Análisis automático de datos para las ciencias biomédicas, medioambientales y agroalimentarias
(Transversal Másteres Universitarios)

Tema 8. Aprendizaje supervisado: Regresión lineal y regresión logística. Sobreaprendizaje.

Juan Carlos Fernández Caballero
Pedro Antonio Gutiérrez Peña
Departamento de Informática y Análisis Numérico
Universidad de Córdoba
Grupo de investigación AYRNA

1. Introducción.
2. Modelo de regresión lineal y polinómica basadas en el método o algoritmo de los mínimos cuadrados para regresión.
3. Modelo de regresión logística para clasificación.
4. Concepto de sobreaprendizaje.

Terminología (recordatorio)

tumor size	*texture*	*perimeter*	…	*outcome*
18.02	27.60	117.5	…	N
17.99	10.38	122.8	…	N
20.29	14.34	135.1	…	R

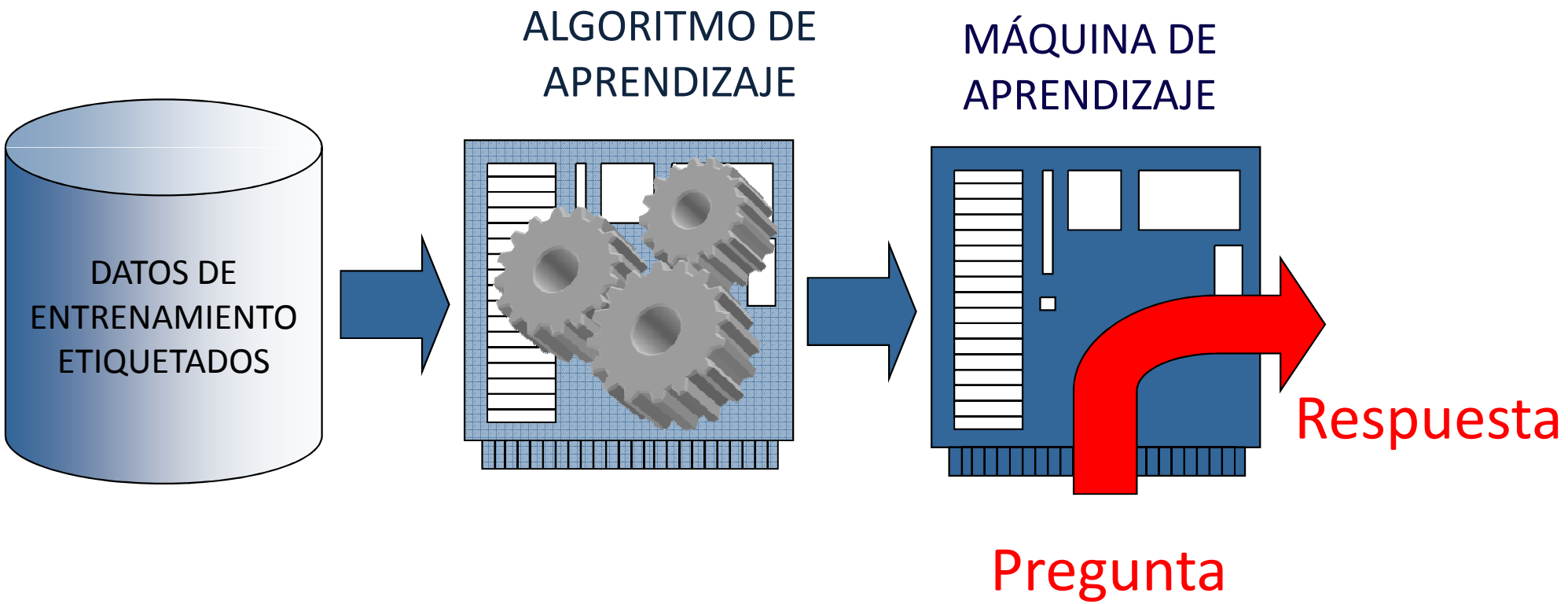
- **Variables explicativas**, variables de entrada, características, atributos o variables independientes (*tumor size*, *texture*, *perimeter*…)
- **Variable a predecir**, variable objetivo, variable dependiente (*outcome*).
- **Filas**: instancias, patrones, ejemplos…
- **Tipo de problema supervisado:**
  - Clasificación: Si la variable a predecir es de tipo categórico, cualitativo, nominal.
  - Regresión: Si la variable a predecir es de tipo continuo, cuantitativo (por ejemplo, una variable que reflejase la esperanza de vida del paciente).

Aprendizaje supervisado

- Tenemos etiquetas, conjunto de ejemplos de la forma **x** = (x_1, x_2, …, x_n, y)
  - Implica la existencia de un maestro o experto que conoce las respuestas reales o etiquetas.
- ¿Qué es lo que buscamos?
  - Función $f : X_1 \times X_2 \times \ldots \times X_n \rightarrow Y$, $x_j \in X_j$, $1 \leq j \leq n$, que mapea las variables de entrada en la variable de salida.
  - Es decir, buscamos un f(x) que sea un buen **predictor para la variable Y**, utilizando para ello una **función de error**.
  - Se intenta **minimizar** una función de error $L(f(x_i),y_i)$ que es el **sumatorio para cada patrón** de lo mala que ha sido una predicción.
  - Ej. Función de error cuadrática (MSE): $L(f(x_i),y_i) = (f(x_i)-y_i)^2/N$, donde $f(x_i)$ es la predicción del modelo, y_i el valor real o etiqueta y N el número total de patrones.

Problema de aprendizaje supervisado

Esquema genérico del proceso de *Machine Learning* Supervisado



Problema de aprendizaje supervisado

- Pasos a dar:
  1. Construir los ejemplos de entrenamiento:
     - Recolectar datos (p.ej. encuestas, datos de pacientes, datos de viviendas, etc).
     - Extraer características a partir de los datos que sean relevantes y no aporten "ruido".
  2. Elegir un modelo, es decir, una representación generalista para f(x), normalmente definimos un espacio de hipótesis o un conjunto de modelos (regresión lineal, árbol, etc).
  3. Elegir una función de error que permita llegar a la mejor hipótesis (CCR, MSE o RMSE por ejemplo).
  4. Elegir un algoritmo de entrenamiento (*Least Mean Squares* para regresión, algoritmo *backpropagation* en redes neuronales, algoritmo C4.5 en árboles, etc) que permite ajustar las hipótesis en base a la función de error.
  5. Entrenar (*train*).
  6. Generalizar (*test*).

Ejemplo: conjunto de datos de cáncer

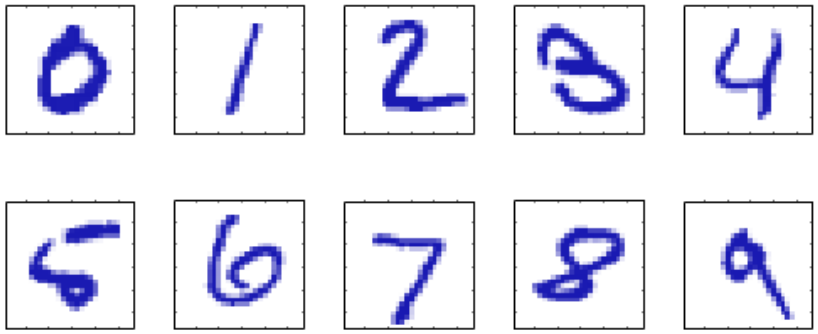
- *n* variables de entrada por tumor y *N* pacientes:

