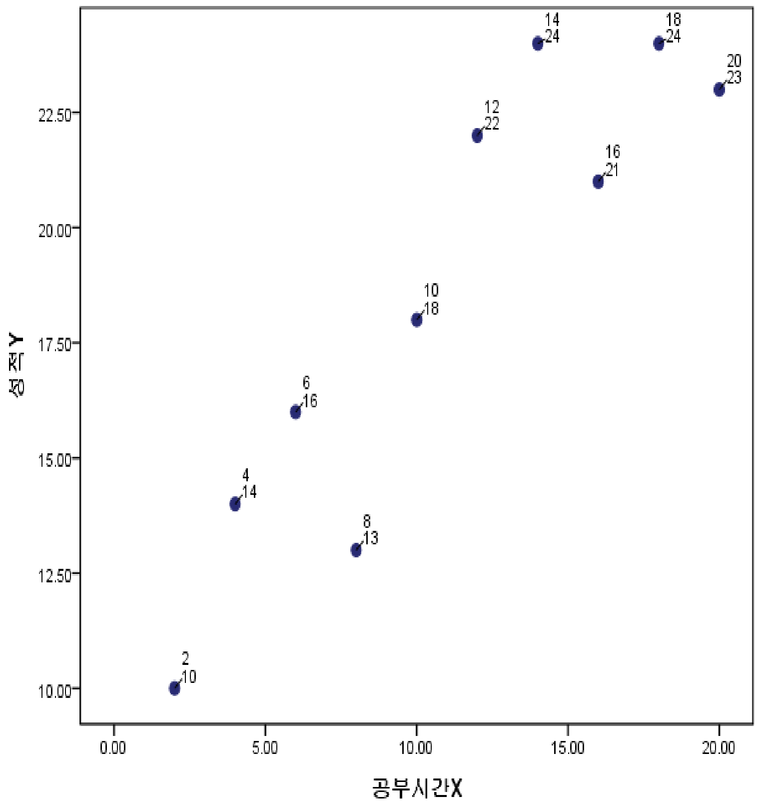
선형회귀분석의 개념과 원리

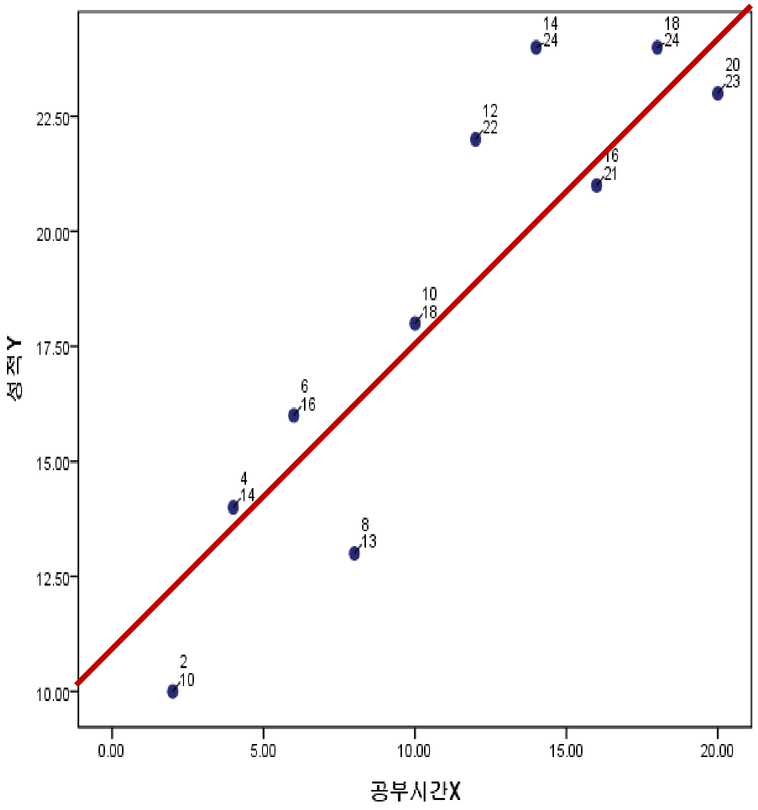
24 선형회귀분석의 개념과 원리

1-1. 선형회귀분석의 개념

가. 개념

- 특성변수와 연속형 레이블 변수 간의 중심을 지나는 **직선**(linear) 관계를 도출하는 것이 목적임





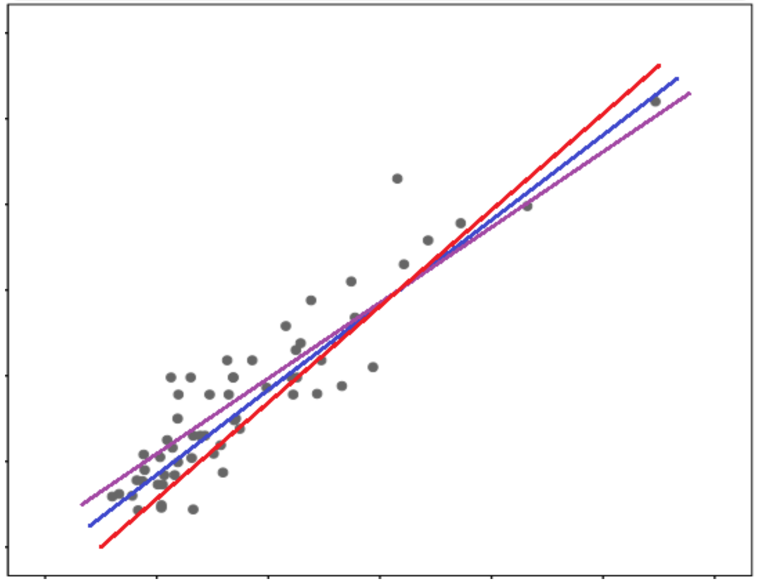
24 선형회귀분석의 개념과 원리

2-1. 선형회귀분석의 원리

가. 최적의 직선찾기

- 어디에 직선을 그을 것인가?

다양한 적합 대안선



✓ 통계적 접근법: **풉다!**

→ 최소제곱법(Least Squared Method)

$$\min\sum e_i^2 = \min(Y_i - a - bX_i)^2$$

$$\sum Y_i = na + b\sum X_i$$

$$\sum X_iY_i = a\sum X_i + b\sum X_i^2$$

$$b = \frac{n\sum X_iY_i - \sum X_i\sum Y_i}{n\sum X_i^2 - (\sum X_i)^2}$$

$$a = \overline{Y} - b\overline{X}$$

✓ 머신러닝 접근법: **대입하여 여러 번 계산한다!**

→비용함수(Cost Function)

$$H(x) - y = \frac{(H(x^{(1)}) - y^{(1)})^2 + (H(x^{(1)}) - y^{(1)})^2 + \cdots + (H(x^{(i)}) - y^{(i)})_2}{n}$$

$$Cost(W,b) = \frac{1}{n}\sum_{i=1}^{n}(H(x^{(i)}) - y^{(i)})^2$$

선형회귀분석의 개념과 원리

2-1. 선형회귀분석의 원리

가. 최적의 직선찾기

- 어디에 직선을 그을 것인가?

no	X	Y	X^2	X*Y
1	2	10	4	20
2	4	14	16	56
3	6	16	36	96
4	8	13	64	104
5	10	18	100	180
6	12	22	144	264
7	14	24	196	336
8	16	21	256	336
9	18	24	324	432
10	20	23	400	460
합계	110	185	1540	2284
평균	11.0	18.5		

