MONASH University

ETC3555

Statistical Machine Learning

The learning problem

1 August 2018

Outline

Verification vs learning

Are we done?

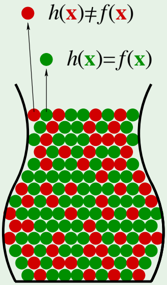
Not so fast! h is fixed.

For <u>this</u> h, ν generalizes to μ.

'**verification**' of h, not **learning**

No guarantee ν will be small.

We need to **choose** from multiple h's.



© Creator: Yaser Abu-Mostafa - LFD Lecture 2

Notation for learning

Notation for learning

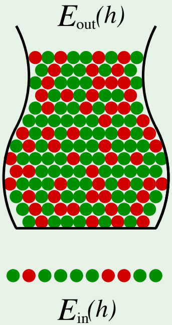
Both μ and ν depend on which hypothesis h

ν is 'in sample' denoted by $E_{\text{in}}(h)$

μ is 'out of sample' denoted by $E_{\text{out}}(h)$

The Hoeffding inequality becomes:

$$\mathbb{P}\left[\ |E_{\text{in}}(h) - E_{\text{out}}(h)| > \epsilon\ \right] \ \leq\ 2e^{-2\epsilon^2 N}$$

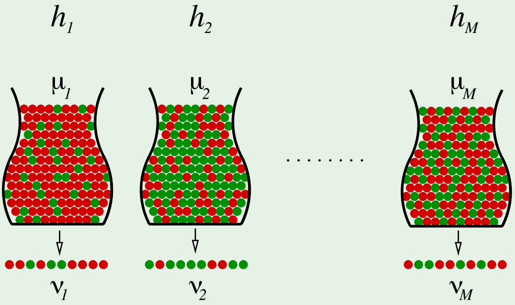


© Creator: Yaser Abu-Mostafa - LFD Lecture 2

Multiple bins

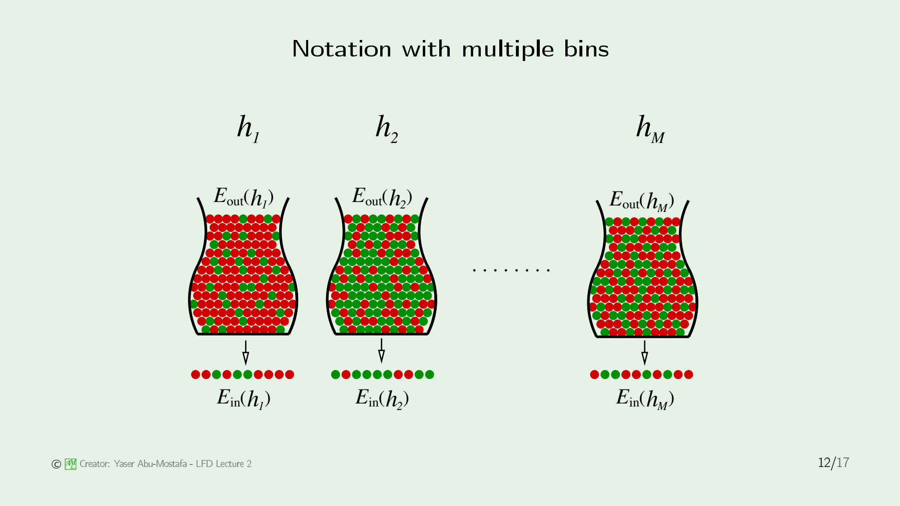
Multiple bins

Generalizing the bin model to more than one hypothesis:



© Creator: Yaser Abu-Mostafa - LFD Lecture 2

Notation with multiple bins



Coin analogy

Hoefdding does not apply to multiple bins!

Coin analogy

Question: If you toss a fair coin 10 times, what is the probability that you will get 10 heads?

Answer: $\approx 0.1\%$

Question: If you toss 1000 fair coins 10 times each, what is the probability that some coin will get 10 heads?