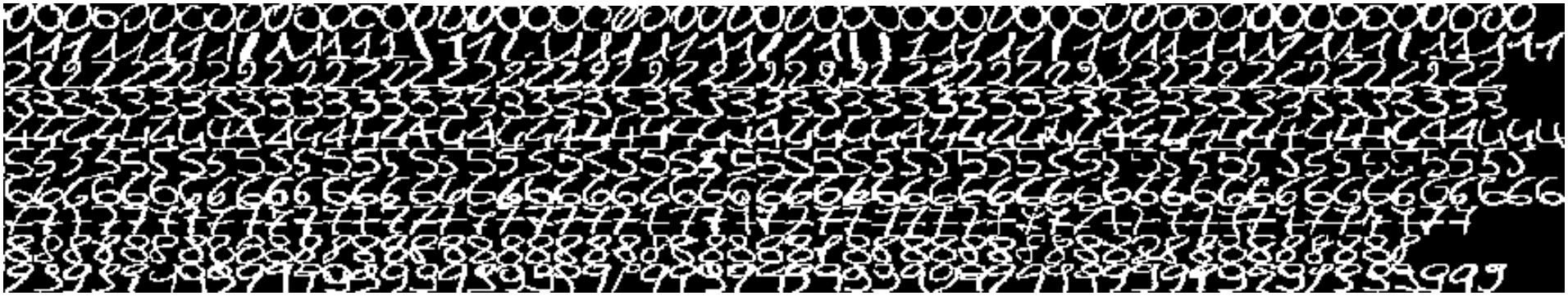
tumor size	*texture*	*perimeter*	…	*outcome*
18.02	27.60	117.5	…	N
17.99	10.38	122.8	…	N
20.29	14.34	135.1	…	R

- Variable de salida:
  - *Outcome*:
    - R: recurrencia (reaparición) del tumor después de la quimioterapia.
    - N: el tumor no vuelve a aparecer después de la quimioterapia.
  - Dado un nuevo paciente (que no está en nuestra tabla), queremos predecir si el tumor va a reincidir (*N* o *R*).

tumor size	*texture*	*perimeter*	…	*outcome*
15.00	9.00	110.0	…	?¿?¿

Ejemplo: reconocimiento de dígitos escritos (*handwritten digits recognition*)





Muestra de entrenamiento

Ejemplo: reconocimiento de dígitos escritos (*handwritten digits recognition*)

- Formulación matemática del problema:
  - Representar cada imagen como un vector.
  - Pixelar las imágenes en 28x28=758 píxeles.
  - Cada pixel puede tener un valor que significa un nivel de gris.
  - Obtener un clasificador *f(x)*

$$f : \mathbf{x} \rightarrow \{0,1,2,3,4,5,6,7,8,9\}$$

Formulación matemática del problema:

- Suponemos que ***y*** es una función lineal con respecto a ***x,*** con lo que una hipótesis ***h*** se puede definir como:

$$h_{\mathbf{w}}(x)=w_0+w_1x_1+w_2x_2(\ldots)\approx y$$

  w_0, w_1 y w_2 son los parámetros o pesos (también se denotan como β_0, β_1 y β_2)

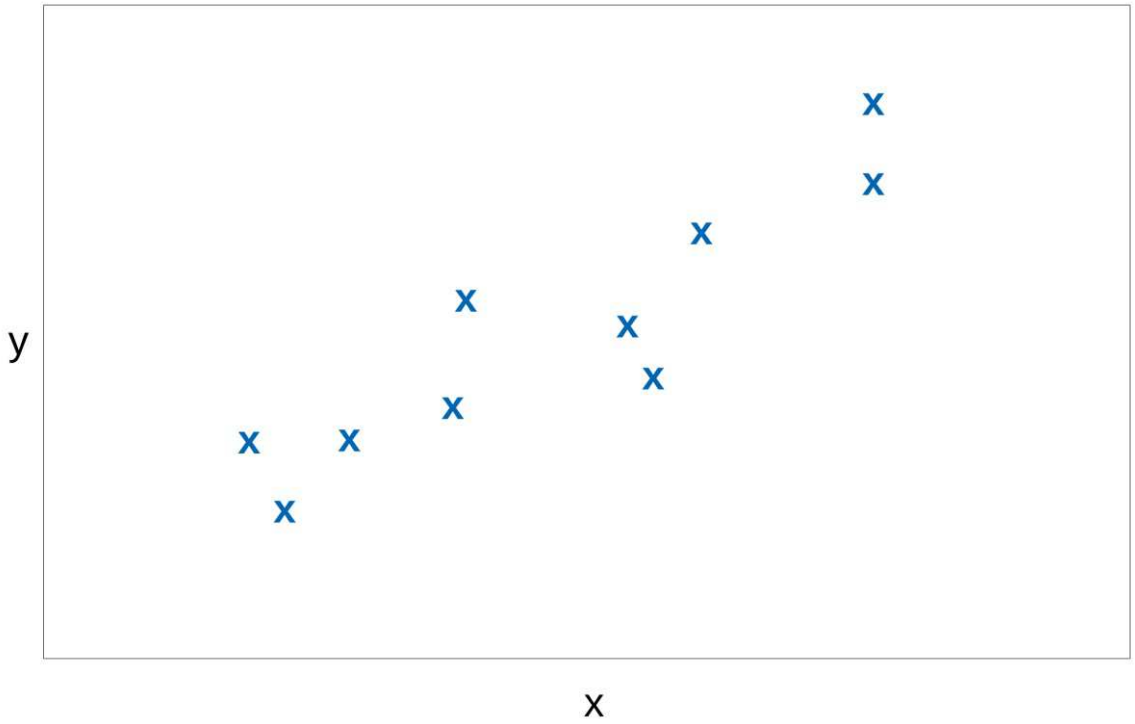
- Para simplificar, solemos añadir un atributo constante x_0=1 (también llamado *bias* o coordenada en el origen):

$$h_{\mathbf{w}}(x)=\sum_{i=0}^{n} w_i x_i = \mathbf{w}\mathbf{x}$$

  donde **w** y **x** son vectores de tamaño n+1, siendo n el número de variables explicativas o atributos.

- Queremos encontrar el vector de coeficientes $\boldsymbol{w} = (w_0, \ldots, w_n)$ tal que $h_{\boldsymbol{w}}(\boldsymbol{x}_i) \approx y_i$
- Deberíamos usar una función de error que minimice la divergencia o error entre las predicciones del modelo y los valores reales.

Regresión lineal. Ejemplo 10 puntos



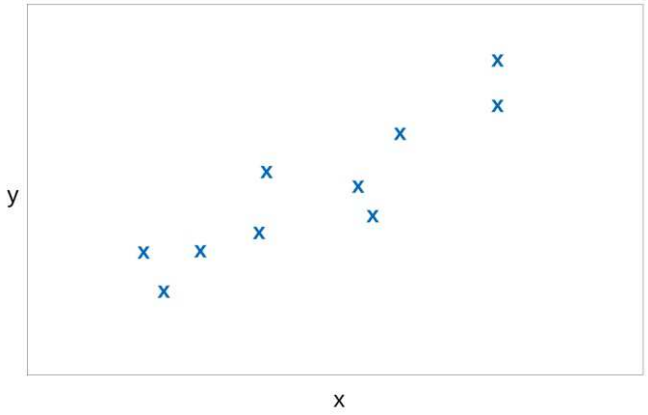
x	y
0.86	2.49
0.09	0.83
-0.85	-0.25
0.87	3.10
-0.44	0.87
-0.43	0.02
-1.10	-0.12
0.40	1.81
-0.96	-0.83
0.17	0.43

- 10 ejemplos de entrenamiento:

  Estimar un $h_w(x)$ a partir de los ejemplos etiquetados (x_i, y_i), para $i = 1, \ldots, 10$.

