Aprendizaje automático

FACULTAD DE INGENIERÍA
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

Componentes del aprendizaje

Components of learning

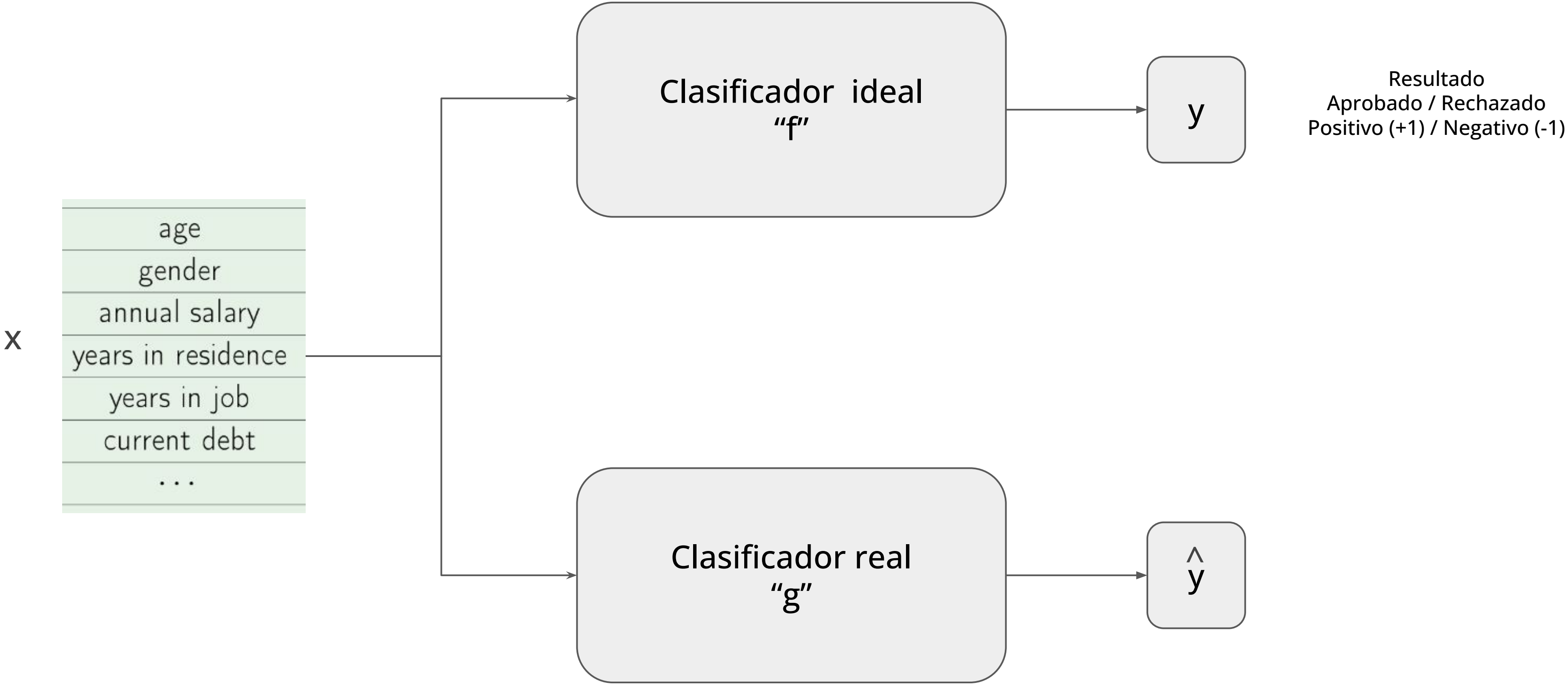
Metaphor: Credit approval

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?

Aprobación de crédito como un problema de clasificación



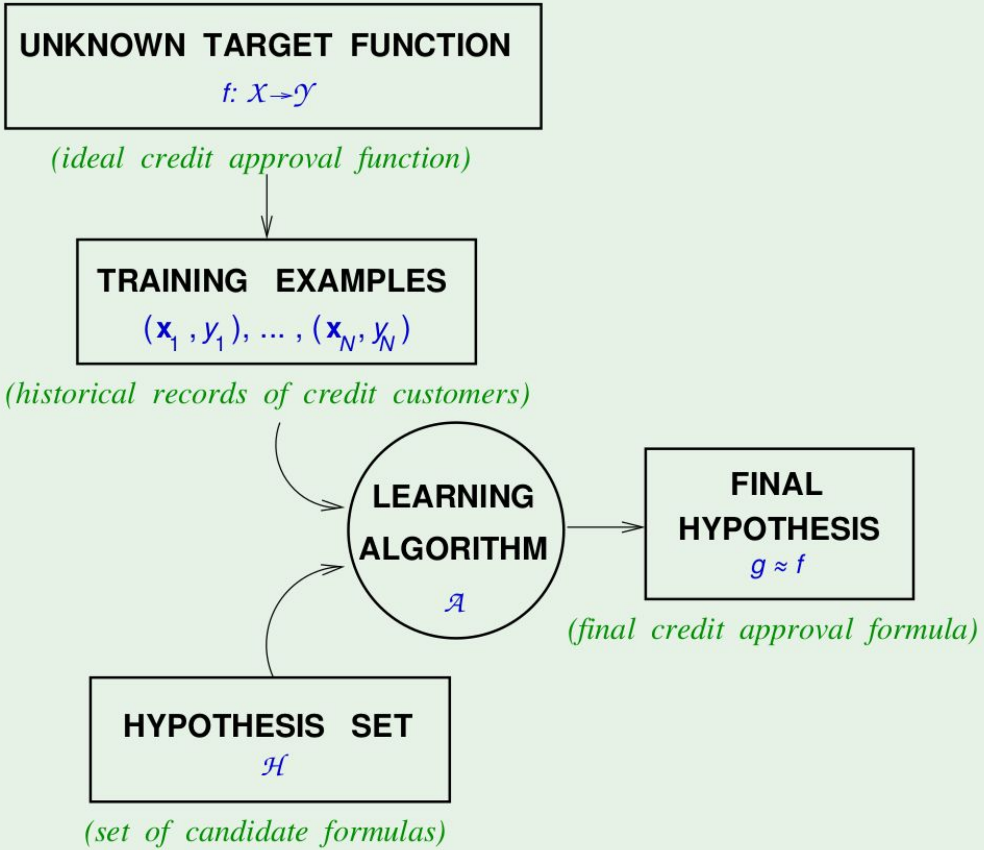
Components of learning

Formalization:

- Input: $\mathbf{x}$ *(customer application)*
- Output: y *(good/bad customer?)*
- Target function: $f : \mathcal{X} \to \mathcal{Y}$ *(ideal credit approval formula)*
- Data: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$ *(historical records)*

$\downarrow \quad \downarrow \quad \downarrow$

- Hypothesis: $g : \mathcal{X} \to \mathcal{Y}$ *(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)

Solution components

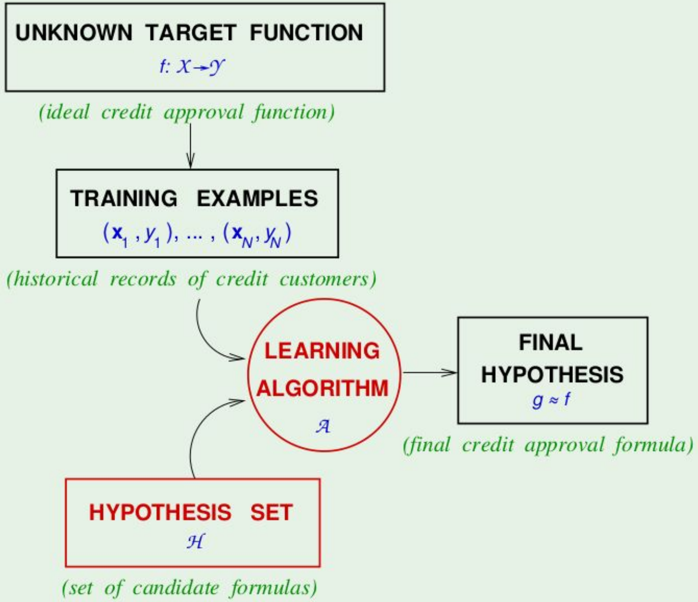
The 2 solution components of the learning problem:

