4/2 목

M2608.001300
Machine Learning Fundamentals & Applications

[2: Feasibility of Learning]

Electrical and Computer Engineering
Seoul National University

Outline

Readings

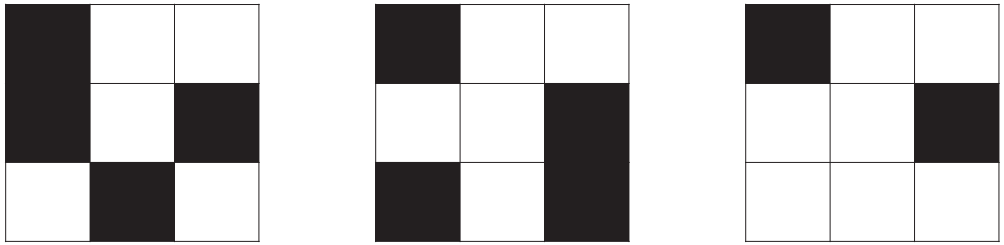
- *Learning from Data* by Abu-Mostafa, Magdon-Ismail, and Lin
  - ▷ Chapter 1: The Learning Problem (Sections 1.3 & 1.4)

Outline

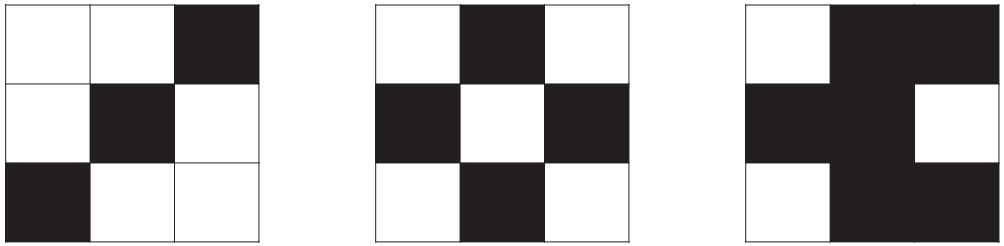
Learning the target function

- target function f = learning objective
  - most important fact about f: it is unknown
- natural question
  - how could a limited data set reveal enough information to determine the entire f?

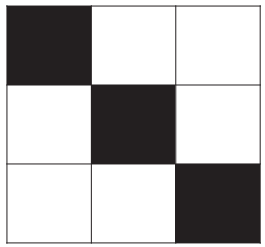
- 6 training examples of a binary target function f



$f = -1$



$f = +1$



$f = ?$

- difficulty in this simple problem: the rule, not the exception

Inference outside training data

- in training data $\mathcal{D}$, we know the value of f on all points
  - but this does not mean that we have learned f
- key question: does $\mathcal{D}$ tell us anything outside $\mathcal{D}$ we did not know before?
  - if so, then we have learned *something*
  - otherwise, we can conclude: learning is not feasible

Example

- consider a Boolean f over 3D input space $\mathcal{X} = \{0, 1\}^3$
  - # distinct input vectors: $2^3 = 8$
  - # distinct Boolean functions on 3 Boolean inputs: $2^{2^3} = 2^8 = 256$

- given: a data set $\mathcal{D}$ of five examples:

$\mathbf{x}_n$	y_n
000	○
001	●
010	●
011	○
100	●

- task: predict f on points never seen (= outside $\mathcal{D}$) before

- any function that agrees with $\mathcal{D}$ could be f (*e.g.* $f_1, \ldots, f_8$ below)

$\mathbf{x}$	y	g	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
000	○	○	○	○	○	○	○	○	○	○
001	●	●	●	●	●	●	●	●	●	●
010	●	●	●	●	●	●	●	●	●	●
011	○	○	○	○	○	○	○	○	○	○
100	●	●	●	●	●	●	●	●	●	●
101		?	○	○	○	○	●	●	●	●
110		?	○	○	●	●	○	○	●	●
111		?	○	●	○	●	○	●	○	●

- purpose of learning
  - find such g that agrees the most with f

- candidates of g
  - g_1: returns • for all the three points
  - g_2: returns ○ for all the three points
  - g_3: returns XOR($\mathbf{x}$)[1] for all the three points
  - g_4: returns the opposite of g_3 for all the three points

$\mathbf{x}$	y	g	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	g_1	g_2	g_3	g_4
000	○	○	○	○	○	○	○	○	○	○				
001	•	•	•	•	•	•	•	•	•	•				
010	•	•	•	•	•	•	•	•	•	•				
011	○	○	○	○	○	○	○	○	○	○				
100	•	•	•	•	•	•	•	•	•	•				
101		?	○	○	○	○	•	•	•	•	•	○	○	•
110		?	○	○	•	•	○	○	•	•	•	○	○	•
111		?	○	•	○	•	○	•	○	•	•	○	•	○

- which is the best?

[1] XOR($\mathbf{x}$) returns • if # 1's in $\mathbf{x}$ is odd; returns ○ otherwise

$\mathbf{x}$	y	g	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	g_1	g_2	g_3	g_4
000	○	○	○	○	○	○	○	○	○	○				
001	●	●	●	●	●	●	●	●	●	●				
010	●	●	●	●	●	●	●	●	●	●				
011	○	○	○	○	○	○	○	○	○	○				
100	●	●	●	●	●	●	●	●	●	●				
101		?	○	○	○	○	●	●	●	●	●	○	○	●
110		?	○	○	●	●	○	○	●	●	●	○	○	●
111		?	○	●	○	●	○	●	○	●	●	○	●	○

- performance evaluation
  - in terms of # points in agreement between g_k and f_i

g_k	on three	on two	on one	on none
g_1	f_8	f_4, f_6, f_7	f_2, f_3, f_5	f_1
g_2	f_1	f_2, f_3, f_5	f_4, f_6, f_7	f_8
g_3	f_2	~~f_1 f_4 f_6~~	~~f_3 f_5 f_8~~	f_7
g_4	f_7	f_3, f_5, f_8	f_1, f_4, f_6	f_2

Are we doomed?

- no matter what $\mathcal{A}$ does or what $\mathcal{H}$ is used
  - no difference as far as performance outside $\mathcal{D}$ is concerned

    ↑ all that matters in learning

- learning an unknown function f? $\Rightarrow$ impossible!
  - f can assume any value outside $\mathcal{D}$
  - we can even prove: f remains unknown outside $\mathcal{D}$

- so what now?

  probability 를 이용해 f와 확률적으로 유사한 g를 학습하는 것이 가능하다.

Outline

Rescue plan

- probability
- we can infer something outside $\mathcal{D}$ using only $\mathcal{D}$
  - in a probabilistic way
  - we give up knowing about f "for sure"

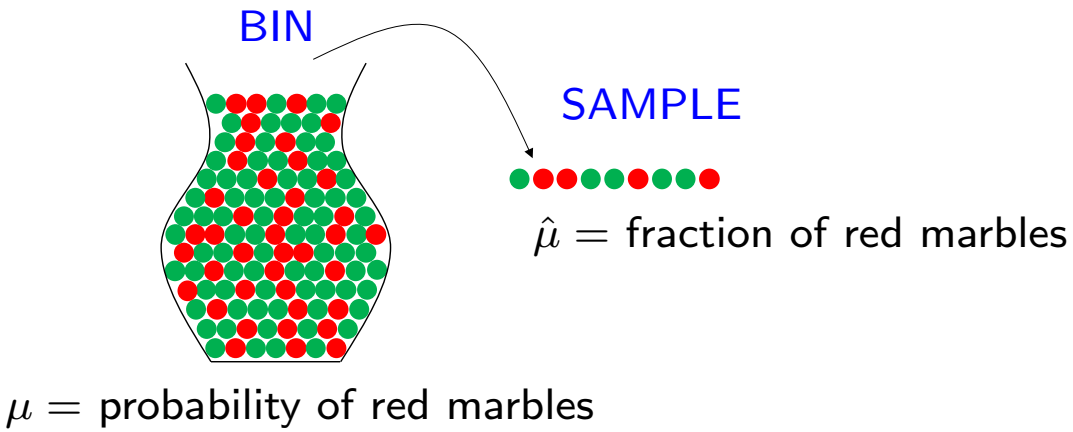
A 'bin' experiment

- consider a bin with red and green marbles (possibly infinitely many)

$$\mathbb{P}[\text{picking a red marble}] = \mu, \qquad \mathbb{P}[\text{picking a green marble}] = 1 - \mu$$