Regresión lineal. Ejemplo 10 puntos

Para nuestro ejemplo, podríamos usar un algoritmo de aprendizaje para regresión que encuentre los $\mathbf{w_i}$, como por ejemplo el método *Least Mean Squares (LMS)* o método de los mínimos cuadrados.
https://es.wikipedia.org/wiki/M%C3%ADnimos_cuadrados



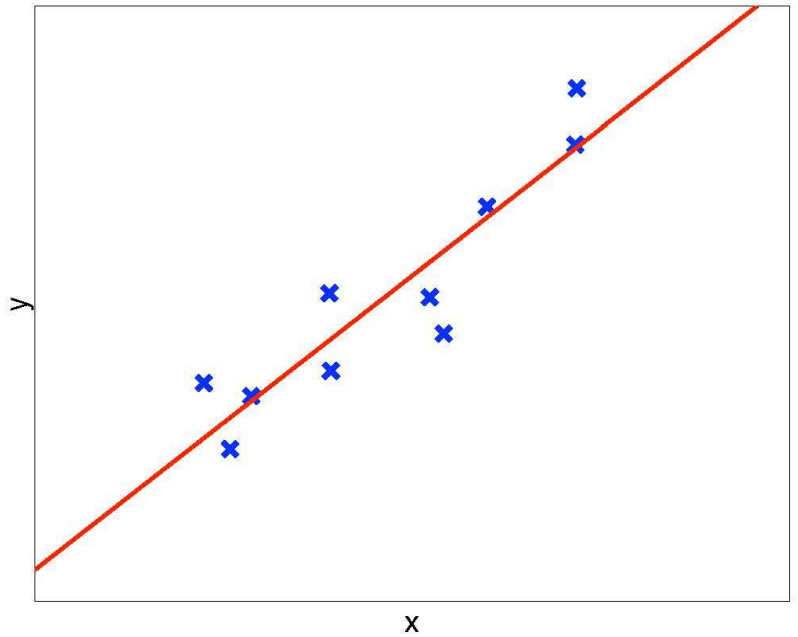
$$X = \begin{bmatrix} 0.86 & 1 \\ 0.09 & 1 \\ -0.85 & 1 \\ 0.87 & 1 \\ -0.44 & 1 \\ -0.43 & 1 \\ -1.10 & 1 \\ 0.40 & 1 \\ -0.96 & 1 \\ 0.17 & 1 \end{bmatrix} \qquad Y = \begin{bmatrix} 2.49 \\ 0.83 \\ -0.25 \\ 3.10 \\ 0.87 \\ 0.02 \\ -0.12 \\ 1.81 \\ -0.83 \\ 0.43 \end{bmatrix}$$

Regresión lineal. Ejemplo 10 puntos

A partir del procedimiento de LMS (se obvian los pasos) se obtendría el siguiente vector de pesos para un modelo de regresión lineal:

$$\mathbf{w} = (X^T X)^{-1} X^T Y = \begin{bmatrix} 4.95 & -1.39 \\ -1.39 & 10 \end{bmatrix}^{-1} \begin{bmatrix} 6.49 \\ 8.34 \end{bmatrix} = \begin{bmatrix} 1.60 \\ 1.05 \end{bmatrix}$$

Recta de regresión con los $\mathbf{w_i}$ $= 1.60x + 1.05.$
o $\boldsymbol{\beta_i}$ (según nomenclatura)



Regresión lineal en Weka

- Regresión Lineal Simple y Regresión Lineal (múltiple) en **Weka**:

- **Regresión Lineal Simple:** Pestaña ***Classify.***
  classifiers.functions.SimpleLinearRegression
  - El modelo selecciona **una única variable independiente** (atributo) para predecir la dependiente.
  - Solo trabaja con **atributos numéricos**. Necesidad de filtros de preprocesado ante atributos nominales
    filters→unsupervised→attributes→ NominalToBinary

- **Regresión Lineal:** Pestaña ***Classify.***
  classifiers.functions.LinearRegression
  - El modelo selecciona **varias variables independientes** (atributos) para predecir la dependiente.

Regresión Lineal Simple y Regresión Lineal Multiple

- **RL. Simple:** Una **única** variable independiente. $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i + \varepsilon$
- **RL. Múltiple:** Existen **N** variables independientes.
  $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \cdots + \hat{\beta}_n x_{ni} + \varepsilon$

Epsilón es una variable aleatoria, es decir, un error o perturbación que se supone que no se ha tenido en cuenta para explicar la variable *y*.

Regresión polinómica

- Si queremos aproximar un polinomio de **grado *d*** a los datos mediante regresión tendríamos la siguiente hipótesis ***h*** (para solo una variable explicativa más la x_0):

$$h_{\mathbf{w}}(x)=w_0+w_1x^1+w_2x^2\,(\ldots)\approx y$$

- Dado un conjunto de datos: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
- **Y** se queda igual y transformamos **X** en una nueva **X**

$$X = \begin{bmatrix} x_1^d & \ldots & x_1^2 & x_1 & 1 \\ x_2^d & \ldots & x_2^2 & x_2 & 1 \\ \vdots & & \vdots & \vdots & \vdots \\ x_m^d & \ldots & x_m^2 & x_m & 1 \end{bmatrix}$$

$$X = \begin{bmatrix} 0.75 & 0.86 & 1 \\ 0.01 & 0.09 & 1 \\ 0.73 & -0.85 & 1 \\ 0.76 & 0.87 & 1 \\ 0.19 & -0.44 & 1 \\ 0.18 & -0.43 & 1 \\ 1.22 & -1.10 & 1 \\ 0.16 & 0.40 & 1 \\ 0.93 & -0.96 & 1 \\ 0.03 & 0.17 & 1 \end{bmatrix} \quad Y = \begin{bmatrix} 2.49 \\ 0.83 \\ -0.25 \\ 3.10 \\ 0.87 \\ 0.02 \\ -0.12 \\ 1.81 \\ -0.83 \\ 0.43 \end{bmatrix}$$

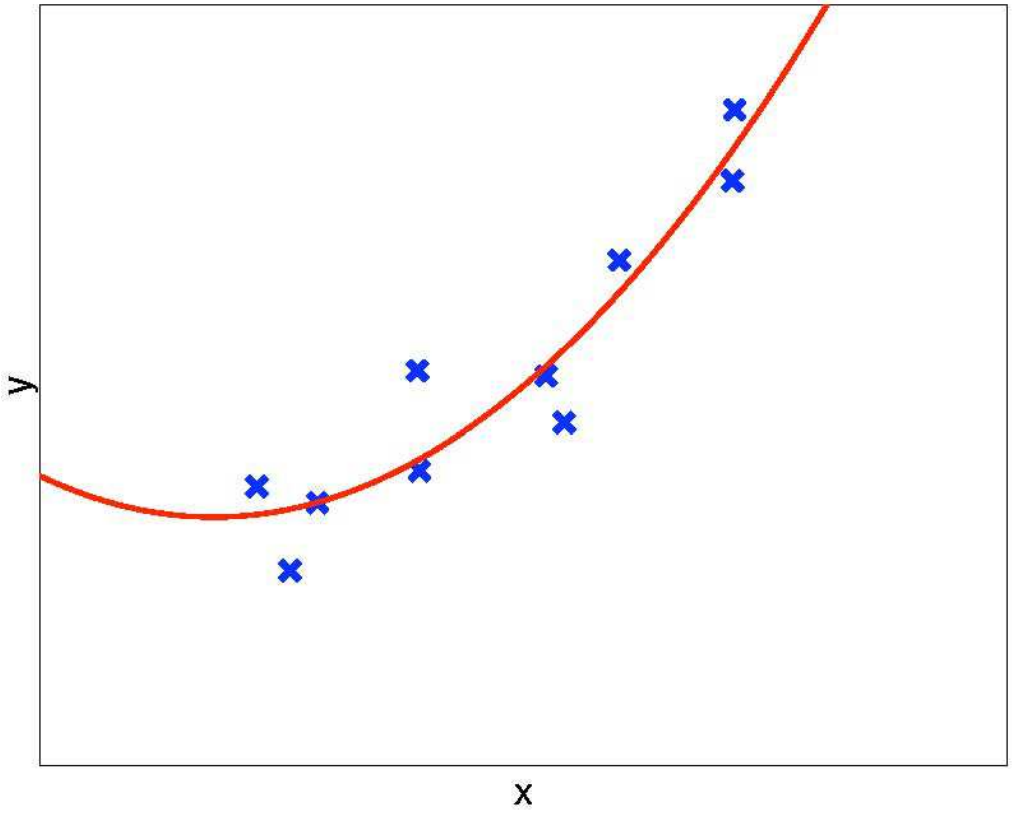
De nuevo resolvemos mediante LMS.

