...,  outside = FALSE, sides = "bl", length = unit(0.03, "npc"), na.rm = FALSE,
 show.legend = NA, inherit.aes = TRUE)
```

```
  - mapping : aes()를 사용하여 매핑할 미적요소, 생략되면 ggplot()에 정의된 미적매핑 사용
  - data : 시각화를 위해 사용될 데이터, 생략되면 ggplot()에 정의된 데이터 사용
  - stat : 시각화에 적용될 통계요소, 기본값은 'identity'
  - position : 시각화에 적용될 위치요소, 기본값은 'identity'
  - ... : 미적요소의 설정
  - outside : 러그를 플로팅 지역 밖에 그릴지 설정하는 논리값
  - sides : 러그를 축의 양쪽에 그릴지 설정하는 논리값
  - length : 러그의 길이 설정
  - na.rm : NA 값을 생략할 것인지를 설정하는 논리값
  - show.legend : 범례를 사용할 것인지를 설정하는 논리값
  - inherit.aes : ggplot()에서 설정한 매핑값을 상속받을지 결정하는 논리값
```

앞의 산점도에 러그를 추가한 산점도는 다음과 같다.

```
p_scatter3 <- p_scatter2 +
  ## X 축을 졸업자_계, Y 축을 취업자_합계_계, color 를 대계열로 매핑한 geom_reg 레이처 추가
  geom_rug(aes(x = 졸업자_계, y = 취업자_합계_계), col= "steelblue", alpha=0.5) +
  labs(title = '러그가 포함된 산점도')