최소제곱법

$$\min \sum e_i^2 = \min (Y_i - a - bX_i)^2$$

$$\hat{y} = bX_i + a$$

위의 식을 편미분하여 정규방정식을 구함

$$\sum Y_i = na + b\sum X_i$$

$$\sum X_i Y_i = a\sum X_i + b\sum X_i^2$$

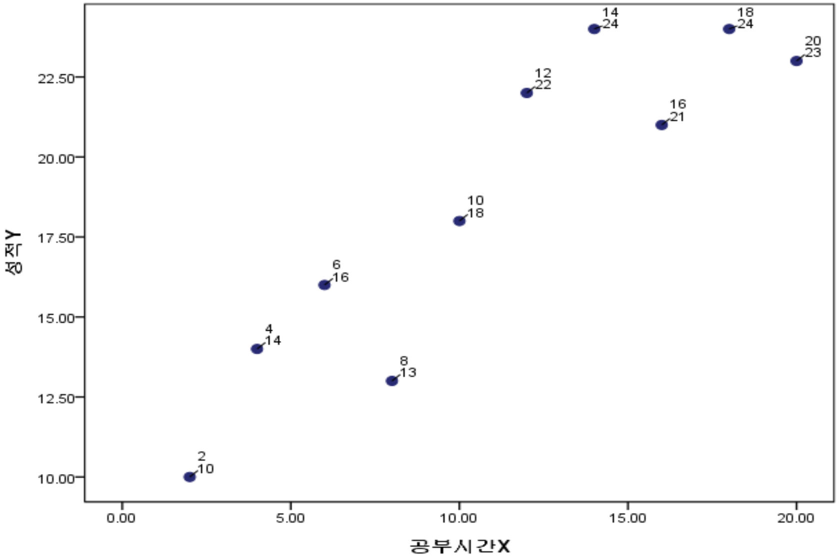
위 두 식의 연립방정식을 풀면

회귀계수(b)

$$b = \frac{n\sum X_i Y_i - \sum X_i \sum Y_i}{n\sum X_i^2 - (\sum X_i)^2}$$

절편(a)

$$a = \bar{Y} - b\bar{X}$$



24 선형회귀분석의 개념과 원리

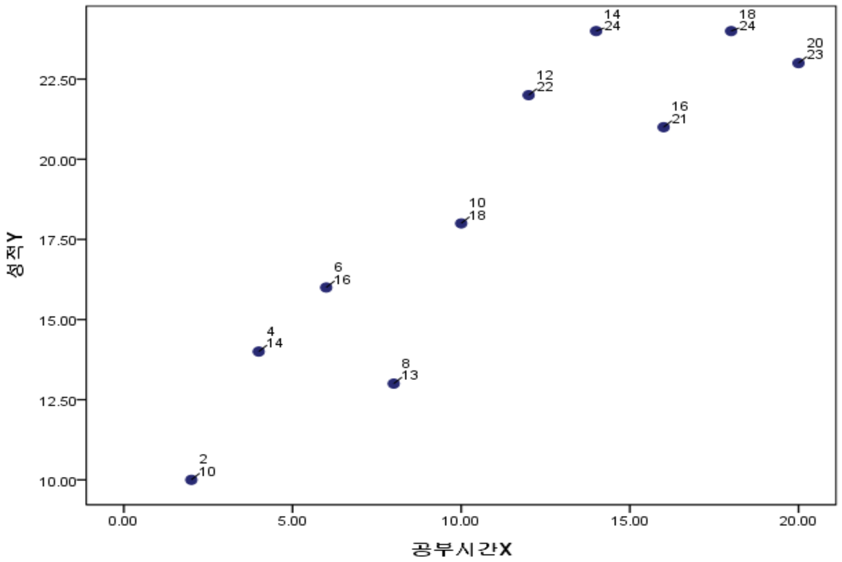
2-1. 선형회귀분석의 원리

가. 최적의 직선찾기

- 어디에 직선을 그을 것인가?

no	X	Y
1	2	10
2	4	14
3	6	16
4	8	13
5	10	18
6	12	22
7	14	24
8	16	21
9	18	24
10	20	23
합계	**110**	**185**
평균	**11.0**	**18.5**

X^2	X*Y
4	20
16	56
36	96
64	104
100	180
144	264
196	336
256	336
324	432
400	460
1540	**2284**



$$b = \frac{n\sum X_i Y_i - \sum X_i \sum Y_i}{n\sum X_i^2 - (\sum X_i)^2} = \frac{(10\times 2884)-(110\times 185)}{10\times 1540 - 110^2} = \frac{22840-20350}{15400-12100} = 0.755$$

