Introduction to Machine Learning

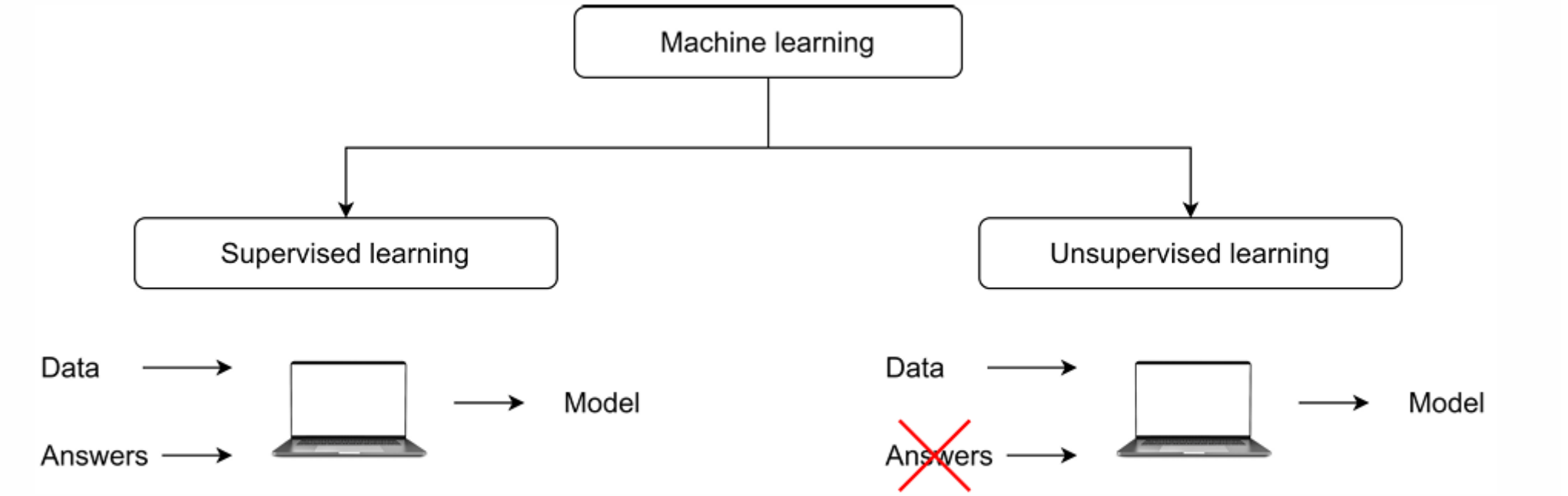
Chaehyun Maeng
mch021206@gmail.com
TNT ML
2024/09/12

TRAIN AND TEST

Contents

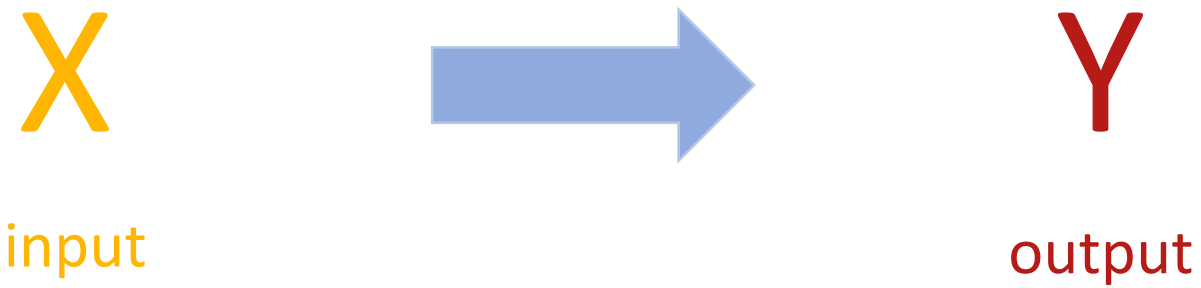
TRAIN AND TEST

Introduction to Machine Learning



TRAIN AND TEST

Supervised learning



Learns from being given “right answers”

TRAIN AND TEST

Supervised learning

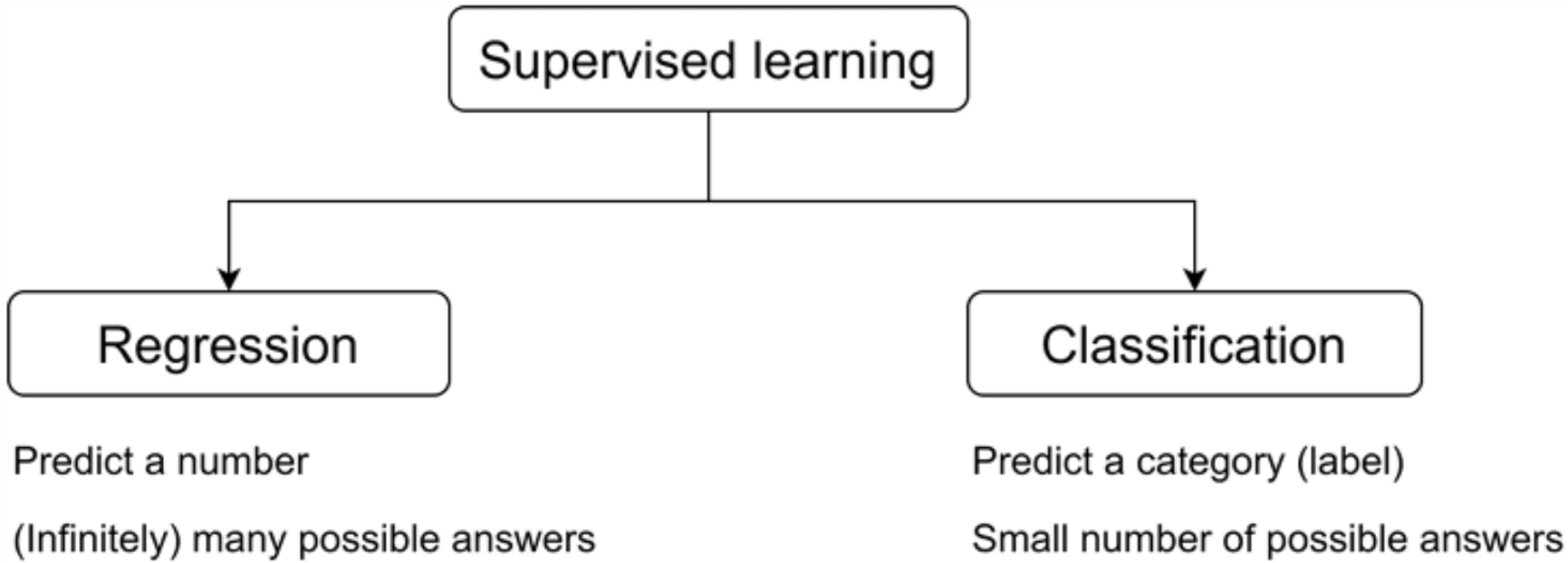
Supervised learning

A supervised learning algorithm learns an X to Y mapping from examples.

Input (X)	Output (Y)	Application
Image	Access? (0/1)	Facial recognition
Recorded speech	Text	Speech recognition
Email	Spam? (0/1)	Spam detection
Ad, user info	Click? (0/1)	Online advertising
Image, radar info	Position of other cars	Autonomous driving