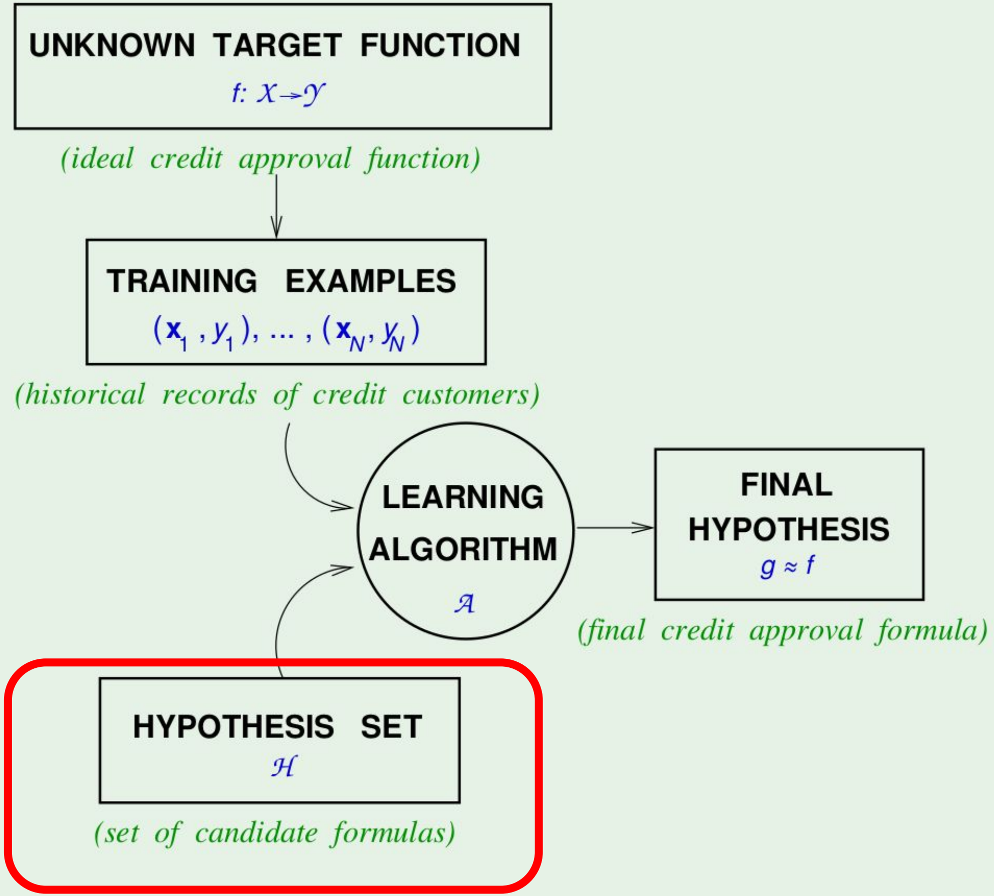
- The Hypothesis Set

  $\mathcal{H} = \{h\} \qquad g \in \mathcal{H}$

- The Learning Algorithm

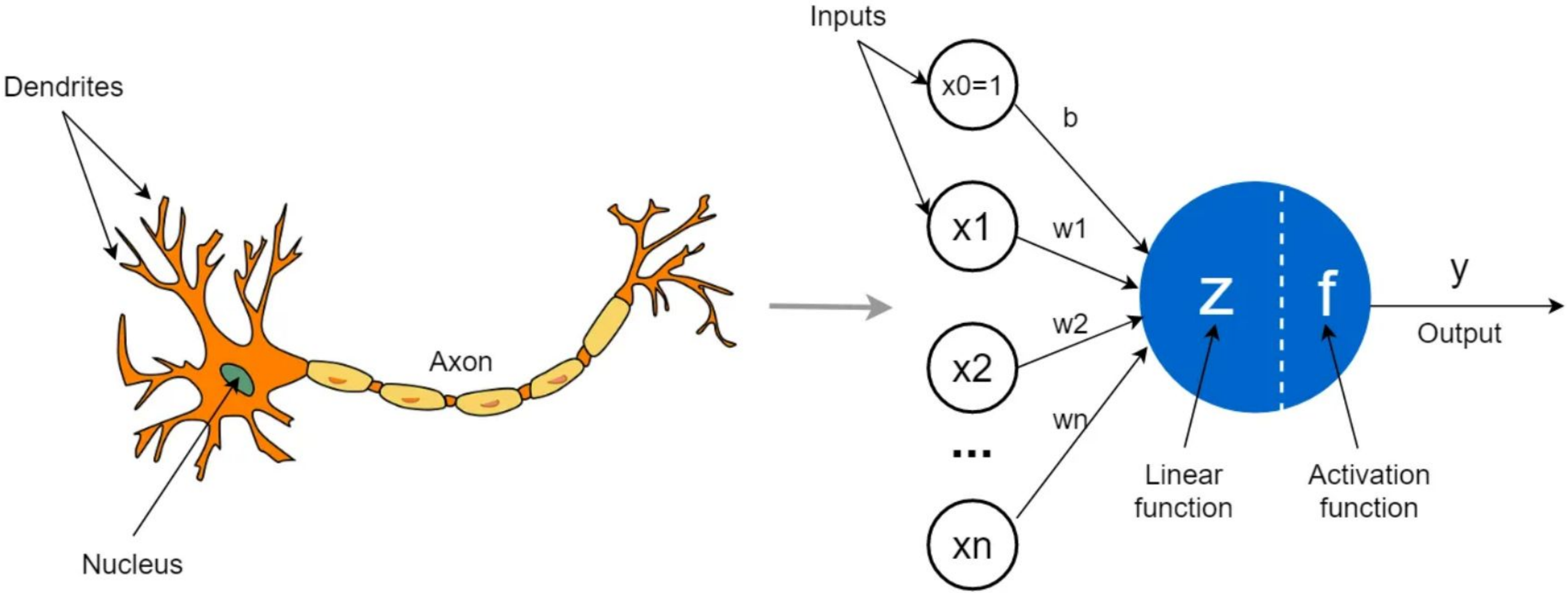
Together, they are referred to as the *learning model*.




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)

Un posible set de hipótesis: el perceptrón

- Inspirado en modelo simplificado de neurona biológica



A simple hypothesis set - the 'perceptron'

For input $\mathbf{x} = (x_1, \cdots, x_d)$ 'attributes of a customer'

x_1	age
x_2	gender
x_3	annual salary
.	years in residence
-	years in job
x_d	current debt
	...

$$\text{Approve credit if} \quad \sum_{i=1}^{d} w_i x_i > \text{threshold},$$

$$\text{Deny credit if} \quad \sum_{i=1}^{d} w_i x_i < \text{threshold}.$$

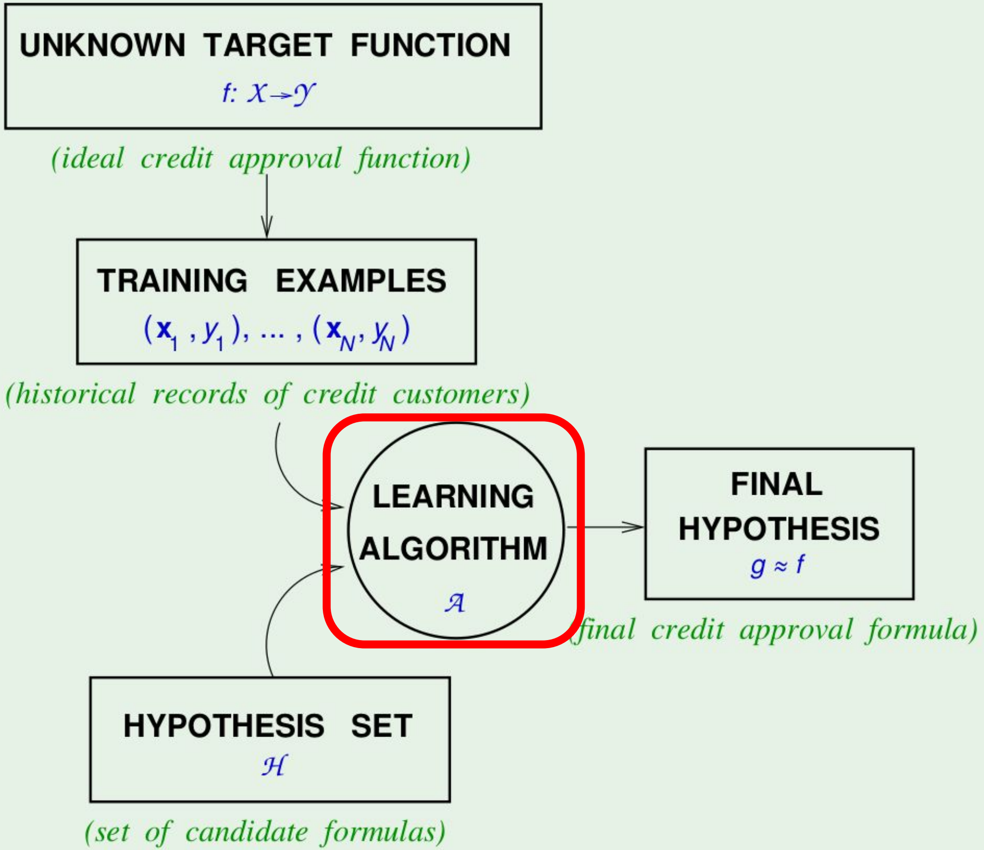
This linear formula $h \in \mathcal{H}$ can be written as

$$h(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) - \text{threshold}\right)$$