p_scatter3
```

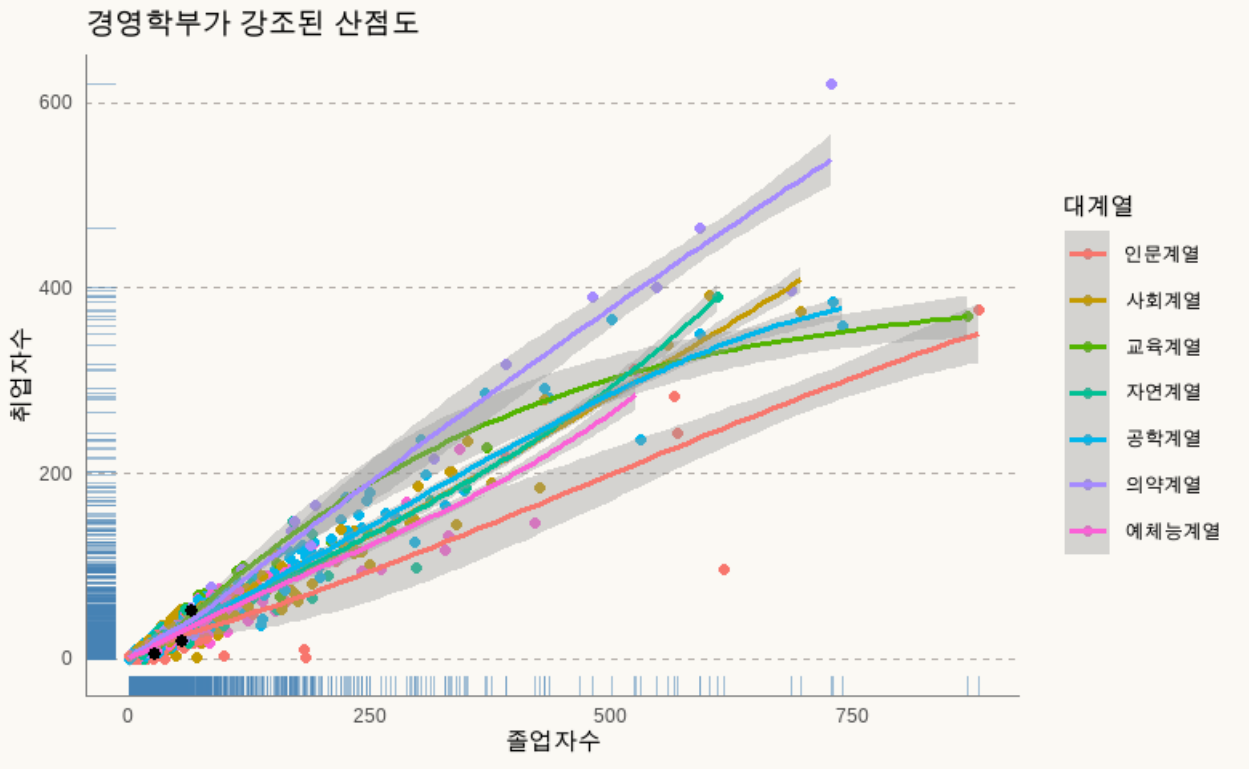


실행결과 5-8. 러그가 포함된 산점도

1.3. 데이터 강조

산점도는 대량의 데이터의 관계성을 찾는데 가장 효과적으로 사용되지만 특정 데이터의 상대적 위치를 표현하는데에도 효과적으로 사용될 수 있는 시각화 방법이다. 전체 데이터의 분포 중 청중의 관심이 있는 데이터의 위치를 표현함으로서 해당 데이터의 현 상태를 한눈에 확인할 수 있다. 다음은 앞서 그렸던 산점도에서 '경영학부'의 상대적 위치를 확인하기 위한 코드이다.