- A esto se le llama regresión polinómica.

Regresión polinómica: caso cuadrático

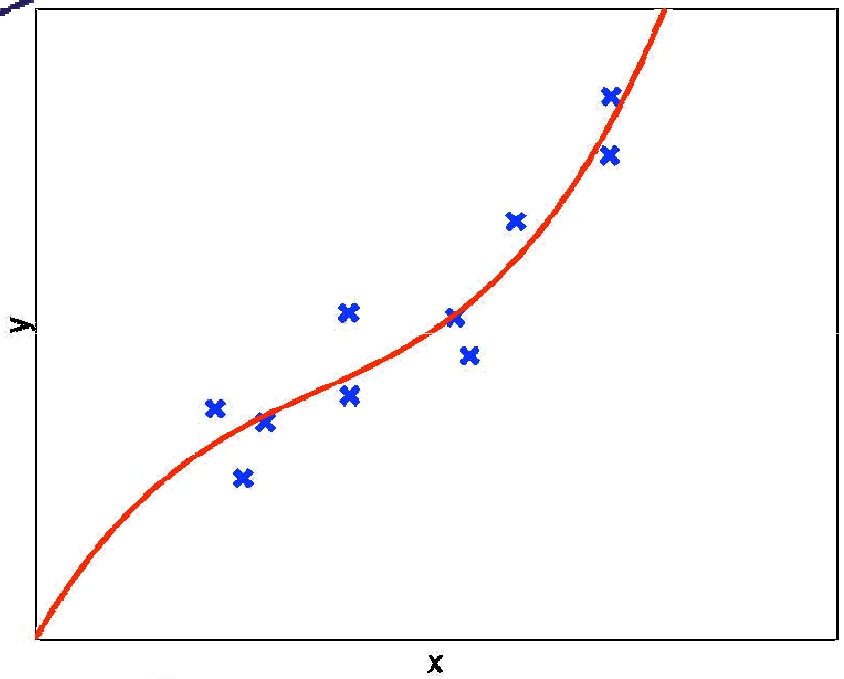
$$\mathbf{w} = (X^T X)^{-1} X^T Y = \begin{bmatrix} 4.11 & -1.64 & 4.95 \\ -1.64 & 4.95 & -1.39 \\ 4.95 & -1.39 & 10 \end{bmatrix}^{-1} \begin{bmatrix} 3.60 \\ 6.49 \\ 8.34 \end{bmatrix} = \begin{bmatrix} 0.68 \\ 1.74 \\ 0.73 \end{bmatrix}$$

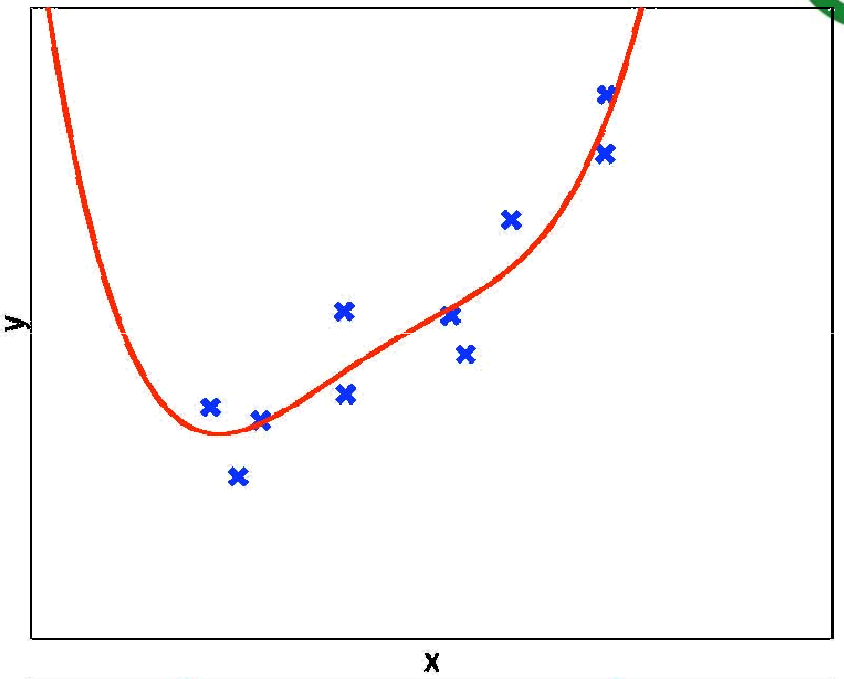
El mejor polinomio de grado 2 es $y = 0.68x^2 + 1.74x + 0.73.$

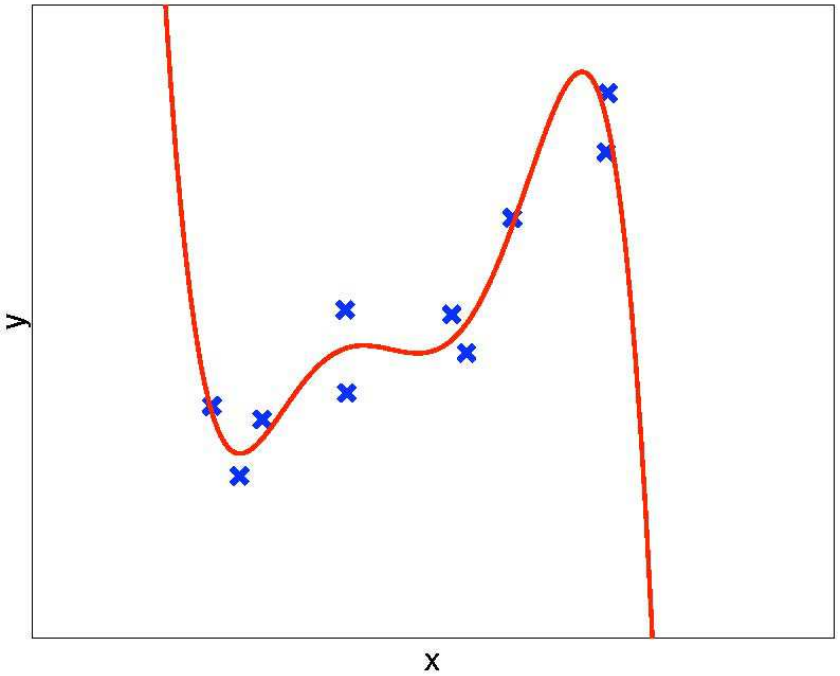


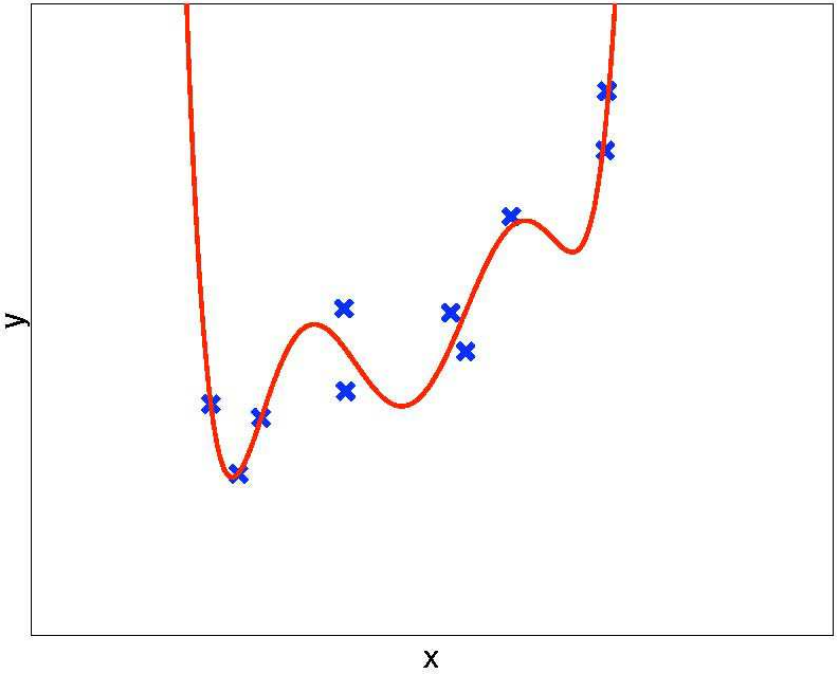
¿Podemos mejorar el ajuste?

