서울시립대학교
UNIVERSITY OF SEOUL

# 3주차:
# 데이터분석 결과의 시각화

서울시립대학교 통계데이터사이언스학과
전종준

# Contents

# 01. 분산분석의 시각화

# 분산분석과 평균의 비교

**분산분석**

- 개념:

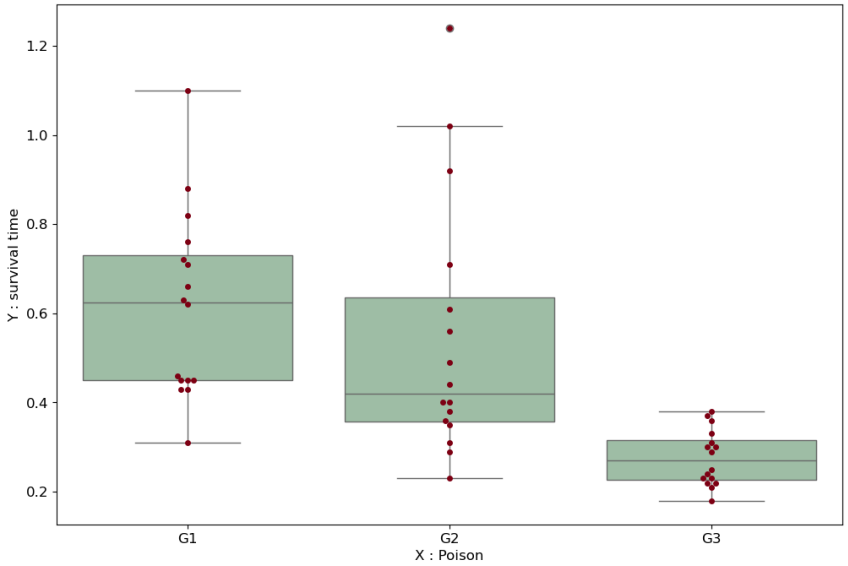
- 통계량:

# 분산분석과 평균의 비교

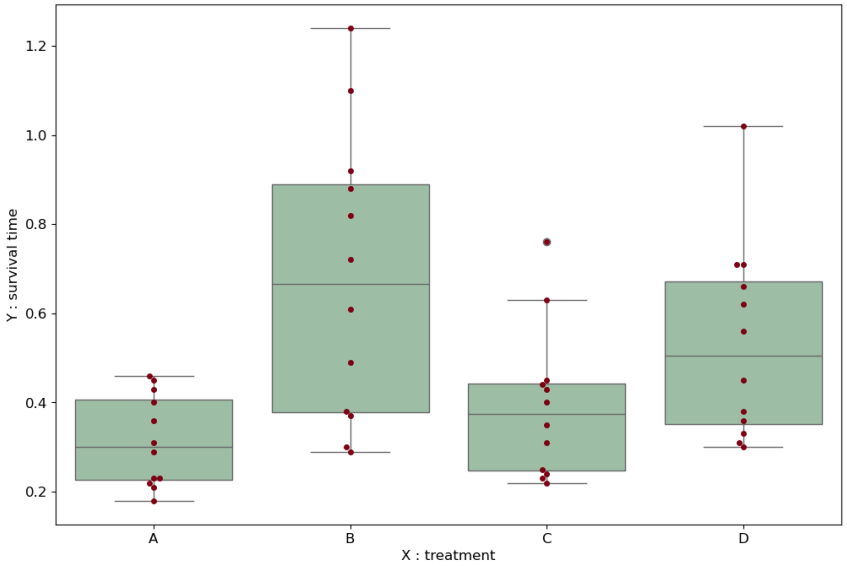
## Dataset

- Poison dataset
  - R 패키지: MASS (Modern Applied Statistics with S)
  - 실험 배경: 쥐에게 특정 독소를 투여한 후 생존 시간 측정
  - 목표: 다양한 독소 처리 방식(treatment) 및 독소유형에 따른 생존 시간 차이 분석
- 변수설명
  - time:  생존 시간
  - poison: 독소 유형 (Factor, 예: G1,G2, G3)
  - treat:  처리 방식 (Treatment group, 예: A, B, C, D)

# 분산분석과 평균의 비교

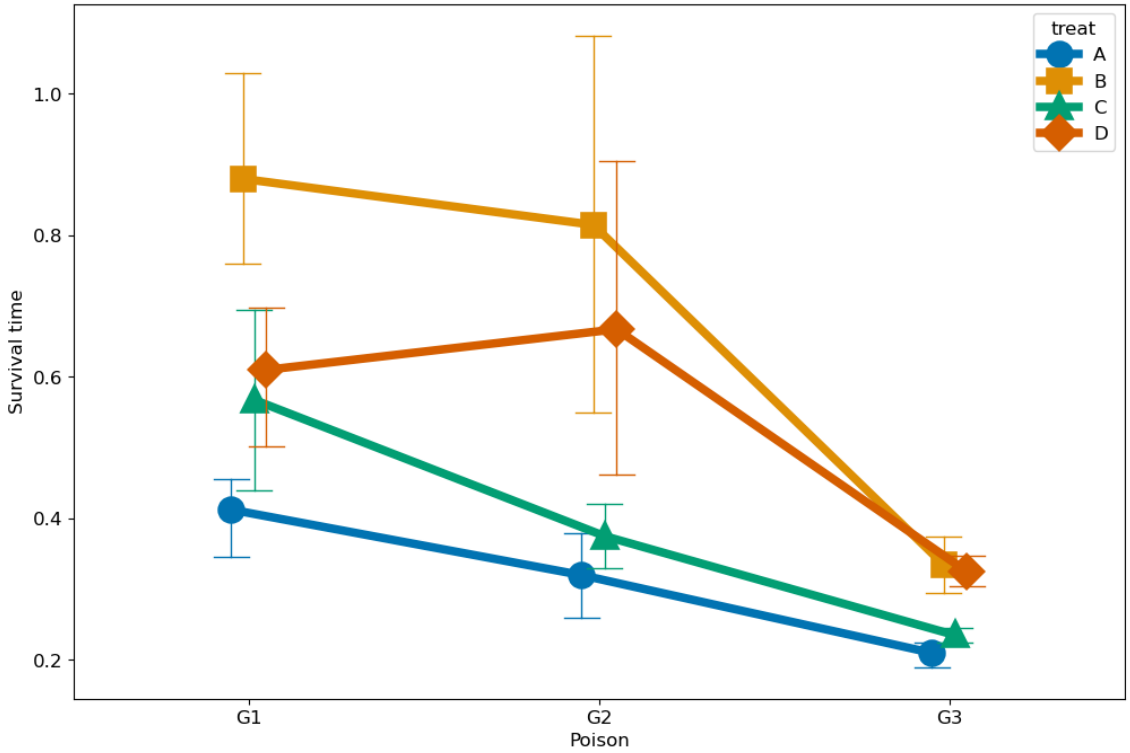



독소 유형에 따른 생존시간의 차이




노출 방법에 따른 생존시간의 차이

# 분산분석과 평균의 비교




독성과 노출 방법에 따른 생존시간의 차이

# 분산분석과 평균의 비교

**평균차이의 분석**

- 독소 유형에 따라 생존시간에 차이가 있는가?
- 노출 형태에 따라 생존시간의 차이가 있는가?
- 특정한 독소유형과 노출형태의 조합이 생존시간에 영향을 주는가?

