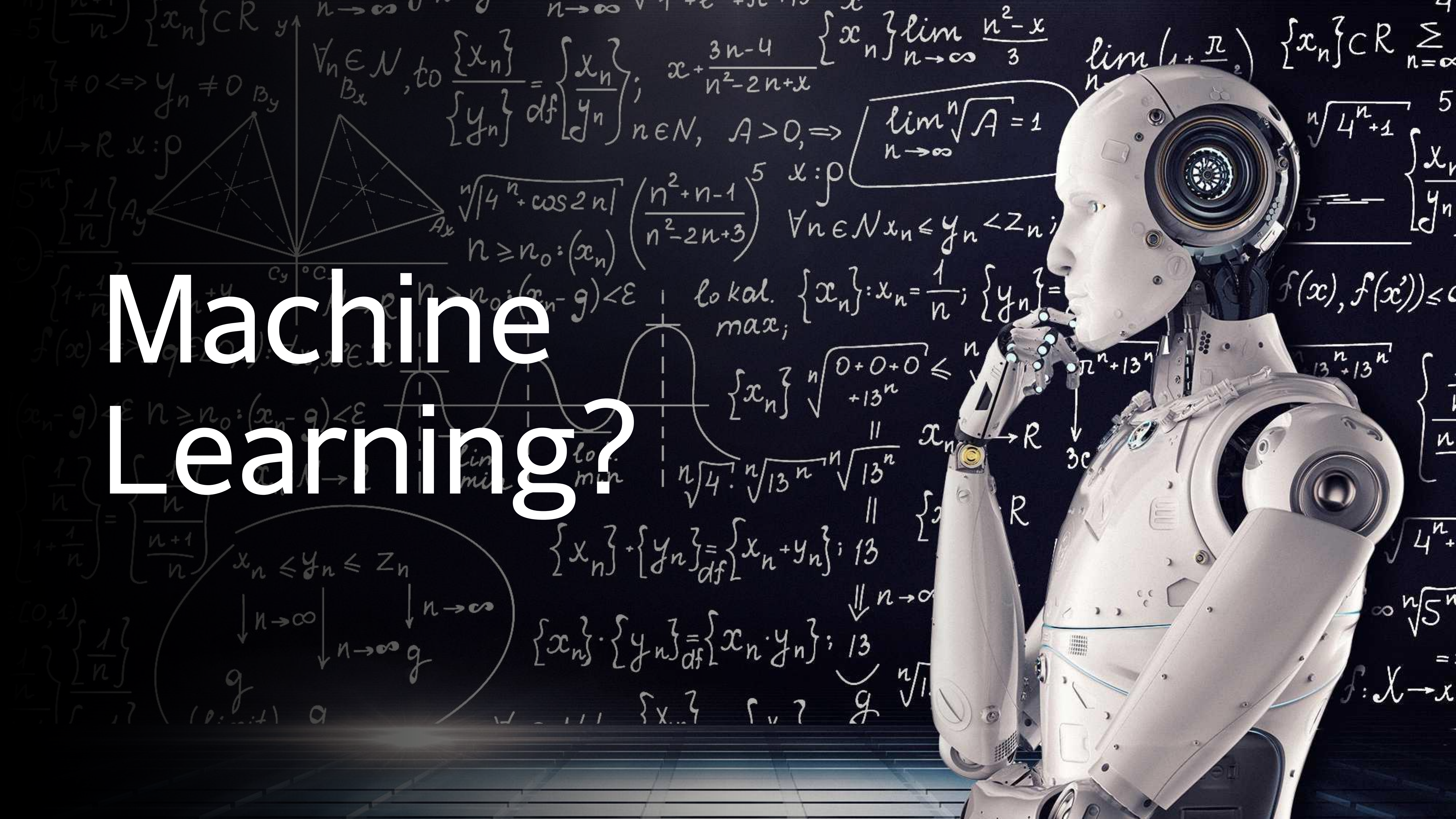
Introduction to Machine Learning

한정연

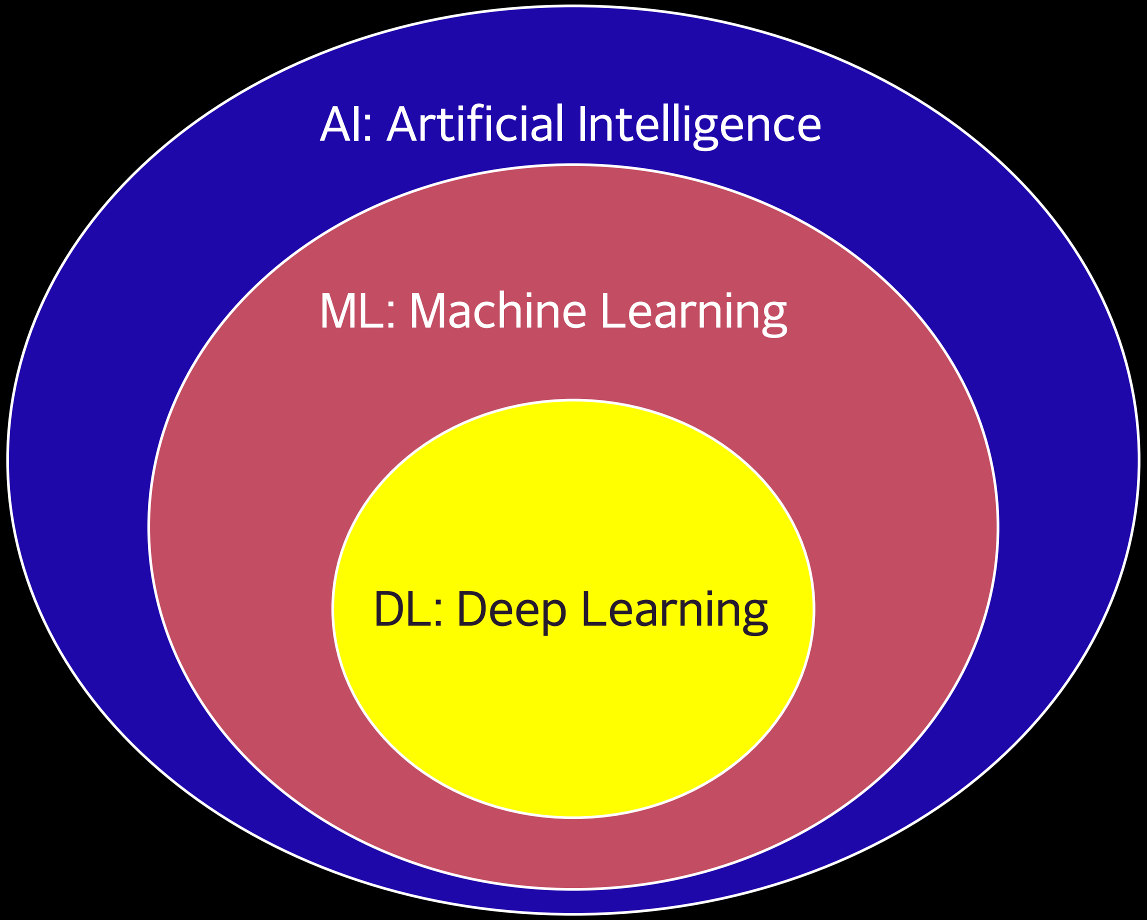
Coursera lecture by Prof. Andrew Ng

Table of contents

Machine Learning?



0. Overview



Machine Learning

Field of study that gives computers the ability to learn **without being explicitly programmed**

Arthur Samuel (1901 - 1990)

Traditional programming vs Machine Learning

미션: 오늘 저녁엔 컴퓨터를 이용해 맛있는 김치찌개를 만들어보자!

1. 레시피를 준다 ➔ Traditional programming

- 돼지고기는 핏물을 빼 주세요
- 엄마 표 김치를 준비해주세요
- 들기름을 이용해 김치와 함께 돼지고기를 볶아주세요
- 다진 마늘. 설탕 한 스푼 씩 넣고 종이컵으로 물 8컵을 넣어 센 불에 끓여주세요
- 돼지고기를 넣고 끓이다 끓기 시작하면 양파 등을 넣고 더 끓여주세요

2. 예시를 주고 추론하여 만든다 ➔ Machine Learning

- A 식당 김치찌개
- B 식당 김치찌개
- C 식당 김치찌개
- D 식당 김치찌개
- E 식당 김치찌개

카카오 평점 별 5개 김치찌개 식당들

백종원씨나 레시피를 아는 경우에는 좋음

레시피를 모르는 경우에는 김치찌개를 못 먹음 ㅠㅠ

우리가 답을 알아야 할 필요없이 컴퓨터가 추론하게끔 함

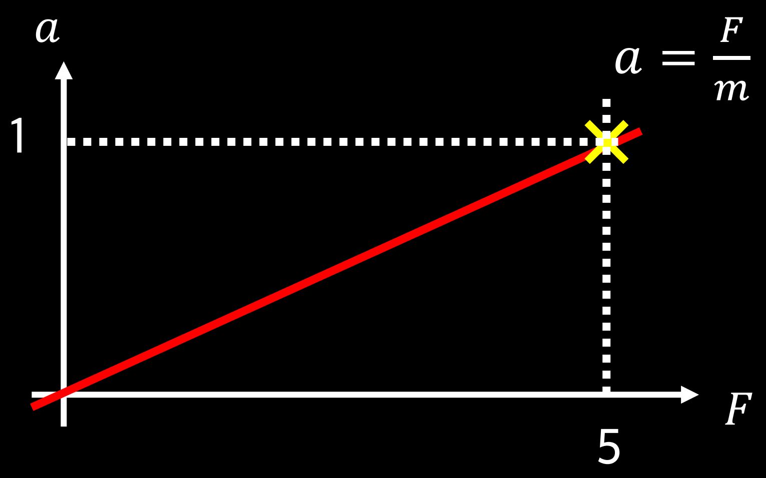
예시가 너무 적으면 추론이 불가능!