TRAIN AND TEST

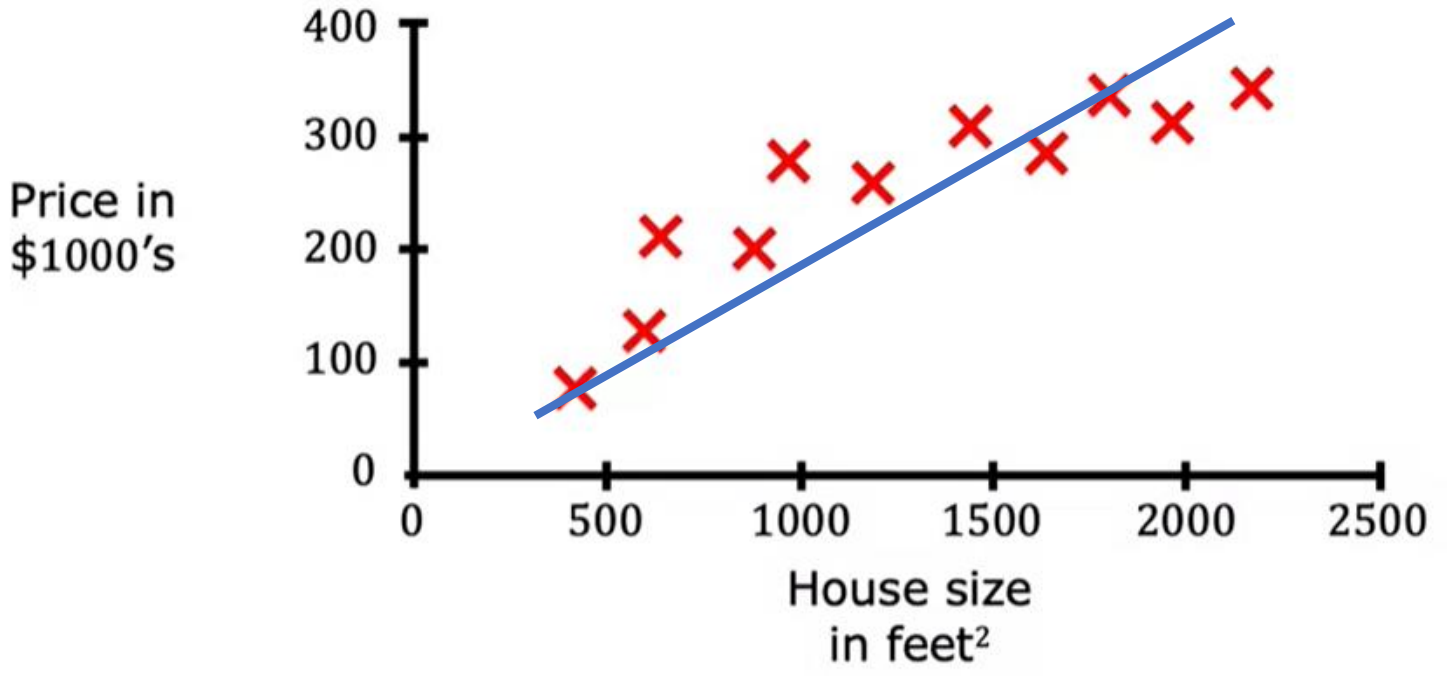
Supervised learning



TRAIN AND TEST

Regression

Housing price prediction



Regression
predict a number
infinitely many possible outputs

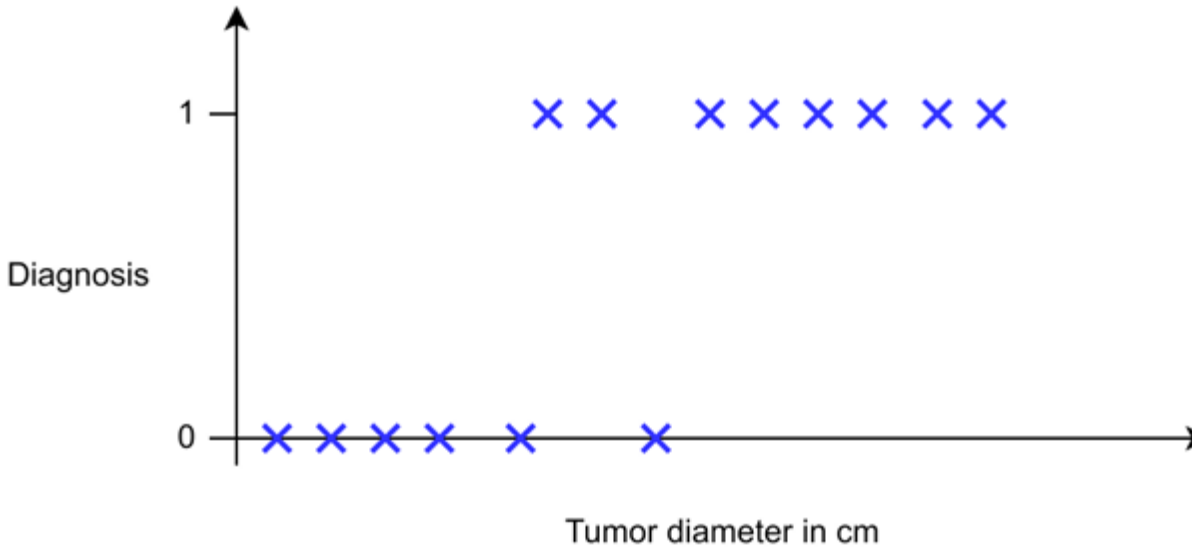
TRAIN AND TEST

Classification

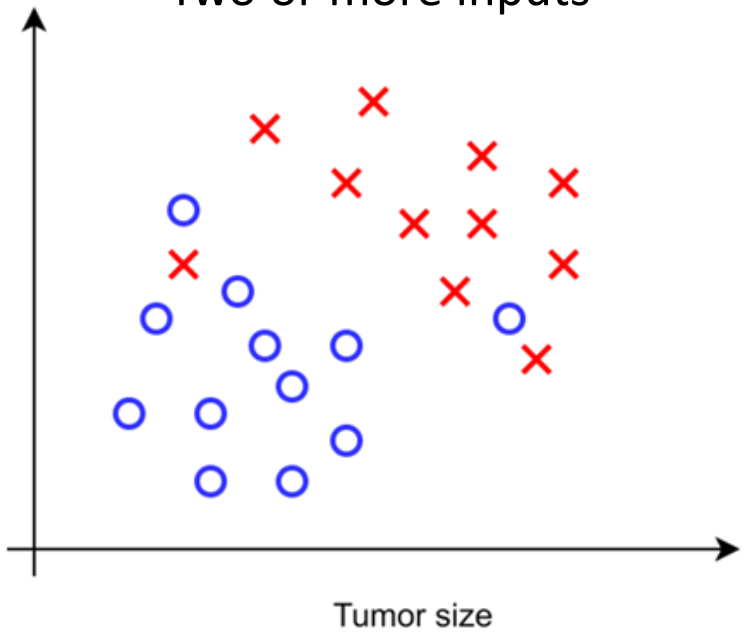
Cancer detection

Classification
predict categories
small number of possible outputs

Only one input



Two or more inputs

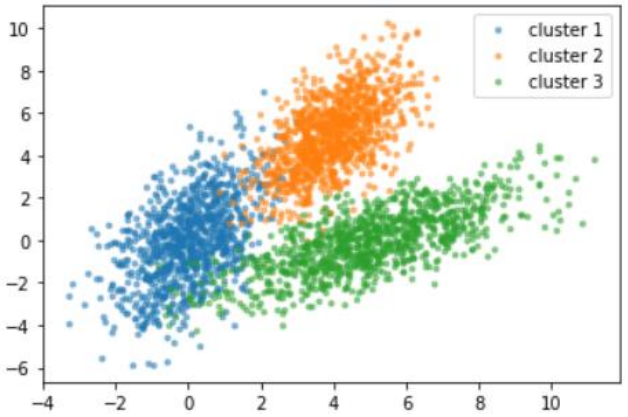


TRAIN AND TEST

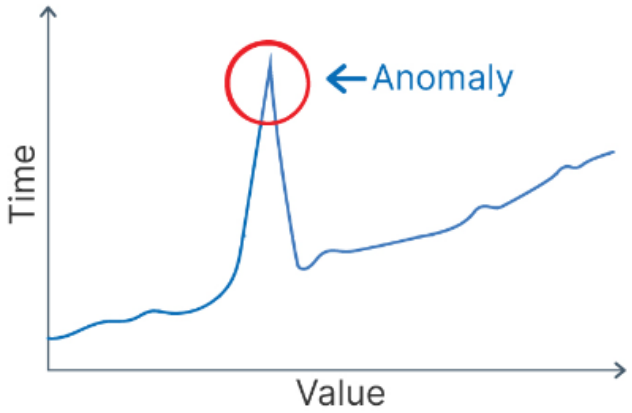
Unsupervised learning

Data only comes with inputs x, but not output labels y.

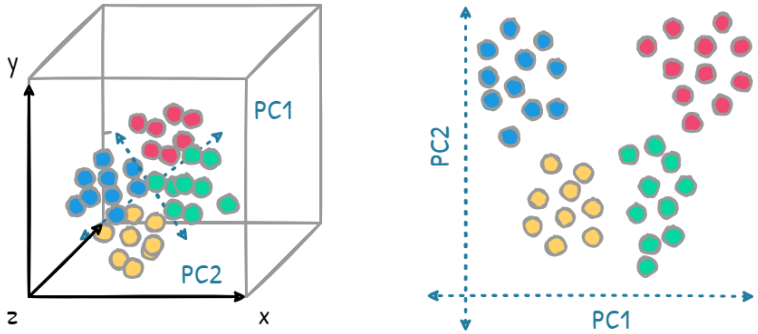
Algorithm has to find structure in the data.