두 그룹의 평균 차이의 분석: 독립 이표본 t검정 (independent two-sample t-test), 쌍 표본 t 검정(paired t-test)

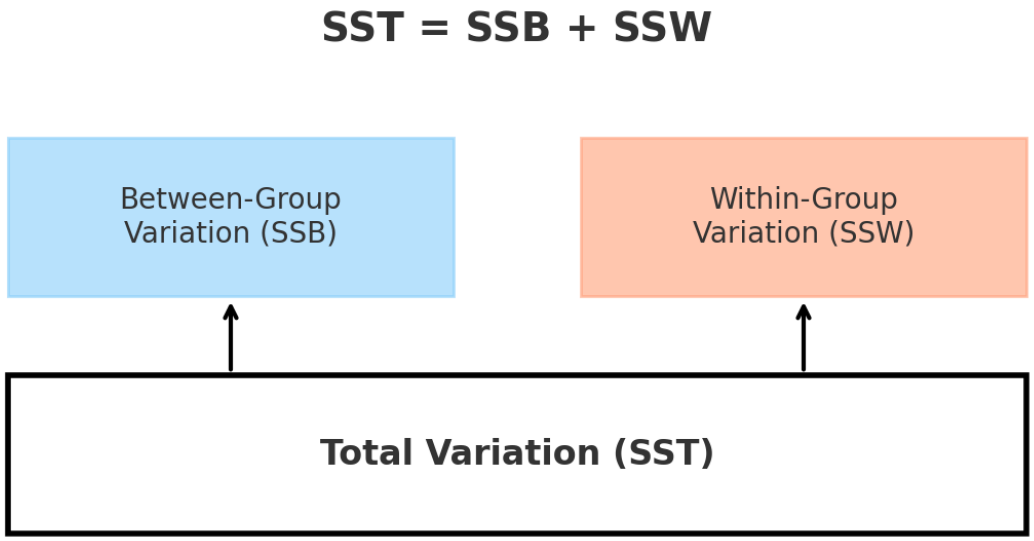
-> 여러 그룹의 평균검정 (귀무가설: 모든 집단의 평균이 같다)

# 분산분석과 평균의 비교

**분산분석의 idea**

- $Y_{ij}$: 집단의 번째 관측값, $\bar{Y}_j$: 번 집단의 평균, $\bar{Y}$: 전체평균
- SST: 자료들 전체의 변동성 ($Y_{ij}$ 가 $\bar{Y}$를 기준으로 얼마나 변이가 큰가?)
- SSB: 집단간 평균의 변동성 ($\bar{Y}_j$가 $\bar{Y}$를 기준으로 얼마나 변이가 큰가? 집단의 크기를 반영함)
- SSW: 집단내 평균의 변동성 (각 $Y_{ij}$ 가 $\bar{Y}_j$를 기준으로 얼마나 변이가 큰가? 집단별로 변이 총합을 계산)
- **SST = SSB + SSW 가 성립하기 때문에 집단간 평균이 다르다면 SSB/SSW 크기가 커질 것이다.**
  **(cf) 사용하는 통계량은 SSB와 SSW를 각 자유도로 나눈 MSB와 MSW 를 사용**

# 분산분석과 평균의 비교



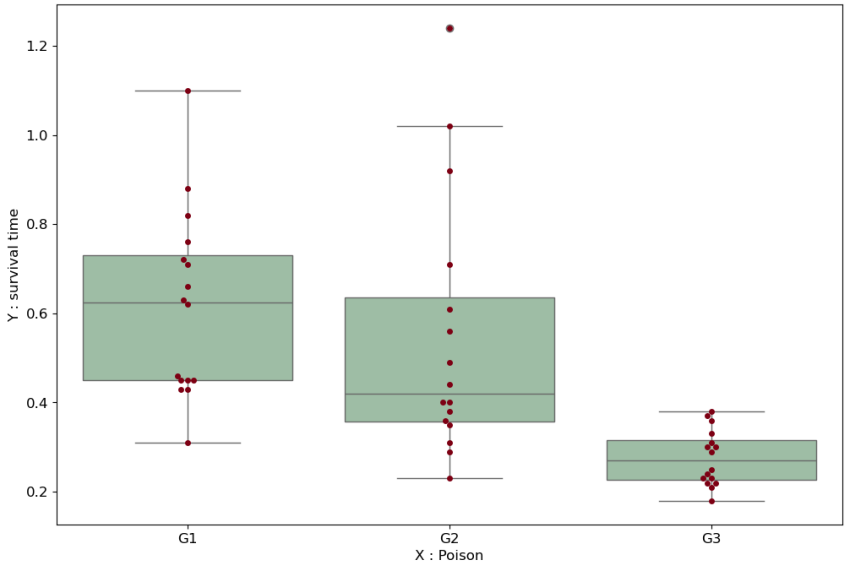

$$SST = \sum_{j=1}^{k} \sum_{i=1}^{n_j} (Y_{ij} - \bar{Y})^2$$

$$SSB = \sum_{j=1}^{k} n_j (\bar{Y}_j - \bar{Y})^2$$

$$SSW = \sum_{j=1}^{k} \sum_{i=1}^{n_j} (Y_{ij} - \bar{Y}_j)^2$$

# 분산분석과 평균의 비교

```
one_way_anova = ols('time ~ C(poison)', df_poi).fit()
result_one = anova_lm(one_way_anova)
result_one
```




| | df | sum_sq | mean_sq | F | PR(>F) |
|---|---|---|---|---|---|
| C(poison) | 2.0000 | 1.0330 | 0.5165 | 11.7860 | 0.0001 |
| Residual | 45.0000 | 1.9721 | 0.0438 | NaN | NaN |

