VNIVERSITAT DE VALÈNCIA OpenCourseWare

Procesado y Análisis de Datos Ambientales.
Curso 2009-2010.
José D. Martín, Emilio Soria, Antonio J. Serrano

TEMA 2: MODELOS CLASIFICADORES

ÍNDICE

- Introducción.
- Clasificadores lineales.
- Árboles de decisión.
- Clasificador basado en redes neuronales: El perceptrón multicapa.

ÍNDICE

- **Introducción.**
- **Clasificadores lineales.**
- **Árboles de decisión.**
- **Clasificador basado en redes neuronales: El perceptrón multicapa.**

INTRODUCCIÓN

- Los problemas de clasificación son fundamentales en el análisis de datos ambientales.
- Bondad en la clasificación Vs. Complejidad de los modelos.
- Es importante analizar el número de clases y el de patrones dentro de cada clase.
- Clasificación no supervisada: agrupamiento (*clustering*).
- **Clasificación supervisada.**

ÍNDICE

- Introducción.
- Clasificadores lineales.
- Árboles de decisión.
- Clasificador basado en redes neuronales: El perceptrón multicapa.

CLASIFICADORES LINEALES (I)

VENTAJA FUNDAMENTAL → **SENCILLEZ**

Aproximación probabilística:

- ***Clasificación binaria: dos clases con probabilidades asociadas p y 1-p.***
- ***Probabilidad de pertenencia de la observación "i" a la clase A definida por una respuesta unidad (salida=1):***

$$p_i = \frac{e^{\beta_0+\beta_1 x_{1i}+....+\beta_J x_{J\,i}}}{1+e^{\beta_0+\beta_1 x_{1i}+....+\beta_J x_{J\,i}}}$$

¡¡Muy parecido a UNA neurona dentro de una estructura de red neuronal artificial!!

CLASIFICADORES LINEALES (II)

Los parámetros del modelo se estiman para maximizar la función de verosimilitud (iteración generalizada de Newton-Raphson, similar a unos mínimos cuadrados generalizados).

Regresión logística con repeticiones:

$$E\left(y_i \middle| x_1 = x_{1i},....,x_J = x_{Ji}\right) = n_i\, p_i = n_i\left(\frac{e^{\beta_0+\beta_1 x_{1i}+....+\beta_J x_{Ji}}}{1+e^{\beta_0+\beta_1 x_{1i}+....+\beta_J x_{Ji}}}\right)$$

Modelo logístico multinomial (K-clases):

$$p_{ik} = \frac{e^{\beta_0+\beta_1 x_{1i}+....+\beta_J x_{Ji}}}{1+\sum_{k=1}^{K-1} e^{\beta_0+\beta_1 x_{1i}+....+\beta_J x_{Ji}}},\ k = 1,2....,(K-1)$$

$$p_{iK} = \frac{1}{1+\sum_{k=1}^{K-1} e^{\beta_0+\beta_1 x_{1i}+....+\beta_J x_{Ji}}},\ k = K$$

ÍNDICE

- Introducción.
- Clasificadores lineales.
- Árboles de decisión.
- Clasificador basado en redes neuronales: El perceptrón multicapa.

ÁRBOLES DE DECISIÓN (I)