Un posible set de hipótesis: el perceptrón

- Características del set de hipótesis ?
    - La función g va a ser:
        - Combinación lineal de las entradas + umbralización

- Cómo se espera que sean los datos de entrada ?
    - Linealmente separables

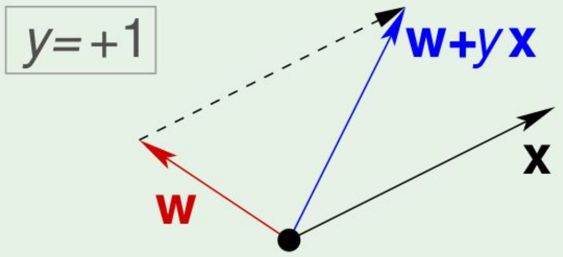


UNKNOWN TARGET FUNCTION
$f: \mathcal{X} \to \mathcal{Y}$
(ideal credit approval function)
TRAINING EXAMPLES
$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
(historical records of credit customers)
LEARNING ALGORITHM
$\mathcal{A}$
FINAL HYPOTHESIS
$g \approx f$
(final credit approval formula)
HYPOTHESIS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

A simple learning algorithm - PLA

The perceptron implements

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^{\intercal}\mathbf{x})$$

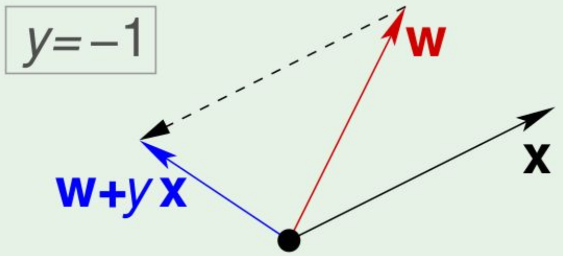


Given the training set:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

pick a misclassified point:

$$\text{sign}(\mathbf{w}^{\intercal}\mathbf{x}_n) \neq y_n$$



and update the weight vector:

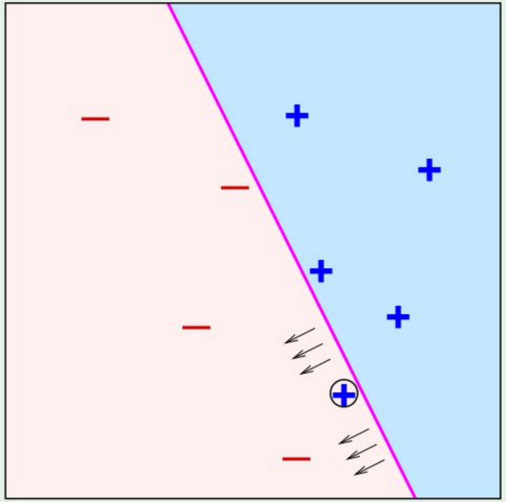
$$\boxed{\mathbf{w} \leftarrow \mathbf{w} + y_n\mathbf{x}_n}$$

Iterations of PLA

- One iteration of the PLA:

$$\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}$$

where $(\mathbf{x}, y)$ is a misclassified training point.



- At iteration $t = 1, 2, 3, \cdots$, pick a misclassified point from

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

and run a PLA iteration on it.

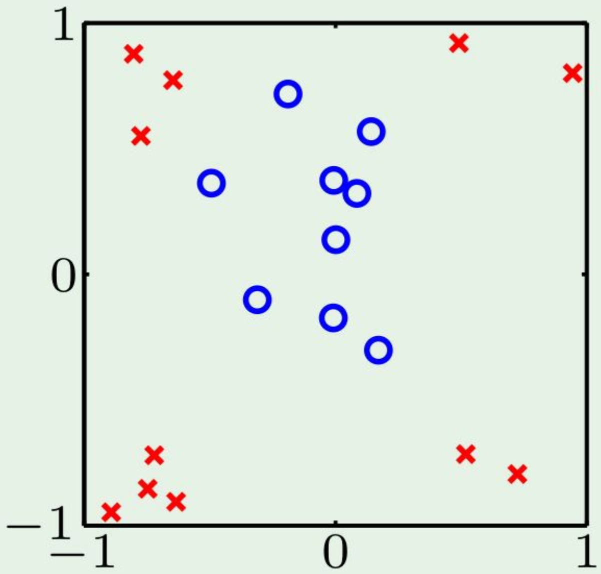
- That's it!

El algoritmo siempre para y llega a una solución ?

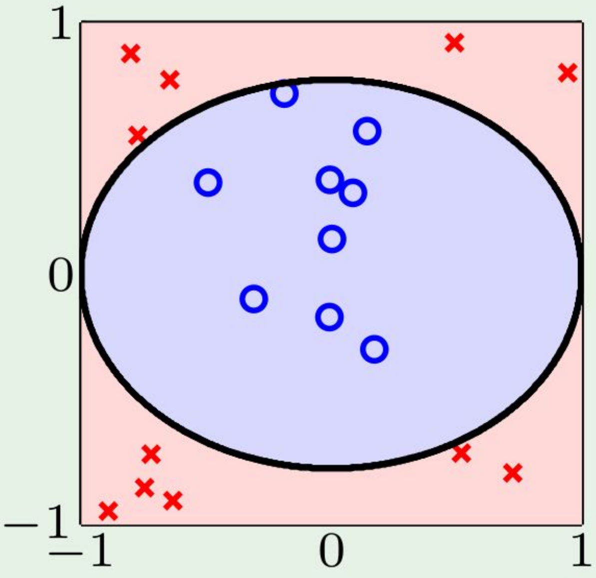
Qué pasa si los datos no son linealmente separables ?

Linear is limited

Data:



Hypothesis:



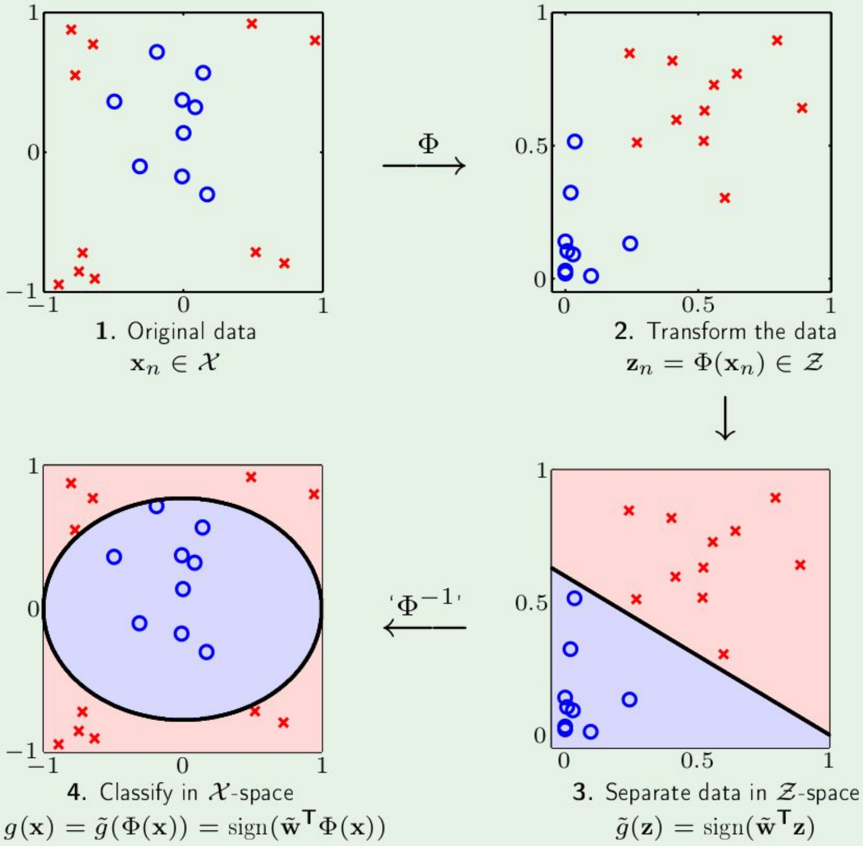
Another example

Credit line is affected by 'years in residence'

but **not** in a linear way!

Nonlinear $[[x_i < 1]]$ and $[[x_i > 5]]$ are better.

Can we do that with linear models?



