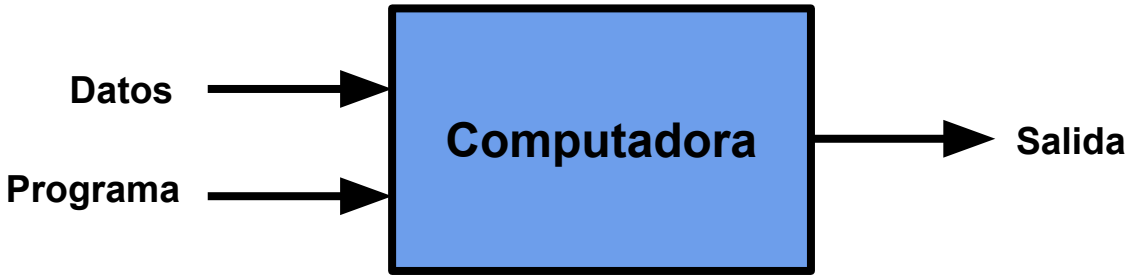
Módulo 3. IA y grandes volúmenes de datos

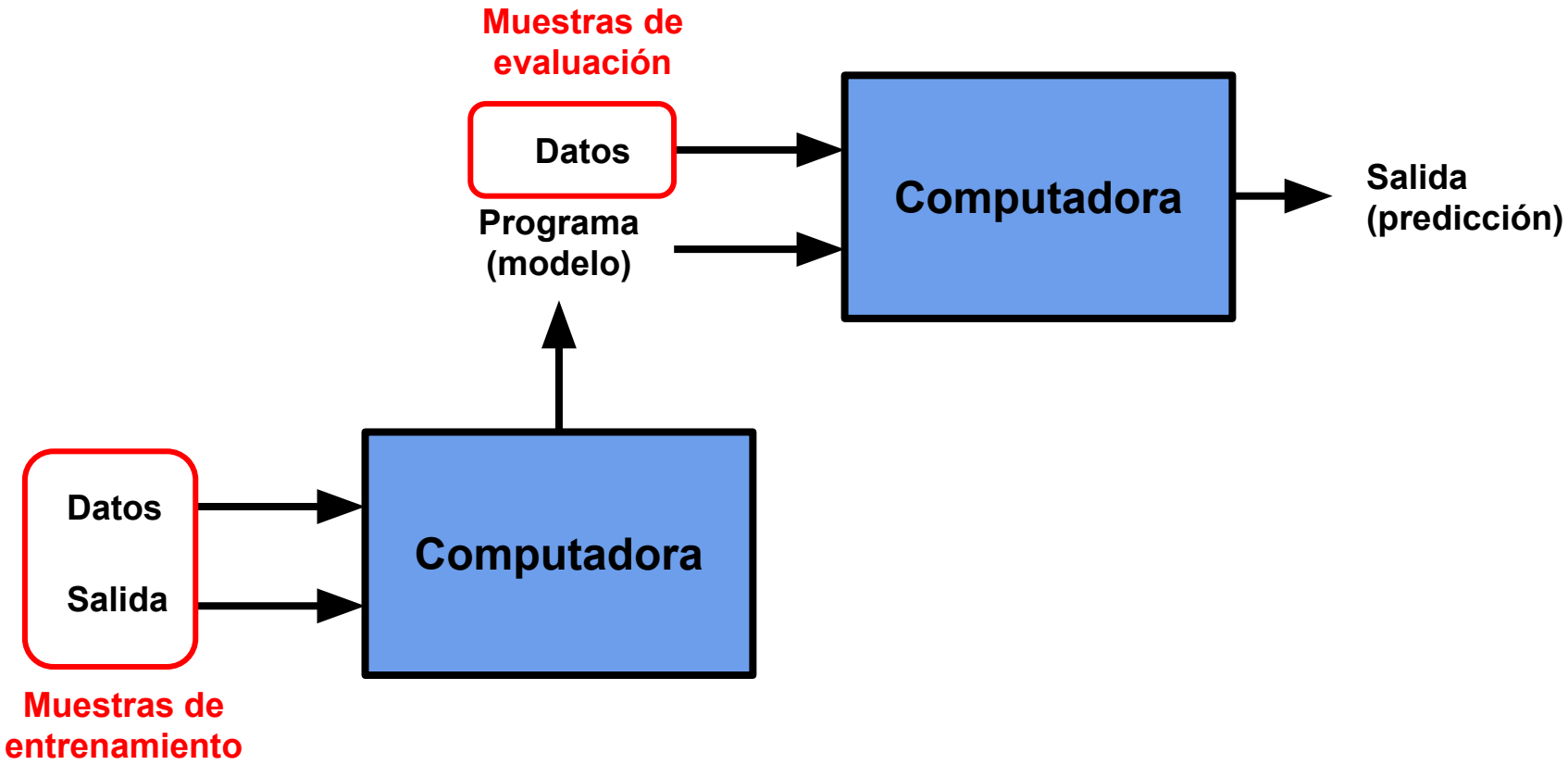
#1. Introducción al aprendizaje automático

Programación tradicional



Aprendizaje automático

Aprendizaje automático: entrenamiento vs. evaluación



Sobre "aprendizaje"

- Se puede ver como la utilización directa o indirecta de la experiencia para aproximar una determinada función.

- La aproximación de dicha función corresponde a una búsqueda en un espacio de hipótesis (espacio de funciones) por aquella que mejor prediga el comportamiento de **datos nuevos**.

- Distintos métodos de aprendizaje automático asumen distintos espacios de hipótesis o utilizan distintas estrategias de búsqueda.