Traditional programming vs Machine Learning

미션: 주어진 힘을 이용해 가속도를 추론하라! 질량은 5kg이다

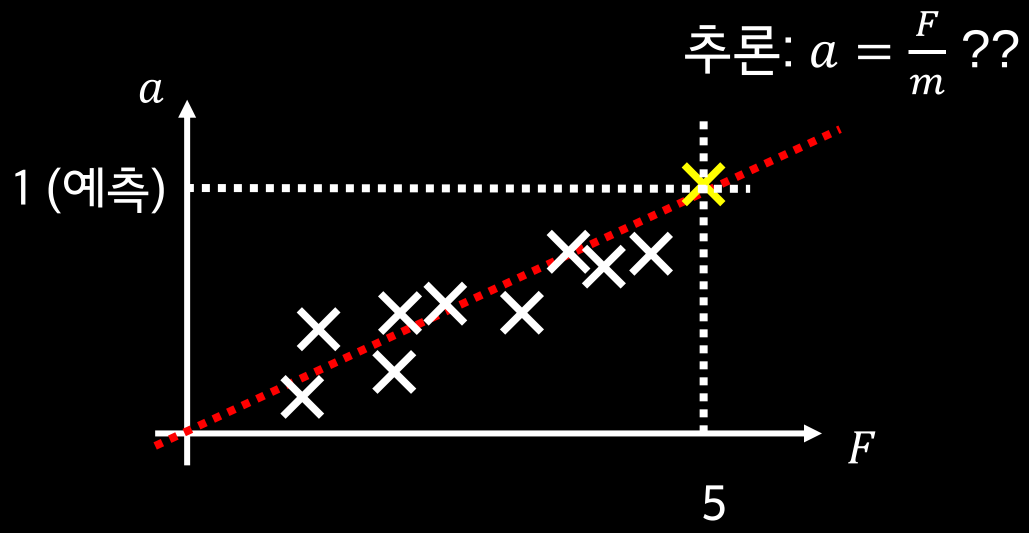
1. Traditional programming

Newton’s second law



➡ 모델 $a = \frac{F}{m}$을 만들고 F에 5를 대입

2. Machine Learning



데이터

a	**F**
0.18	1
0.31	1.5
0.41	2
0.48	2.5
0.62	3
0.71	3.5
…	

➡ 데이터로부터 추론된 모델 (statistical modelling) $a = \frac{F}{m}$을 통한 예측 (prediction)

Machine Learning의 종류

1. Supervised Learning

주어진 데이터의 feature X에 대해 Label (정답) Y가 존재할 때

Feature X Label Y

Ex: **주어진 힘 F**에 대해서 **가속도 a**가 주어짐 (Regression)

Feature X Label Y

Ex: 김치찌개의 **재료들**과 만들어진 김치찌개의 **맛의 평가 (맛있음/없음)**가 주어짐 (Classification)

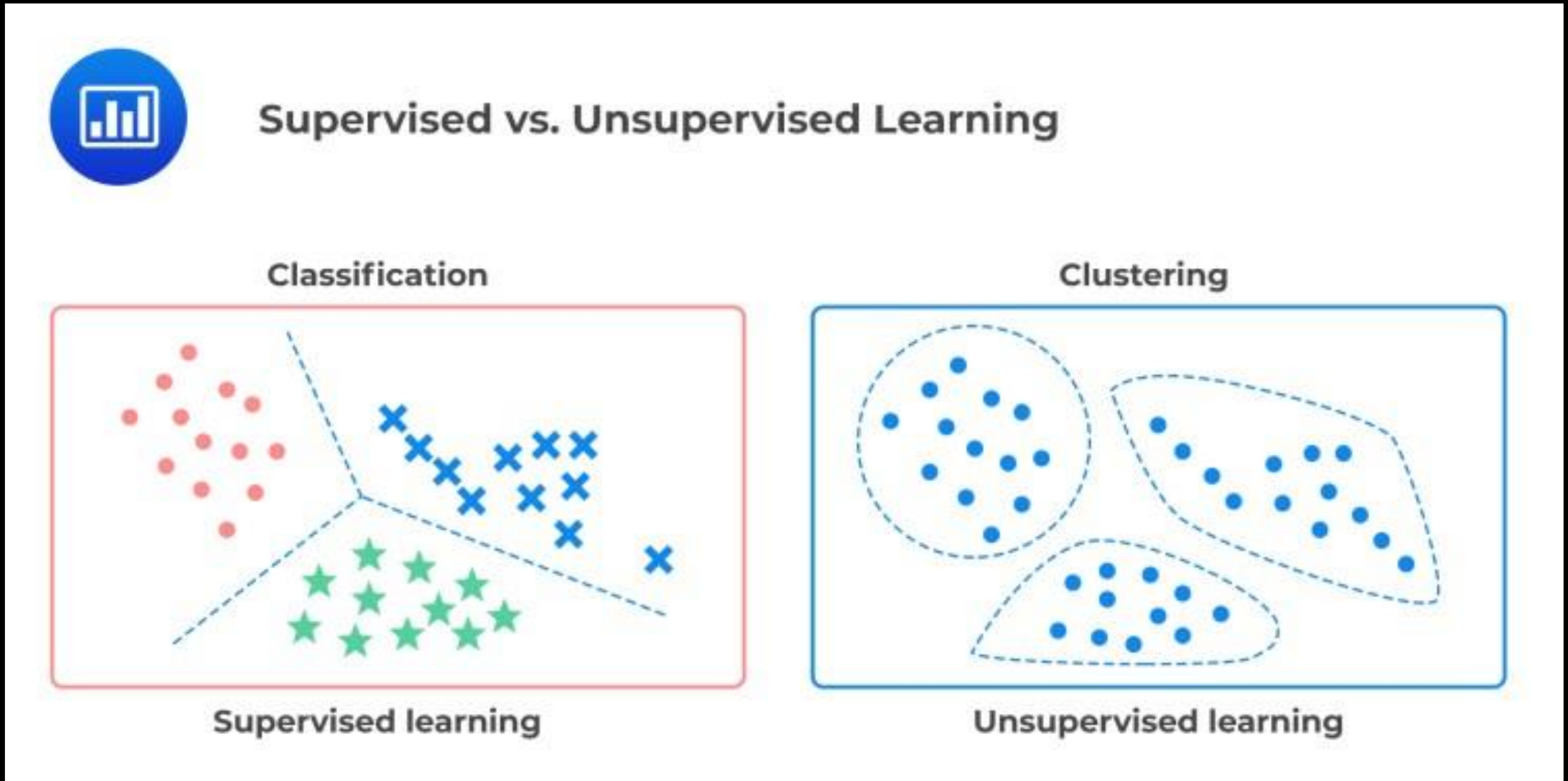
2. Unsupervised Learning

주어진 데이터의 feature X에 대해 Label (정답) Y가 존재하지 않을 때

Feature X

Ex: 전국의 김치찌개 **재료들**이 주어짐 ➔ 맵기에 따라 분류/스타일에 따라 분류/ … (Clustering)

Supervised vs Unsupervised Learning



Label이 있음

Label이 없음

Reference: https://www.linkedin.com/pulse/supervised-vs-unsupervised-learning-whats-difference-smriti-saini/

1. Supervised Learning

Supervised Learning

Features　　　　　Label

X → Y

(Input: vector, scalar, …)　　　　　(Output: vector, scalar, …)

"주어진 정답" 으로부터 배운다.

How the learning works?