Φ
$\longrightarrow$
1. Original data
$\mathbf{x}_n \in \mathcal{X}$
2. Transform the data
$\mathbf{z}_n = \Phi(\mathbf{x}_n) \in \mathcal{Z}$
3. Separate data in $\mathcal{Z}$-space
$\tilde{g}(\mathbf{z}) = \text{sign}(\tilde{\mathbf{w}}^{\mathsf{T}}\mathbf{z})$
'Φ^{-1}'
4. Classify in $\mathcal{X}$-space
$g(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x})) = \text{sign}(\tilde{\mathbf{w}}^{\mathsf{T}}\Phi(\mathbf{x}))$

What transforms to what

$$\mathbf{x} = (x_0, x_1, \cdots, x_d) \xrightarrow{\Phi} \mathbf{z} = (z_0, z_1, \cdots\cdots\cdots, z_{\tilde{d}})$$

$$\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N \xrightarrow{\Phi} \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N$$

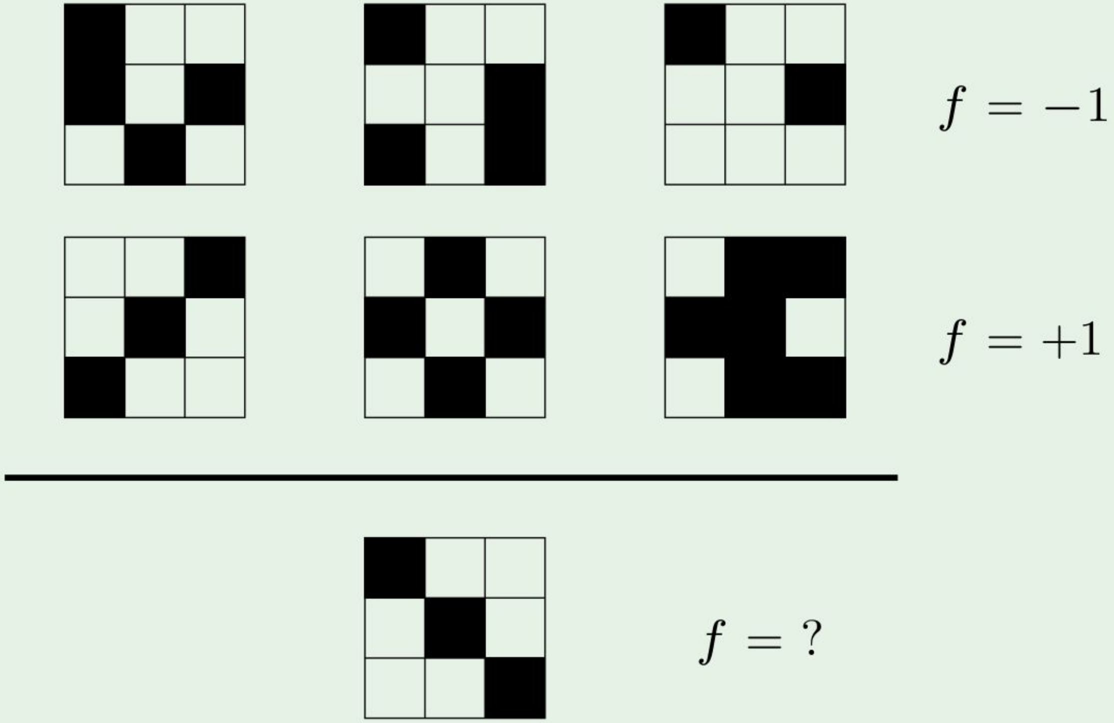
$$y_1, y_2, \cdots, y_N \xrightarrow{\Phi} y_1, y_2, \cdots, y_N$$

No weights in $\mathcal{X}$ $\qquad$ $\tilde{\mathbf{w}} = (w_0, w_1, \cdots\cdots\cdots, w_{\tilde{d}})$

$$g(\mathbf{x}) = \text{sign}(\tilde{\mathbf{w}}^{\mathsf{T}} \Phi(\mathbf{x}))$$

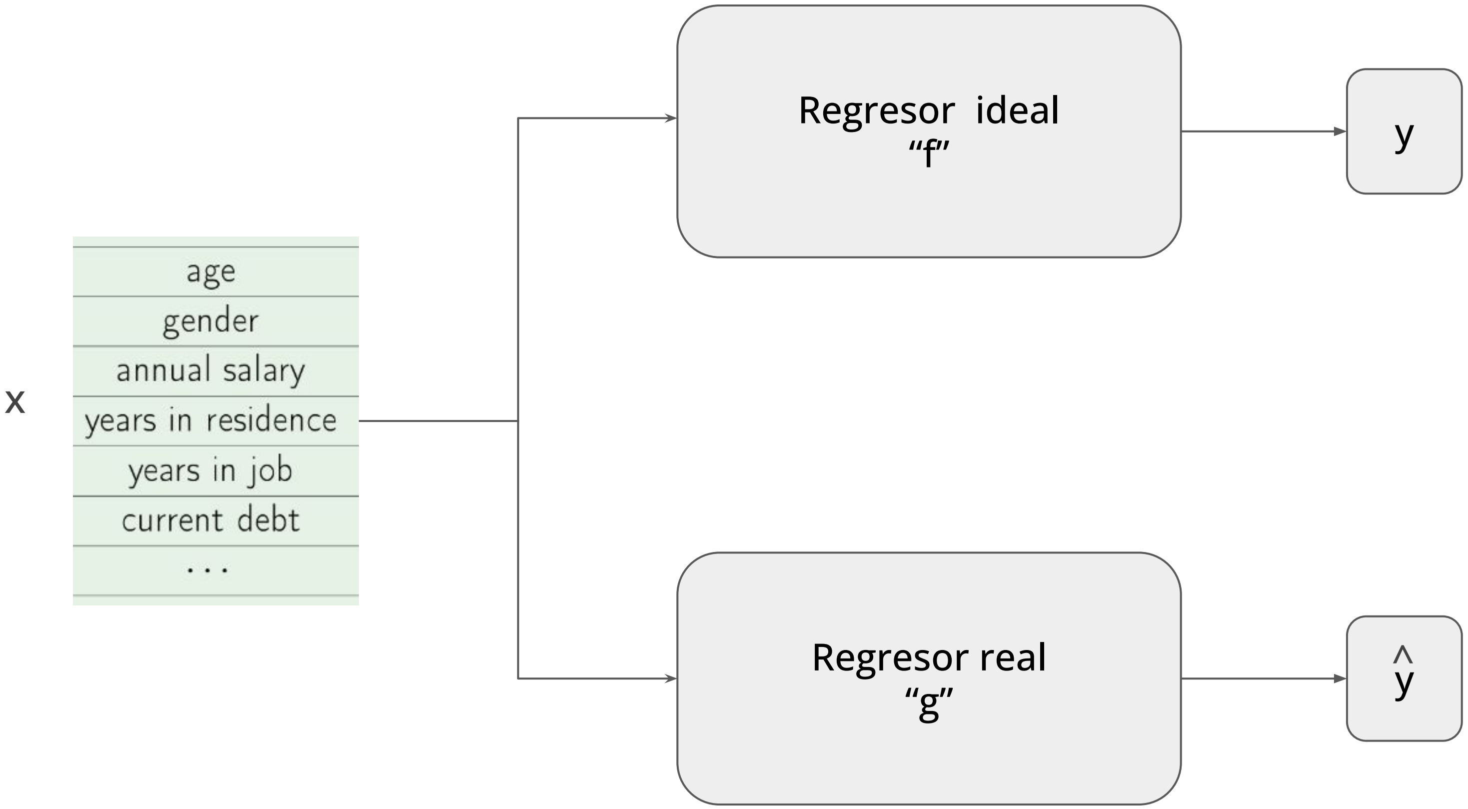
Adivinanza

A Learning puzzle



Volvemos al crédito...

Aprobación de nivel de crédito como un problema de regresión



Resultado
Línea de crédito
$

Credit again

Classification: Credit approval (yes/no)

Regression: Credit line (dollar amount)

Input: $\mathbf{x} =$

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

Linear regression output: $h(\mathbf{x}) = \sum_{i=0}^{d} w_i \, x_i = \mathbf{w}^{\intercal}\mathbf{x}$

The data set

Credit officers decide on credit lines:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

$y_n \in \mathbb{R}$ is the credit line for customer $\mathbf{x}_n$.

Linear regression tries to replicate that.