Tipos de problemas

- **Aprendizaje supervisado (inductivo)**
  Datos de entrenamiento + salida esperada

- **Aprendizaje no supervisado**
  Datos de entrenamiento (sin salida esperada)

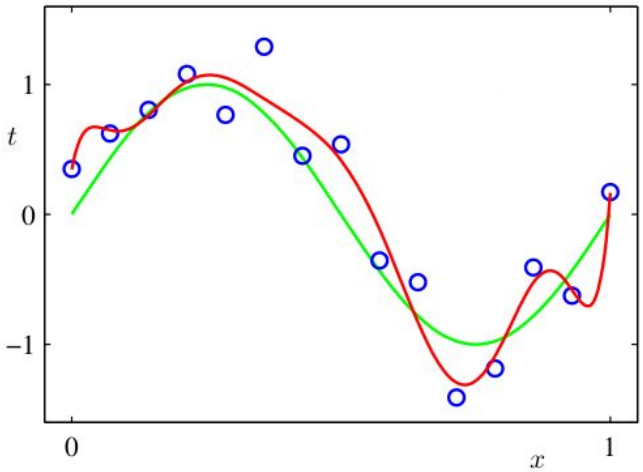
- **Aprendizaje semi-supervisado**
  Datos de entrenamiento + **pocas** salida esperadas

- **Aprendizaje auto-supervisado**
  Datos de entrenamiento auto generados (*tareas pretexto*)

- **Aprendizaje por refuerzo**
  "Recompensas" por secuencias de acciones

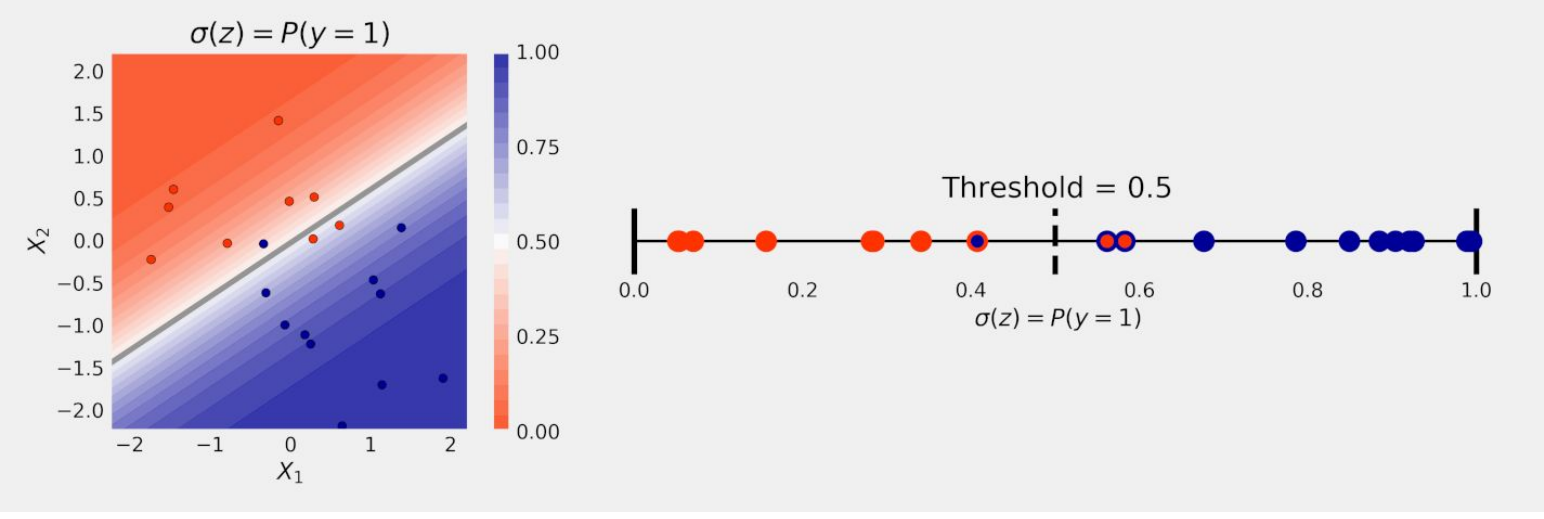
Aprendizaje supervisado: regresión

- Dados $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$
- Aprender una $f(x)$ que permita predecir y a partir de x
    - Si y está en $\mathbb{R}^n$ → **regresión**



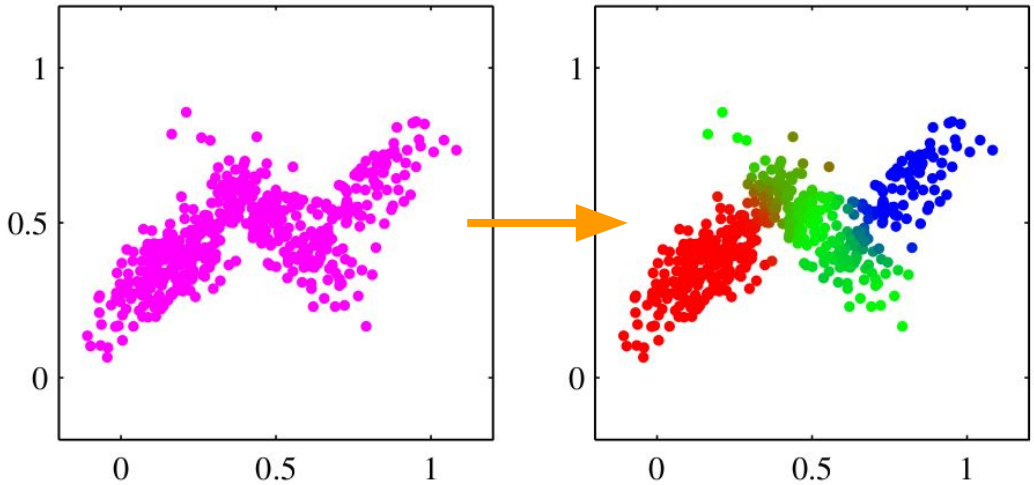
Aprendizaje supervisado: clasificación

- Dados $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$
- Aprender una $f(x)$ que permita predecir y a partir de x
    - Si y es categórica → **clasificación**