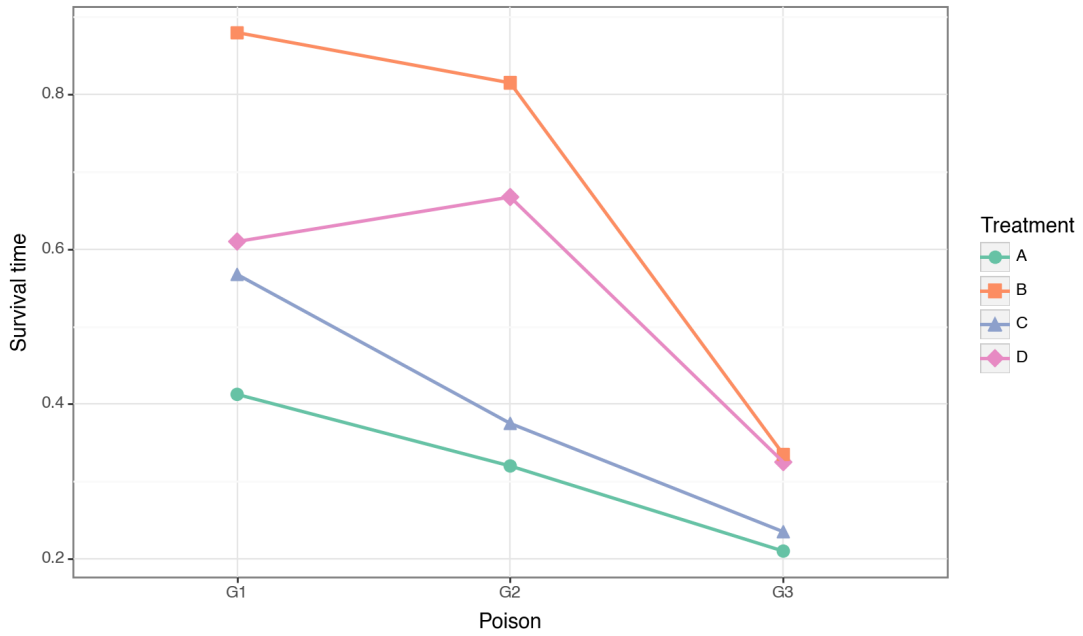
# 분산분석과 평균의 비교

```
two_way_anova = ols('time ~ C(poison) * C(treat)', df_poi).fit()

result_two = anova_lm(two_way_anova)

result_two
```

| | df | sum_sq | mean_sq | F | PR(>F) |
|---|---|---|---|---|---|
| C(poison) | 2.0000 | 1.0330 | 0.5165 | 23.2217 | 0.0000 |
| C(treat) | 3.0000 | 0.9212 | 0.3071 | 13.8056 | 0.0000 |
| C(poison):C(treat) | 6.0000 | 0.2501 | 0.0417 | 1.8743 | 0.1123 |
| Residual | 36.0000 | 0.8007 | 0.0222 | NaN | NaN |

```
from plotnine import *
import pandas as pd

plot = (
ggplot(df_poi, aes(x='poison', y='time', color='treat', shape='treat', group='treat'))
+ stat_summary(fun_y=np.mean, geom='point', size=3)
+ stat_summary(fun_y=np.mean, geom='line', size=1)
+ scale_shape_manual(values=['o', 's', '^', 'D
+ scale_color_brewer(type='qual', palette='Set2') +
labs(x='Poison', y='Survival time', color='Treatment', shape='Treatment')
+ theme_bw()
+ theme(figure_size=(8, 5))
)
print(plot)
```

# 분산분석과 평균의 비교



# 02. GGPLOT2 문법

# ggplot2

## ggplot 문법

- 파이썬에서는 plotline 패키지를 이용해서 ggplot 문법을 그대로 사용할 수 있음
  - from plotline import *  # 시각화 함수를 모두 이름공간에 등록
- 기본 문법
  - ggplot(data=None, mapping=None)
    data 는 데이터프레임을 사용, mapping 은 시작속성을 대응시킴
  - ggplot(data=boston_df, mapping=aes(x='RM', y='Price ' )) 에서 aes(x='RM', y='Price') → RM을 x축, Price를 y축으로 시각화

# ggplot2

## ggplot 문법

- aes 속성 (속성에 data d에 있는 변수를 지정하거나 고유 속성값을 지정할 수 있음)
  - x, y: x축, y축 변수 지정
  - color: 선 색상 (범례에 반영됨)
  - fill: 면 색상 (히스토그램, 박스플롯 등)
  - shape: 점의 모양
  - size: 점, 선 등의 크기
  - alpha: 투명도 (0~1)
  - linetype: 선 스타일 (점선, 실선 등)
  - group: 선 연결 그룹 지정 (특히 geom_line() 등에서 중요)

# ggplot2

## ggplot 문법

- aes 속성 예시
  - ggplot(boston_df, aes(x='Price'))
  - ggplot(df_poi, aes(x='poison', y='time', color='treat', shape='treat', group='treat'))

# ggplot2

**ggplot 문법**

- \+ 연산자를 통해서 시각화 요소들을 전달하고 시각화 함수를 구동함
  - ggplot() + 시각화함수 + 속성 조정함수 + ....
- geon_histogram()
  - geom_histogram(mapping=None, data=None, bins=30, binwidth=None, fill=None, color=None, alpha=None, ...)
  - 히스토그램 함수, y 축의 높이를 변경할 싶은 경우에는 mapping 을 ase(y = '..density..') 로 설정해 줌

# ggplot2

**ggplot 문법**

- geon_point()
  - geom_point(size=3, alpha=0.8)
  - size, color 에 변수값을 대응시키면 팔레트에 따라 색상 조정가능
- facet_wrap('~변수이름')
  - 그래픽 함수와 함께 사용해서 여러 개의 subplot 을 그릴 수 있음

# 회귀분석

## Boston Housing dataset

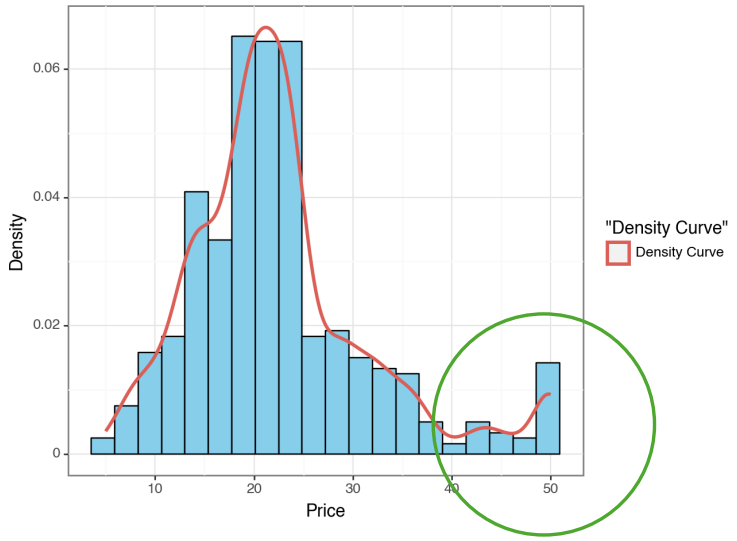
- 미국 보스턴(Boston) 지역의 주택 가격(단위: $1,000)을 그 지역의 다양한 사회적, 경제적, 환경적 변수를 바탕으로 예측
- 변수
  - Price : 주택 가격
  - RM : 주택 1가구당 평균 방의 개수
  - NOX : 10ppm 당 농축 일산화질소
  - AGE : 1940년 이전에 건축된 소유주택의 비율
  - LSTAT : 소득 하위계층의 비율(%)

