(4-5) Analysis of Wine data

1) 데이터 세트 "winequality-red.csv"에 대해서

● 데이터 설명

주어진 데이터는, 각각 'fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar', 'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density', 'pH', 'sulphates', 'alcohol', 'quality' 12가지 feature에 해당하는 값 들이고 이 중 본 분석에서 예측하고자 하는 target feature는 'quality'이다.

● 데이터 전처리

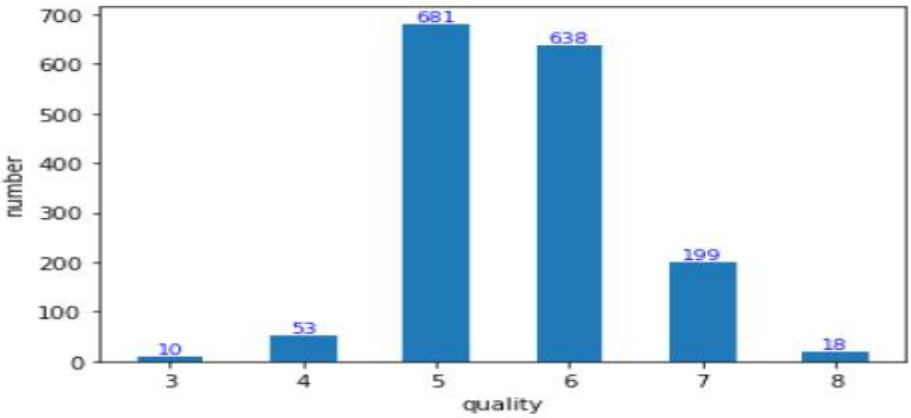
주어진 csv파일에서는 모든 정보들이 하나의 column에 담겨 있어, 각각의 feature를 하나의 column으로 할당해서 총 12개의 column에 해당하는 정보를 넣었고 'new-winequality-red.csv'라는 파일을 만들어 이 곳에 저장하였다.

```
import pandas as pd
df_red = pd.read_csv('winequality-red.csv')
before_colname = 'fixed acidity;"volatile acidity";"citric acid";"residual sugar";"chlorides";"free sulfur dioxide";"total sulfur dioxide";"density";"pH";"sulphates";"alcohol";"quality"'
column_name = before_colname.split(';')
for i in range(len(column_name)):
    column_name[i]=column_name[i].replace("'","")
    column_name[i]=column_name[i].replace('"',"")
if not df_red.columns[0] == 'x':
    df_red.rename(columns = {'fixed acidity;"volatile acidity";"citric acid";"residual sugar";"chlorides";"free sulfur dioxide";"total sulfur dioxide";"density";"pH";"sulphates";"alcohol";"quality"':'x'}, inplace = True)
df_red = df_red.x.str.split(';')
df_red = df_red.apply(lambda x: pd.Series(x))
df_red.columns = column_name
df_red.to_csv('new-winequality-red.csv')
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

1599 rows × 12 columns

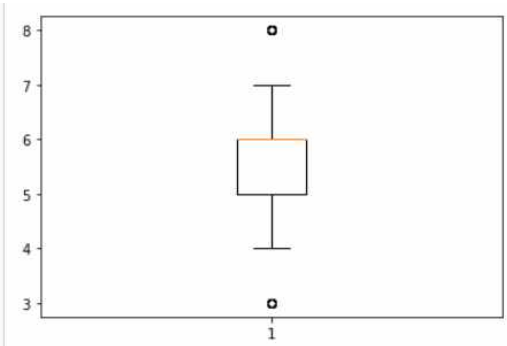
```
import matplotlib.pyplot as plt
y_axis = [0,0,0,0,0,0]
x_range = range(3,9)
for i in range(len(df_red['quality'])):
    if df_red['quality'][i] == 3:
        y_axis[0] += 1
    elif df_red['quality'][i] == 4:
        y_axis[1] += 1
    elif df_red['quality'][i] == 5:
        y_axis[2] += 1
    elif df_red['quality'][i] == 6:
        y_axis[3] += 1
    elif df_red['quality'][i] == 7:
        y_axis[4] += 1
    elif df_red['quality'][i] == 8:
        y_axis[5] += 1
    else:
        break

plt.bar(x_range, y_axis, width = 0.5)
for i, v in enumerate(x_range):
    plt.text(v, y_axis[i], y_axis[i],                    # 좌표 (x축 = v, y축 = y[0]..y[1], 표시 = y[0]..y[1])
             fontsize = 9,
             color='blue',
             horizontalalignment='center',   # horizontalalignment (left, center, right)
             verticalalignment='bottom')     # verticalalignment (top, center, bottom)
plt.xlabel('quality')
plt.ylabel('number')
plt.show()
```



quality 데이터의 분포를 막대그래프로 살펴본 결과, 3-8점사이에 분포되어있었고, 3점과 8점은 너무 적고 5,6점은 너무 많아 분포가 대칭을 이루지 않는다. 이것은 분석에 영향을 줄 것이다. quality의 통계적 세부사항은 아래와 같다.

```
import numpy as np
quality_statistic = [0,0,0,0,0,0,0,0] # count, mean, std, min, q1, q2, q3, max 순서
name = ['count', 'mean', 'std', 'min', 'q1', 'q2', 'q3', 'max']
quality_statistic[0] = len(df_red['quality'])
quality_statistic[1] = np.mean(df_red['quality'])
quality_statistic[2] = np.std(df_red['quality'])
quality_statistic[3] = df_red['quality'].quantile(0)
quality_statistic[4] = df_red['quality'].quantile(.25)
quality_statistic[5] = df_red['quality'].quantile(.5)
quality_statistic[6] = df_red['quality'].quantile(.75)
quality_statistic[7] = df_red['quality'].quantile(1)
for i in range(len(quality_statistic)):
    print(name[i] + ':', quality_statistic[i])
```

```
count: 1599
mean: 5.6360225140712945
std: 0.8073168769639486
min: 3.0
q1: 5.0
q2: 6.0
q3: 6.0
max: 8.0
```



#Distribution of quality

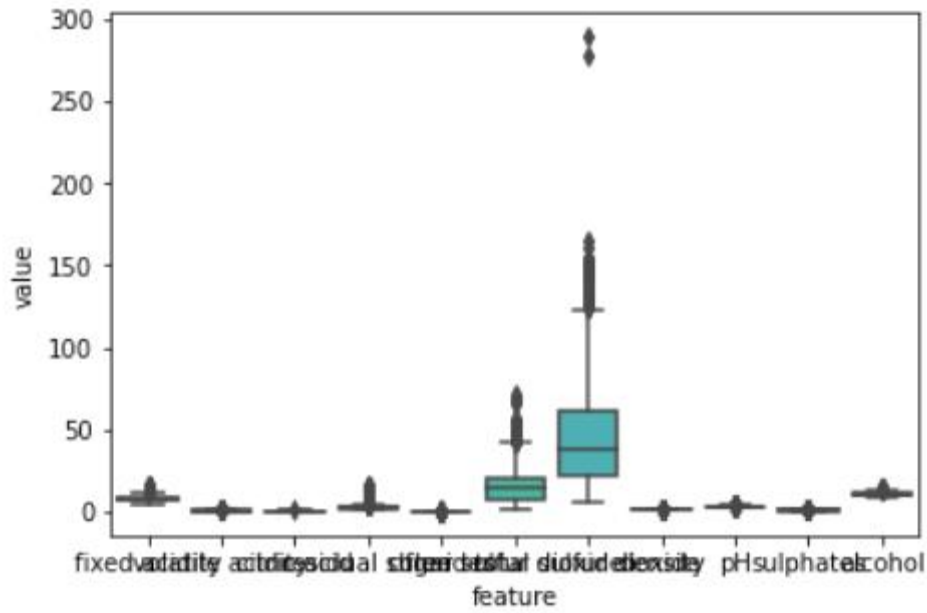
```
plt.boxplot(df_red['quality'])
plt.title('Distribution of quality')
plt.show
```

#Preprocess data

```
import seaborn as sns
col_list = ['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
        'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
        'pH', 'sulphates', 'alcohol']
```

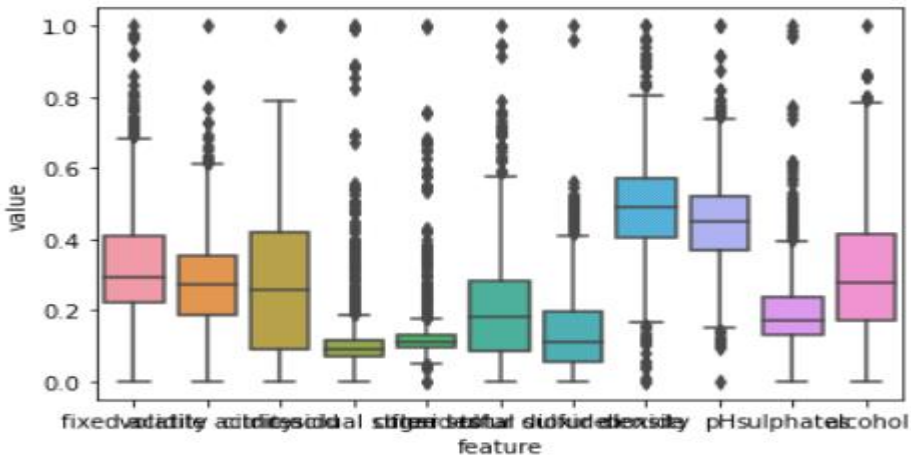
```
data = []
section = ['fixed acidity']*1599+['volatile acidity']*1599+['citric acid']*1599+['residual sugar']*1599+['chlorides']*1599+['free sulfur dioxide']*1599+['total sulfur dioxide']*1599+['density']*1599+['pH']*1599+['sulphates']*1599+['alcohol']*1599
for i in range(11):
    for j in range(1599):
        data.append(df_red.iloc[j,i])
aa = pd.DataFrame(data)
aa.insert(0,'section',section)
aa.columns = ['feature','value']
sns.boxplot(x="feature",y="value",data=aa)
plt.figure(figsize=(12,3))
plt.show()
```

각각의 featur에 대한 데이터를 그래프로 나타내면 아래와 같다. 각 feature 별로 값의 차이가 많이나므로 모든 데이터가 0-1사이의 값을 갖도록 normalize를 해준다.



#Distribution of features - After normalization

```
from sklearn.preprocessing import MinMaxScaler
df_normalized = df_red[:]
scaler = MinMaxScaler()
df_normalized[:] = scaler.fit_transform(df_normalized[:])
df_normalized['quality'] = df_red['quality']
```



아까와 같은방법으로 normalize한 데이터들이다.

● 데이터 시각화

Correlation between features

정규화한 데이터를 이용하여 피어슨 상관계수를 구해보면 다음과 같다.