Aprendizaje no supervisado

- Dados $x_1, x_2, ..., x_n$
- Aprender la estructura interna de los datos
    - p.ej. *clustering*

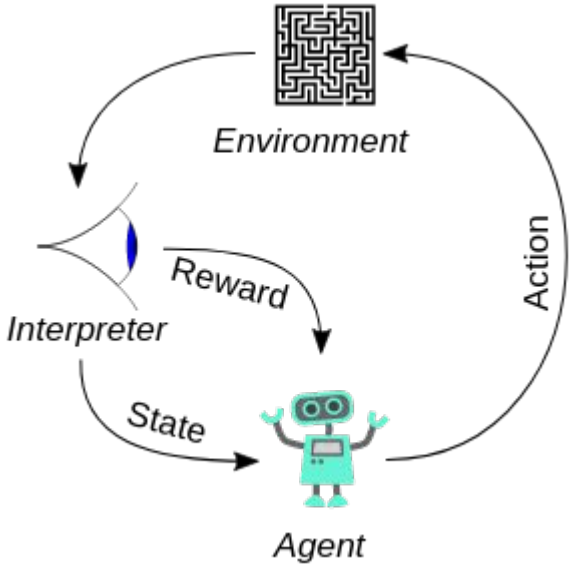


Aprendizaje auto supervisado

- Dados $x_1, x_2, ..., x_n$
- Utilizar estructura interna para generar *tareas pretexto*
    - p.ej.: predecir siguiente elemento en una secuencia
- (pre)entrenar para aprender a representar bien los datos
- Adaptar a la tarea de interés (regresión, clasificación, …)

Aprendizaje por refuerzo

- Dada una secuencia de estados y acciones con recompensa (*reward*), generar una política (*policy*) (secuencia de acciones) que nos indique qué hacer ante un determinado estado



Aprendizaje (supervisado)

Funciones $\mathcal{F}$

$f : \mathcal{X} \to \mathcal{Y}$

Datos de entrenamiento

$\{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}$

Aprendizaje

find $\hat{f} \in \mathcal{F}$
s.t. $y_i \approx \hat{f}(x_i)$

Predicción

$y = \hat{f}(x)$

x

Dato nuevo

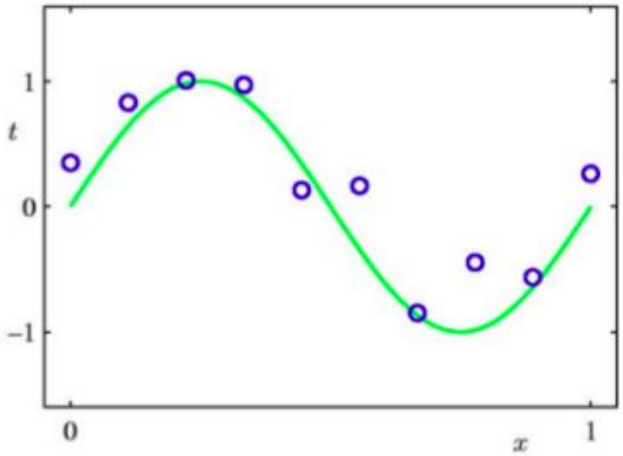
Regresión

Regresión

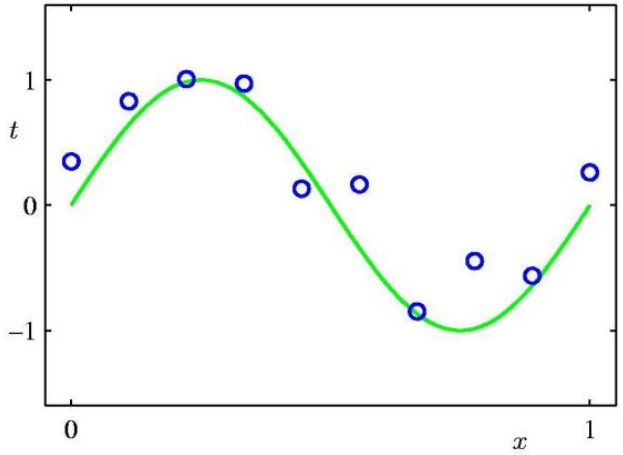
- Disponemos de N pares de entrenamiento (observaciones)