1. Input과 label로 이루어진 "Training Set"을 준비

2. Algorithm (선형 회귀, 로지스틱 회귀 등등)이 Training set을 기반으로 학습
➔ $F(x)$를 생성!

3. Algorithm을 통해 생성한 $F(x)$에 "Test Set"의 feature들을 대입하여
"예측값" $\hat{Y}$를 뽑아냄

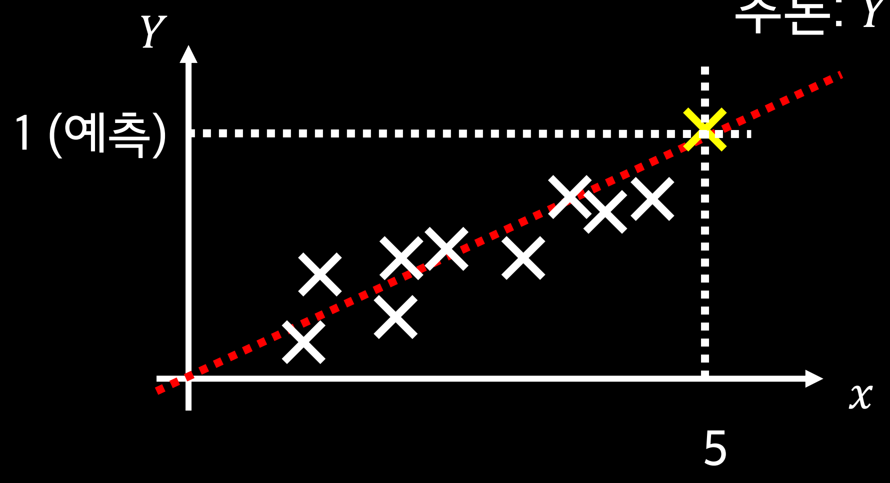
Test set Features　　　　Prediction

$$X \xrightarrow{F(x)} \hat{Y}$$

4. Test Set의 정답 (label)과 예측 (prediction)이 얼마나 일치하는지 점수화
➔ 모델 성능

Linear Regression (선형 회귀)

추론: $\hat{Y} = wx$



데이터

x	$Y\ (\neq \hat{Y})$
0.18	1
0.31	1.5
0.41	2
0.48	2.5
0.62	3
0.71	3.5
…	

Goal: 정답 Y를 예측값 $\hat{Y}$를 통해 예측하기! 이 때, 예측 값은 다음과 같은 식으로 주어진다.

$$\hat{Y}^{(i)} = F\left(x^{(i)}\right) = wx^{(i)} + b$$

i번째 예측 값

Weight
(feature의 중요도)

Feature
(i번째 input)

Bias (Scalar)

Linear Regression (선형 회귀)

집값 데이터

크기 (m^3)	침실 갯수	연식 (년)	집값 (억)
1416	3	20	15
1026	2	16	12
2312	1	24	10
321	2	4	2.5
581	1	6	3
2890	3	7	21
…			

Feature: 크기, 침실 개수, 연식

$\longrightarrow$ $[x_1, x_2, x_3] = \vec{x}$

Label: 집값 $\longrightarrow$ Y

일반화된 선형회귀 방정식

$$\hat{Y}^{(i)} = F\left(x^{(i)}\right) = \vec{w} \cdot \vec{x}^{(i)} + b$$

Weight vector
(각 feature들의 중요도)

Feature vector
(i번째 Input vector)

Bias (Scalar)

Note: 벡터들은 방향벡터 ($\hat{x}_1, \hat{x}_2, \hat{x}_3$
로 표현되며 크기가 1인 벡터임)를
이용하여 표현할 수 있다.

$$\vec{x} = [x_1, x_2, x_3] = x_1\hat{x}_1 + x_2\hat{x}_2 + x_3\hat{x}_3$$

$$\vec{w} = [w_1, w_2, w_3] = w_1\hat{x}_1 + w_2\hat{x}_2 + w_3\hat{x}_3$$

Classification: 분류 문제

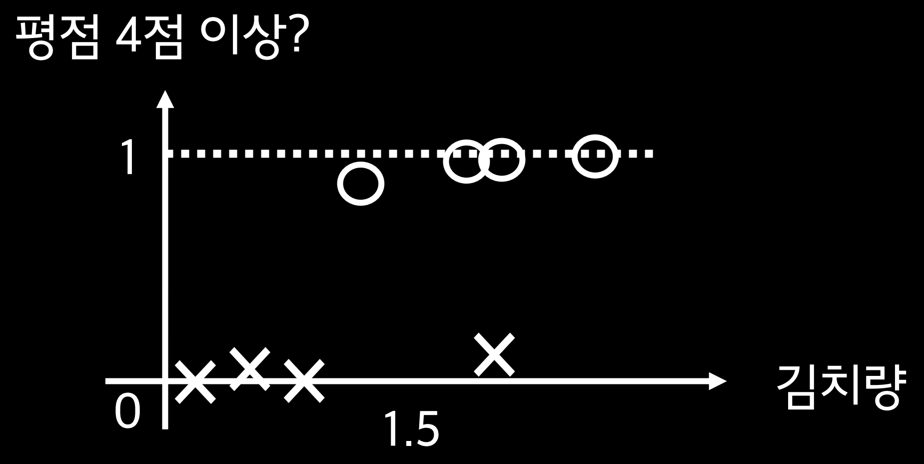
김치찌개 데이터

김치량 (kg)	야채 종류	국물량 (L)	평점 4점 이상?
1.416	5	1	Y
1.026	4	1.2	N
2.312	3	0.8	Y
2.21	4	1.1	Y
1.81	3	0.9	N
2.89	5	0.7	Y
…			

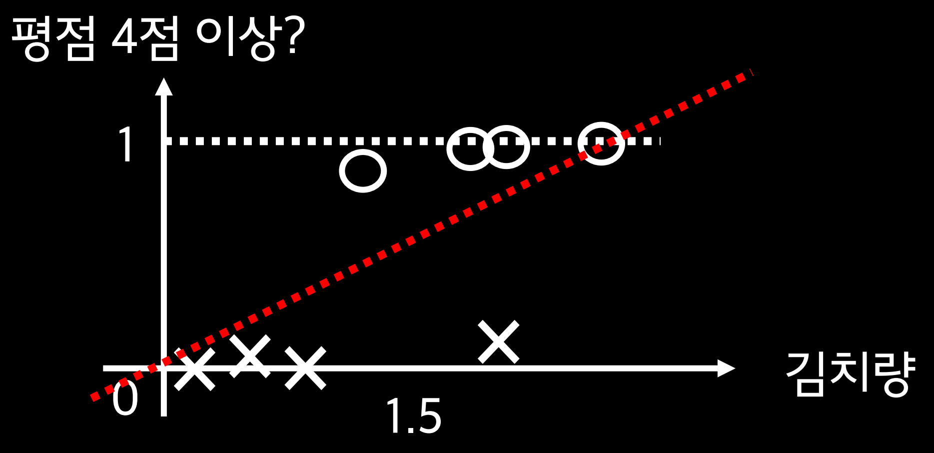
Label: 평점 4점 이상?

→ Y = 0 (No) / 1 (Yes)

(Classification)



선형회귀?



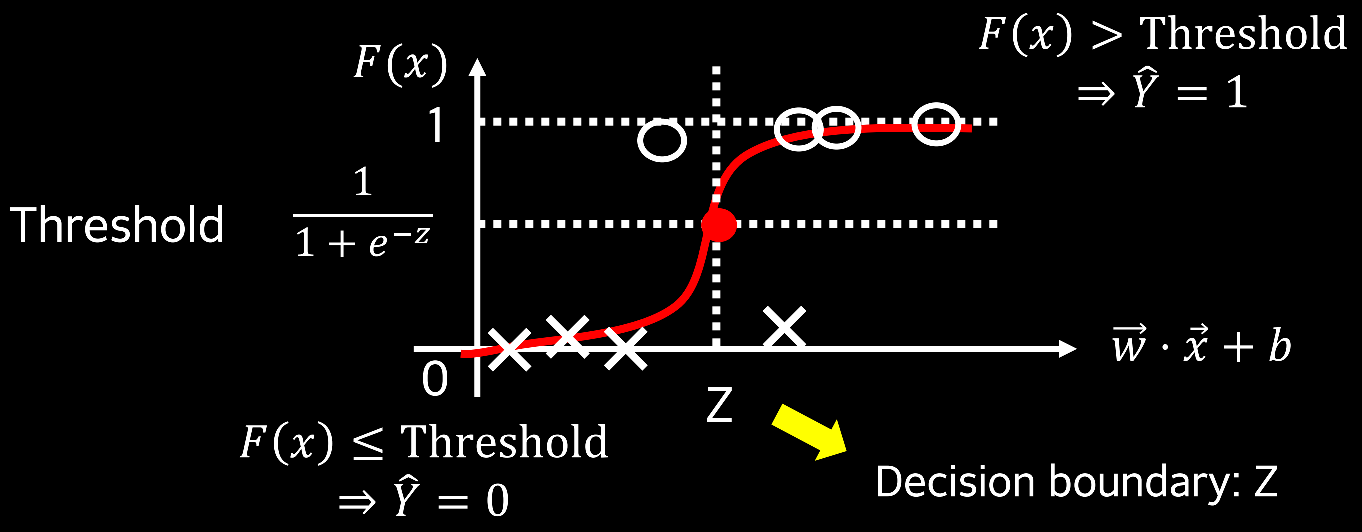
선형회귀는 잘 작동하지 않음!

Logistic Regression (로지스틱 회귀)

일반화된 로지스틱 회귀 방정식 (Sigmoid 함수)

$$\hat{Y}^{(i)} = F\left(x^{(i)}\right) = \frac{1}{1 + e^{-(\vec{w}\cdot\vec{x}^{(i)}+b)}}$$

함수 형태



2. Gradient Descent

Cost Function

Y	$\hat{Y}$
실제 값(Label)과 예측 값(Prediction)이 얼마나 차이가 나는가?	