$$a = \overline{Y} - b\overline{X} = 18.5 - 0.755 \times 11 = 10.917$$

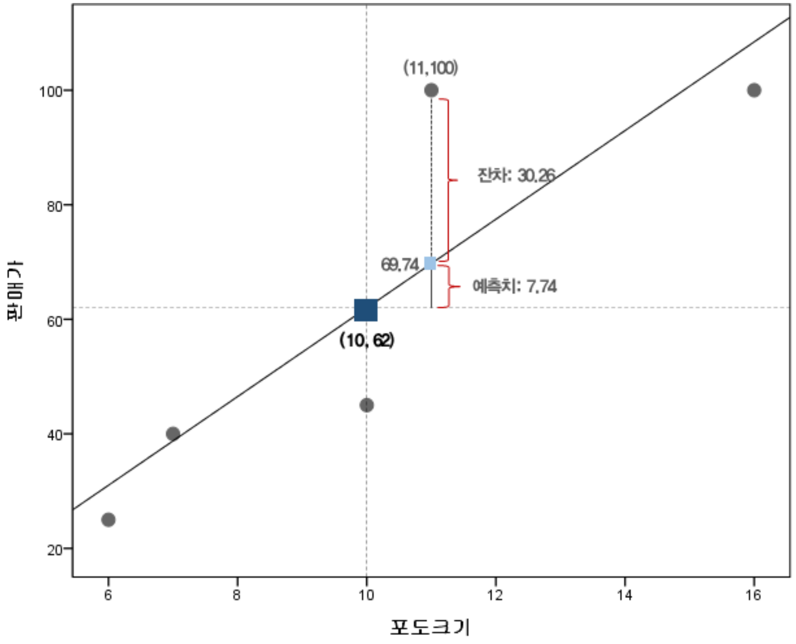
$$Y(\text{성적}) = 0.755 \times X(\text{공부시간}) + 10.195$$

24 선형회귀분석의 개념과 원리

2-1. 선형회귀분석의 원리

나. R^2과 RMSEA

- 직선과 데이터 간에 얼마나 **일치**하는가: R-square
- 직선과 데이터 간에 얼마나 **불일치**하는가: RMSEA



개념	산식
자료의 실제치	$T = R + E$
SSE	$\sum E = \sum (Y_i - \hat{Y}_i)^2$
SSR	$\sum R = \sum (\hat{Y}_i - \bar{Y})^2$
SST	$SST = SSR + SSE$
R^2	$R^2 = \frac{SSR}{SST}$ $\quad R^2 = 1 - \frac{SSE}{SST}$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