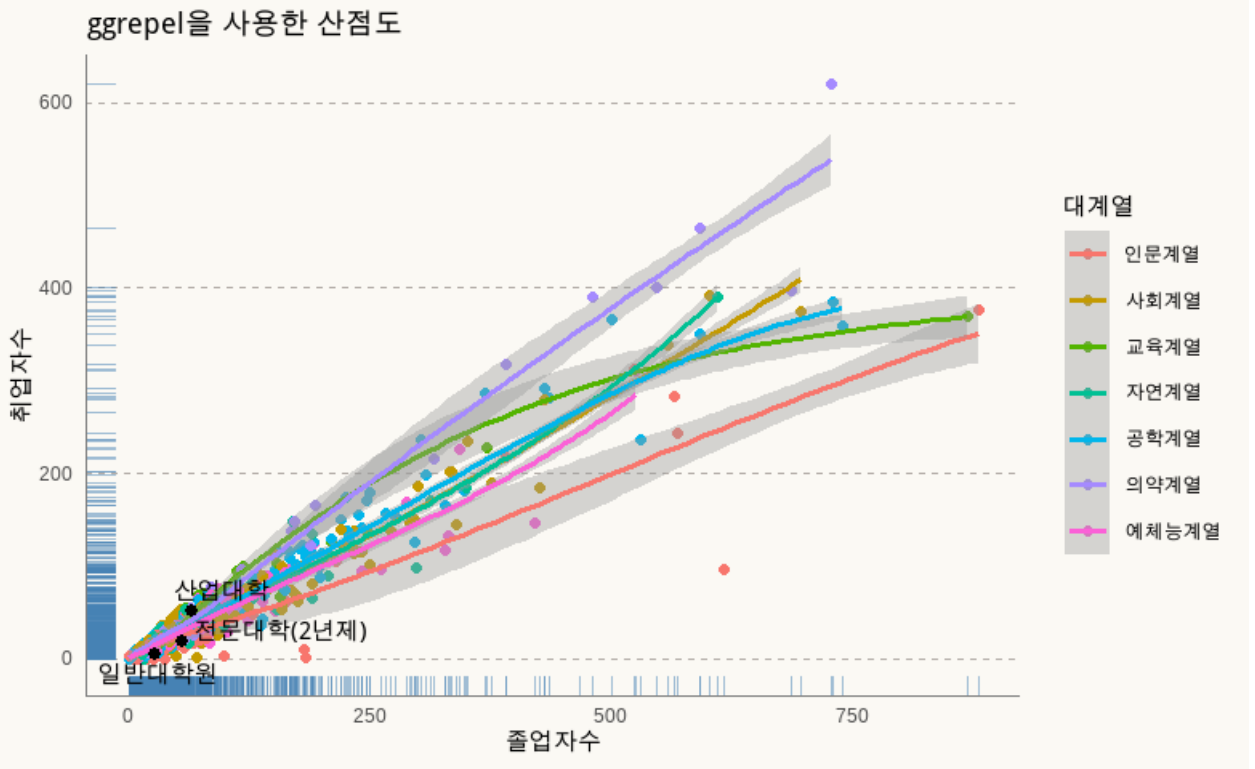
```
p_scatter4 <- p_scatter3 +
  ## '경영학부' 로 필터링된 데이터에서 X 축을 졸업자_계, Y 축을 취업자_합계_계, color 를 대계열로 매핑하고 color 와 size 를 설정한 geom_point 레이어 추가
  geom_point(data = df_취업통계_sample |> filter(학과명 == '경영학부'), aes(x = 졸업자_계, y = 취업자_합계_계), color = 'black', size = 2) +
  labs(title = '경영학부가 강조된 산점도')

