4

Multicllinearity

Simple can be harder than complex: You have to work hard to get your thinking clean to make it simple. But it's worth it in the end because once you get there, you can move mountains.

–Steve Jobs

개요

개념

다중 회귀모형($Y_i = a + b_1X_{i1} + b_2X_{i2} + \ldots + b_pX_{ip} + e_i$)는 예측변수와 목표변수간의 관계에 대한 분리하여(isolate) 유의성 검정(F-검정)과 각 예측변수의 유의성(t-검정)에 가장 큰 관심이 있다.

- 서로 다른 예측변수의 상대적 중요 정도는 무엇인가?(표준화 회귀계수)
- 목표변수에 대한 예측변수의 설명력의 크기는 얼마인가? (OLS 추정 회귀계수) - 예측변수(X_k) 한 단위 증감 시 목표변수는 b_k만큼 변동한다.
- 이런 해석은 다른 예측변수들이 고정되었다는 가정이 필요하다.

즉, 예측변수가 서로 독립(상관계수 0)이면 예측변수의 회귀계수(b_k , 상대적 효과, 설명력, marginal effect)에 의해 위의 질문에 답이 가능하다.

그러나 현실에서 예측변수가 완벽하게 독립인 경우는 없다. 예측변수들 간 상관계수가 낮으면(유의하지 않으면) 여전히 회귀계수 해석이 적절하다.

그러나 만약 예측변수간에 상관관계가 높으면(유의하면) OLS 회귀계수 추정과 검정(추정 분산이 변함)이 쓸모가 없게 되는데 이를 다중공선성(Multicollinearity) 문제라 한다.

- 예측변수 간의 상관관계가 높다는 것은 (변수의 유사성이 높다는 것이고) 목표변수를 설명하는 부분이 겹친다는 것이다.
- 예측변수가 목표변수를 잘 설명한다는 것은 목표변수 값의 변동 방향을 매우 잘 예측(변동 방향의 일치, 물론 회귀계수가 음이면 역방향)한다는 것을 의미한다.
- 그러므로 상관관계가 높은 예측변수들 간에는 목표변수 변동 설명 부분이 겹친다.

구조적 structural 다중공선성 : 설정된 회귀모형이 예측변수의 다항식 항을 포함한 경우 x^2, x^3, …

데이터 다중공선성 : 예측변수 간의 높은 상관 관계로 인한 발생

다중공선성 문제는 무엇인가?

회귀모형 $\underline{y} = X\underline{b} + \underline{e}$의 $\underline{b}$의 OLS 추정치는 $\hat{\underline{b}} = (X'X)^{-1}X'\underline{y}$이다. 설명변수($X$) 간의 상관관계가 높으면 (예: 두 설명변수의 상관관계가 높으면 (데이터 행렬의 한 열(변수)이 다른 열(변수와 상관계수가 큰 변수)로 표현되므로 $det|X'X| \approx 0$ 되고 $(X'X)^{-1} = \dfrac{adj(X'X)}{det|X'X|}$의 값이 매우 커진다.

- 이로 인하여, OLS 추정치 $\hat{\underline{b}} = (X'X)^{-1}X'\underline{y}$이므로 추정값이 불안해진다. - 다중공선성 문제를 일으키지 않는 예측변수의 회귀계수에 비해 변동이 커진다. - 회귀모형의 작은 변화에도 회귀계수의 추정치는 매우 민감하게 반응한다.
- 그리고 OLS 추정의 추정분산($s^2(\hat{\underline{b}}) = MSE(X'X)^{-1}$) 또한 매우 커져 회귀계수의 부호까지 바뀌는 문제가 발생한다.
- 회귀계수 추정 오차가 커지면 추정 정확도가 떨어지고 더 이상 회귀계수 유의성을 판단하는 유의확률(p-값)에 대한 신뢰성도 낮아진다. - 즉, 회귀계수 유의성 검정의 신뢰성 저하, -> 추정 회귀모형 신뢰성 낮아짐

과연 반드시 해결해야 하나?