Answer: $\approx 63\%$

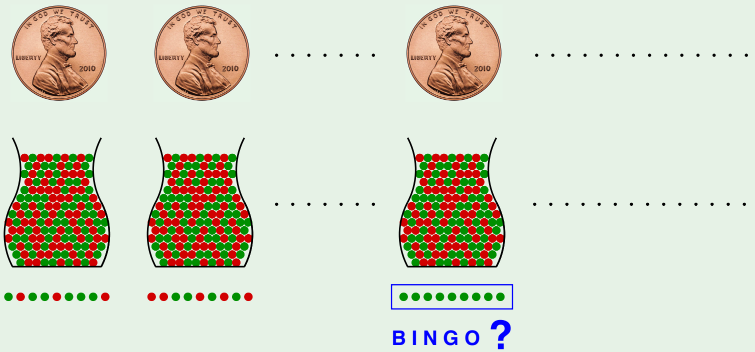
© Creator: Yaser Abu-Mostafa - LFD Lecture 2

Coin analogy

1. $\frac{1}{2} \times \frac{1}{2} \times \cdots \times \frac{1}{2}$ (10 times) is $\approx \frac{1}{1000} = 0.1\%$.
2. The probability of not getting 10 heads for one coin is $\approx (1 - \frac{1}{1000})$.
3. The probability of not getting 10 heads for any of 1000 coins is $\approx (1 - \frac{1}{1000})^{1000}$.
4. $\lim_{n \to \infty}(1 - \frac{1}{n})^n = \frac{1}{e}$.
5. $\frac{1}{e} \approx \frac{1}{2.718} \approx 0.37$
6. the probability of this not happening, i.e. at least one coin of the 1000 coins will give 10 heads, is 1 minus $0.37 = 0.63$

From coins to learning

From coins to learning



© Creator: Yaser Abu-Mostafa - LFD Lecture 2

From coins to learning

The Hoeffding inequality applies to each bin individually. The inequality states that

$$\mathbb{P}[|E_{\text{in}}(h) - E_{\text{out}}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N} \quad \text{for any } \epsilon > 0$$

where

1. the hypothesis h is fixed before the data is generated,
2. the probability is with respect to random data sets $\mathcal{D}$.

The assumption "h is fixed before the data set is generated" is critical to the validity of the bound.

From coins to learning

In learning, we consider an entire hypothesis set, say $\mathcal{H} = \{h_1, h_2, \ldots, h_M\}$ (with a finite number of hypotheses), instead of just one hypothesis h. Then, the learning algorithm picks the final hypothesis g based on $\mathcal{D}$.

The statement we would like to make is **not**

$$\mathbb{P}[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon] \text{ is small for any fixed } h_m \in \mathcal{H},$$

where $m = 1, 2, \ldots, M$, but **rather**

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \text{ is small for the final hypothesis } g.$$

A simple solution

$$|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon$$

$$\Longrightarrow$$

$$\begin{aligned} & |E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| > \epsilon \\ \text{or } & |E_{\text{in}}(h_2) - E_{\text{out}}(h_2)| > \epsilon \\ \dots & \\ \text{or } & |E_{\text{in}}(h_M) - E_{\text{out}}(h_M)| > \epsilon \end{aligned}$$

A simple solution

A simple solution

$$\mathbb{P}[\ |E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\ ] \ \leq\ \mathbb{P}[\quad |E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| > \epsilon$$
$$\textbf{or}\ |E_{\text{in}}(h_2) - E_{\text{out}}(h_2)| > \epsilon$$
$$\cdots$$
$$\textbf{or}\ |E_{\text{in}}(h_M) - E_{\text{out}}(h_M)| > \epsilon\ ]$$
$$\leq \sum_{m=1}^{M} \mathbb{P}\left[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon\right]$$

© Creator: Yaser Abu-Mostafa - LFD Lecture 2

16/17

The final verdict

The final verdict

$$\mathbb{P}[\ |E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\ ] \ \leq \ \sum_{m=1}^{M} \mathbb{P}\left[|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)| > \epsilon\right]$$

$$\leq \ \sum_{m=1}^{M} 2e^{-2\epsilon^2 N}$$

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2{\color{red}M}e^{-2\epsilon^2 N}$$

© Creator: Yaser Abu-Mostafa - LFD Lecture 2

17/17

Generalization error

- The out-of-sample error E_{out} measures how well our training on $\mathcal{D}$ has generalized to unseen data points. E_{out} is based on the performance over the entire input space $\mathcal{X}$.
- The in-sample error E_{in} is based on the training data points.
- The *generalization error* is the discrepancy between E_{in} and E_{out}. Generalization error is also used as another name for E_{out} (but not in this unit).
- The Hoeffding inequality provides a way to charaterize the generalization error with a probabilistic bound.

