# 4강. Regression

2023.01.05.

양희철
hcyang@cnu.ac.kr

# Regression

## Linear regression

- 쌍으로 관찰된 연속형 변수들 사이의 관계에 있어서 한 변수를 원인으로 하고 다른 변수들을 결과로 하는 분석
- 독립변수와 종속변수 사이 선형식을 구하고 그 식을 이용하여 변수값들이 주어졌을 때 종속변수의 변수 값을 예측하는 분석방법
- 독립변수의 개수에 따라 단순 선형과 다중 선형으로 구분

# Regression

## Linear regression

- Example: house pricing prediction
  - Problem formulation
    - $x^{(i)}$: input variables or input features (ex. living area)
    - $y^{(i)}$: an output or a target variable (ex. price)
    - $\left(x^{(i)}, y^{(i)}\right)$: a training example
    - $\left\{\left(x^{(i)}, y^{(i)}\right); i = 1, \ldots, m\right\}$: a training set

# Regression

## Linear regression

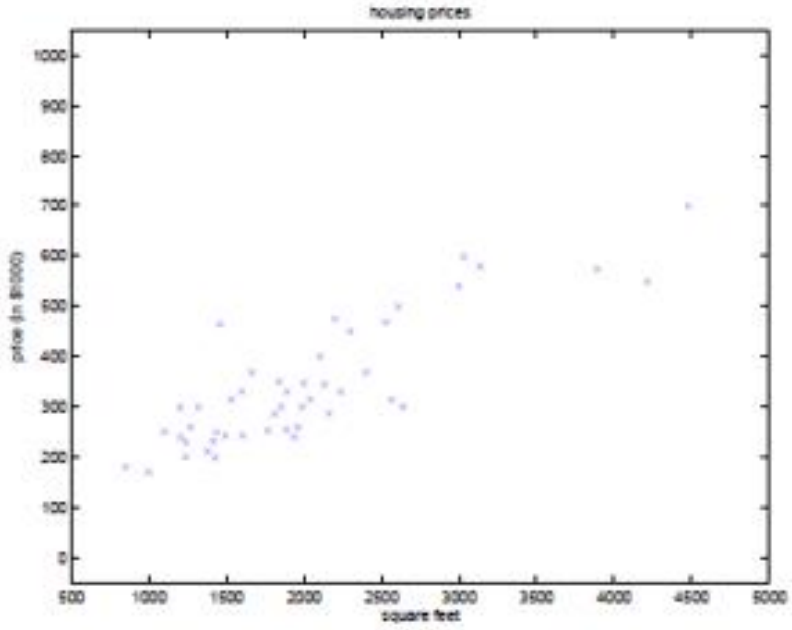
- Example: house pricing prediction
  - Linear model
    - $\hat{y}^{(i)} = w_1 x^{(i)} + w_0$: linear model with two parameters $w_1$ and $w_0$
  - Prediction error
    - $e^{(i)} = y^{(i)} - \left(w_1 x^{(i)} + w_0\right)$ for $i = 1, \ldots, m$.
    - $e = e^{(1)} + \cdots + e^{(m)}$
  - We determine $w_1$ and $w_0$ which minimize $e$.

# Regression

## Linear regression

- Example: house pricing prediction
  - Portland의 living area와 price간의 관계

| Living area ($feet^2$) | Price (1000$s) |
|---|---|
| 2104 | 400 |
| 1600 | 330 |
| 2400 | 369 |
| 1416 | 232 |
| 3000 | 540 |
| ⋮ | ⋮ |



# Regression

## Linear regression

- Example: house pricing prediction
  - We can solve it by least squares (LS) method
    - Linear model

$$\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{k=0}^{1} w_k x_k = \boldsymbol{w}^T \boldsymbol{x} \; (x_0 = 1)$$

    - Mean square error

$$J(\boldsymbol{w}) = \frac{1}{2m} \sum_{i=1}^{m} \left( h_{\boldsymbol{w}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right)^2$$

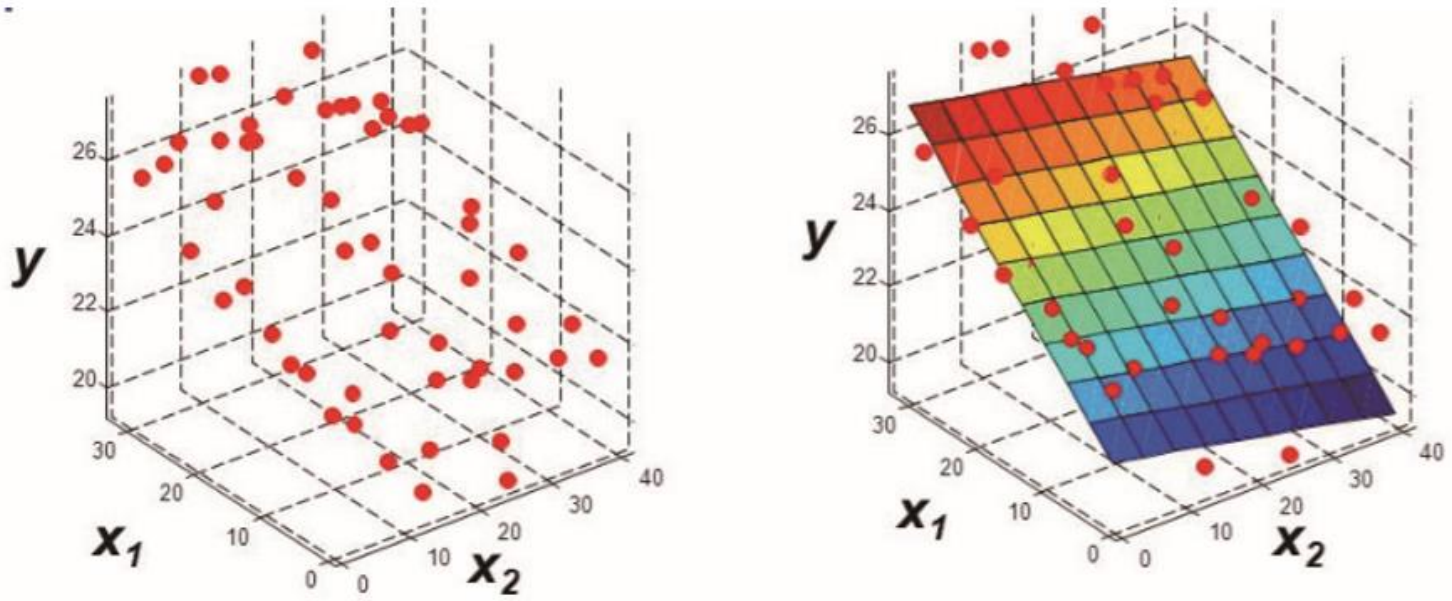
    - To minimize mean square error, solve $\frac{\partial}{\partial \boldsymbol{w}} J(\boldsymbol{w}) = 0$ for $\boldsymbol{w}$. (Hard to solve!)