다음의 경우가 연구목적이라면 굳이 해결할 필요는 없다.

예측(prediction)이 주목적인 경우 ***

다중공선성은 회귀계수 추정오차에 영향을 주지만 모형의 정확, 예측에는 영향을 주지 않는다. 오히려 예측력을 높이는 효과가 있음. 결정계수(Determination Coefficient)가 높아짐

결론적으로 빅데이터의 예측모형 개념에서는 다중공선성 문제를 진단할 필용 없음

그리고 모형 내 예측변수 회귀계수 부호가 (목표변수) 상관계수 부호와 동일하다면 진단하지 않아도 된다.

통제변수가 다중공선성 문제를 일으키는 경우

통제변수를 있는 모형의 경우 실험변수에만 다중공선성 문제가 없다면 해결할 필요 없음

The fact that some or all predictor variables are correlated among themselves does not, in general, inhibit **our ability to obtain a good fit** nor does it tend to affect **inferences about mean responses or predictions of new observations**. —Applied Linear Statistical Models, p289, 4th Edition.

Baron, R. M. and Kenny, D. A. (1986) "The Moderator(조절)-**Mediator(매개)** Variable"

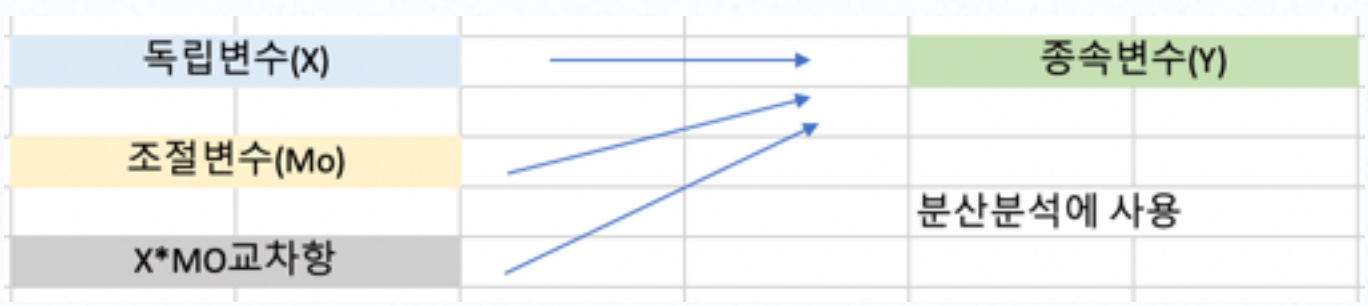
매개효과 Meditator

(순서1) $y = a + b_{11}X + e$: b_{11} 유의해야 한다.

(순서2) $y = a + b_{21}Me + e$: b_{21} 유의해야 한다.

(순서3) $y = a + b_{31}X + b_{32}Me + e$: b_{32} (매개효과) 유의하고 b_{31}은 b_{11}보다 작아지고 유의하지 않을 수 있음

조절효과 Moderate



$y = a + b_1X + b_2MO + b_3X * MO + e$

- b_3는 반드시 유의해야 한다.

맛보기

예제데이터

3장 변수 선택 결과를 유의한 예측변수 11개를 활용하여 다중공선성 진단을 실시하였다.

```
smsa_select=smsa_clean[['jan_temp', 'july_temp', 'rainfall', 'mortality_index',
      'education', 'white_color_ratio', 'person_houshold', 'household_income',
      'S02Pot', 'southern', 'pop_density_log','non_white_ratio_sqrt', 'NOxPot_log']]
smsa_select.columns
```

```
Index(['jan_temp', 'july_temp', 'rainfall', 'mortality_index', 'education',
       'white_color_ratio', 'person_houshold', 'household_income', 'S02Pot',
       'southern', 'pop_density_log', 'non_white_ratio_sqrt', 'NOxPot_log'],
```

```
smsa_cor=smsa_select.corr()
smsa_cor['mortality_index'].sort_values(ascending=False)
```

```
mortality_index          1.000000
non_white_ratio_sqrt     0.628984
rainfall                 0.433114
S02Pot                   0.419118
person_houshold          0.368016
july_temp                0.321828
NOxPot_log               0.279829
pop_density_log          0.275985
jan_temp                -0.015952
household_income        -0.283297
white_color_ratio       -0.289346
education               -0.508087
```