Clustering



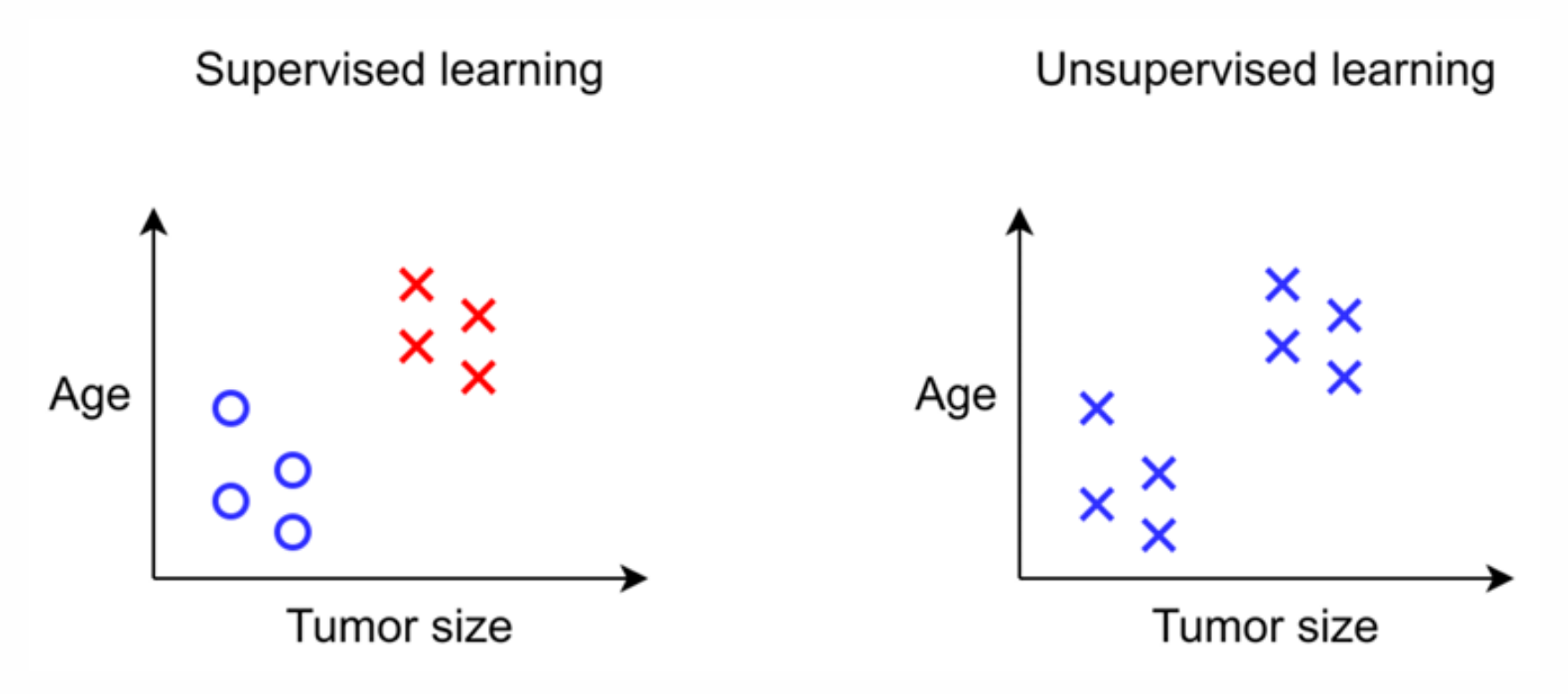
Anomaly detection



Dimensionality reduction

TRAIN AND TEST

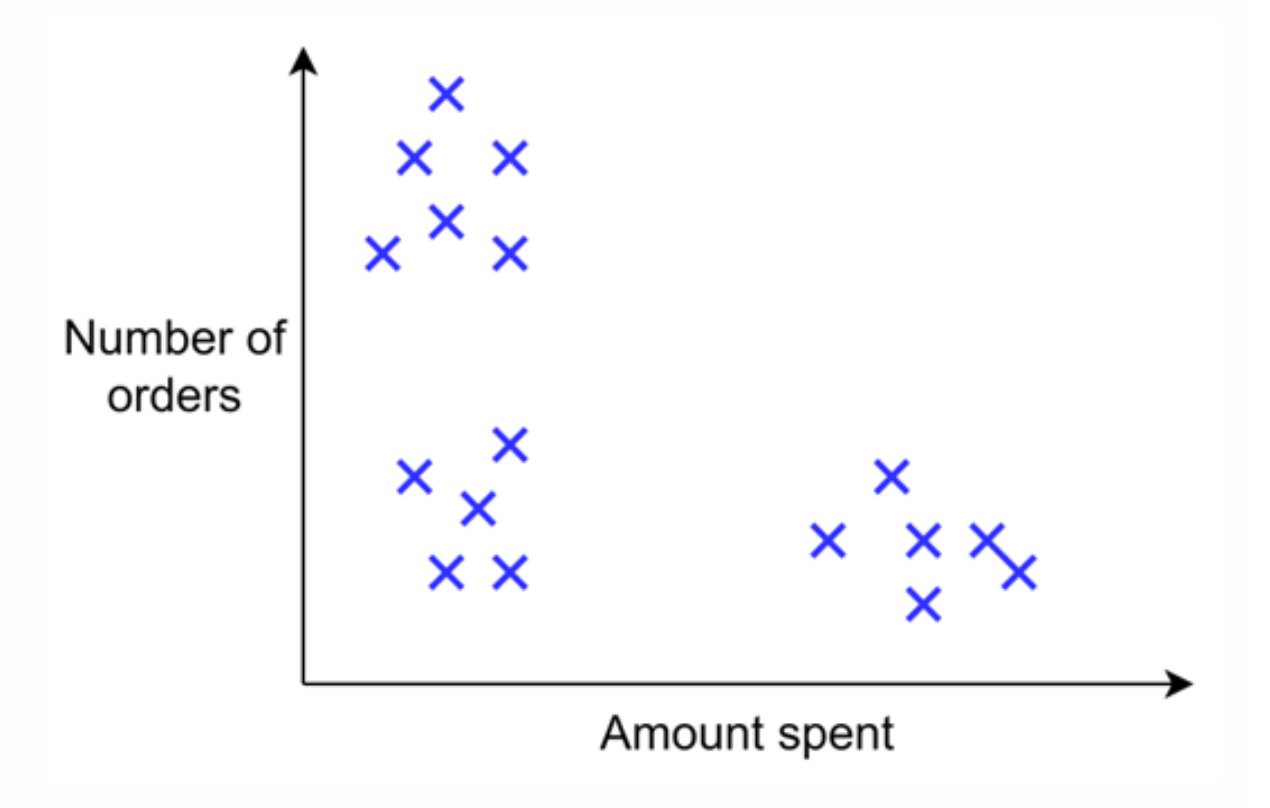
Unsupervised learning



TRAIN AND TEST

Clustering

finding groups of similar customers

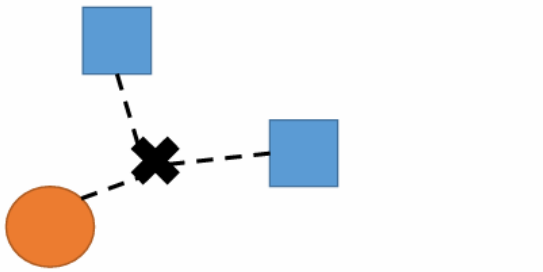


TRAIN AND TEST

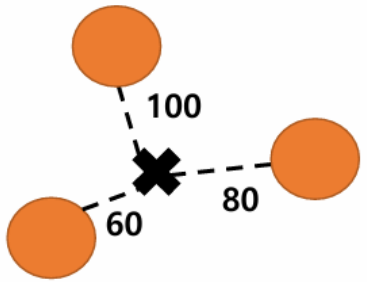
K-Nearest Neighbors (K-NN)

K-최근접 이웃

- 새로운 데이터를 예측할 때 학습 데이터에서 가장 가까운 K개의 이웃을 기준으로 예측 수행
- 주로 유클리드 거리 측정 방식 사용 $$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$
- **분류**: K개의 이웃 중 가장 많은 클래스가 속한 클래스를 새로운 데이터의 클래스로 예측
- **회귀**: K개의 이웃의 값의 평균을 계산하여 새로운 데이터의 값을 예측