# Regression

## Linear regression

- Multivariable linear regression
  - 변수가 2개 이상인 경우

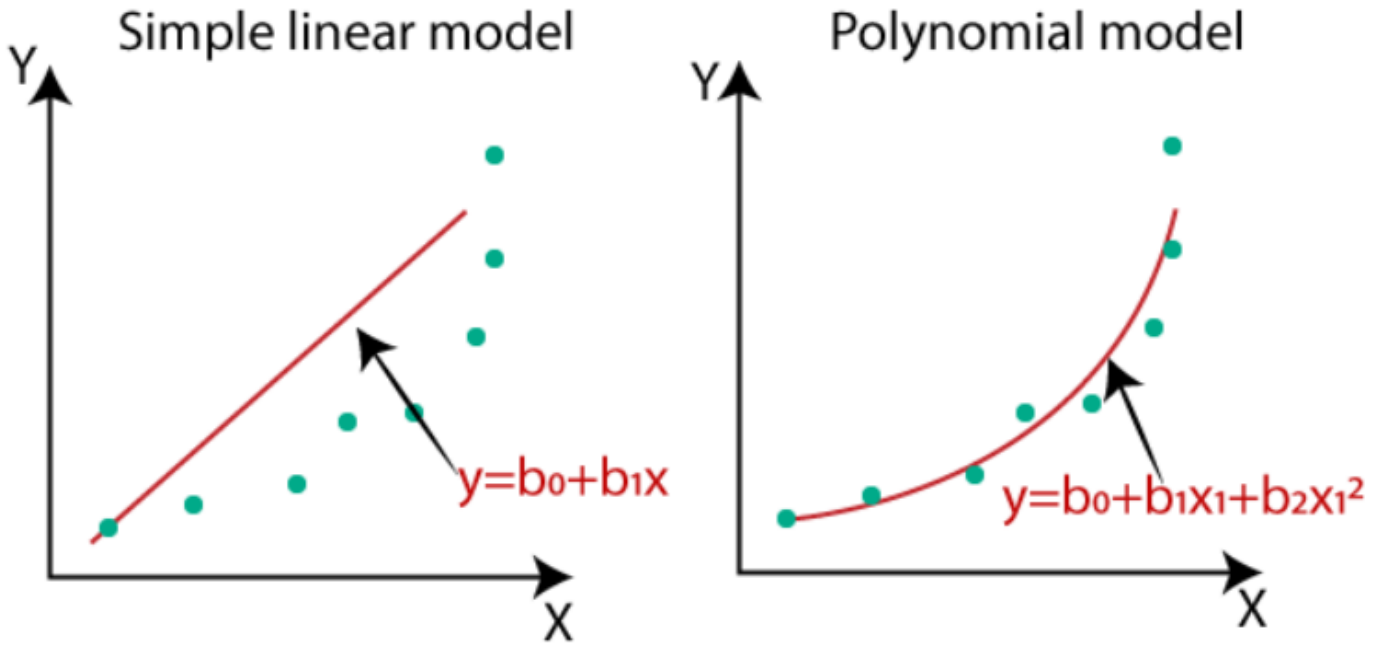
$$h_{\boldsymbol{w}}(x) = \sum_{i=1}^{n} w_i x_i = \boldsymbol{w}^T \boldsymbol{x}$$



