다중선형회귀

독립변수가 두 개 이상인 경우의 회귀분석

분석 정확도를 높이기 위해 적절하지 않은 변수를 추려내는 과정을 반복적으로 수행하여 최적의 독립변수 그룹을 찾아내는 것을 목표로 한다.

#01. 작업 준비

1) 패키지 참조

```
from pandas import read_excel, DataFra
from statsmodels.formula.api import ol
from matplotlib import pyplot as plt
import seaborn as sb
import sys
import os
import statsmodels.api as sm
```

```
sys.path.append(os.path.dirname(os.pat
from helper import ext_ols, my_ols
```

2) 데이터 가져오기

필드	설명
CRIM	범죄율
ZN	25,000 평방피트를 초과 거주지역 비율
INDUS	비소매상업지역 면적 비율
CHAS	찰스강의 경계에 위치한 경우는 1, 아니면 0
NOX	일산화질소 농도

다중선형회귀

#01. 작업 준비

1) 패키지 참조

2) 데이터 가져오기

#02. 파이썬의 ols 객체로 분석 (맛보기)

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

분석 수행

분석 결과 표 확인

VIF가 10 이상인 값을 제외하고 다시 분석

변수를 다시 선정하여 진행

#04. 결과 비교하기

필드	설명
RM	주택당 방 수
AGE	1940년 이전에 건축된 주택의 비율
DIS	직업센터의 거리
RAD	방사형 고속도로까지의 거리
TAX	재산세율
PTRATIO	학생/교사 비율
B	인구 중 흑인 비율
LSTAT	인구 중 하위 계층 비율
MEDV	집값
CAT.MEDV	$3000 이상 여부

```
df = read_excel("https://data.hossam.k
df
```

	CRIM	ZN	INDUS	CHAS	NO
0	0.00632	18.0	2.31	0	0.53
1	0.02731	0.0	7.07	0	0.46
2	0.02729	0.0	7.07	0	0.46
3	0.03237	0.0	2.18	0	0.45
4	0.06905	0.0	2.18	0	0.45
...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.57
502	0.04527	0.0	11.93	0	0.57
503	0.06076	0.0	11.93	0	0.57
504	0.10959	0.0	11.93	0	0.57
505	0.04741	0.0	11.93	0	0.57

506 rows × 15 columns

```
df.shape
```

```
(506, 15)
```

#02. 파이썬의 ols 객체로 분석 (맛보기)

임의의 독립변수를 선정하여 분석 수행

```
model = ols("MEDV ~ CRIM + INDUS", dat
fit = model.fit()
fit.summary()
```

OLS Regression Results

Dep. Variable:	MEDV	**R-squared:**	0.27
Model:	OLS	**Adj. R-squared:**	0.27
Method:	Least Squares	**F-statistic:**	96.8
Date:	Wed, 26 Jul 2023	**Prob (F-statistic):**	2.66 36
Time:	10:39:36	**Log-Likelihood:**	-175
No. Observations:	506	**AIC:**	3522
Df Residuals:	503	**BIC:**	3534
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|
Intercept	29.2483	0.670	43.624	0.000
CRIM	-0.2455	0.044	-5.536	0.000
INDUS	-0.5234	0.056	-9.414	0.000

Omnibus:	193.751	**Durbin-Watson:**	0.739
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	653.88
Skew:	1.800	**Prob(JB):**	1.03e-142
Kurtosis:	7.248	**Cond. No.**	27.7

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

```
cls = list(df.columns)
cls.remove("MEDV")
cls.remove("CAT. MEDV")
cls
```

```
['CRIM',
 'ZN',
 'INDUS',
```

```
'CHAS',
'NOX',
'RM',
'AGE',
'DIS',
'RAD',
'TAX',
'PTRATIO',
'B',
'LSTAT']
```