p_scatter4
```



실행결과 5-9. 경영학부가 강조된 산점도

경영학부가 강조된 산점도에서 좌측 하단의 검정색 점으로 해당 데이터를 강조하였다. 하지만 이 점만으로는 강조가 좀 부족한 듯 보인다. 이런 경우 시각화에 주석(annotation)을 적절히 사용해주면 더 강조될 수 있다.

```
## ggrepel 패키지 설치
if(!require(ggrepel)) {
  install.packages('ggrepel')
  library(ggrepel)
}

p_scatter5 <- p_scatter4 +
  ## '경영학부' 로 필터링된 데이터에서 X 축을 졸업자_계, Y 축을 취업자_합계_계, label 을 학제로 매핑한 geom_text_repel 레이어 추가
  geom_text_repel(data = df_취업통계_sample |> filter(학과명 == '경영학부'), aes(x = 졸업자_계, y = 취업자_합계_계, label = 학제)) +
  labs(title = 'ggrepel 을 사용한 산점도')

p_scatter5
```



실행결과 5-10. ggrepel 을 사용한 산점도

위의 코드에서 `ggrepel` 패키지를 사용했다. 이 패키지는 `geom_text()`와 `geom_label()`을 확장시킨 함수를 제공하는 패키지로 표현해야 할 문자의 위치를 적절히 잡아주는데 미적요소가 서로 겹쳐서 알아보기 어려울 때 겹침을 피해서 문자를 표시하는 기능을 제공한다. `ggrepel` 패키지에서 주로 사용되는 함수는 `geom_text_repel()`과 `geom_label_repel()` 등이 있다. 사용법은 `geom_text()`와 `geom_label()`과 거의 유사하다.[2]

1.4. 클러스터 스캐터 플롯(Cluster Scatter Plot)

청중에게 특정 데이터 그룹의 분포를 전달하기 위해서 해당 그룹에 대한 범위를 표현해야할 경우가 발생한다. 이 경우에 산점도에 원으로 해당 그룹을 표시해 줄 수 있다. 이런 시각화를 위해서 `ggalt` 패키지의 `geom_encircle()`을 사용할 수 있다. `geom_encircle()`은 `data` 매개변수로 전달된 데이터를 둘러싸는 원을 그려주는 함수이다.

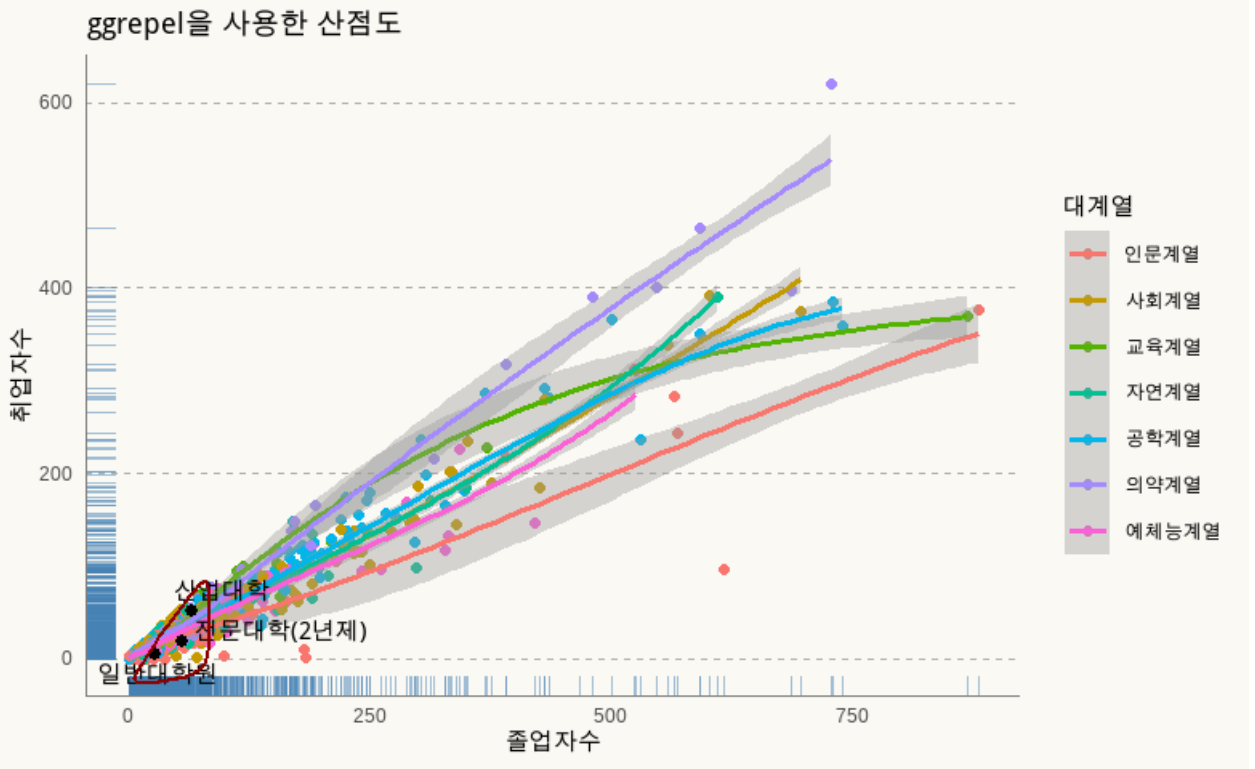
---

[2] ggrepel 의 사용법은 해당 패키지의 vignette 사이트(https://cran.r-project.org/web/packages/ggrepel/vignettes/ggrepel.html)를를){.uri} 참조하라.

```
geom_encircle(mapping = NULL, data = NULL, stat = "identity", position = "identi
ty", na.rm = FALSE, show.legend = NA, inherit.aes = TRUE, ...)
  - mapping : aes()를 사용하여 매핑할 미적요소, 생략되면 ggplot()에 정의된 미적매핑
사용
  - data : 시각화를 위해 사용될 데이터, 생략되면 ggplot()에 정의된 데이터 사용
  - stat : 시각화에 적용될 통계요소, 기본값은 'identity'
  - position : 시각화에 적용될 위치요소, 기본값은 'identity'
  - ... : 미적요소의 설정
  - na.rm : NA 값을 생략할 것인지를 설정하는 논리값
  - show.legend : 범례를 사용할 것인지를 설정하는 논리값
  - inherit.aes : ggplot()에서 설정한 매핑값을 상속받을지 결정하는 논리값
```

다음 코드는 앞에서 강조했던 '경영학부' 데이터를 감싸는 원을 그리는 코드이다.