# Regression

## Polynomial regression

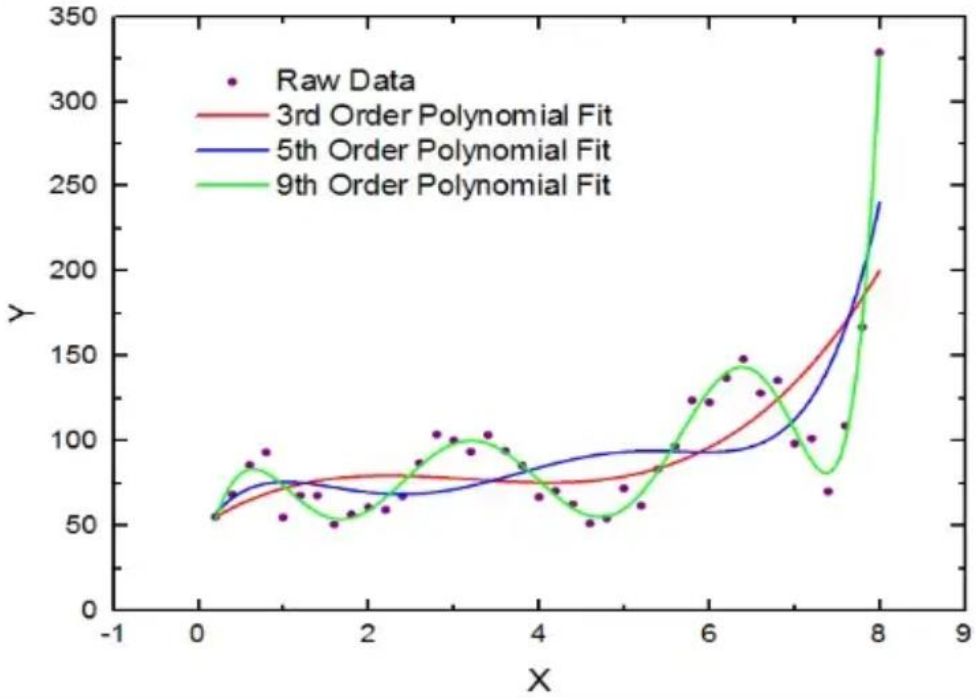
- Regression with high-order polynomials
  - $\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{j=0}^{M} w_j x^j$



# Regression

## Polynomial regression

- Regression with high-order polynomials
  - $\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{j=0}^{M} w_j x^j$



# Regression

## Linear regression

- Practice 1: linear regression
  - 키와 몸무게 데이터에 대한 선형 회귀 분석

```
import numpy as np
from sklearn import linear_model      # scikit-learn 모듈을 가져온다

regr = linear_model.LinearRegression()
```

```
X = [[164], [179], [162], [170]]      # 다중회귀에도 사용하도록 함
y = [53, 63, 55, 59]                  # y = f(X)의 결과
regr.fit(X, y)
```

# Regression

## Linear regression

- Practice 1: linear regression
  - 키와 몸무게 데이터에 대한 선형 회귀 분석

```
coef = regr.coef_                # 직선의 기울기
intercept = regr.intercept_      # 직선의 절편
score = regr.score(X, y)         # 학습된 직선이 데이터를 얼마나 잘 따르나

print("y =", coef, "* X + ", intercept)
print("The score of this line for the data: ", score)
```

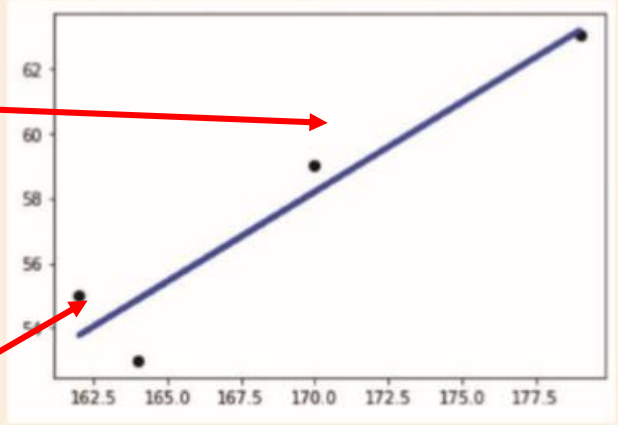
```
y = [0.55221745] * X +  -35.686695278969964
The score of this line for the data:  0.903203123105647
```

# Regression

## Linear regression

- Practice 2: linear regression

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import linear_model  # scikit-learn 모듈을 가져온다

regr = linear_model.LinearRegression()

X = [[164], [179], [162], [170]]  # 선형회귀의 입력은 2차원으로 만들어야 함
y = [53, 63, 55, 59]              # y = f(X)의 결과값
regr.fit(X, y)

# 학습 데이터와 y 값을 산포도로 그린다.
plt.scatter(X, y, color='black')

# 학습 데이터를 입력으로 하여 예측값을 계산한다.
y_pred = regr.predict(X)

# 학습 데이터와 예측값으로 선그래프로 그린다.
# 계산된 기울기와 y 절편을 가지는 직선이 그려진다
plt.plot(X, y_pred, color='blue', linewidth=3)
plt.show()
```



# Regression

## Linear regression

- Practice 3: multivariable linear regression

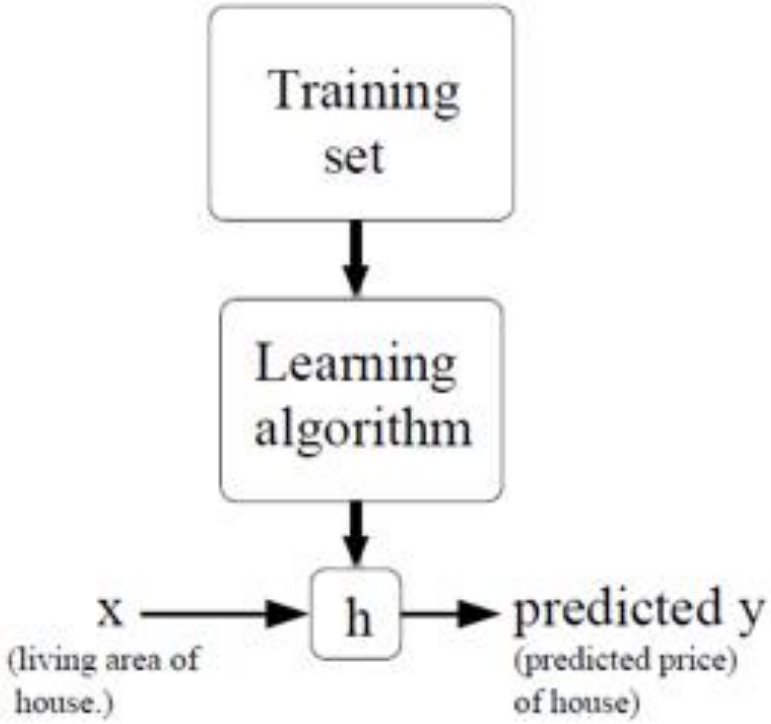
```
import numpy as np
from sklearn import linear_model

regr = linear_model.LinearRegression()