Generalization bound

The Hoeffding inequality states that

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N} \quad \text{for any } \ \epsilon > 0$$

This can be rephrased as follows. Pick a tolerance level δ, for example $\delta = 0.01$, and assert with probability at least $1 - \delta$ that

$$E_{\text{out}}(g) \leq E_{\text{in}}(g) + \sqrt{\frac{1}{2N}\, ln\left(\frac{2M}{\delta}\right)}.$$

This is called a *generalization bound* since it bounds E_{out} in terms of E_{in}

Generalization bound

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$$
$$\implies 1 - \mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| \leq \epsilon] \leq 2Me^{-2\epsilon^2 N}$$
$$\implies \mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| \leq \epsilon] \geq 1 - 2Me^{-2\epsilon^2 N}$$

- With probability at least $1 - 2Me^{-2\epsilon^2 N}$, $|E_{\text{in}}(g) - E_{\text{out}}(g)| \leq \epsilon$, which implies $E_{\text{out}}(g) \leq E_{\text{in}}(g) + \epsilon$
- If $\delta = 2Me^{-2\epsilon^2 N}$, then $\epsilon = \sqrt{\frac{1}{2N}\, ln\left(\frac{2M}{\delta}\right)}$

Note

- The error bound depends on *M*, the size of the hypothesis set $\mathcal{H}$
- If $\mathcal{H}$ is an infinite set, the bound goes to infinity and becomes useless
- *M* can be replaced with something finite (the effective number of hypotheses), so that the bound is meaningful.

$$\mathbb{P}[\ |E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\ ] \ \leq\ 4\ m_{\mathcal{H}}(2N)\ e^{-\frac{1}{8}\epsilon^2 N}$$

The Vapnik-Chervonenkis Inequality

We will not cover the VC Inequality.

Outline

Feasibility of learning

- If we insist on a deterministic answer, i.e. $\mathcal{D}$ tells us something certain about f outside of $\mathcal{D}$, then the answer is no.
- If we accept a probabilistic answer, i.e. $\mathcal{D}$ tells us something likely about f outside of $\mathcal{D}$, then the answer is yes.

Feasibility of learning

What we know so far

Learning is feasible. It is likely that

$$E_{\text{out}}(g) \approx E_{\text{in}}(g)$$

Is this learning?

We need $g \approx f$, which means

$$E_{\text{out}}(g) \approx 0$$

© Creator: Yaser Abu-Mostafa - LFD Lecture 4

Feasibility of learning

The 2 questions of learning

$E_{\text{out}}(g) \approx 0$ is achieved through:

$$\underbrace{E_{\text{out}}(g) \approx E_{\text{in}}(g)}_{\text{Lecture 2}} \quad \text{and} \quad \underbrace{E_{\text{in}}(g) \approx 0}_{\text{Lecture 3}}$$

Learning is thus split into 2 questions:

1. Can we make sure that $E_{\text{out}}(g)$ is close enough to $E_{\text{in}}(g)$?
2. Can we make $E_{\text{in}}(g)$ small enough?

© Creator: Yaser Abu-Mostafa - LFD Lecture 4

- The Hoeffding Inequality addresses question 1 only.
- We answer the second question after running the learning algorithm on the the training data.

Feasibility of learning

The complexity of $\mathcal{H}$.

Question 1: According to the Hoeffding Inequality, a larger M increases the risk that $E_{\text{in}}(g)$ will be a poor estimate of $E_{\text{out}}(g)$ $\implies$ we need to control M (a measure of the complexity of $\mathcal{H}$).

Question 2: We stand a better chance if $\mathcal{H}$ is more complex $\implies$ a more complex $\mathcal{H}$ gives us more flexibility in finding some g that fits the data well.

The complexity of f.

Question 1: If we fix the hypothesis set and the number of training examples, the inequality provides the same bound $\implies$ The complexity of f does not affect how well $E_{\text{in}}(g)$ approximates $E_{\text{out}}(g)$.