최근접 이웃은 사각형이
2개이므로, X의 클래스는



X = (100+60+80)/3 = 80

TRAIN AND TEST

K-Nearest Neighbors (K-NN)

예측 값을 평가하기 위해 결정계수(coefficient of determination) 사용 = R^2

$$R^2 = 1 - \frac{(\text{타겟} - \text{예측})^2\text{의 합}}{(\text{타겟} - \text{평균})^2\text{의 합}}$$

- 타겟의 평균 정도를 예측하는 수준이라면 0에 가까워지고,
  예측이 타겟에 아주 가까워지면 1에 가까운 값을 갖게 됨

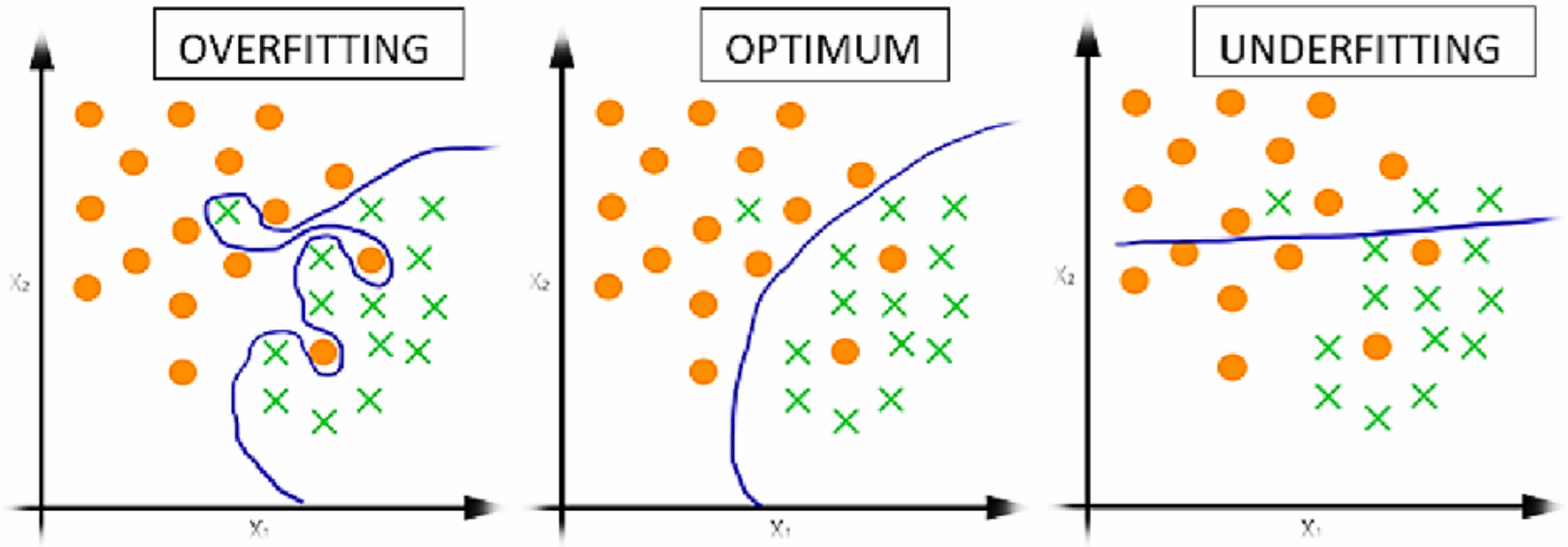
TRAIN AND TEST

K-Nearest Neighbors (K-NN)

지나치게 작은 k값은 overfitting, 지나치게 큰 값은 underfitting 유발

모델이 학습 데이터에 너무 과하게 맞춰져서,
학습 데이터의 잡음이나 세부적인 특성까지 모두 학습

너무 많은 데이터 포인트가 예측에 관여하면,
모델이 타겟 데이터의 미세한 차이를 반영하지 못함

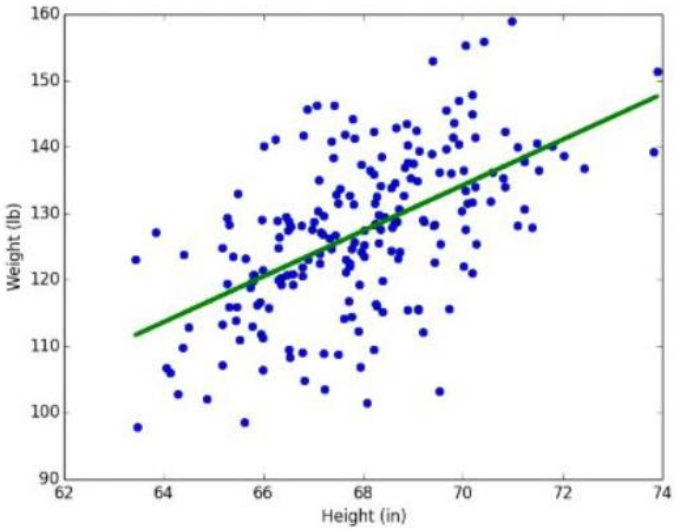


TRAIN AND TEST

Linear Regression

선형 회귀 (Linear Regression)

