

4주차 3차시 및 5주차 1차시: Py 실습(단순회귀분석)

1. 단순회귀분석-공식 이용
2. 단순회귀분석-명령어 이용 및 그림 그리기
3. 단순회귀분석-회귀모형의 형태

1. 단순회귀분석-공식 이용

b2-ch2-1.py

```
import numpy as np
from scipy.stats.distributions import t as tdist
xx = [2,3,4,5,6]
yy = [4,4,6,6,10]
print("Data of x is :", xx)
print("Data of y is :", yy)
x = np.array(xx)
y = np.array(yy)
n = len(x)
sum_x = np.sum(x)
sum_y = np.sum(y)
m_x = np.mean(x)
m_y = np.mean(y)
sum_xy = np.sum(np.multiply(x, y))
sum_xsq = np.sum(x**2)
sum_ysq = np.sum(y**2)
print("Number of Sample is :", n)
print("Sum of x is :", sum_x)
print("Sum of y is :", sum_y)
print("Mean of x is :", m_x)
print("Mean of y is :", m_y)
print("Sum of x is :", sum_x)
print("Sum of y is :", sum_y)
print("Sum of x*y is :", sum_xy)
print("Sum of x-square is :", sum_xsq)
print("Sum of y-square is :", sum_ysq)
```

X	2	3	4	5	6
Y	4	4	6	6	10

$$\sum X_i = 20, \sum Y_i = 30, \sum X_i Y_i = 134, \\ \bar{X} = 4, \bar{Y} = 6, \sum X_i^2 = 90, \sum Y_i^2 = 204$$

b2-ch2-1.py(계속)

```

beta1 = (sum_xy-m_x*sum_y)/ (sum_xsq-m_x*sum_x)

beta0 = m_y - beta1*m_x

print("OLS Estimate of beta1 is :", round(beta1,3))

print("OLS Estimate of beta0 is :", round(beta0,3))

b1hat = np.cov(x,y,ddof=1)[0,1] / np.var(x, ddof=1)

b0hat = m_y - b1hat*m_x

print("OLS Estimate of beta1 is :", round(b1hat,3))

print("OLS Estimate of beta0 is :", round(b0hat,3))

dx = x - m_x

dy = y - m_y

sum_dxdy = np.sum(np.multiply(dx, dy))

sum_dxsq = np.sum(dx**2)

sum_dy_sq = np.sum(dy**2)

ssr = sum_dy_sq - beta1*sum_dxdy

print("Deviation of x is :", dx)

print("Deviation of y is :", dy)

print("Sum of dx*dy is :", sum_dxdy)

print("Sum of dx-square is :", sum_dx_sq)

print("Sum of dy-square is :", sum_dy_sq)

print("Sum of dx-square is :", sum_dx_sq)

print("Sum of residual-square is :", round(ssr,3))

sigusq = ssr/(n-2)

beta1_v = sigusq / sum_dx_sq

beta0_v = sigusq*(sum_xsq / (n*sum_dx_sq))

print("Variance of residual is :", round(sigusq,3))

print("Variance of beta1 is :", round(beta1_v,3))

print("Variance of beta0 is :", round(beta0_v,3))
  
```

$$\therefore \hat{\beta}_1 = \frac{\sum_{i=1}^n X_i Y_i - \bar{X} \sum_{i=1}^n Y_i}{\sum_{i=1}^n X_i^2 - \bar{X} \sum_{i=1}^n X_i} = \frac{134 - 4 \times 30}{90 - 4 \times 20} = \frac{14}{10} = 1.4$$

$$\therefore \hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X} - (1.4)(4) = 0.4$$

$$\sum_{i=1}^n (X_i - \bar{X})^2 = \sum_{i=1}^n X_i^2 - \bar{X} \sum_{i=1}^n X_i = 90 - (4)(20) = 10$$

$$\sum_{i=1}^n (Y_i - \bar{Y})^2 = \sum_{i=1}^n Y_i^2 - \bar{Y} \sum_{i=1}^n Y_i = 204 - (6)(30) = 24$$

$$\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y}) = \sum_{i=1}^n X_i Y_i - \bar{X} \sum_{i=1}^n Y_i = 134 - (4)(30) = 14$$

$$\sum_{i=1}^n \hat{e}_i^2 = \sum_{i=1}^n (Y_i - \bar{Y})^2 - \hat{\beta}_1 \sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y}) = 24 - (1.4)(14) = 4.4$$

$$\therefore \hat{\sigma}_u^2 = \frac{4.4}{3} = 1.4667$$

$$\text{var}(\hat{\beta}_1) = \hat{\sigma}_u^2 \frac{1}{\sum_{i=1}^n (X_i - \bar{X})^2} = (1.4667) \left(\frac{1}{10}\right) = 0.14667$$