예제 데이터 활용 맛보기

다중공선성 문제 맛보기 위하여

- 목표변수 사망율지수(mortality_index)와 상관관계 가장 높은(가장 잘 설명하는) 비백인비율_제곱근과 높은 상관관계 예측변수 '7월기온'(0.58) 선택하였음
- 비백인비율_제곱근 상관관계가 매우 작으나 목표변수와는 상관관계 높은 변수 household-income 가구소득 선택

```
non_white_ratio_sqrt    1.000000
mortality_index         0.628984
july_temp               0.584860
jan_temp                0.456895
person_houshold         0.321250
population_log          0.259143
```

```
smsa_cor=smsa_select.corr()
smsa_cor['non_white_ratio_sqrt'].sort_values(ascending=False)
```

```
non_white_ratio_sqrt    1.000000
mortality_index         0.628984
july_temp               0.584860
jan_temp                0.456895
person_houshold         0.321250
rainfall                0.239187
NOxPot_log              0.223220
S02Pot                  0.202570
pop_density_log         0.047611
white_color_ratio      -0.047115
household_income       -0.057993
education              -0.160263
```

비백인 비율_제곱근 -> 사망률지수 단순 모형

```
import statsmodels.api as sm
X=sm.add_constant(smsa_select['non_white_ratio_sqrt'])
y=smsa_select['mortality_index']
fit=sm.OLS(y, X).fit()
print(fit.summary())
```

[결정계수 39.6%]

- 비백인비율_제곱근 1단위 변화 시 30 사망률 높아짐

	coef	std err	t	P>|t|	[0.025	0.975]
const	842.9998	17.289	48.758	0.000	808.378	877.621
non_white_ratio_sqrt	30.6452	5.017	6.108	0.000	20.599	40.692

비백인 비율_제곱근, 7월기온(상관관계 높음) -> 사망률지수 단순 모형

```
X=sm.add_constant(smsa_select[['non_white_ratio_sqrt','july_temp']])
```

[결정계수 39.9%]

- 7월 기온 유의하지도 않음 : 비백인비율_제곱근 변수와 상관관계가 높아 사망률 지수를 설명하는 부분이 겹쳐 유의하지 않게 되었음
- 결정계수 증가도 매우 미미하고 비백인 비율의 추청오차(std err)도 5.017->6.223으로 높아짐 - 다중공선성 문제로 인하여 추정이 왜곡됨

	coef	std err	t	P>|t|	[0.025	0.975]
const	907.2381	118.545	7.653	0.000	669.765	1144.712
non_white_ratio_sqrt	32.6392	6.223	5.245	0.000	20.172	45.106
july_temp	-0.9492	1.733	-0.548	0.586	-4.420	2.522

비백인 비율_제곱근, 가구소득(상관관계 낮음) -> 사망률지수 단순 모형

```
X=sm.add_constant(smsa_select[['non_white_ratio_sqrt','household_income']])
```

[결정계수 45.7%]

- 목표변수, 사망률지수와 상관 관계 낮지만(설명력 부족) 비백인 비율이 설명하고 남은 부분에 대한 설명력은 충분하여 유의하다.
- 결정계수 증가도 6% 증가하였고 비백인 비율의 추청오차(std err)도 5.017->4.807로 낮아짐

	coef	std err	t	P>|t|	[0.025	0.975]
const	960.1407	49.510	19.393	0.000	860.960	1059.321
non_white_ratio_sqrt	29.9455	4.807	6.230	0.000	20.316	39.575
household_income	-0.0035	0.001	-2.510	0.015	-0.006	-0.001

