회귀 진단 (Regression Diagnostics)

- 구축된 회귀 모델이 데이터를 얼마나 잘 설명하는지, 그리고 회귀 분석의 기본 가정들을 만족하는지를 체계적으로 검토하는 과정
- 이 과정을 통해 모델의 신뢰성을 평가하고, 문제가 발견될 경우 모델을 개선하기 위한 방향을 설정 가능
- 주요 진단 항목에는 선형성, 잔차의 정규성, 등분산성, 독립성, 그리고 다중공선성 등이 있습니다.

적용 가능한 상황

- 선형 회귀 모델(OLS)을 구축한 후, 해당 모델이 통계적으로 유의미하고 신뢰할 수 있는지 검증하고자 할 때 필수적으로 수행됩니다.
- 모델의 예측 성능이 기대에 미치지 못할 때, 어떤 가정에 위배되어 문제가 발생했는지 원인을 파악하기 위해 사용됩니다.
- 회귀 분석 결과를 해석하고 보고하기 전에, 분석의 타당성을 확보하기 위한 목적으로 수행됩니다.

---

1. 회귀 모델의 기본 가정 진단

선형 회귀 모델이 최적의 추정치(BLUE: Best Linear Unbiased Estimator)가 되기 위해서는 잔차(Residuals)가 다음의 가정을 만족해야 합니다.

1.1. 선형성 (Linearity)

- **개념**: 독립변수와 종속변수 간의 관계가 선형적이라는 가정입니다.
- **진단 방법**:
  - **잔차 대 예측값 산점도 (Residuals vs. Fitted Plot)**: 예측값(Fitted values)에 대한 잔차(Residuals)를 산점도로 그렸을 때, 잔차들이 `y=0` 선을 기준으로 특별한 패턴 없이 무작위로 흩어져 있어야 합니다. 만약 굽은 형태(e.g. 2차 함수)의 패턴이 보인다면 선형성 가정을 위배한 것입니다.
- **위반 시 해결 방안**:
  - **다항 회귀**: 독립변수의 고차항(x^2, x^3)을 모델에 추가합니다.
  - **변수 변환**: 독립변수나 종속변수에 로그, 제곱근 등 비선형 변환을 적용합니다.
  - **비선형 모델 사용**: GAM, 트리 기반 모델 등 다른 비선형 모델을 고려합니다.

1.2. 잔차의 정규성 (Normality of Residuals)

- **개념**: 잔차가 정규분포를 따라야 한다는 가정입니다. 이는 회귀 계수의 유의성 검정(t-검정, F-검정)에 대한 신뢰도를 부여합니다.
- **진단 방법**:
  - **Q-Q Plot (Quantile-Quantile Plot)**: 잔차의 분위수와 정규분포의 분위수를 비교하는 그래프로, 점들이 45도 직선에 가깝게 분포하면 정규성을 만족한다고 봅니다.
  - **히스토그램/커널 밀도 추정**: 잔차의 분포를 시각화하여 정규분포 형태와 비교합니다.
  - **정규성 검정**: Shapiro-Wilk 검정, Kolmogorov-Smirnov 검정 등 통계적 검정을 수행합니다. (단, 표본이 매우 클 경우 사소한 차이에도 귀무가설을 기각할 수 있어 시각적 방법과 병행하는 것이 좋습니다.)
- **위반 시 해결 방안**:
  - **변수 변환**: 종속변수에 로그, Box-Cox 변환 등을 적용하여 분포를 정규분포에 가깝게 만듭니다.
  - **이상치 제거**: 극단적인 잔차를 유발하는 이상치를 확인하고 제거 또는 수정합니다.

1.3. 잔차의 등분산성 (Homoscedasticity)