$$\text{var}(\hat{\beta}_0) = \hat{\sigma}_u^2 \frac{\sum X_i^2}{n \sum_{i=1}^n (X_i - \bar{X})^2} = (1.4667) \left(\frac{90}{5 \times 10}\right) = 2.64006$$

b2-ch2-1.py(계속)

```

rsq = beta1**2 * sum_dxsq / sum_dysq
print("R-square is :", round(rsq,3))
beta1_t = beta1 / np.sqrt(beta1_v)
print("t-statistic of beta1 is :", round(beta1_t,3))
prob_t = tdist.cdf(beta1_t, df=3)
crit_t = tdist.ppf(0.975, df=3)
print("Cumulative Probability up to beta1_t is :", round(prob_t,3))
print("Right-Hand-Side Critical Value of t-Distribution w/ 0.05% significance level is :", round(crit_t,3))
b1hat_lb = b1hat_crit_t*np.sqrt(beta1_v)
b1hat_ub = b1hat+crit_t*np.sqrt(beta1_v)
print("Lower bound of 95% Confidence Interval for beta1 is :", round(b1hat_lb,3))
print("Upper bound of 95% Confidence Interval for beta1 is :", round(b1hat_ub,3))
x0 = 7
yhat = beta0+beta1*x0
sigesq_ind = sigusq*(1 + (1/n) + ((x0-m_x)**2/sum_dxsq))
sige_ind = np.sqrt(sigesq_ind)
yhat_ind_lb = yhat - crit_t*sige_ind
yhat_ind_ub = yhat + crit_t*sige_ind
print("Point Prediction of y given x=7 is :", yhat)
print("Variance of Prediction Error for Individual Prediction is :", round(sigesq_ind,3))
print("Standard Error of Prediction Error for Individual Prediction is :", round(sige_ind,3))
print("Lower bound of 95% Interval Prediction for Individual is :", round(yhat_ind_lb,3))
print("Upper bound of 95% Interval Prediction for Individual is :", round(yhat_ind_ub,3))
sigesq_mean = sigusq*((1/n) + ((x0-m_x)**2/sum_dxsq))
sige_mean = np.sqrt(sigesq_mean)
yhat_mean_lb = yhat - crit_t*sige_mean
yhat_mean_ub = yhat + crit_t*sige_mean
print("Point Prediction of y given x=7 is :", yhat)
print("Variance of Prediction Error for Mean Prediction is :", round(sigesq_mean,3))
print("Standard Error of Prediction Error for Mean Prediction is :", round(sige_mean,3))
print("Lower bound of 95% Interval Prediction for Mean is :", round(yhat_mean_lb,3))
print("Upper bound of 95% Interval Prediction for Mean is :", round(yhat_mean_ub,3))

```

$$R^2 = \frac{\hat{\beta}_1^2 \sum_{i=1}^n (X_i - \bar{X})^2}{\sum_{i=1}^n (Y_i - \bar{Y})^2} = \frac{19.6}{24} = 0.817$$

$$t = \frac{1.4 - 0}{0.383} = 3.655 \sim t_3$$

$$\hat{\beta}_1 \pm t_{(n-2, \frac{\alpha}{2})} se(\hat{\beta}_1) = 1.4 \pm 3.182(0.383) = [0.181, 2.618]$$

$$X_0 = 7 \text{ 일 때}$$

$$\hat{Y} = 0.4 + (1.4)(7) = 10.2 \text{ 억 원}$$

$$\sigma_\epsilon^2 = \sigma_u^2 \left(1 + \frac{1}{n} + \frac{(X_0 - \bar{X})^2}{\sum_{i=1}^n x_i^2} \right) = (1.4667) \left(1 + \frac{1}{5} + \frac{(7-4)^2}{10} \right) = 3.08$$

$$\sigma_\epsilon^2 = \sigma_u^2 \left(\frac{1}{n} + \frac{(X_0 - \bar{X})^2}{\sum_{i=1}^n x_i^2} \right) = (1.4667) \left(\frac{1}{5} + \frac{(7-4)^2}{10} \right) = 1.61$$

$$\hat{\beta}_0 + \hat{\beta}_1 X_0 \pm t_{(n-2, \frac{\alpha}{2})} \sigma_\epsilon = 10.2 \pm (3.182)\sqrt{3.08} = [4.61, 15.78]$$