$$\{(x_i, y_i)\}_{i=1}^{N} = \{(x_1, y_1), \cdots, (x_N, y_N)\}$$

- El problema de regresión consiste en estimar *f(x)* a partir de estos datos

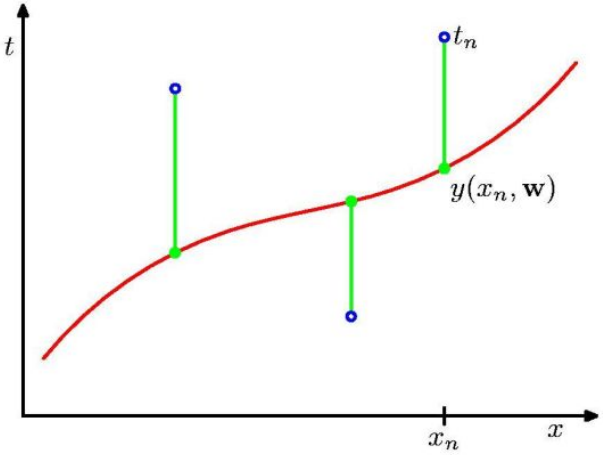


Regresión polinomial



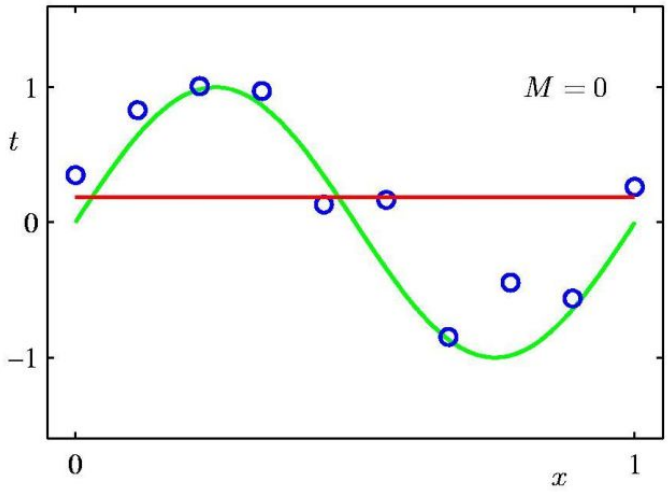
- En verde se ilustra la función "verdadera" (inaccesible)
- Las muestras son uniformes en x y poseen ruido en y
- Modelo predictivo: polinomio de orden M

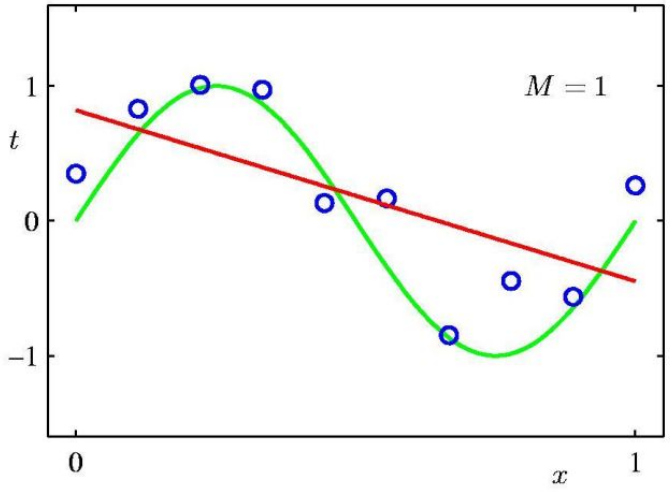
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

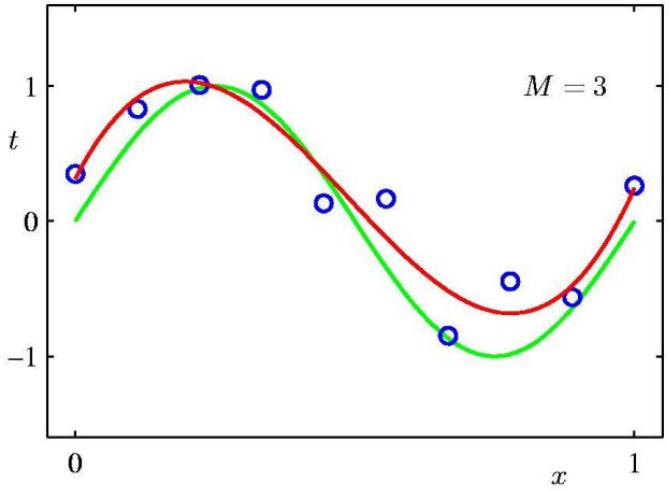


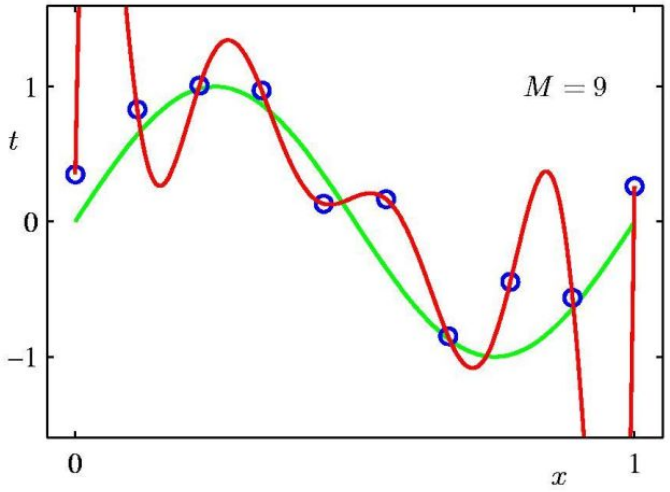
- Utilizaremos una **<u>función de costo</u>** (error cuadrático) para medir el error en la predicción de y mediante $y(x; \mathbf{w})$