# 03. 회귀분석의 시각화

# 회귀분석

## Boston Housing dataset (단위면적당 가격의 분포)

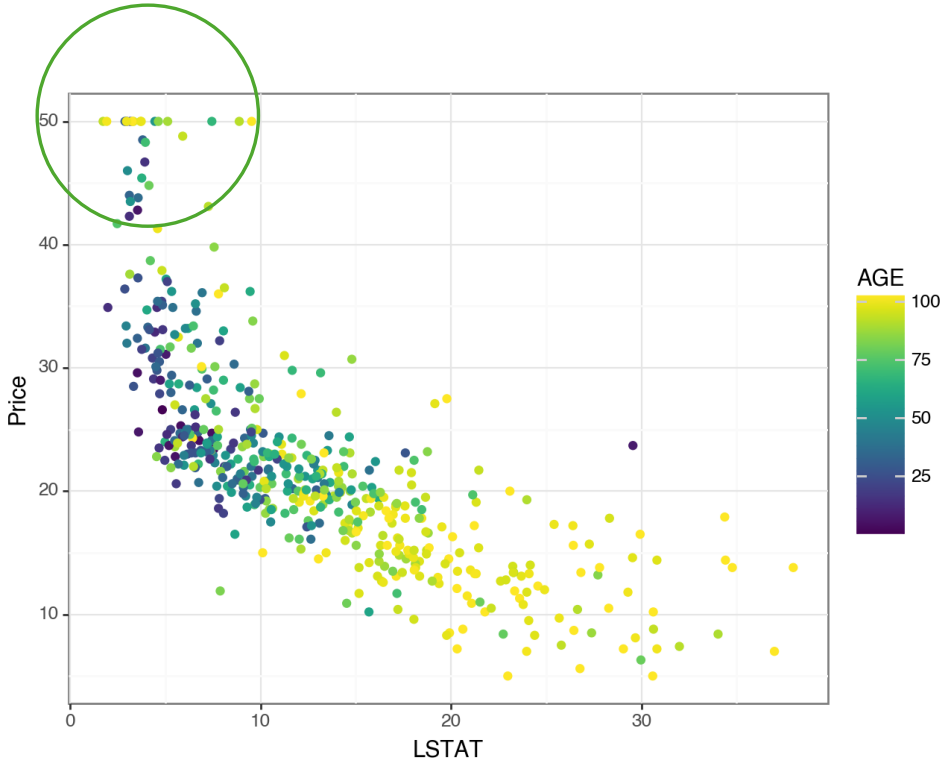
```
plot = (
    ggplot(boston_df, aes(x='Price'))
        + geom_histogram(aes(y='..density..'),
                bins=20, fill='skyblue', color='black')
        + geom_density(aes(color='"Density Curve"'), size=1.2)
        + labs(x='Price', y='Density')
        + theme_bw()
    )
print(plot)
```



# 회귀분석

## Boston Housing dataset

```
plot = (
    ggplot(boston_df, aes(x='LSTAT', y='Price',
                color='AGE'))
    + geom_point()
    + labs(x='LSTAT', y='Price')
    + theme_bw()
    )
print(plot)
```



# 회귀분석

## Boston Housing dataset(다중플롯 그리기)

- Wide Format: 하나의 행(row)에 여러 변수의 값이 열(column) 로 펼쳐져 있는 형태

| ID | RM | NOX | AGE | LSTAT |
|---|---|---|---|---|
| 1 | 6.5 | 0.45 | 66.2 | 12.4 |
| 2 | 7.2 | 0.38 | 45.1 | 8.2 |
| 3 | 5.9 | 0.55 | 78.3 | 15.9 |

- Long Format: 여러 변수들의 값을 행(row) 으로 나열하고,

| ID | Variable | Value |
|---|---|---|
| 1 | RM | 6.5 |
| 1 | NOX | 0.45 |
| 1 | AGE | 66.2 |
| 1 | LSTAT | 12.4 |

# 회귀분석

## Boston Housing dataset(다중플롯 그리기)

- 포맷의 변환 (pandas 활용)
  - Wide format -> Long Format

    df_long = boston_df[['RM', 'NOX', 'AGE', 'LSTAT']].**melt**(var_name='Variable',
    value_name='Value')

    df_long = boston_df[['RM', 'NOX', 'AGE', 'LSTAT']].melt(**id_vars ='Price'**,
    var_name='Variable', value_name='Value')

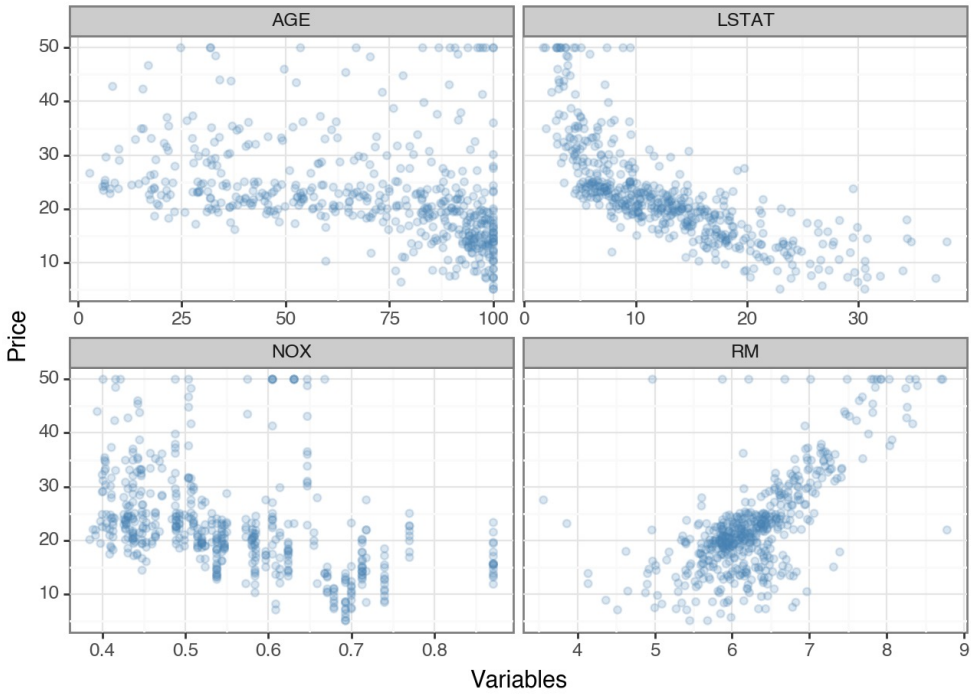
  - Long Format -> Wide format

    df_wide = df_long.pivot(columns='Variable', values='Value')

