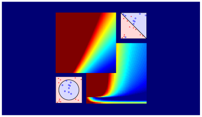
Machine Learning Foundations
(機器學習基石)



Lecture 8: Noise and Error

Hsuan-Tien Lin (林軒田)

htlin@csie.ntu.edu.tw

Department of Computer Science
& Information Engineering

National Taiwan University
(國立台灣大學資訊工程系)

Roadmap

1. When Can Machines Learn?
2. Why Can Machines Learn?

Lecture 7: The VC Dimension

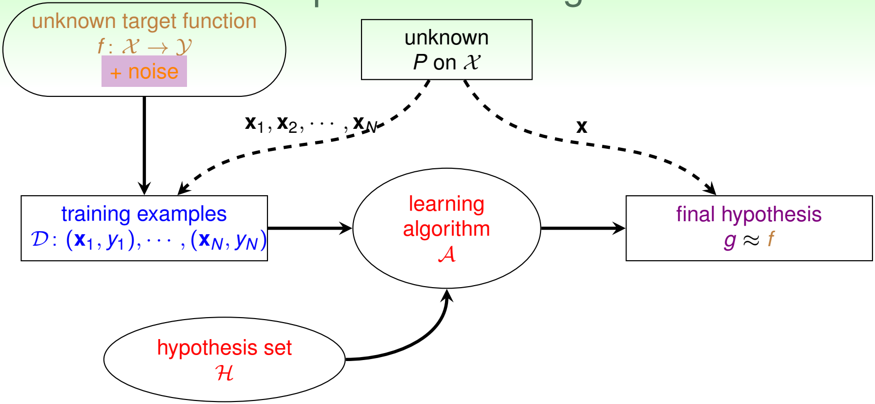
learning happens
if finite d_{vc}, large N, and low E_{in}

Lecture 8: Noise and Error

- Noise and Probabilistic Target
- Error Measure
- Algorithmic Error Measure

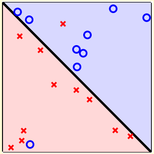
3. How Can Machines Learn?
4. How Can Machines Learn Better?

Recap: The Learning Flow



what if there is noise?

Noise



briefly introduced noise before pocket algorithm

age	23 years
gender	female
annual salary	NTD 1,000,000
year in residence	1 year
year in job	0.5 year
current debt	200,000

credit? $\{\text{no}(-1), \text{yes}(+1)\}$

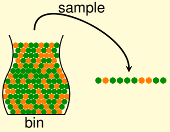
but more!

- noise in y: good customer, 'mislabeled' as bad?
- noise in y: same customers, different labels?
- noise in $\mathbf{x}$: inaccurate customer information?

does VC bound work under noise?

Probabilistic Marbles

one key of VC bound: marbles!



'deterministic' marbles

- marble $\mathbf{x} \sim P(\mathbf{x})$
- deterministic color $[\![f(\mathbf{x}) \neq h(\mathbf{x})]\!]$

'probabilistic' (noisy) marbles

- marble $\mathbf{x} \sim P(\mathbf{x})$
- probabilistic color $[\![y \neq h(\mathbf{x})]\!]$ with $y \sim P(y|\mathbf{x})$

same nature: can estimate $\mathbb{P}[\text{orange}]$ if $\overset{i.i.d.}{\sim}$

VC holds for $\underbrace{\mathbf{x} \overset{i.i.d.}{\sim} P(\mathbf{x}), y \overset{i.i.d.}{\sim} P(y|\mathbf{x})}_{(\mathbf{x},y) \overset{i.i.d.}{\sim} P(\mathbf{x},y)}$

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

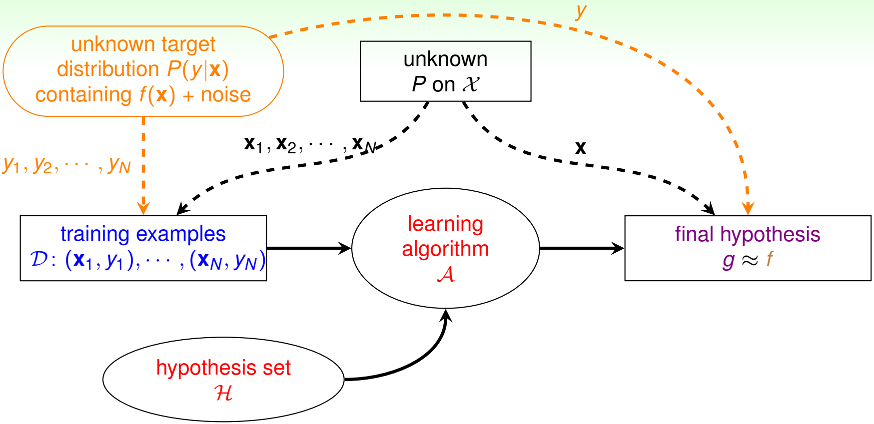
characterizes behavior of 'mini-target' on one $\mathbf{x}$