분석 수행

```
model, fit, summary, table, result, g
```

```
summary
```

OLS Regression Results

Dep. Variable:	MEDV	**R-squared:**	0.74
Model:	OLS	**Adj. R-squared:**	0.73
Method:	Least Squares	**F-statistic:**	108.
Date:	Wed, 26 Jul 2023	**Prob (F-statistic):**	6.72 135
Time:	10:39:36	**Log-Likelihood:**	-149
No. Observations:	506	**AIC:**	3026
Df Residuals:	492	**BIC:**	3085
Df Model:	13		
Covariance Type:	nonrobust		

다중선형회귀

#01. 작업 준비

1) 패키지 참조

2) 데이터 가져오기

#02. 파이썬의 ols 객체로 분석 (맛보기)

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

분석 수행

분석 결과 표 확인

VIF가 10 이상인 값을 제외하고 다시 분석

변수를 다시 선정하여 진행

#04. 결과 비교하기

	coef	std err	t	P>|t|
Intercept	36.4595	5.103	7.144	0.000
CRIM	-0.1080	0.033	-3.287	0.001
ZN	0.0464	0.014	3.382	0.001
INDUS	0.0206	0.061	0.334	0.738
CHAS	2.6867	0.862	3.118	0.002
NOX	-17.7666	3.820	-4.651	0.000
RM	3.8099	0.418	9.116	0.000
AGE	0.0007	0.013	0.052	0.958
DIS	-1.4756	0.199	-7.398	0.000
RAD	0.3060	0.066	4.613	0.000
TAX	-0.0123	0.004	-3.280	0.001
PTRATIO	-0.9527	0.131	-7.283	0.000
B	0.0093	0.003	3.467	0.001
LSTAT	-0.5248	0.051	-10.347	0.000

Omnibus:	178.041	**Durbin-Watson:**	1.078
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	783.12
Skew:	1.521	**Prob(JB):**	8.84e-171
Kurtosis:	8.281	**Cond. No.**	1.51e+

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 1.51e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

분석 결과 표 확인

```
table
```

		B	표준 오차	β	
종속변수	독립변수				
MEDV	CRIM	-0.1080	0.033	0	-3
	ZN	0.0464	0.014	0	3
	INDUS	0.0206	0.061	0	0
	CHAS	2.6867	0.862	0	3
	NOX	-17.7666	3.820	0	-4
	RM	3.8099	0.418	0	9
	AGE	0.0007	0.013	0	0
	DIS	-1.4756	0.199	0	-7
	RAD	0.3060	0.066	0	4
	TAX	-0.0123	0.004	0	-3
	PTRATIO	-0.9527	0.131	0	-7
	B	0.0093	0.003	0	3
	LSTAT	-0.5248	0.051	0	-1

VIF가 10 이상인 값을 제외하고 다시 분석

```
model2, fit2, summary2, table2, result
```

```
summary2
```

다중선형회귀

#01. 작업 준비

1) 패키지 참조

2) 데이터 가져오기

#02. 파이썬의 ols 객체로 분석 (맛보기)

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

분석 수행

분석 결과 표 확인

VIF가 10 이상인 값을 제외하고 다시 분석

변수를 다시 선정하여 진행

#04. 결과 비교하기

OLS Regression Results

Dep. Variable:	MEDV	**R-squared:**	0.26
Model:	OLS	**Adj. R-squared:**	0.25
Method:	Least Squares	**F-statistic:**	59.6
Date:	Wed, 26 Jul 2023	**Prob (F-statistic):**	5.23 33
Time:	10:39:36	**Log-Likelihood:**	-176
No. Observations:	506	**AIC:**	3534
Df Residuals:	502	**BIC:**	3551
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|
Intercept	21.9715	0.449	48.937	0.000
CRIM	-0.3398	0.042	-8.108	0.000
ZN	0.1199	0.015	7.761	0.000
CHAS	6.1729	1.392	4.435	0.000

Omnibus:	150.219	**Durbin-Watson:**	0.821
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	358.81
Skew:	1.524	**Prob(JB):**	1.21e-78

