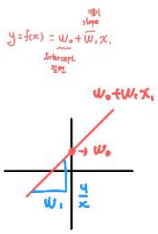
교 육 일 지

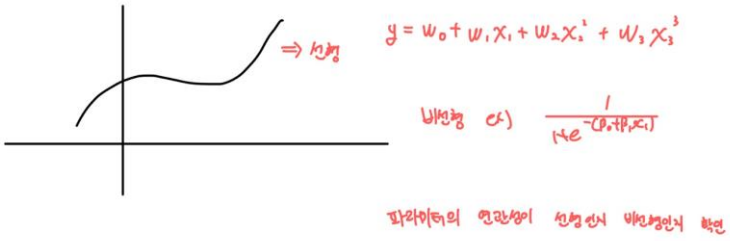
교육 제목	머신러닝 기초
교육 일시	2021.10.13
교육 장소	YGL-C6
교육 내용	

1. 선형회귀(Linear Regresstion)

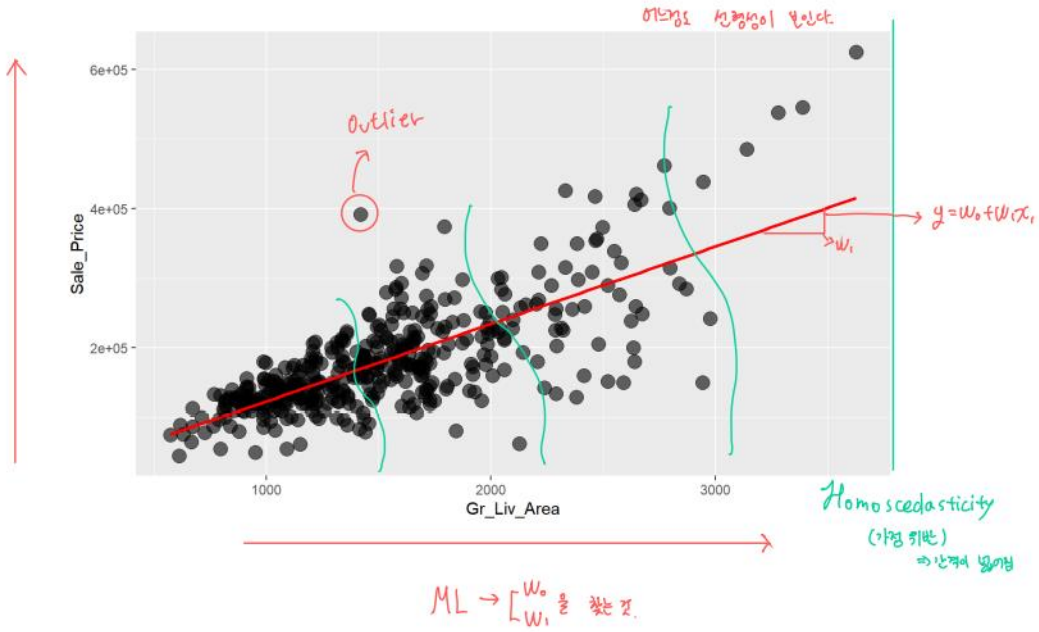
- 지도학습
- 목적변수가 연속형인 경우 사용
- 정규성 독립성, 등분산성을 만족해야함
- Feature가 하나인 경우 단순회귀(Simple)



- Feature가 여러개인 경우 중회귀 (Multiple)
- 2차항 이상이 포함된 경우 다항회귀(Polynomial Regression)



Gr_Liv_Area 와 Sale_Price 간의 관계



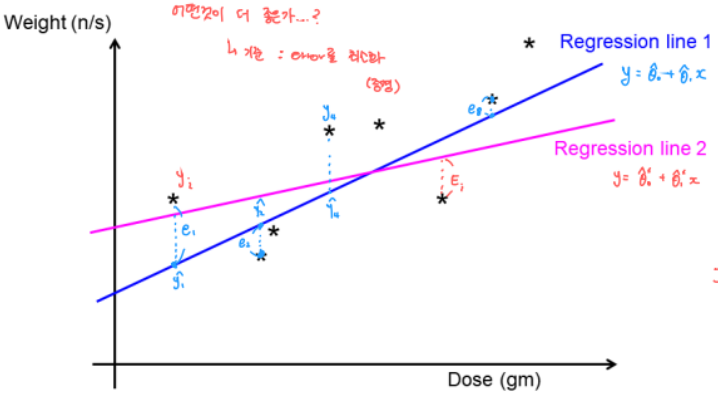
최소제곱추정법

최소제곱추정량 (Ordinary least square estimation)