Regresión polinómica: grado 3, 6…

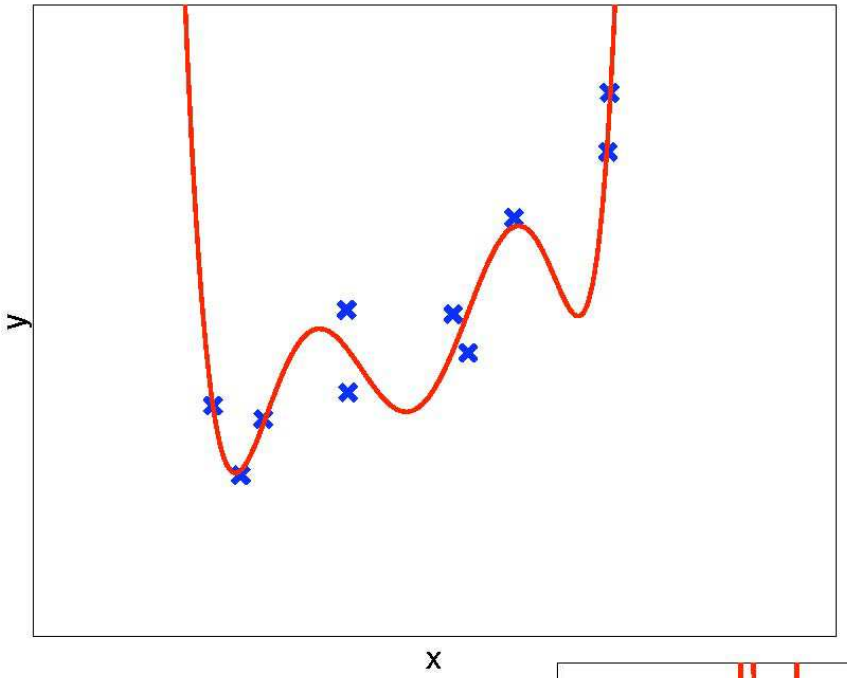


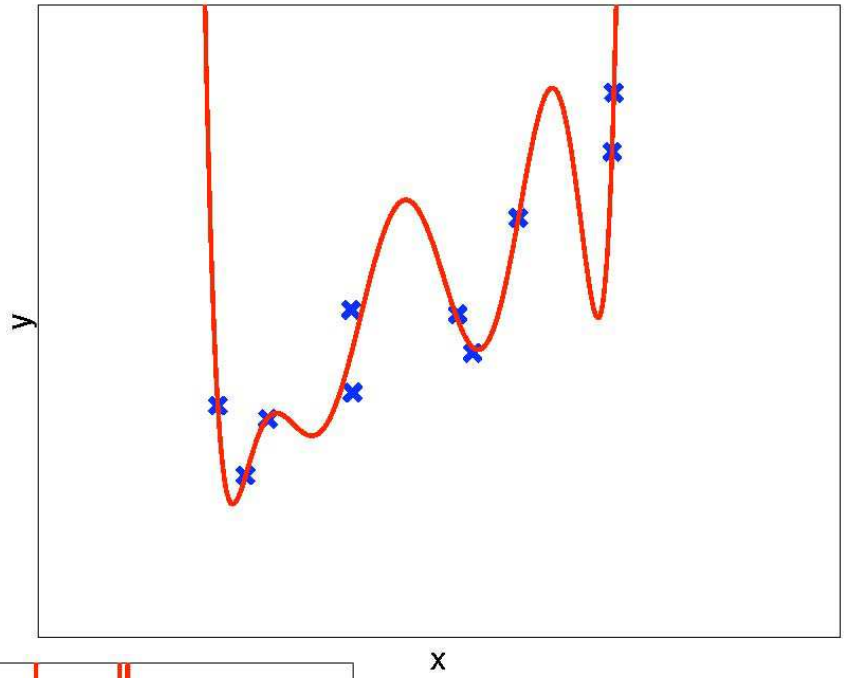


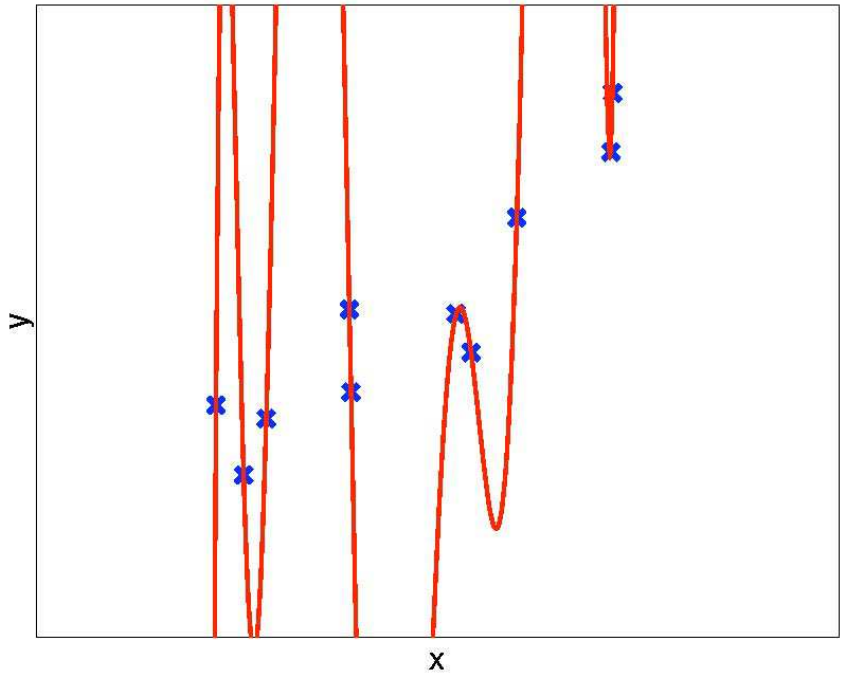




Regresión polinómica: grado 7, 9…







Regresión polinómica en Weka

- Weka no posee un algoritmo para obtener modelos de regresión polinómica de grado *d* como tal, pero tiene muchos más métodos de regresión que utilizan funciones de diversas tipologías. Algunos ejemplos son:

classifiers.functions.Multilayerperceptron:
Regresión con redes neuronales artificiales.

classifiers.functions.SMOreg
Regresión a partir de máquinas de vectores soporte.

classifiers.functions.RandomForest
Regresión a partir de árboles de decisión.

- Nos centraremos solamente en la regresión lineal para obtener modelos para problemas de regresión.

Regresión logística (clasificación)

- La regresión también se podría aplicar a problemas de clasificación → Regresión logística.

- Al igual que en la regresión lineal, consiste en predecir un valor de salida a partir de unas variables de entrada, pero en este caso la salida son etiquetas.

- Para clasificación, por tanto, habrá un modelo de regresión para cada una de las clases.

- Como la **salida** de un modelo de regresión lineal es numérica, lo que haremos será **transformarla en un valor de probabilidad**.

- Para ello se usa la función *Softmax* que indicará la **probabilidad de pertenencia de un patrón *i* a una clase**. Para un problema de 2 clases:

$$p_i = P(y_i = 1 \mid \mathbf{x}_i) = \frac{\exp(\boldsymbol{\beta}^{\mathrm{T}} \mathbf{x}_i)}{1 + \exp(\boldsymbol{\beta}^{\mathrm{T}} \mathbf{x}_i)} = \frac{1}{1 + \exp(-\boldsymbol{\beta}^{\mathrm{T}} \mathbf{x}_i)}$$

- Si $p_i > 0.5$ diríamos que el patrón se asignaría a la clase A o clase positiva.