```r
## ggalt 패키지 설치
if(!require('ggalt')) {
  install.packages('ggalt')
  library(ggalt)
}

p_scatter6 <- p_scatter5 +
  ## geom_encircle()로 '경영학부'를 둘러싸는 원 레이어 추가
  geom_encircle(data = df_취업통계_sample |> filter(학과명 == '경영학부'), aes(x =
 졸업자_계, y = 취업자_합계_계), color="darkred", size = 2)

p_scatter6
```

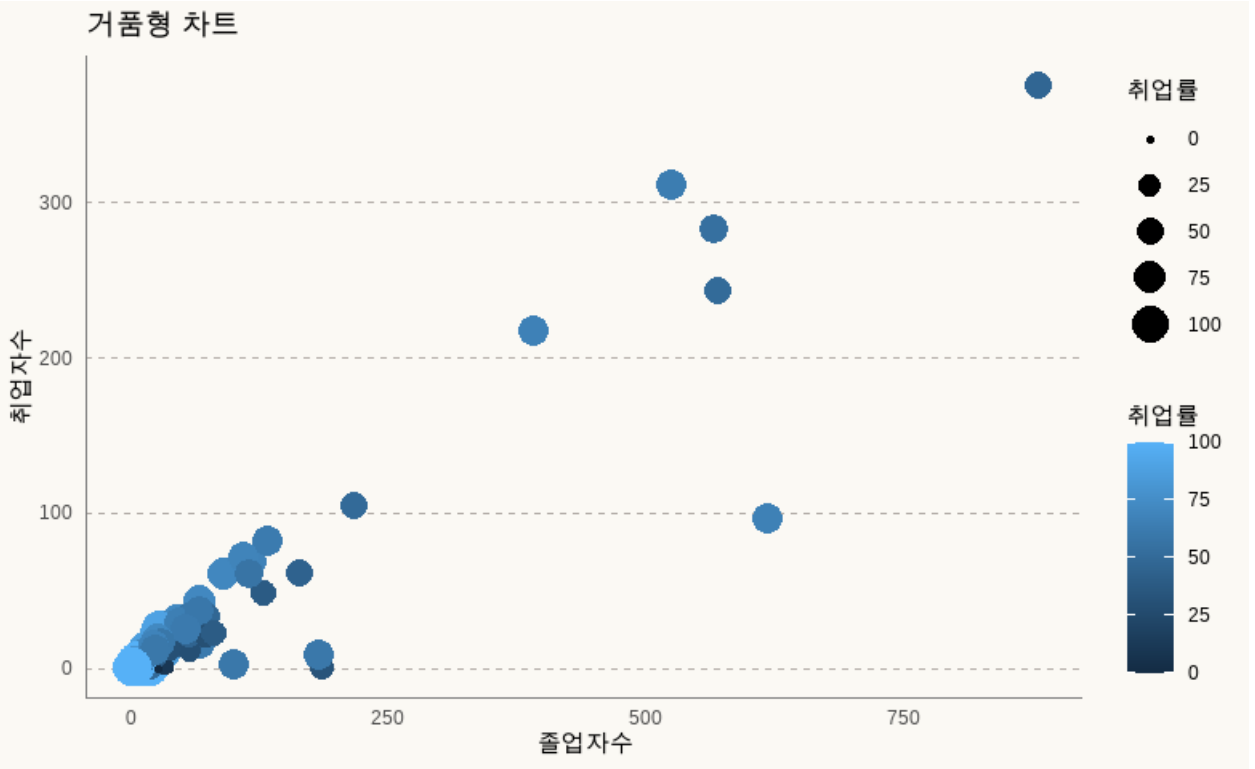


2. 거품형 차트(Bubble Chart)

산점도는 X 축과 Y 축의 2 차원 데이터를 사용하는 시각화 방법이다. 단 두개의 변수만의 관계를 살펴보는 것은 데이터의 관계성을 찾는데 한계가 있어 추가적인 변수들을 표현하는 방법들이 많이 제공되고 있다. 이에 대표적으로 사용되는 시각화 방법이 거품형 차트이다. 거품형 차트는 2018 년 발간된 한스 로슬링의 '팩트풀니스(Factfullness)'에서 주로 사용되며 유명세를 탄 사각화 방법이다. X 축과 Y 축의 변수에 기초하여 표현되는 산점도 데이터 포인트의 크기를 매핑한 추가적 변수를 사용함으로써 3 차원 산점도를 표현하는 방법이다. 거품형 차트는 특별한 함수가 제공되는 것이 아니고 `geom_point()`에 `size` 미적요소를 매핑하여 크기를 조절하여 생성한다. 다음은 졸업자수와 취업자수의 산점도에 취업률을 추가로 매핑한 거품 차트를 생성하는 코드이다.