Kurtosis:	5.779	**Cond. No.**	103.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```
table2
```

		B	표준 오차	β	
종속 변수	**독립 변수**				
MEDV	**CRIM**	-0.3398	0.042	0	-8.108
	ZN	0.1199	0.015	0	7.761
	CHAS	6.1729	1.392	0	4.435

```
result2
```

```
'R(0.263), R^2(0.258), F(59.67), 유의확
```

```
goodness2
```

```
'MEDV에 대하여 CRIM,ZN,CHAS로 예측하는 회
```

```
varstr2
```

```
['CRIM의 회귀계수는 -0.3398(p<0.05)로, M
 'ZN의 회귀계수는 0.1199(p<0.05)로, MEDV
```

```
'CHAS의 회귀계수는 6.1729(p<0.05)로, ME
```

변수를 다시 선정하여 진행

```
names3 = ["CRIM", "INDUS", "CHAS", "NO
result = my_ols(df, x=names3, y="MEDV"
```

```
result.summary
```

OLS Regression Results

Dep. Variable:	MEDV	**R-squared:**	1.00
Model:	OLS	**Adj. R-squared:**	1.00
Method:	Least Squares	**F-statistic:**	7.50
Date:	Wed, 26 Jul 2023	**Prob (F-statistic):**	0.00
Time:	10:39:36	**Log-Likelihood:**	1462
No. Observations:	506	**AIC:**	-2.92
Df Residuals:	493	**BIC:**	-2.91
Df Model:	12		
Covariance Type:	nonrobust		

	coef	std err	t	P>
Intercept	1.288e-14	7.35e-14	0.175	0.8
CRIM	1.931e-	4.61e-	0.419	0.6

다중선형회귀

#01. 작업 준비

1) 패키지 참조

2) 데이터 가져오기

#02. 파이썬의 ols 객체로 분석 (맛보기)

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

분석 수행

분석 결과 표 확인

VIF가 10 이상인 값을 제외하고 다시 분석

변수를 다시 선정하여 진행

#04. 결과 비교하기

	16	16		
INDUS	4.97e-16	8.56e-16	0.581	0.5
CHAS	3.9e-15	1.26e-14	0.310	0.7
NOX	1.776e-15	5.57e-14	0.032	0.9
RM	-3.941e-15	6.52e-15	-0.604	0.5
AGE	-2.492e-15	1.9e-16	-13.116	0.0
DIS	2.227e-15	2.76e-15	0.806	0.4
TAX	3.691e-16	3.28e-17	11.262	0.0
PTRATIO	-7.147e-16	1.86e-15	-0.385	0.7
B	1.258e-16	3.93e-17	3.204	0.0
LSTAT	-5.967e-16	8.03e-16	-0.743	0.4
MEDV	1.0000	6.35e-16	1.57e+15	0.0

Omnibus:	19.254	**Durbin-Watson:**	0.451
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	8.992
Skew:	-0.041	**Prob(JB):**	0.0112
Kurtosis:	2.352	**Cond. No.**	1.51e+0

다중선형회귀

#01. 작업 준비

1) 패키지 참조

2) 데이터 가져오기

#02. 파이썬의 ols 객체로 분석 (맛보기)

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

분석 수행

분석 결과 표 확인

VIF가 10 이상인 값을 제외하고 다시 분석

변수를 다시 선정하여 진행

#04. 결과 비교하기

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.51e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

```
result.table
```

		B	표준 오차	β	
종속변수	독립변수				
MEDV	CRIM	1.931e-16	4.61e-16	0	0
	INDUS	4.97e-16	8.56e-16	0	0
	CHAS	3.9e-15	1.26e-14	0	0
	NOX	1.776e-15	5.57e-14	0	0
	RM	-3.941e-15	6.52e-15	0	-
	AGE	-2.492e-15	1.9e-16	0	-
	DIS	2.227e-15	2.76e-15	0	0
	TAX	3.691e-16	3.28e-17	0	1
	PTRATIO	-7.147e-16	1.86e-15	0	-
	B	1.258e-16	3.93e-17	0	3

		B	표준 오차	β	
종속변수	독립변수				
	LSTAT	-5.967e-16	8.03e-16	0	-
	MEDV	1.0000	6.35e-16	0	1

#04. 결과 비교하기