```
col_list = ['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
        'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
        'pH', 'sulphates', 'alcohol', 'quality']
lst = [df_red[col_list[0]],df_red[col_list[1]],df_red[col_list[2]],df_red[col_list[3]],df_red[col_list[4]],df_red[col_list[5]],df_red[col_list[6]],df_red[col_list[7]],df_red[col_list[8]],df_red[col_list[9]],df_red[col_list[10]],df_red[col_list[11]]]
df_corr = pd.DataFrame(lst).T
corr = df_corr.corr(method='pearson')
corr
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1.000000	-0.256131	0.671703	0.114777	0.093705	-0.153794	-0.113181	0.668047	-0.682978	0.183006	-0.061668	0.124052
volatile acidity	-0.256131	1.000000	-0.552496	0.001918	0.061298	-0.010504	0.076470	0.022026	0.234937	-0.260987	-0.202288	-0.390558
citric acid	0.671703	-0.552496	1.000000	0.143577	0.203823	-0.060978	0.035533	0.364947	-0.541904	0.312770	0.109903	0.226373
residual sugar	0.114777	0.001918	0.143577	1.000000	0.055610	0.187049	0.203028	0.355283	-0.085652	0.005527	0.042075	0.013732
chlorides	0.093705	0.061298	0.203823	0.055610	1.000000	0.005562	0.047400	0.200632	-0.265026	0.371260	-0.221141	-0.128907
free sulfur dioxide	-0.153794	-0.010504	-0.060978	0.187049	0.005562	1.000000	0.667666	-0.021946	0.070377	0.051658	-0.069408	-0.050656
total sulfur dioxide	-0.113181	0.076470	0.035533	0.203028	0.047400	0.667666	1.000000	0.071269	-0.066495	0.042947	-0.205654	-0.185100
density	0.668047	0.022026	0.364947	0.355283	0.200632	-0.021946	0.071269	1.000000	-0.341699	0.148506	-0.496180	-0.174919
pH	-0.682978	0.234937	-0.541904	-0.085652	-0.265026	0.070377	-0.066495	-0.341699	1.000000	-0.196648	0.205633	-0.057731
sulphates	0.183006	-0.260987	0.312770	0.005527	0.371260	0.051658	0.042947	0.148506	-0.196648	1.000000	0.093595	0.251397
alcohol	-0.061668	-0.202288	0.109903	0.042075	-0.221141	-0.069408	-0.205654	-0.496180	0.205633	0.093595	1.000000	0.476166
quality	0.124052	-0.390558	0.226373	0.013732	-0.128907	-0.050656	-0.185100	-0.174919	-0.057731	0.251397	0.476166	1.000000

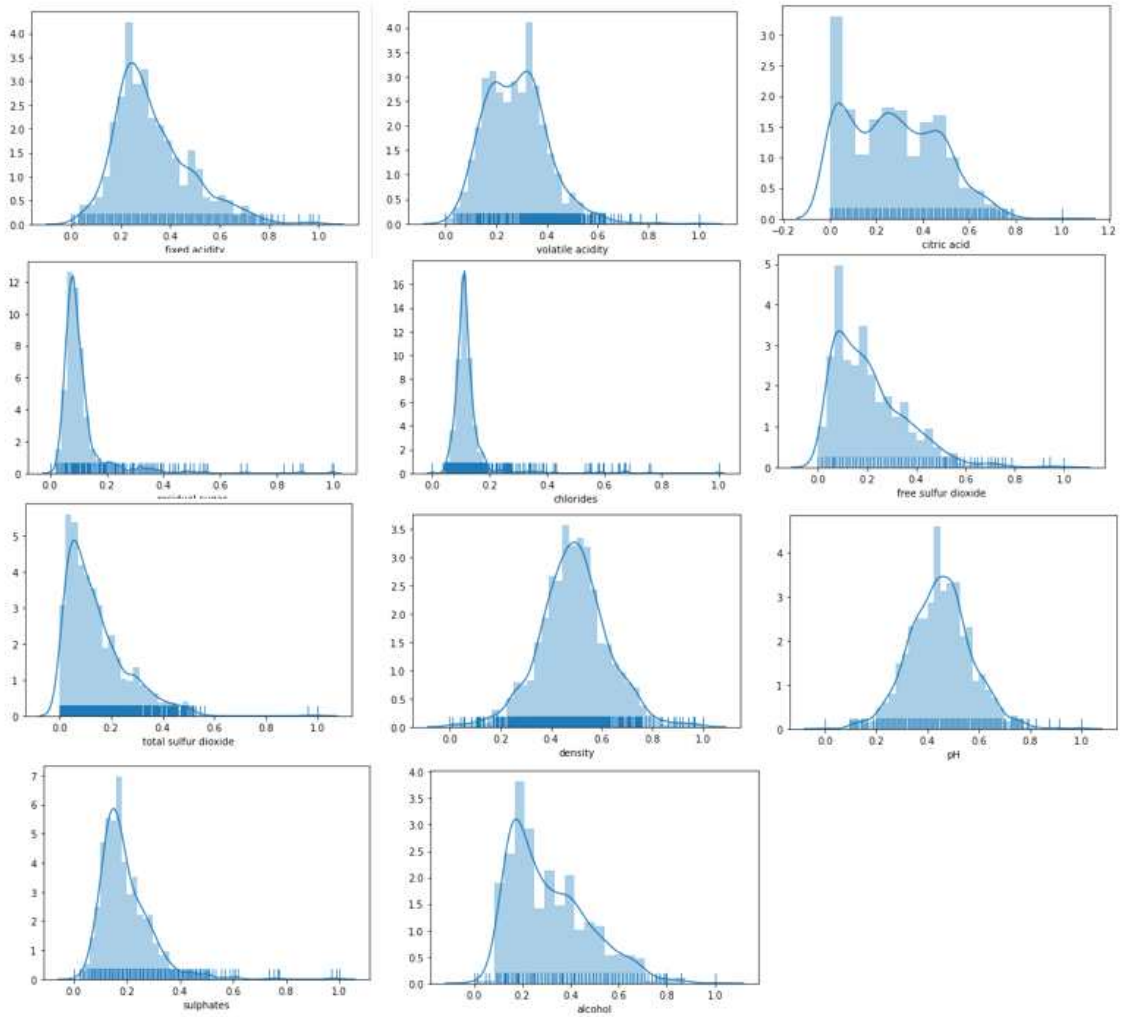
이 중 corr이 0.5가 넘는 것을 찾아보면, 다음과 같은 6가지가 존재한다.

fixed acidity & citric acid : 0.6717034347641084
fixed acidity & density : 0.6680472921189742
fixed acidity & pH : -0.6829781945685356
volatile acidity & citric acid : -0.552495684559583
citric acid & pH : -0.5419041447395134
free sulfur dioxide & total sulfur dioxide : 0.6676664504810229

#Distribution of each feature

각 독립변수의 데이터가 어떤 분포를 따르는지 그래프를 그려보았다.

```
import seaborn as sns
data = df_red['fixed acidity']
sns.distplot(data, kde=True, rug=True)
plt.show()
```