Cost function $J(\vec{w}, b) \coloneqq \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \hat{Y}^{(i)}\right)^2$

(왜 제곱? 1. 미분가능, 2. 모든 지점에서 음수가 되지 않음)

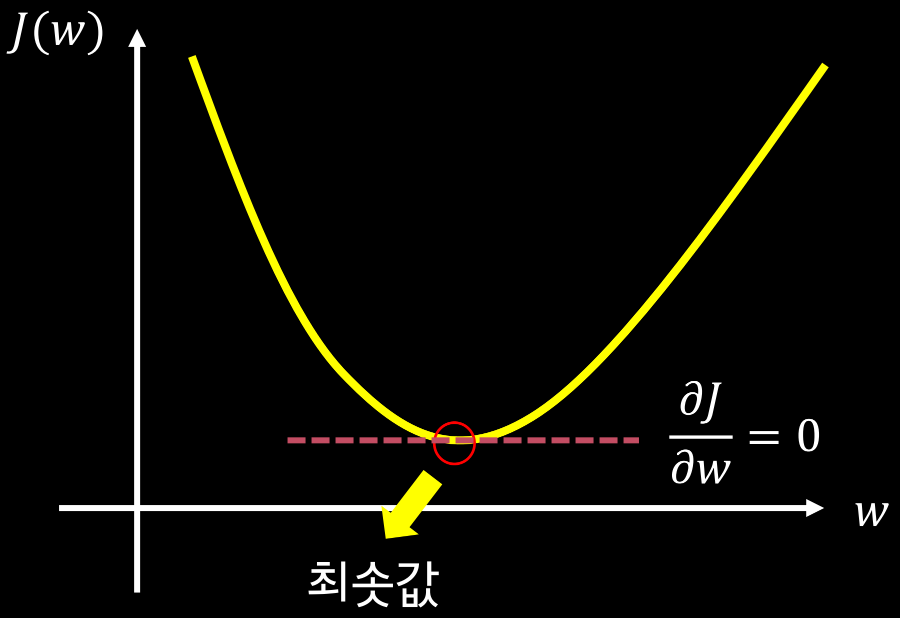
Goal: Cost function을 최소화 ➔ 예측을 최대한 실제 값과 비슷하게 하고싶음!

Idea: 변수가 Weight $\vec{w}$, Bias b이므로, 각각의 편미분이 0이 되는 지점이 최소!

왜 편미분이 0이 되는 지점?

Ex. Linear Regression with a single feature ($b = 0$)

$$J(w) := \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \hat{Y}^{(i)}\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - wx^{(i)}\right)^2$$

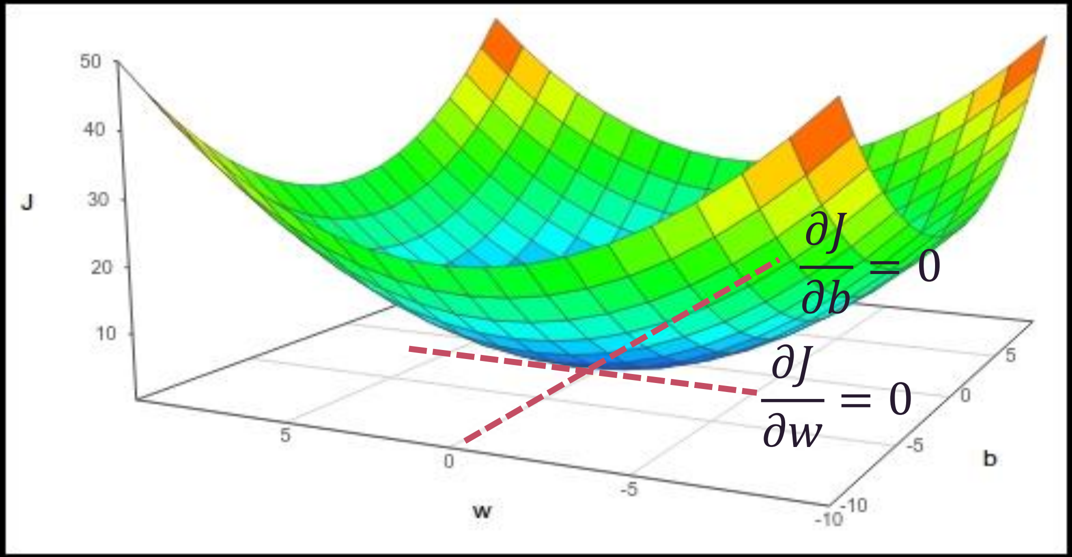


이와 같이 볼록한 함수 (Convex Function)인 경우
편미분이 0인 지점 = 최솟값이 됨

왜 편미분이 0이 되는 지점?

Ex. Linear Regression with a single feature ($b \neq 0$)

$$J(w,b) := \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \hat{Y}^{(i)}\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - wx^{(i)} - b\right)^2$$



모든 변수의 미분이 0이 되는 값을 찾는 것이 목표!

Cost Function은 실제로 굉장히 찾기 힘들다

$$J(\vec{w}, b) \coloneqq \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \hat{Y}^{(i)}\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \vec{w}\cdot\vec{x}^{(i)} - b\right)^2$$

Feature가 d차원인 경우 → $d+1$ 차원에서의 plot이 필요

$$J(\vec{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - w_1 x_1^{(i)} - w_2 x_2^{(i)} - \cdots - w_d x_d^{(i)} - b\right)^2$$

1) 모든 점의 미분을 구하는 것은 힘들다. ➔ 계산을 효율적으로 할 수 없을까?

2) 만약에 우리가 다루는 것이 Linear Regression 모델이 아니 라서 convexity (볼록성)이 보장되지 않은 경우에는?

Gradient Descent

Cost Function의 최솟값을 효율적으로 찾을 수 있는 알고리즘

1) 임의의 초기값을 변수$(\vec{w}, b)$ 에 대입

2) $\vec{w}, b$를 계속 업데이트!

3) 최솟값에 다다를 때 까지 알고리즘을 반복!

업데이트된 변수

$$
\begin{aligned}
w_1' &= w_1 - \alpha \frac{\partial}{\partial w_1} J(\vec{w}, b) \\
w_2' &= w_2 - \alpha \frac{\partial}{\partial w_2} J(\vec{w}, b) \\
&\vdots \\
b' &= b - \alpha \frac{\partial}{\partial b} J(\vec{w}, b)
\end{aligned}
$$

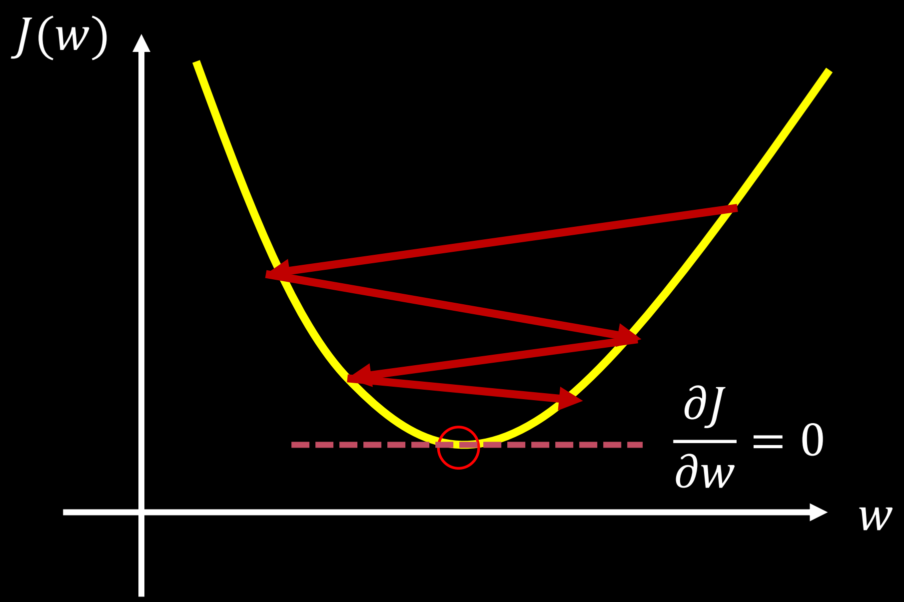
$\boxed{\alpha = \text{Learning rate}}$

Gradient Descent

Learning rate α의 적절한 선택이 필요!

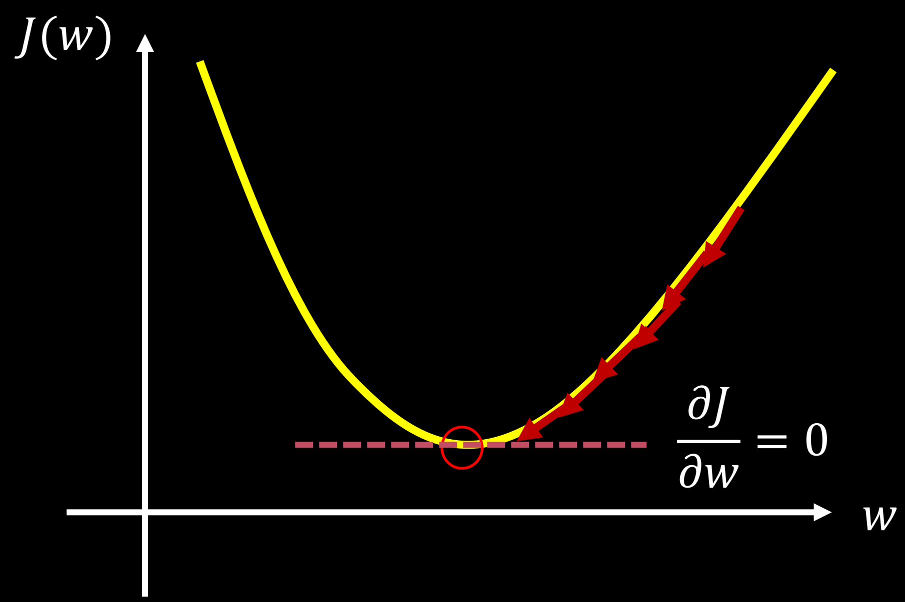
Ex. Single feature, $b = 0$

α가 큰 경우



학습의 속도가 빠르나 너무 크면, 최저점에 수렴하지 못할 수 있음

α가 작은 경우



최저점에 무조건 수렴하지만, 학습 속도가 느림

Digression: 편미분 (Partial Differential)

Gradient Vector $\vec{\nabla}$: 미분 벡터

$$\boxed{\begin{aligned} w_1' &= w_1 - \alpha \frac{\partial}{\partial w_1} J(\vec{w}, b) \\ w_2' &= w_2 - \alpha \frac{\partial}{\partial w_2} J(\vec{w}, b) \\ &\vdots \qquad\qquad \vdots \end{aligned}}$$

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

$\boxed{\alpha = \text{Learning rate}}$

Vector Form

$$\vec{w}' = \vec{w} - \alpha \frac{\partial}{\partial \vec{w}} J(\vec{w}, b) \coloneqq \vec{w} - \alpha \vec{\nabla} J(\vec{w}, b)$$

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

Digression: 편미분 (Partial Differential)

Gradient Vector $\vec{\nabla}$: 어떻게 계산?