선형회귀분석 실습

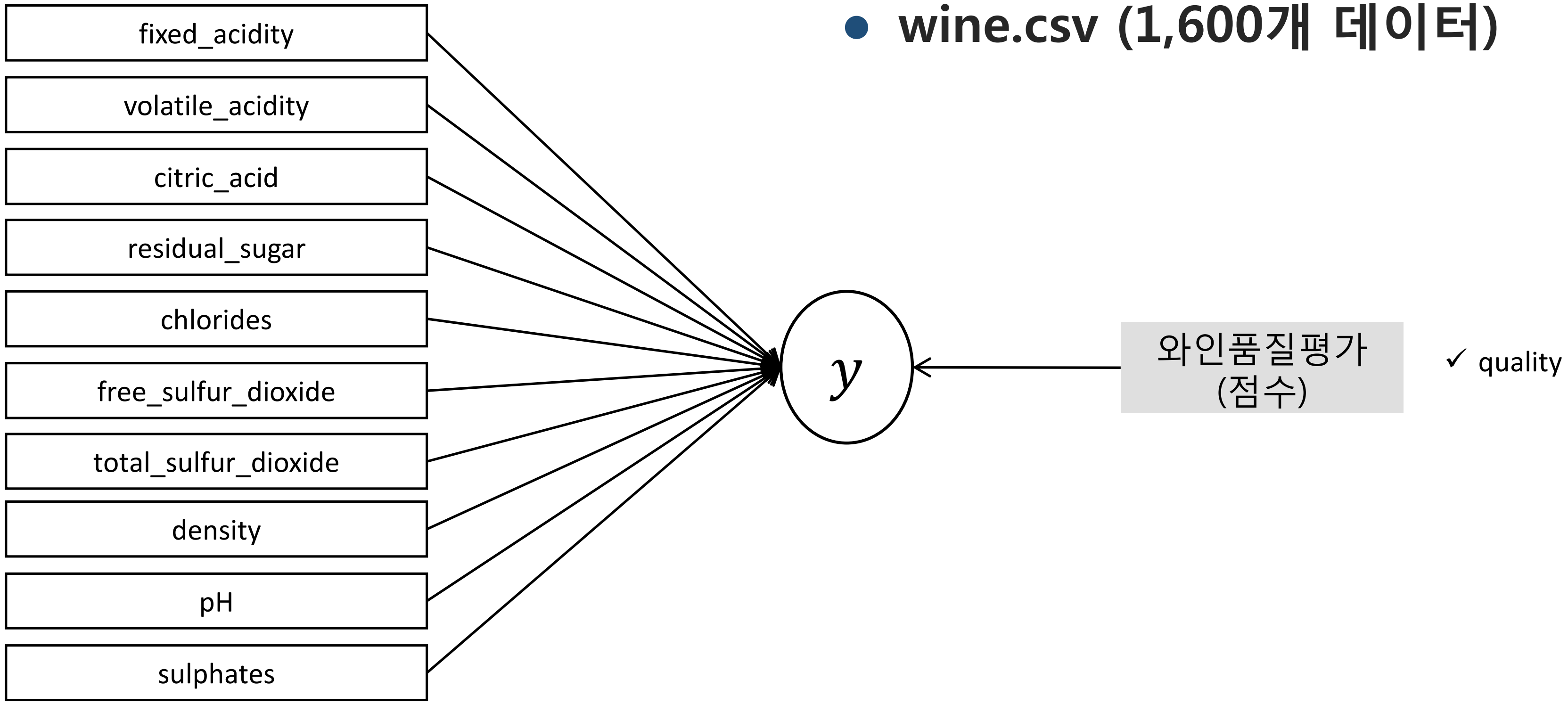
25 선형회귀분석 실습

1-1. 분석사례소개

가. 분석사례

- **분석사례**는 와인의 생산/제조 특성치가 와인의 품질평가(y)에 미치는 **예측**의 문제임

- **wine.csv (1,600개 데이터)**



25 선형회귀분석 실습

1-1. 분석사례소개

나. Linear Regression 라이브러리

- **사이킷런 Linear Regression** 의 라이브러리 구성은 아래와 같음.

sklearn.linear_model.LinearRegression

```
class sklearn.linear_model. LinearRegression (fit_intercept=True, normalize=False, copy_X=True,
n_jobs=None)
```
[source]

- **이 중 주요 매개변수는 다음과 같음**

- normalize: False, True
- copy_X: True일 경우 특성변수가 표준화될 경우 별도 X 생성, 아닐 경우 덮어 씌어짐

25 선형회귀분석 실습

2-1. 분석실습1

1. 라이브러리 및 데이터 불어오기

```
import warnings
warnings.filterwarnings("ignore")
```

```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data=pd.read_csv("wine.csv",sep=',')
```

```
data.head()
```

	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	free_sulfur_dioxi
0	7.4	0.70	0.00	1.9	0.076	11
1	7.8	0.88	0.00	2.6	0.098	25
2	7.8	0.76	0.04	2.3	0.092	15
3	11.2	0.28	0.56	1.9	0.075	17
4	7.4	0.70	0.00	1.9	0.076	11

2. 단일회귀분석

```
import statsmodels.api as sm
```

```
model = sm.OLS(data['quality'],sm.add_constant(data['alcohol'])).fit()
```

```
print (model.summary())
```

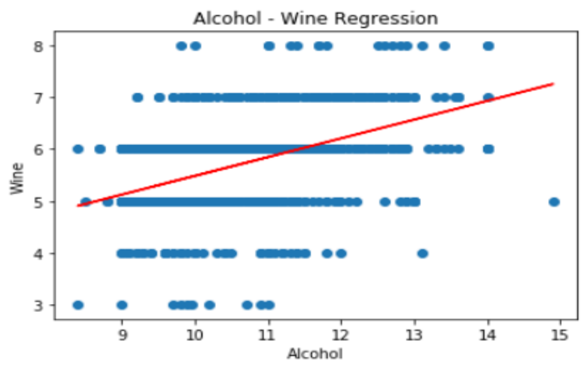
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                quality   R-squared:                       0.227
Model:                            OLS   Adj. R-squared:                  0.226
Method:                 Least Squares   F-statistic:                     468.3
Date:                Tue, 30 Oct 2018   Prob (F-statistic):           2.83e-91
Time:                        14:10:38   Log-Likelihood:                -1721.1
No. Observations:                1599   AIC:                             3446.
Df Residuals:                    1597   BIC:                             3457.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.8750      0.175     10.732      0.000       1.532       2.218
alcohol        0.3608      0.017     21.639      0.000       0.328       0.394
==============================================================================
Omnibus:                       38.501   Durbin-Watson:                   1.748
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               71.758
Skew:                          -0.154   Prob(JB):                     2.62e-16
Kurtosis:                       3.991   Cond. No.                         104.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly speci
fied.
```