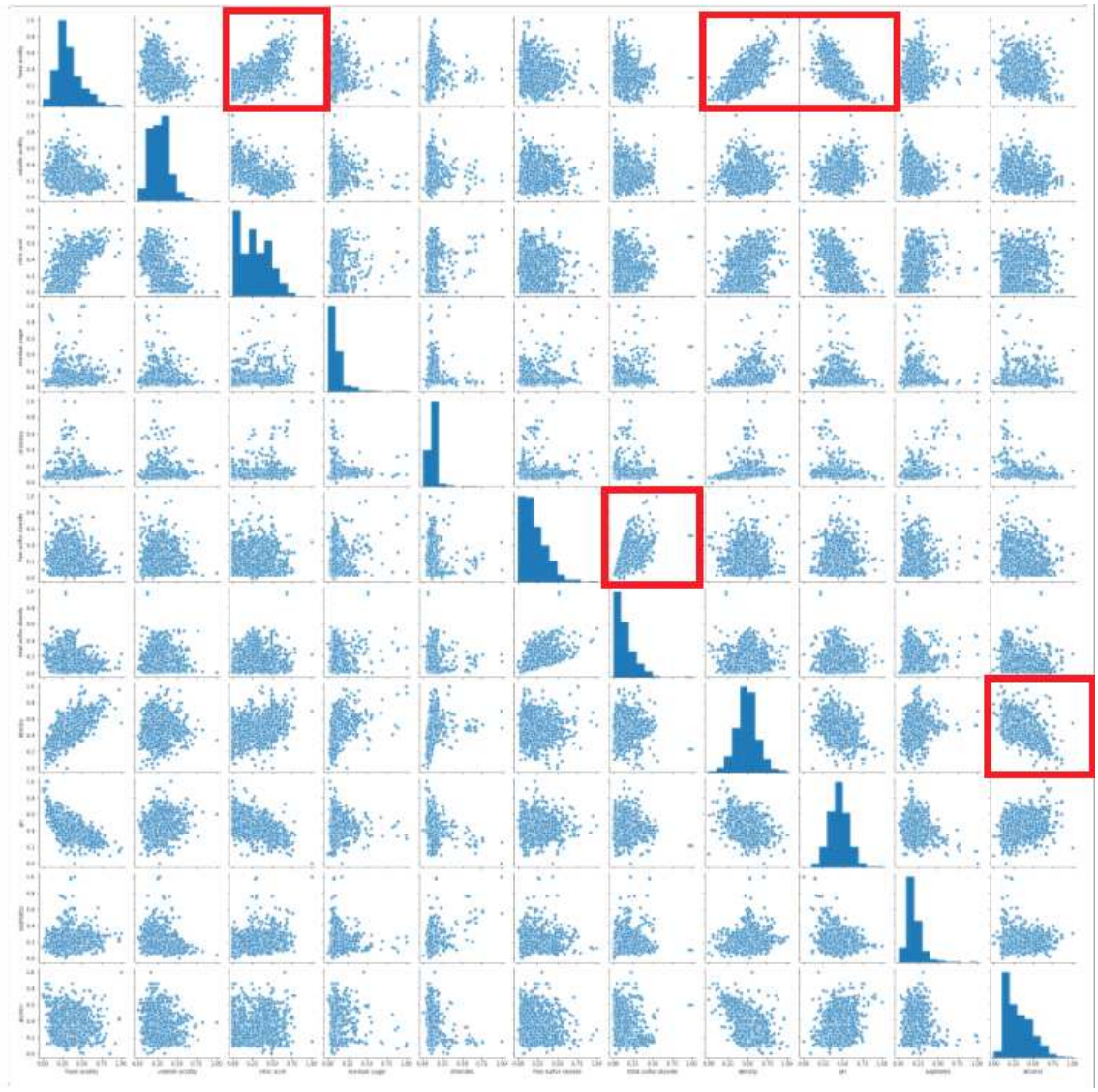


대부분의 변수가 정규분포 모양을 보였지만 citric acid는 확실하게 정규분포 모양을 띄지 않았고 이는 영향이 있을 것이라 예상된다.

#Pair plot between features

두 개의 독립변수 간 산점도를 나타내보았다.

```
sns.pairplot(df_red.iloc[:,:11])
plt.show()
```



육안으로 보았을 때, 두 독립변수간에 상관관계가 있어 보이는 그래프에 빨간 네모표시를 하였다.

● 회귀 모델 구축

#주어진 모든 독립변인 변수를 이용하여 OLS를 이용한 다중선형회귀분석을 실시였다.

```
x_data = df_red[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
        'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
        'pH', 'sulphates', 'alcohol']]
target = df_red[['quality']]

import statsmodels.api as sm
# for b0, 상수항 추가
x_data1 = sm.add_constant(x_data, has_constant = "add")

# OLS 검정
multi_model = sm.OLS(target, x_data1)
fitted_multi_model = multi_model.fit()
fitted_multi_model.summary()
```

OLS Regression Results

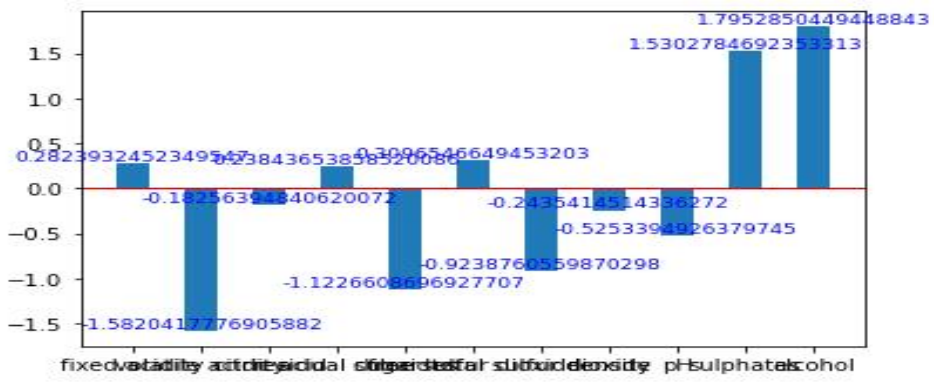
Dep. Variable:	quality	R-squared:	0.361
Model:	OLS	Adj. R-squared:	0.356
Method:	Least Squares	F-statistic:	81.35
Date:	Sat, 05 Jun 2021	Prob (F-statistic):	1.79e-145
Time:	20:38:36	Log-Likelihood:	-1569.1
No. Observations:	1599	AIC:	3162.
Df Residuals:	1587	BIC:	3227.
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	5.7126	0.151	37.848	0.000	5.416	6.009
fixed acidity	0.2824	0.293	0.963	0.336	-0.293	0.858
volatile acidity	-1.5820	0.177	-8.948	0.000	-1.929	-1.235
citric acid	-0.1826	0.147	-1.240	0.215	-0.471	0.106
residual sugar	0.2384	0.219	1.089	0.276	-0.191	0.668
chlorides	-1.1227	0.251	-4.470	0.000	-1.615	-0.630
free sulfur dioxide	0.3097	0.154	2.009	0.045	0.007	0.612
total sulfur dioxide	-0.9239	0.206	-4.480	0.000	-1.328	-0.519
density	-0.2435	0.295	-0.827	0.409	-0.821	0.334
pH	-0.5253	0.243	-2.159	0.031	-1.003	-0.048
sulphates	1.5303	0.191	8.014	0.000	1.156	1.905
alcohol	1.7953	0.172	10.429	0.000	1.458	2.133

Omnibus:	27.376	Durbin-Watson:	1.757
Prob(Omnibus):	0.000	Jarque-Bera (JB):	40.965
Skew:	-0.168	Prob(JB):	1.27e-09
Kurtosis:	3.708	Cond. No.	40.9

그 결과 R-squared는 0.361, Adj. R-squared는 0.356이 나왔고, 이 모델의 P-value가 0.05보다 작아 귀무가설을 유의수준 0.05수준에서 기각할 수 있고 따라서 회귀성이 존재한다고 생각할 수 있다.
또한, 각 독립변수에서의 p-value를 살펴보면, ‘fixed acidity’, ‘citric acid’, ‘residual sugar’, ‘density’ 변수가 0.05보다 큰 것을 확인할 수 있었다. 그리고, 각 coef를 그래프로 나타내보면,

```
y = fitted_multi_model.params[1:]
x_range = ['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
        'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
        'pH', 'sulphates', 'alcohol']
y_ = [y[0],y[1],y[2],y[3],y[4],y[5],y[6],y[7],y[8],y[9],y[10]]
plt.axhline(y=0.0, color='r', linewidth=1)
plt.bar(x_range, y_, width = 0.5)
for i, v in enumerate(x_range):
    plt.text(v, y_[i], y_[i],                    # 좌표 (x축 = v, y축 = y[0]..y[1], 표시 = y[0]..y[1])
             fontsize = 9,
             color='blue',
             horizontalalignment='center',   # horizontalalignment (left, center, right)
             verticalalignment='bottom')     # verticalalignment (top, center, bottom)
```



p-value가 0.05를 넘지 못하는 독립변수는 coef 절대값이 0.3을 넘지 못하고 나머지 변수들에 비해 가장 작은 4가지 변수임을 확인할 수 있었다.
('fixed acidity', 'citric acid', 'residual sugar', 'density')

앞의 내용에서 corr이 0.5가 넘는 독립변수 쌍은 다음과 같았다.

fixed acidity & citric acid : 0.6717034347641084
fixed acidity & density : 0.6680472921189742
fixed acidity & pH : -0.6829781945685356
volatile acidity & citric acid : -0.552495684559583
citric acid & pH : -0.5419041447395134
free sulfur dioxide & total sulfur dioxide : 0.6676664504810229

'fixed acidity', 'citric acid' 두 독립변수는 다중회귀 모델에서 베타 값도 작고 p-value도 0.05보다 높으며 두 독립변수간의 상관관계도 0.5가 넘으므로 다중공선성 문제가 생길 수도 있다고 판단되어 두 독립변수를 제거하고 다시 OLS를 이용한 다중선형회귀 분석을 실시해 보았다.