Note: 편미분 할 때는 미분하는 변수 외의 나머지 변수는 상수로 둔다.

Ex. $J(\vec{w}, b) = w_2^2 + 4w_1w_2 + 2w_1^2 + b^2,\ \alpha = 0.5$

$$\begin{cases} \dfrac{\partial}{\partial b} J(\vec{w}, b) = 2b \\ \vec{\nabla} J(\vec{w}, b) = \boxed{4(w_1 + w_2)\hat{x}_1} + \boxed{2(2w_1 + w_2)\hat{x}_2} \end{cases}$$

x_1방향 ($\hat{x}_1$)의 w_1의 변화

x_2방향 ($\hat{x}_2$)의 w_2의 변화

업데이트된 변수

$$\begin{cases} w_1' = w_1 - 0.5(4w_1 + 4w_2) = -w_1 - 2w_2 \\ w_2' = w_2 - 0.5(4w_1 + 2w_2) = -2w_1 \\ \vec{w} = -(w_1 + 2w_2)\hat{x}_1 - 2w_1\hat{x}_2 \\ b' = b - 0.5(2b) = 0 \end{cases}$$

Gradient Descent: Linear Regression

$$J(\vec{w}, b) := \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \hat{Y}^{(i)}\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \vec{w}\cdot\vec{x}^{(i)} - b\right)^2$$

1) $\vec{\nabla} J(\vec{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial \vec{w}}\left(Y^{(i)} - \vec{w}\cdot\vec{x}^{(i)} - b\right)^2$

2) $\frac{\partial}{\partial w_1} J(\vec{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\boxed{2}(\boxed{-}x_1^{(i)})\left(Y^{(i)} - w_1 x_1^{(i)} - w_2 x_2^{(i)} - \cdots - b\right)$

제곱을 미분해서 나옴

- 부호가 붙음!

제곱에서 1승으로 떨어짐

Note: $\frac{d}{dx} f\left(g(x)\right) = f'\left(g(x)\right)g'(x)$ (기억이 안나면 "합성함수의 미분"이라고 검색하세요!)

3) $w_2, w_3, \ldots, w_d, b$에 대해서 편미분 시행

Gradient Descent: Linear Regression

$$J(\vec{w}, b) \coloneqq \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \hat{Y}^{(i)}\right)^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y^{(i)} - \vec{w}\cdot\vec{x}^{(i)} - b\right)^2$$

앞선 결과들을 정리하면 (Proof!)

$$\vec{w}' = \vec{w} - \alpha\vec{\nabla}J(\vec{w}, b) = \vec{w} - \frac{2\alpha}{n}\sum_{i=1}^{n}(\vec{w}\cdot\vec{x} + b \ - Y^{(i)})\vec{x}^{(i)}$$

$$b' = b - \alpha\frac{\partial}{\partial b}J(\vec{w}, b) = b - \frac{2\alpha}{n}\sum_{i=1}^{n}\ (\vec{w}\cdot\vec{x} + b \ - Y^{(i)})$$

Gradient Descent: Logistic Regression

$$J(\vec{w}, b) \coloneqq -\frac{1}{n}\sum_{i=1}^{n}\left[Y^{(i)}\log\left(\frac{1}{1+e^{-\left(\vec{w}\cdot\vec{x}^{(i)}+b\right)}}\right)+\left(1-Y^{(i)}\right)\log\left(1-\frac{1}{1+e^{-\left(\vec{w}\cdot\vec{x}^{(i)}+b\right)}}\right)\right]$$

선형회귀와 비슷하게 회귀방정식 (예측값)과
실제값의 차이에 비례하는 형태로 나옴

$$\vec{w}' = \vec{w} - \alpha\vec{\nabla}J(\vec{w}, b) = \vec{w} - \frac{2\alpha}{n}\sum_{i=1}^{n}\left(\frac{1}{1+e^{-\left(\vec{w}\cdot\vec{x}^{(i)}+b\right)}} - Y^{(i)}\right)\vec{x}^{(i)}$$

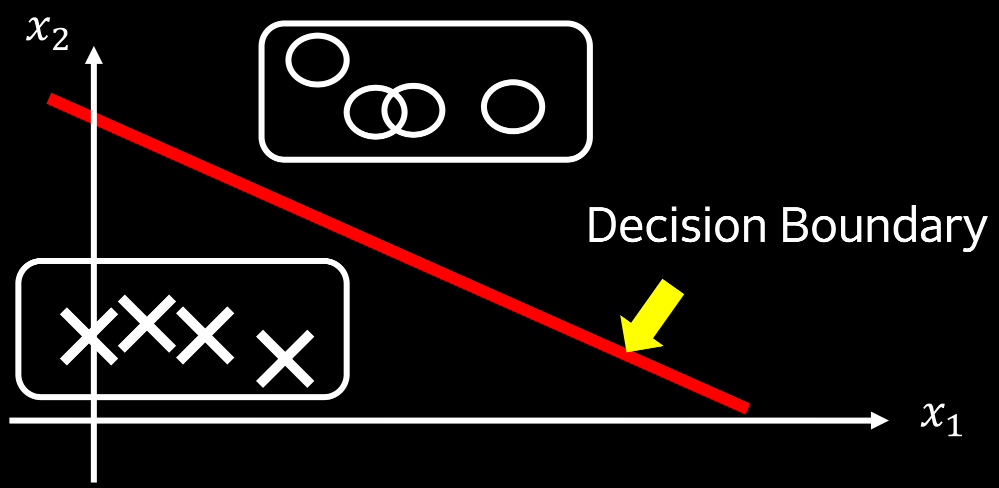
$$b' = b - \alpha\frac{\partial}{\partial b}J(\vec{w}, b) = b - \frac{2\alpha}{n}\sum_{i=1}^{n}\left(\frac{1}{1+e^{-\left(\vec{w}\cdot\vec{x}^{(i)}+b\right)}} - Y^{(i)}\right)$$

Proof는 스킵합니다.

3. Feature maps and Kernel Method

Classification using Support Vector Machine (SVM)

