# 05.01 확률론적 선형 회귀모형

### 6) 단일계수 t-검정 (single coefficient t-test)

정규화된 모수오차를 검정통계량으로 사용하면

$$\frac{\hat{w}}{se_i}$$

1. wi가 0인지 아닌지 검정 가능

$$H_0 : w_i = 0 (i = 0, 1, \ldots, K-1, K\text{개의가중치})$$

2. wi가 40인지로 놓고도 검증 가능

*코드 : print(result.t_test("X1 = 40)) *X1=40이란 건, X1가중치가 40 이라고 적는 것과 같은 의미

$$H_0 : w_i = 40 (i = 0, 1, \ldots, K-1, K\text{개의가중치})$$

3. w1 = w2 인지도 검증 가능

*코드 : print(result.nottem.t_test("C(month)[01] = C(month)]02"))

$$H_0 : w_1 = w_2 (i = 0, 1, \ldots, K-1, K\text{개의가중치})$$

### 7) 회귀분석 F-검정

- 단일계수 t-검정 : single coefficient t-test. 단일 계수(w_i) 에 대한 검정

- F-검정 : 모든 가중치가 다 쓸모 없다(종속변수와 feature들은 모두 상관성이 없다.)
"이 모델은 쓸모 없다" 를 반론하고 싶을 때, 사용 (이 가설이 아주 낮은 p값으로 기각되어야 좋음)
보통 "어느 모델이 성능이 더 좋다"를 증명할 때, 사용하는 검정 (성능이 좋을 수록 p값이 낮게 나올 것)

$$H_0 : w_0 = w_1 = w_2 = \ldots = w_{k-1} = 0$$

- 현실적으로 이런 H_0 가설은 받아들여질 가능성은 없다.
- 모두 쓸모 없다는 가설이 reject가 되더라도 0.01 로 기각, 0.000000001로 기각 되느냐의 차이
- 결국, 0.0000001로 기각되어야, p-value가 더 작게 기각되어야 역설적으로 '모델'이 쓸모 있다는 확률적 증명이 된다

### 8) statsmodel 패키지 회귀분석 결과표 해석 (05.01 14p)

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | value | R-squared: | 0.930 |
| Model: | OLS | Adj. R-squared: | 0.927 |
| Method: | Least Squares | F-statistic: | 277.3 |
| Date: | Mon, 17 Jun 2019 | Prob (F-statistic): | 2.96e-125 |
| Time: | 20:16:26 | Log-Likelihood: | -535.82 |
| No. Observations: | 240 | AIC: | 1096. |
| Df Residuals: | 228 | BIC: | 1137. |
| Df Model: | 11 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| C(month)[01] | 39.6950 | 0.518 | 76.691 | 0.000 | 38.675 | 40.715 |
| C(month)[02] | 39.1900 | 0.518 | 75.716 | 0.000 | 38.170 | 40.210 |
| C(month)[03] | 42.1950 | 0.518 | 81.521 | 0.000 | 41.175 | 43.215 |
| C(month)[04] | 46.2900 | 0.518 | 89.433 | 0.000 | 45.270 | 47.310 |
| C(month)[05] | 52.5600 | 0.518 | 101.547 | 0.000 | 51.540 | 53.580 |
| C(month)[06] | 58.0400 | 0.518 | 112.134 | 0.000 | 57.020 | 59.060 |
| C(month)[07] | 61.9000 | 0.518 | 119.592 | 0.000 | 60.880 | 62.920 |
| C(month)[08] | 60.5200 | 0.518 | 116.926 | 0.000 | 59.500 | 61.540 |
| C(month)[09] | 56.4800 | 0.518 | 109.120 | 0.000 | 55.460 | 57.500 |
| C(month)[10] | 49.4950 | 0.518 | 95.625 | 0.000 | 48.475 | 50.515 |
| C(month)[11] | 42.5800 | 0.518 | 82.265 | 0.000 | 41.560 | 43.600 |
| C(month)[12] | 39.5300 | 0.518 | 76.373 | 0.000 | 38.510 | 40.550 |

| | | | |
|---|---|---|---|
| Omnibus: | 5.430 | Durbin-Watson: | 1.529 |
| Prob(Omnibus): | 0.066 | Jarque-Bera (JB): | 5.299 |
| Skew: | -0.281 | Prob(JB): | 0.0707 |
| Kurtosis: | 3.463 | Cond. No. | 1.00 |

## 05.02 회귀분석의 기하학

- 투영행렬, 햇행렬, 영향도행렬
- 잔차행렬

중요하기 때문에, 유도하는 증명 해봐야함

## 05.03 레버리지와 아웃라이어

지금까지는 데이터 행렬 X 의 열단위 접근 (개별 feature에 대한 이야기(가중치))

이제부터는 데이터 행렬 X 의 행단위 접근 (개별 데이터에 대한 이야기)

- 개별 데이터 표본 하나하나가 회귀분석 결과에 미치는 영향력 분석
  : 레버리지 분석 / 아웃라이어 분석

## 1) 레버리지

※ 참고용!