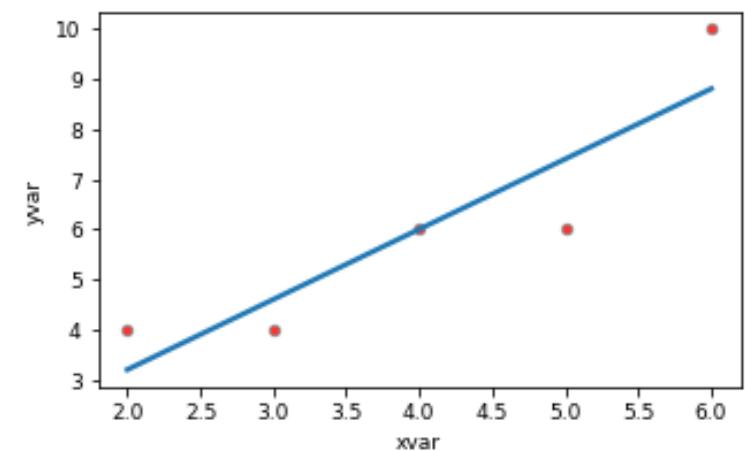
$$\hat{\beta}_0 + \hat{\beta}_1 X_0 \pm t_{(n-2, \frac{\alpha}{2})} \sigma_\epsilon = 10.2 \pm (3.182)\sqrt{1.61} = [6.15, 14.24]$$

2. 단순회귀분석-명령어 이용 및 그림 그리기

b2-ch2-2.py

```

import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import statsmodels.formula.api as smf
import statsmodels.api as sm
from statsmodels.sandbox.regression.predstd import wls_prediction_std
# initialize list of lists
data = [[2,4], [3,4], [4,6],[5,6],[6,10]]
# Create the pandas DataFrame
df = pd.DataFrame(data, columns=['xvar', 'yvar'])
# print dataframe.
df
plt.rcParams['figure.figsize'] = (5,3) # 그림 크기를 전체적으로(globally) 설정
plt.rcParams.update({'font.size': 9}) # 그림 폰트를 전체적으로(globally) 설정
sns.replot(x='xvar', y='yvar', data=df,order=1, ci=None, scatter_kws={'color':'r', 's':20, 'edgecolor':'grey'})
plt.savefig("C:/BOOK\PyBasics/PyEm/code/b2-ch2-2.png")
model = smf.ols(formula ='yvar ~ xvar', data=df)
olsfit = model.fit()
print(olsfit.summary())
b = olsfit.params # b[0] = Intercept, b[1] = slope
print(f'b:{\n{b}\n}')
# 추정계수를 표로 작성
est = smf.ols('yvar ~ xvar', df).fit()
print(est.summary().tables[1])
# y의 추정치를 계산
y_hat = b[0] + b[1] * df['xvar']
y_hat
print("Fitted value of y is:", f'{y_hat}')
  
```



OLS Regression Results

Dep. Variable:	yvar	R-squared:	0.817			
Model:	OLS	Adj. R-squared:	0.756			
Method:	Least Squares	F-statistic:	13.36			
Date:	Mon, 25 Mar 2024	Prob (F-statistic):	0.0354			
Time:	13:49:41	Log-Likelihood:	-6.7751			
No. Observations:	5	AIC:	17.55			
Df Residuals:	3	BIC:	16.77			
Df Model:	1					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
Intercept	0.4000	1.625	0.246	0.821	-4.771	5.571
xvar	1.4000	0.383	3.656	0.035	0.181	2.619
Omnibus:	nan	Durbin-Watson:	2.509			
Prob(Omnibus):	nan	Jarque-Bera (JB):	0.396			
Skew:	-0.174	Prob(JB):	0.821			
Kurtosis:	1.667	Cond. No.	13.4			

	coef	std err	t	P> t	[0.025	0.975]
Intercept	0.4000	1.625	0.246	0.821	-4.771	5.571
xvar	1.4000	0.383	3.656	0.035	0.181	2.619

```

Fitted value of y is:
0    3.2
1    4.6
2    6.0
3    7.4
4    8.8
  
```

b2-ch2-2.py(계속)