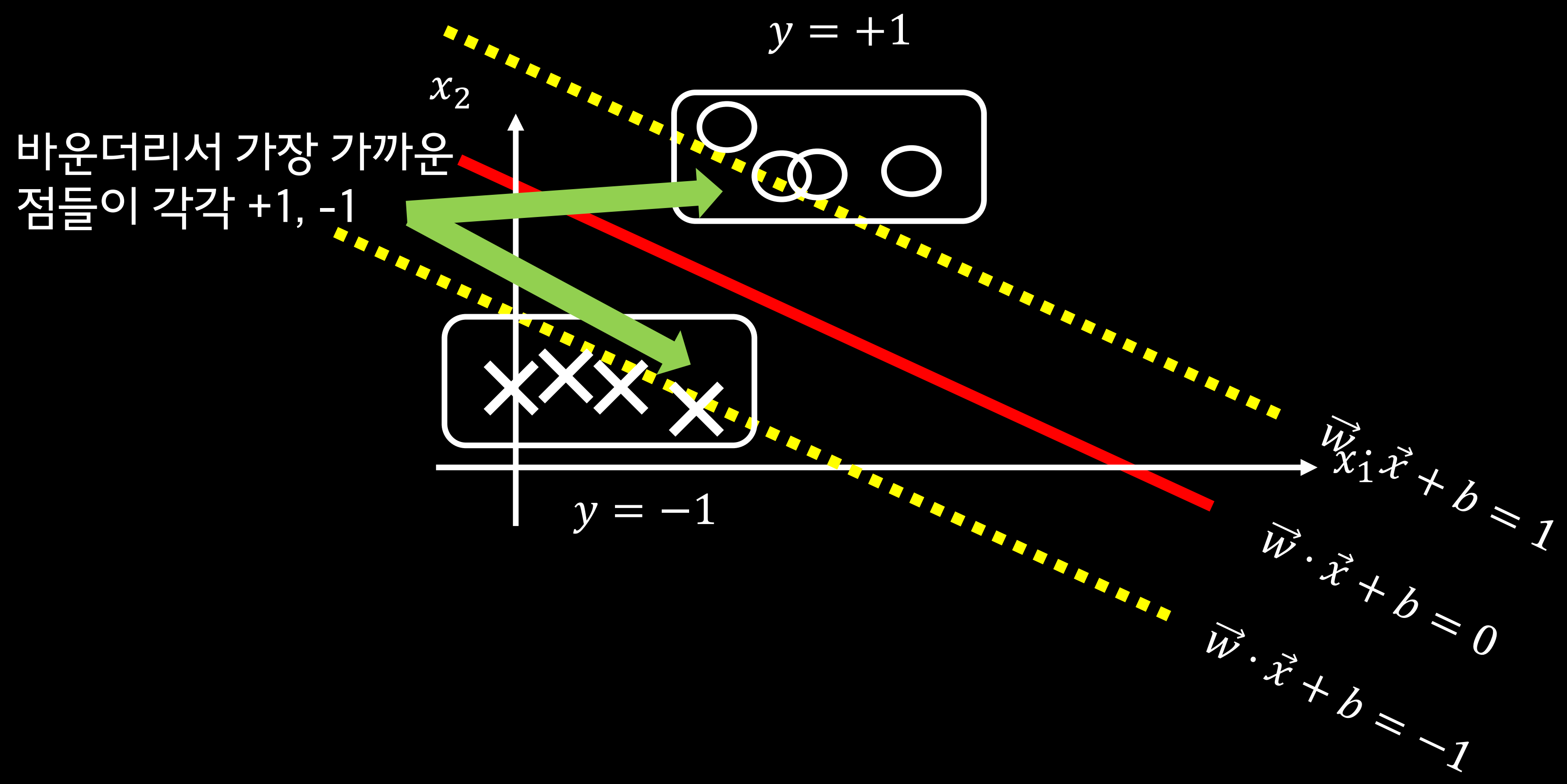
회귀곡선을 찾는 대신 데이터의 그룹을 찾기 (두 개의 카테고리)



Q. Decision Boundary를 결정하는 방법?

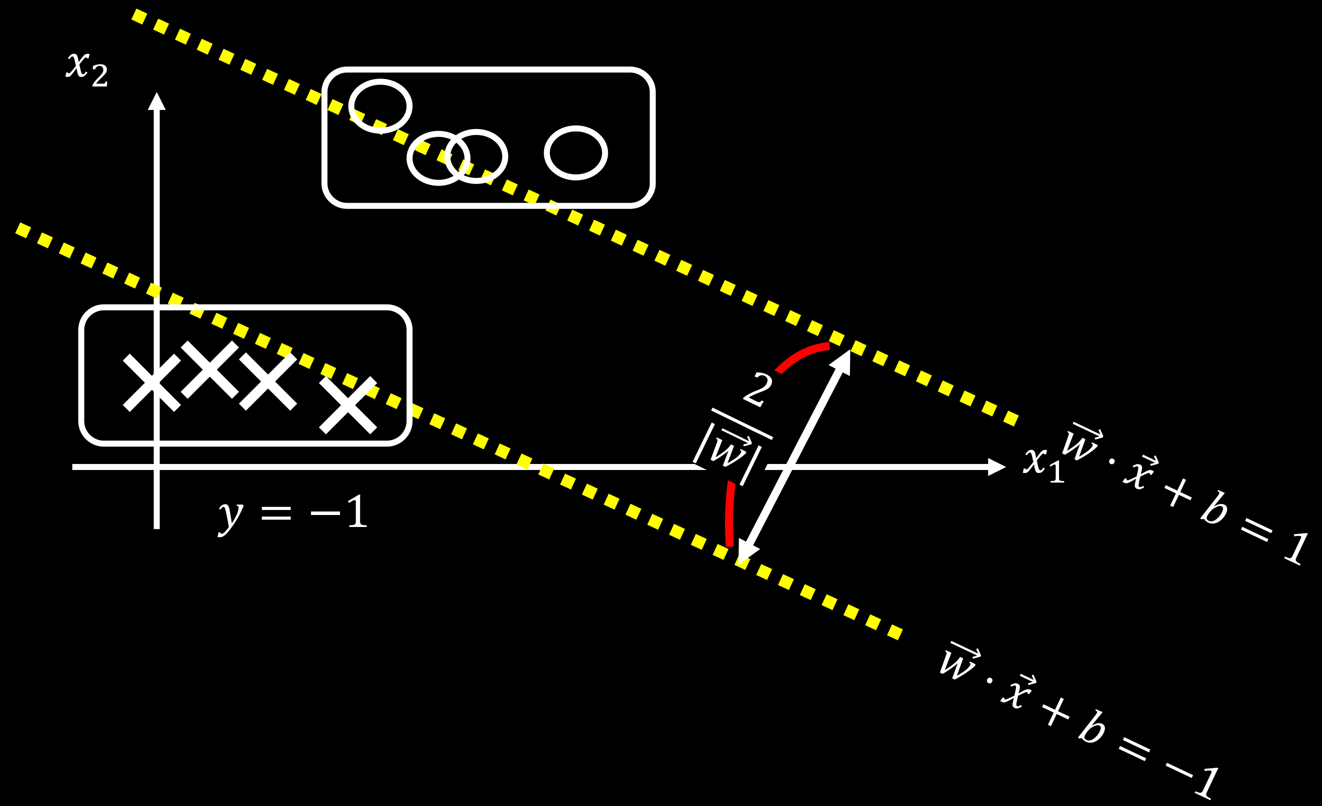
Support Vector Machine (SVM)

Idea: Boundary in Linear Regression form



Support Vector Machine (SVM)

Idea: 최대한 두 그룹 간의 거리가 멀었으면! ➔ 구분이 잘 되었으면



Goal: 사이 거리가 최대가 되도록! ➔ $|\vec{w}|$가 최소가 되도록!

Digression: Lagrange function

Condition: $|\vec{w}|$를 최소화 → $\frac{1}{2}|\vec{w}|^2$를 최소화

+ Restriction: $\vec{w} \cdot \vec{x}^{(i)} + b = \pm 1 \Rightarrow y^{(i)}\left(\vec{w} \cdot \vec{x}^{(i)} + b\right) > +1$ ($y^{(i)} > +1$ (평면 위), < -1 (평면 아래))

(모든 점은 노란 색 선(평면)의 경계나 그 밖에 존재한다.)

Lagrangian $L(\vec{w}, b, \lambda)$ 을 최소화 하라

$$L(\vec{w}, b, \lambda) = \frac{1}{2}|\vec{w}|^2 - \sum_{i=1}^{N} \lambda_i [y^{(i)}\left(\vec{w} \cdot \vec{x}^{(i)} + b\right) - 1]$$

각 데이터 포인트마다 Lagrange multiplier λ > 0가 필요함!

Digression: Lagrange function

Goal: Lagrangian $L(\vec{w}, b, \lambda)$ 을 최소화 하라

→ Cost function을 최소화하는 것처럼 각 편미분이 0이 되는 지점을 찾자!

$$\boxed{\vec{\nabla}L = 0 = \vec{w} - \sum_{i=1}^{N} \lambda_i \, y^{(i)} \vec{x}^{(i)}}$$

$$\boxed{\frac{\partial L}{\partial b} = 0 = -\sum_{i=1}^{N} \lambda_i \, y^{(i)}}$$

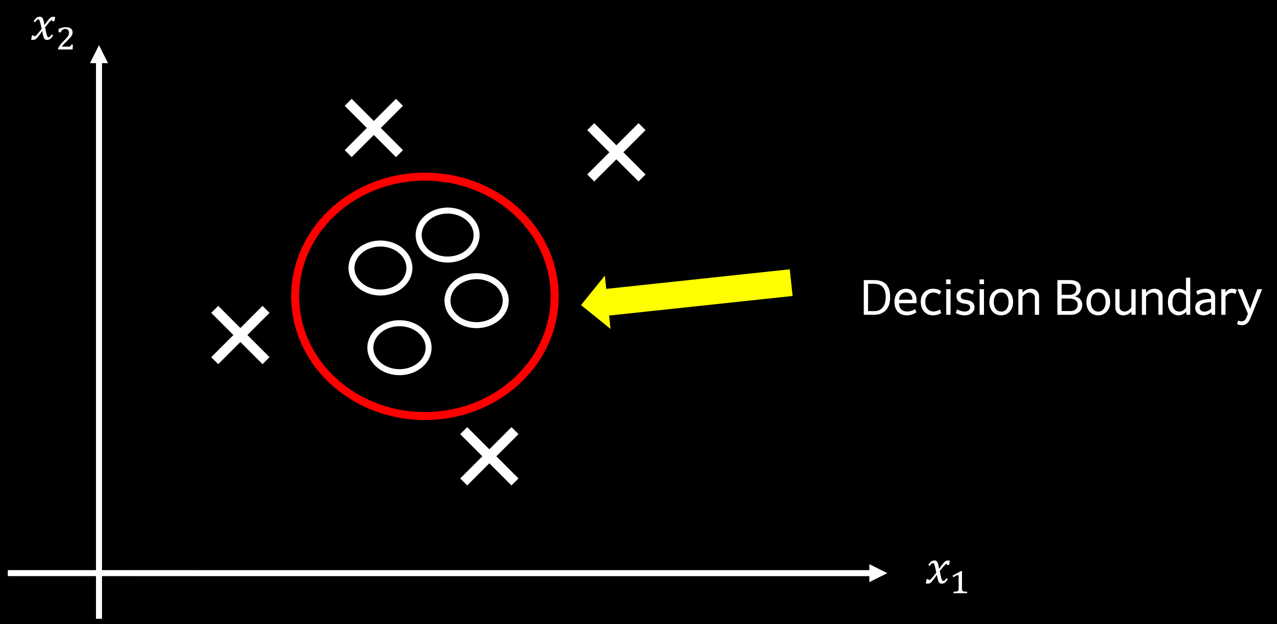
$$\Downarrow$$

Lagrangian $L(\lambda)$ 을 최소화 하라

$$L(\lambda) = \sum_{i=1}^{N} \lambda_i - \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y^{(i)} y^{(j)} \vec{x}^{(i)} \vec{x}^{(j)}$$

Kernel Method

Question: Boundary가 선형이 아닌 비선형이어야 하는 경우에는?



