출처가 명시되지 않은 모든 자료(이미지 등)는 조성현 강사님 블로그 및 강의 자료 기반.

[머신러닝-회귀]

[Linear Regression]

앞에서 살펴본 모델들이 결정 경계를 찾아내는 데 집중하는 모델이었던 것과 달리, 회귀 모델은 입력 데이터가 주어졌을 때, 타겟이 나타날 조건부 확률을 근거로 데이터를 분류한다.

이름에서도 알 수 있듯, 입력 데이터와 출력 데이터 간 선형 관계가 있음을 가정하고 선형 모델을 만들어 알고자 하는 예측하는 방식이다.

1. 원리

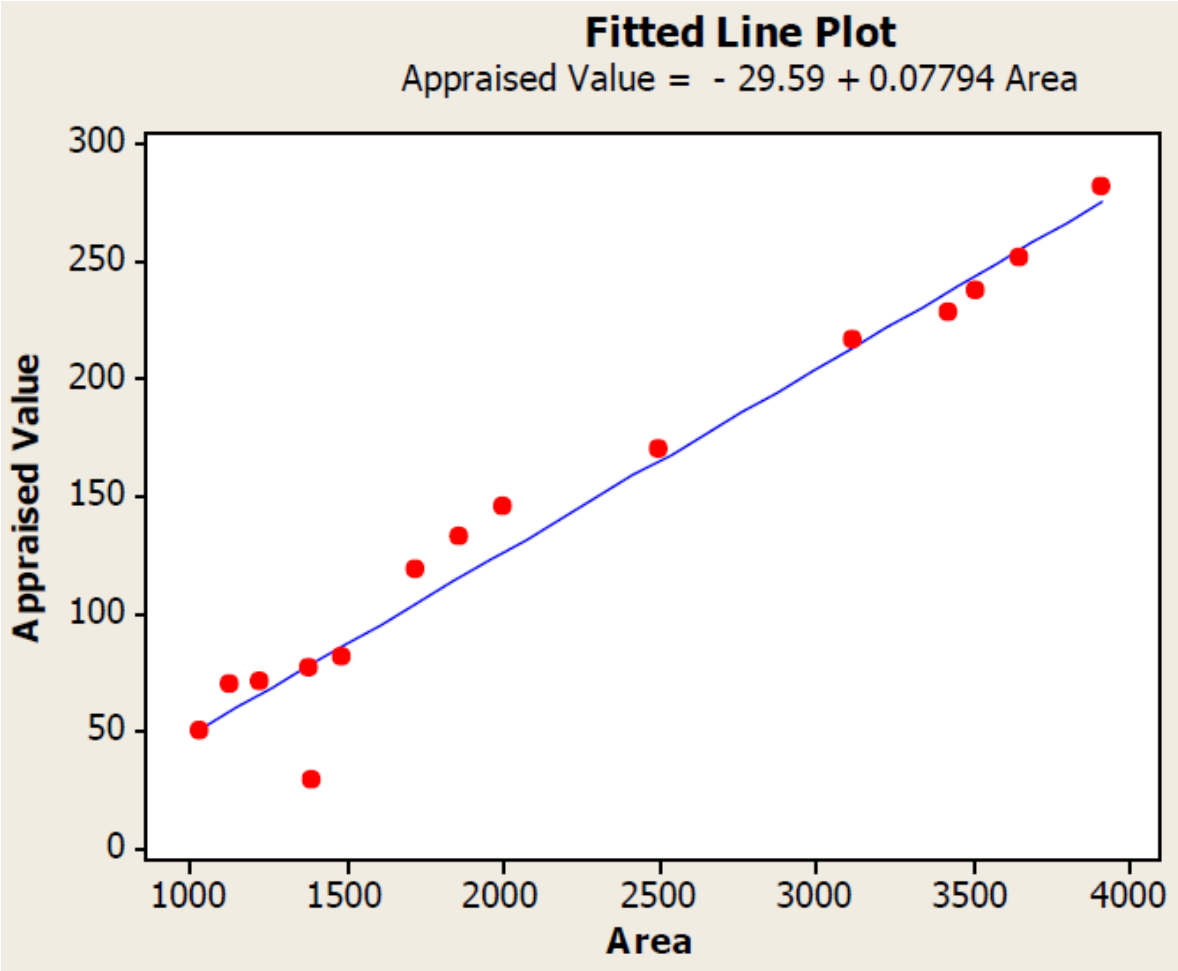


그림 출처: [https://ratsgo.github.io/machine%20learning/2017/07/03/regression/](https://ratsgo.github.io/machine learning/2017/07/03/regression/)

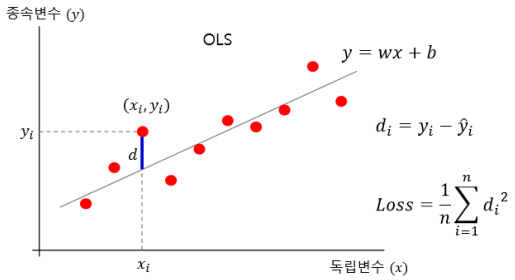
feature 데이터 X와 target 데이터 Y 간 관계를 선형으로 가정하고, 학습 데이터로부터 둘 사이 선형 관계를 이를 가장 잘 표현할 수 있는 계수를 찾아 내고자 한다. 예컨대 위의 그림에서와 같이 집 크기와 가격의 관계, 시험 공부 시간과 시험 점수 간의 관계 등이 그것이다.

