Machine Learning

André Panisson
Data Science Laboratory - ISI Foundation
andre.panisson@isi.it

Complex Networks Thematic School, Les Houches, April 7-18, 2014

“Machine learning is a scientific discipline concerned with the design and development of algorithms that take as input empirical data, such as that from sensors or databases, and yield patterns or predictions thought to be features of the underlying mechanism that generated the data” (Wikipedia)

Components of learning

Metaphor: **Credit approval**

Applicant information:

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

Approve credit?

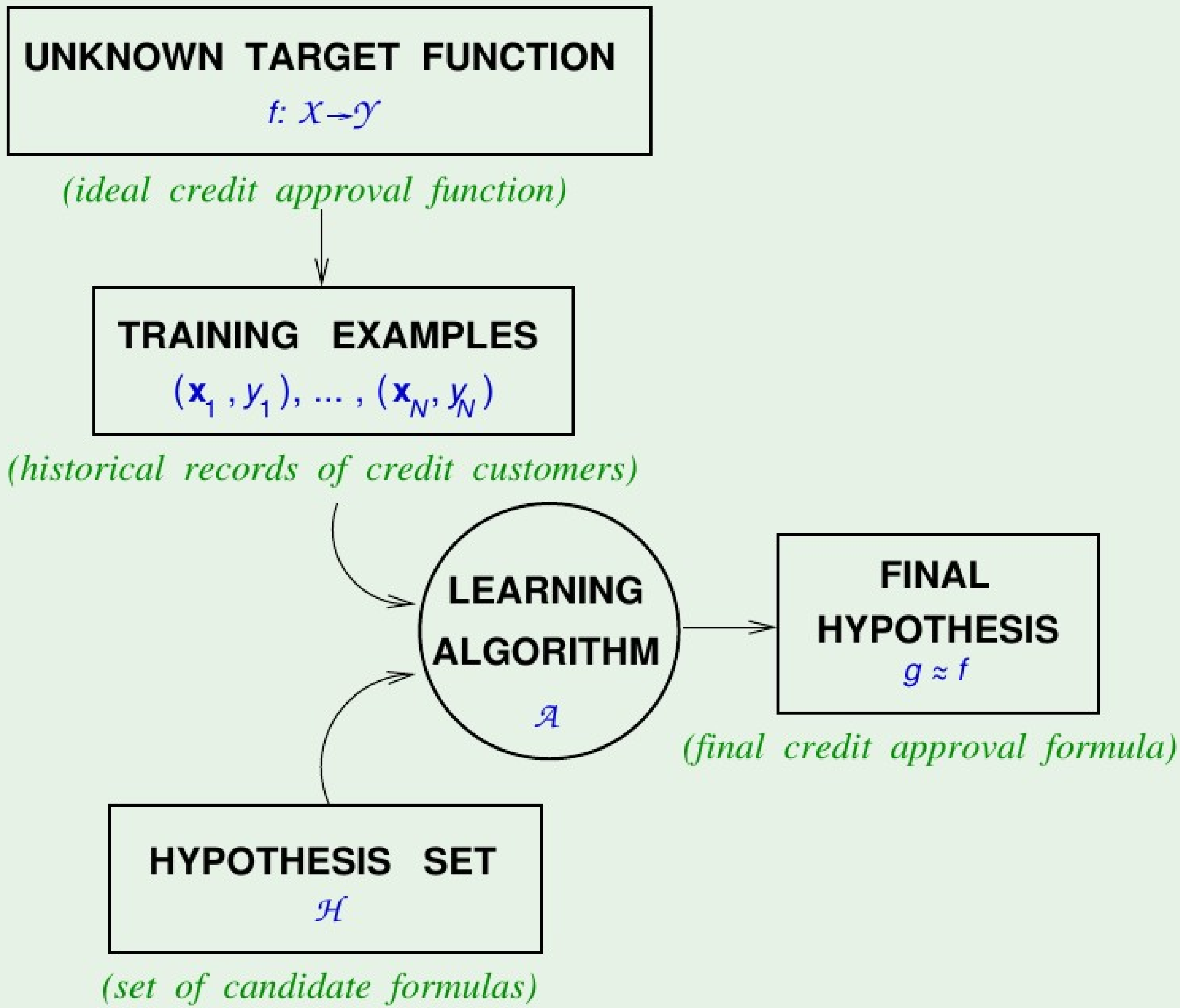
Components of learning

Formalization:

- Input: $\mathbf{x}$ (customer application)
- Output: $\mathbf{y}$ (good/bad customer?)
- Target function: $f\colon \mathcal{X} \to \mathcal{Y}$ (ideal credit approval formula)
- Data: $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \cdots, (\mathbf{x}_N, \mathbf{y}_N)$ (historical records)

↓ ↓ ↓

- Hypothesis: $g\colon \mathcal{X} \to \mathcal{Y}$ (formula to be used)


UNKNOWN TARGET FUNCTION
$f: \mathcal{X} \to \mathcal{Y}$
(ideal credit approval function)
TRAINING EXAMPLES
$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
(historical records of credit customers)
LEARNING ALGORITHM
$\mathcal{A}$
FINAL HYPOTHESIS
$g \approx f$
(final credit approval formula)
HYPOTHESIS SET
$\mathcal{H}$
(set of candidate formulas)

What does $h \approx f$ means?