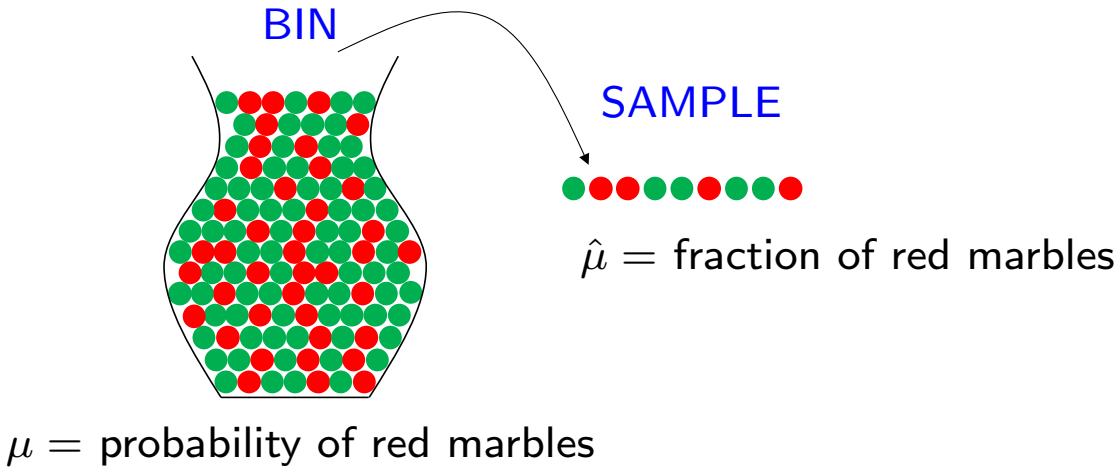
  - the value of μ: unknown

- pick N marbles independently (with replacement)
  - the faction of red marbles within the sample $\triangleq \hat{\mu}$



Does $\hat{\mu}$ say anything about μ?

- no *e.g.* sample can be mostly green while bin is mostly red
  - *possible* but not *probable*
- yes sample frequency $\hat{\mu}$ is likely ~~unlikely~~ bin frequency μ



- how to quantify the relationship between $\hat{\mu}$ and μ?

Hoeffding inequality

for any sample size N and $\epsilon > 0$:

$$\mathbb{P}[|\hat{\mu} - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N} \tag{1}$$

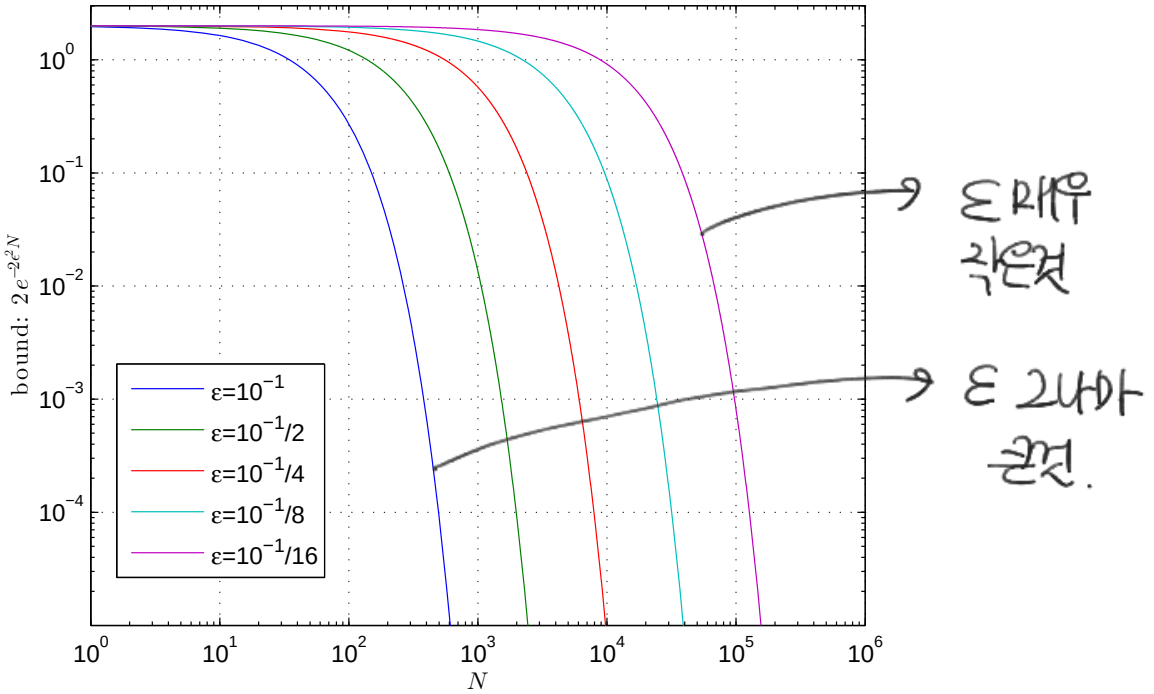
- $\mathbb{P}$: probability of an event (here, wrt random sample)
- the bound does not depend on μ or on bin size

- in words
  - as the sample size N grows, it becomes exponentially unlikely that $\hat{\mu}$ will deviate from μ by more than our 'tolerance' ϵ

- in other words, the statement "$\mu = \hat{\mu}$" is
  - PAC (probably approximately correct)

$$\mathbb{P}[|\hat{\mu} - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N} \quad (1)$$

- if we choose ϵ to be very small (to make $\hat{\mu}$ approximate μ well)
  - need a big sample size N to make the rhs of (1) small



Wassily Hoeffding

- the key player in the bound $2e^{-2\epsilon^2 N}$: N
  - if $N \to \infty$, $\mu \approx \hat{\mu}$ with very high probability (albeit *not for sure*)

Example

$$\mathbb{P}[|\hat{\mu} - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N} \quad (1)$$

- $N = 1,000$

$$\begin{aligned} 99\% \text{ of the time} \quad & \mu - 0.05 \leq \hat{\mu} \leq \mu + 0.05 \quad (\epsilon = 0.05) \\ 99.9999996\% \text{ of the time} \quad & \mu - 0.10 \leq \hat{\mu} \leq \mu + 0.10 \quad (\epsilon = 0.10) \end{aligned}$$

- meaning: if we repeatedly pick a sample of size 1,000

$$\mu \in [\hat{\mu} - 0.05, \hat{\mu} + 0.05]$$

  - and we are right 99% of the time about this

- we learned *something* : we reached outside $\mathcal{D}$ to μ

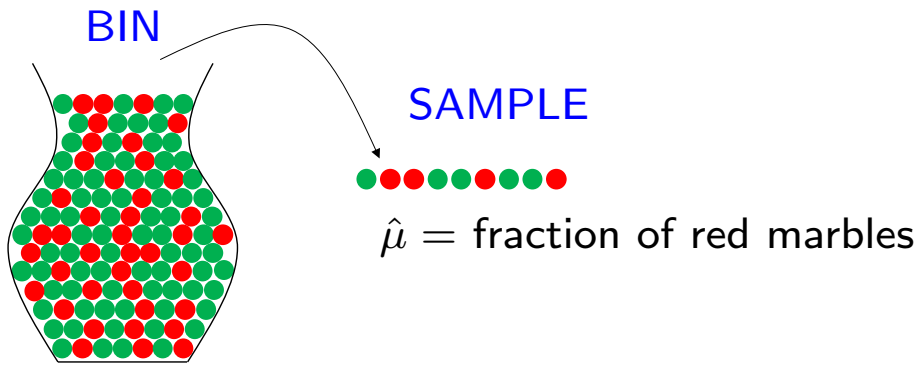
Outline

Are the bin model and learning connected?