# 남자는 0, 여자는 1을 넣어 차원을 추가하였음
# 입력데이터를 2차원으로 만들어야 함
X = [[164, 1],[167, 1],[165, 0],[170, 0],[179, 0],[163, 1],[159, 0],[166, 1]]
y = [43, 48, 47, 66, 67, 50, 52, 44]      # y 값은 1차원 데이터
regr.fit(X, y)          # 학습
print('계수 :', regr.coef_ )
print('절편 :', regr.intercept_)
print('점수 :', regr.score(X, y))
print('은지와 동민이의 추정 몸무게 :', regr.predict([[166, 1], [166, 0]]))
```

# Regression

## Machine learning based regression

- Recap: house pricing prediction
  - 예시: Portland의 living area와 price간의 관계

| Living area ($feet^2$) | Price (1000$s) |
|---|---|
| 2104 | 400 |
| 1600 | 330 |
| 2400 | 369 |
| 1416 | 232 |
| 3000 | 540 |
| ⋮ | ⋮ |

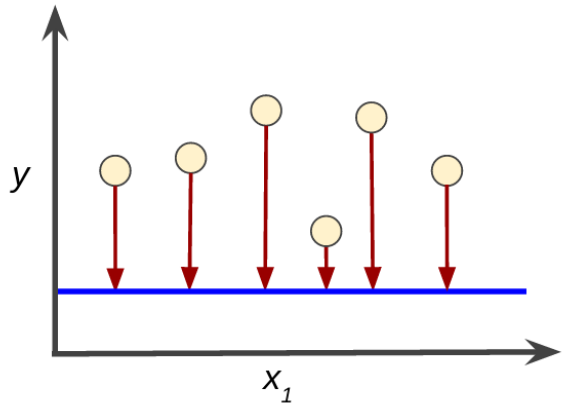


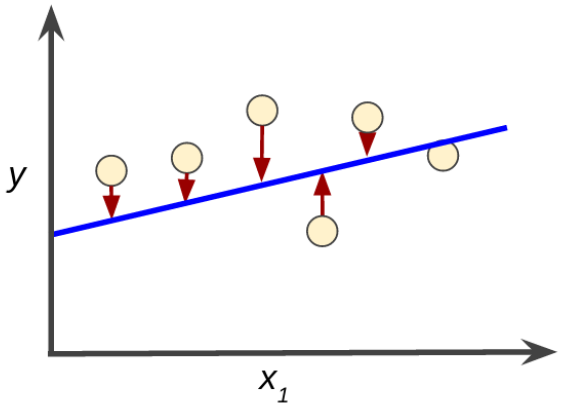
# Regression

## Machine learning based regression

- 손실 함수(Loss function)