#2가지 독립변수를 제외한 후 OLS를 이용한 다중선형회귀분석을 실시.

```
 import statsmodels.api as sm
x_data = df_red[['volatile acidity','residual sugar','chlorides','free sulfur dioxide','total sulfur dioxide','density','pH','sulphates','alcohol']]
target = df_red[['quality']]

# for b0, 상수항 추가
x_data2 = sm.add_constant(x_data, has_constant = "add")

# OLS 검정
multi_model = sm.OLS(target, x_data2)
fitted_multi_model = multi_model.fit()
fitted_multi_model.summary()
```

OLS Regression Results

Dep. Variable:	quality	R-squared:	0.360
Model:	OLS	Adj. R-squared:	0.356
Method:	Least Squares	F-statistic:	99.22
Date:	Sat, 05 Jun 2021	Prob (F-statistic):	4.56e-147
Time:	21:49:56	Log-Likelihood:	-1570.1
No. Observations:	1599	AIC:	3160.
Df Residuals:	1589	BIC:	3214.
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	5.7237	0.141	40.625	0.000	5.447	6.000
volatile acidity	-1.4787	0.147	-10.035	0.000	-1.768	-1.190
residual sugar	0.1680	0.197	0.855	0.393	-0.218	0.553
chlorides	-1.2274	0.239	-5.133	0.000	-1.696	-0.758
free sulfur dioxide	0.3461	0.152	2.279	0.023	0.048	0.644
total sulfur dioxide	-1.0099	0.196	-5.143	0.000	-1.395	-0.625
density	-0.1031	0.162	-0.638	0.524	-0.420	0.214
pH	-0.6248	0.154	-4.063	0.000	-0.926	-0.323
sulphates	1.5071	0.189	7.989	0.000	1.137	1.877
alcohol	1.8265	0.131	13.912	0.000	1.569	2.084

독립변수 2개를 제외하고 보았을 때, R-squared, Adjusted R-squared 거의 변화가 없어, 두가지 변수를 제외하여도 문제가 없다는 것을 확인할 수 있었다. 또한, 결과를 보았을 때, 모든 독립변수를 이용하였을 때, p-value가 0.05를 넘지 않고 기여도가 작았던 'residual sugar', 'density'가 여전히 작은 기여도로 보이는 것으로 확인되었고, 이 두가지 변수도 제외하여 총 4가지 변수를 제외하고 OLS를 이용한 다중선형회귀 분석을 실시해 보았다.

#4가지 독립변수를 제외한 후 OLS를 이용한 다중선형회귀분석을 실시.

```
import statsmodels.api as sm
x_data = df_red[['volatile acidity','chlorides','free sulfur dioxide','total sulfur dioxide','pH','sulphates','alcohol']]
target = df_red[['quality']]

# for b0, 상수항 추가
x_data3 = sm.add_constant(x_data, has_constant = "add")

# OLS 검정
multi_model = sm.OLS(target, x_data3)
fitted_multi_model = multi_model.fit()
fitted_multi_model.summary()
```

Dep. Variable:	quality	R-squared:	0.359
Model:	OLS	Adj. R-squared:	0.357
Method:	Least Squares	F-statistic:	127.6
Date:	Sat, 05 Jun 2021	Prob (F-statistic):	5.32e-149
Time:	23:54:04	Log-Likelihood:	-1570.5
No. Observations:	1599	AIC:	3157.
Df Residuals:	1591	BIC:	3200.
Df Model:	7		
Covariance Type:	nonrobust		

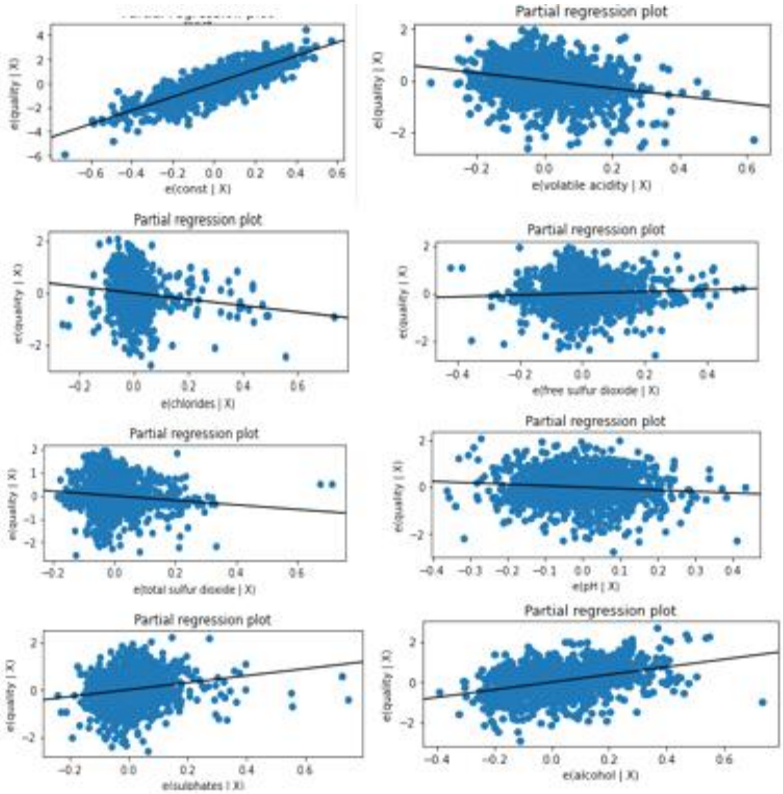
	coef	std err	t	P>|t|	[0.025	0.975]
const	5.6675	0.094	60.274	0.000	5.483	5.852
volatile acidity	-1.4786	0.147	-10.043	0.000	-1.767	-1.190
chlorides	-1.2087	0.238	-5.076	0.000	-1.676	-0.742
free sulfur dioxide	0.3605	0.151	2.389	0.017	0.064	0.657
total sulfur dioxide	-0.9855	0.194	-5.070	0.000	-1.367	-0.604
pH	-0.6130	0.149	-4.106	0.000	-0.906	-0.320
sulphates	1.4741	0.184	8.031	0.000	1.114	1.834
alcohol	1.8805	0.109	17.225	0.000	1.666	2.095

결과를 보면 R-squared 값이나 각 변수의 coef 값이 크게 변화가 없는 것을 확인할 수 있고, 나머지 변수의 p-value가 0.005이하이므로 이 모델을 사용하기로 한다.

● 회귀 가정 확인

(회귀 모델은 위의 4가지 독립변수를 제외한 모델을 사용)

선형성 확인

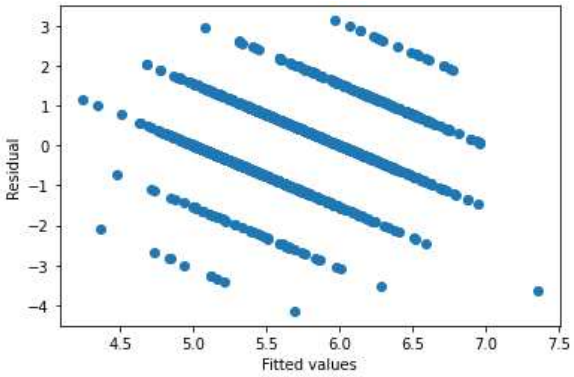



```
fig = sm.graphics.plot_regress_exog(fitted_multi_model, "alcohol", fig=plt.figure(figsize=(10, 5)))
plt.show()
```

각각의 변수에 대해 partial regression plot을 그려본 결과, 선형성을 보임을 확인할 수 있었다.

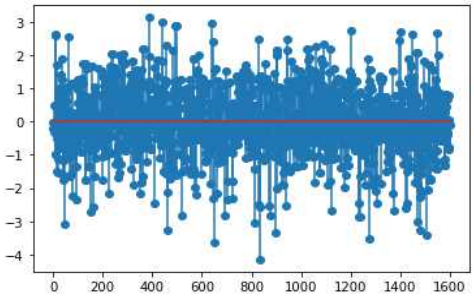
error의 등분산성 확인

```
sqrt_mse = np.sqrt(fitted_multi_model.mse_resid)
std_res = fitted_multi_model.resid/sqrt_mse ## studentized residual
predicted_val = fitted_multi_model.predict(x_data3)
fig = plt.scatter(predicted_val,std_res)
plt.xlabel('Fitted values')
plt.ylabel('Residual')
plt.show()
```



예측값과 잔차의 그래프를 보았을 때, 랜덤하지 않고 일정한 규칙을 갖고 있는 것을 확인할 수 있었다. 따라서 등분산성을 만족하지 않는다.

error의 독립성 확인



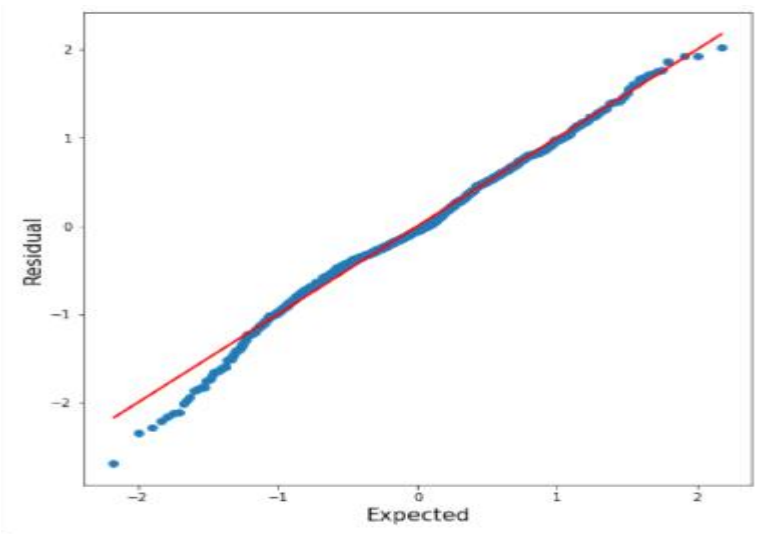

```
import statsmodels
statsmodels.stats.stattools.durbin_watson(std_res, axis=0)
plt.stem(std_res)
plt.show()
```