# 회귀분석

## Boston Housing dataset(다중플롯 그리기)

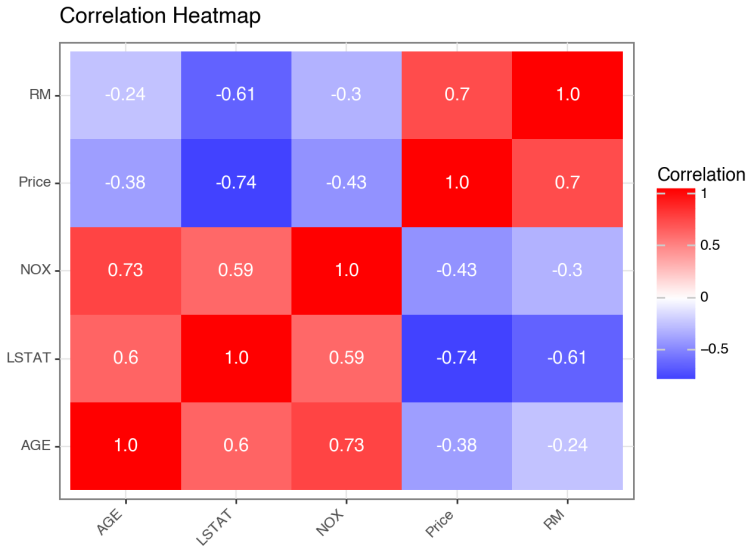
```
df_long = boston_df.melt(id_vars='Price',
          var_name='Variable', value_name='Value')

plot = (
     ggplot(df_long, aes(x='Value', y='Price'))
          + geom_point(alpha=0.2, color='steelblue')
          + facet_wrap('~Variable', scales='free_x')
          + labs(x='Variables', y='Price')
          + theme_bw()
          )
print(plot)
```



# 회귀분석

## Boston Housing dataset(다중플롯 그리기)

```
corr_mat = boston_df.corr()
corr_long = corr_mat.reset_index().melt(id_vars='index')
corr_long.columns = ['Var1', 'Var2', 'Correlation']
plot = (
     ggplot(corr_long, aes(x='Var1', y='Var2', fill='Correlation'))
          + geom_tile()
          + geom_text(aes(label='round(Correlation, 2)'), color='white')
          + scale_fill_gradient2(low='blue', high='red', mid='white',
          midpoint=0)
          + theme_bw()
          + labs(title='Correlation Heatmap', x='', y='')
          + theme(axis_text_x=element_text(rotation=45, ha='right'))
          )
print(plot)
```



# 회귀분석

## Boston Housing dataset(다중플롯 그리기)

```
model = sm.OLS.from_formula("Price ~
        RM+NOX+AGE+LSTAT", data=boston_df)
result = model.fit()
print(result.summary())
```

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | Price | R-squared: | 0.641 |
| Model: | OLS | Adj. R-squared: | 0.638 |
| Method: | Least Squares | F-statistic: | 223.2 |
| Date: | Sun, 23 Mar 2025 | Prob (F-statistic): | 8.13e-110 |
| Time: | 19:59:43 | Log-Likelihood: | -1581.4 |
| No. Observations: | 506 | AIC: | 3173. |
| Df Residuals: | 501 | BIC: | 3194. |
| Df Model: | 4 | | |
| Covariance Type: | nonrobust | | |

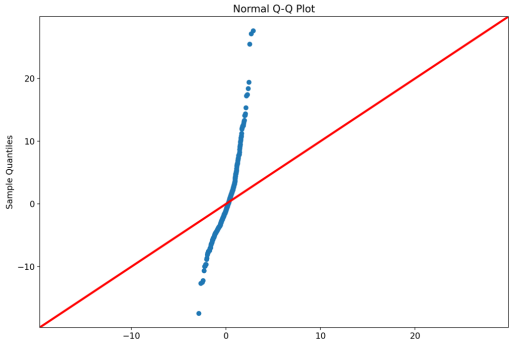
| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| Intercept | 0.5415 | 3.398 | 0.159 | 0.873 | -6.134 | 7.217 |
| RM | 4.9989 | 0.454 | 11.009 | 0.000 | 4.107 | 5.891 |
| NOX | -4.6426 | 3.246 | -1.430 | 0.153 | -11.019 | 1.734 |
| AGE | 0.0205 | 0.014 | 1.489 | 0.137 | -0.007 | 0.047 |
| LSTAT | -0.6522 | 0.055 | -11.755 | 0.000 | -0.761 | -0.543 |

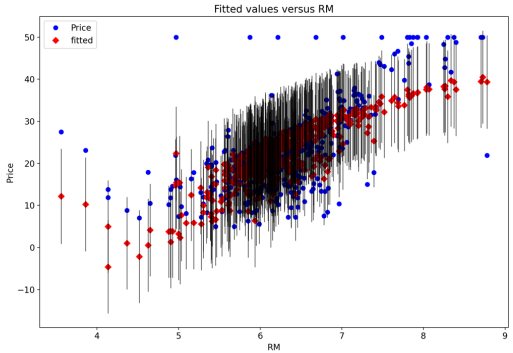
| | | | |
|---|---|---|---|
| Omnibus: | 144.315 | Durbin-Watson: | 0.848 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 438.867 |
| Skew: | 1.343 | Prob(JB): | 5.03e-96 |
| Kurtosis: | 6.688 | Cond. No. | 1.19e+03 |

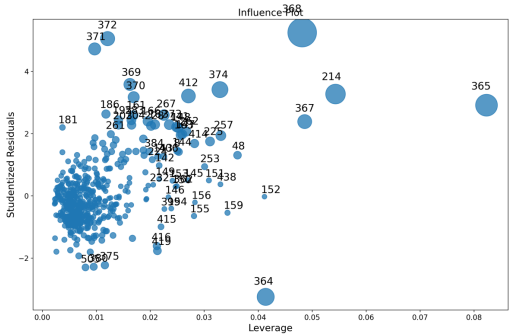
# 회귀분석

## 잔차분석의 시각화

```
#잔차분석의 시각화
sm.qqplot(result.resid, line='45')
plt.title("Normal Q-Q Plot")
plt.show()

sm.graphics.plot_fit(result, 'RM')
plt.show()

sm.graphics.influence_plot(result)
plt.tight_layout()
plt.show()
```