  - 선형회귀식과 실제 값의 오차
  - 선형회귀에서 평균제곱오차(mean square error)는 머신러닝 모델을 구축할 때 작을수록 원본과의 오차가 적은 것이므로 추측한 값의 정확성이 높다고 할 수 있음

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

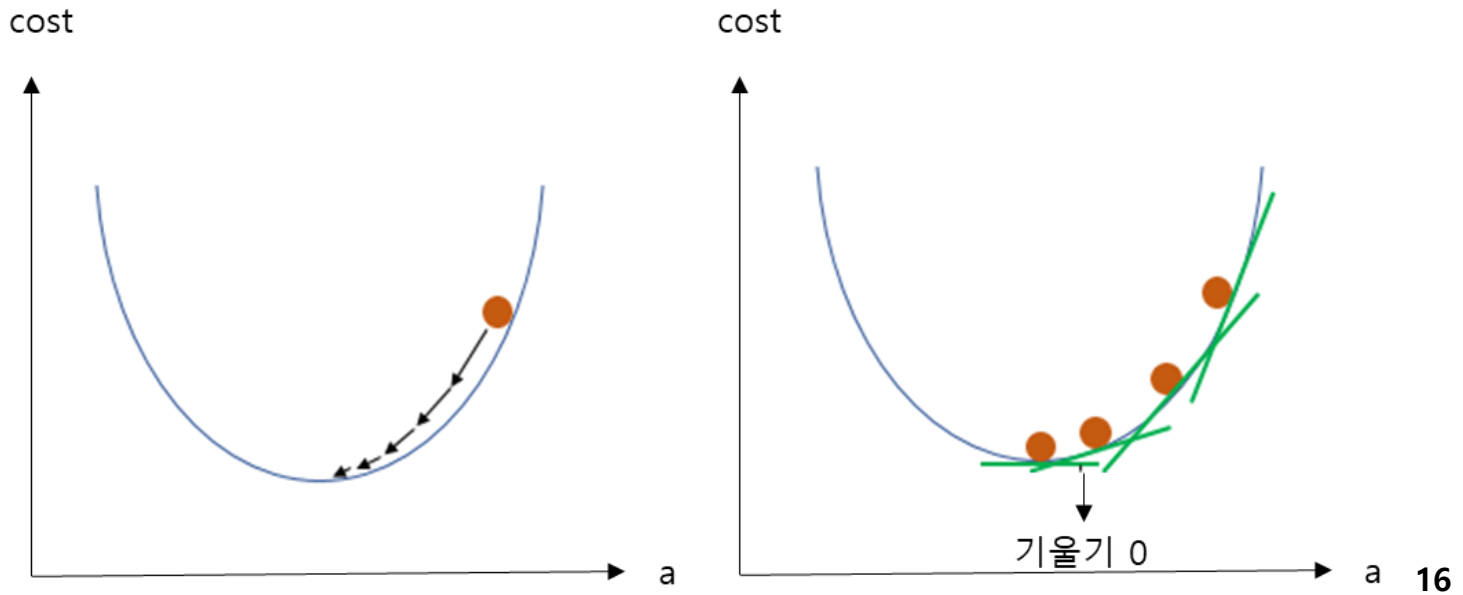


# Regression

## Machine learning based regression

- Gradient descent
  - 손실 함수를 최소화하는 매개변수를 찾는 방법
  - 손실 함수 값이 가장 낮은 지점을 찾아가도록 손실 함수의 기울기를 구해 최적값을 찾아가는 방법

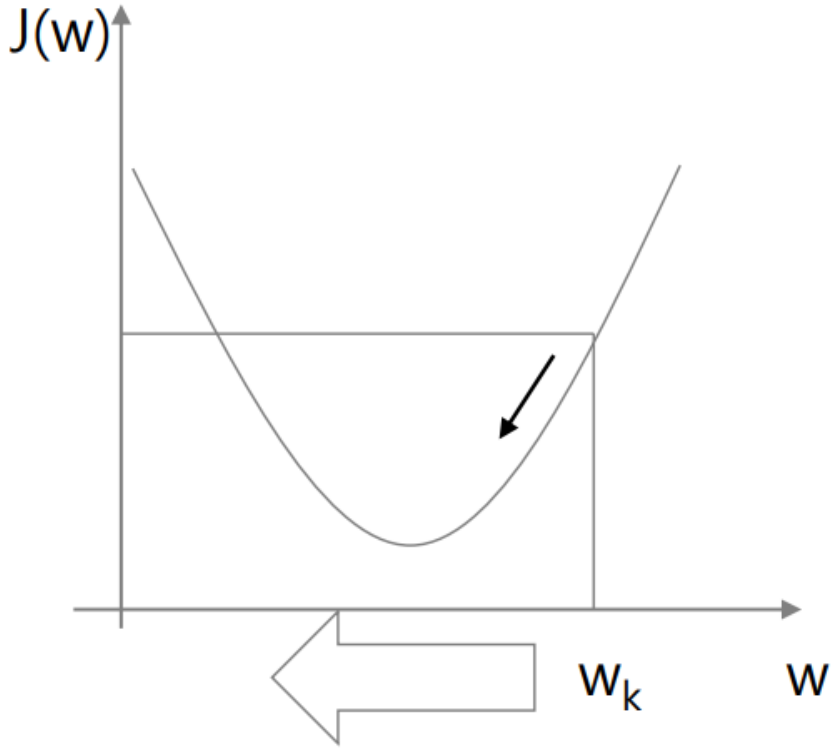


# Regression

## Machine learning based regression

- Gradient descent
  - Decision 1. Where to go?
    - Gradient descent of the objective function

$$w_{k+1} = w_k + (-\frac{\partial J(w)}{\partial w})$$



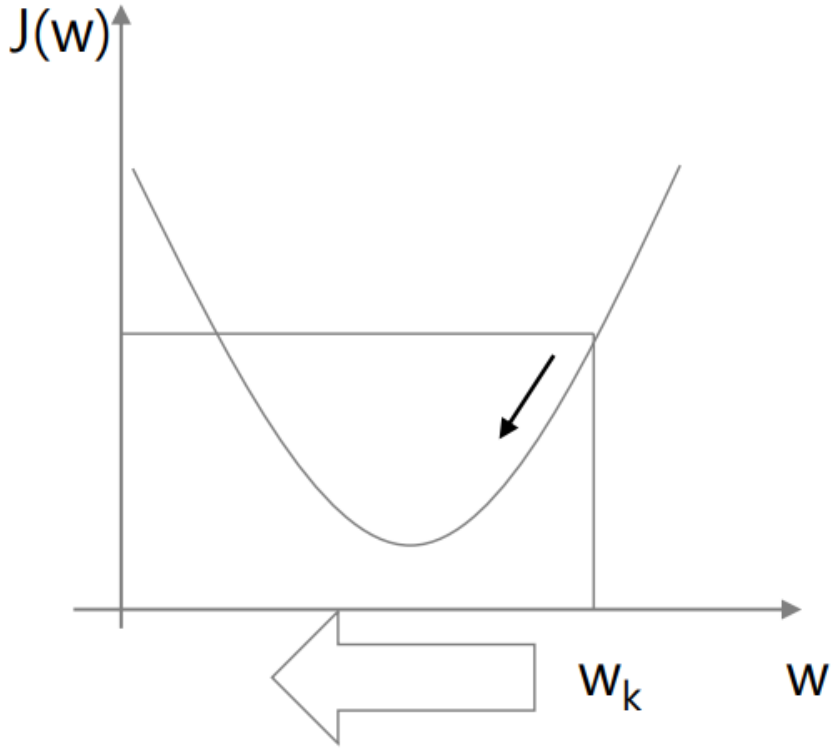
# Regression

## Machine learning based regression

- Gradient descent
  - Decision 2. How far?