- Si $p_i < 0.5$ diríamos que el patrón se asignaría a la clase B o clase negativa.

- De manera genérica, para *J* clases, habrá un modelo de regresión lineal como el siguiente, donde la probabilidad de pertenencia a la última clase se calcula restando a 1 la probabilidad obtenida con el resto. Esto se conoce como método de máxima verosimilitud.

$$P(\mathbf{x}_i \in C_j) = p_{ij} = \frac{\exp(\boldsymbol{\beta}^j \mathbf{x}_i)}{1+\sum_{m=1}^{J-1}\exp(\boldsymbol{\beta}^m \mathbf{x}_i)} = \frac{\exp(\beta_0^j + \sum_{i=1}^{k}\beta_i^j x_{ij})}{1+\sum_{m=1}^{J-1}\exp(\beta_0^j + \sum_{i=1}^{k}\beta_i^j x_{ij})},$$

$$\text{para } j = 1,\ldots,J\text{-}1$$

$$P(\mathbf{x}_i \in C_J) = p_{iJ} = 1 - \sum_{j=1}^{J-1} p_{ij}$$

Regresión logística (clasificación)

- Ejemplo de modelos de regresión lineal y su transformación mediante *Softmax* a valores de probabilidad para un problema de *K* clases siendo *K=3* y *n* el número de atributos o variables independientes *x*.

$$f_1(\mathbf{x}, \hat{\theta}) = \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\beta}_i x_i$$

$$f_2(\mathbf{x}, \hat{\theta}) = \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\beta}_i x_i$$

?

Función Softmax

$$p_1(\mathbf{x}, \hat{\theta}) = \frac{e^{f_1(\mathbf{x}, \hat{\theta})}}{1 + \sum_{k=1}^{K-1} e^{f_k(\mathbf{x}, \hat{\theta})}}$$

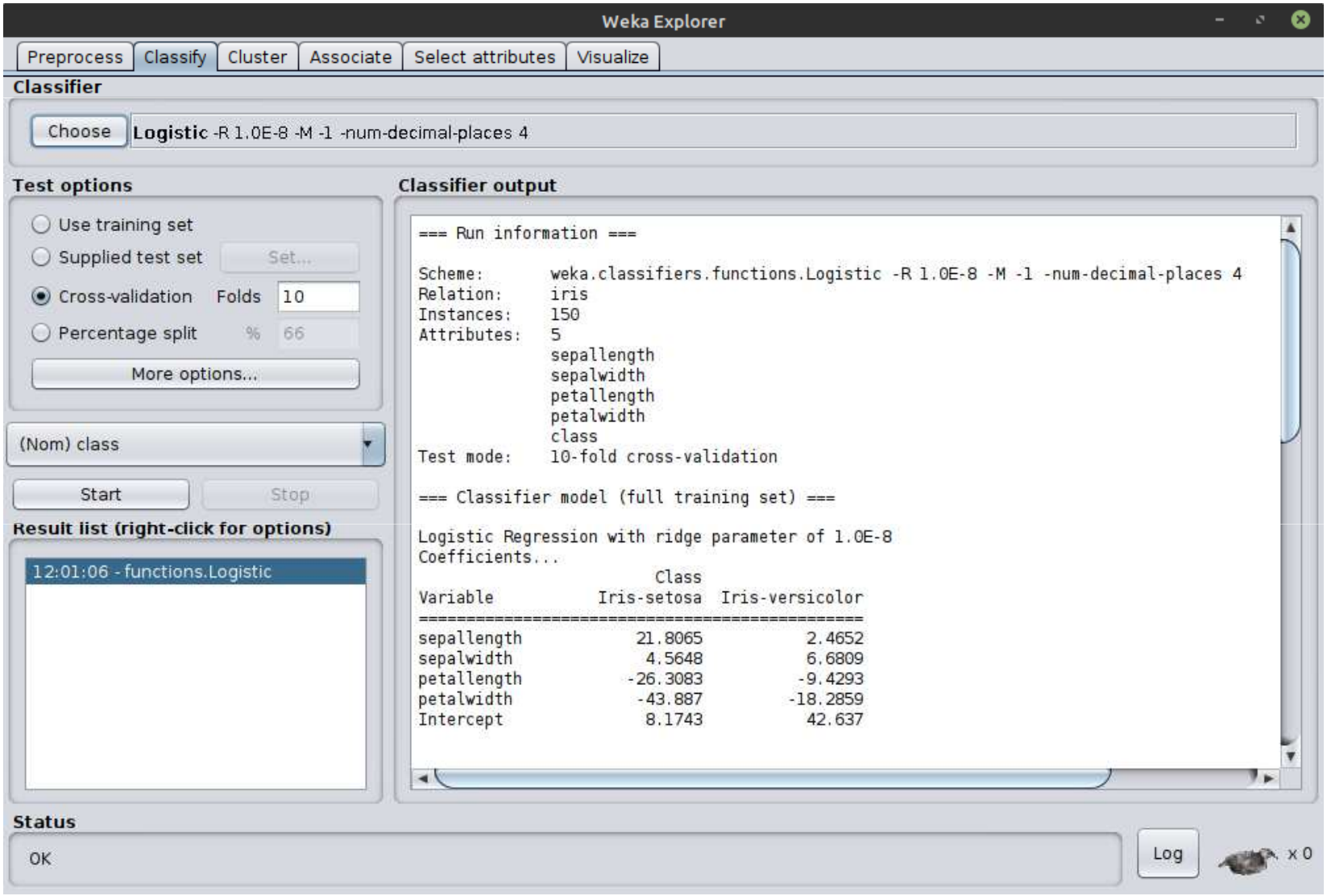
$$p_2(\mathbf{x}, \hat{\theta}) = \frac{e^{f_2(\mathbf{x}, \hat{\theta})}}{1 + \sum_{k=1}^{K-1} e^{f_k(\mathbf{x}, \hat{\theta})}}$$

$$p_3(\mathbf{x}, \hat{\theta}) = 1 - \sum_{k=1}^{K-1} p_k(\mathbf{x}, \hat{\theta}) = \frac{1}{1 + \sum_{k=1}^{K-1} e^{f_k(\mathbf{x}, \hat{\theta})}}$$

Regresión logística (clasificación)

Regresión Logística en Weka: Pestaña ***Classify.***

classifiers.functions.Logistic



- Existe **otra manera de calcular las probabilidades de salida** en cuanto a la pertenencia de un patrón a una clase, que es la que se usa para el algoritmo denominado *Logitboost*.

Ejemplo de modelos de regresión lineal y su transformación mediante *Softmax* a valores de probabilidad para un problema de *K* clases siendo *K=3* y *n* el número de atributos o variables independientes *x*, usando el algoritmo *Logitboost*.

$$f_1(\mathbf{x},\hat{\theta}) = \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\beta}_i x_i$$

$$f_2(\mathbf{x},\hat{\theta}) = \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\beta}_i x_i$$

$$f_3(\mathbf{x},\hat{\theta}) = \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\beta}_i x_i$$

Función Softmax

$$p_1(\mathbf{x},\hat{\theta}) = \frac{e^{f_1(\mathbf{x},\hat{\theta})}}{\sum_{k=1}^{K} e^{f_k(\mathbf{x},\hat{\theta})}}$$