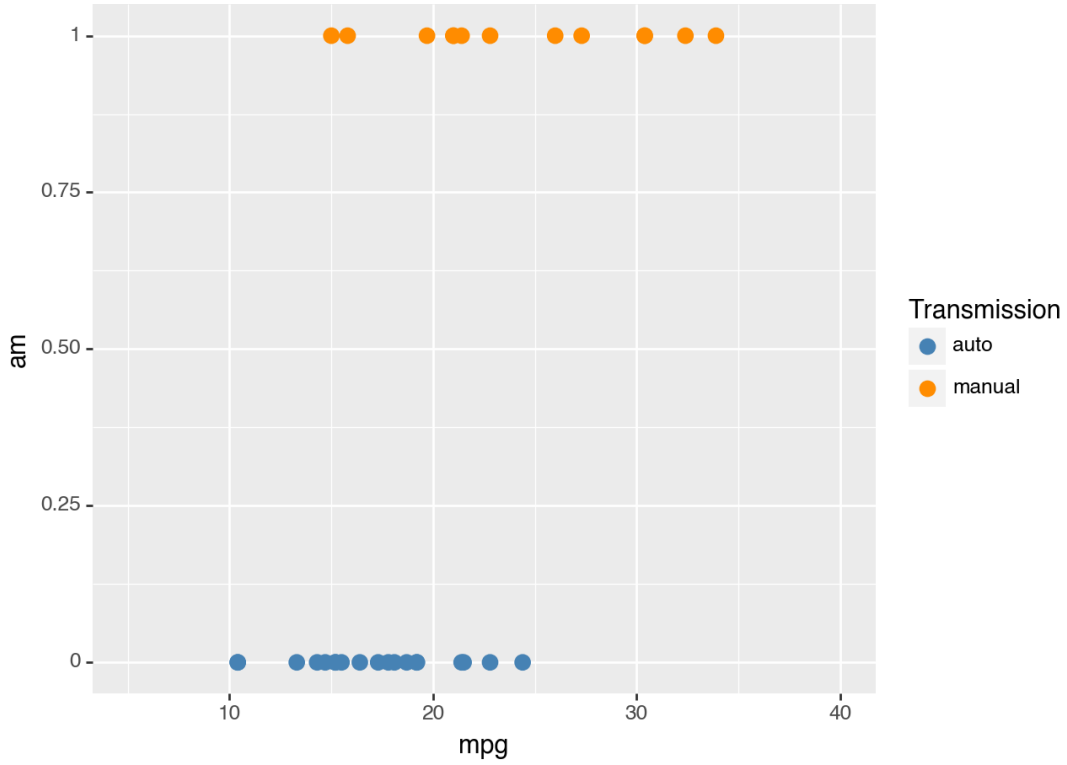



# 04. 로지스틱 회귀분석의 시각화

# 로지스틱 회귀분석

**이진분류분석**

- 연비(mpg) 와 변속기 형태 (am)의 관계 분석
  - am: 0은 자동변속기, 1 은 수동변속기를 나타냄



# 로지스틱 회귀분석

### 시각화 코드

```
url = "http://ranking.uos.ac.kr/class/VIS/data/mtcars.csv"
df = pd.read_csv(url)
df = df[['mpg','am']]
df['am'] = df['am'].astype('int')
# mpg: 연비 (1갤런의 연료로 몇 마일을 가는가?)
# am: 변속기 형태 (0: 자동변속기, 1: 수동변속기)
plot = (ggplot(data=df, mapping=aes(x='mpg', y='am', color='factor(am)'))
        + geom_point(size=3)
        + scale_color_discrete(labels=['auto', 'manual'])
        + labs(color="Transmission")
        + ylim(0,1)
        + xlim(5,40)
        )
plot
```

# 로지스틱회귀분석

**선형회귀와 로지스틱 회귀분석의 비교 (일변량)**

- 회귀모형

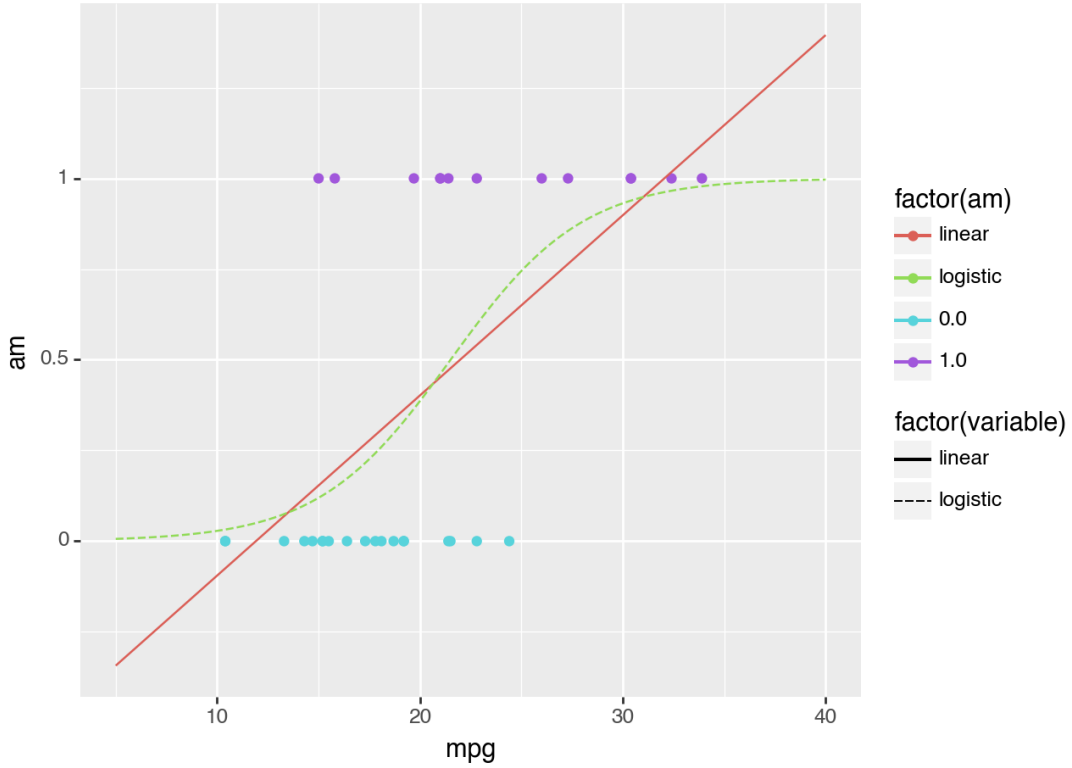
$$\hat{y} = f(x)$$

- 선형회귀모형

$$f(x) = \beta_0 + \beta_1 x$$

- 로지스틱회귀모형

$$f(x) = \frac{\exp(\beta_0 + \beta_1 x)}{1 + \exp(\beta_0 + \beta_1 x)}$$



# 로지스틱회귀분석

**선형회귀와 로지스틱 회귀분석의 비교 (일변량)**

- 적합방법 (손실함수)

$$l\big(y, f(x)\big)$$

- ols의 손실함수

$$l\big(y, f(x)\big) = \big(y - f(x)\big)^2$$

- cross entropy 손실함수

$$l\big(y, f(x)\big) = y \log f(x) + (1 - y) \log(1 - f(x))$$

# 로지스틱회귀분석

**의사결정**

- 선형회귀모형: 예측대상 -> 반응변수의 조건부 기대값 $E(Y|X)$

- 로시스틱 회귀모형
  - 예측대상 1: 반응변수의 조건부 기대값 $E(Y|X) = P(Y = 1|X)$ 인 경우는 추정된

$$f(x) = \frac{\exp(\beta_0 + \beta_1 x)}{1 + \exp(\beta_0 + \beta_1 x)}$$

  - 예측대상2: target 변수 Y인 경우 h (threshold) 를 정하여

$$\mathrm{I}(f(x) \geq h\ )$$

# 로지스틱회귀분석

**의사결정**

- 베이즈 의사결정경계
  - $B = \left\{x: P(Y = 1|X = x) = \frac{1}{2}\right\}$

- 베이즈 최적의사결정:오분류률을 최소화할 수 있는 의사결정방법
  - 만약 $P(Y = 1|X = x) \geq \frac{1}{2}$ 이라면 $\hat{Y} = 1$ , 그렇지 않으면 $\hat{Y} = 0$