↳ OLS

- Linear model, $\hat{y}(x_i) = \hat{\theta}_0 + \hat{\theta}_1 x_i$

How to fit regression line



$$\frac{J(\theta_0,\theta_1)}{\partial\theta_0} = 0 \quad \frac{J(\theta_0,\theta_1)}{\partial\theta_1} = 0 \Rightarrow \frac{J(\theta_0,\theta_1)}{\partial\theta_0} = \sum_{i=1}^{n} 2(y_i - \theta_0 - \theta_1 x_i)\cdot(-1) = 0 \quad \frac{J(\theta_0,\theta_1)}{\partial\theta_1} = \sum_{i=1}^{n} 2(y_i - \theta_0 - \theta_1 x_i)(-x_i) = 0$$

$R_1 = e_1 + e_2 + e_3 + \cdots + e_n + e_n$

$R_1 = \sum_{i=1}^{n} e_i$ → $\sum_{i=1}^{n} e_i^2$

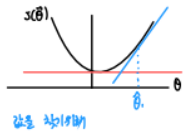
$R_2 = \sum_{i=1}^{n} E_i$ → $\sum_{i=1}^{n} E_i^2$

⇒ 음수가 발생가능

⇒ 이것을 해결 제곱

$$\min \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_i)^2$$

$$J(\vec{\theta}) = \min_{\theta_0,\theta_1} \sum_{i=1}^{n} (y_i - \hat{\theta}_0 - \hat{\theta}_1 x_1)^2$$



minimum의 값을 찾아내기 위해 0 만듦 ⇒ 미분해서 0이 되는점

손실함수 (Loss function, J)

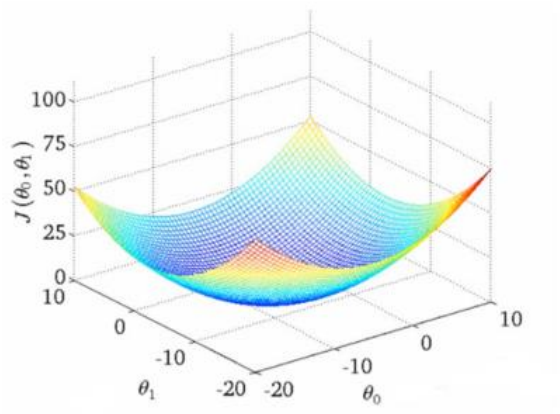
$J(\theta) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - \hat{\theta}_0 - \hat{\theta}_1 x)^2 = \sum_{i=1}^{n} e_i^2$

- $\hat{y}_i = \hat{\theta}_0 + \hat{\theta}_1 x_i$
- Sum of Squares of the Errors (SSE) = $\sum_i^n e_i^2$

- Goal is to solve for $\hat{\theta}_0$ and $\hat{\theta}_1$ to minimize the objective function.

$$\hat{\theta}_o, \hat{\theta}_1 = argmin_{\theta_0,\theta_1} \sum_i^n e_i^2 = argmin_{\theta_0,\theta_1} J(\theta)$$

Convex function



- $\frac{\partial J(\theta)}{\partial \hat{\theta}_0} = 0, \quad \hat{\theta}_0 = \overline{y} - \hat{\theta}_1 \overline{x}$
- $\frac{\partial J(\theta)}{\partial \hat{\theta}_1} = 0, \quad \hat{\theta}_1 = \frac{\sum_i^n (x_i - \overline{x})(y_i - \overline{y})}{\sum_i^n (x_i - \overline{x})^2}$
- $\sum_i^n (x_i - \overline{x})(y_i - \overline{y}) = S_{xy}$ $\overline{y} - \hat{\theta}_1 \overline{x}$
- $\sum_i^n (x_i - \overline{x})^2 = S_{xx}$ $\frac{\sum_i^n (x_i - \overline{x})(y_i - \overline{y})}{\sum_i^n (x_i - \overline{x})^2}$

파이썬으로 직접 구현해보고 sklearn 모듈의 LinearRegression 모델과 비교해보기

Simple Linear Regression

- 선형회귀 최소제곱추정법의 Loss_Fucntion을 직접 구현해보기
- 구현한 결과를 `sklearn`의 반환값과 비교해보기

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
x = np.arange(1,30,1,dtype = np.int16)
x
```

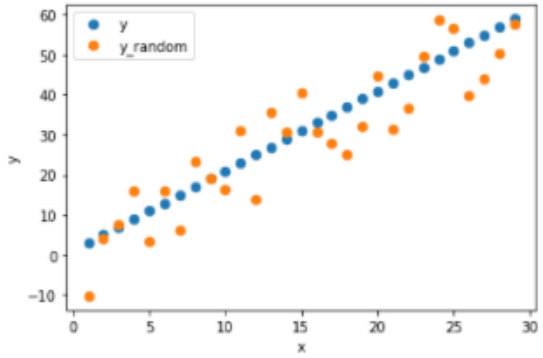
```
7]: array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29], dtype=int16)
```