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$


M = 0
t
x
1
0
−1
0
1

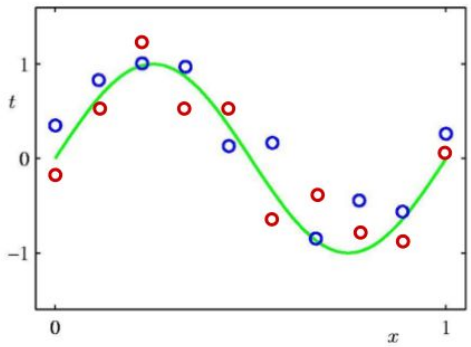

M = 1
t
x
1
0
−1
0
1

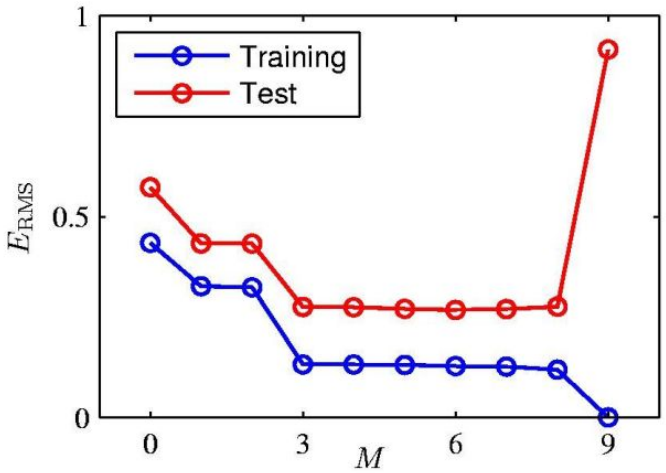

M = 3
t
x
1
0
−1
0
1


M = 9
t
x
1
0
−1
0
1

Sobreajuste (*overfitting*)

- Datos de test: otra muestra de los misma función subyacente
- El error de entrenamiento se hace cero, pero el de test crece con M





Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$

Bondad de ajuste vs. complejidad de modelo

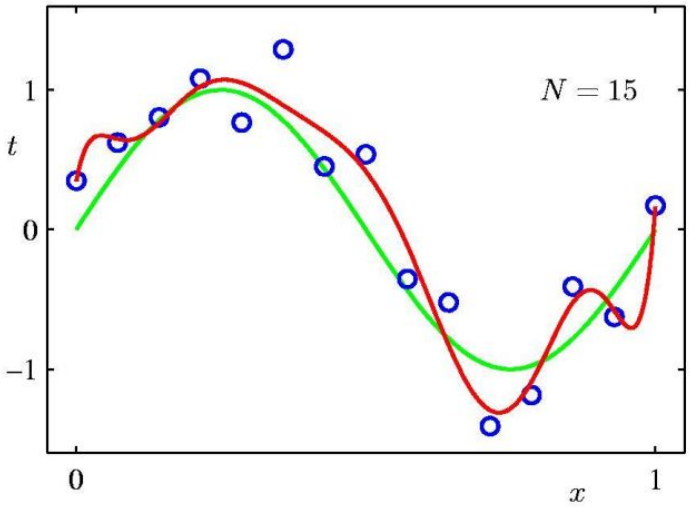
- Si el modelo tiene tantos grados de libertad como los presentes en los datos de entrenamiento, puede ajustarlos perfectamente

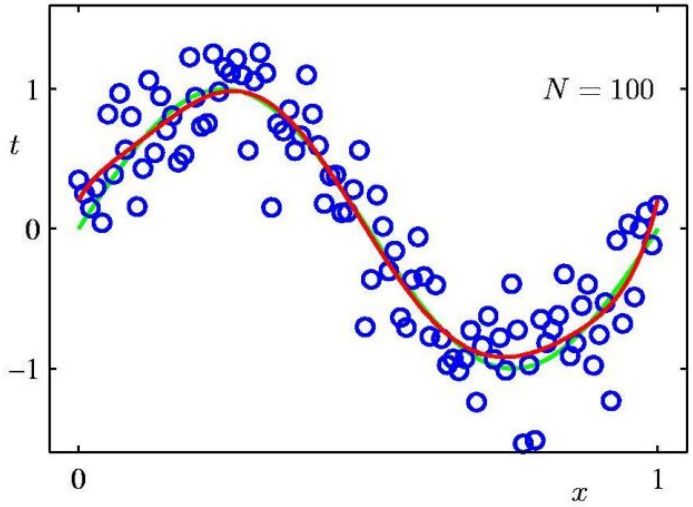
- El objetivo en aprendizaje automático no es el ajuste perfecto, sino la **generalización** a conjuntos nuevos (no vistos en entrenamiento)

- Podemos decir que un modelo generaliza, si puede explicar datos nuevos empleando una complejidad acotada

Prevenir el sobreajuste (I)

- Agregar más datos (más que la "complejidad" del modelo)





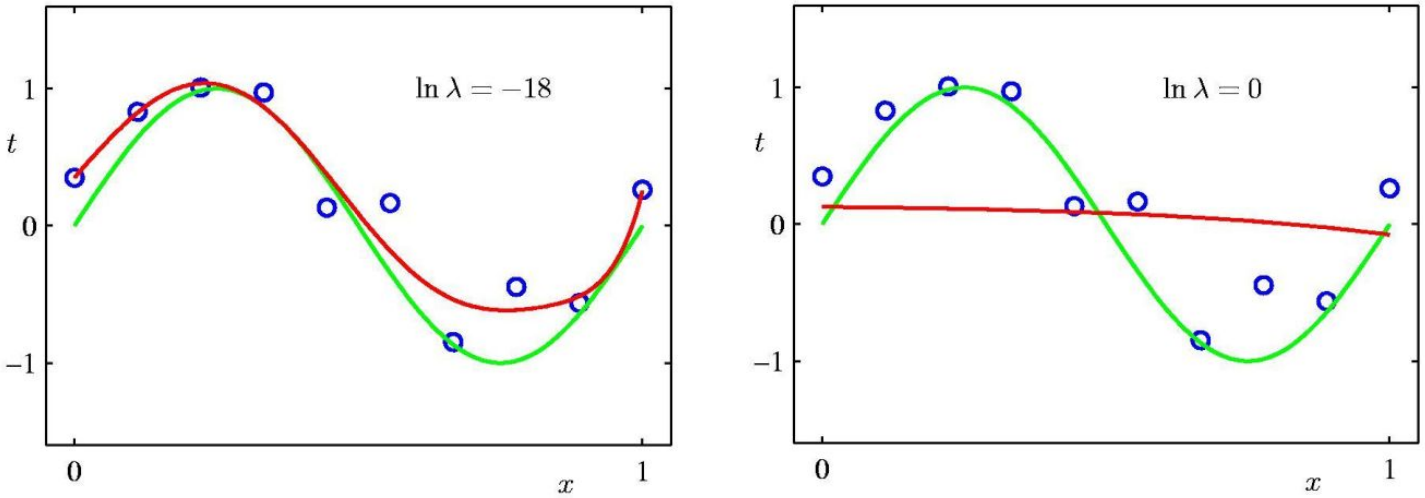
Prevenir el sobreajuste (II)