```
p_bubble <- df_취업통계_sample |> filter(대계열 == '인문계열') |>
  ggplot()

p_bubble +
  ##  X 축을 졸업자_계, Y 축을 취업자_합계_계, size 를 취업률_계, color 를 취업률_계로 매핑한 geom_point 레이어 생성
  geom_point(aes(x = 졸업자_계, y = 취업자_합계_계, size = 취업률_계, color = 취업률
```

```
_계)) +
  labs(title = '거품형 차트', x = '졸업자수', y = '취업자수', color = '취업률', siz
e = '취업률')
```



실행결과 5-11. 거품형 차트

3. 상관도

앞에서 살펴본 산점도들은 데이터의 전반적인 분포에 따라 상관 정도를 육안으로 확인하는데 사용된다. 하지만 사용자가 미리 선택한 변수들 간의 상관관계를 확인할 수 있고 그 상관관계가 어느 정도 강한지에 대해서는 알기가 어렵다. 이를 위해 사용하는 것이 상관도이다. 상관도는 여러개의 변수들을 대상으로 각각의 변수들끼리 상관관계가 어느 정도인지를 수치로 표현함으로서 변수들간의 상관강도를 확인할 수 있다.

상관도를 생성하는 함수는 여러개가 있다. 여기서는 `ggcorrplot` 패키지에서 제공하는 `ggcorrplot()`을 사용하여 상관도를 생성하도록 하겠다.

```
ggcorrplot(corr, method = c("square", "circle"), type = c("full", "lower", "uppe
r"), ggtheme = ggplot2::theme_minimal, title = "", show.legend = TRUE, legend.ti
tle = "Corr", show.diag = FALSE, colors = c("blue", "white", "red"), outline.col
or = "gray", hc.order = FALSE, hc.method = "complete", lab = FALSE,   lab_col =
```