# 로지스틱회귀분석

**베이즈의사결정경계와 확률분포**

- $X|Y$ 는 연속형 확률분포를 가정
- $f_k(x)$: *PDF of X|Y=k for k = 0, 1*
- $\pi_k = P(Y = k)$
- 베이즈 정리에 의해서

$$\Pr(Y = 1|X = x) = \frac{\pi_1 f_1(x)}{\pi_0 f_0(x) + \pi_1 f_1(x)}$$

- 베이즈 의사결정경계:

$$B = \{x : \pi_1 f_1(x) = \pi_0 f_0(x)\}$$

# 로지스틱회귀분석

**베이즈의사결정경계와 확률분포**

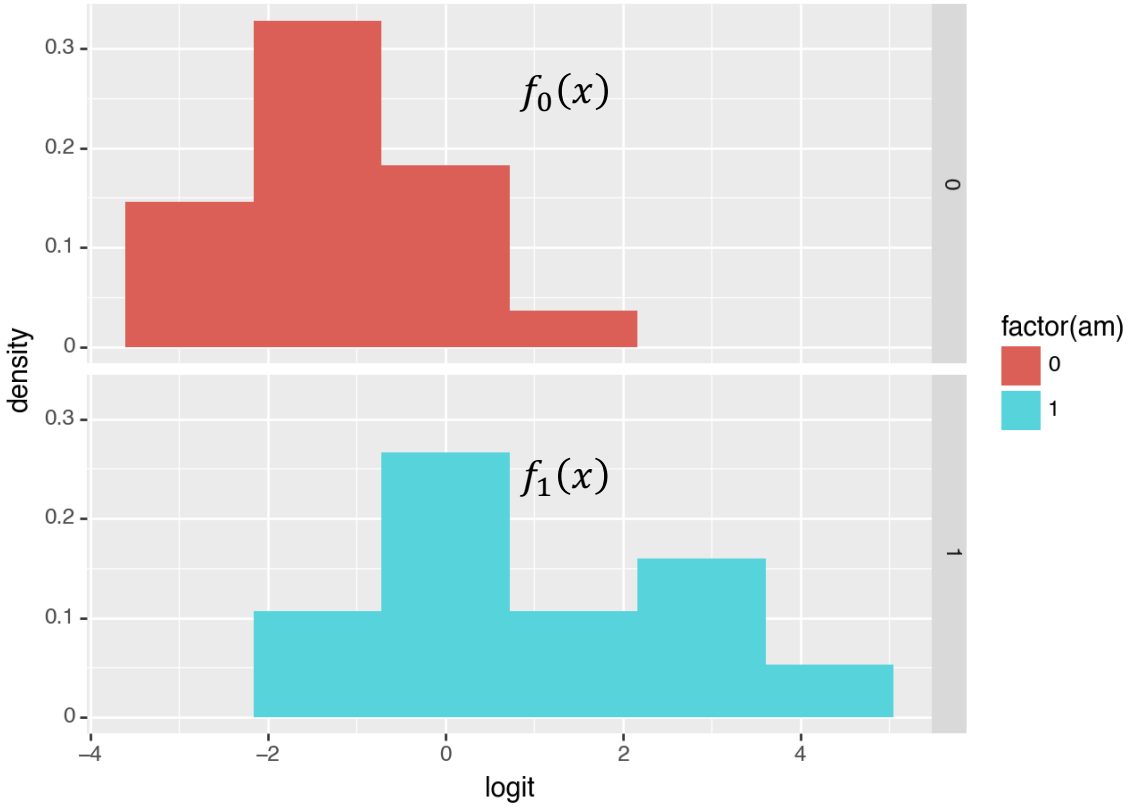
- 베이즈최적 의사결정

  $B_1 = \{x: \pi_1 f_1(x) \geq \pi_0 f_0(x)\}$, $B_0 = \{x: \pi_1 f_1(x) < \pi_0 f_0(x)\}$ 라고 할때,

  $\hat{Y} = 1$ if $x \in B_1$, $\hat{Y} = 0$,

  otherwise

- 연습: $X|Y = 1 \sim N(0,1), X|Y = 0 \sim N(0,-1)$, $\mathrm{P}(Y = 1) = 1/4$, $\mathrm{P}(Y = 0) = 3/4$ -> 베이즈최적의사결정 구간을 구하시오.



# 로지스틱회귀분석

**분류분석에서 확률분포의 비교**

- 분류 모형적합: 과거의 데이터로 성능이 좋은 모형을 선택
- 현장에 적용: 훈련데이터로 사용하지 않은 데이터가 새로운 입력으로 들어옴
  (과거의 데이터 패턴과 유사한가 아니면 많이 달라졌는가?)
- 과거의 $f_k(x)$: *PDF of X|Y=k* , $\pi_k = P(Y = k)$ *for k = 0,1 와 새로운 데이터에서 PDF (CDF)및 확률을 비교*
  *(분포간의 거리 측도 이용, 상대빈도의 비교)*

# 로지스틱회귀분석

**실습예제**

```
import statsmodels.formula.api as smf
model = smf.logit(formula='am ~ mpg', data=df).fit()
mpg_grid = np.linspace(5, 40, 100)
pred_df = pd.DataFrame({'mpg': mpg_grid})
pred_df['logistic'] = model.predict(pred_df)

model = smf.ols(formula='am ~ mpg', data=df).fit()
pred_df['linear'] = model.predict(pred_df)

pred_df_long = pred_df.melt(id_vars='mpg')
df['am'] = df['am'].astype('float')
plot = (
        ggplot(df, aes(x='mpg', y='am'))
        + geom_point(aes(color ='factor(am)'))
        + geom_line(data=pred_df_long, mapping=aes(x='mpg', y='value',
        linetype='factor(variable)', color='factor(variable)'))
        )
plot
```