- seemingly different
  - unknown in the bin problem: just the value of μ
  - unknown in learning: an entire function $f : \mathcal{X} \to \mathcal{Y}$

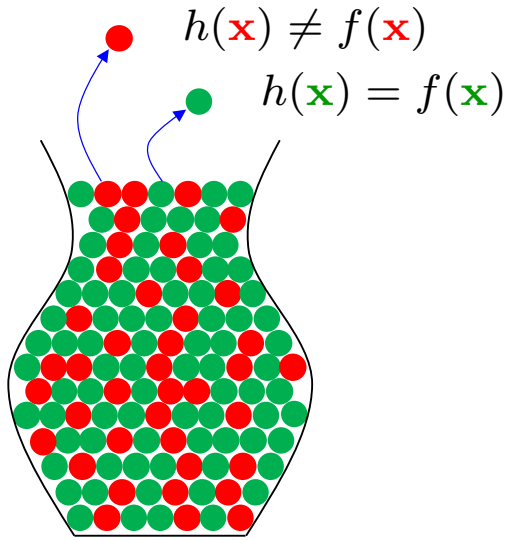
- but we can connect them:
  - $\mu =$ error rate a hypothesis makes when approximating f

- obvious
  - marble $\equiv \mathbf{x} \in \mathcal{X}$
  - bin $\equiv \mathcal{X}$
  - sample $\equiv \mathcal{D}$
- less obvious
  - ball color $\equiv$?
  - $\mu \equiv$?
  - $\hat{\mu} \equiv$?



Bin coloring $\equiv$ hypothesis h

- take a *particular* hypothesis $h \in \mathcal{H}$ and compare it to f on each $\mathbf{x} \in \mathcal{X}$:
  - color the point $\mathbf{x}$ $\begin{cases} \text{green} & \text{if } h(\mathbf{x}) = f(\mathbf{x}) : \text{hypothesis right} \\ \text{red} & \text{if } h(\mathbf{x}) \neq f(\mathbf{x}) : \text{hypothesis wrong} \end{cases}$



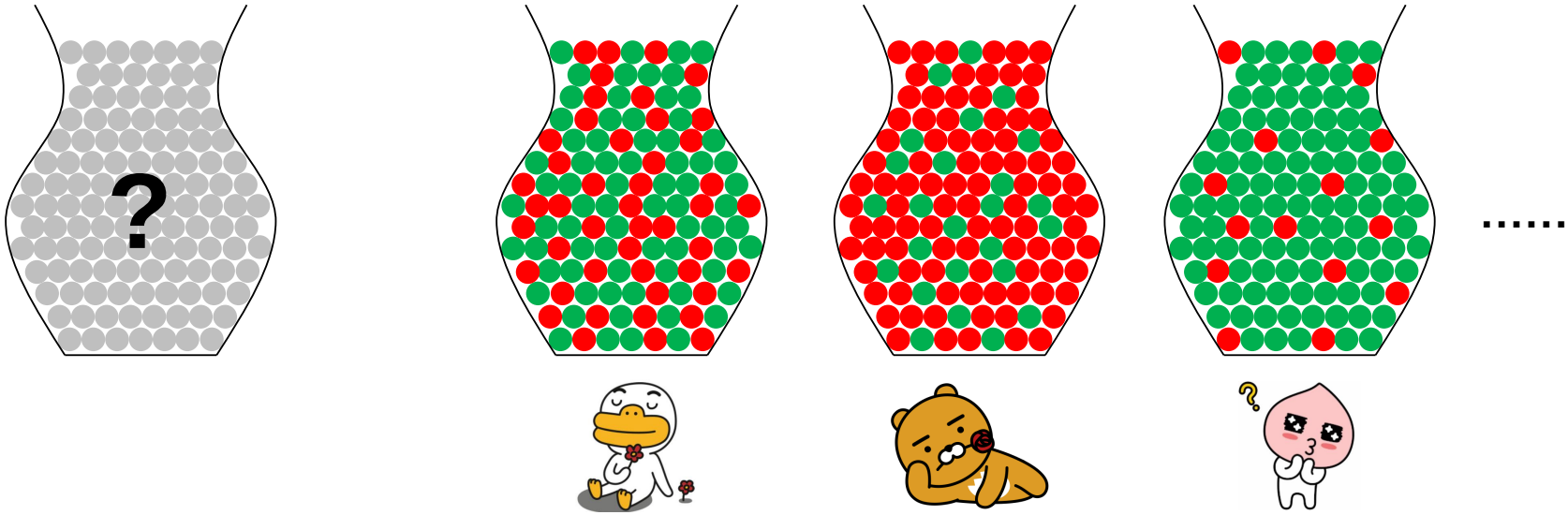
- this means:
  - $\mu \equiv$ error rate a hypothesis makes when approximating f

즉, 통계학과 연결시켜 우리의 g를 평가할수 있다.

- different h

  $\Rightarrow$ different coloring of the balls in the bin

  $\Rightarrow$ different μ (*i.e.* error rate depends on the hypothesis used)



- learning algorithm $\mathcal{A}$ picks the best h (*i.e.* g)

Sample $\equiv$ training data set $\mathcal{D}$

- points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ in $\mathcal{D}$
  - picked independently according to distribution P
  - a random sample of red and green points
  - the color of each point in $\mathcal{D}$ will be known to us
- P: some probability distribution over $\mathcal{X}$ 확률에 의해 D를 pick.
  - μ is allowed to be unknown $\longrightarrow$ P can be unknown as well
- $\hat{\mu}$ estimates μ, telling us something about f