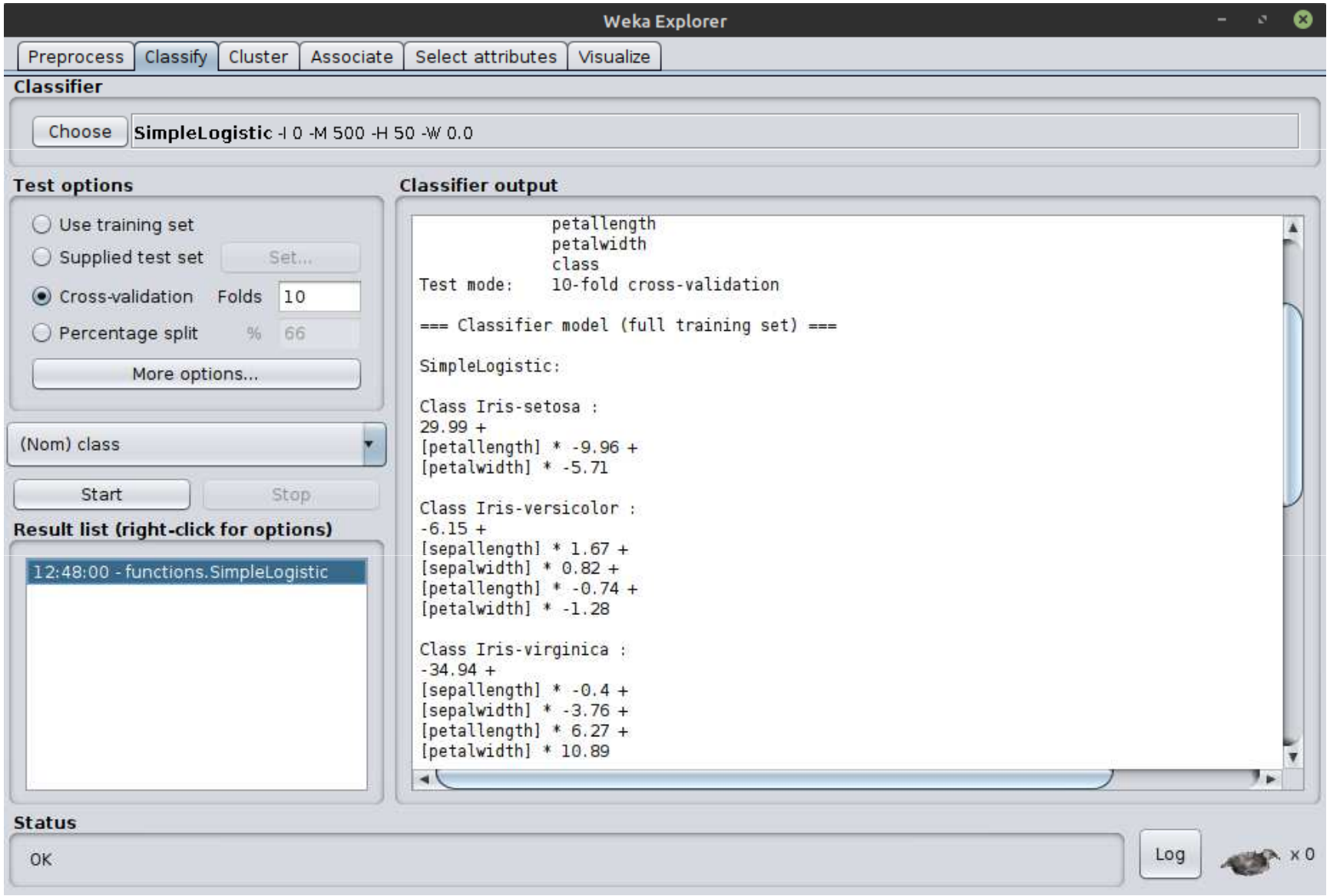
$$p_2(\mathbf{x},\hat{\theta}) = \frac{e^{f_2(\mathbf{x},\hat{\theta})}}{\sum_{k=1}^{K} e^{f_k(\mathbf{x},\hat{\theta})}}$$

$$p_3(\mathbf{x},\hat{\theta}) = \frac{e^{f_3(\mathbf{x},\hat{\theta})}}{\sum_{k=1}^{K} e^{f_k(\mathbf{x},\hat{\theta})}}$$

Regresión logística (clasificación)

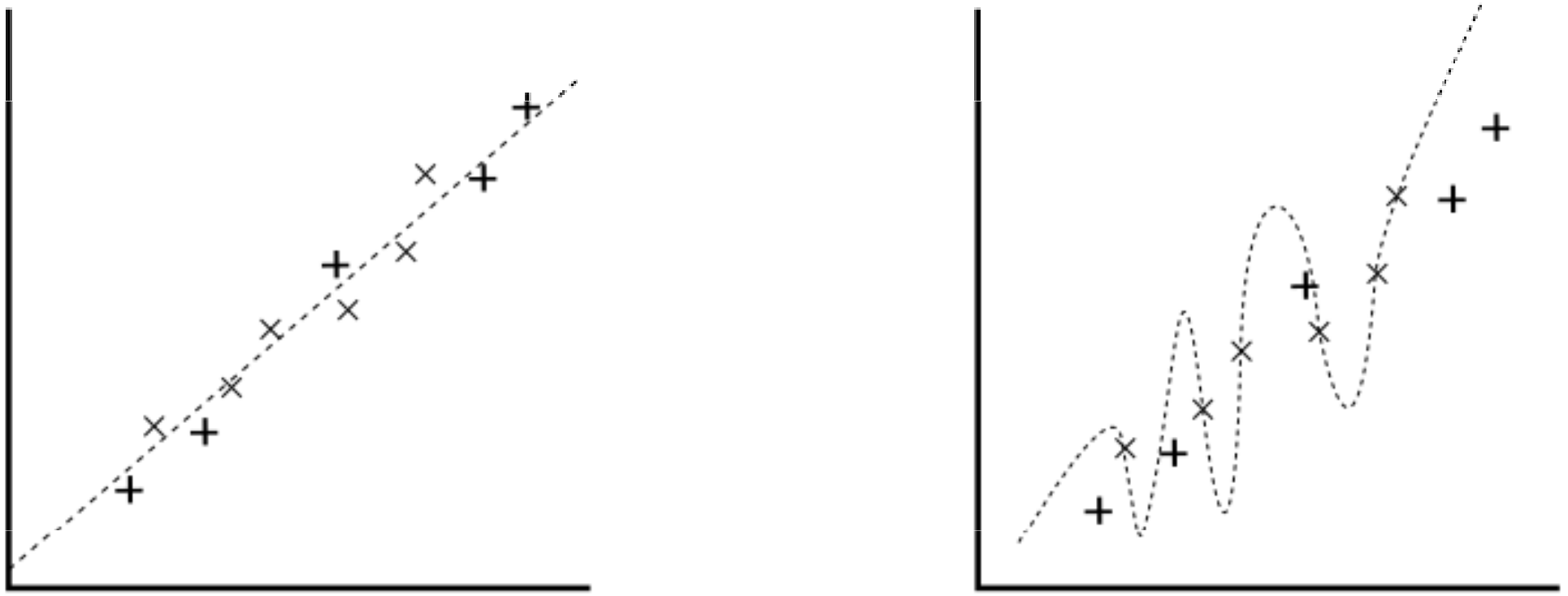
Regresión Logística (verosimilitud) en Weka: Pestaña *Classify.*

classifiers.functions.SimpleLogistic



Sobreaprendizaje

Aprender demasiado bien los datos de entrenamiento nos puede llevar a generalizar peor.



× Ejemplos de entrenamiento

\+ Ejemplos nuevos

La regresión polinómica de la derecha tendrá más error antes los nuevos ejemplos que la regresión lineal de la izquierda.

Habilidad de generalización y Sobre-aprendizaje

- La capacidad de generalización es muy importante en aprendizaje automático:
  - ¿Podrá el algoritmo predecir correctamente el valor de *y* de cualquier **x** desconocida?

- La hipótesis o modelo que se utilizó puede predecir muy bien las **x** conocidas pero no las que no aparecen en el conjunto de **datos *D***:
  - A esto se le denomina sobreaprendizaje.