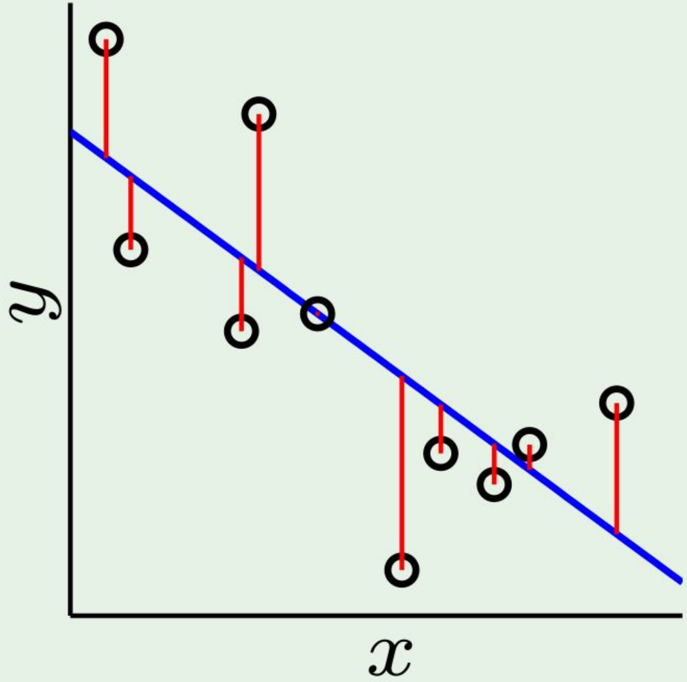
How to measure the error

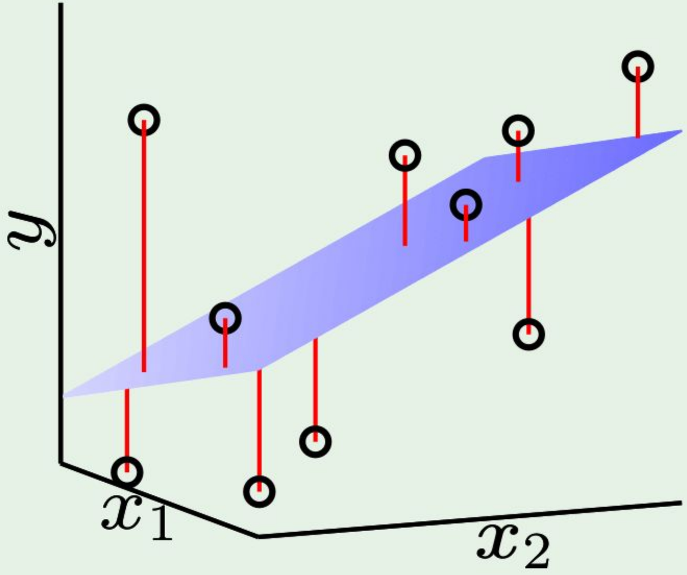
How well does $h(\mathbf{x}) = \mathbf{w}^{\intercal}\mathbf{x}$ approximate $f(\mathbf{x})$?

In linear regression, we use squared error $(h(\mathbf{x}) - f(\mathbf{x}))^2$

in-sample error: $E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^{N} (h(\mathbf{x}_n) - y_n)^2$

Illustration of linear regression





The expression for E_{in}

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^{\mathsf{T}}\mathbf{x}_n - y_n)^2$$

$$= \frac{1}{N}\|\mathrm{X}\mathbf{w} - \mathbf{y}\|^2$$

where

$$\mathrm{X} = \begin{bmatrix} \text{—}\mathbf{x}_1^{\mathsf{T}}\text{—} \\ \text{—}\mathbf{x}_2^{\mathsf{T}}\text{—} \\ \vdots \\ \text{—}\mathbf{x}_N^{\mathsf{T}}\text{—} \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

Minimizing E_{in}

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\|\mathrm{X}\mathbf{w} - \mathbf{y}\|^2$$

$$\nabla E_{\text{in}}(\mathbf{w}) = \frac{2}{N}\mathrm{X}^{\intercal}(\mathrm{X}\mathbf{w} - \mathbf{y}) = \mathbf{0}$$

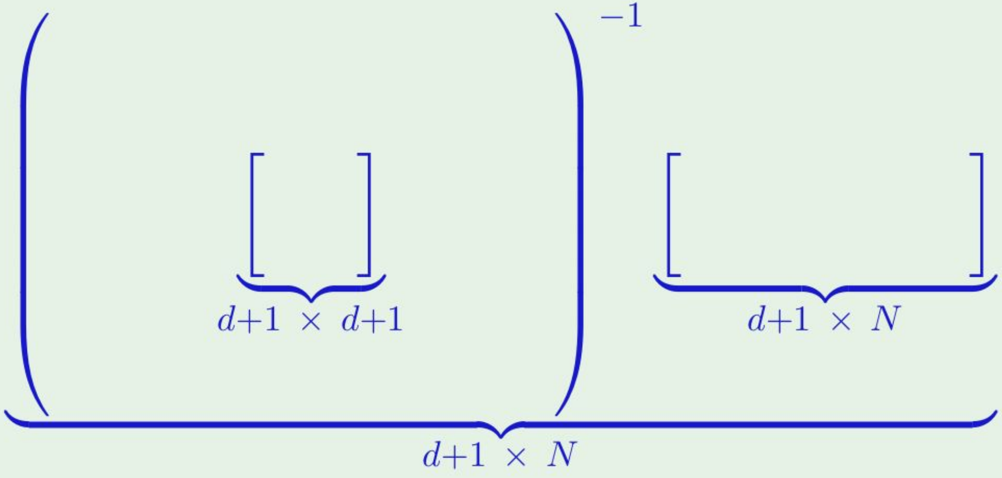
$$\mathrm{X}^{\intercal}\mathrm{X}\mathbf{w} = \mathrm{X}^{\intercal}\mathbf{y}$$

$$\mathbf{w} = \mathrm{X}^{\dagger}\mathbf{y} \quad \text{where} \quad \mathrm{X}^{\dagger} = (\mathrm{X}^{\intercal}\mathrm{X})^{-1}\mathrm{X}^{\intercal}$$

$\mathrm{X}^{\dagger}$ is the '**pseudo-inverse**' of X

The pseudo-inverse

$$X^{\dagger} = (X^{\top}X)^{-1}X^{\top}$$



The linear regression algorithm

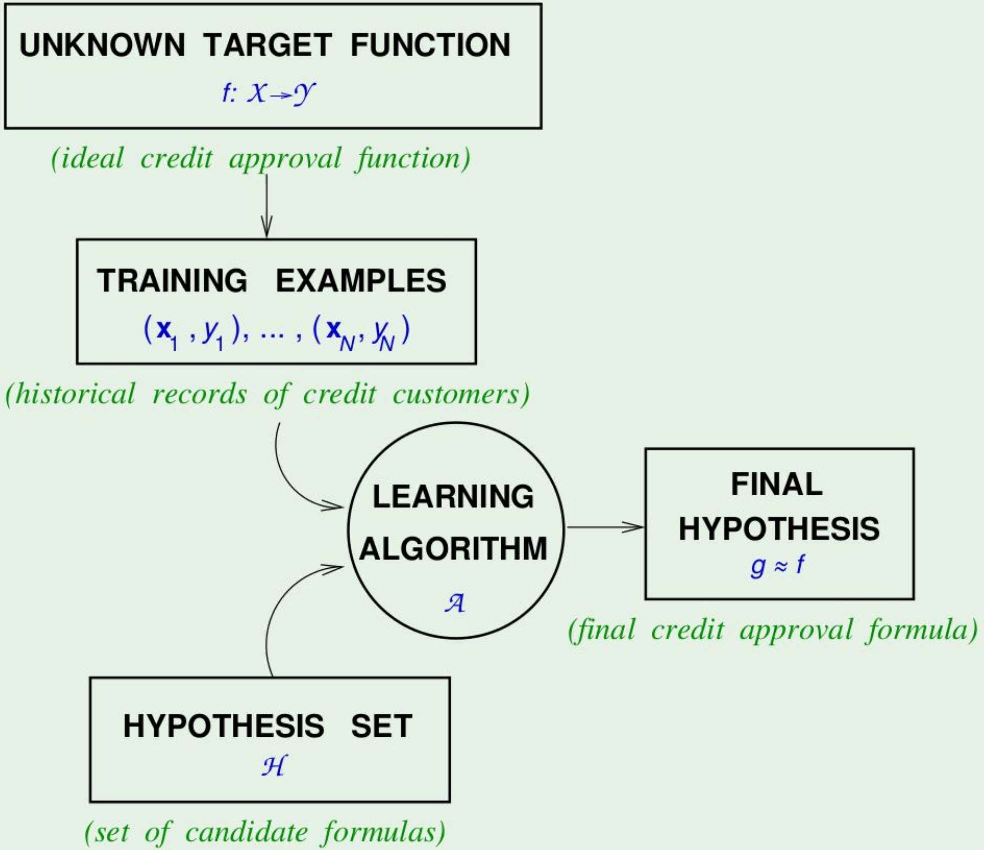
1: Construct the matrix X and the vector $\mathbf{y}$ from the data set $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$ as follows

$$\underbrace{\mathrm{X} = \begin{bmatrix} —\mathbf{x}_1^{\mathsf{T}}— \\ —\mathbf{x}_2^{\mathsf{T}}— \\ \vdots \\ —\mathbf{x}_N^{\mathsf{T}}— \end{bmatrix}}_{\text{input data matrix}}, \qquad \underbrace{\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}}.$$

2: Compute the pseudo-inverse $\mathrm{X}^\dagger = (\mathrm{X}^{\mathsf{T}}\mathrm{X})^{-1}\mathrm{X}^{\mathsf{T}}$.