Question 2: The data from a complex f are harder to fit than the data from a simple f (large $E_{\text{in}}(g)$). We can increase the complexity of $\mathcal{H}$, but then $E_{\text{out}}(g)$ will not be as close to $E_{\text{in}}(g)$.

Approximation-generalization tradeoff

Approximation-generalization tradeoff

Small E_{out}: good approximation of f out of sample.

More complex $\mathcal{H} \Longrightarrow$ better chance of **approximating** f

Less complex $\mathcal{H} \Longrightarrow$ better chance of **generalizing** out of sample

Ideal $\mathcal{H} = \{f\}$ winning lottery ticket ☺

© Creator: Yaser Abu-Mostafa - LFD Lecture 8 3/22

Quantifying the tradeoff

Quantifying the tradeoff

VC analysis was one approach: $E_{\text{out}} \leq E_{\text{in}} + \Omega$

Bias-variance analysis is another: decomposing E_{out} into

1. How well $\mathcal{H}$ can approximate f
2. How well we can zoom in on a good $h \in \mathcal{H}$

Applies to **real-valued targets** and uses **squared error**

© AM Creator: Yaser Abu-Mostafa - LFD Lecture 8

Bias and variance decomposition

Start with E_{out}

$$E_{\text{out}}(g^{(\mathcal{D})}) = \mathbb{E}_{\mathbf{x}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x})\right)^2\right]$$

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[E_{\text{out}}(g^{(\mathcal{D})})\right] &= \mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_{\mathbf{x}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x})\right)^2\right]\right] \\ &= \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x})\right)^2\right]\right]\end{aligned}$$

Now, let us focus on:

$$\mathbb{E}_{\mathcal{D}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x})\right)^2\right]$$

© Creator: Yaser Abu-Mostafa - LFD Lecture 8

Bias and variance decomposition

Bias and variance

$$\mathbb{E}_{\mathcal{D}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x})\right)^2\right] = \underbrace{\mathbb{E}_{\mathcal{D}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2\right]}_{\text{var}(\mathbf{x})} + \underbrace{\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right)^2}_{\text{bias}(\mathbf{x})}$$

Therefore, $\mathbb{E}_{\mathcal{D}}\left[E_{\text{out}}(g^{(\mathcal{D})})\right] = \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\left(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x})\right)^2\right]\right]$

$$= \mathbb{E}_{\mathbf{x}}[\text{bias}(\mathbf{x}) + \text{var}(\mathbf{x})]$$

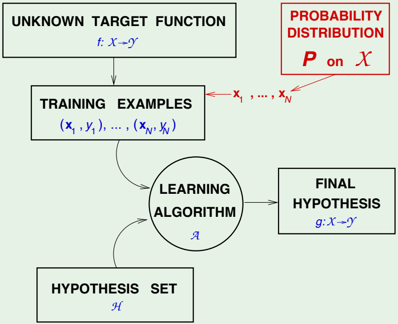
$$= \quad \text{bias} \quad + \quad \text{var}$$

© Creator: Yaser Abu-Mostafa - LFD Lecture 8

Outline

Learning diagram

The learning diagram - where we left it



© Creator: Yaser Abu-Mostafa - LFD Lecture 4

6/22

Error measures

What does "$h \approx f$" mean?

Error measure: $E(h, f)$

Almost always *pointwise definition*: $\mathrm{e}\left(h(\mathbf{x}), f(\mathbf{x})\right)$

Examples:

Squared error: $\mathrm{e}\left(h(\mathbf{x}), f(\mathbf{x})\right) = \left(h(\mathbf{x}) - f(\mathbf{x})\right)^2$

Binary error: $\mathrm{e}\left(h(\mathbf{x}), f(\mathbf{x})\right) = [\![h(\mathbf{x}) \neq f(\mathbf{x})]\!]$

© Creator: Yaser Abu-Mostafa - LFD Lecture 4

From pointwise to overall

From pointwise to overall

Overall error $E(h, f)$ = average of pointwise errors $\mathrm{e}\left(h(\mathbf{x}), f(\mathbf{x})\right)$.