    - Introduce learning rate $\eta$

$$w_{k+1} = w_k + \eta(-\frac{\partial J(w)}{\partial w})$$

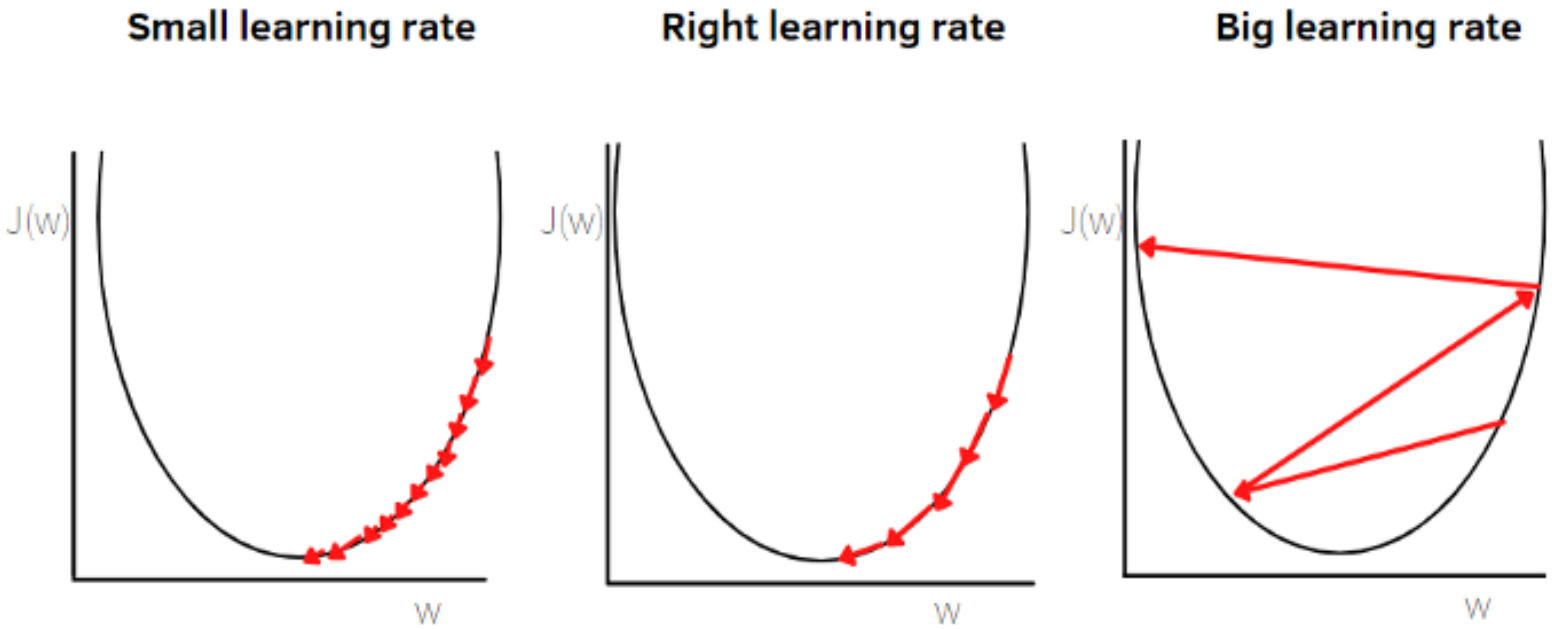
$$= w_k - \eta \frac{\partial J(w)}{\partial w}$$



# Regression

## Machine learning based regression

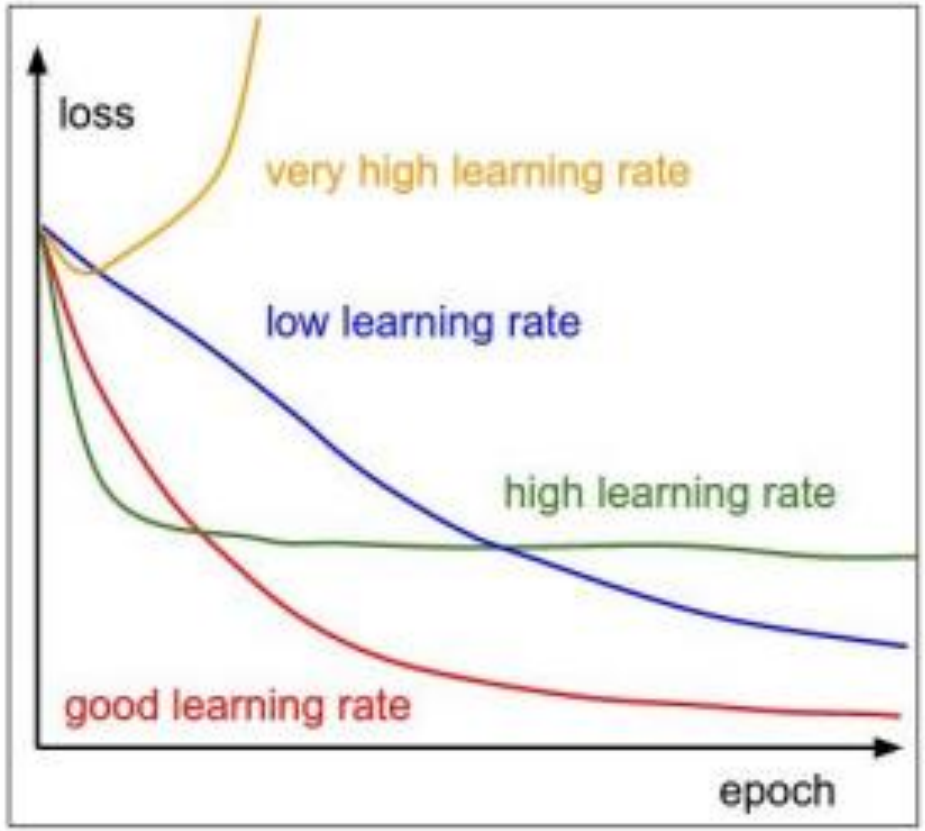
- Gradient descent
  - Decision 2. How far?
    - Learning rate의 영향



# Regression

## Machine learning based regression

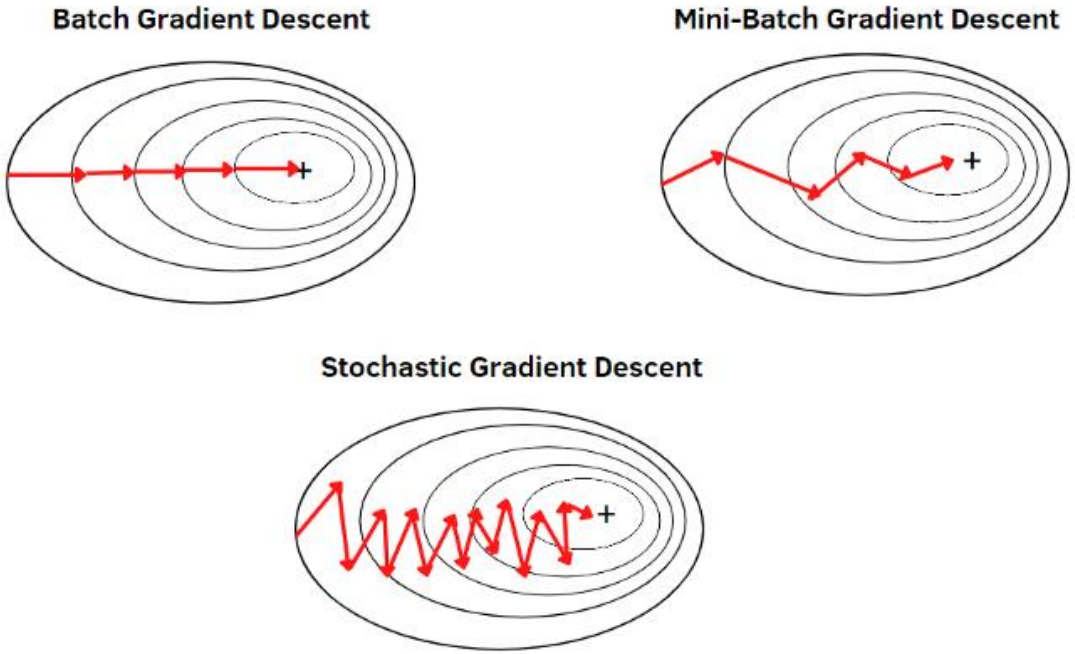
- Gradient descent
  - Decision 2. How far?
    - Learning rate의 영향



# Regression

## Machine learning based regression

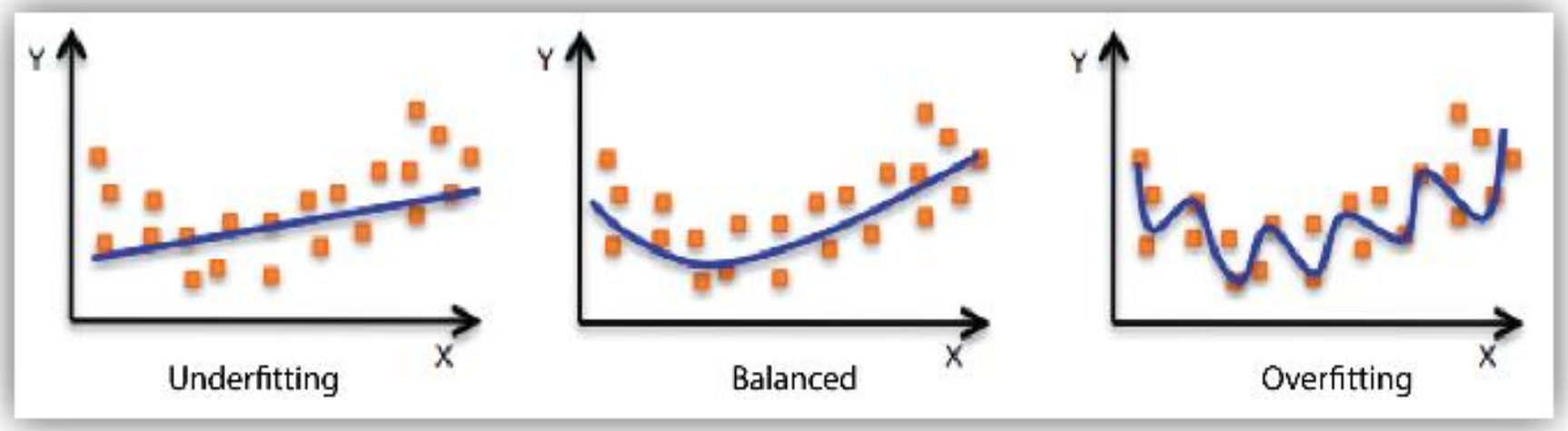
- Gradient descent
  - Batch GD vs Mini-batch GD vs SGD



# Regression

## Machine learning based regression

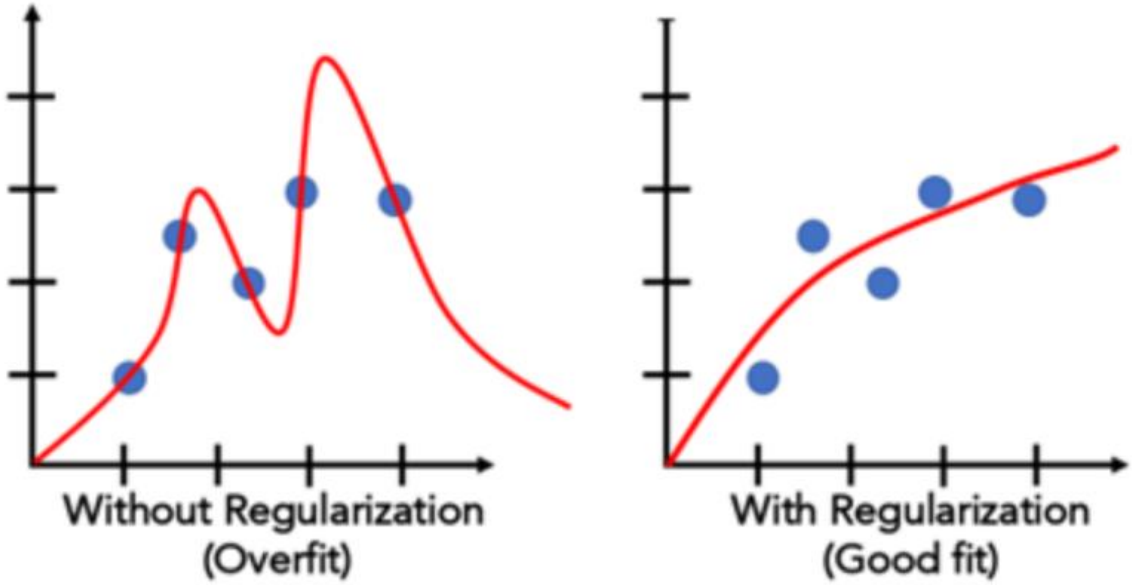
- Regularization