- can be viewed as 'ideal mini-target' + noise, e.g.
  - $P(\circ|\mathbf{x}) = 0.7$, $P(\times|\mathbf{x}) = 0.3$
  - ideal mini-target $f(\mathbf{x}) = \circ$
  - 'flipping' noise level = 0.3
- deterministic target f: special case of target distribution
  - $P(y|\mathbf{x}) = 1$ for $y = f(\mathbf{x})$
  - $P(y|\mathbf{x}) = 0$ for $y \neq f(\mathbf{x})$

goal of learning:

predict ideal mini-target (w.r.t. $P(y|\mathbf{x})$)
on often-seen inputs (w.r.t. $P(\mathbf{x})$)

The New Learning Flow



VC still works, pocket algorithm explained :-)

Fun Time

Let's revisit PLA/pocket. Which of the following claim is true?

1. In practice, we should try to compute if $\mathcal{D}$ is linear separable before deciding to use PLA.
2. If we know that $\mathcal{D}$ is not linear separable, then the target function f must not be a linear function.
3. If we know that $\mathcal{D}$ is linear separable, then the target function f must be a linear function.
4. None of the above

Reference Answer: (4)

(1) After computing if $\mathcal{D}$ is linear separable, we shall know $\mathbf{w}^*$ and then there is no need to use PLA. (2) What about noise? (3) What about 'sampling luck'? :-)

Error Measure

final hypothesis
$g \approx f$

- how well? previously, considered out-of-sample measure

$$E_{\text{out}}(g) = \mathop{\mathbb{E}}_{\mathbf{x} \sim P} [\![g(\mathbf{x}) \neq f(\mathbf{x})]\!]$$

- more generally, error measure $E(g, f)$
- naturally considered
  - out-of-sample: averaged over unknown $\mathbf{x}$
  - pointwise: evaluated on one $\mathbf{x}$
  - classification: $[\![$prediction $\neq$ target$]\!]$

classification error $[\![\ldots]\!]$:
often also called '0/1 error'

Pointwise Error Measure

can often express $E(g, f)$ = averaged $\text{err}(g(\mathbf{x}), f(\mathbf{x}))$, like

$$E_{\text{out}}(g) = \mathop{\mathbb{E}}_{\mathbf{x} \sim P} \underbrace{[\![g(\mathbf{x}) \neq f(\mathbf{x})]\!]}_{\text{err}(g(\mathbf{x}), f(\mathbf{x}))}$$

—err: called pointwise error measure

in-sample

$$E_{\text{in}}(g) = \frac{1}{N} \sum_{n=1}^{N} \text{err}(g(\mathbf{x}_n), f(\mathbf{x}_n))$$

out-of-sample

$$E_{\text{out}}(g) = \mathop{\mathbb{E}}_{\mathbf{x} \sim P} \text{err}(g(\mathbf{x}), f(\mathbf{x}))$$

will mainly consider pointwise err for simplicity

Two Important Pointwise Error Measures

$$\text{err}\left(\underbrace{g(\mathbf{x})}_{\tilde{y}}, \underbrace{f(\mathbf{x})}_{y}\right)$$

0/1 error

$$\text{err}(\tilde{y}, y) = [\![\tilde{y} \neq y]\!]$$

- correct or incorrect?
- often for classification

squared error

$$\text{err}(\tilde{y}, y) = (\tilde{y} - y)^2$$

- how far is $\tilde{y}$ from y?
- often for regression

how does err 'guide' learning?