  i.e. if $\hat{\mu} \approx 0 \Rightarrow h \approx f$ over entire $\mathcal{X}$

Revised: the basic learning step

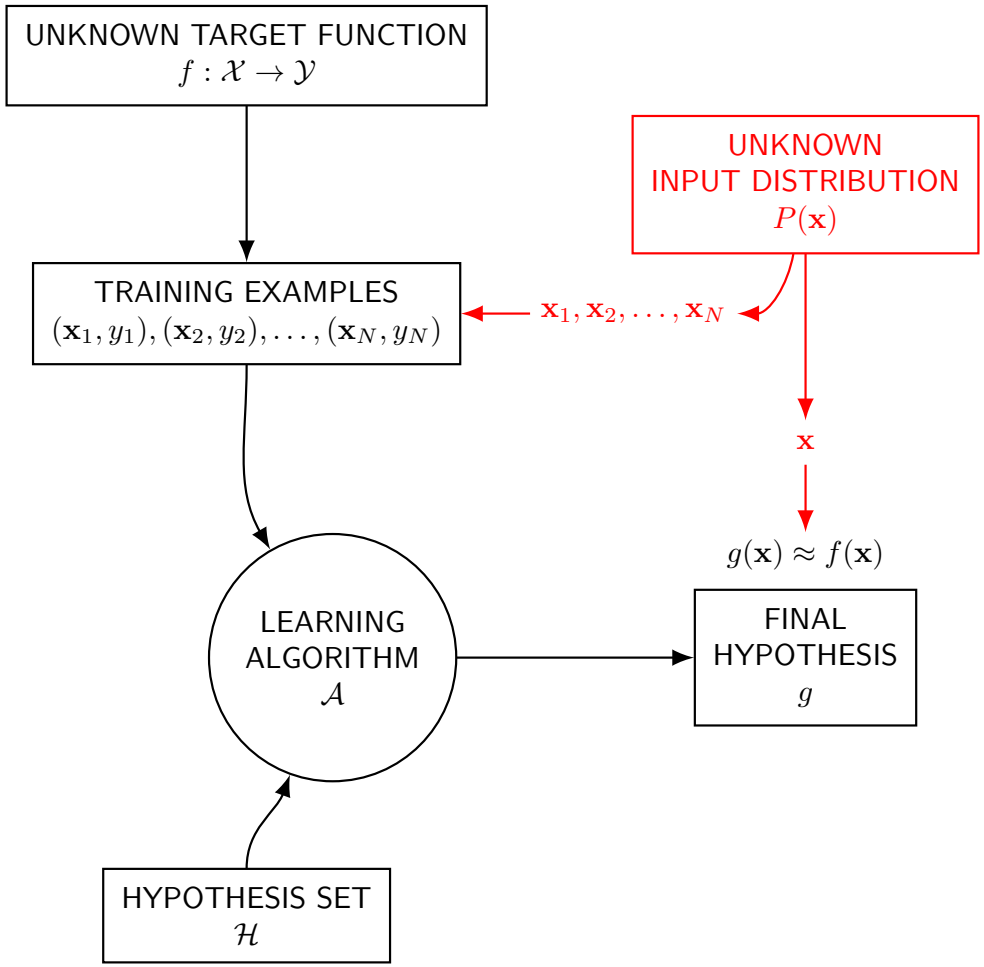


Figure 1: probabilistic component added to the basic learning step

Outline

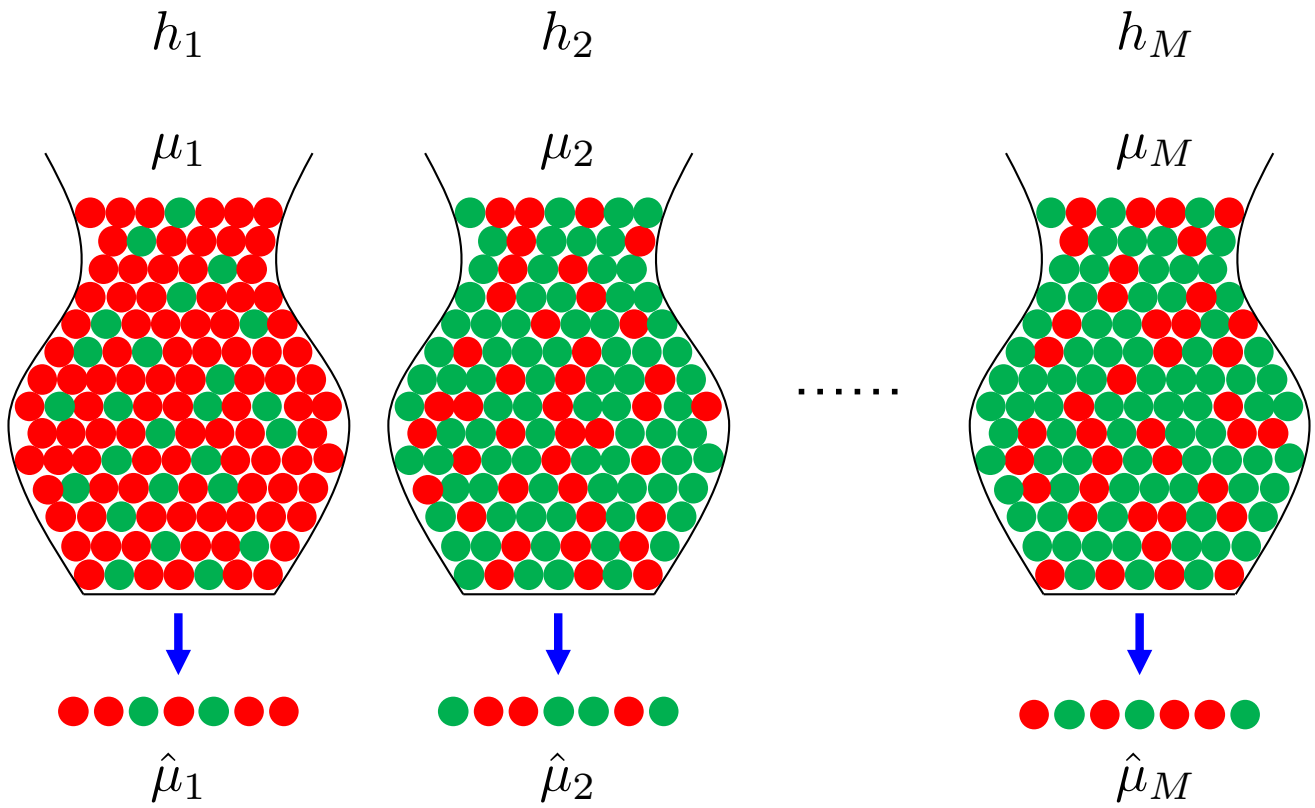
Are we done?

- no. how do we know $\hat{\mu} \approx 0$ in general?
- in real learning
  - explore an entire hypothesis set $\mathcal{H}$
  - look for some $h \in \mathcal{H}$ that has a small error rate
  - choose the best from multiple h's

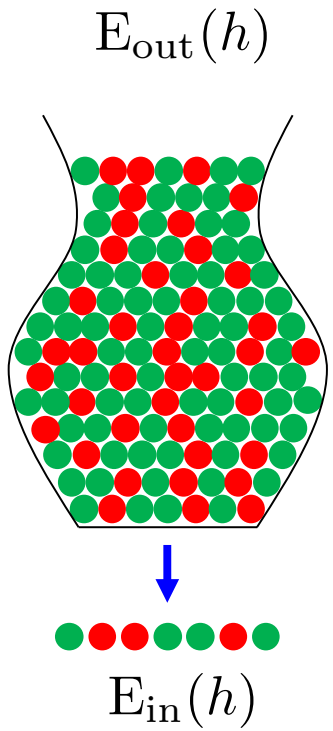
for example, "selection bias 문제" is one of the challenging problem.

Multiple bins

- in order to capture real learning
  - extend the bin equivalence to multiple hypotheses
  - different $h \rightarrow$ different $\hat{\mu} \rightarrow$ different approximation of μ



- both $\hat{\mu}$ and our belief in μ depend on which hypothesis h is used
  - $\hat{\mu}$ is in-sample error denoted by $\mathrm{E}_{\mathrm{in}}(h)$ — Ein: in sample error, function of h.
  - μ is out of sample error denoted by $\mathrm{E}_{\mathrm{out}}(h)$



- Hoeffding inequality now becomes

$$\mathbb{P}[|\mathrm{Ein}(h)-\mathrm{Eout}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N} \qquad (2)$$

N will dependent

In-sample error $\mathrm{E}_{\mathrm{in}}(h)$

- error rate within a sample:

$$\mathrm{E}_{\mathrm{in}}(h) = \text{(faction of } \mathcal{D} \text{ where } f \text{ and } h \text{ disagree)}$$

$$= \frac{1}{N}\sum_{n=1}^{N} [\![h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]\!] \tag{3}$$

⇒ binary classification 의 예시 (multiple은 다른 식 사용한 것)

  - corresponds to $\hat{\mu}$ in the bin model
  - notation: $[\![\text{true}]\!] = 1$, $[\![\text{false}]\!] = 0$

- random variable that depends on the sample (just like $\hat{\mu}$)

같은 hypothesis 여도 sample 달라지면 error rate도 달라진다.

Out-of-sample error $\mathrm{E}_{\mathrm{out}}(h)$

$$\mathrm{E}_{\mathrm{out}}(h) = \mathbb{P}[h(\mathbf{x}) \neq f(\mathbf{x})] \tag{4}$$

- corresponds to μ in the bin model
- unknown but not random (just like μ)

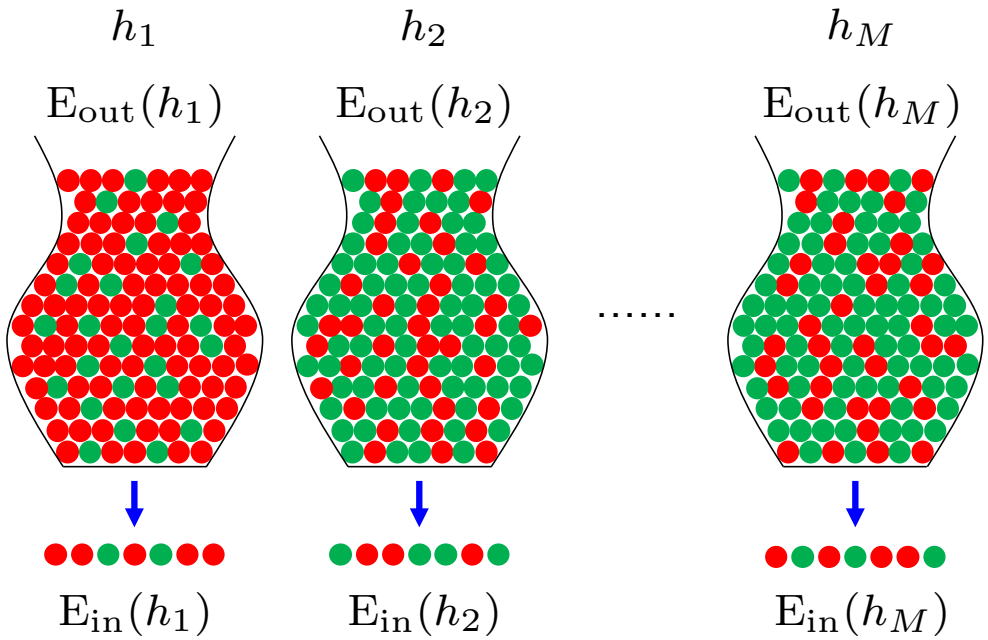
- comparison
  - E_{in} is random but known
  - $\mathrm{E}_{\mathrm{out}}$ is fixed but unknown