	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Prevenir el sobreajuste (II)

- Regularización: penalizar valores grandes de los coeficientes

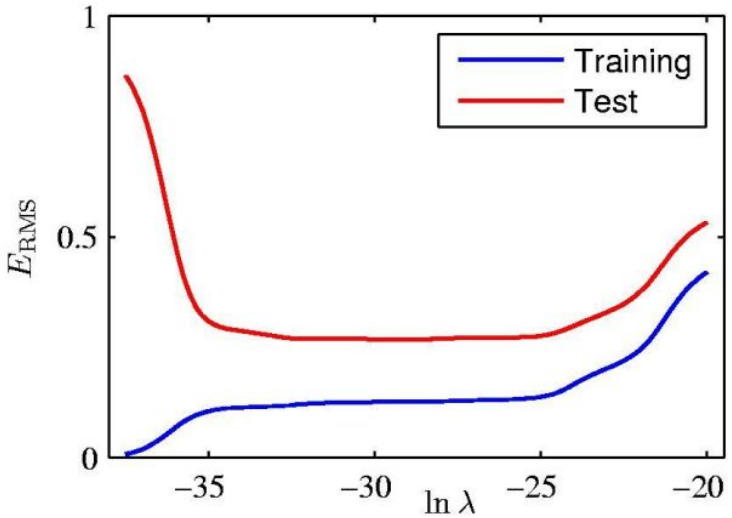
$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$



Prevenir el sobreajuste (II)

- Regularización: penalizar valores grandes de los coeficientes

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$



Término de regularización (*ridge*)

λ = hiperparámetro

Prevenir el sobreajuste (II)

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Regresión polinomial como regresión lineal

$$x \mapsto \mathbf{z} = \begin{pmatrix} x \\ x^2 \\ \vdots \\ x^M \end{pmatrix} \qquad \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{pmatrix}$$

$$\begin{aligned} y(x; \mathbf{w}) &= w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M \\ &= w_0 + \sum_{j=1}^{M} w_j x^j = w_0 + \sum_{j=1}^{M} w_j z_j \\ &= w_0 + \mathbf{w}^T \mathbf{z} \end{aligned}$$

$$\mathbf{a}^T \mathbf{b} \equiv \langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{D} a_i b_i$$

Regresión polinomial como regresión lineal

$$x \mapsto \mathbf{z} = \begin{pmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^M \end{pmatrix} \qquad \mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_M \end{pmatrix}$$

$$\begin{aligned} y(x; \mathbf{w}) &= w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M \\ &= \sum_{j=0}^{M} w_j x^j \\ &= \mathbf{w}^T \mathbf{z} \end{aligned}$$

Regresión lineal: solución de mínimos cuadrados

- Dataset: $\{(x_1, t_1), \cdots, (x_N, t_N)\} \mapsto \{(\mathbf{z}_1, t_1), \cdots, (\mathbf{z}_N, t_N)\}$
- Función de costo: $E(\mathbf{W}) = \frac{1}{2}\sum_{i=1}^{N}\left(y(x_i; \mathbf{w}) - t_i\right)^2 = \frac{1}{2}\sum_{i=1}^{N}\left(\mathbf{w}^T\mathbf{z}_i - t_i\right)^2$

$$\mathbf{Z} = \begin{pmatrix} — & \mathbf{z}_1^T & — \\ & \vdots & \\ — & \mathbf{z}_N^T & — \end{pmatrix} \in \mathbb{R}^{N\times M} \qquad \mathbf{y} = \begin{pmatrix} t_1 \\ \vdots \\ t_N \end{pmatrix} \in \mathbb{R}^{N\times 1} \qquad \|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w}$$

$$E(\mathbf{w}) = (\mathbf{Z}\mathbf{w} - \mathbf{y})^T(\mathbf{Z}\mathbf{w} - \mathbf{y}) \quad \rightarrow \quad \mathbf{w}^* = \left(\mathbf{Z}^T\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{y}$$