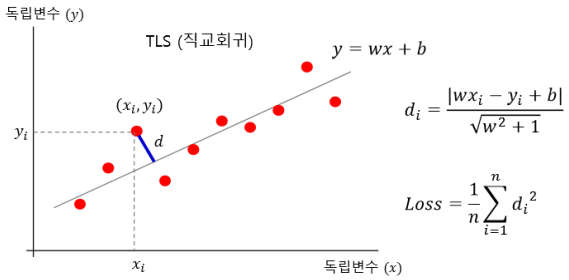
2. 계수 추정 방식

위에서 살펴봤듯, X와 Y 사이에 선형 관계가 있다고 가정하므로, 둘 간의 관계를 나타내는 직선의 방정식을 찾아내는 것이 목표가 된다. 이 때, 선형 회귀 선을 찾아내기 위한 계수 추정 방식으로 2가지가 있다. **OLS**(*Ordinary Least Squares*)와 **TLS**(*Total Least Squares*)이다.

두 방식 모두 Least Squares 방식을 이용한다. loss function으로 데이터와 선형 회귀선까지의 거리 제곱합의 평균을 최소화하는 방식을 사용한다. 제곱의 평균으로 계산하는 방법을 squared error라고 하기 때문에, 선형회귀 방법에서 최소화해야 할 목적함수는 **MSE(*Mean Squared Error*)**가 된다.

MSE를 최소화하는 W, b를 찾는 것이 선형회귀의 방법론이다. OLS와 TLS는 거리 d를 어떻게 측정할 것인지에 따라 달라진다.





OLS

OLS에서 거리 `d`는 데이터로부터 회귀선까지 y축에 평행한 방향으로 만나는 지점까지의 거리를 의미한다. 모든 데이터로부터 회귀선까지의 거리를 계산한 뒤, 최소제곱법을 이용해 그 거리가 최소가 되는 지점을 찾는 것이다.

주로 X 데이터는 변동이 없다고 가정할 때 사용한다. X축이 기준이 되고, X축 데이터에 대한 Y축 데이터에서만 오차가 발생한다. 한 변수를 독립 변수로 놓고, 그 변수 하나에만 종속되는 다른 변수에 대해 그 관계를 분석하고 싶을 때 사용한다.

TLS

직교회귀 방식이라고도 한다. TLS에서 거리 `d`는 데이터로부터 회귀선까지의 수직 거리를 의미한다. 모든 데이터로부터 회귀선까지의 수직 거리 총 합을 계산한뒤, 그 제곱합이 최소가 되는 계수를 찾는 것이다.

OLS에서와 달리 두 변수가 모두 독립변수일 때 유효한 방식이다. 기준 축이 없다. X축 변수와 Y축 변수가 모두 변동하고, 둘 다 오차를 포함할 수 있다.

실습 1. 선형 회귀 모델 학습

사이킷런에서는 OLS 방식만 지원한다. 선형회귀 방식이 무엇인지 아는 것이 목적이므로, 임의의 데이터 (X, y)를 생성한 후 회귀 직선을 찾는다.

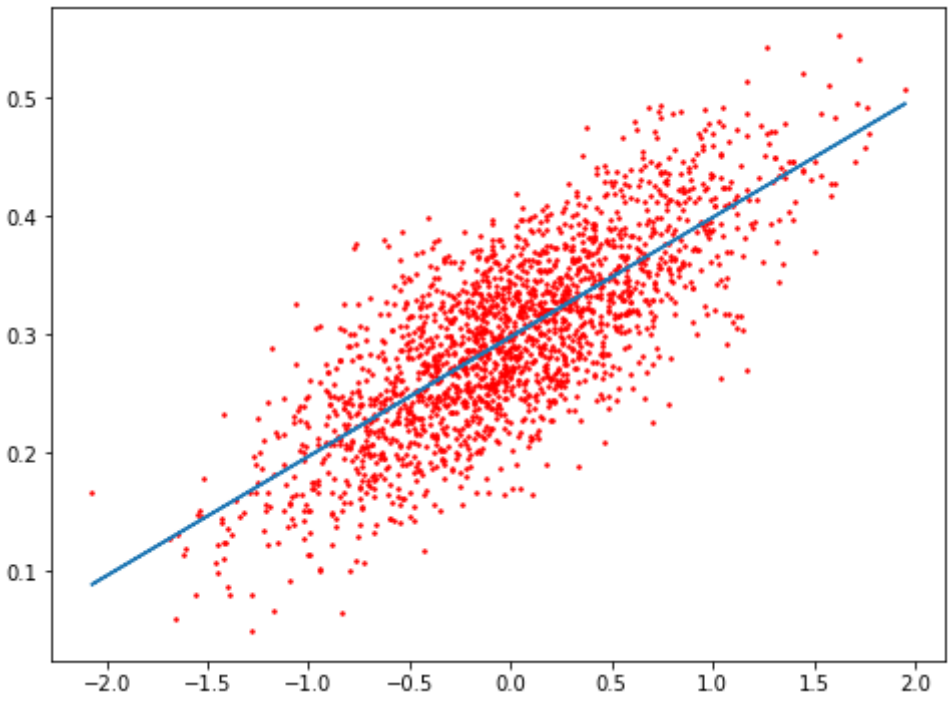
- 임의의 선형 관계 데이터 생성하는 방법

```
# create sample data
def createData(a, b, n):
    result_X, result_y = [], []

    for i in range(n):
        x = np.random.normal(0.0, 0.6)
        y = a*x + b+ np.random.normal(0.0, 0.05) # y = ax + b + e, error term
0.0 ~ 0.05
        result_X.append(x)
        result_y.append(y)

    return np.array(result_X).reshape(-1, 1), np.array(result_y).reshape(-1, 1)
```