Multiple hypotheses

- consider an entire hypothesis set $\mathcal{H}$ instead of just one h
  - assume for now $\mathcal{H}$ has a *finite* number of hypotheses

$$\mathcal{H} = \{h_1, h_2, \ldots, h_M\} \quad M : \text{hypotheses set의 크기} \tag{5}$$

- we can then construct a bin equivalent by having M bins



Recall: basic probability rules

for events A and B:

- implication

$$\text{if } A \Rightarrow B \ (A \subseteq B) \text{ then } \mathbb{P}[A] \leq \mathbb{P}[B]$$

- union bound

$$\mathbb{P}[A \text{ or } B] = \mathbb{P}[A \cup B] \leq \mathbb{P}[A] + \mathbb{P}[B]$$

- Bayes' rule

$$\mathbb{P}[A|B] = \frac{\mathbb{P}[B|A] \cdot \mathbb{P}[A]}{\mathbb{P}[B]}$$

Uniform version of Hoeffding inequality

- Hoeffding inequality (1) does not apply to multiple bins
  - the inequality applies to each bin individually

- a simple but crude fix:
  - g has to be one of the h_m's (regardless of $\mathcal{A}$ and $\mathcal{D}$)
  - therefore

$$|\mathrm{E}_{\mathrm{in}}(g) - \mathrm{E}_{\mathrm{out}}(g)| > \epsilon \quad \Longrightarrow \quad \begin{aligned} & |\mathrm{E}_{\mathrm{in}}(h_1) - \mathrm{E}_{\mathrm{out}}(h_1)| > \epsilon \\ \text{or } & |\mathrm{E}_{\mathrm{in}}(h_2) - \mathrm{E}_{\mathrm{out}}(h_2)| > \epsilon \\ & \cdots \\ \text{or } & |\mathrm{E}_{\mathrm{in}}(h_M) - \mathrm{E}_{\mathrm{out}}(h_M)| > \epsilon \end{aligned}$$

- using the union bound

$$\begin{aligned}
\mathbb{P}[|\mathrm{E}_{\text{in}}(g) - \mathrm{E}_{\text{out}}(g)| > \epsilon] &\leq \mathbb{P}[\quad |\mathrm{E}_{\text{in}}(h_1) - \mathrm{E}_{\text{out}}(h_1)| > \epsilon \\
&\qquad \text{or } |\mathrm{E}_{\text{in}}(h_2) - \mathrm{E}_{\text{out}}(h_2)| > \epsilon \\
&\qquad \cdots \\
&\qquad \text{or } |\mathrm{E}_{\text{in}}(h_M) - \mathrm{E}_{\text{out}}(h_M)| > \epsilon] \\
&\leq \sum_{m=1}^{M} \mathbb{P}[|\mathrm{E}_{\text{in}}(h_m) - \mathrm{E}_{\text{out}}(h_m)| > \epsilon] \\
&\leq 2Me^{-2\epsilon^2 N}
\end{aligned}$$

- in the end we get a 'uniform' version of (2):

$$\mathbb{P}[|\mathrm{E}_{\text{in}}(g) - \mathrm{E}_{\text{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N} \tag{6}$$

Downside for uniform estimates

- probability bound $2Me^{-2\epsilon^2 N}$
  1. factor of M looser than the bound for a single hypothesis
  2. only meaningful if M is finite

- we will improve on these limitations in later lectures

Outline

Ultimate goal of learning

- learning produces a hypothesis g to approximate unknown f
  - if successful, then g should approximate f well
  - *i.e.* $E_{out}(g) \approx 0$ ⟵ the ultimate goal of learning
- however, this is not what we get from the probabilistic analysis
  - what we get instead: $\mathrm{E}_{\mathrm{out}}(g) \approx \mathrm{E}_{\mathrm{in}}(g)$
- to conclude that $\mathrm{E}_{\mathrm{out}}(g) \approx 0$ ⟵ what we cannot ascertain
  - we still have to make $E_{in}(g) \approx 0$ (확인가능) ⟵ what we can ascertain

  ⇒ we can use E_{in} as a proxy for $\mathrm{E}_{\mathrm{out}}$

(대리인)

: 내공에는 E_{test}를 대리인으로 사용할것.

The feasibility of learning

- is split into "two" questions:

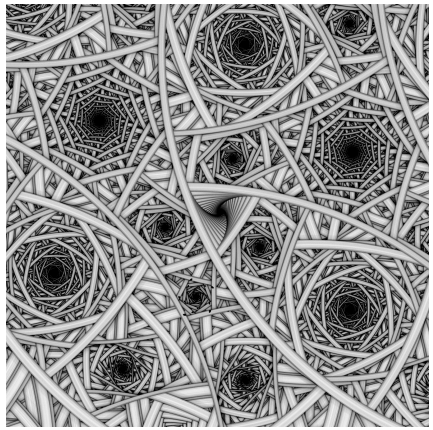
> 1. can we make sure that $E_{out}(g) \simeq E_{in}(g)$?
> 2. can we make $E_{in}(g)$ small enough?

- Hoeffding inequality: $\mathbb{P}[|\mathrm{E}_{\mathrm{in}}(g) - \mathrm{E}_{\mathrm{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$
  - addresses the first question only

- the second question is answered after we
  - run the learning algorithm on the actual data, and
  - see how small we can get E_{in} to be

Complexity of components of learning

- breaking down the feasibility of learning into these two questions
  - provides further insight into the role of different components
- one such insight is
  - about the "complexity" of these components
- we consider two types:
  - complexity of hypothesis set $\mathcal{H}$: M
  - complexity of target function f



Complexity of hypothesis set $\mathcal{H}$

- to measure the 'complexity' of $\mathcal{H}$
  - we use M: the number of hypotheses in $\mathcal{H}$
- for $\mathrm{E}_{\mathrm{out}}(g) \approx \mathrm{E}_{\mathrm{in}}(g)$: keep $\mathcal{H}$ less complex
  - if M goes up $\longrightarrow$ we run more risk that $\mathrm{E}_{\mathrm{in}}(g)$ will be a poor estimator of $\mathrm{E}_{\mathrm{out}}(g)$ [according to (6)]
- for $\mathrm{E}_{\mathrm{in}}(g) \approx 0$: better if $\mathcal{H}$ is more complex
  - g has to come from $\mathcal{H}$ $\longrightarrow$ more complex $\mathcal{H}$ gives more flexibility in finding g that fits data well for small $\mathrm{E}_{\mathrm{in}}(g)$
- this trade-off in the complexity of $\mathcal{H}$
  - a major theme in learning theory

Complexity of target function f

- recall

> 1. can we make sure that $E_{out}(g) \simeq E_{in}(g)$?
> 2. can we make $E_{in}(g)$ small enough?

- intuitively, a complex f should be harder to learn than a simple f
  - let's see if this can be inferred from the two questions above

- Hoeffding inequality: $\mathbb{P}[|\mathrm{E}_{\mathrm{in}}(g) - \mathrm{E}_{\mathrm{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$
  - independent of complexity of f
  - but this doesn't mean: we can learn complex f as easily as simple f
  - this inequality affects the first question only

- if f gets more complex $\longrightarrow$ the second question comes into play
  - data from more complex f: harder to fit

  $\Rightarrow$ higher $\mathrm{E}_{\mathrm{in}}(g)$

- we might try more complex $\mathcal{H}$
  - we can fit the data better $\Rightarrow$ lower $\mathrm{E}_{\mathrm{in}}(g)$
  - but then $\mathrm{E}_{\mathrm{out}}(g)$ will not be as close to $\mathrm{E}_{\mathrm{in}}(g)$ due to (6)