$$y = \beta_0 + \beta_1 x$$



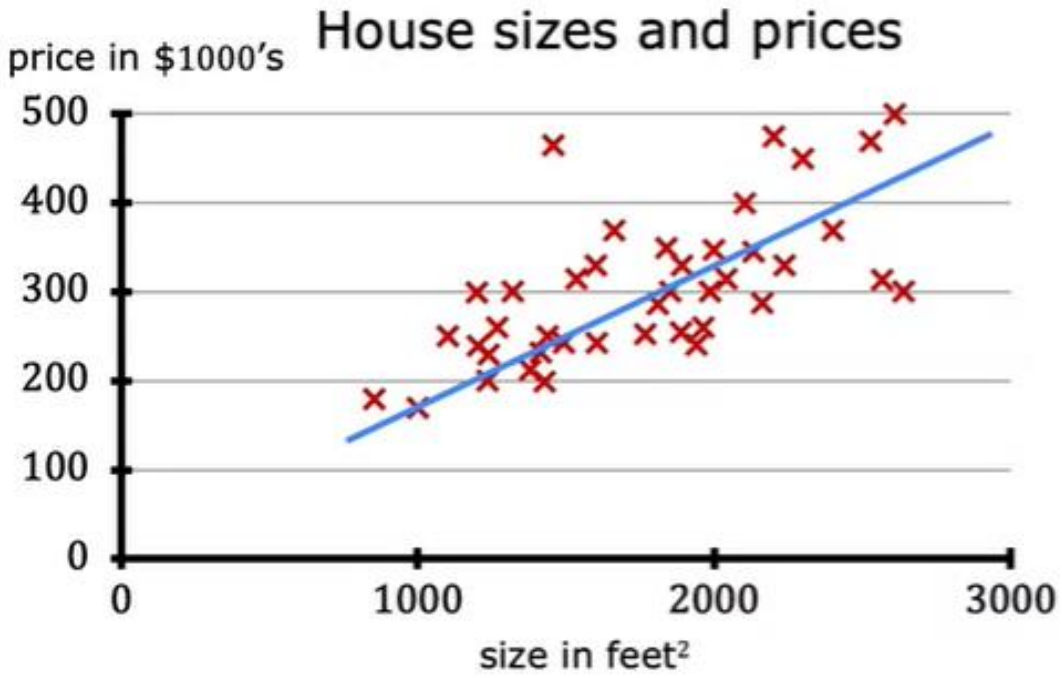
독립 변수 xi : 다른 변수와 독립적으로 작용하는 변수

종속 변수 Y : 독립변수의 결과에 의존하여 결정되는 값

독립 변수(입력)와 **종속 변수(출력)** 간의 **선형 관계**를 모델링하는 가장 기본적인 회귀 기법

TRAIN AND TEST

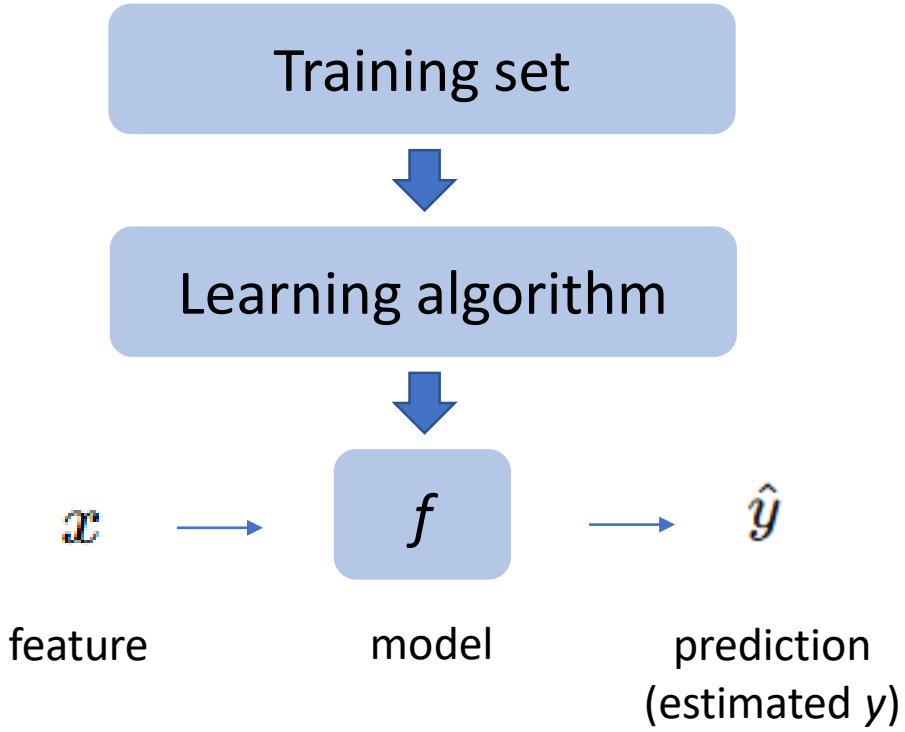
Linear Regression



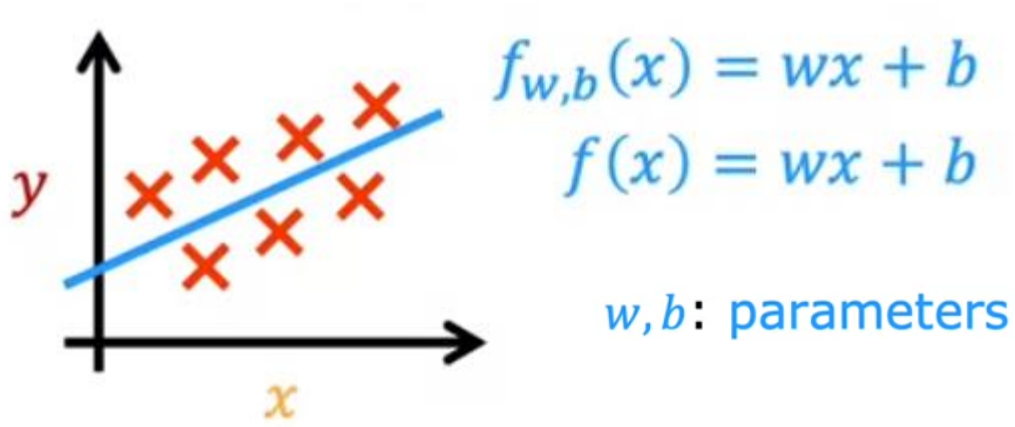
size in feet²	price in $1000's
2104	400
1416	232
1534	315
852	178
...	...
3210	870

TRAIN AND TEST

Linear Regression



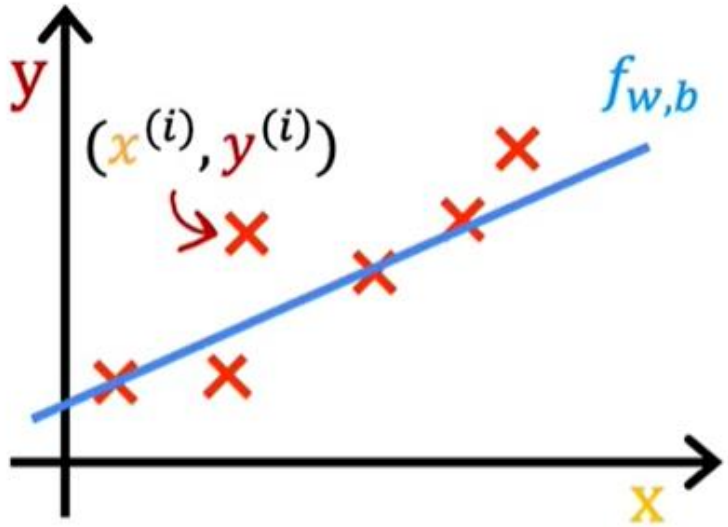
How to represent f?



→ Linear regression with one variable

TRAIN AND TEST

Linear Regression



$$\hat{y}^{(i)} = f_{w,b}(x^{(i)})$$

$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

Cost function: Squared error cost function

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

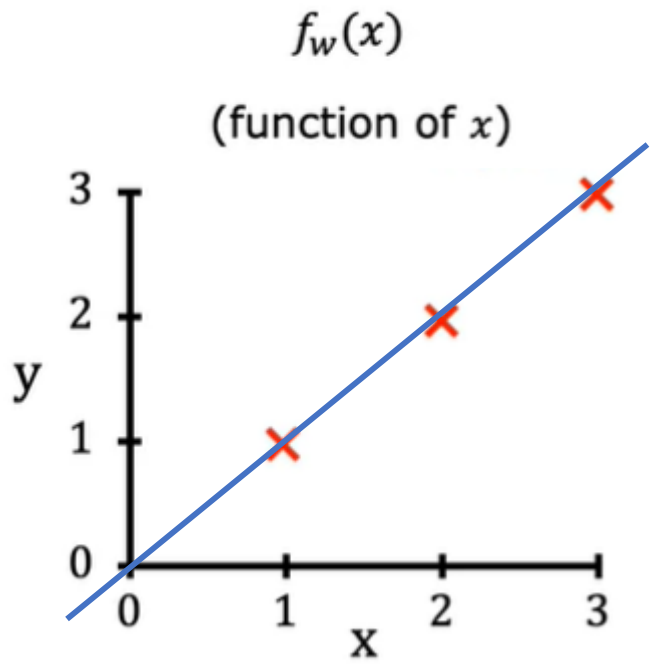
→ 예측 값 $\hat{y}^{(i)}$가 실제 목표 $y^{(i)}$에 가깝도록 하는 w와 b 찾기

→ 비용 함수를 작게 만드는 w와 b 찾기

Goal: minimize J(w, b)

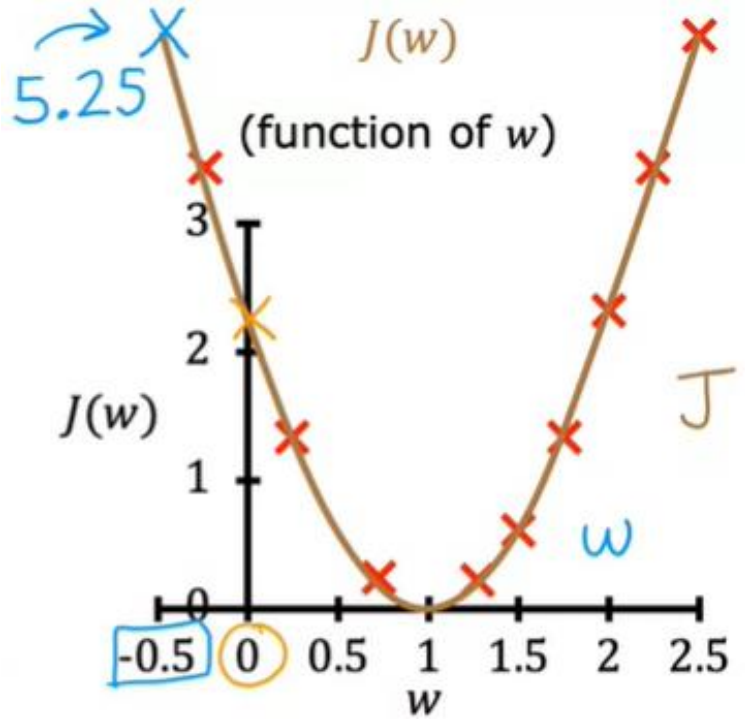
TRAIN AND TEST

Linear Regression



$w = 1$

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



→ choose w to minimize J(w)