Objective: minimize some error measure $\boldsymbol{E(h,f)}$

Almost always pointwise definition: $e(h(x), f(x))$

Examples:

Mean squared error (regression)

$$e(h(x), f(x)) = (h(x) - f(x))^2$$

Mean binary error (classification)

$$e(h(x), f(x)) = [\![h(x) \neq f(x)]\!]$$

Overall error $\boldsymbol{E(h,f)}$ is

the average of pointwise errors $e(h(x), f(x))$

In-sample error:

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

Out-of-sample error:

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

What we want to do?

$$\mathbf{E_{out} \approx 0}$$

What we can do?

$\mathbf{E_{in} \approx 0}$ (approximation)

$\mathbf{E_{in} \approx E_{out}}$ (generalization)

The learning problem is thus split in 2 questions:

- Can we make $\mathbf{E_{in}}(g)$ small enough?

- Can we make sure that $\mathbf{E_{in}}(g)$ is close enough to $\mathbf{E_{out}}(g)$?

related to complexity of g

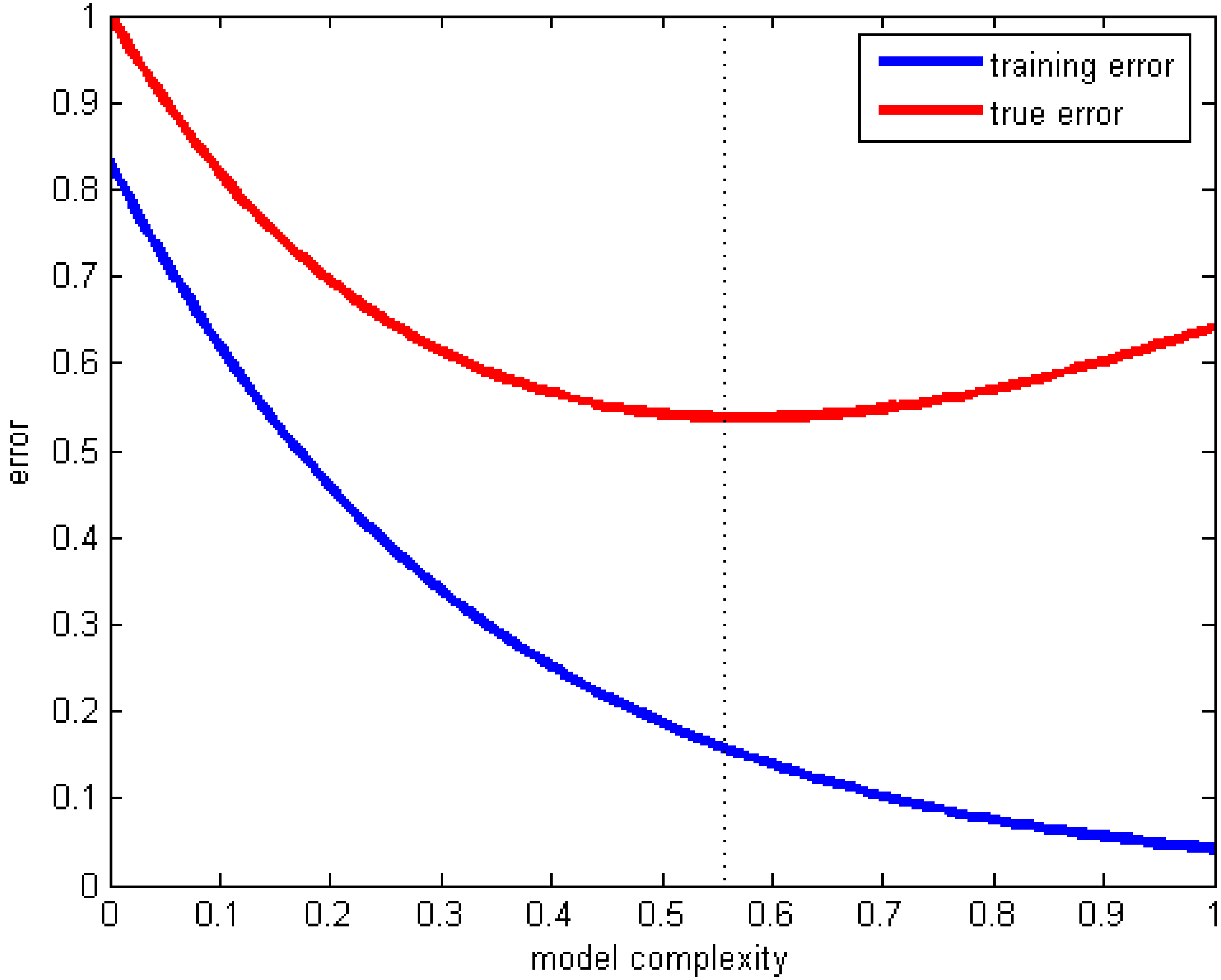
$$\mathbb{P}[|E_{out}(g)-E_{in}(g)|>\epsilon]\leqslant 2\,M\,e^{-2\epsilon^2 N}$$