- simply put
  - a complex f is harder to learn as we expected

Preview: what the theory will allow us to understand

- feasibility of learning for infinite M

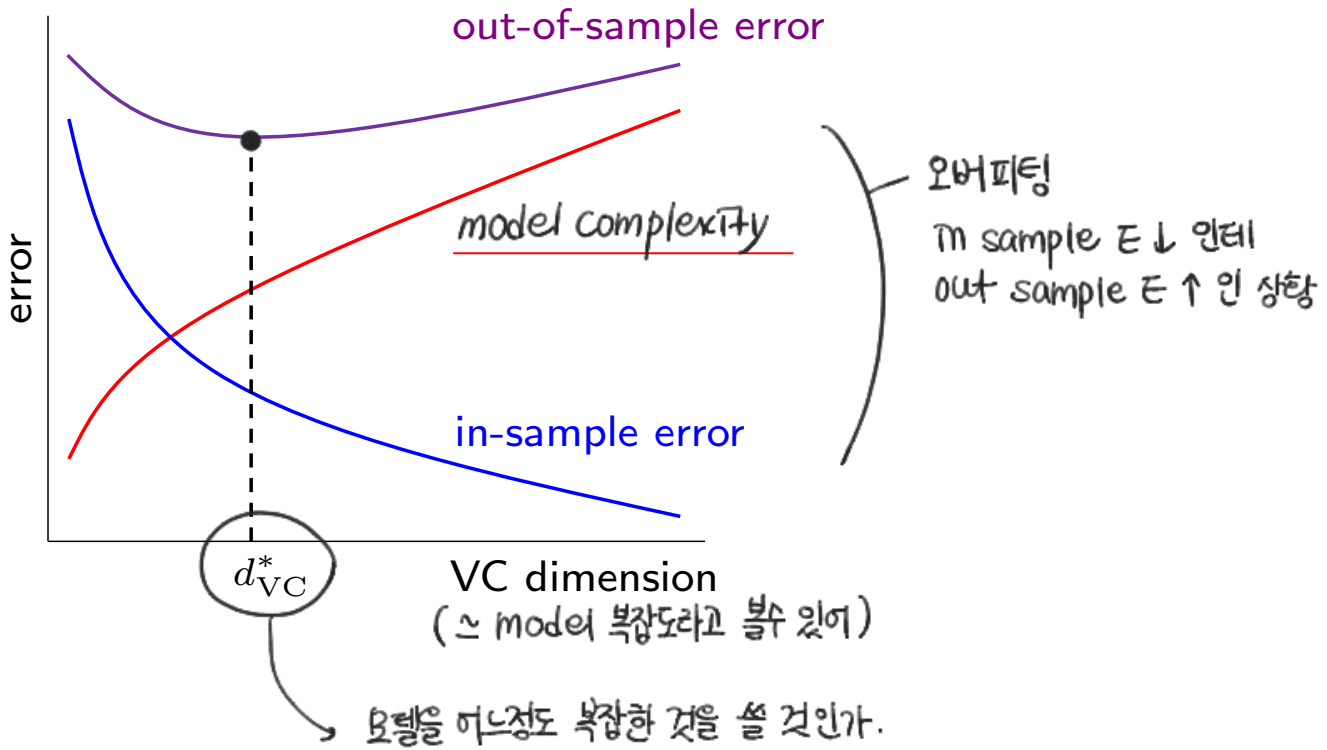
  무한개 원소 갖는 집합

  (그냥 휘프팅 쓰면 러닝 불가능)

- tradeoff:

$M \uparrow$

model complexity	$\uparrow$	$\mathrm{E_{in}}$	$\downarrow$
model complexity	$\uparrow$	$\mathrm{E_{out}} - \mathrm{E_{in}}$	$\uparrow$

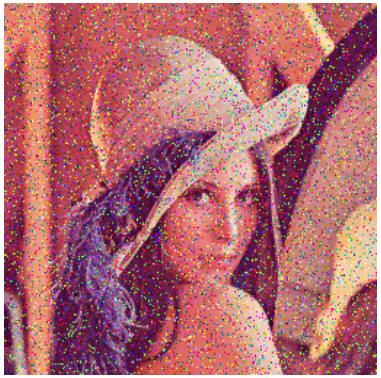


Outline

Error and noise

- error (aka cost, objective, risk):
  - about what "$h \simeq f$" means ($g \simeq f$)
  - quantifies how far we are from the target
  - the choice of an error measure affects the outcome of learning

- noise: (error와 다르다)
  - about the nature of the target function
  - often noise makes the output of f not uniquely determined by the input
    → fix fin이 아닌 conditional probability.



noisy Lena

Pointwise error

- in principle: an error measure quantifies
  - how well each hypothesis h approximates f

$$\text{Error} = \text{E}(h, f) \tag{7}$$

  - this is based on the entire h and f

전체에 대한 에러를 한번에 정의하기 힘들어

- in practice: an error measure is almost always defined

∴ 한 점씩 에러를 정의

  - pointwise on individual input points:

$$\text{e}(h(\mathbf{x}), f(\mathbf{x}))$$

  - examples:

squared error $\text{e}(h(\mathbf{x}), f(\mathbf{x})) = (h(\mathbf{x}) - f(\mathbf{x}))^2$

binary error $\text{e}(h(\mathbf{x}), f(\mathbf{x})) = [\![h(\mathbf{x}) \neq f(\mathbf{x})]\!]$

From pointwise to overall

- overall error $\mathrm{E}(h, f) =$ average of pointwise errors $\mathsf{e}\left(h(\mathbf{x}), f(\mathbf{x})\right)$
- in-sample error:

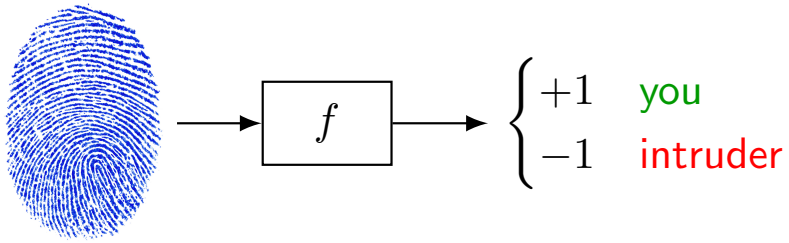
$$\mathrm{E}_{\mathrm{in}}(h) = \frac{1}{N} \sum_{n=1}^{N} \mathsf{e}(h(\mathbf{x}_n), f(\mathbf{x}_n)) \tag{8}$$

- out-of-sample error:

$$\mathrm{E}_{\mathrm{out}}(h) = \underbrace{\mathbb{E}_{\mathbf{x}}}_{\text{expectation}}[\mathsf{e}(h(\mathbf{x}), f(\mathbf{x}))] \tag{9}$$

How to choose the error measure

- ex) fingerprint verification
- two types of error
  1. false accept
  2. false reject
- how to penalize each type?



		f (target)	
		$+1$	-1
h (hypothesis)	$+1$	no error	false accept
	-1	false reject	no error

For supermarkets

- supermarket verifies fingerprint for discounts
- false reject is costly
  - customer gets annoyed!
- false accept is minor
  - gave away a discount and
  - intruder left their fingerprint

		f (target)	
		$+1$	-1
h (hypothesis)	$+1$	0	1 (false accept)
	-1	10 (false reject)	0

For the CIA

- CIA verifies fingerprint for security
- false accept is a disaster!
- false reject can be tolerated
  - try again; you are an employee

		f (target)	
		$+1$	-1
h (hypothesis)	$+1$	0	1000 (false accept)
	-1	1 (false reject)	0