Linear Regression

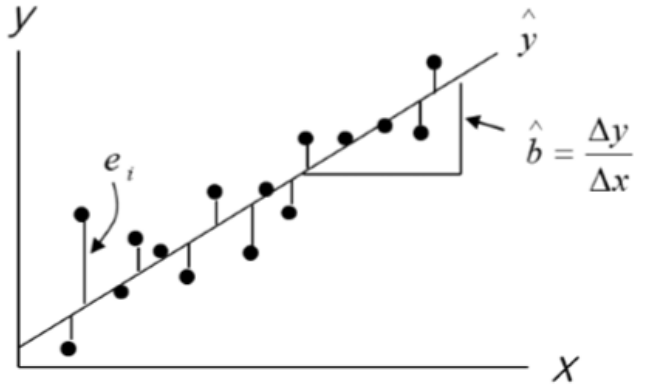
선형 회귀 모델 훈련

- **모델 훈련** : 모델이 훈련세트에 가장 잘 맞도록 모델 파라미터를 설정하는 것

  → 모델이 훈련 데이터에 얼마나 잘 들어맞는지 측정 필요

- **성능 측정 지표** : 평균 제곱근 오차 (RMSE)

  → RMSE를 최소화하는 θ를 찾아야 함



MSE (Mean Squared Error):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

RMSE (Root Mean Squared Error):

$$RMSE = \sqrt{MSE}$$

TRAIN AND TEST

Gradient Descent

Step 1. Start with some *w, b* (set *w*=0, *b*=0)

Step 2. Keep changing *w, b* to reduce *J*(*w,b*)

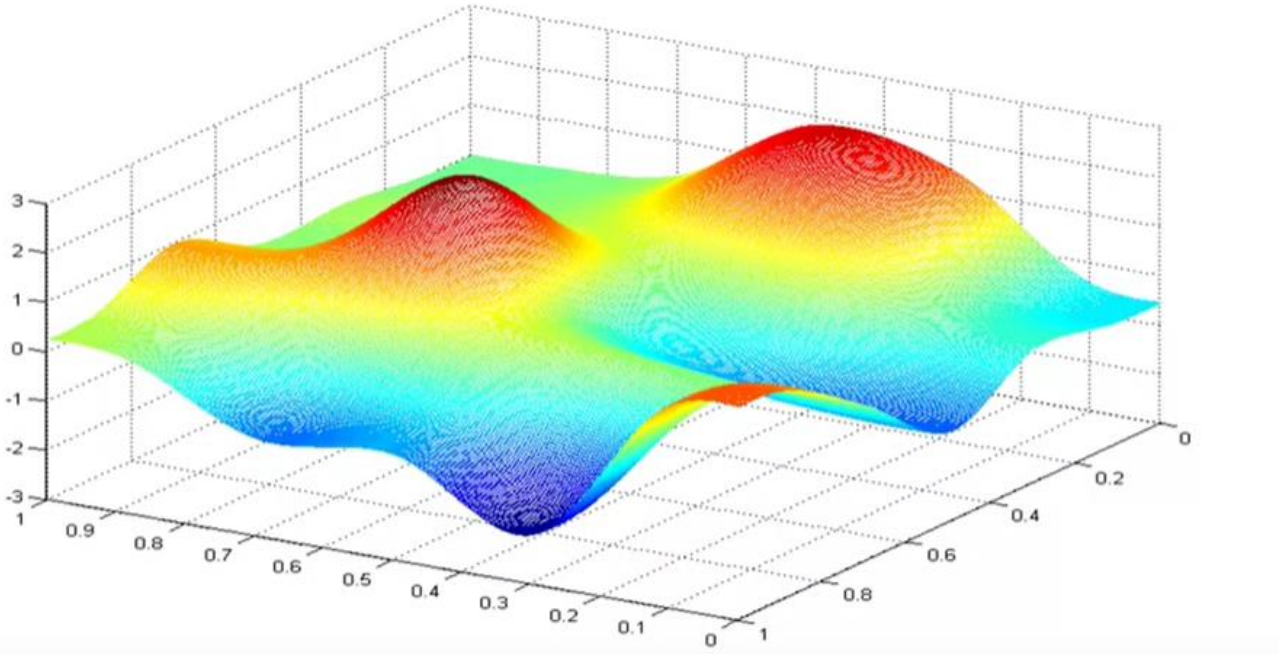
Step 3. Until we settle at or near a minimum

derivative

learning rate

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

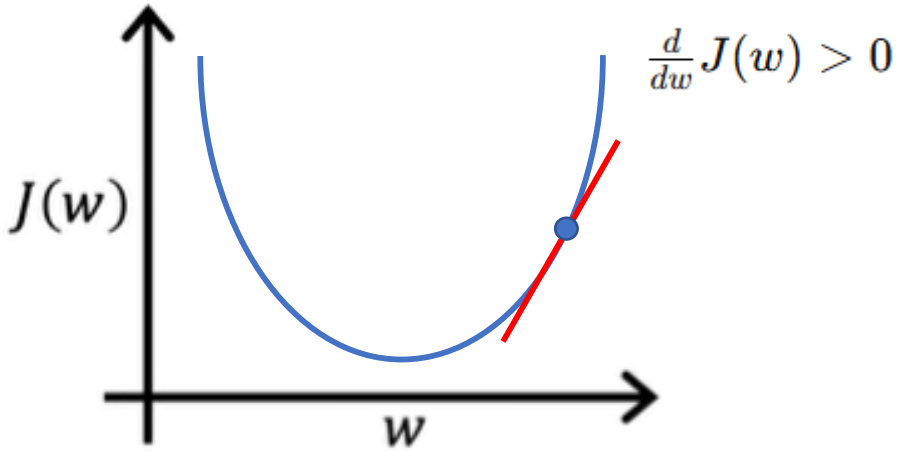
$$b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$



TRAIN AND TEST

Gradient Descent

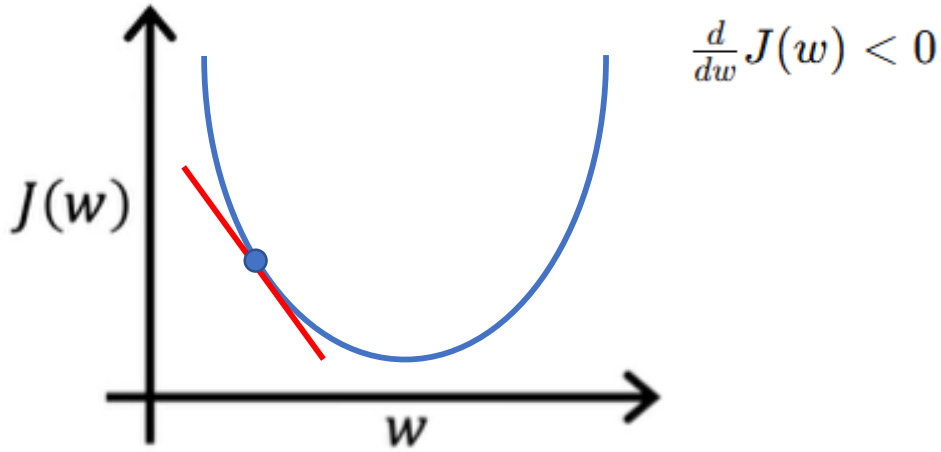
기울기가 양수일 때



$$w = w - \alpha \cdot (\text{positive number})$$

→ w 감소

기울기가 음수일 때



$$w = w - \alpha \cdot (\text{negative number})$$

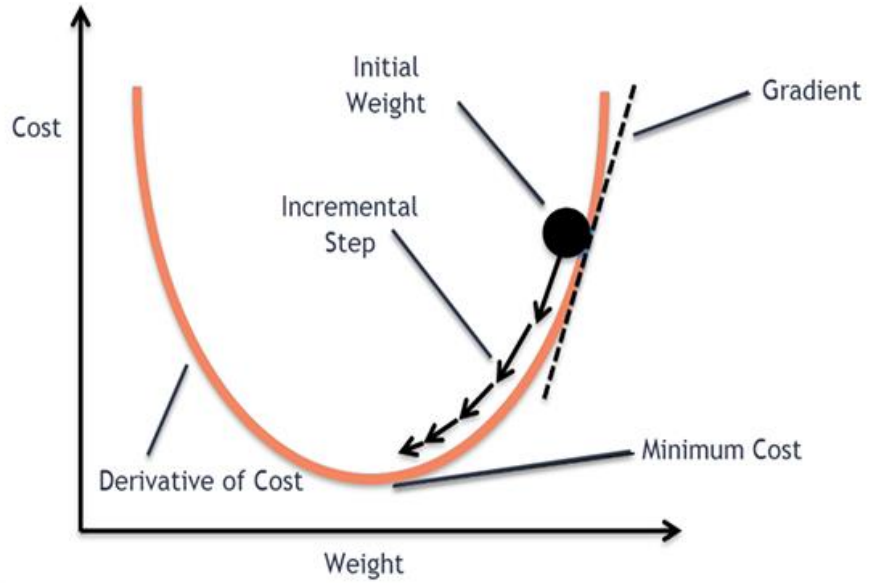
→ w 증가

TRAIN AND TEST

Gradient Descent

경사 하강법 (Gradient Descent)