- **개념**: 모든 예측값 수준에서 잔차의 분산이 일정해야 한다는 가정입니다. 만약 예측값에 따라 잔차의 분산이 변하는 현상(이분산성, Heteroscedasticity)이 나타나면, 회귀 계수 추정치의 효율성이 떨어지고 표준오차 추정에 문제가 생겨 통계적 유의성 검정을 신뢰할 수 없게 됩니다.
- **진단 방법**:
  - **잔차 대 예측값 산점도**: 예측값이 커짐에 따라 잔차가 퍼지는 형태(깔때기 모양)나 모이는 형태가 나타나면 이분산성을 의심합니다.
  - **등분산성 검정**: Breusch-Pagan 검정, White 검정 등을 수행합니다.
- **위반 시 해결 방안**:
  - **변수 변환**: 분산을 안정화시키기 위해 종속변수에 로그나 제곱근 변환을 적용합니다.
  - **가중 최소 제곱법 (WLS)**: 분산이 작은 관측치에 더 높은 가중치를 부여하여 회귀 모델을 추정합니다.
  - **강건 표준오차 (Robust Standard Errors)**: 이분산성이 존재하더라도 일관성 있는 표준오차를 추정하는 방법을 사용합니다. (`statsmodels`에서 `fit(cov_type='HC0', 'HC1', ...)` 옵션)

1.4. 잔차의 독립성 (Independence of Residuals)

- **개념**: 잔차들끼리 서로 상관관계가 없어야 한다는 가정입니다. 특히 시계열 데이터에서 현재의 오차가 이전의 오차에 영향을 받는 '자기상관(Autocorrelation)'이 문제가 됩니다.
- **진단 방법**:
  - **더빈-왓슨 통계량 (Durbin-Watson Statistic)**: 잔차의 자기상관을 검정하는 통계량입니다.
    - 값이 2에 가까우면 자기상관이 없습니다.
    - 0에 가까우면 양의 자기상관을, 4에 가까우면 음의 자기상관을 의심합니다.
  - **ACF Plot (Autocorrelation Function Plot)**: 시차(lag)에 따른 잔차의 자기상관을 시각적으로 보여줍니다.
- **위반 시 해결 방안**:
  - **시계열 모델 사용**: AR, ARIMA 등 자기상관 구조를 모델링할 수 있는 시계열 모델을 적용합니다.
  - **차분 (Differencing)**: 시계열 데이터의 경우, 변수를 차분하여 정상성(stationarity)을 확보하고 자기상관을 제거합니다.

---

2. 다중공선성 (Multicollinearity)

- **개념**: 회귀 모델에서 일부 독립변수가 다른 독립변수와 강한 선형 관계를 가질 때 발생하는 문제입니다. 이 경우, 특정 변수가 종속변수에 미치는 고유한 영향을 정확히 추정하기 어려워집니다.
- **문제점**:
  - 회귀 계수 추정치의 분산이 커져 매우 불안정해집니다. (데이터가 약간만 바뀌어도 계수가 크게 변동)
  - 회귀 계수의 표준오차가 커져 t-통계량이 작아지므로, 개별 변수들의 p-value가 커져 통계적으로 유의하지 않다는 결론이 나올 수 있습니다. (모델 전체의 설명력(F-통계량)은 높은데, 개별 변수는 유의하지 않은 현상)
- **진단 방법**:
  - **분산 팽창 계수 (Variance Inflation Factor, VIF)**: 하나의 독립변수를 다른 독립변수들로 선형회귀한 모델의 결정계수(R^2)를 이용하여 계산합니다. ($VIF = 1 / (1 - R^2)$)
    - 일반적으로 VIF 값이 10 이상이면 다중공선성이 심각하다고 판단하며, 5 이상일 때도 주의가 필요하다고 봅니다.

    - **상관 행렬 (Correlation Matrix)**: 독립변수들 간의 상관계수를 확인합니다. 상관계수가 매우 높은 (e.g. > 0.8) 변수 쌍이 있다면 다중공선성을 의심할 수 있습니다. (단, VIF는 여러 변수 간의 관계를 종합적으로 보므로 더 정확합니다.)
- **위반 시 해결 방안**:
    - **변수 제거**: VIF가 높은 변수들 중 다른 변수와 개념적으로 중복되거나 덜 중요한 변수를 제거합니다.
    - **변수 결합/파생 변수 생성**: 상관관계가 높은 변수들을 합치거나 평균을 내어 새로운 변수를 만듭니다.
    - **주성분 분석 (PCA)**: 상관관계가 높은 변수들을 소수의 주성분으로 변환하여 회귀 분석에 사용합니다. (단, 주성분의 해석이 어려워질 수 있습니다.)
    - **규제 모델 사용 (Ridge, Lasso)**: 릿지(Ridge) 회귀는 다중공선성 문제에 특히 강건하게 작동하여 계수 추정치를 안정화시키는 효과가 있습니다.