Feature mapping

각 Feature (x들)을 비선형 변환하여 새로운 벡터로 변환 ➔ Weight vector도 새로운 벡터에 따라서 정의해야 함

Ex)

$$x \rightarrow \boxed{\vec{\Phi}(x)} \rightarrow \vec{\Phi}(x) = [x, x^2]$$

Lagrangian $L(\lambda)$ 을 최소화 하라

$$L(\lambda) = \sum_{i=1}^{N} \lambda_i - \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i \lambda_j y^{(i)} y^{(j)} \left(\vec{x}^{(i)} \cdot \vec{x}^{(j)}\right)$$

$$\downarrow$$

$$L(\lambda) = \sum_{i=1}^{N} \lambda_i - \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i \lambda_j y^{(i)} y^{(j)} \, \vec{\Phi}\left(\vec{x}^{(i)}\right) \cdot \vec{\Phi}\left(\vec{x}^{(j)}\right)$$

Feature mapping

문제점 1. $\Phi(\vec{x})$를 일반적으로 어떻게 찾아야 하나?

문제점 2. 차원이 커지기 때문에 계산량이 많아진다.

~~문제점 3. 머신러닝에 정이 떨어지기 시작한다.~~

Kernel Method

Answer: Kernel을 도입하여 벡터의 내적으로 표현한다.

$$K\left(\vec{x}^{(i)}, \vec{x}^{(j)}\right) := \vec{\Phi}\left(\vec{x}^{(i)}\right) \cdot \vec{\Phi}(\vec{x}^{(j)})$$

Point 1: 대칭이다.

Point 2: 대표적인 커널들이 알려져 있으므로, $\vec{\Phi}$ 를 몰라도 K만을 이용해서 Lagrangian 계산 가능

Linear

$$K\left(\vec{x}^{(i)}, \vec{x}^{(j)}\right) = \vec{x}^{(i)} \cdot \vec{x}^{(j)}$$

Sigmoid

$$K\left(\vec{x}^{(i)}, \vec{x}^{(j)}\right) = \tanh(a(\vec{x}^{(i)} \cdot \vec{x}^{(j)}) + b)$$
$$(a, b > 0)$$

Polynomial

$$K\left(\vec{x}^{(i)}, \vec{x}^{(j)}\right) = \left(\vec{x}^{(i)} \cdot \vec{x}^{(j)} + \mathrm{c}\right)^{d}$$
$$(c > 0, d > 1)$$

Gaussian

$$K\left(\vec{x}^{(i)}, \vec{x}^{(j)}\right) = \exp(-\left|\vec{x}^{(i)} - \vec{x}^{(j)}\right|^{2} / 2\sigma^{2})$$
$$(\sigma > 0)$$

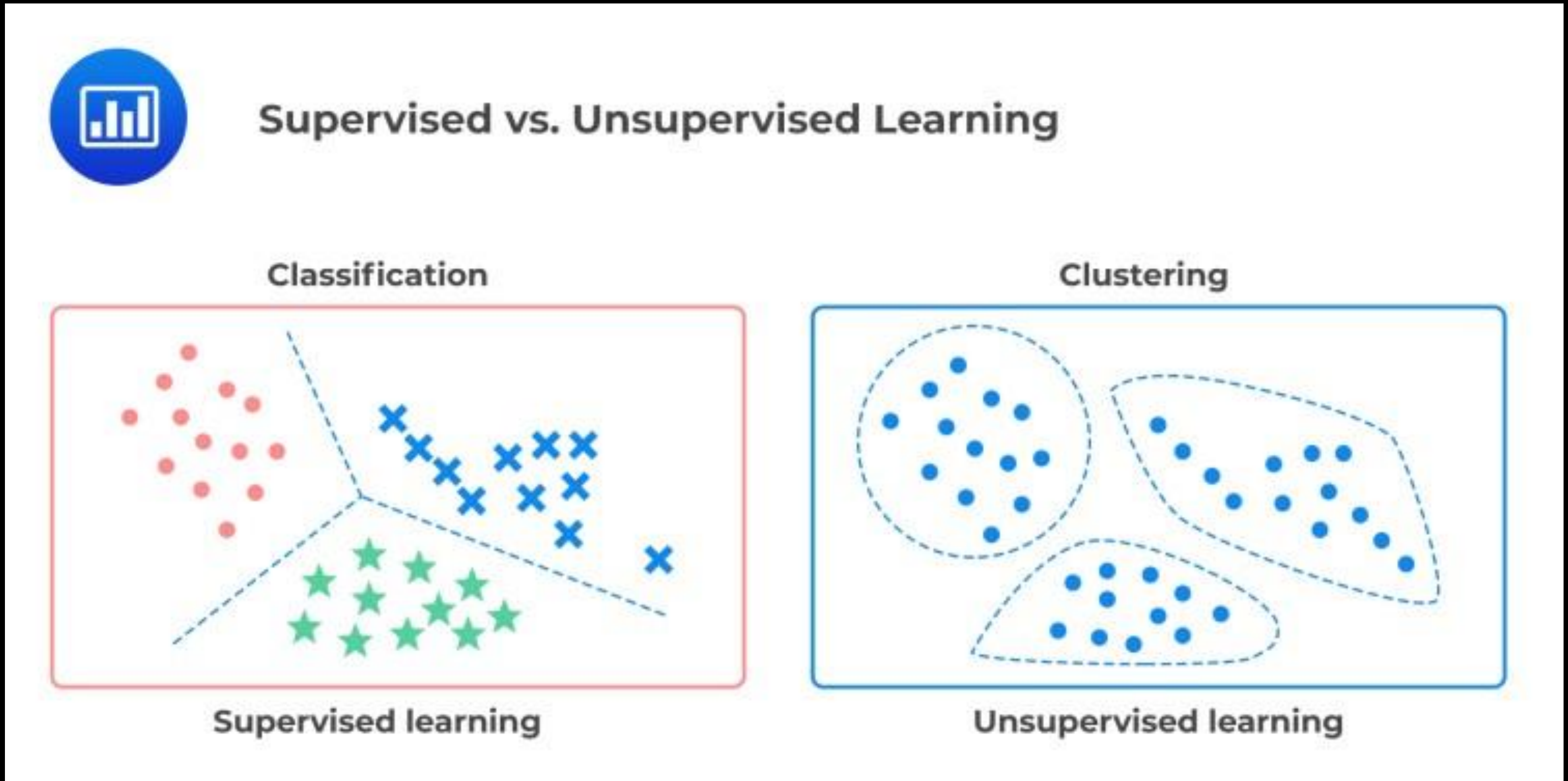
Kernel Method in Quantum Computer

선형성을 띄는 것 ➔ 양자컴퓨터에서 효율적으로 처리 가능

텍스트북 세션 Quantum Feature maps and Kernels 참고!

4. Unsupervised Learning

Supervised vs Unsupervised Learning



Label이 있음

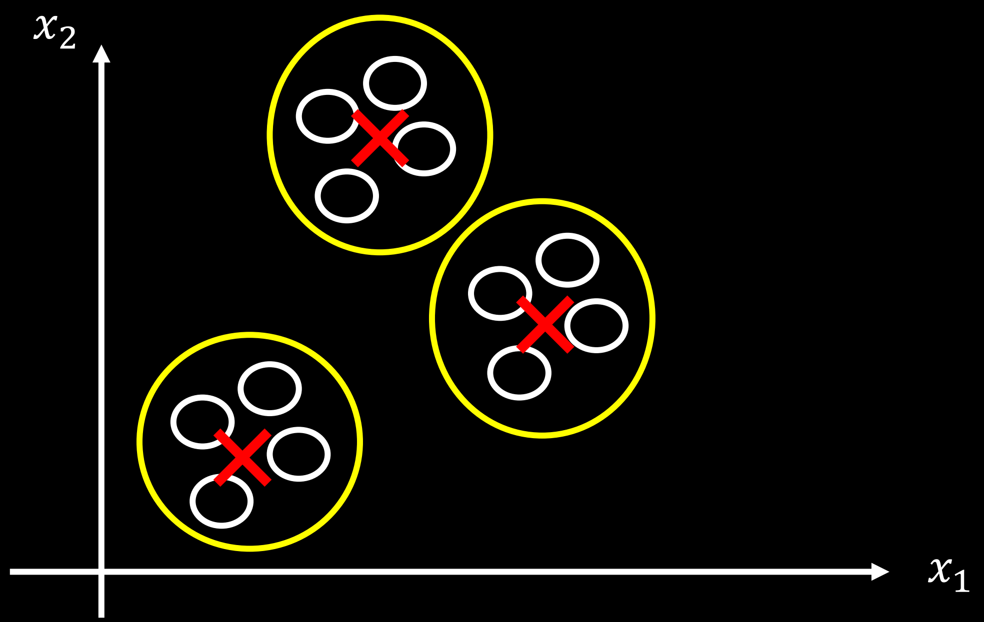
Label이 없음

Reference: https://www.linkedin.com/pulse/supervised-vs-unsupervised-learning-whats-difference-smriti-saini/

K-Means Clustering

- 데이터를 K개의 클러스터로 나눈다.