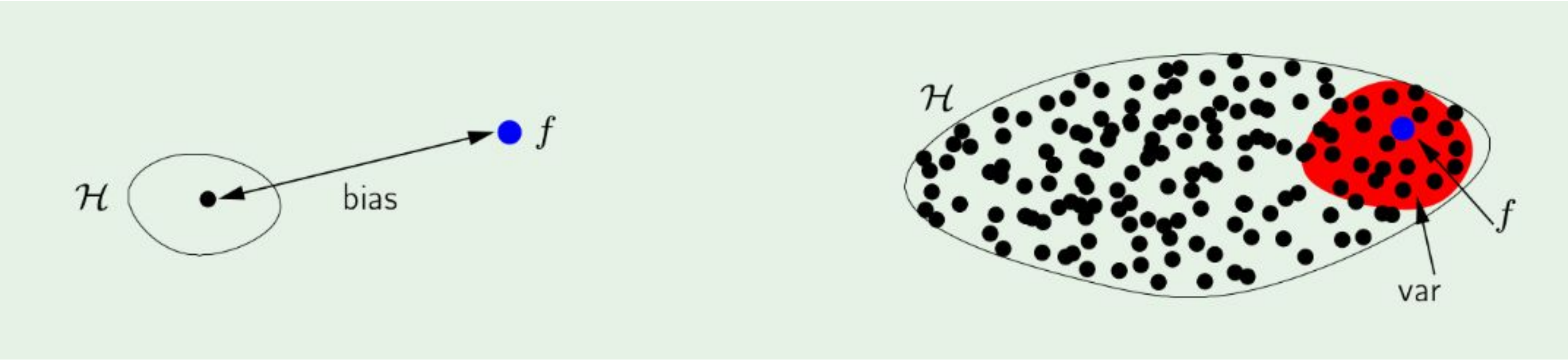
3: Return $\mathbf{w} = \mathrm{X}^\dagger \mathbf{y}$.

Aproximación y generalización


UNKNOWN TARGET FUNCTION
$f: \mathcal{X} \rightarrow \mathcal{Y}$
(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)

El problema

- El set de hipótesis elegido influye en:
  - El error dentro de la muestra E_in (aproximación a las muestras disponibles)
  - El error fuera de la muestra E_out (error fuera de muestra)
- Queremos que nuestro modelo
  - aproxime bien a las muestras (x,y) disponibles
  - generalice bien para nuevos datos
- Va a existir un compromiso



Approximation-generalization tradeoff

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

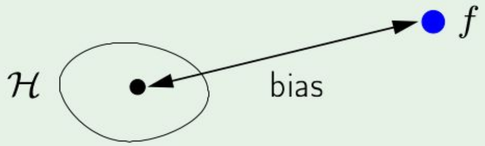
More complex $\mathcal{H} \Longrightarrow$ better chance of **approximating** f

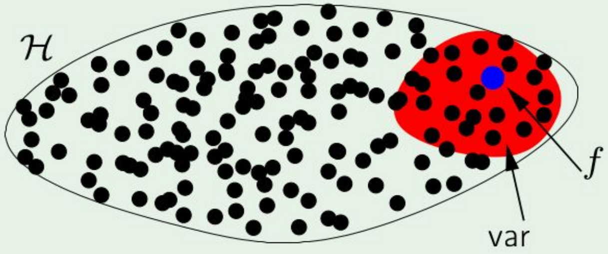
Less complex $\mathcal{H} \Longrightarrow$ better chance of **generalizing** out of sample

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

The tradeoff

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





$$\downarrow \qquad \mathcal{H}\uparrow \qquad \uparrow$$

Quantifying the tradeoff

Bias-variance analysis

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**

Example: sine target

$f:[-1,1]\to\mathbb{R}$ $\qquad f(x)=\sin(\pi x)$

Only two training examples! $\qquad N=2$

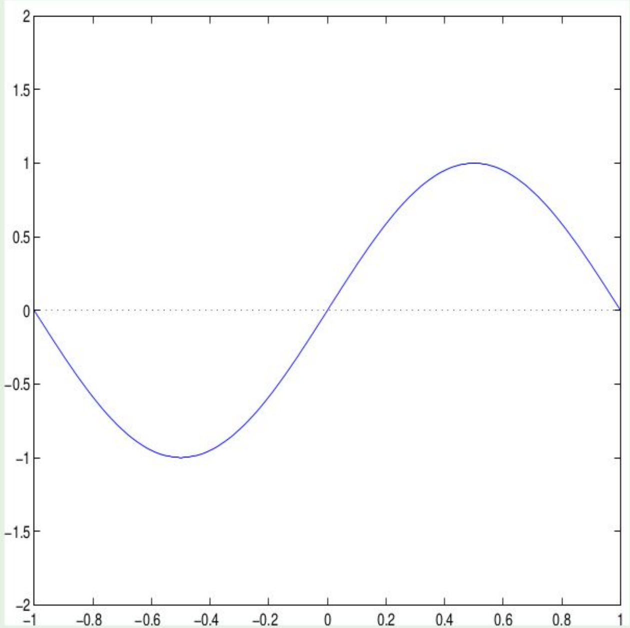
Two models used for learning:

$\mathcal{H}_0$: $\quad h(x)=b$

$\mathcal{H}_1$: $\quad h(x)=ax+b$

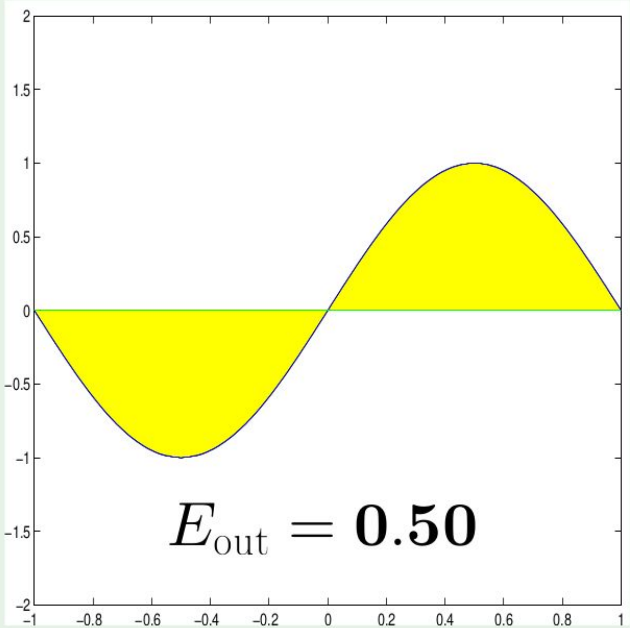
Which is better, $\mathcal{H}_0$ or $\mathcal{H}_1$?