그래프를 확인한 결과 잔차의 규칙성이 보이지 않고, Durbin-Watson test결과 DW값이 1.7499674160815368로, 2와 근접한 값이 나왔으므로 error의 독립성이 있다고 할 수 있다.

error의 정규성 확인

정규성을 확인하기 위해 Q-Q plot을 그려보았다.

```
## qq plot
from scipy.stats import norm
sqrt_mse = np.sqrt(fitted_multi_model.mse_resid) ## square root of mse
num_const = 0.375 ## 백분위 분자 수정 계수
denom_const = 0.25 ## 백분위 분모 수정계수
## 오름차순으로 정렬했을 때 잔차의 순위
rank = [sorted(fitted_multi_model.resid).index(x)+1 for x in fitted_multi_model.resid] ## 인덱스가 0부터 시작하므로 1을 더한다.
expected_value = [] ## 이론적 잔차값
for i in range(len(fitted_multi_model.resid)):
    p = (rank[i]-num_const)/(len(fitted_multi_model.resid)+denom_const) ## 백분위
    expected_value.append(sqrt_mse*norm.ppf(p))
fig = plt.figure(figsize=(8,8))
fig.set_facecolor('white')
font_size = 15
plt.scatter(expected_value,fitted_multi_model.resid) ## 잔차도 출력
plt.plot(expected_value,expected_value,color='red')
plt.xlabel('Expected', fontsize=font_size)
plt.ylabel('Residual', fontsize=font_size)
plt.show()
```



Q-Q plot을 보았을 때, 직선 주위에 위치하므로 이는 정규성 가정을 만족한다고 볼 수 있다.

● 성능 및 다양한 회귀 방법 비교

최종 모델은 ‘fixed acidity’, ‘citric acid’, ‘residual sugar’, ‘density’ 4개의 변수를 제외한 다중선형회귀 모델을 사용하였고, 코드는 다음과 같다.

```
==================================================================
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
import statsmodels.api as sm
from scipy.stats import norm
import statsmodels

df_red = pd.read_csv('winequality-red.csv')
before_colname = 'fixed acidity;"volatile acidity";"citric acid";"residual sugar";"chlorides";"free sulfur dioxide";"total sulfur dioxide";"density";"pH";"sulphates";"alcohol";"quality"'
column_name = before_colname.split(';')
for i in range(len(column_name)):
    column_name[i]=column_name[i].replace("'","")
    column_name[i]=column_name[i].replace('"',"")
if not df_red.columns[0] == 'x':
    df_red.rename(columns = {'fixed acidity;"volatile acidity";"citric acid";"residual sugar";"chlorides";"free sulfur dioxide";"total sulfur dioxide";"density";"pH";"sulphates";"alcohol";"quality"':'x'}, inplace = True)
df_red = df_red.x.str.split(';')
df_red = df_red.apply(lambda x: pd.Series(x))
df_red.columns =['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
        'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
        'pH', 'sulphates', 'alcohol', 'quality']
df_normalized = df_red.drop(['quality'],axis='columns')
scaler = MinMaxScaler()
df_normalized[:] = scaler.fit_transform(df_normalized[:])
df_normalized['quality'] = df_red['quality']
df_red = df_normalized

x_data = df_red[['volatile acidity','chlorides','free sulfur dioxide','total sulfur dioxide','pH','sulphates','alcohol']]
target = df_red[['quality']]

x_data3 = sm.add_constant(x_data, has_constant = "add")
# OLS 검정
multi_model = sm.OLS(target.astype(float), x_data3.astype(float))
fitted_multi_model = multi_model.fit()
fitted_multi_model.summary()
==================================================================
```

그 때의 성능 R-squared 값은 0.359였다. 다른 모델을 사용한 경우의 R-squared 값은 다음과 같다.

Method name	R-squared
Linear Regression	0.359
Ridge Regression	0.35900
SVM	0.294737467

#ridge regression

```
from sklearn.linear_model import Ridge
from sklearn.metrics import r2_score

x_data = df_red[['volatile acidity',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',       'pH', 'sulphates', 'alcohol']]
target = df_red[['quality']]
ridge = Ridge().fit(x_data, target)
y_pred = ridge.predict(x_data)
r2 = r2_score(df_red['quality'],y_pred)
r2
```

#SVM

```
import sklearn.svm as svm
import sklearn.metrics as mt
from sklearn.model_selection import cross_val_score, cross_validate
x_data = df_red[['volatile acidity',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',       'pH', 'sulphates', 'alcohol']]
target = df_red[['quality']]
clf =svm.SVC(kernel = 'rbf')
clf = clf.fit(x_data, target)
y_pred = clf.predict(x_data)
r2 = r2_score(df_red['quality'],y_pred)
r2
```

2) 데이터 세트 "winequality-white.csv"에 대해서

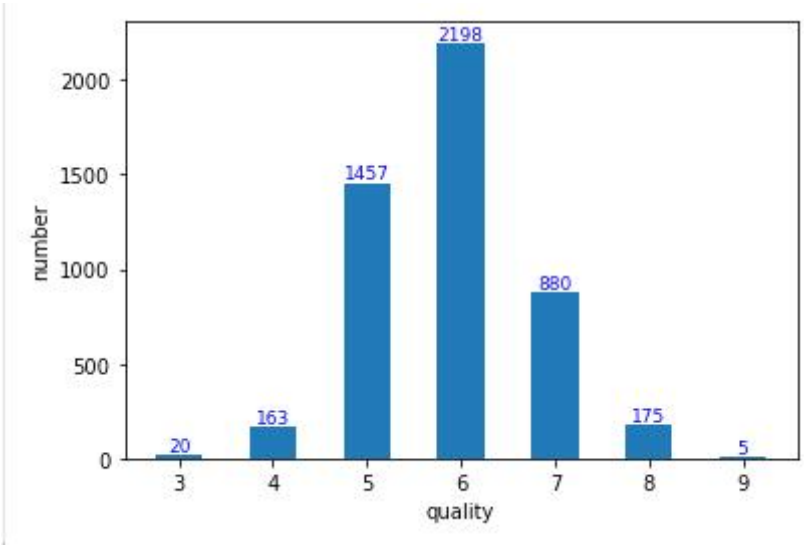
● 데이터 설명

red wine 데이터와 동일한 종류의 데이터이다.

● 데이터 전처리

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	0.45	8.8	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	0.49	9.5	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	0.44	10.1	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	9.9	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	9.9	6
...	...	...	...	...	...	...	...	...	...	...	...	...
4893	6.2	0.21	0.29	1.6	0.039	24.0	92.0	0.99114	3.27	0.50	11.2	6
4894	6.6	0.32	0.36	8.0	0.047	57.0	168.0	0.99490	3.15	0.46	9.6	5
4895	6.5	0.24	0.19	1.2	0.041	30.0	111.0	0.99254	2.99	0.46	9.4	6
4896	5.5	0.29	0.30	1.1	0.022	20.0	110.0	0.98869	3.34	0.38	12.8	7
4897	6.0	0.21	0.38	0.8	0.020	22.0	98.0	0.98941	3.26	0.32	11.8	6

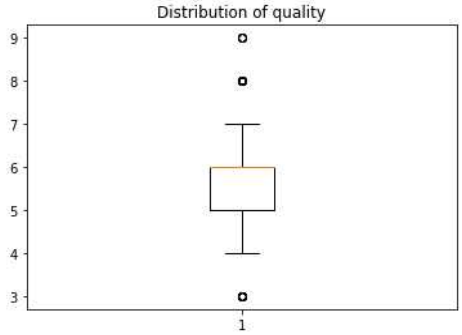
4898 rows × 12 columns

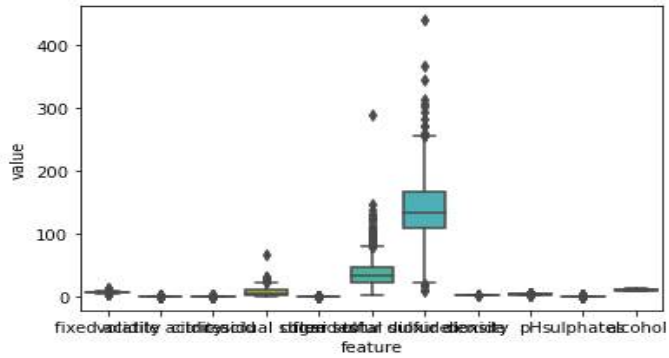


quality 데이터의 분포를 막대그래프로 살펴본 결과, 3-9점사이에 분포되어있었고, 분포가 완벽한 대칭을 이루지 않는다. 이것은 분석에 영향을 줄 것이다. quality의 통계적 세부사항은 아래와 같다.