다중공선성

sm.OLS 분석결과에는 다중공선성 진단하는 상태지수 결과를 보여준다. 출력결과 마지막 라인에 다음이 출력되면 다중공선성 문제가 발생하였다는 것임

```
[2] The condition number is large, 1.38e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

(비백인비율, 7월기온)

```
[160] 1 fit.condition_number
1380.1125871077602
```

(비백인비율, 가구소득) 조차 다중공선성 있는 것으로 진단됨

```
[158] 1 fit.condition_number
245.52710101003868
```

진단도구

진단도구

산점도_상관계수

산점도 행렬이나 상관계수를 계산하여 상관 관계가 높은 설명변수들을 판단하고 다중공선성 문제가 일어날 것이라는 예상을 한다. 회귀분석의 시작은 산점도 행렬과 상관계수 행렬이다. 상관계수 행렬은 종속변수에 영향을 주는 설명변수를 사전에 판단하고 설명변수 간 높은 상관관계로 인해 발생하는 다중공선성을 사전 진단할 수 있음

- 간단히 진단할 수 있으나 다중공선성 유의성을 검정할 수 없고 두 예측변수만의 쌍체(pairwise) 진단만 가능하다.

분산팽창지수 Variance Inflation Index $VIF_k = \dfrac{1}{1 - R_k^2}$

k번째 예측변수를 종속변수(Y)로 하고 나머지 예측변수를 설명변수(X's)로 하여 회귀분석 한 후 결정계수를 구한다. 결정계수 값이 R_k^2이다.

- 결정계수 R_k^2값이 크다는 것은 X_k가 다른 예측변수들에 의해 충분히 설명되고 있다는 것이고 이는 바로 다중공선성 문제가 발생할 수 있다는 증거이다.
- 다수의 예측변수들에 의해 설명되는 정도를 나타내지만 상관계수와 같이 두 개의(쌍체) 예측변수에 의한 진단에는 적절하지 못함

일반적으로 VIF 진단 기준값은 상이함

- 10 이상: (Hair, J. F. Jr., Anderson, R. E., Tatham, R. L. & Black, W. C. (1995). Multivariate Data Analysis (3rd ed). New York: Macmillan.)
- 5이상 : Ringle, Christian M., Wende, Sven, & Becker, Jan-Michael. (2015).
- 빅데이터에서는 3, 2.5를 사용하기도 함.

상태지수 Condition Index $CI = \sqrt{\frac{\lambda_{max}}{\lambda_k}}$

- 예측변수의 공분산(상관계수)행렬을 이용하여 고유값 eigen value(λ_k) 를 구한다.
- 이에 대응하는 Norm 1인 고유벡터가 주성분 부하 loading 이다.
- 고유값은 원변수(설명변수)의 선 형결합에 의해 만들어진 주성분 변수의 원변수 변동에 대한 설명력이다. 그러므로 고유값가 크다는 것(상태지수 값이 큰 값) 주성분의 원 변수 변동에 설명력이 크다는 것을 의미한다.
- 교유벡터 내의 부하크기가 상대적으로 큰 변수들 간에는 상관관계가 높다는 것을 의미한다.
- 상태지수가 10이상(일반적으로 30이상)인 부하벡터에서 부하계수가 상대적으로 큰 변수들에 의해 다중공선성 문제가 발생한다고 진단할 수 있다.

권장 진단 방법