f

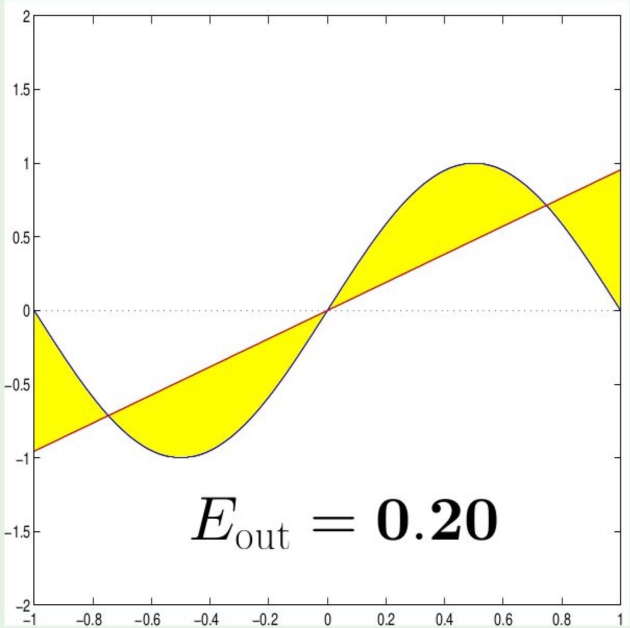


Approximation - $\mathcal{H}_0$ versus $\mathcal{H}_1$

$\mathcal{H}_0$

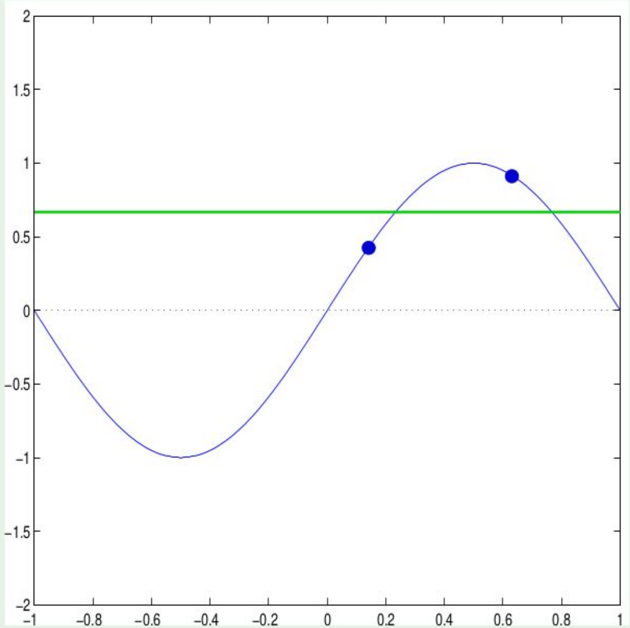


$\mathcal{H}_1$

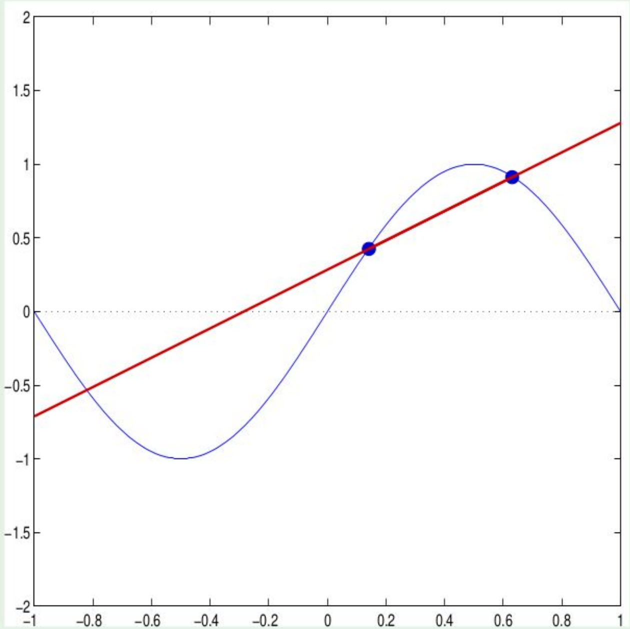


Learning - $\mathcal{H}_0$ versus $\mathcal{H}_1$

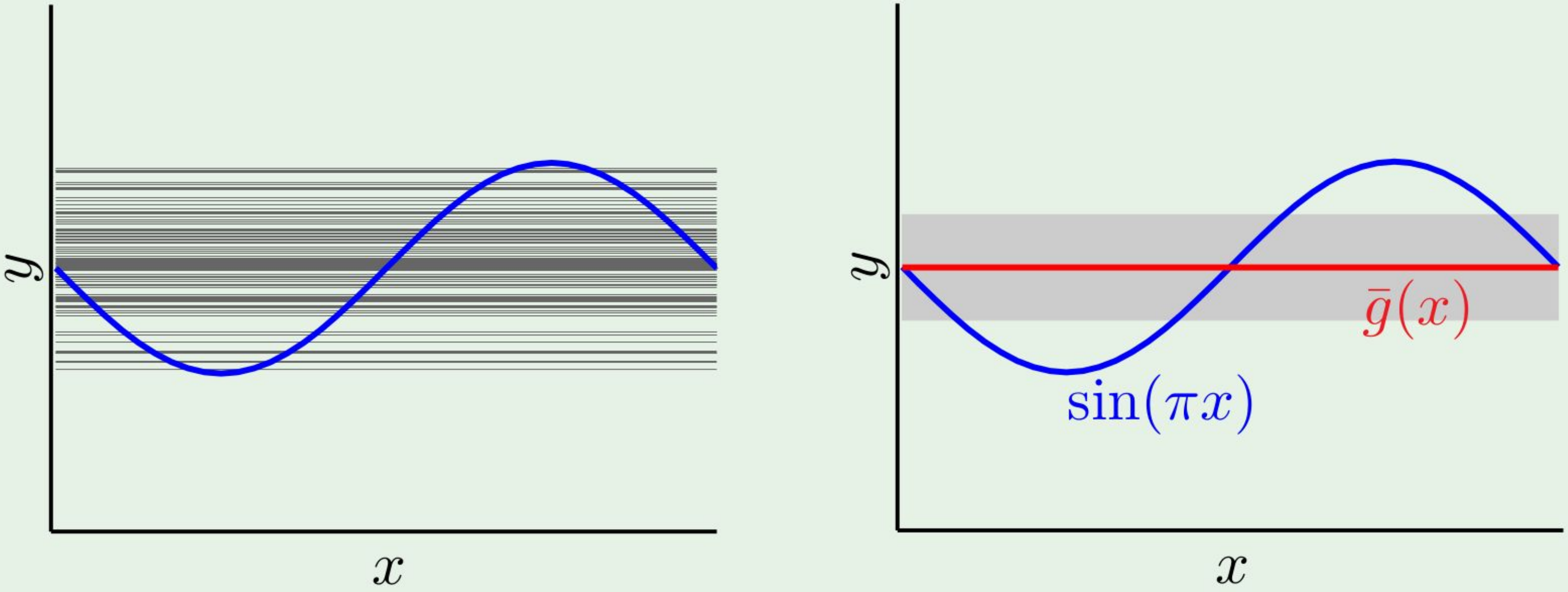
$\mathcal{H}_0$



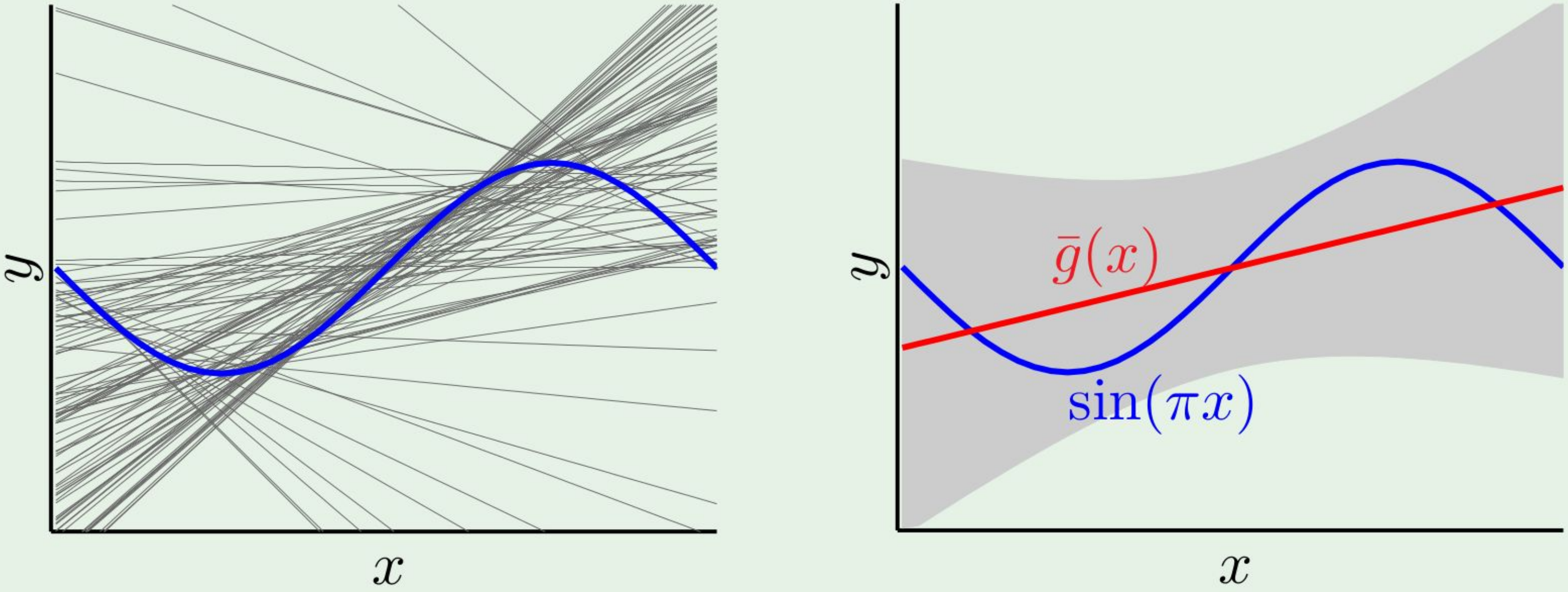
$\mathcal{H}_1$



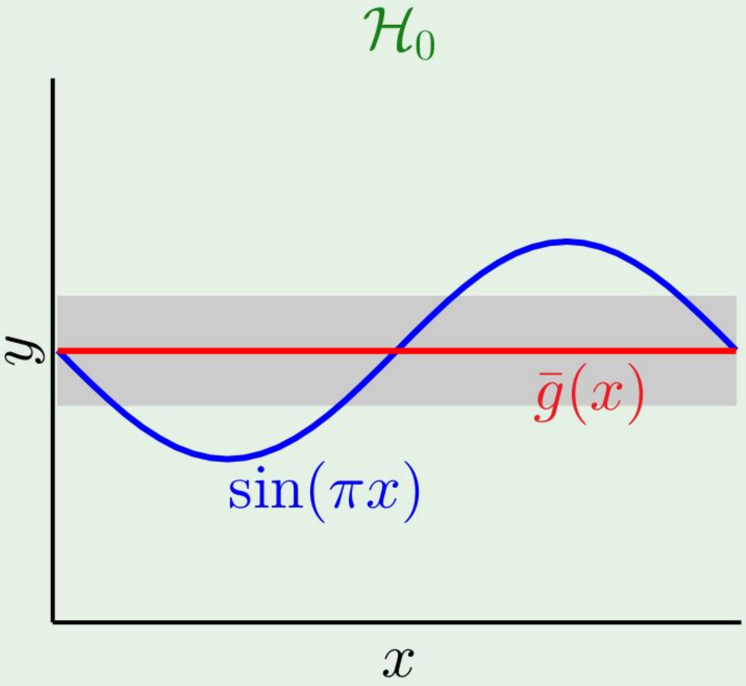
Bias and variance - $\mathcal{H}_0$