(앞에서 계속)

```
# prediction interval 추정
x = np.asarray([2,3,4,5,6])
y = np.asarray([4,4,6,6,10])
X = sm.add_constant(x)
re = sm.OLS(y, X).fit()
prstd, iv_l, iv_u = wls_prediction_std(re)

#Significance level:
#sl = 0.05

#Evaluate mean value at a required point x0. Here, at the point (1.0,7.0) for N_model=1:
x0 = np.asarray([1.0, 7.0])# If you have no constant in your model, remove the first 1.0. For more dimensions, add the desired values.
predictions = re.get_prediction(x0)

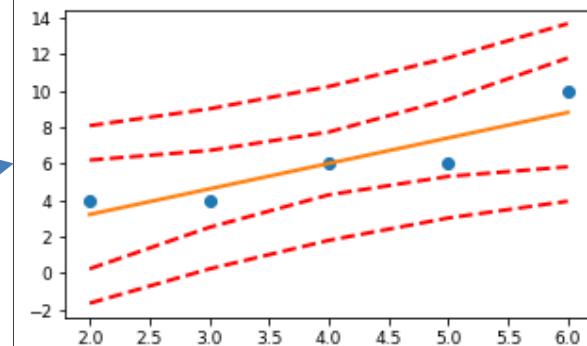
print("95% Prediction Interval for Individual and Mean Prediction is",f':\n{predictions.summary_frame(alpha=0.05)}\n')
#predictions.summary_frame(alpha=0.05)

from statsmodels.stats.outliers_influence import summary_table
st, data, ss2 = summary_table(re, alpha=0.05)
fittedvalues = data[:, 2]
predict_mean_se = data[:, 3]
predict_mean_ci_low, predict_mean_ci_upp = data[:, 4:6].T
predict_ci_low, predict_ci_upp = data[:, 6:8].T

# Check we got the right things
print(np.max(np.abs(re.fittedvalues - fittedvalues)))
print(np.max(np.abs(iv_l - predict_ci_low)))
print(np.max(np.abs(iv_u - predict_ci_upp)))
plt.plot(x, y, 'o')
plt.plot(x, fittedvalues, '-')
plt.plot(x, predict_ci_low, 'r--', lw=2)
plt.plot(x, predict_ci_upp, 'r--', lw=2)
plt.plot(x, predict_mean_ci_low, 'r--', lw=2)
plt.plot(x, predict_mean_ci_upp, 'r--', lw=2)
plt.show()
```

95% Prediction Interval for Individual and Mean Prediction is :

	mean	mean_se	mean_ci_lower	mean_ci_upper	obs_ci_lower	obs_ci_upper
0	10.2	1.270171	6.15775	14.24225	4.614829	15.785171



3. 단순회귀분석-회귀모형의 형태

b2-ch2-3.py

```
import numpy as np
import statsmodels.api as sm
x = np.asarray([2,3,4,5,6])
y = np.asarray([4,4,6,6,10])
X = sm.add_constant(x)
lx = np.log(x)
ly = np.log(y)
IX = sm.add_constant(lx)
rx = 1/x
rX = sm.add_constant(rx)

ols_1 = sm.OLS(y, X).fit()
ols_2 = sm.OLS(ly, IX).fit()
ols_3 = sm.OLS(y, rX).fit()
ols_4 = sm.OLS(ly, X).fit()
ols_5 = sm.OLS(y, IX).fit()

ols_1.summary().tables[1]
ols_2.summary().tables[1]
ols_3.summary().tables[1]
ols_4.summary().tables[1]
ols_5.summary().tables[1]

print("Summary of ols_1 is : ",ols_1.summary().tables[1])
print("Summary of ols_2 is : ",ols_2.summary().tables[1])
print("Summary of ols_3 is : ",ols_3.summary().tables[1])
print("Summary of ols_4 is : ",ols_4.summary().tables[1])
print("Summary of ols_5 is : ",ols_5.summary().tables[1])
```

```
Summary of ols_1 is :
            coef    std err        t     P>|t|      [0.025    0.975]
-----
const      0.4000    1.625     0.246    0.821    -4.771    5.571
x1         1.4000    0.383     3.656    0.035     0.181    2.619
-----
Summary of ols_2 is :
            coef    std err        t     P>|t|      [0.025    0.975]
-----
const      0.7112    0.315     2.258    0.109    -0.291    1.714
x1         0.7755    0.230     3.377    0.043     0.045    1.506
-----
Summary of ols_3 is :
            coef    std err        t     P>|t|      [0.025    0.975]
-----
const     10.0909    2.138     4.719    0.018     3.286   16.896
x1       -14.1066    6.821    -2.068    0.130    -35.813    7.600
-----
Summary of ols_4 is :
            coef    std err        t     P>|t|      [0.025    0.975]
-----
const      0.8365    0.206     4.057    0.027     0.180    1.493
x1         0.2238    0.049     4.605    0.019     0.069    0.378
-----
Summary of ols_5 is :
            coef    std err        t     P>|t|      [0.025    0.975]
-----
const     -0.2655    2.405    -0.110    0.919    -7.918    7.387
x1         4.7615    1.753     2.716    0.073    -0.817   10.340
-----
```