- VIF 통계량 3 이상, 최종적으로 유의하다고 판단된 설명변수의 피어슨 상관계수 활용
- 만약 최종 모형에서 추정된 해당 설명변수 회귀계수 부호와 종속변수와 해당 설명변수의 피어슨 상관계수 부호가 동일하다면, 다중공선성 문제를 해결하지 않아도 됨.

해결방안

구조적 다중공선성 문제 해결 - 예측변수 centering (중심화)

설명변수의 다항식으로 인한 다중공선성은 평균 이동으로 해결 가능함

표준화는 데이터의 평균과 표준편차를 활용하여 단위를 없앤 개념임 : $z_i = \frac{x_i - \bar{x}}{s(x)}$

중심화는 데이터의 평균을 0으로 이동한 개념임, 산포는 그대로 : $c_i = x_i - \bar{x}$

문제변수 제거

- 다중공선성 문제가 되는 설명변수를 제외한다.
- 여러개가 존재하는 경우 목표변수와 상관관계가 낮은 예측변수를 제외한다.

주성분 분석 : http://203.247.53.31/Stat_Notes/adv_stat/MDA/MDA_PCA.pdf

예측변수를 (서로 독립인)주성분 변수로 변환하여 사용한다. $PC = LX$ (L=부하행렬)

주성분 분석(PCA, Principal Component Analysis)은 다음 원칙에 의해 원변수의 선형 결합인 주성분(principal components)을 얻는다.

- 주성분 변수 간에는 서로 상관 관계가 전혀 없다. (독립이다)
- 첫번째 주성분은 변수들의 변동을(분산, 이를 변수가 가진 정보로 표현) 가장 많이 설명하고 계속 구해지는 2, 3, …번째 주성분은 자료의 나머지 정보들을 설명하고 크기는 점점 줄어 든다.
- 주성분 변수(Principal Component)는 원변수(회귀모형의 예측변수 X's)의 선형결합이며 서로는 독립이다.
- 주성분 변수는 서로 독립이므로 주성분 변수를 설명변수로 사용한다면 다중공선성 문제가 발생하지 않는다.

능형 회귀분석 Ridge Regression

다중공선성은 회귀계수의 추정오차를 증가시키므로 불편성(OLS는 불편 추정량이다)을 포기하는 대신 MSE(Mean Square of Error 평균오차자승합)를 최소화 하는 편기(biased) 추정량을 구하는 추정 방법을 사용함으로써 다중공성선 문제를 해결하는데 이를 능형 회귀분석((Ridge Regression)이라 한다.

- 편기 추정량 $\underline{\hat{b}}_* = (X'X + cI)X'\underline{y}$의 $MSE(\underline{\hat{b}}_*) = E(\underline{\hat{b}}_* - \underline{b})^2$를 최소화 하는 $0 < c < 1$를 구하면 이를 능형 추정량이라 한다.
- 상수 c와 VIF 산점도를 이용하여 각 예측변수의 추정 회귀계수 값이 안정화 되는 c 값을 최적값으로 설정한다. 다소 주관적이다.

파이썬 코드

파이썬 코드

VIF 계산하기

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
vif = pd.DataFrame()
X=smsa_select.drop(columns=['mortality_index','southern'])
vif["VIF Factor"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
vif.index=X.columns; print(vif)
```