```
plt.scatter(data['alcohol'], data['quality'],label = 'Actual Data')
plt.plot(data['alcohol'],model.params[0]+model.params[1]*data['alcohol'],
         c ='r',label="Regression fit")
plt.title('Alcohol - Wine Regression')
plt.xlabel('Alcohol')
plt.ylabel('Wine')
plt.show()
```

25 선형회귀분석 실습

2-1. 분석실습2




```python
from sklearn.model_selection import train_test_split
```

```python
X_train,X_test,y_train,y_test = train_test_split(
    data['alcohol'], data["quality"], train_size = 0.7,random_state=42)
```

```python
X_train.head()
```

```
925     11.0
363     10.2
906     11.0
426     11.4
1251     9.8
Name: alcohol, dtype: float64
```

```python
X_train = pd.DataFrame(X_train)
X_test = pd.DataFrame(X_test)
y_train = pd.DataFrame(y_train)
y_test = pd.DataFrame(y_test)
```

```python
from sklearn.linear_model import LinearRegression
```

```python
lr = LinearRegression().fit(X_train, y_train)
```

```python
print("lr.coef_: {}".format(lr.coef_))
print("lr.intercept_: {}".format(lr.intercept_))
```

```
lr.coef_: [[0.37699262]]
lr.intercept_: [1.69472474]
```

```python
print("훈련 세트 R-square: {:.2f}".format(lr.score(X_train, y_train)))
print("테스트 세트 R-square: {:.2f}".format(lr.score(X_test, y_test)))
```

```
훈련 세트 R-square: 0.24
테스트 세트 R-square: 0.19
```

3. 다중회귀분석

```python
X=data[data.columns[0:11]]
```

```python
y = data[['quality']]
```

```python
X.head()
```

	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	free_sulfur_dioxide	total_sul
0	7.4	0.70	0.00	1.9	0.076	11.0	
1	7.8	0.88	0.00	2.6	0.098	25.0	
2	7.8	0.76	0.04	2.3	0.092	15.0	
3	11.2	0.28	0.56	1.9	0.075	17.0	
4	7.4	0.70	0.00	1.9	0.076	11.0	

```python
X_train, X_test, y_train, y_test = train_test_split(X, y,train_size = 0.7,
                                                     random_state=42)
```

```python
x_train_new = sm.add_constant(X_train)
x_test_new = sm.add_constant(X_test)
```

```python
x_train_new.head()
```

	const	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	free_sulfur_dioxid
925	1.0	8.6	0.22	0.36	1.9	0.064	53.(
363	1.0	12.5	0.46	0.63	2.0	0.071	6.(
906	1.0	7.2	0.54	0.27	2.6	0.084	12.(
426	1.0	6.4	0.67	0.08	2.1	0.045	19.(
1251	1.0	7.5	0.58	0.14	2.2	0.077	27.(

25 선형회귀분석 실습

2-1. 분석실습3