#Distribution of quality

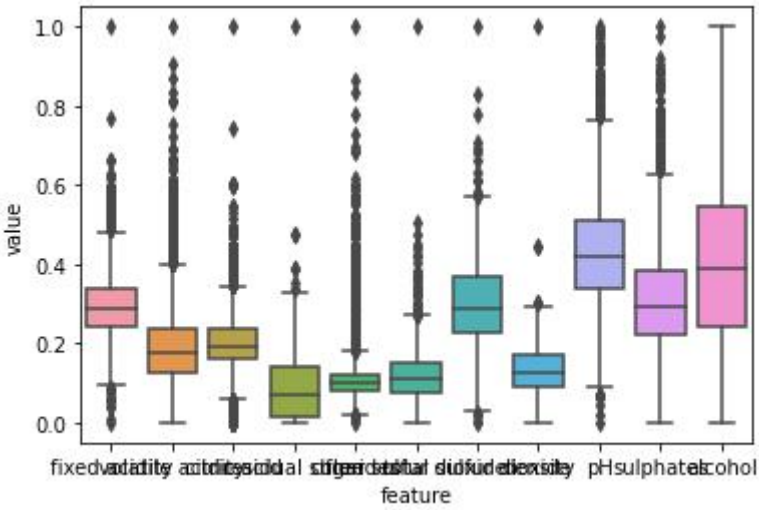
count: 4898
mean: 5.87790935075541
std: 0.8855481621683685
min: 3.0
q1: 5.0
q2: 6.0
q3: 6.0
max: 9.0





각각의 featur에 대한 데이터를 그래프로 나타내면 위의 그림과 같다. 각 feature별로 값의 차이가 많이나므로 모든 데이터가 0-1사이의 값을 갖도록 normalize를 해준다.

#Distribution of features - After normalization



● 데이터 시각화

Correlation between features

정규화한 데이터를 이용하여 피어슨 상관계수를 구해보면 다음과 같다.

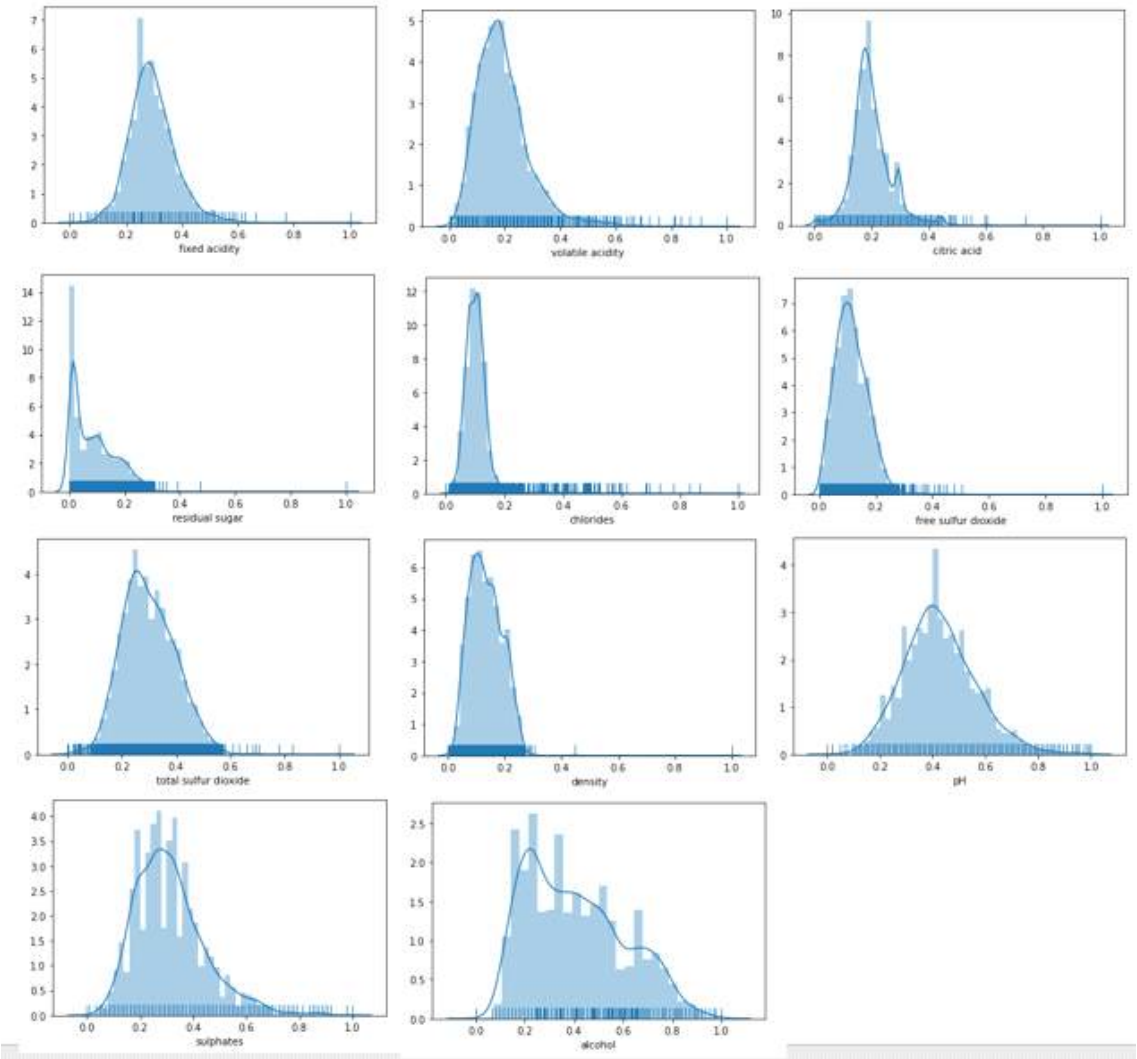
	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1.000000	-0.022697	0.289181	0.089021	0.023086	-0.049396	0.091070	0.265331	-0.425858	-0.017143	-0.120881	-0.113663
volatile acidity	-0.022697	1.000000	-0.149472	0.064286	0.070512	-0.097012	0.089261	0.027114	-0.031915	-0.035728	0.067718	-0.194723
citric acid	0.289181	-0.149472	1.000000	0.094212	0.114364	0.094077	0.121131	0.149503	-0.163748	0.062331	-0.075729	-0.009209
residual sugar	0.089021	0.064286	0.094212	1.000000	0.088685	0.299098	0.401439	0.838966	-0.194133	-0.026664	-0.450631	-0.097577
chlorides	0.023086	0.070512	0.114364	0.088685	1.000000	0.101392	0.198910	0.257211	-0.090439	0.016763	-0.360189	-0.209934
free sulfur dioxide	-0.049396	-0.097012	0.094077	0.299098	0.101392	1.000000	0.615501	0.294210	-0.000618	0.059217	-0.250104	0.008158
total sulfur dioxide	0.091070	0.089261	0.121131	0.401439	0.198910	0.615501	1.000000	0.529881	0.002321	0.134562	-0.448892	-0.174737
density	0.265331	0.027114	0.149503	0.838966	0.257211	0.294210	0.529881	1.000000	-0.093591	0.074493	-0.780138	-0.307123
pH	-0.425858	-0.031915	-0.163748	-0.194133	-0.090439	-0.000618	0.002321	-0.093591	1.000000	0.155951	0.121432	0.099427
sulphates	-0.017143	-0.035728	0.062331	-0.026664	0.016763	0.059217	0.134562	0.074493	0.155951	1.000000	-0.017433	0.053678
alcohol	-0.120881	0.067718	-0.075729	-0.450631	-0.360189	-0.250104	-0.448892	-0.780138	0.121432	-0.017433	1.000000	0.435575
quality	-0.113663	-0.194723	-0.009209	-0.097577	-0.209934	0.008158	-0.174737	-0.307123	0.099427	0.053678	0.435575	1.000000

이 중 corr이 0.5가 넘는 것을 찾아보면, 다음과 같은 4쌍이 존재한다.

residual sugar & density : 0.838966454904577
total sulfur dioxide & free sulfur dioxide : 0.6155009650098442
density & total sulfur dioxide : 0.529881323878613
alcohol & density : -0.7801376214258094