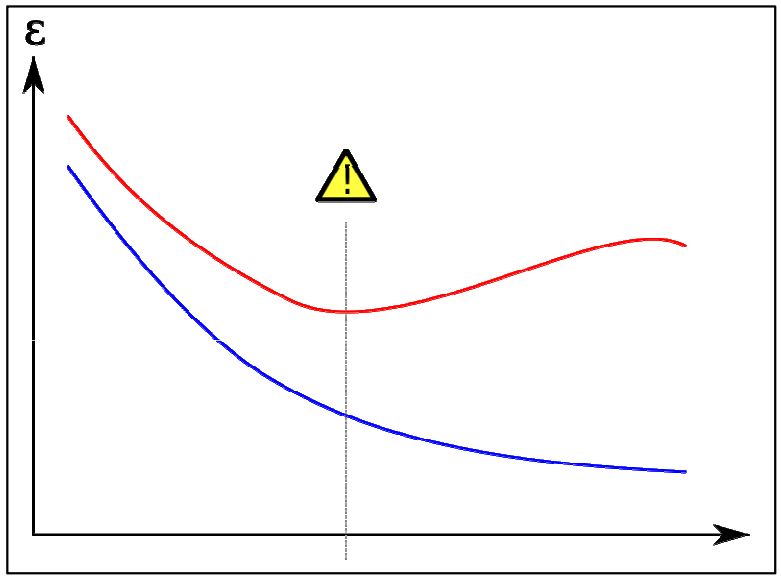
- Cada hipótesis *h* tiene un error desconocido para nuevas instancias que nunca ha visto o **datos *U***, y lo denotamos como $J_U(h)$.

- Nosotros solo medimos el error empírico sobre una muestra de entrenamiento, y lo denotamos como $J_D(h)$.

Habilidad de generalización y Sobre-aprendizaje

- Dadas dos hipótesis h_1 and h_2 (puede ser sobre el mismo tipo de modelo, por ejemplo dos modelos de regresión lineal), que se comparan en un conjunto de datos *D*, partimos de que $J_D(h_1) < J_D(h_2)$ (h_1 *es mejor que* h_2)
- Si sobre *U* se obtiene $J_U(h_2) < J_U(h_1)$, entonces hemos incurrido en sobreaprendizaje.
  - El modelo h_1 se ajusta bien a entrenamiento ($J_D(h_1) < J_D(h_2)$) pero no generaliza bien sobre instancias nuevas ($J_U(h_2) < J_U(h_1)$).
  - El modelo h_2 no se ajusta tan bien a entrenamiento ($J_D(h_1) < J_D(h_2)$) pero generaliza mejor que h_1 sobre instancias nuevas ($J_U(h_2) < J_U(h_1)$).
  - h_1 terminado memorizando el conjunto de entrenamiento.
- En el ejemplo anterior de regresión, subir el **grado *d*** puede dar lugar a sobreaprendizaje (memorización) de los datos.
- Tenemos que incorporar mecanismos que eviten el sobreaprendizaje.
- "En igualdad de condiciones, la explicación más sencilla suele ser la más probable".

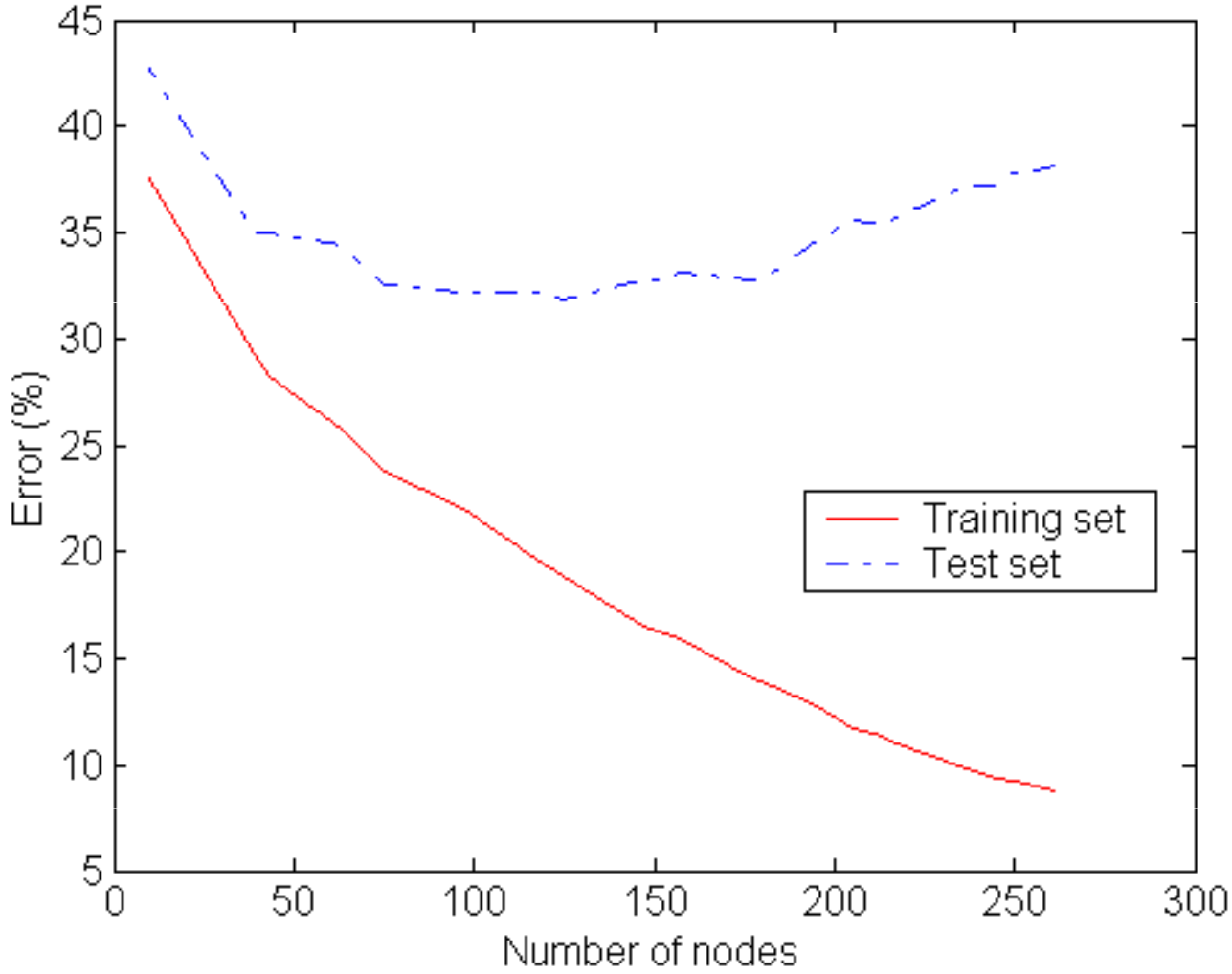
Habilidad de generalización VS Sobre-aprendizaje



Curva azul = error en entrenamiento.
Curva roja = error en datos nuevos.

- El error de entrenamiento suele decrecer con la complejidad de *h* (grado *d* en el ejemplo).
- El error en nuevos datos suele decrecer inicialmente y luego
- empezar a aumentar.
- ¿Cuál es la mejor *h?* Encontrar el grado *d, tal que* $J_U(h)$ sea mínimo, es decir, que el error sea mínimo sobre nuevos datos.
- División de *D* en los conjuntos disjuntos: **entrenamiento** y **test**, y mejorar el grado *d* hasta que se **estabilice el error en test**.

Sobreaprendizaje debido a la complejidad del clasificador



"*Number of nodes*" es el símil respecto al grado *d*.

d	Error_{train}	Error_{test}
1	0.2188	0.3558
2	0.1504	0.3095
3	0.1384	0.4764
4	0.1259	1.1770
5	0.0742	1.2828
6	0.0598	1.3896
7	0.0458	38.819
8	0.0000	6097.5
9	0.0000	6097.5

- El valor óptimo es *d=2*.
- Sobreaprendizjae para *d*>2.
- Errores muy altos con *d*=8 o *d*=9.

Análisis automático de datos para las ciencias biomédicas, medioambientales y agroalimentarias
(Transversal Másteres Universitarios)

Tema 8. Aprendizaje supervisado: Regresión lineal y regresión logística. Sobreaprendizaje.

Juan Carlos Fernández Caballero
Pedro Antonio Gutiérrez Peña
Departamento de Informática y Análisis Numérico
Universidad de Córdoba
Grupo de investigación AYRNA