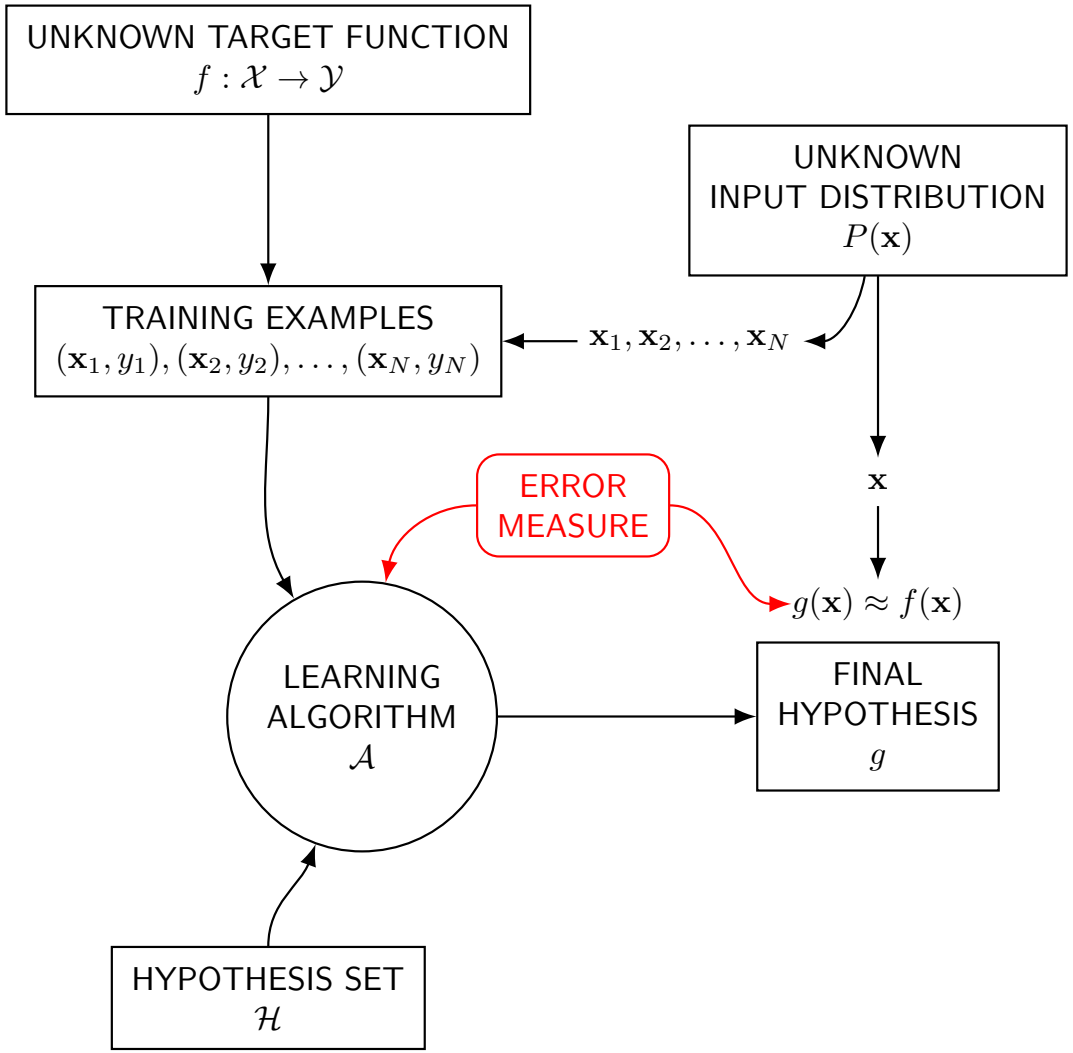
Take-home message

- the error measure should be specified by the user (문제에 따라 다르다)
  - but this is not always possible

1. the user may not provide an error specification
2. the weighted cost may be a difficult objective function for optimizers to work with

- alternatives
  - *plausible* measures: squared error, Gaussian noise (사람들이 흔히 사용하는것)
  - friendly measures: closed-form solution, convex optimization

The learning diagram: with error measures



Outline

Noisy targets

- the 'target function' is not always a *function*
- consider the credit-card approval:

feature	value
age	23 years
gender	female
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...

- two 'identical' customers but two different behaviors

- usually, the data we learn from
  - not generated by a "deterministic" target function
    하게 하지 x : 확률로 생각하자.
  - instead, generated in a noisy way such that the output is not uniquely determined by the input

- ex) the credit 'function' distribution
  - is not really a deterministic function but a noisy one
  - but still can be modeled within the framework we have

Interpolation versus regression

- given training data $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$

- interpolation
  (not learning)
  - $y_n = f(\mathbf{x}_n)$
  - find a deterministic function $f(\mathbf{x}) \in \mathbb{R}$ such that $y_n = f(\mathbf{x}_n)$

- regression:
  noise ϵ is added
  - $y_n = f(\mathbf{x}_n) + \epsilon$ (같은 $\mathbf{x}$에 대해 다른 y가 나올수 있다)
  - find a model $g(\mathbf{x})$ that approximates y_n

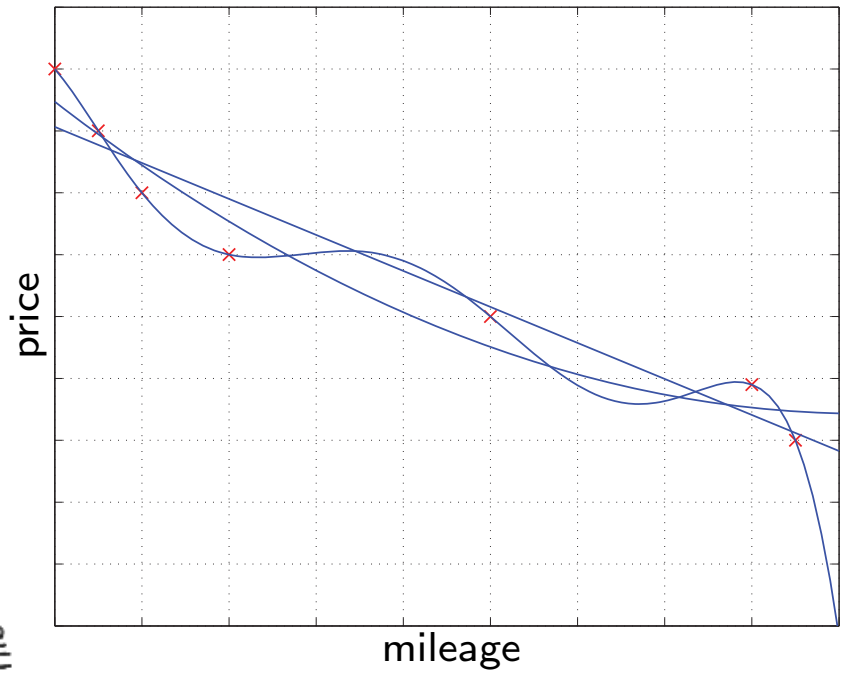


Figure 2: linear, second-order, and sixth-order polynomial fits ($N = 7$)

Target 'distribution'

- instead of $y = f(\mathbf{x})$, we can take output y to be a random variable
  - that is affected by, rather than determined by, the input $\mathbf{x}$
- formally,
  - we have a target *distribution* $y \sim P(y|\mathbf{x})$
  - instead of a target function $y = f(\mathbf{x})$