```
"black", lab_size = 4, p.mat = NULL, sig.level = 0.05, insig = c("pch", "blank
"), pch = 4, pch.col = "black", pch.cex = 5, tl.cex = 12, tl.col = "black", tl.s
rt = 45, digits = 2)
  - corr : 시각화할 상관 행렬
  - method : 상관관계 표현에 사용될 도형의 형태
  - type : 상관도의 표현 방법, full 은 전체 매트릭스, lower 는 아래쪽 대각선 부분만, u
pper 는 위쪽 대각선 부분만 상관도 표현
  - ggtheme : 상관도 표현에 사용할 테마 선택
  - title : 상관도의 제목 설정
  - show.legend : 범례를 사용할 것인지를 설정하는 논리값
  - legend.title : 상관도의 범례 제목 설정
  - show.diag : 상관 계수를 주대각선에 표시할지 여부를 설정하는 논리값
  - colors : 상관도에 사용할 색 설정
  - outline.color : 상관도의 외부 선 색 설정
  - hc.order, hc.method : hclust 함수를 사용한 순서 및 응집 결정 방법 설정
  - lab, lab_col, lab_size : 상관 계수 라벨의 크기와 색상
```

`ggcorrplot()`은 지금까지 사용했던 `ggplot2`의 함수들과 다른 것이 다른 `ggplot2` 함수들은 `ggplot` 객체를 첫번째 매개변수로 사용했지만 `ggcorrplot()`은 `ggplot` 객체가 아닌 상관 행렬을 첫번째 매개변수로 사용한다. 그렇다면 상관 행렬을 만들어야 하겠다.

상관행렬은 상관도를 생성하기 위해 필요한 변수들이 각각 가로, 세로로 설정되어 가로, 세로에 해당하는 변수들의 상관계수로 표현된 행렬을 말한다. 같은 변수들이 가로, 세로로 표현되기 때문에 대각선에는 자기 자신과의 상관관계이기 때문에 상관계수가 1 이 되고 이 대각선을 사이에 두고 양쪽으로 구분된 지역들은 서로 같은 상관계수를 가지게 된다. 이 상관행렬을 생성하기 위해서는 `cor()`를 사용한다.

```
cor(x, use = "everything", method = c("pearson", "kendall", "spearman")
  - x : 수치형 벡터
  - use : 결측치 발생시 처리 방법 설정
  - method : 상관계수를 계산하는데 사용할 방법 설정
```

앞서 생성했던 df_취업통계_sample 을 사용하여 상관도를 생성하는 코드는 다음과 같다.

```r
## ggcorrplot 패키지 설치
if(!require('ggcorrplot')) {
  install.packages('ggcorrplot')
  library(ggcorrplot)
}
```

```r
## cor()를 사용하여 상관행렬 생성
corr <- round(cor(df_취업통계_sample[, 13:19]), 1)

## 상관도 생성
ggcorrplot(corr,
           type = "lower",  ## 하부 대각선만 표시
           lab = TRUE,   ## 상관계수 표시
           lab_size = 3, ## 상관계수 라벨 크기
           method="square", ## 상관도 표시방법
           colors = c("tomato1", "white", "skyblue1"),  ## 상관도 표시 색 설정
           title="취업 상세 정보의 상관도", ## 상관도 제목 설정
           ggtheme=theme_bw)
```

실행결과 5-12. 취업 상세 정보의 상관도

4. 히트맵

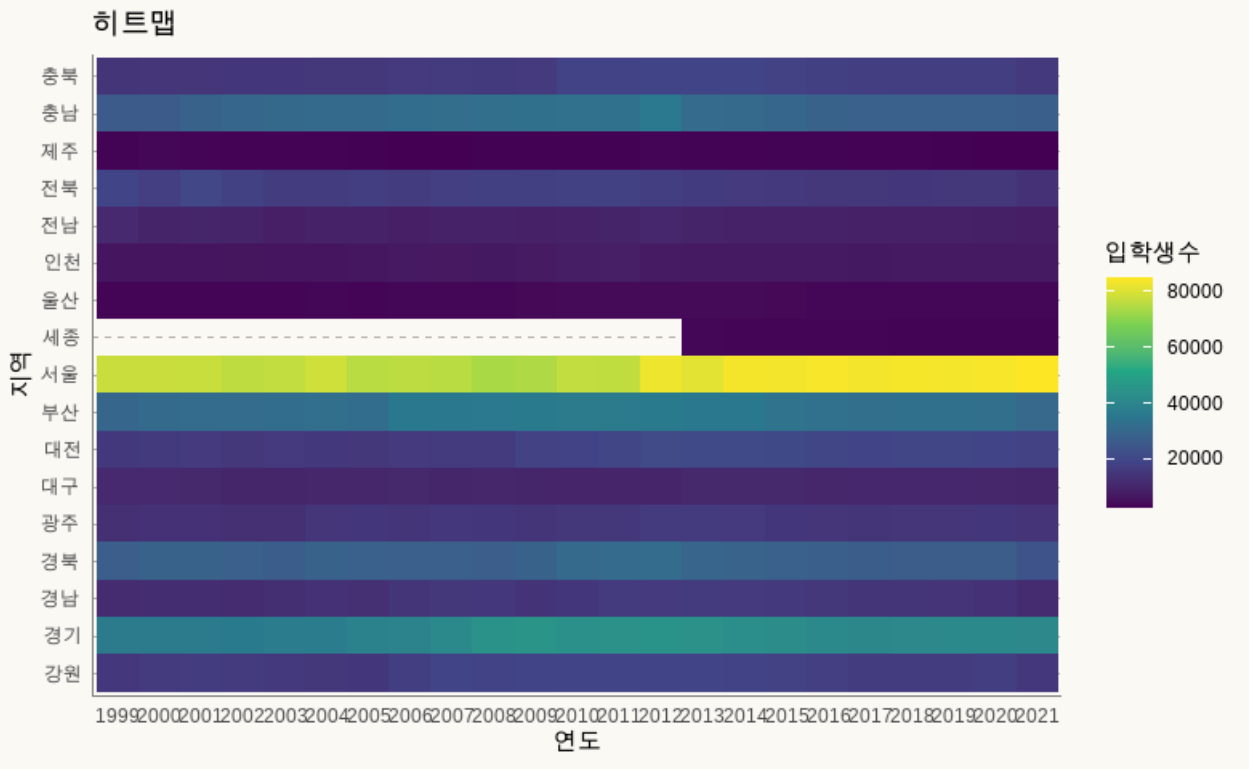
히트 맵(heat map)은 X 축과 Y 축의 일정한 간격으로 나누고 해당 셀안에 위치하는 데이터의 개수에 따른 온도(Heat)를 표현하는 시각화 방법이다. 일반적으로 X 축과 Y 축에 골고루

분포한 데이터의 밀도를 시각화하는 방법으로 `ggplot2` 의 `geom_tile()`을 사용하여 생성할 수 있다.