- 여러 종류의 문제에서 최적의 해법을 찾을 수 있는 방법
- **비용 함수(cost function)를 최소화**하기 위해 반복해서 파라미터를 조정하는 것
- 파라미터 벡터 θ에 대해 현재 gradient를 계산하고, 감소하는 방향으로 진행 $\rightarrow$ 0에 도달한 경우 = 최솟값에 도달



TRAIN AND TEST

Gradient Descent

- 경사 하강법의 가장 중요한 파라미터

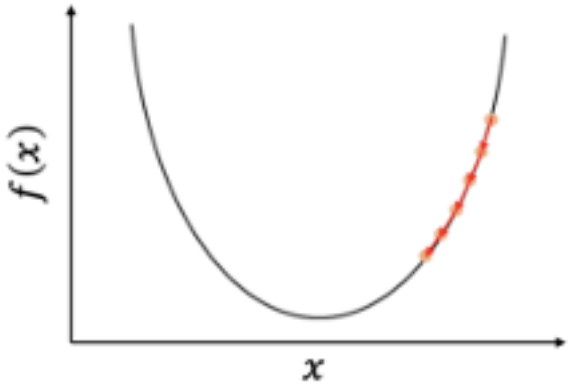
  = step의 크기 = 학습률 (learning rate) **'하이퍼파라미터'**

- **학습률이 너무 작으면?**

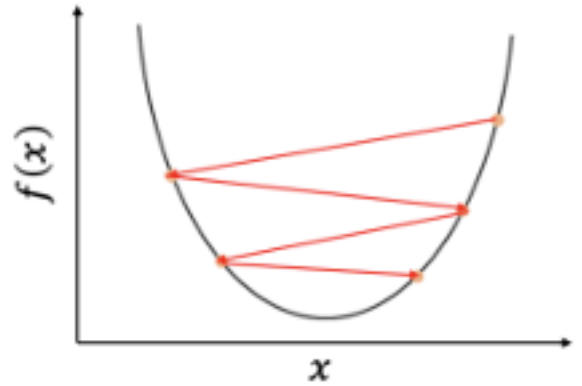
  알고리즘이 수렴하기 위해 반복을 많이 진행해야 하므로 시간이 오래 걸림

- **학습률이 너무 크면?**

  알고리즘을 더 큰 값으로 발산할 수 있으며, 최솟값에 도달하지 못함



α가 너무 작은 경우 (수렴이 늦음)

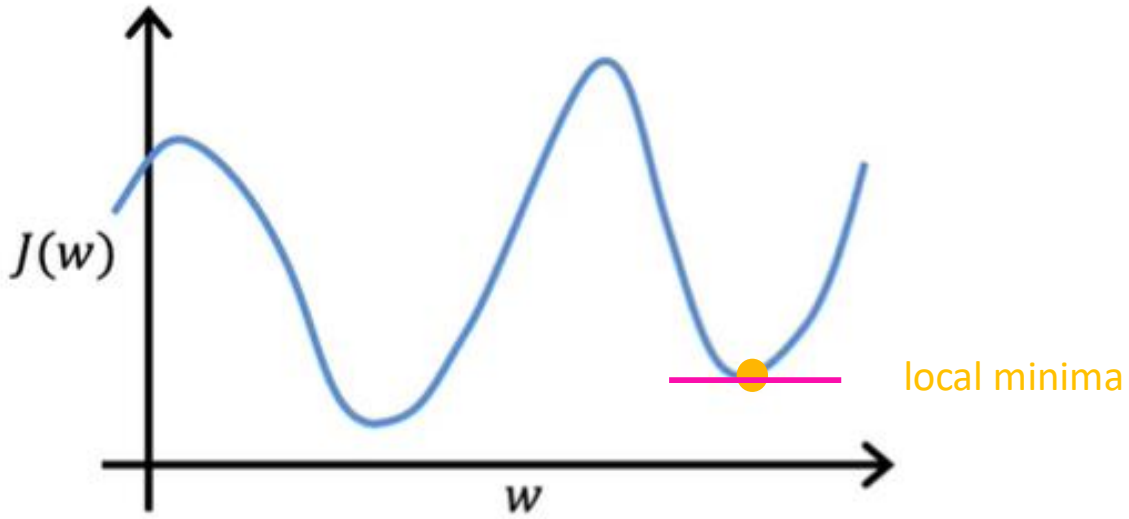


α가 너무 큰 경우 (진동)

TRAIN AND TEST

Gradient Descent

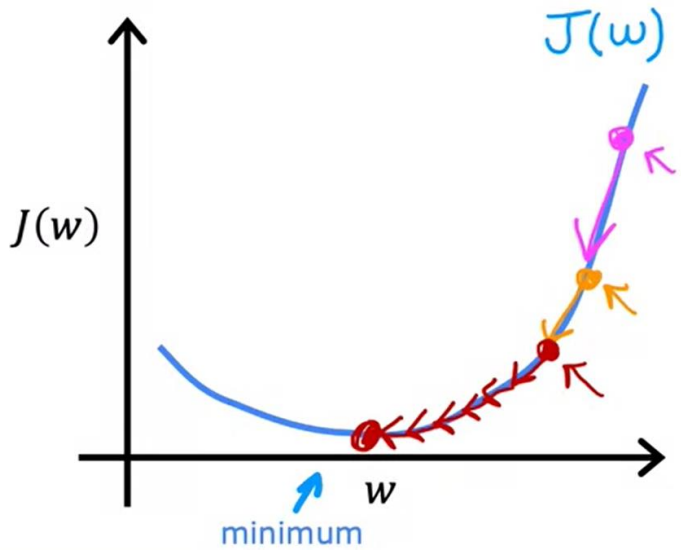
선형 회귀의 MSE 함수는 볼록 함수 형태 → local minima X



$$w = w - \alpha \frac{d}{dw} J(w)$$

$$w = w - \alpha \cdot 0$$

$$w = w$$



local minimum에 가까울수록,
도함수는 더 작아지고
업데이트되는 정도도 더 작아짐

TRAIN AND TEST

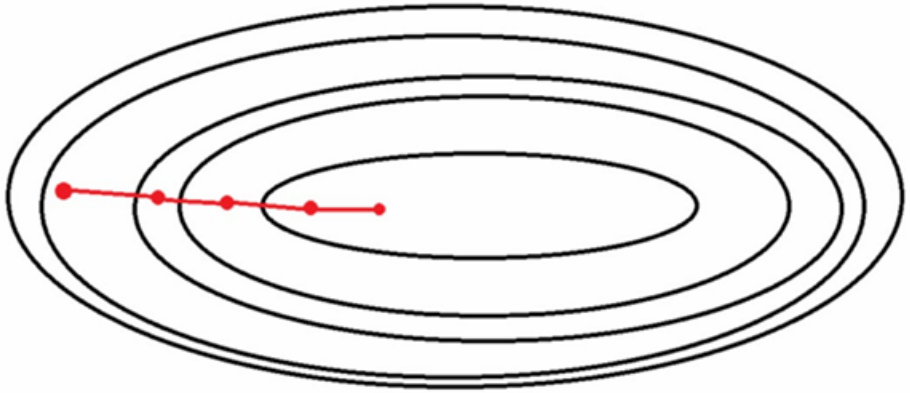
Batch Gradient Descent

배치 경사 하강법 (BGD)

	x size in feet²	y price in $1000's
(1)	2104	400
(2)	1416	232
(3)	1534	315
(4)	852	178
...	...	...
(47)	3210	870