- 원래 데이터 샘플과 회귀선과의 차이



3. 성능 지표

위의 실습에서 `model.score()`을 이용해 모델의 오류를 계산해 보면 0.58983206322300 61이라는 값이 나온다.

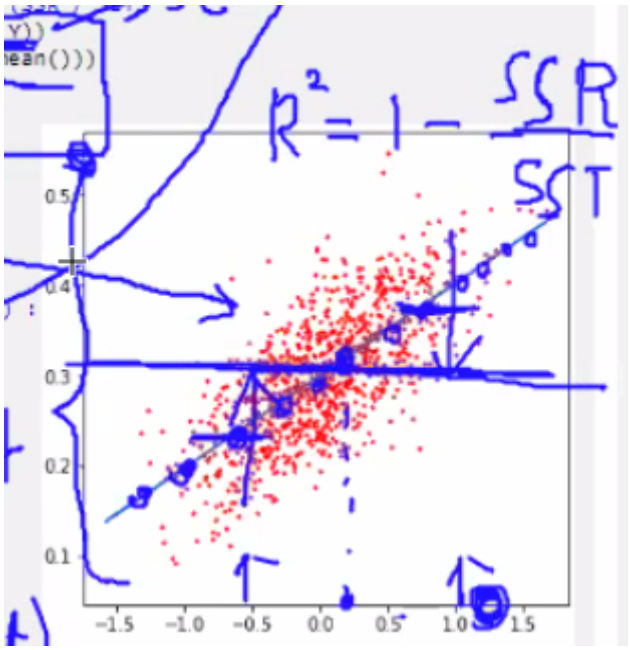
그런데 이러한 회귀 모델에서는 성과를 어떻게 측정하는 것일까? 분류 문제에서는 실제 라벨과 예측 라벨이 얼마나 같은지를 기준으로 정확도를 측정한다. 그런데 회귀의 경우는 예측 데이터 값이 전부 다 실수 값이기 때문에 분류 문제와 같은 방식으로 정확도를 측정할 수 없다.

따라서 회귀 모형에서는 통계학에서의 `R-square` 점수를 성능 지표로 활용한다.

$$R^2 = 1 - \frac{SSR}{SST}$$

- `SSR` : 회귀 모델로 설명이 되지 않은 변동. 회귀 모델로 예측한 값과 실제 데이터 값의 평균의 차이.
- `SSE` : 회귀 모델로 설명할 수 있는 변동. 실제 데이터값과 회귀 모델로 예측한 값 차이의 제곱합.
- `SST` : `SSR`과 `SSE`의 합. 총 변동.

식을 보면 `R-square` 결정계수는 회귀 모델이 모델로서 어느 정도의 변동을 설명할 수 있는지를 나타내는 수치임을 알 수 있다. (X축 데이터로 종속 변수를 얼마나 잘 설명할 수 있는지를 나타내는, 설명력을 측정하는 지표이다.)



실습 2. Boston: 정확도 측정

위에서의 설명력 지표를 직접 계산해 보고, `model.score()`와 동일하게 도출되는지 확인하자. `NumPy` 모델을 사용하면 된다.

```
y_pred = lr_model.predict(X_test)

# sklearn model R2
score = lr_model.score(X_test, y_test)

# manual R2
SSR = np.sum(np.square(y_pred - y_test))
SST = np.sum(np.square(y_test = y_test.mean()))
R2 = 1 - SSR / SST
```

두 값이 같은 것을 알 수 있다. 사이킷런에서 `model.score()`는 분류 모델과 회귀 모델에서 반환하는 값이 다르다. 모델에 맞게 반환해 준다!