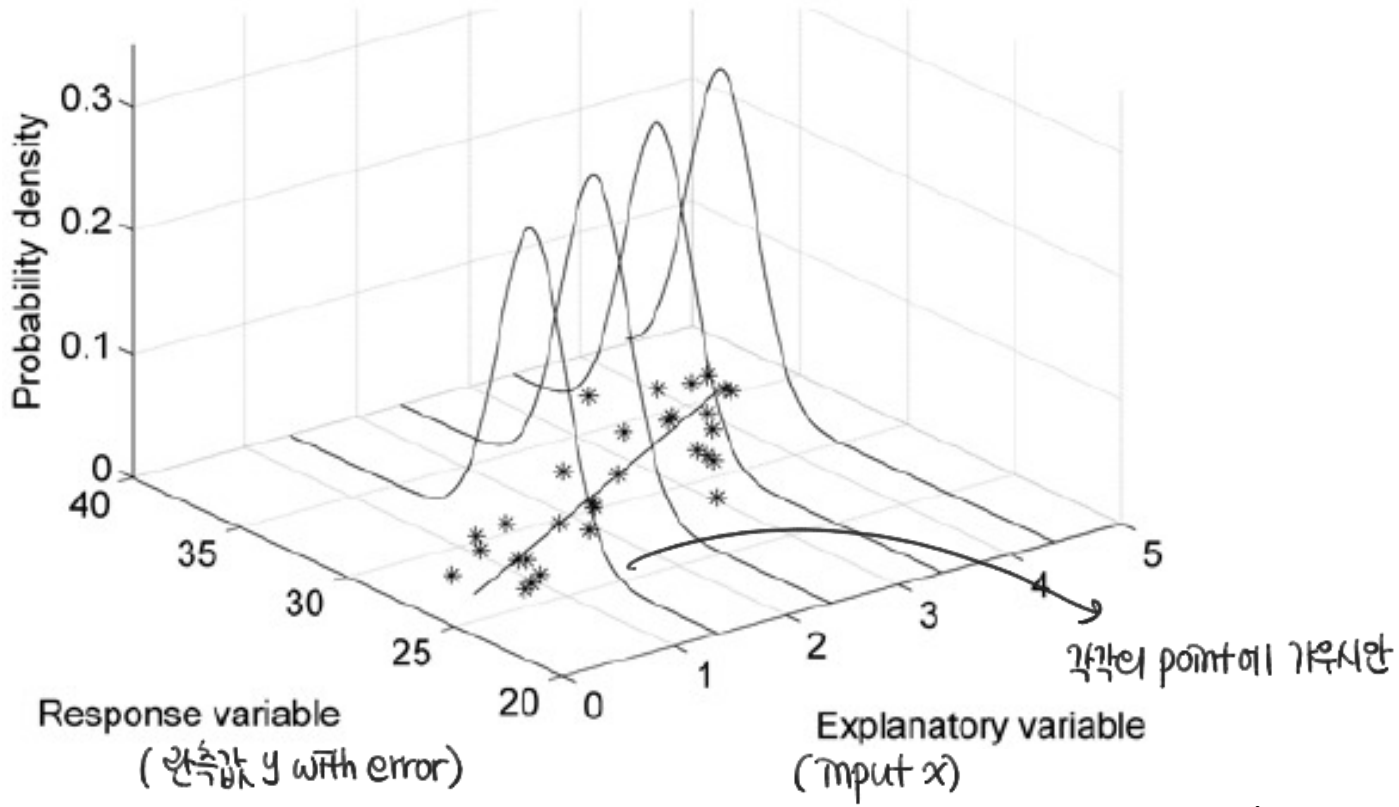


Figure 3: A schematic diagram of a linear regression analysis, where data (n = 30, indicated by *) were generated from the linear function with the slope = 2.5, intercept = 25.0, and standard deviation = 1.5. The estimated parameters using a linear regression were slope = 2.45, intercept = 25.3, and standard deviation = 1.48. The estimated regression line is shown as a straight line, whereas four normal distributions indicate the assumed normal distributions of data at four values of the explanatory variable (1.4, 2.4, 3.4, and 4.4).

source: http://www.seaturtle.org/mtn/archives/mtn122/mtn122p1.shtml

We can think of

- noisy target as deterministic target plus added noise:
  - noisy target $y = \underbrace{f(\mathbf{x})}_{\text{deterministic target } = \mathbb{E}(y|\mathbf{x})} + \underbrace{y - f(\mathbf{x})}_{\text{pure noise}}$
- deterministic target as the average of noisy target:
  - $f(\mathbf{x}) = \mathbb{E}(y|\mathbf{x})$
- deterministic target as a special case of noisy target:
  - $P(y|\mathbf{x})$ is zero except for $y = f(\mathbf{x})$

The learning diagram: including noisy target

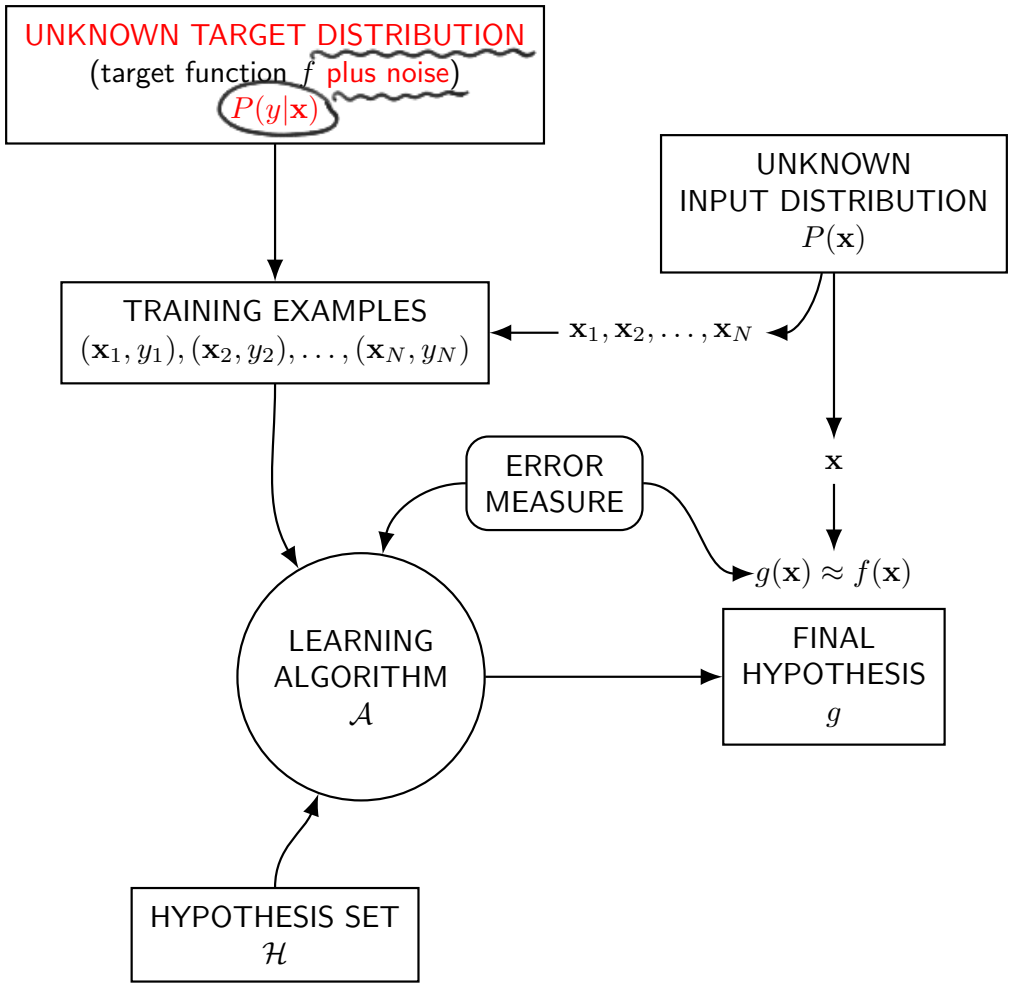


Figure 4: the general (supervised) learning problem

Distinction between $P(y|\mathbf{x})$ and $P(\mathbf{x})$

- both convey probabilistic aspects of $\mathbf{x}$ and y
  - but different roles of $P(y|\mathbf{x})$ and $P(\mathbf{x})$ in learning

- $P(y|\mathbf{x})$, the target distribution
  - what we are trying to learn

- $P(\mathbf{x})$, the input distribution
  - only quantifies the relative importance of the point $\mathbf{x}$

- merging $P(\mathbf{x})P(y|\mathbf{x})$ as $P(\mathbf{x}, y)$ mixes the two concepts
  - sample $(\mathbf{x}, y)$ is now generated by the joint distribution:

$$P(\mathbf{x})P(y|\mathbf{x})$$

Outline

Summary

- learning *unknown* target f: only possible in a probabilistic sense
  - we need hypothesis $g \approx f$, which means $\mathrm{E}_{\mathrm{out}}(g) \approx 0$
- feasibility of learning is split into two questions
  1. can we make sure that $\mathrm{E}_{\mathrm{out}}(g) \approx \mathrm{E}_{\mathrm{in}}(g)$?
  2. can we make $\mathrm{E}_{\mathrm{in}}(g) \approx 0$?
- answering Q1: theoretical (first half of this course)
  - Hoeffding's inequality: $\mathbb{P}[|\mathrm{E}_{\mathrm{in}}(g) - \mathrm{E}_{\mathrm{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$
  - VC analysis and bias-variance analysis
- answering Q2: practical (second half)
  - various learning paradigms will help us achieve that

- error measure quantifies the meaning of $\approx$ in expression $g \approx f$
  - ideally, should be specified by the user
  - practical alternatives: *plausible/friendly* measures

- averaging *pointwise* error gives
  - in-sample: $\mathrm{E}_{\mathrm{in}}(h) = \frac{1}{N} \sum_n \mathrm{e}(h(\mathbf{x}_n), f(\mathbf{x}_n))$
  - out-of-sample: $\mathrm{E}_{\mathrm{out}}(h) = \mathbb{E}_{\mathbf{x}}[\mathrm{e}(h(\mathbf{x}), f(\mathbf{x}))]$

- noisy targets
  - $y \sim P(y|\mathbf{x})$ instead of $y = f(\mathbf{x})$ [y is a random variable]
  - $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ generated by $P(\mathbf{x}, y) = P(\mathbf{x})P(y|\mathbf{x})$
  - $\mathrm{E}_{\mathrm{out}}(h) = \mathbb{E}_{\mathbf{x},y}[\mathrm{e}(h(\mathbf{x}), y)]$