Ideal Mini-Target

interplay between noise and error:

$P(y|\mathbf{x})$ and err define ideal mini-target $f(\mathbf{x})$

$$P(y=1|\mathbf{x}) = 0.2, P(y=2|\mathbf{x}) = 0.7, P(y=3|\mathbf{x}) = 0.1$$

$$\mathrm{err}(\tilde{y}, y) = [\![\tilde{y} \neq y]\!]$$

$$\tilde{y} = \begin{cases} 1 & \text{avg. } \mathrm{err}\ 0.8 \\ 2 & \text{avg. } \mathrm{err}\ 0.3 (*) \\ 3 & \text{avg. } \mathrm{err}\ 0.9 \\ 1.9 & \text{avg. } \mathrm{err}\ 1.0 \text{(really? :-))} \end{cases}$$

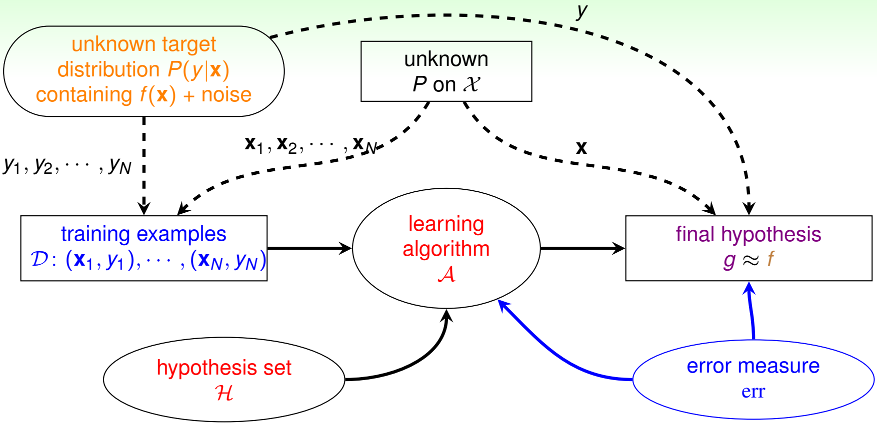
$$f(\mathbf{x}) = \underset{y \in \mathcal{Y}}{\mathrm{argmax}}\ P(y|\mathbf{x})$$

$$\mathrm{err}(\tilde{y}, y) = (\tilde{y} - y)^2$$

$$\begin{cases} 1 & \text{avg. } \mathrm{err}\ 1.1 \\ 2 & \text{avg. } \mathrm{err}\ 0.3 \\ 3 & \text{avg. } \mathrm{err}\ 1.5 \\ 1.9 & \text{avg. } \mathrm{err}\ 0.29 (*) \end{cases}$$

$$f(\mathbf{x}) = \sum_{y \in \mathcal{Y}} y \cdot P(y|\mathbf{x})$$

Learning Flow with Error Measure



extended VC theory/'philosophy'
works for most $\mathcal{H}$ and err

Fun Time

Consider the following $P(y|\mathbf{x})$ and $\text{err}(\tilde{y}, y) = |\tilde{y} - y|$. Which of the following is the ideal mini-target $f(\mathbf{x})$?

$$P(y = 1|\mathbf{x}) = 0.10, P(y = 2|\mathbf{x}) = 0.35,$$
$$P(y = 3|\mathbf{x}) = 0.15, P(y = 4|\mathbf{x}) = 0.40.$$

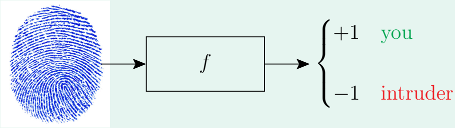
1. $2.5 =$ average within $\mathcal{Y} = \{1, 2, 3, 4\}$
2. $2.85 =$ weighted mean from $P(y|\mathbf{x})$
3. $3 =$ weighted median from $P(y|\mathbf{x})$
4. $4 = \text{argmax}\, P(y|\mathbf{x})$

Reference Answer: (3)

For the 'absolute error', the weighted median provably results in the minimum average err.