```
실제집값 = df["MEDV"]
실제집값
```

```
0       24.0
1       21.6
2       34.7
3       33.4
4       36.2
       ...
501     22.4
502     20.6
503     23.9
504     22.0
505     11.9
Name: MEDV, Length: 506, dtype: float6
```

```
result1 = fit.predict(df.filter(cls))
result1
```

```
0       30.003843
1       25.025562
2       30.567597
3       28.607036
4       27.943524
```

다중선형회귀

#01. 작업 준비

1) 패키지 참조

2) 데이터 가져오기

#02. 파이썬의 ols 객체로 분석 (맛보기)

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

분석 수행

분석 결과 표 확인

VIF가 10 이상인 값을 제외하고 다시 분석

변수를 다시 선정하여 진행

#04. 결과 비교하기

```
          ...
501    23.533341
502    22.375719
503    27.627426
504    26.127967
505    22.344212
Length: 506, dtype: float64
```

```
result2 = fit2.predict(df.filter(['CR
result2
```

```
0      24.127287
1      21.962235
2      21.962242
3      21.960515
4      21.948050
          ...
501    21.950232
502    21.956131
503    21.950867
504    21.934273
505    21.955404
Length: 506, dtype: float64
```

```
result3 = result.fit.predict(df.filte
result3
```

```
0      24.0
1      21.6
2      34.7
3      33.4
4      36.2
       ...
501    22.4
502    20.6
503    23.9
504    22.0
```

```
505    11.9
Length: 506, dtype: float64
```

```
result_df = DataFrame({
    "실제집값":실제집값,
    "예측집값1":result1,
    "예측집값2":result2,
    "예측집값3":result3
})
result_df
```

	실제집값	예측집값1	예측집값2	예측집값3
0	24.0	30.003843	24.127287	24.0
1	21.6	25.025562	21.962235	21.6
2	34.7	30.567597	21.962242	34.7
3	33.4	28.607036	21.960515	33.4
4	36.2	27.943524	21.948050	36.2
...	...	...	...	...
501	22.4	23.533341	21.950232	22.4
502	20.6	22.375719	21.956131	20.6
503	23.9	27.627426	21.950867	23.9
504	22.0	26.127967	21.934273	22.0
505	11.9	22.344212	21.955404	11.9

506 rows × 4 columns

```
plt.rcParams["figure.figsize"] = (20,

fig, (ax1, ax2, ax3) = plt.subplots(3,

sb.lineplot(data=result_df.filter(['실
sb.lineplot(data=result_df.filter(['실
sb.lineplot(data=result_df.filter(['실
```

다중선형회귀

#01. 작업 준비

1) 패키지 참조

2) 데이터 가져오기

#02. 파이썬의 ols 객체로 분석 (맛보기)

#03. 모듈화 한 기능을 활용

1) 모든 변수 사용하기

모든 독립변수의 이름을 리스트로 생성

분석 수행

분석 결과 표 확인

VIF가 10 이상인 값을 제외하고 다시 분석

변수를 다시 선정하여 진행

#04. 결과 비교하기

```
plt.show()
plt.close()
```

```
c:\Users\leekh\AppData\Local\Programs\
  fig.canvas.print_figure(bytes_io, **
c:\Users\leekh\AppData\Local\Programs\
  fig.canvas.print_figure(bytes_io, **
c:\Users\leekh\AppData\Local\Programs\
  fig.canvas.print_figure(bytes_io, **
c:\Users\leekh\AppData\Local\Programs\
  fig.canvas.print_figure(bytes_io, **
c:\Users\leekh\AppData\Local\Programs\
  fig.canvas.print_figure(bytes_io, **
c:\Users\leekh\AppData\Local\Programs\
  fig.canvas.print_figure(bytes_io, **
```