```
geom_tile(mapping = NULL, data = NULL, stat = "identity", position = "identity",
 ..., linejoin = "mitre", na.rm = FALSE, show.legend = NA, inherit.aes = TRUE)
  - mapping : aes()를 사용하여 매핑할 미적요소, 생략되면 ggplot()에 정의된 미적매핑 사용
  - data : 시각화를 위해 사용될 데이터, 생략되면 ggplot()에 정의된 데이터 사용
  - stat : 시각화에 적용될 통계요소, 기본값은 'identity'
  - position : 시각화에 적용될 위치요소, 기본값은 'identity'
  - ... : 미적요소의 설정
  - linejoin : 선 겹침 스타일 설정
  - na.rm : NA 값을 생략할 것인지를 설정하는 논리값
  - show.legend : 범례를 사용할 것인지를 설정하는 논리값
  - inherit.aes : ggplot()에서 설정한 매핑값을 상속받을지 결정하는 논리값
```

다음은 3 장에서 생성한 긴 형태의 입학자 데이터를 히트맵으로 표한한 코드이다.

```
## df_입학자_long 의 데이터를 필터링하고 ggplot 객체로 생성
df_입학자_long |> filter(지역 != '전체') |> filter(학교종류 == '일반대학') |>
  ggplot() +
  ### geom_tile()로  X 축을 연도, Y 축을 지역, fill 을 입학생수로 매핑한 히트맵 생성
  geom_tile(aes(x = 연도, y = 지역, fill = 입학생수)) +
  labs(title = '히트맵') +
  ## 색 타입을 설정
  scale_fill_continuous(type = "viridis")
```


히트맵
충북
충남
제주
전북
전남
인천
울산
세종
서울
부산
대전
대구
광주
경북
경남
경기
강원
지역
1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021
연도
입학생수
80000
60000
40000
20000

실행결과 5-13. 히트맵