→ m = 47

$$\sum_{i=1}^{m} (f_{w,b}(x^{(i)}) - y^{(i)})^2$$



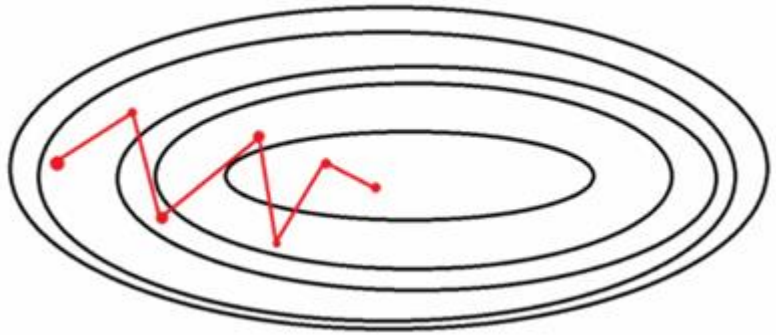
- 전체 데이터 셋에 대해 오차를 구한 뒤, 기울기를 한번만 계산하여 모델의 파라미터 업데이트
- 전체 데이터를 모두 한 번에 처리하기 때문에 많은 메모리 필요

TRAIN AND TEST

Stochastic Gradient Descent

확률적 경사 하강법 (SGD)

- 전체 데이터 중 하나의 데이터를 무작위로 선택하여 그 샘플에 대한 gradient를 계산
- 매 반복에서 매우 적은 데이터만 처리하기 때문에 알고리즘이 매우 빠름
- 매 반복에서 하나의 샘플만 메모리에 있으면 되므로, 매우 큰 훈련 세트도 훈련 가능
- cost function이 최솟값에 다다를 때까지 부드럽게 감소하지 않고 위아래로 요동치면서 평균적으로 감소 → Global minimum에 다다르지 못할 가능성 높음



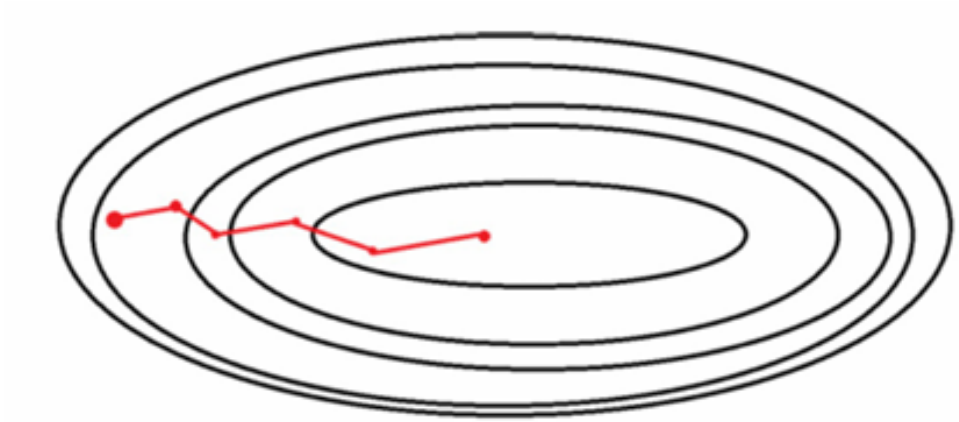
TRAIN AND TEST

Mini-batch Gradient Descent

미니배치 경사 하강법

- BGD와 SGD의 절충안
- 전체 데이터를 batch size개씩 나누어 배치로 학습 (batch는 하이퍼파리미터)
- mini batch라 부르는 임의의 작은 샘플 세트에 대해 gradient 계산
- 행렬 연산에 최적화된 하드웨어 (GPU)를 사용해서 얻는 성능 향상
- Mini batch를 어느정도 크게 하면 SGD보다 덜 불규칙하게 움직임

  → 즉, SGD보다 최솟값에 더 가까이 도달하게 될 것



TRAIN AND TEST

Quiz!

Exercise 12: *KNN algorithm in Finance*

A group of analysts at the bank *Stratton Oakmont* uses chart analysis to recommend stocks to *buy*, *hold* or *sell*. The following features were recorded for all recommendations that were later proven to be correct:

Feature	Description
Stock	ID of stock (do not use this feature for distance computations)
ADMI	Average directional movement index (technical indicator)
RSI	Relative strength index (technical indicator)
Recommendation	Given recommendation (buy, hold or sell) - this is the target feature

You have been asked to develop a machine learning system that provides recommendations based on the two technical indicators ADMI and RSI. The following preprocessed training and validation data is available:

TRAIN AND TEST

Training objects

Stock	ADMI	RSI	Recommendation
1	1	1.0	Sell
2	2	0.5	Sell
3	3	0.5	Sell
4	4	2.5	Sell
5	1	2.0	Hold
6	2	3.5	Hold
7	2	1.5	Hold
8	3	2.0	Hold
9	1	3.0	Buy
10	2	4.5	Buy
11	2	3.0	Buy
12	3	4.0	Buy

Validation objects

Stock	ADMI	RSI	Recommendation
13	1	5	Buy
14	0	2	Hold
15	4	2	Hold
16	3	1	Sell

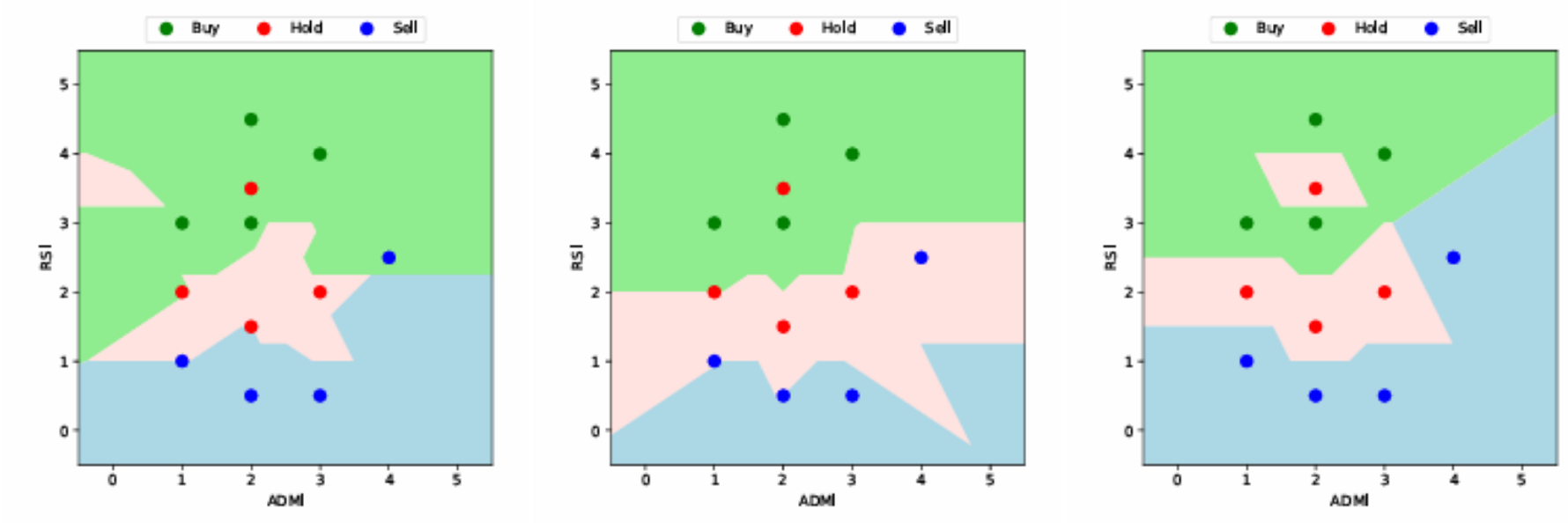
(a) Complete the following Euclidean distance matrix. You do not need to standardise the data.

$d^E(i_1, i_2)$	$i_2 = 13$	$i_2 = 14$	$i_2 = 15$	$i_2 = 16$
$i_1 = 1$	4.00	1.41	3.16	2.00
$i_1 = 2$	4.61	2.50	2.50	1.12
$i_1 = 3$	4.92	3.35		0.50
$i_1 = 4$	3.91		0.50	1.80
$i_1 = 5$	3.00	1.00	3.00	2.24
$i_1 = 6$		2.50	2.50	2.69
$i_1 = 7$	3.64	2.06	2.06	1.12
$i_1 = 8$	3.61	3.00	1.00	1.00
$i_1 = 9$	2.00	1.41	3.16	2.83
$i_1 = 10$	1.12	3.20	3.20	3.64
$i_1 = 11$	2.24	2.24	2.24	2.24
$i_1 = 12$	2.24	3.61	2.24	3.00

(b) What labels does the KNN algorithm with $k = 1$ assign to the validation objects?

TRAIN AND TEST

(d) The decision boundaries below were obtained with $k \in \{1, 2, 3\}$. Which decision boundary corresponds to which value of k?



TRAIN AND TEST

TRAIN AND TEST