```
y = 2*x + 1
y

# Random Normal Distribution
y_random = y + np.random.normal(loc = 0, scale = 8 , size = len(y))  # location  = 0 , sta = 8
y_random
```

```
3]: array([-10.22406365,   4.12800259,   7.55968155,  16.16012052,
         3.29790979,  16.03246156,   6.27664086,  23.4472299 ,
        19.25819473,  16.46589217,  31.12573442,  13.89857975,
        35.71119525,  30.56621783,  40.50449049,  30.71386538,
        27.8633732 ,  25.12169442,  32.05750377,  44.58083399,
        31.37383174,  36.57414409,  49.47223494,  58.80118812,
        56.54798789,  39.99139556,  44.1233851 ,  50.26133161,
        57.68540812])
```

```
plt.plot(x,y,',',markersize = 13,label = 'y')
plt.plot(x,y_random,',',markersize = 13,label = 'y_random')
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.show()
```



Parameter Estimation

y_random을 실제 데이터라 생각하고 target 찾기

$$\beta_1 = \frac{S_{xy}}{S_{xx}}$$

$$\beta_0 = \overline{y} - \beta_1 \overline{x}$$

```
Sxy = (x - np.mean(x)) * (y_random - np.mean(y_random))
Sxx = (x - np.mean(x)) ** 2

Sxy = Sxy.sum()
Sxx = Sxx.sum()

print(f'Sxy: {Sxy}\nSxx : {Sxx} ')

beta_1 = Sxy / Sxx
beta_0 = np.mean(y_random) - beta_1*np.mean(x)

print(f'beta_0 : {beta_0.round(3)}\nbeta_1 : {beta_1.round(3)}')
```

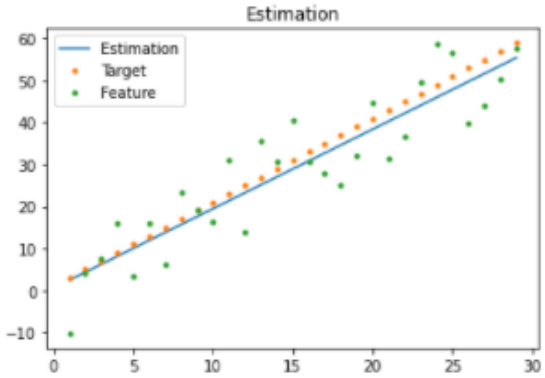
```
Sxy: 3841.363778088995
Sxx : 2030.0
beta_0 : 0.56
beta_1 : 1.892
```

Visualization Estimation

- 대부분의 경우 target를 알 수 없다

```
Result = beta_1 * x + beta_0

plt.plot(x,Result,label = 'Estimation')
plt.plot(x,y,'.',label = 'Target')
plt.plot(x,y_random,'.',label = 'Feature')
plt.title("Estimation")
plt.legend()
plt.show()
```



Sklearn을 활용해서 앞서 구현한 결과와 비교해보기

```
from sklearn.linear_model import LinearRegression

lr = LinearRegression() # 객체 생성
```

```
x_2d =x.reshape(-1,1) # x의 demention을 2d로 변경
print(np.shape(x), np.shape(x_2d))
```

```
(29,) (29, 1)
```

model 에 값을 넣어서 값 확인

- sklearn 모듈을 사용해서 쉽게 Estimation 가능

LinearRegression.coef_ = β_1
LinearRegression.intercept_ = β_0

```
lr.fit(x_2d, seed_y_random)
print(f'coef : {lr.coef_} ₩n₩nbefore_beta_1 : {seed_beta_1}₩n₩nintercept : {lr.intercept_}₩nbefore_bet
```