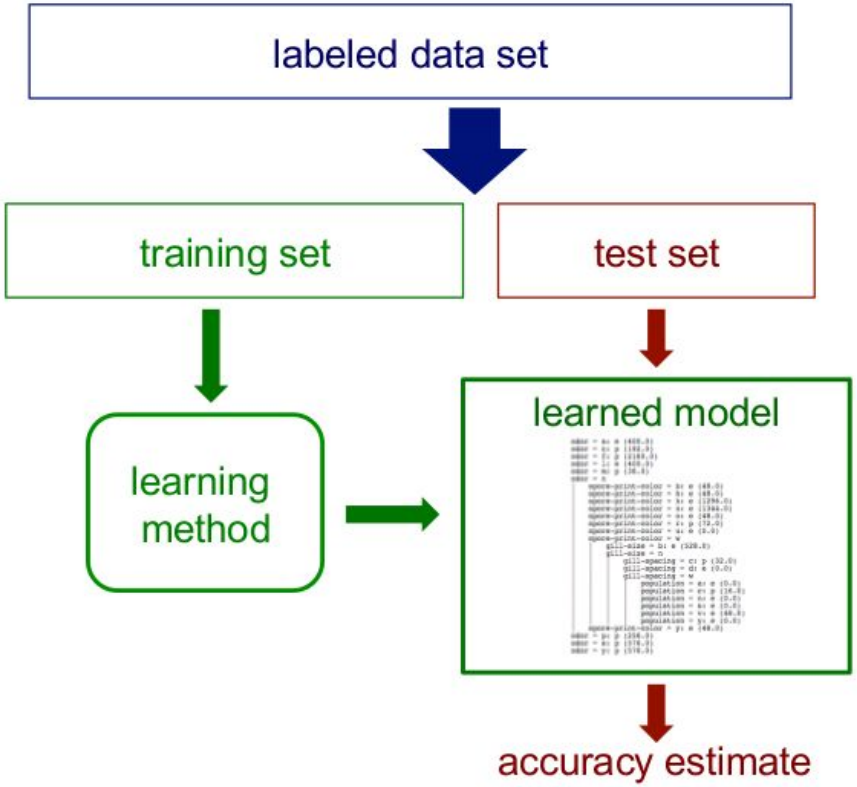
$$E(\mathbf{w}) = (\mathbf{Z}\mathbf{w} - \mathbf{y})^T(\mathbf{Z}\mathbf{w} - \mathbf{y}) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \quad \rightarrow \quad \mathbf{w}^* = \left(\mathbf{Z}^T\mathbf{Z} + \lambda\mathbf{I}\right)^{-1}\mathbf{Z}^T\mathbf{y}$$

Petersen & Pedersen, The matrix cookbook (2012)

Técnicas de validación

Conjunto de test

Cómo obtener una estimación insesgada de la *performance* del modelo?



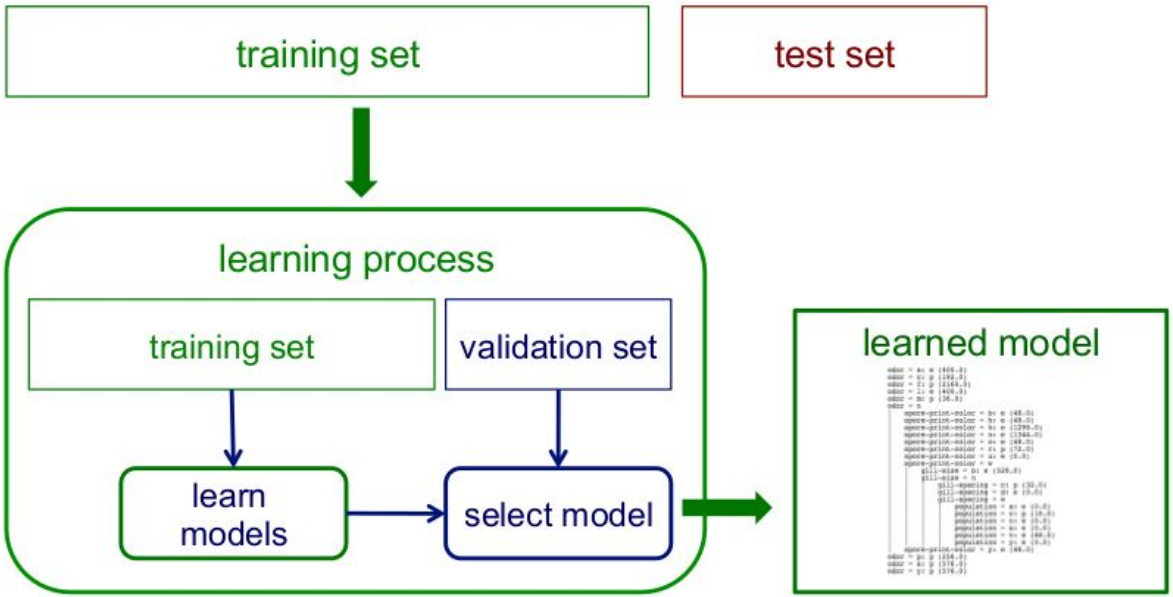
Conjunto de test

Cómo obtener una estimación insesgada de la *performance* del modelo?

- Durante el aprendizaje el modelo no debe acceder bajo ningún motivo a datos del conjunto de test
  - En métodos transductivos se puede permitir acceso a los datos crudos (x) pero no a las anotaciones (y)
- Si las anotaciones del conjunto de test influencian de **cualquier manera** el aprendizaje, las estimaciones de performance estarán sesgadas.