# 로지스틱회귀분석

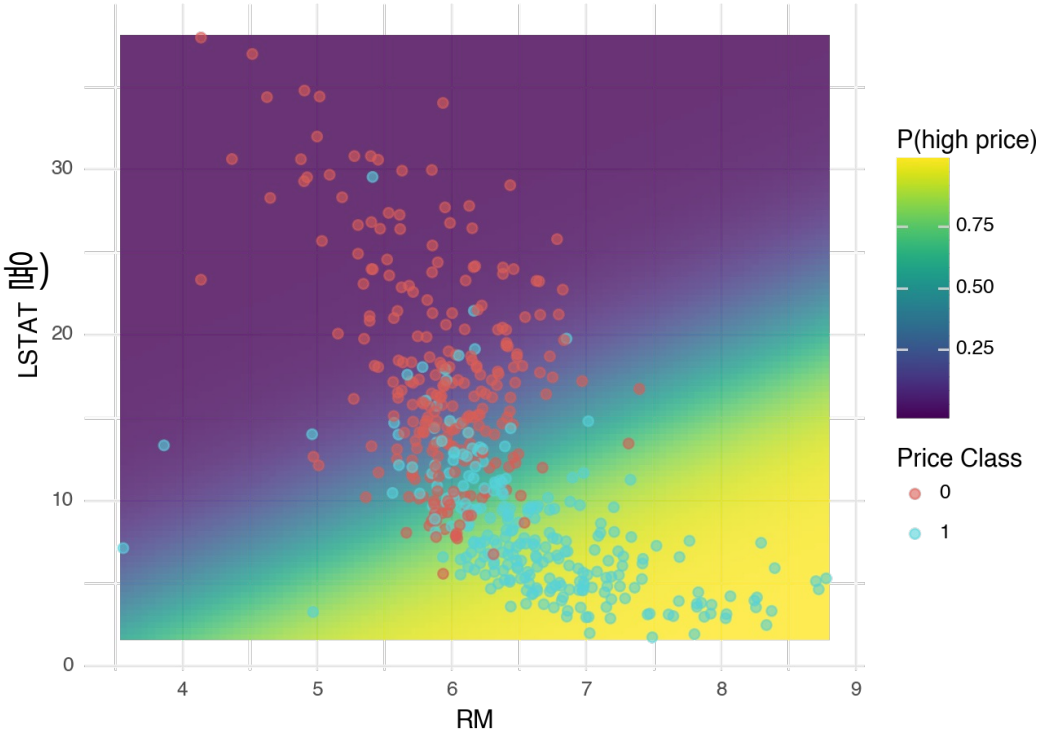
**실습예제**

```
import statsmodels.formula.api as smf
model = smf.logit(formula='am ~ mpg', data=df).fit()
pred_df = pd.DataFrame(df['am']).astype('int')
pred_df['logit'] = model.predict(df, linear=True)
plot = (
    ggplot(pred_df, aes(x='logit', fill='factor(am)'))
    + geom_histogram(aes(y ='..density..'), bins=6)
    + facet_grid('am~')
)

plot
```

# 로지스틱회귀분석

**로지스틱 회귀모형 (이변량 분석)**

- boston housing data
- Y: 중위수 가격 이상 유무
- X: RM(방 평균개수), LSTAT (저소득층비율)



# 로지스틱회귀분석

**실습코드**

```
path_file = r"http://ranking.uos.ac.kr/class/VIS/data/"
boston_df = pd.read_csv(path_file + 'boston.csv')
del boston_df["Unnamed: 0"]
boston_df = boston_df[["Price", "RM", "LSTAT"]]
boston_df['Y'] = (boston_df['Price']>=boston_df['Price'].median()).astype(int)

model = smf.logit('Y ~ RM + LSTAT', data=boston_df).fit()

lon_range = np.linspace(boston_df['RM'].min(), boston_df['RM'].max(), 100)
lat_range = np.linspace(boston_df['LSTAT'].min(), boston_df['LSTAT'].max(), 100)
lon_grid, lat_grid = np.meshgrid(lon_range, lat_range)

grid_df = pd.DataFrame({
         'RM': lon_grid.ravel(),
         'LSTAT': lat_grid.ravel()
         })
grid_df['prob'] = model.predict(grid_df)
```

# 로지스틱회귀분석

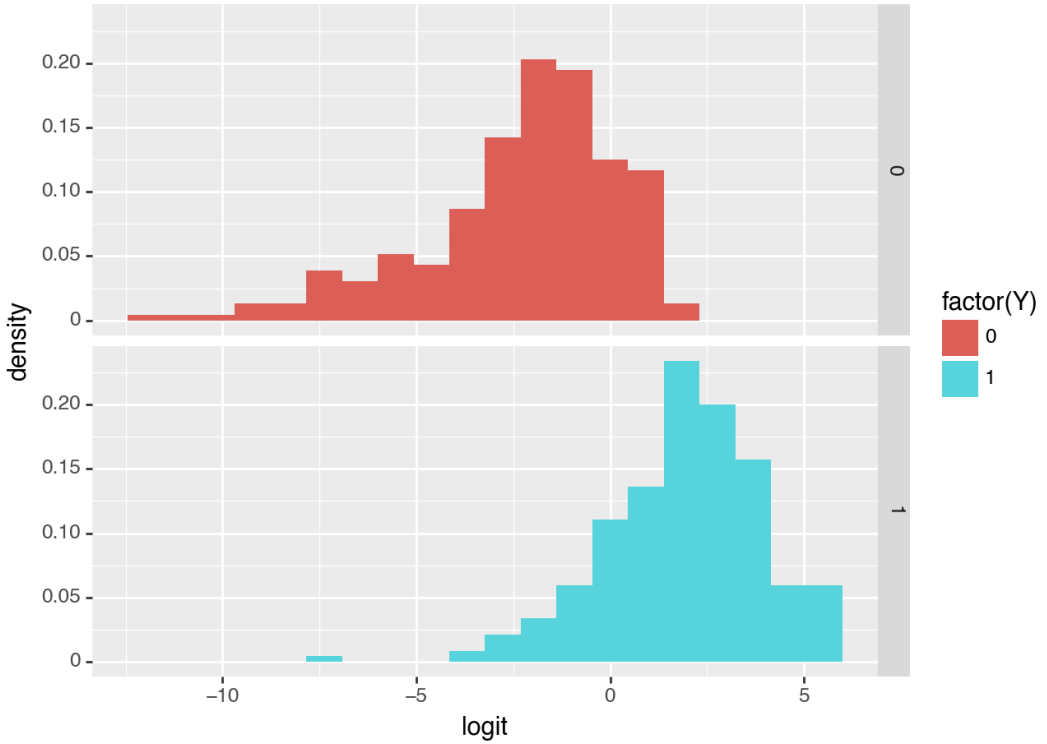
**실습코드**

```
plot = (
ggplot()
    + geom_tile(data=grid_df, mapping=aes(x='RM', y='LSTAT', fill='prob'),
    alpha=0.8)
    + geom_point(data=boston_df, mapping=aes(x='RM', y='LSTAT',
    color='factor(Y)'),
    size=2, alpha=0.6)
    + labs(fill='P(high price)', color='Price Class')
    + theme_minimal()
)

print(plot)
```

# 로지스틱회귀분석

### 집단별 추정 로그오즈 분석

```
boston_df['logit'] = model.predict(boston_df,
linear = True)
plot = (
ggplot(boston_df, aes(x='logit',
         fill='factor(Y)'))
        + geom_histogram(aes(y ='..density..'),
         bins=20)
         + facet_grid('Y~')
         )
plot
```



# 정리하기

1. **분산분석의 목적은 여러집단의 평균을 비교하는 것**
    1. boxplot 을 통한 평균비교
    2. 교호작용의 시각화
2. **회귀분석은 (연속형인) target 변수와 predictor 의 관계를 밝히는 것**
    1. 산점도 (geom_point)을 통한 관계의 시각화
    2. 집단별 변수간의 관계의 시각화/ 회귀진단
3. **로지스틱회귀분석은 이산형 이진변수와 predictor 의 관계를 밝히는 것**
    1. 로지스틱회귀모형을 이용한 조건부확률의 추정과 시각화
    2. 집단별 예측 확률의 로그 오즈의 분포 비교