```
coef : [2.01038216]
before_beta_1 : 2.010382155133716

intercept : 1.0308305920375211
before_beta_0 : 1.0308305920375282
```

2. 머신러닝

규칙을 일일이 프로그래밍하지 않아도 자동으로 데이터에서 규칙을 학습하는 알고리즘

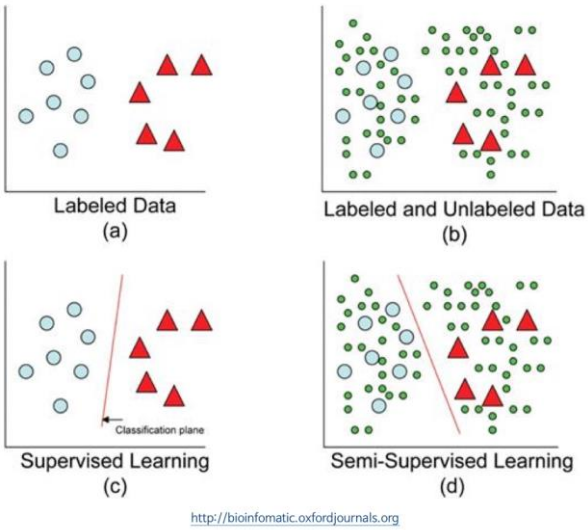
머신러닝은 통계학과 깊은 관련이 있다.

2.1 기계학습의 분류

- Supervised Learning : 학습 데이터에 레이블(label)포함
  - Classification (분류)
  - Regresstion(회귀)
- Unsupervised Learning : 학습데이터에 레이블(Label) 없음
  - Clustering(군집화)
  - Visualization & Dimensionality Reduction(차원 감소)

    ex) PCA(Principal Component Analysis) : 정보량 손실을 최소한으로 Feature차원 감소
  - Association Rule Learning
- Semi-Supervised Learning : 레이블이 있는 데이터 & 없는 데이터

- Reinforcement Learning : Agent가 state의 reward와 penalty를 주며 최고의 Reward를 받도록 학습



3. 모델링 과정

3.1 Split Training Set and Test set

ML 모델의 학습 및 성능 평가를 위해서 자료를 분할

- Training Set

  모델의 알고리즘 Learning에 사용될 Feature들을 결정

  주로 전체 자료수의 70%로 설정

  Training Set : 모델의 알고리즘 Learning

  Validation Set : 학습 시 hyperparameter 조정 및 overfitting 방지

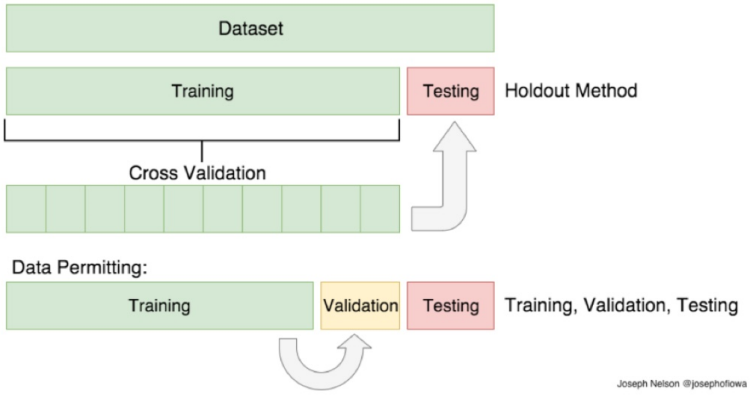
- Test Set