$K = 3$ Clustering



Q. 어떻게 학습?

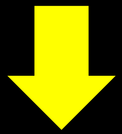
A. 각 클러스터의 중심(X)이 각 데이터와의 거리가 최소화 되도록 학습

Dimensional Reduction

- Feature들 중 중요한 feature만 뽑아내거나
- Feature를 적절히 조합하여 중요한 feature 조합만 뽑아낸다.

Feature Selection

$$\vec{x} = [x_1, x_2, x_3, x_4, x_5, x_6]$$



$$\vec{x}' = [x_1, x_5]$$

Principle Component Analysis

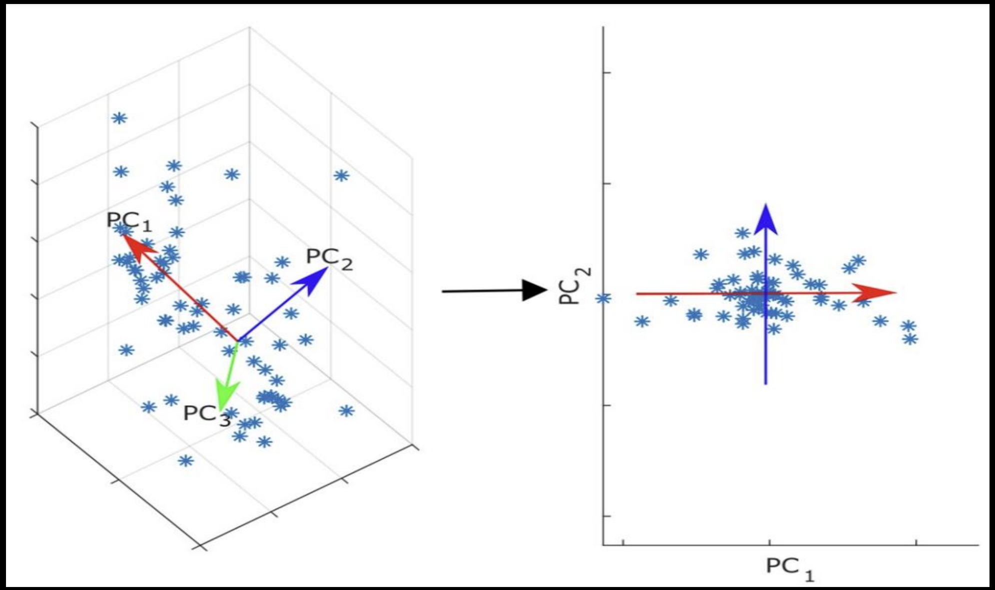
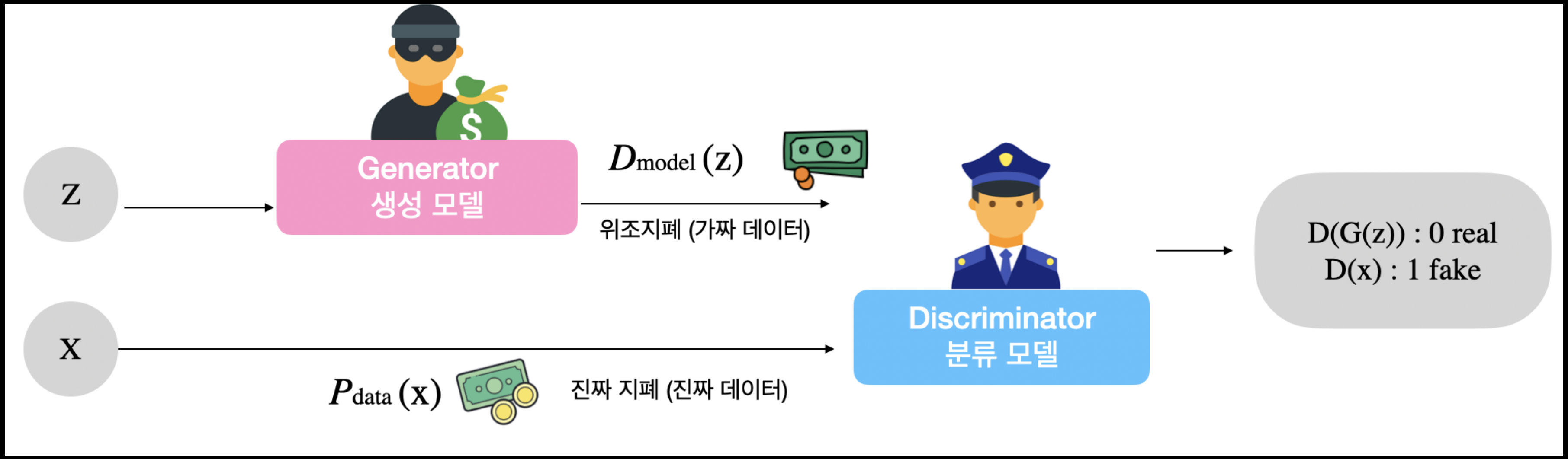


Figure reference: Migenda N, Möller R, Schenck W, PLOS ONE 16(3): e0248896 (2021)

Generative Adversarial Network (GAN)

Goal: 위조지폐를 만들어보자 (?!!)



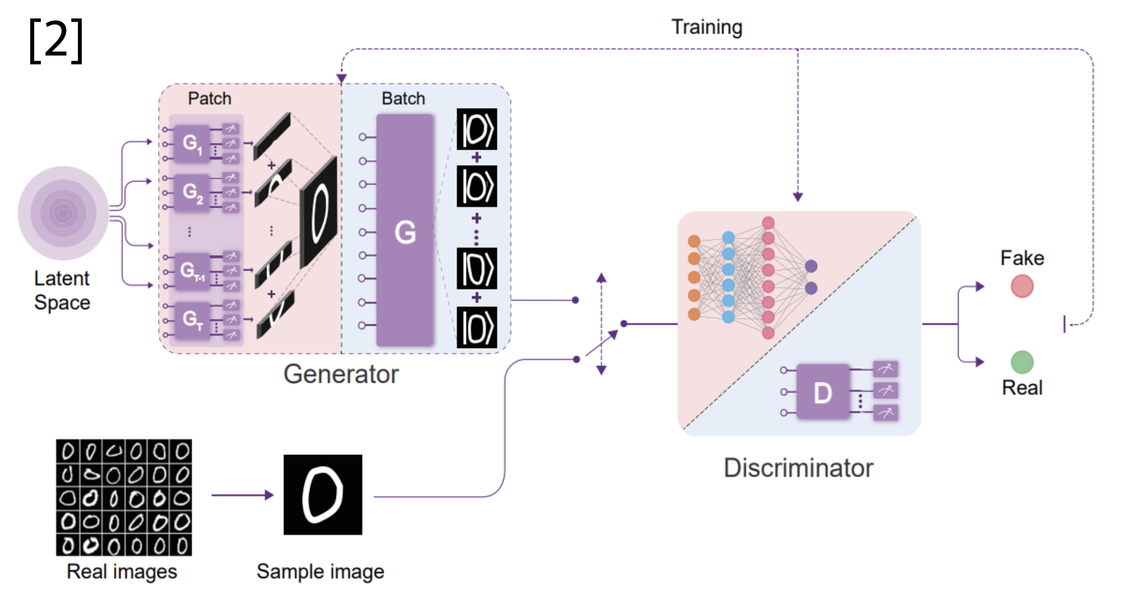
Idea: Discriminator가 실제와 생성된 가짜 모델을 구별하고 생성 모델은 실제 모델을 학습하여 실제와 가짜를 구별할 수 없게끔 (adversarial) 만든다.

Figure from https://wegonnamakeit.tistory.com/54

Quantum Generative Adversarial Network (QuGAN)

Goal: 양자 상태를 최대한 비슷하게 복사해보자!

Advantage: Classical model에 비해서 Exponential speedup이 가능하다 [1]



현 수준 초전도체 양자컴퓨터에서 구현

텍스트북 세션 Quantum Generative Adversarial Network 참고!

Reference: [1] Quantum Machine Intelligence 4(28) (2022), [2] Phys. Rev. Applied 16, 024051 (2021)

Summary

- Machine Learning: 주어진 데이터를 기반 추론 학습
- Supervised Learning: 회귀(Regression), 분류(Classification), …
- Gradient Descent ➔ Cost Function의 최저점을 찾아가는 과정
- Feature space and Kernel method: Nonlinear ➔ Linear
- Unsupervised Learning: Clustering, Dimensional Reduction, GAN, …