- VIF 10 이상 기준 다중공선성 문제 발생 예측 변인 : SO2Pot 제외하고는 모든 예측변수가 가중공선성 문제 발생

	VIF Factor
jan_temp	27.750201
july_temp	616.722113
rainfall	23.089294
education	287.279709
white_color_ratio	136.968539
person_houshold	370.326960
household_income	88.121416
S02Pot	4.801501
pop_density_log	516.930372
non_white_ratio_sqrt	16.171589
NOxPot_log	19.282673

VIF 자동 실행, VIF 10 이상인 다중공선성 문제를 일으키는 예측변수 자동 제외

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
def calculate_vif_(X, thresh=10.0):
    variables = list(range(X.shape[1]))
    dropped = True
    while dropped:
        dropped = False
        vif = [variance_inflation_factor(X.iloc[:, variables].values, ix)
               for ix in range(X.iloc[:, variables].shape[1])]
        maxloc = vif.index(max(vif))
        if max(vif) > thresh:
            print('dropping \'' + X.iloc[:, variables].columns[maxloc] +
                  '\' at index: ' + str(maxloc))
            del variables[maxloc]
            dropped = True
    print('Remaining variables:')
    print(X.columns[variables])
    return X.iloc[:, variables]

X0=calculate_vif_(X)
```

```
dropping 'july_temp' at index: 1
dropping 'pop_density_log' at index: 7
dropping 'education' at index: 2
dropping 'person_houshold' at index: 3
dropping 'white_color_ratio' at index: 2
dropping 'household_income' at index: 2
dropping 'jan_temp' at index: 0
Remaining variables:
Index(['rainfall', 'S02Pot', 'non_white_ratio_sqrt', 'NOxPot_log']
```

최종모형

```
import statsmodels.api as sm
X=sm.add_constant(X0)
y=smsa_select['mortality_index']
fit=sm.OLS(y, X).fit(); print(fit.summary())
```

- 다중공선성 문제가 해결되었고 개별 예측변수의 회귀계수 부호와 상관계수의 부호가 일치한다.
- NOxPot_로그는 유의하지 않다.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        mortality_index   R-squared:                       0.620
Model:                            OLS   Adj. R-squared:                  0.592
Method:                 Least Squares   F-statistic:                     22.03
Date:                Wed, 22 Apr 2020   Prob (F-statistic):           8.01e-11
Time:                        23:15:57   Log-Likelihood:                -298.57
No. Observations:                  59   AIC:                             607.1
Df Residuals:                      54   BIC:                             617.5
Df Model:                           4
Covariance Type:            nonrobust
==================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
const                749.9201     27.390     27.380      0.000     695.007     804.833
rainfall               2.3070      0.536      4.300      0.000       1.231       3.383
S02Pot                 0.2734      0.115      2.381      0.021       0.043       0.504
non_white_ratio_sqrt  21.1766      4.500      4.706      0.000      12.154      30.199
NOxPot_log             8.3901      6.853      1.224      0.226      -5.350      22.130
==============================================================================
```

X=sm.add_constant(X0.drop(columns=['NOxPot_log']))

```
R-squared:                       0.609
```

```
==================================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
const                771.5640     21.014     36.716      0.000     729.451     813.677
rainfall               2.0005      0.477      4.197      0.000       1.045       2.956
S02Pot                 0.3669      0.086      4.263      0.000       0.194       0.539
non_white_ratio_sqrt  22.6356      4.359      5.193      0.000      13.900      31.371
```

상태지수 계산하기

```
import numpy as np
X=smsa_select.drop(columns=['mortality_index','southern'])
xs=np.array(X)
corr=np.corrcoef(xs, rowvar=0) #correlation matrix
w, v = np.linalg.eig(corr) #eigen values, eigen vectors
```

상태지수가 30을 넘는 것은 없으므로 상태지수만으로 파악이 불가능하다.

```
w # 고유값 출력 (원변수 개수 13개 만큼의 고유값 출력)
```

```
array([2.9071272 , 2.164814  , 1.92085147, 1.1298768 , 0.82616978,
       0.13451137, 0.16033665, 0.5569677 , 0.34140549, 0.43233538,
       0.42560416])
```

```
np.sqrt(w.max()/w)  #condition index
```

```
array([1.        , 1.15883532, 1.23022664, 1.60404472, 1.87584679,
       4.64892473, 4.25809762, 2.28463598, 2.91807715, 2.5931141 ,
       2.61353961])
```

상태지수 활용 다중공선성 문제 파악 방법을 알아보기 위하여 상태지수가 가장 큰 6번째(4.64) 고유벡터를 출력해 보자.