```
multi_model = sm.OLS(y_train,x_train_new).fit()
print (multi_model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                quality   R-squared:                       0.361
Model:                            OLS   Adj. R-squared:                  0.355
Method:                 Least Squares   F-statistic:                     56.90
Date:                Tue, 30 Oct 2018   Prob (F-statistic):          8.34e-100
Time:                        14:10:39   Log-Likelihood:                -1103.5
No. Observations:                1119   AIC:                             2231.
Df Residuals:                    1107   BIC:                             2291.
Df Model:                          11
Covariance Type:            nonrobust
========================================================================================
                           coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------
const                   17.9626     25.237      0.712      0.477     -31.555      67.480
fixed_acidity            0.0235      0.031      0.769      0.442      -0.036       0.083
volatile_acidity        -1.0996      0.145     -7.599      0.000      -1.384      -0.816
citric_acid             -0.2479      0.177     -1.402      0.161      -0.595       0.099
residual_sugar           0.0077      0.018      0.429      0.668      -0.028       0.043
chlorides               -1.6736      0.500     -3.344      0.001      -2.656      -0.692
free_sulfur_dioxide      0.0046      0.003      1.706      0.088      -0.001       0.010
total_sulfur_dioxide    -0.0033      0.001     -3.723      0.000      -0.005      -0.002
density                -14.2396     25.750     -0.553      0.580     -64.763      36.284
pH                      -0.3192      0.227     -1.404      0.161      -0.766       0.127
sulphates                0.8128      0.135      6.007      0.000       0.547       1.078
alcohol                  0.2920      0.032      9.268      0.000       0.230       0.354
==============================================================================
Omnibus:                       29.060   Durbin-Watson:                   2.001
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               50.192
Skew:                          -0.193   Prob(JB):                     1.26e-11
Kurtosis:                       3.963   Cond. No.                     1.13e+05
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.13e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

```
y_pred = multi_model.predict(x_test_new)
y_pred_df = pd.DataFrame(y_pred)
y_pred_df.columns = ['y_pred']
pred_data = pd.DataFrame(y_pred_df['y_pred'])
y_test_new = pd.DataFrame(y_test)
```

```
pred_data['y_test'] = pd.DataFrame(y_test_new['quality'])
```

```
pred_data.head()
```

	y_pred	y_test
803	5.356763	6
124	5.090715	5
350	5.625538	6
682	5.448861	5
1326	5.744784	6

```
multi_model2 = sm.OLS(y_test,x_test_new).fit()
print (multi_model2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                quality   R-squared:                       0.372
Model:                            OLS   Adj. R-squared:                  0.357
Method:                 Least Squares   F-statistic:                     25.22
Date:                Tue, 30 Oct 2018   Prob (F-statistic):           5.60e-41
Time:                        14:10:39   Log-Likelihood:                -460.03
No. Observations:                 480   AIC:                             944.1
Df Residuals:                     468   BIC:                             994.1
Df Model:                          11
Covariance Type:            nonrobust
========================================================================================
                           coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------
const                   30.8445     39.952      0.772      0.440     -47.662     109.351
fixed_acidity            0.0159      0.050      0.315      0.753      -0.083       0.115
volatile_acidity        -1.0251      0.224     -4.570      0.000      -1.466      -0.584
citric_acid              0.0325      0.270      0.120      0.904      -0.498       0.563
residual_sugar           0.0416      0.027      1.522      0.129      -0.012       0.095
chlorides               -2.4644      0.779     -3.165      0.002      -3.994      -0.935
free_sulfur_dioxide      0.0040      0.004      1.070      0.285      -0.003       0.011
total_sulfur_dioxide    -0.0034      0.001     -2.546      0.011      -0.006      -0.001
density                -25.6932     40.816     -0.629      0.529    -105.899      54.512
pH                      -0.6864      0.360     -1.904      0.057      -1.395       0.022
sulphates                1.1899      0.216      5.521      0.000       0.766       1.613
alcohol                  0.2400      0.050      4.848      0.000       0.143       0.337
==============================================================================
Omnibus:                        0.207   Durbin-Watson:                   1.932
Prob(Omnibus):                  0.902   Jarque-Bera (JB):                0.159
Skew:                          -0.044   Prob(JB):                        0.924
Kurtosis:                       3.012   Cond. No.                     1.17e+05
==============================================================================
```

25 선형회귀분석 실습

2-1. 분석실습4

4. scikit-learn을 이용한 회귀분석

```
from sklearn.linear_model import LinearRegression
```

```
linear1=LinearRegression()
```

```
linear1.fit(X_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

```
linear1.score(X_train, y_train)
```

```
0.3611982441321648
```

```
linear1.score(X_test, y_test)
```

```
0.3513885332517386
```

```
pred_train=linear1.predict(X_train)
```

```
pred_test=linear1.predict(X_test)
```

```
linear2=LinearRegression(normalize=True)
```

```
linear2.fit(X_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=True)
```

```
linear2.score(X_train, y_train)
```

```
0.36119824413216456
```

```
linear2.score(X_test, y_test)
```

```
0.3513885332517399
```