코드 예시

`statsmodels`는 회귀 진단을 위한 풍부한 기능과 시각화 도구를 제공합니다.

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor
import matplotlib.pyplot as plt
import seaborn as sns

# 1. 데이터 준비 및 모델 학습
# 보스턴 주택 가격 데이터 사용
from sklearn.datasets import fetch_california_housing
housing = fetch_california_housing()
X = pd.DataFrame(housing.data, columns=housing.feature_names)
y = housing.target
X = sm.add_constant(X) # 상수항 추가

model = sm.OLS(y, X)
results = model.fit()
print(results.summary())
'''
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.606
Model:                            OLS   Adj. R-squared:                  0.606
Method:                 Least Squares   F-statistic:                     3970.
Date:                Fri, 10 Oct 2025   Prob (F-statistic):               0.00
Time:                        17:32:43   Log-Likelihood:                -22624.
No. Observations:               20640   AIC:                         4.527e+04
Df Residuals:                   20631   BIC:                         4.534e+04
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
```

```
const        -36.9419      0.659    -56.067      0.000     -38.233     -35.650
MedInc         0.4367      0.004    104.054      0.000       0.428       0.445
HouseAge       0.0094      0.000     21.143      0.000       0.009       0.010
AveRooms      -0.1073      0.006    -18.235      0.000      -0.119      -0.096
AveBedrms      0.6451      0.028     22.928      0.000       0.590       0.700
Population -3.976e-06   4.75e-06     -0.837      0.402   -1.33e-05    5.33e-06
AveOccup      -0.0038      0.000     -7.769      0.000      -0.005      -0.003
Latitude      -0.4213      0.007    -58.541      0.000      -0.435      -0.407
Longitude     -0.4345      0.008    -57.682      0.000      -0.449      -0.420
==============================================================================
Omnibus:                     4393.650   Durbin-Watson:                   0.885
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            14087.596
Skew:                           1.082   Prob(JB):                         0.00
Kurtosis:                       6.420   Cond. No.                     2.38e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.38e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
'''
# 2. 잔차 및 예측값 계산
residuals = results.resid
fitted = results.fittedvalues

# 3. 회귀 가정 진단
# 3.1. 선형성 & 3.3. 등분산성: 잔차 대 예측값 산점도
sns.residplot(x=fitted, y=residuals, lowess=True, line_kws={'color': 'red', 'lw':
1})
plt.title('Residuals vs Fitted')
plt.xlabel('Fitted values')
plt.ylabel('Residuals')
plt.show()
# 해석: 빨간색 lowess 곡선이 수평선(y=0)에서 크게 벗어나지 않고, 잔차들이 패턴 없이 흩
어져 있으면 OK.

# 3.2. 정규성: Q-Q Plot
sm.qqplot(residuals, line='s')
plt.title('Q-Q Plot of Residuals')
plt.show()
# 해석: 점들이 빨간색 대각선 위에 놓여 있으면 정규성 만족.

# 3.4. 독립성: 더빈-왓슨 통계량
# results.summary() 결과표의 'Durbin-Watson' 값 확인 (약 0.885으로 양의 자기상관 의
심)

# 4. 다중공선성 진단: VIF
# VIF 계산을 위한 함수 정의
def calculate_vif(X_df):
    vif = pd.DataFrame()
    vif["variables"] = X_df.columns
    vif["VIF"] = [variance_inflation_factor(X_df.values, i) for i in
range(X_df.shape[1])]
```

```
    return vif

# 상수항(const)은 VIF 계산에서 제외
vif_results = calculate_vif(X.drop('const', axis=1))
print("\n--- VIF 결과 ---")
print(vif_results.sort_values('VIF', ascending=False))
'''
    variables         VIF
7   Longitude  633.711654
6    Latitude  559.874071
2    AveRooms   45.993601
3   AveBedrms   43.590314
0      MedInc   11.511140
1    HouseAge    7.195917
4  Population    2.935745
5    AveOccup    1.095243
'''
# 해석: AveRooms, AveBedrms 등의 VIF가 높게 나타나는 것을 확인할 수 있음.
# 특히 위도(Latitude)와 경도(Longitude)는 함께 위치 정보를 나타내므로 VIF가 높게 나올
수 있음.
```