  - Overfitting: 모델이 훈련 데이터에 너무 잘 맞지만 일반성이 떨어지는 문제가 발생
  - Underfitting: 모델이 너무 단순해서 데이터의 포함된 의미를 제대로 학습하지 못하는 문제가 발생



# Regression

## Machine learning based regression

- Regularization
  - Regularization improves the generalization of the regression model



# Regression

## Machine learning based regression

- Regularization
  - Norm penalties: limit to the model capacity
    - L1 norm regularization: Encourages sparsity

$$\hat{\mathcal{L}}(W) = \alpha||W||_1 + \mathcal{L}(W)$$

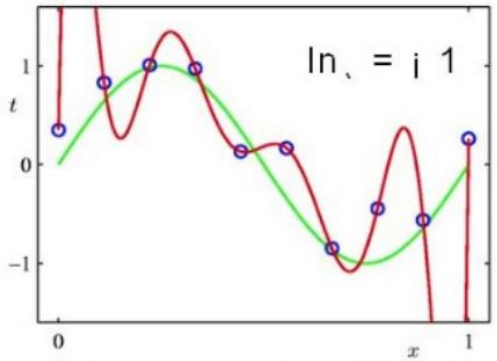
    - Squared L2 norm regularization: Encourages small weights

$$\hat{\mathcal{L}}(W) = \frac{\alpha}{2}||W||_2^2 + \mathcal{L}(W) = \frac{\alpha}{2}\sum_i \sum_j w_{ij}^2 + \mathcal{L}(W)$$

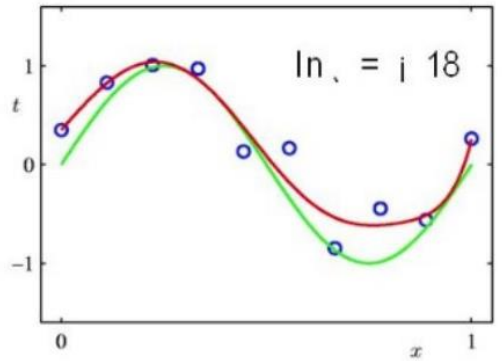