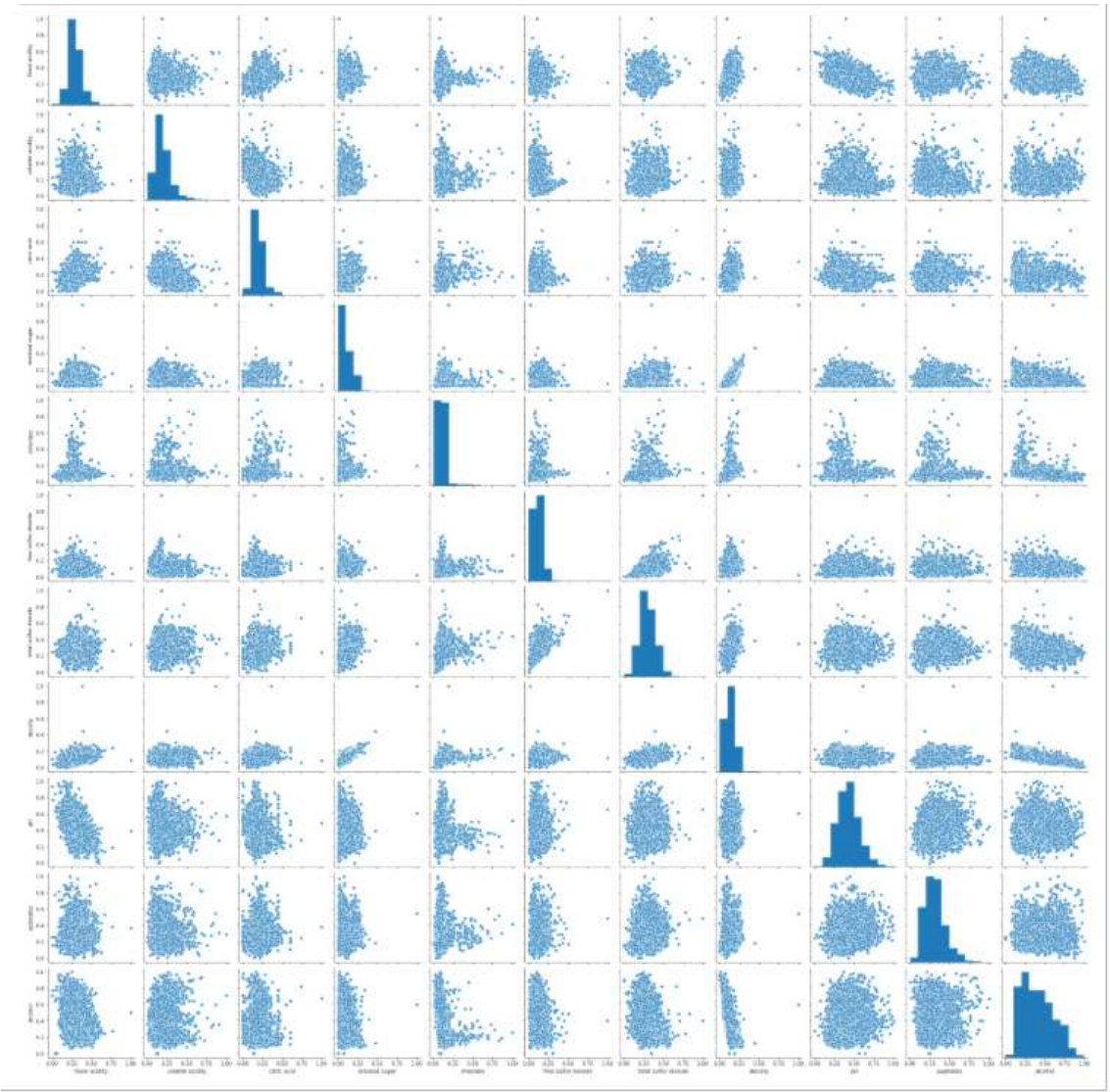
#Distribution of each feature

각 독립변수의 데이터가 어떤 분포를 따르는지 그래프를 그려보았다.



대부분의 변수가 정규분포 모양을 보였지만 굴곡이 많이 보이는 그래프도 있었고, 이는 영향이 있을 것이라 예상된다.

#Pair plot between features



● 회귀 모델 구축

#주어진 모든 독립변인 변수를 이용하여 OLS를 이용한 다중선형회귀분석을 실시였다.

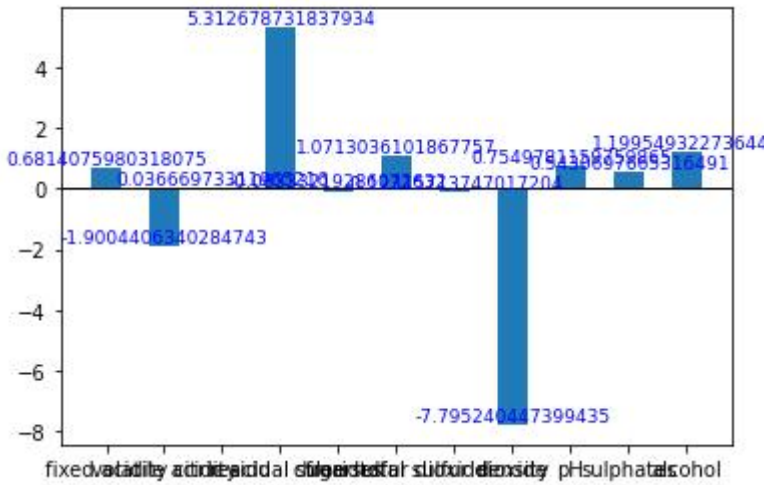
OLS Regression Results

Dep. Variable:	quality	R-squared:	0.282
Model:	OLS	Adj. R-squared:	0.280
Method:	Least Squares	F-statistic:	174.3
Date:	Sun, 06 Jun 2021	Prob (F-statistic):	0.00
Time:	02:43:13	Log-Likelihood:	-5543.7
No. Observations:	4898	AIC:	1.111e+04
Df Residuals:	4886	BIC:	1.119e+04
Df Model:	11		
Covariance Type:	nonrobust		

그 결과 R-squared는 0.282, Adj. R-squared는 0.280이 나왔고, 이 모델의 P-value가 0.05보다 작아 귀무가설을 유의수준 0.05수준에서 기각할 수 있고 따라서 회귀성이 존재한다고 생각할 수 있다.

	coef	std err	t	P>|t|	[0.025	0.975]
const	5.5509	0.107	51.650	0.000	5.340	5.762
fixed acidity	0.6814	0.217	3.139	0.002	0.256	1.107
volatile acidity	-1.9004	0.116	-16.373	0.000	-2.128	-1.673
citric acid	0.0367	0.159	0.231	0.818	-0.275	0.348
residual sugar	5.3127	0.491	10.825	0.000	4.351	6.275
chlorides	-0.0833	0.184	-0.452	0.651	-0.444	0.278
free sulfur dioxide	1.0713	0.242	4.422	0.000	0.596	1.546
total sulfur dioxide	-0.1232	0.163	-0.756	0.450	-0.443	0.196
density	-7.7952	0.989	-7.879	0.000	-9.735	-5.856
pH	0.7550	0.116	6.513	0.000	0.528	0.982
sulphates	0.5431	0.086	6.291	0.000	0.374	0.712
alcohol	1.1995	0.150	7.988	0.000	0.905	1.494

또한, 각 독립변수에서의 p-value를 살펴보면, 'chlorides', 'citric acid', 'total sulfur dioxide' 변수가 0.05보다 큰 것을 확인할 수 있었다. 그리고, 각 coef를 그래프로 나타내보면,



p-value가 0.05를 넘지 못하는 3가지 독립변수는 coef 절대값이 0.13을 넘지 못하고 나머지 변수들에 비해 가장 작은 3가지 변수임을 확인할 수 있었다. ('chlorides', 'citric acid', 'total sulfur dioxide')

위의 3가지 독립변수는 다중회귀 모델에서 베타 값도 작고 p-value도 0.05보다 높으므로, 세 독립변수를 제거하고 다시 OLS를 이용한 다중선형회귀 분석을 실시해 보았다.

3가지 독립변수를 제외한 후 OLS를 이용한 다중선형회귀분석을 실시.

OLS Regression Results

Dep. Variable:	quality	**R-squared:**	0.282
Model:	OLS	**Adj. R-squared:**	0.281
Method:	Least Squares	**F-statistic:**	239.7
Date:	Sun, 06 Jun 2021	**Prob (F-statistic):**	0.00
Time:	02:52:52	**Log-Likelihood:**	-5544.1
No. Observations:	4898	**AIC:**	1.111e+04
Df Residuals:	4889	**BIC:**	1.116e+04
Df Model:	8		
Covariance Type:	nonrobust		

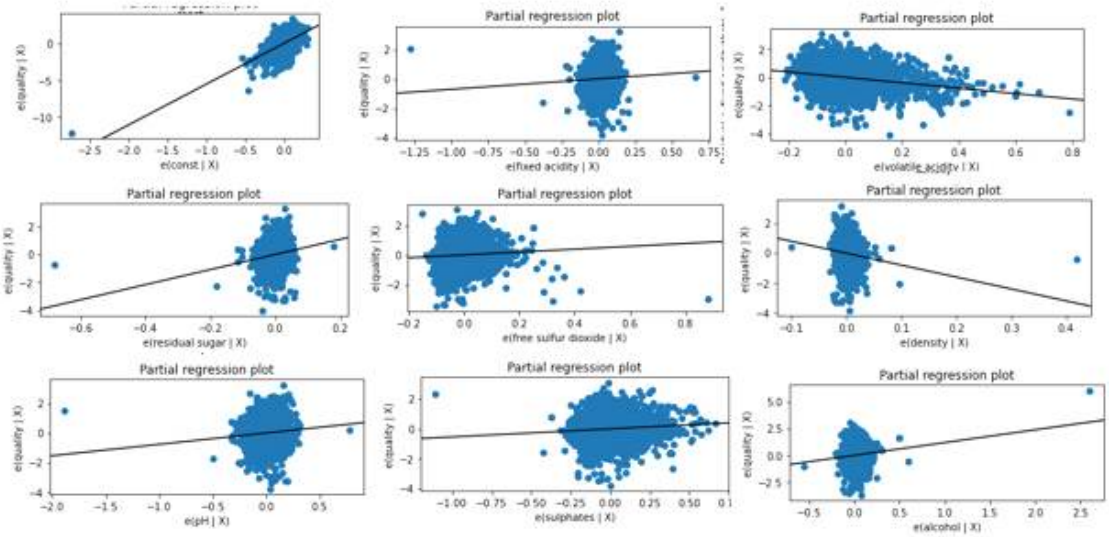
	coef	std err	t	P>|t|	[0.025	0.975]
const	5.5398	0.104	53.209	0.000	5.336	5.744
fixed acidity	0.7083	0.212	3.333	0.001	0.292	1.125
volatile acidity	-1.9259	0.112	-17.242	0.000	-2.145	-1.707
residual sugar	5.4016	0.475	11.370	0.000	4.470	6.333
free sulfur dioxide	0.9612	0.194	4.950	0.000	0.580	1.342
density	-8.0031	0.952	-8.411	0.000	-9.868	-6.138
pH	0.7636	0.114	6.717	0.000	0.541	0.987
sulphates	0.5405	0.086	6.287	0.000	0.372	0.709
alcohol	1.1976	0.149	8.021	0.000	0.905	1.490

독립변수 3개를 제외하고 보았을 때, R-squared, Adjusted R-squared 거의 변화가 없어, 세가지 변수를 제외하여도 문제가 없다는 것을 확인할 수 있었다. 또한, 각 변수의 coef 값이 크게 변화가 없는 것을 확인할 수 있고, 나머지 변수의 p-value가 0.005이하이므로 이 모델을 사용하기로 한다.