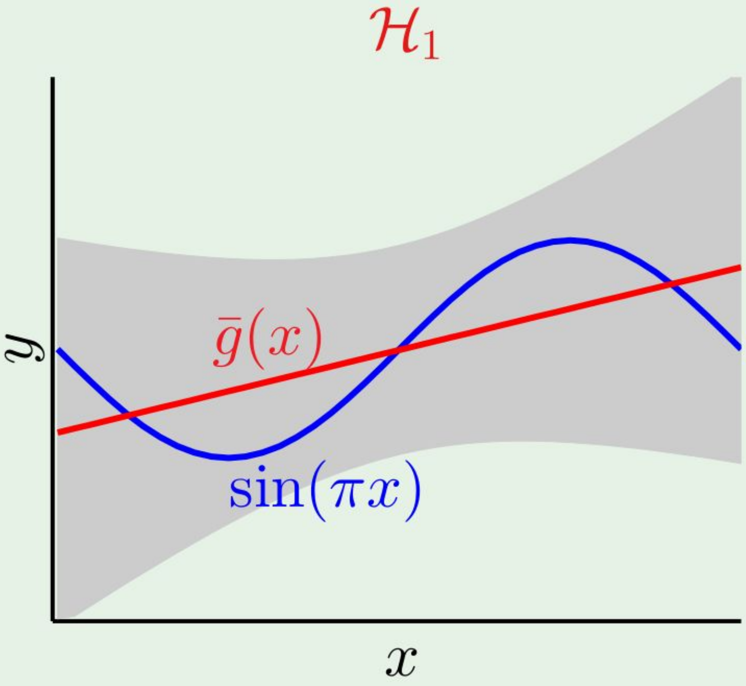
Bias and variance - $\mathcal{H}_1$



and the winner is . . .



bias = **0.50** var = **0.25**



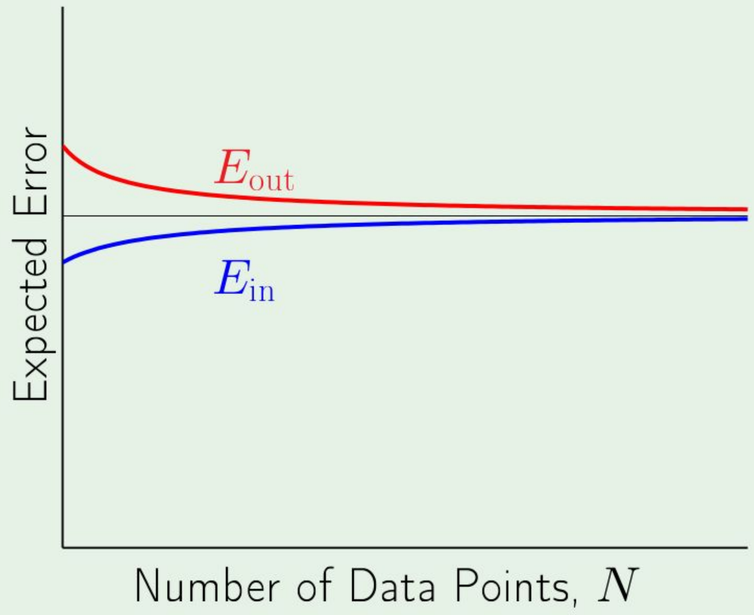
bias = **0.21** var = **1.69**

Lesson learned

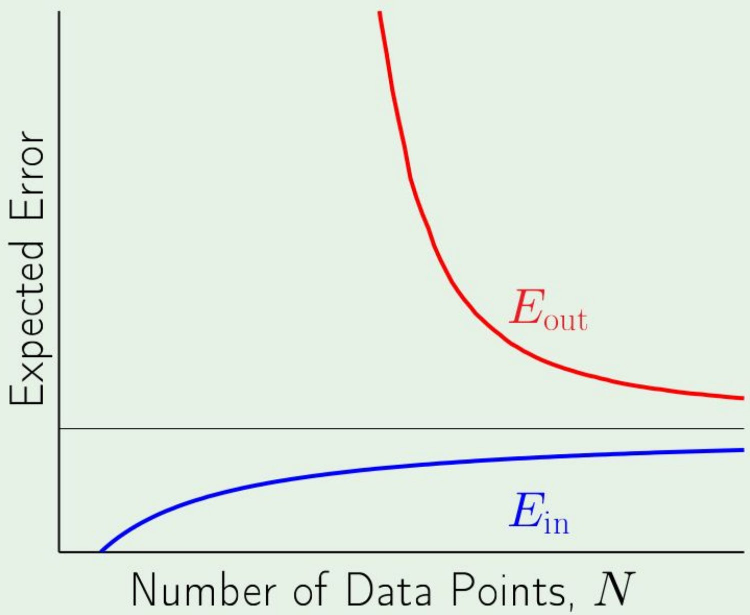
Match the **'model complexity'**

to the data resources, not to the target complexity

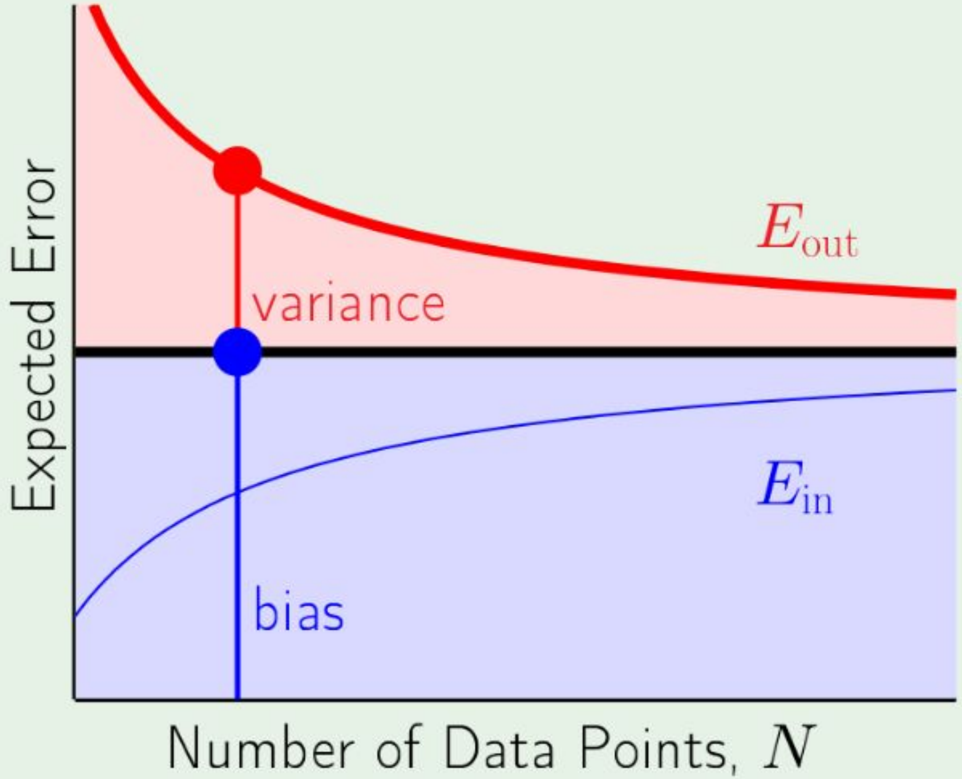
The curves



Simple Model



Complex Model



bias-variance