# Regression

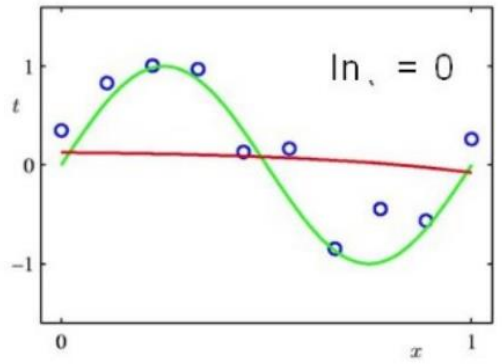
## Machine learning based regression

- Regularization
  - Norm penalties: limit to the model capacity




| | $\ln\lambda = -\infty$ | $\ln\lambda = -18$ | $\ln\lambda = 0$ |
|---|---|---|---|
| $w_0^\star$ | 0.35 | 0.35 | 0.13 |
| $w_1^\star$ | 232.37 | 4.74 | -0.05 |
| $w_2^\star$ | -5321.83 | -0.77 | -0.06 |
| $w_3^\star$ | 48568.31 | -31.97 | -0.05 |
| $w_4^\star$ | -231639.30 | -3.89 | -0.03 |
| $w_5^\star$ | 640042.26 | 55.28 | -0.02 |
| $w_6^\star$ | -1061800.52 | 41.32 | -0.01 |
| $w_7^\star$ | 1042400.18 | -45.95 | -0.00 |
| $w_8^\star$ | -557682.99 | -91.53 | 0.00 |
| $w_9^\star$ | 125201.43 | 72.68 | 0.01 |



# Regression

## Summary

- 좋은 regression 모델이란?
  - 데이터의 양
  - 모델의 특징(feature) 개수
  - 적절한 규제(regularization)