Choice of Error Measure

Fingerprint Verification



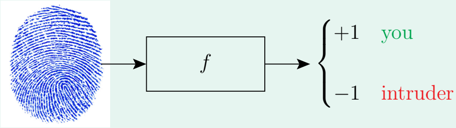
two types of error: false accept and false reject

		g	
		+1	-1
f	+1	no error	false reject
	-1	false accept	no error

0/1 error penalizes both types equally

Fingerprint Verification for Supermarket

Fingerprint Verification



two types of error: false accept and false reject

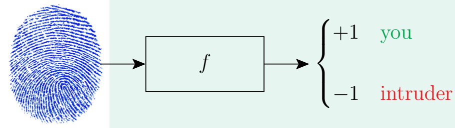
		g	
		+1	-1
f	+1	no error	false reject
	-1	false accept	no error

		g	
		+1	-1
f	+1	0	10
	-1	1	0

- supermarket: fingerprint for discount
- false reject: very unhappy customer, lose future business
- false accept: give away a minor discount, intruder left fingerprint :-)

Fingerprint Verification for CIA

Fingerprint Verification



two types of error: false accept and false reject

		g: +1	g: -1
f	+1	no error	false reject
	-1	false accept	no error

		g: +1	g: -1
f	+1	0	1
	-1	1000	0

- CIA: fingerprint for entrance
- false accept: very serious consequences!
- false reject: unhappy employee, but so what? :-)

Take-home Message for Now

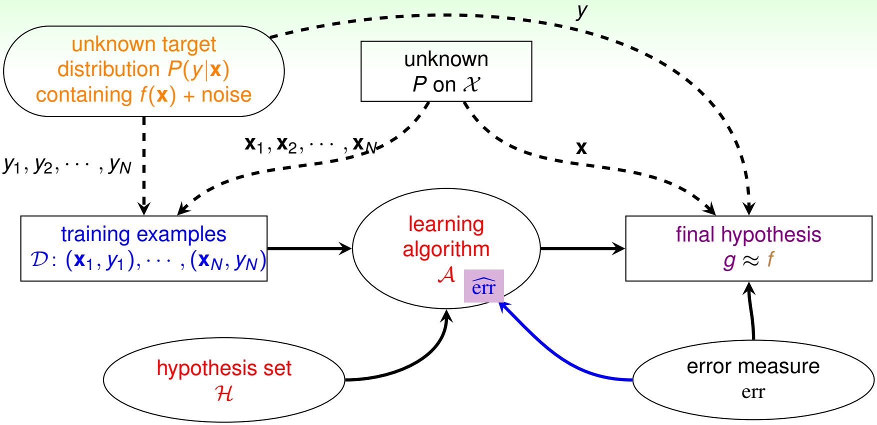
err is application/user-dependent

Algorithmic Error Measures $\widehat{\text{err}}$

- true: just err
- plausible:
  - 0/1: minimum 'flipping noise'—NP-hard to optimize, remember? :-)
  - squared: minimum Gaussian noise
- friendly: easy to optimize for $\mathcal{A}$
  - closed-form solution
  - convex objective function

$\widehat{\text{err}}$: more in next lectures

Learning Flow with Algorithmic Error Measure



err: application goal;
$\widehat{\text{err}}$: a key part of many $\mathcal{A}$

Fun Time

Consider err below for CIA. What is $E_{in}(g)$ when using this err?

		g: +1	g: -1
f	+1	0	1
	-1	1000	0

1. $\frac{1}{N}\sum_{n=1}^{N} [\![y_n \neq g(\mathbf{x}_n)]\!]$
2. $\frac{1}{N}\left(\sum_{y_n=+1} [\![y_n \neq g(\mathbf{x}_n)]\!] + 1000 \sum_{y_n=-1} [\![y_n \neq g(\mathbf{x}_n)]\!]\right)$
3. $\frac{1}{N}\left(\sum_{y_n=+1} [\![y_n \neq g(\mathbf{x}_n)]\!] - 1000 \sum_{y_n=-1} [\![y_n \neq g(\mathbf{x}_n)]\!]\right)$
4. $\frac{1}{N}\left(1000 \sum_{y_n=+1} [\![y_n \neq g(\mathbf{x}_n)]\!] + \sum_{y_n=-1} [\![y_n \neq g(\mathbf{x}_n)]\!]\right)$

Reference Answer: 2

When $y_n = -1$, the false positive made on such $(\mathbf{x}_n, y_n)$ is penalized 1000 times more!

Summary

1. When Can Machines Learn?
2. Why Can Machines Learn?

Lecture 7: The VC Dimension

Lecture 8: Noise and Error

- Noise and Probabilistic Target

  can replace $f(\mathbf{x})$ by $P(y|\mathbf{x})$
- Error Measure

  affect 'ideal' target
- Algorithmic Error Measure

  user-dependent $\Longrightarrow$ plausible or friendly

- next: more algorithms, please? :-)

3. How Can Machines Learn?
4. How Can Machines Learn Better?