● 회귀 가정 확인

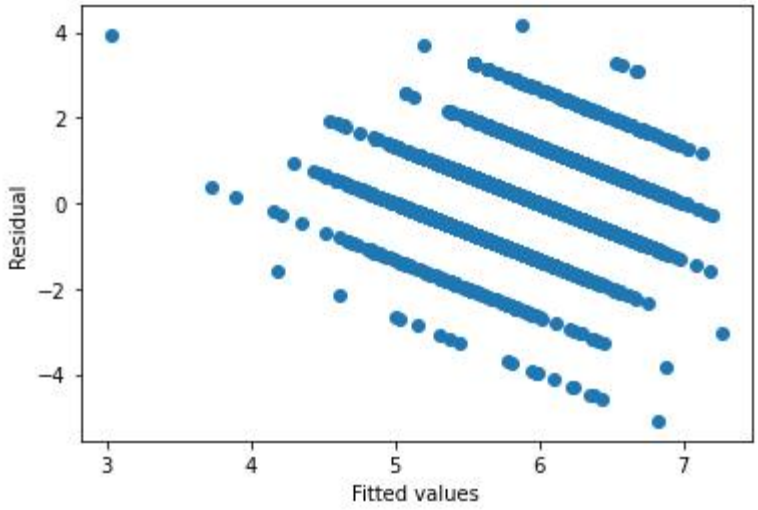
(회귀 모델은 위의 3가지 독립변수를 제외한 모델을 사용)

선형성 확인



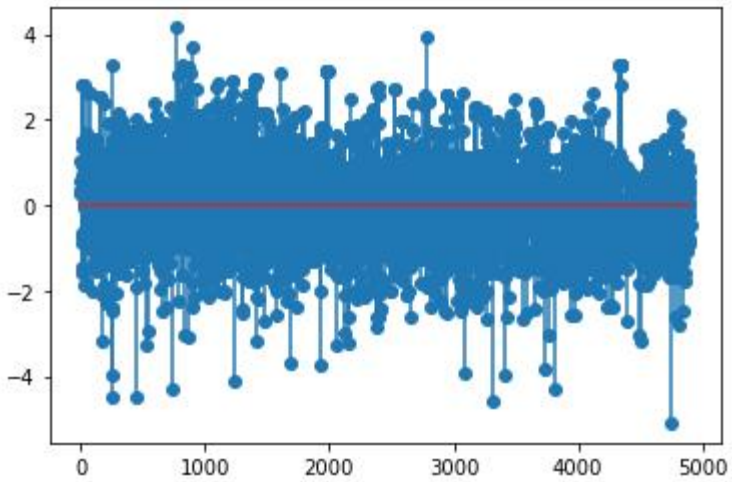
각각의 변수에 대해 partial regression plot을 그려본 결과, 선형성을 보임을 확인할 수 있었다.

error의 등분산성 확인



예측값과 잔차의 그래프를 보았을 때, 랜덤하지 않고 일정한 규칙을 갖고 있는 것을 확인할 수 있었다. 따라서 등분산성을 만족하지 않는다.

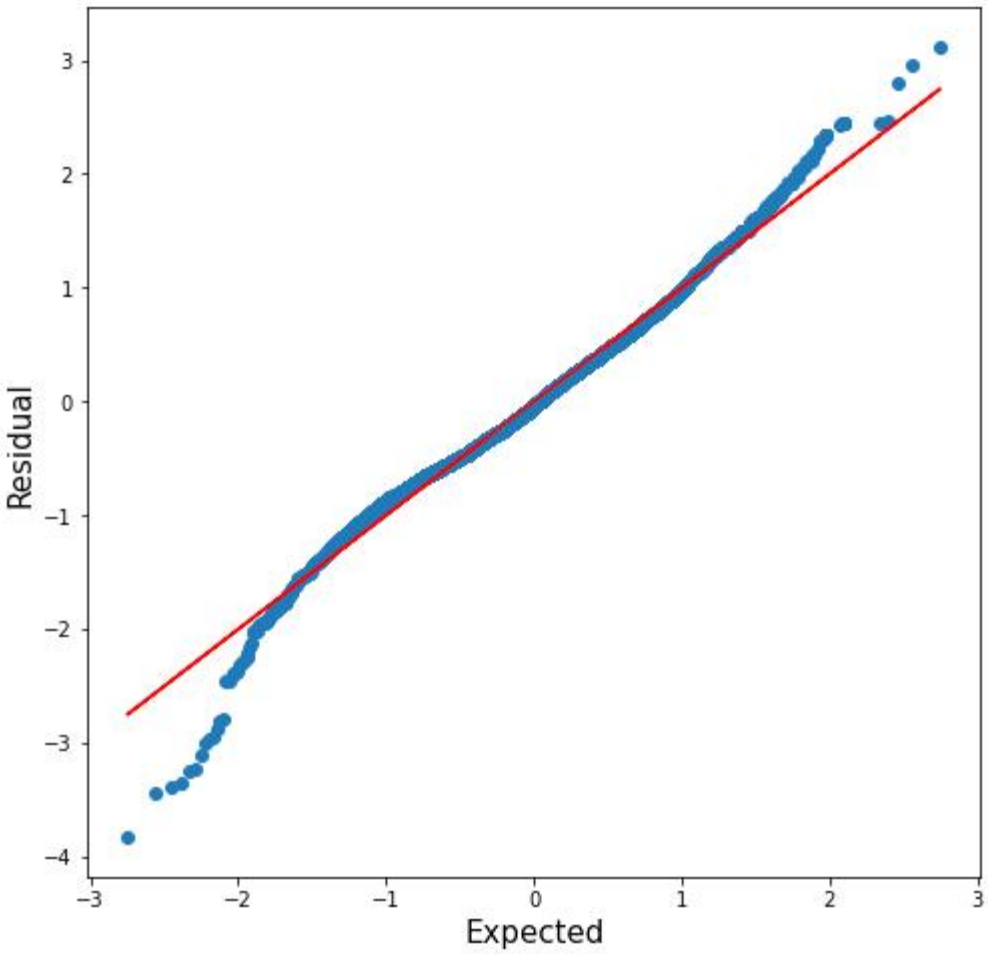
error의 독립성 확인



그래프를 확인한 결과 잔차의 규칙성이 보이지 않고, Durbin-Watson test 결과 DW값이 1.6213143706376654로, 2와 근접한 값이 나왔으므로 error의 독립성이 있다고 할 수 있다.

error의 정규성 확인

정규성을 확인하기 위해 Q-Q plot을 그려보았다.



Q-Q plot을 보았을 때, 직선 주위에 위치하므로 이는 정규성 가정을 만족한다고 볼 수 있다.

● 성능 및 다양한 회귀 방법 비교

최종 모델은 'chlorides', 'citric acid', 'total sulfur dioxide' 3개의 변수를 제외한 다중선형회귀 모델을 사용하였고, 코드는 다음과 같다.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
import statsmodels.api as sm
from scipy.stats import norm
import statsmodels

df_white = pd.read_csv('winequality-white.csv')
df_normalized = df_white.drop(['quality'],axis='columns')
scaler = MinMaxScaler()
df_normalized[:] = scaler.fit_transform(df_normalized[:])
df_normalized['quality'] = df_white['quality']
df_white = df_normalized
x_data = df_white[['fixed acidity','volatile acidity','residual sugar','free sulfur dioxide','density','pH','sulphates','alcohol']]
target = df_white[['quality']]
x_data2 = sm.add_constant(x_data, has_constant = "add")
# OLS 검정
multi_model = sm.OLS(target.astype(float), x_data2.astype(float))
fitted_multi_model = multi_model.fit()
fitted_multi_model.summary()
```

그 때의 성능 R-squared 값은 0.282였다. 다른 모델을 사용한 경우의 R-squared 값은 다음과 같다.

Method name	R-squared
Linear Regression	0.282
Ridge Regression	0.27667691
SVM	0.208276

ridge regression

```
from sklearn.linear_model import Ridge
from sklearn.metrics import r2_score

x_data = df_white[['fixed acidity', 'volatile acidity', 'residual sugar',
        'free sulfur dioxide', 'density',
        'pH', 'sulphates', 'alcohol']]
target = df_white[['quality']]
ridge = Ridge().fit(x_data, target)
y_pred = ridge.predict(x_data)
r2 = r2_score(df_white['quality'],y_pred)
r2
```

#SVM

```
import sklearn.svm as svm
import sklearn.metrics as mt
from sklearn.model_selection import cross_val_score, cross_validate
x_data = df_white[['fixed acidity', 'volatile acidity', 'residual sugar',
        'free sulfur dioxide', 'density',
        'pH', 'sulphates', 'alcohol']]
target = df_white[['quality']]
clf =svm.SVC(kernel = 'rbf')
clf = clf.fit(x_data, target)
y_pred = clf.predict(x_data)
r2 = r2_score(df_white['quality'],y_pred)
r2
```