In-sample error:

$$E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^{N} \mathrm{e}\left(h(\mathbf{x}_n), f(\mathbf{x}_n)\right)$$

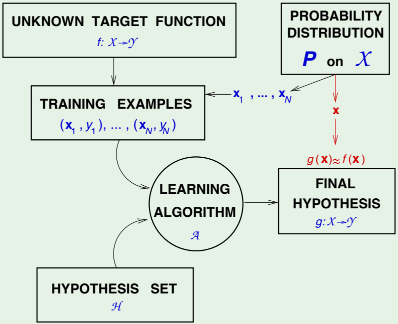
Out-of-sample error:

$$E_{\text{out}}(h) = \mathbb{E}_{\mathbf{x}}\left[\mathrm{e}\left(h(\mathbf{x}), f(\mathbf{x})\right)\right]$$

© Creator: Yaser Abu-Mostafa - LFD Lecture 4

Learning diagram updated

The learning diagram - with pointwise error



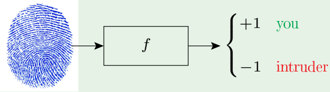
© Creator: Yaser Abu-Mostafa - LFD Lecture 4

9/22

How to choose the error measure?

How to choose the error measure

Fingerprint verification:



Two types of error:

false accept and *false reject*

How do we penalize each type?

		f: $+1$	f: -1
h	$+1$	no error	*false accept*
	-1	*false reject*	no error

© Creator: Yaser Abu-Mostafa - LFD Lecture 4

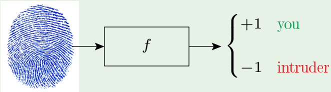
The supermarket example

The error measure - 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: $+1$	f: -1
h	$+1$	0	1
	-1	10	0

© Creator: Yaser Abu-Mostafa - LFD Lecture 4

The CIA example

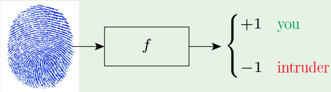
The error measure - for the CIA

CIA verifies fingerprint for security

False accept is a disaster!

False reject can be tolerated
Try again; you are an employee ☺

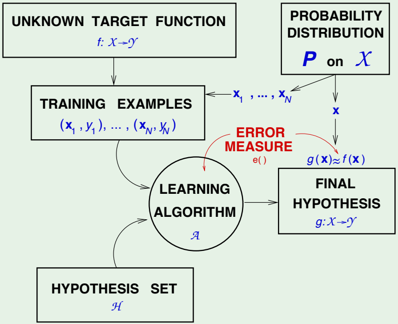
		f	
		$+1$	-1
h	$+1$	0	1000
	-1	1	0



© Creator: Yaser Abu-Mostafa - LFD Lecture 4

Learning diagram updated

The learning diagram - with error measure



© AML Creator: Yaser Abu-Mostafa - LFD Lecture 4 14/22

Outline

Noisy targets

Noisy targets

The 'target function' is not always a *function*

Consider the credit-card approval:

age	23 years
annual salary	$30,000
years in residence	1 year
years in job	1 year
current debt	$15,000
...	...

two 'identical' customers $\longrightarrow$ two different behaviors

© Creator: Yaser Abu-Mostafa - LFD Lecture 4

Target distribution

Target 'distribution'

Instead of $y = f(\mathbf{x})$, we use target *distribution*:

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

$(\mathbf{x}, y)$ is now generated by the joint distribution:

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

Noisy target = deterministic target $f(\mathbf{x}) = \mathbb{E}(y|\mathbf{x})$ plus noise $y - f(\mathbf{x})$

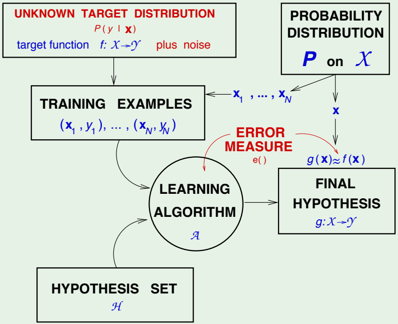
Deterministic target is a special case of noisy target:

$$P(y \mid \mathbf{x}) \text{ is zero except for } y = f(\mathbf{x})$$

© AM Creator: Yaser Abu-Mostafa - LFD Lecture 4 16/22

Final learning diagram

The learning diagram - including noisy target



© Creator: Yaser Abu-Mostafa - LFD Lecture 4

17/22