Conjunto(s) de validación

Cómo obtener una estimación insesgada de la *performance* del modelo **durante el entrenamiento**? (ajuste de hiperparámetros)

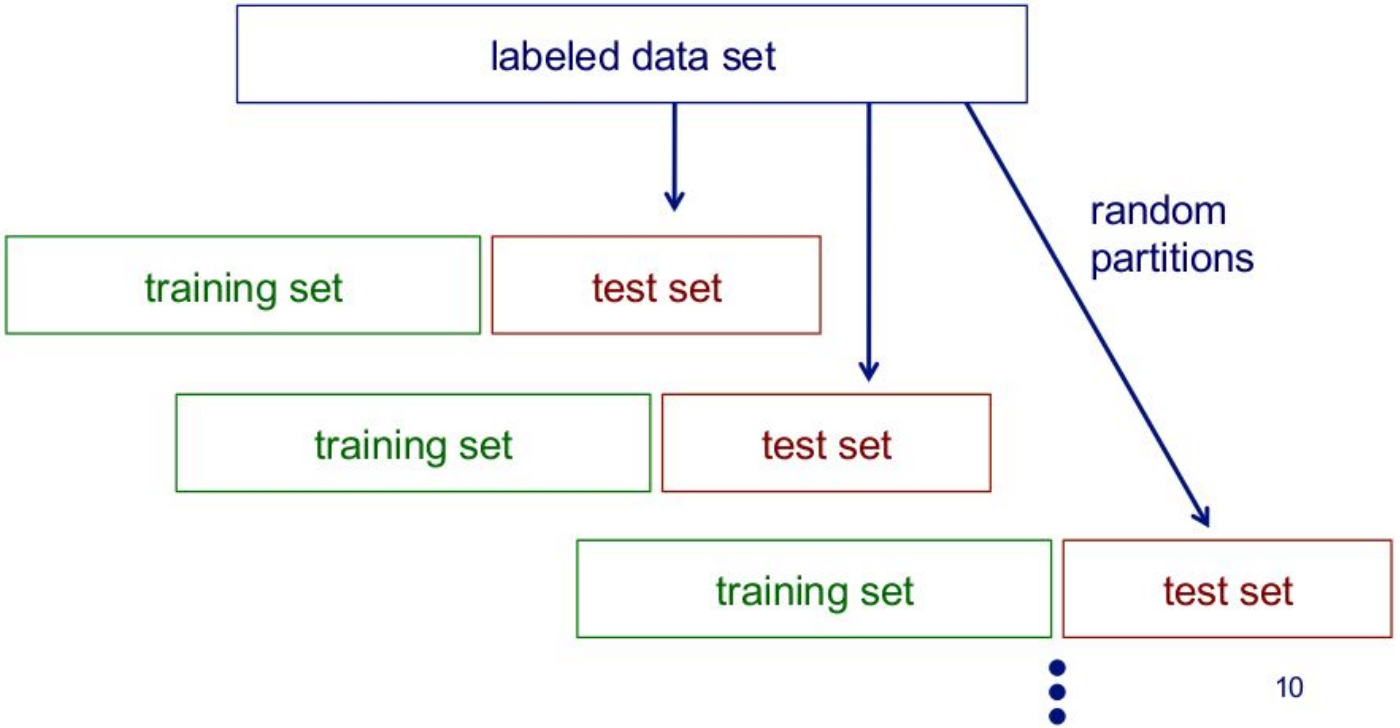


Limitación de usar solo un conjunto de train/test

- Los datos pueden ser insuficientes para crear conjuntos de entrenamiento y test lo suficientemente grandes
  - Un conjunto de test grande nos da una mejor medida de la performance del modelo (menor varianza)
  - Un conjunto de entrenamiento grande es más representativo del universo de entradas posibles
- Un solo conjunto de entrenamiento no nos da información sobre la sensibilidad del modelo ante cambios en los conjuntos de entrada

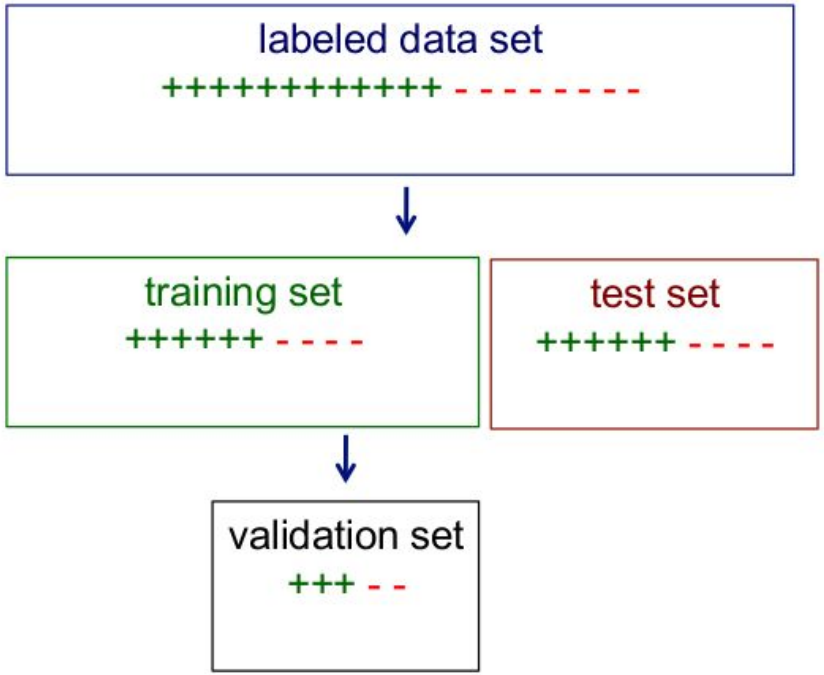
Remuestreo aleatorio

- podemos abordar el segundo punto mediante remuestreo



Muestreo estratificado

- En problemas de clasificación, podemos requerir que las proporciones de clases se mantengan en cada subconjunto



Validación cruzada (*cross-validation*)

- podemos considerar conjuntos de validación independientes y obtener una estimación respecto de la sensibilidad