  최종 선택된 모델의 성능 평가 (전체 자료수의 30%)

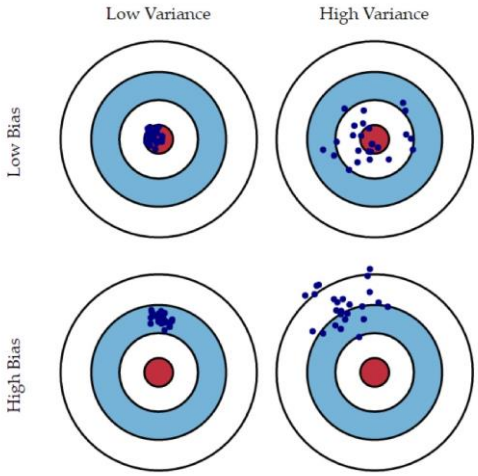
  자료의 양이 적을 경우 생략 가능

일반적인 ML 예측 과정

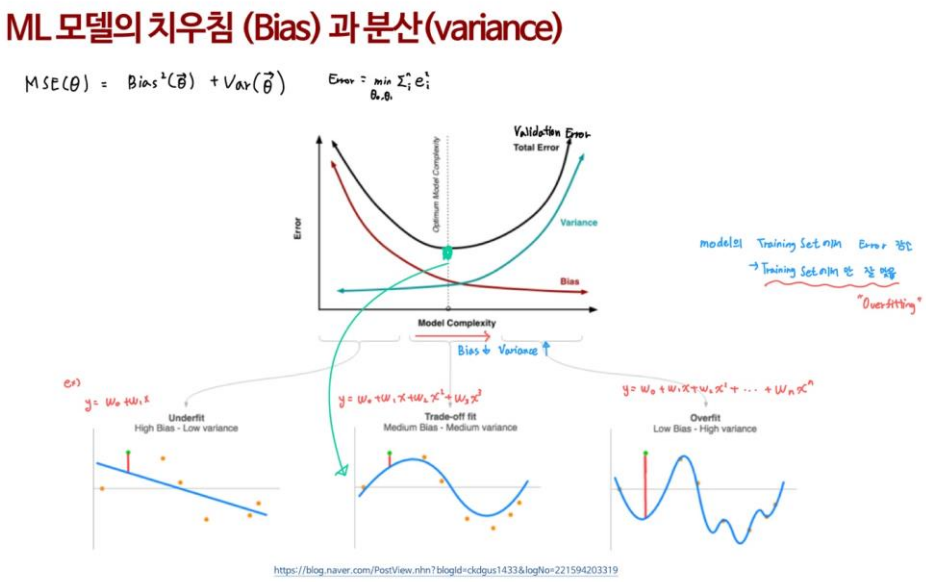
- 학습세트 (Training set): 머신러닝 모델을 학습할 때 사용
- 검증세트 (Validation set): 하이퍼 파라메터 결정할 때
- 테스트 세트 (Test set) : 학습된 모델을 평가할 때



4. ML 모델의 Bias(치우침)과 Variance(분산)



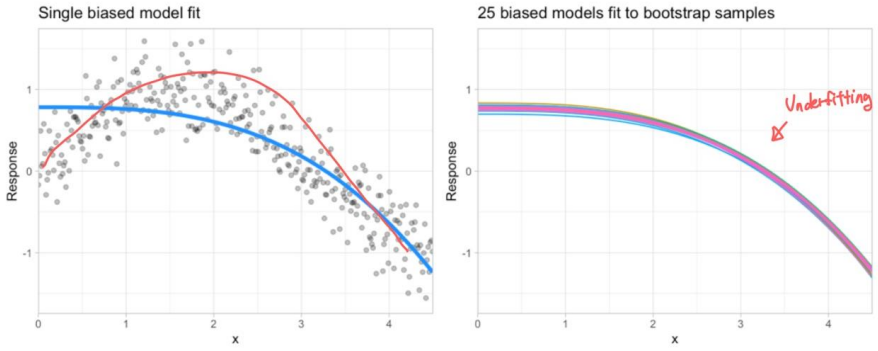
https://miro.medium.com



4.1 Bias(치우침)

Bias는 모델의 실제값 과 예측치 간의 차이를 의미

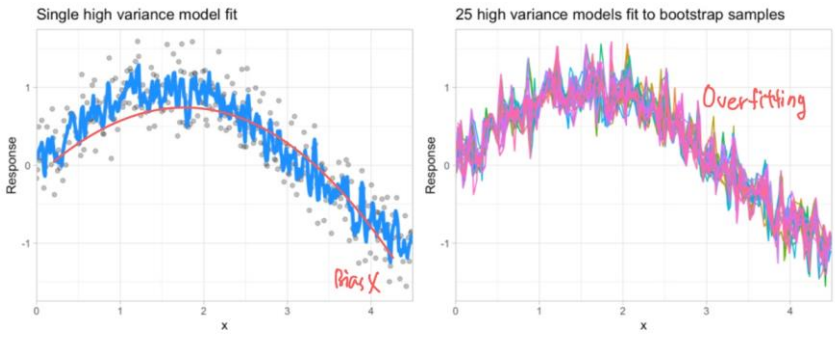
Underfitting(과소적합)은 Bias가 높은 모델이 발생하기 쉽다.



4.2 Variance(분산)

Variance(분산)은 주어진 데이터에서 모델 예측의 변이로 정의

Overfitting(과적합)은 분산이 높은 모델이 발생하기 쉬움.



5. Training Set과 Test Set의 분할

   무작위 샘플링(Random Sampling)

   계층화 샘플링(Stratified Random Sampling)

   ⇨ 무작위로 샘플링을 할 경우 데이터의 분포에 맞지 않게 분리 되어 학습이 잘 안 될 수도 있다.

   ⇨ 데이터를 분할 할 때 데이터의 분포에 맞게 Training Set와 Test set을 분리 시켜줘야 한다.

6. 머신러닝 모델 평가

   K-fold 교차검증(k-fold Cross Validation(CV))

   - k-fold교차 검증(일명 k-fold CV)는 훈련 데이터를 동일한 크기의 k 그룹(k-fold)로 무작위로 나누는 리샘플링 방법
   - k-fold CV의 추정치는 k-test Error를 평균화하여 계산