**잔차행렬과 투영행렬**

벡터 $a$에서 다른 벡터 $b$를 변형하는 과정은 변형행렬(transforma matrix) $T$를 곱하는 연산으로 나타낼 수 있다.

$$b = Ta$$

종속값 벡터 $y$를 잔차 벡터 $e$로 변형하는 변환 행렬 $M$를 정의하자. 이 행렬을 **잔차행렬(residual matrix)**이라고 한다.

$$e = My$$

종속값 벡터 $y$를 예측값 벡터 $\hat{y}$로 변형하는 변환 행렬 $H$를 정의하자.. 이 행렬을 **투영행렬(projection matrix)**이라고 한다.

$$\hat{y} = Hy$$

잔차행렬은 다음과 같이 구한다.

$$\begin{aligned} e &= y - \hat{y} \\ &= y - Xw \\ &= y - X(X^TX)^{-1}X^Ty \\ &= (I - X(X^TX)^{-1}X^T)y \\ &= My \end{aligned}$$

투영행렬은 다음과 같이 구한다.

$$\begin{aligned} \hat{y} &= y - e \\ &= y - My \\ &= (I - M)y \\ &= X(X^TX)^{-1}X^Ty \\ &= Hy \end{aligned}$$

따라서 $M$, $H$는 각각 다음과 같다.

$$H = X(X^TX)^{-1}X^T$$
$$M = I - X(X^TX)^{-1}X^T$$

투영 행렬은 y로부터 ^기호가 붙은 $\hat{y}$를 계산한다고 해서 **햇(hat) 행렬** 또는 **영향도 행렬(influence matrix)**이라고 부르기도 한다. 영향도 행렬이라는 명칭의 의미는 아웃라이어 분석에서 다시 다룬다.

레버리지 : 실제 종속변수 값 $y$ 가 $\hat{y}$ 에 미치는 영향

레버리지 : 영향도 행렬($H$)의 대각성분 $h_{ii}$

$$\hat{y} = Hy$$

레버리지의 성질

1. $$0 \leq h_{ii} \leq 1$$
2. $$\mathrm{tr}(H) = \sum_{i}^{N} h_{ii} = \mathrm{K}$$

[시사점]

**1. 현실적으론 각각의 레버리지값(H의 대각성분)은 대부분 매우 작게 나오기 마련**

why? 현실에선

데이터의 갯수(대각성분의 갯수) $\mathrm{N}$ >> 모수의 갯수(가중치, 열의 갯수) $\mathrm{K}$

작은 수 K를 N 으로 쪼개서 가져가면, 각 대각성분은 그만큼 작아질 수 밖에!

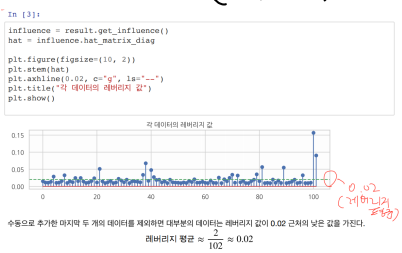
**2. 레버리지의 평균값**

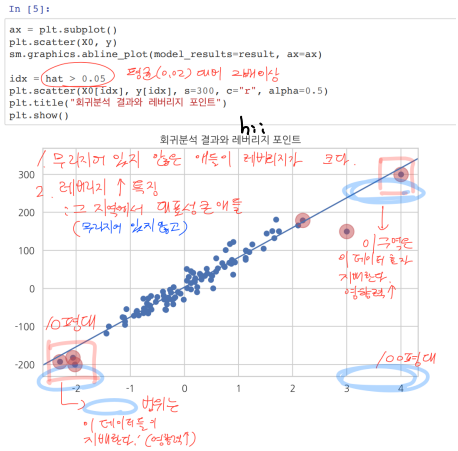
$$h_{ii} \approx \frac{K}{N}$$

보통, 이 평균값의 2~4배 보다 레버리지 값이 크면, 레버리지가 크다고 이야기 함

# 2) statsmodels를 이용한 레버리지 계산 (hii)

**코드** (4page, 5page)



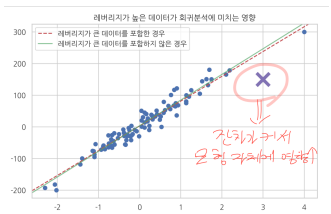



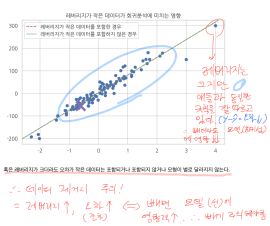

[시사점]

1. 무리지어 있지 않은 애들이 레버리지가 큼
2. 큰 레버리지 특징 : 그 지역에서 대표성 큰 애들 (혼자 그 구간을 담당하는 데이터)

# 3) 레버리지 영향 (hii)

레버리지의 영향 크기 : 해당 데이터의 잔차 크기에 달려있음 (6,7 page)



[시사점]