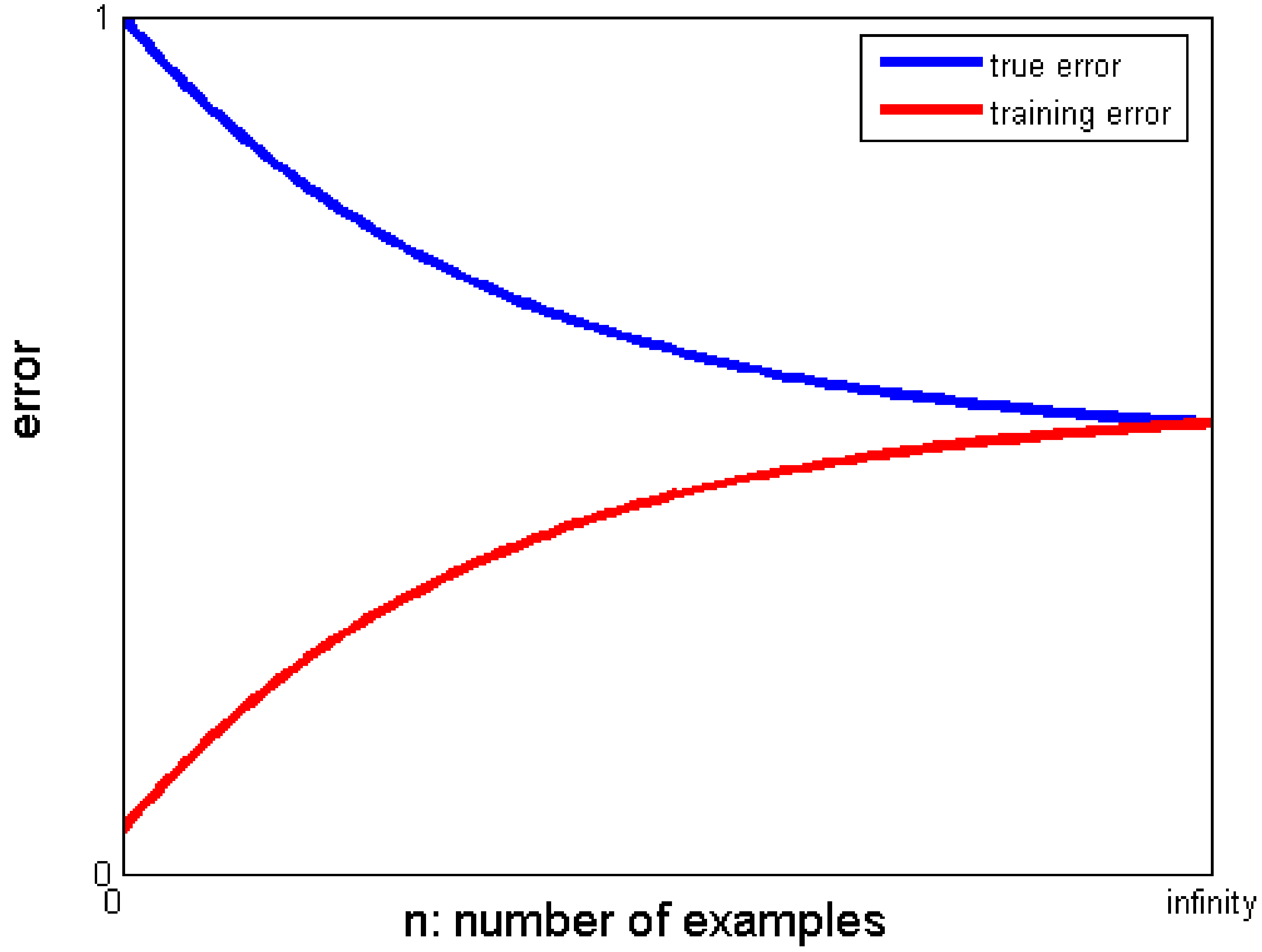
(from Hoeffding's inequality, Vapnik–Chervonenkis theory)

We are always searching for a model that is, at the same time:

- Sufficiently complex to reduce the prediction error as much as possible
- Sufficiently simple to generalize to unknown data

This tradeoff is also known as Bias-Variance Tradeoff


training error
true error
error
model complexity
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


1
0
error
true error
training error
0
infinity
n: number of examples

But in practice, what we do?

Try to approximate E_{in} to 0

and at the same time

Try to estimate E_{out} via **cross-validation**

IPython Notebook: leshouches05