1월기온(0.4)와 가구소득(0.46), NOxPot_로그(-0.63) 다중공선성 문제를 야기한 것으로 판단할 수 있음
- 상태지수가 높은 고유값에 대응하는 고유벡터에서 값(부하) 크기가 상대적으로 큰 변수가 다중공선성 문제를 야기한다.

```
v[:,5] # 6번째(0부터 시작) 고유값에 대응하는 고유벡터(부하값)
```

```
array([ 0.4013106 , -0.34577973, -0.090313  ,  0.07441605,  0.05045414,
        0.25011954, -0.03885085,  0.46395098,  0.12846699,  0.0167445 ,
       -0.63799886])
```

```
X.columns
```

```
Index(['jan_temp', 'july_temp', 'humidity', 'rainfall', 'education',
       'white_color_ratio', 'person_houshold', 'household_income', 'HCPot',
       'S02Pot', 'pop_density_log', 'non_white_ratio_sqrt', 'population_log',
       'NOxPot_log'],
```

능형회귀 : L2 Regularization

- c(alpha=)의 값은 분석자가 사전에 입력해야 한다.

```
from sklearn.linear_model import Ridge
X=smsa_nor.drop(columns=['mortality_index','southern','population'])
y=smsa_nor['mortality_index']
clf=Ridge(alpha=0.7) #디폴트 = 1, c와 동일
clf.fit(X, y)
```

```
import numpy as np
np.set_printoptions(precision=4,suppress = True)
print(clf.intercept_,clf.coef_)   #절편, 회귀계수 추정값
```

```
998.1915992869888 [ -1.8708  -0.6841   1.9231 -10.3582  -1.5515 -11.5026  -0.0004   0.0537
  10.7461  29.7777  14.6699]
```

다중공선성 문제를 안고 있는 회귀분석의 결정계수 73.7%보다 낮아졌지만 다중공선성 문제를 해결하여 회귀계수의 추정 분산을 줄였다.

```
clf.score(X,y) #결정계수
```

```
0.7554053102473611
```

```
clf.predict(X) #적합값
```

```
array([ 950.58383844,  911.82089142,  908.57022341,  976.98531749,
       1047.1609193 , 1063.12038157,  919.56616077,  918.65298494,
        965.5112866 ,  937.894528  , 1006.94620231, 1014.85606671,
```

주성분분석 이용

```
from sklearn.preprocessing import StandardScaler
y=smsa_select['mortality_index'];X=smsa_select.drop(columns=['mortality_index','southern'])
X_s=StandardScaler().fit_transform(X) #scale standardization
from sklearn.decomposition import PCA
pca=PCA(n_components=5) #80% rule 0.8
df_pca=pd.DataFrame(pca.fit_transform(X_s)) #PC variables
df_pca.columns=['PC1','PC2','PC3','PC4','PC5']
df_pca.set_index(smsa_clean.index,inplace=True)
```

예측변수를 축약하여 상관계수 0인, 독립인 주성분변수 5개를 도출하였다. [주성분 변수는 원변수의 선형결합으로 선형계수(부하)에 저장되어 있다. pca.components_]

```
loadings=pd.DataFrame(pca.components_)
loadings.columns=X.columns
loadings
```

	jan_temp	july_temp	rainfall	education	white_color_ratio	person_houshold
0	0.051400	-0.360921	-0.410440	0.415928	0.307786	-0.366914
1	0.147158	0.101690	0.005273	-0.244588	-0.024394	0.048051
2	0.577792	0.432497	0.151261	0.221279	0.325389	-0.155311
3	-0.028984	0.037645	0.435880	-0.156057	0.336097	-0.458754
4	-0.513703	0.196921	0.052620	0.223594	0.518238	0.512334

```
import statsmodels.api as sm
y=smsa_clean['mortality_index']
X=sm.add_constant(df_pca) #add intercept term
fit=sm.OLS(y, X).fit();print(fit.summary())
```

2개의 주성분 변수만 유의하다.[결정계수=63%]

	coef	std err	t	P>|t|	[0.025	0.975]
const	941.1731	5.143	182.986	0.000	930.857	951.489
PC1	-17.8017	3.017	-5.901	0.000	-23.852	-11.751
PC2	26.0993	3.496	7.466	0.000	19.088	33.111
PC3	-0.5989	3.711	-0.161	0.872	-8.043	6.845
PC4	-2.0831	4.839	-0.430	0.669	-11.788	7.622
PC5	5.6955	5.659	1.006	0.319	-5.654	17.045