1. 데이터 제거 시 주의사항

    1) **'레버리지', '잔차' 모두 큰 데이터를 빼면**, 모델(회귀선) 자체가 흔들릴 수 있는 영향력을 갖기 때문에, 주의해야함

    **그런데 '잔차'는 우리가 아는 그 잔차가 아닌, '표준화된 잔차'를 봐야 한다!**

# 4) 아웃라이어

아웃라이어 : '표준화된 잔차'가 큰 데이터

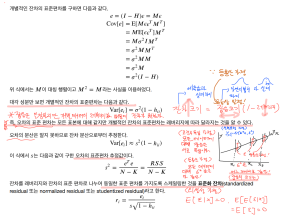
$y - \hat{y} = e$(잔차)

표준화된 잔차 : 잔차를 표준화한 것

# 4-1) 표준화 잔차

- 데이터 각각의 개별적인 영향들을 제거해, 모든 데이터의 잔차를 표준화된 상태에서 비교할 수 있게 함

  (개별적인 영향 : 개별 데이터의 레버리지 값)



개별 데이터의 잔차 ==>> 표준편차가 레버리지에 따라 달라짐 (9page)

- 원래 목적대로, 실 데이터 - 모델 간의 차이를 보려면, 이 개별적인 요인들을 다 제거해준 값으로 비교해줘야 공정한 비교!
  - 레버리지가 큰 데이터는 잔차크기가 상대적으로 작게 나옴. 모델과 차이가 큼에도 불구하고

# 4-2) So, 어떤 데이터를 제거해야 하는 가?

- 표준화된 잔차로 본 아웃라이어를 제거

  *대개는, 표준화 잔차가 2~4보다 크면 아웃라이어로 봄

  *엄밀하게는, Cook's Distance -> Fox' outlier recommendation을 기준으로 판단

- 대신, 레버리지가 큰 데이터는 일단 다시 한 번 살펴봐야 함(모델, 회귀선에 주는 영향이 크기 때문)

# 5) statsmodels 를 이용한 '표준화 잔차' 계산

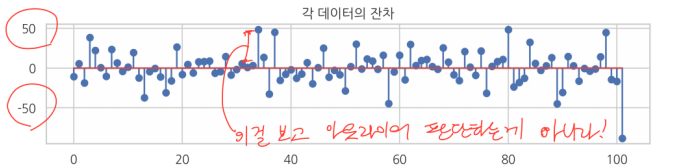
- regressionresult 객체의 regid 속성

In [9]:

```
plt.figure(figsize=(10, 2))
plt.stem(result.resid)
plt.title("각 데이터의 잔차")
plt.show()
```

표준화 X

regid_pearson 표준화 O




In [10]:

```
plt.figure(figsize=(10, 2))
plt.stem(result.resid_pearson)
plt.axhline(3, c="g", ls="--")
plt.axhline(-3, c="g", ls="--")
plt.title("각 데이터의 표준화 잔차")
plt.show()
```

왜냐면, 개별 데이터의 잔차 크기가 레버리지 값에 반비례 영향을 받기 때문에, 그냥 잔차 값으로 이는 선형적 모형과 동떨어진 결론을 본수 없다!

표준화한 상태!

# 5-1) Cook's Distance

$$D_i = \frac{r_i^2}{\mathrm{RSS}}\left[\frac{h_{ii}}{(1-h_{ii})^2}\right] \qquad r_i = \frac{e_i}{s\sqrt{1-h_{ii}}}$$

= 레버리지 x 표준화잔차.

## 아웃라이어 판단 기준

Fox' Outlier Recommendation 은 Cook's Distance가 다음과 같은 기준값보다 클 때 아웃라이어로 판단하자는 것이다.

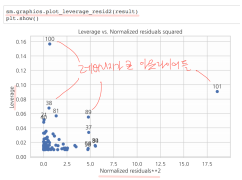
$$D_i > \frac{4}{N-K-1}$$

(데이터 개수) (가중치 개수)

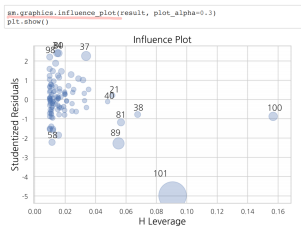
↳ 주사에상이라는 말.!

# 5-2) 레버리지가 큰 아웃라이어 시각화

- plot_leverage_resid2




- influence_plot




# 5-3) Cook's distance - Fox에 의한 아웃라이어 판단

- 제거 대상 (잔차 or 레버리지가 기준 이상으로 큰 데이터)

```
from statsmodels.graphics import utils

cooks_d2, pvals = influence.cooks_distance
K = influence.k_vars
fox_cr = 4 / (len(y) - K - 1)
idx = np.where(cooks_d2 > fox_cr)[0]

ax = plt.subplot()
plt.scatter(X0, y)
plt.scatter(X0[idx], y[idx], s=300, c="r", alpha=0.5)
utils.annotate_axes(range(len(idx)), idx,
                    list(zip(X0[idx], y[idx])), [(-20, 15)] * len(idx), size="sm
all", ax=ax)
plt.title("Fox Recommendaion으로 선택한 아웃라이어")
plt.show()
```