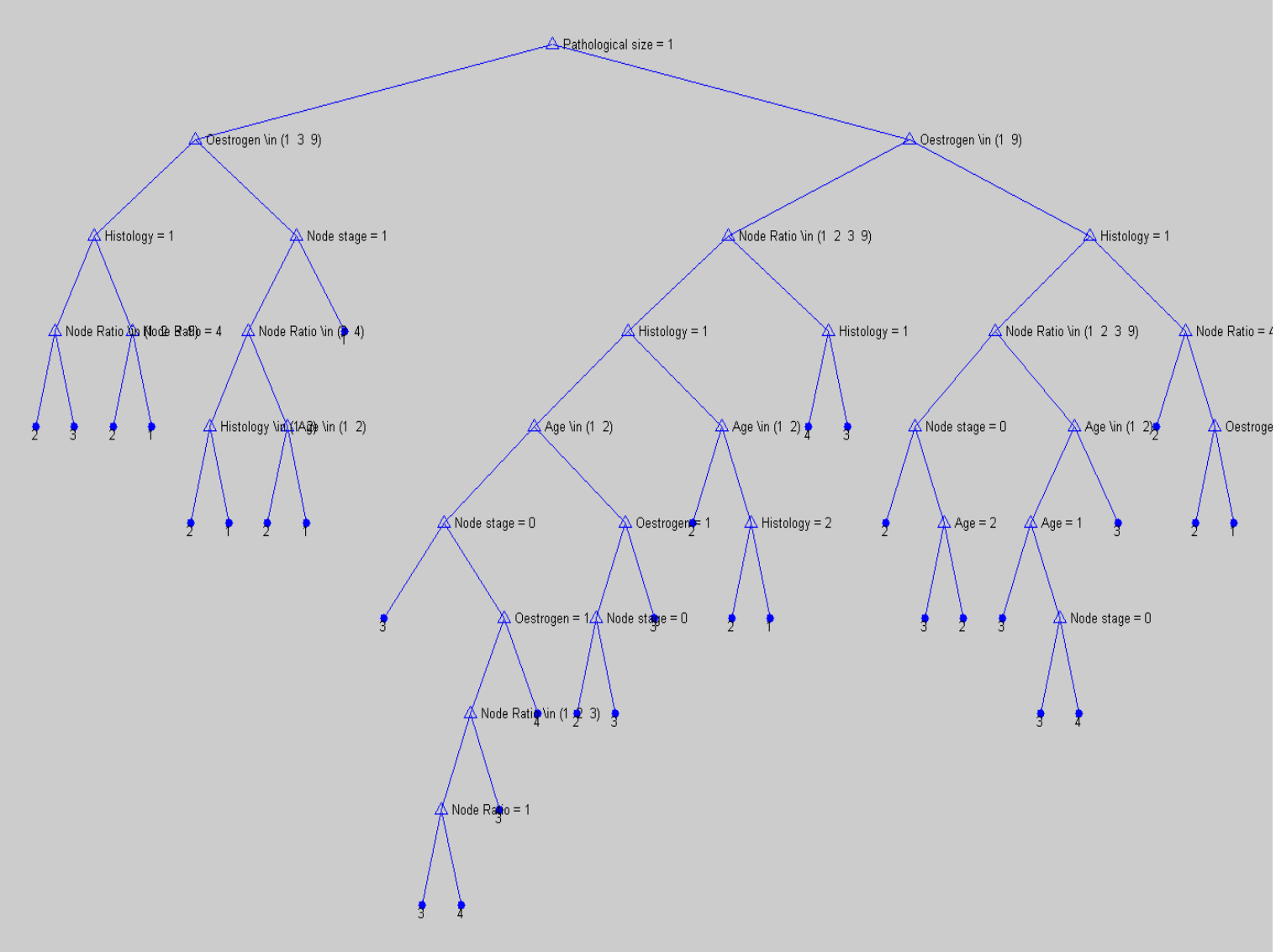
- El clasificador se representa gráficamente por un árbol de decisión.
- El objetivo es tener buenos clasificadores a partir de árboles lo más sencillos posibles.
- A partir de los árboles más sencillos, pueden explorarse árboles más complejos hasta llegar a un compromiso entre exactitud y complejidad.

ÁRBOLES DE DECISIÓN (II)

Un árbol de decisión es un árbol donde:

- **Los nodos que no son hojas se etiquetan con atributos.**
- **Las ramificaciones que salen del nodo etiquetado con el atributo A se etiquetan con los posibles valores de ese atributo.**
- **Las hojas del árbol se etiquetan con las clasificaciones.**

ÁRBOLES DE DECISIÓN (III)



ÁRBOLES DE DECISIÓN (IV)

• Un árbol de decisión puede representar cualquier función discreta de las entradas.

• Hace falta aplicar un “bias” para decidir el árbol con el que nos quedamos: árbol más pequeño, menos profundo, menos nodos, mejor predictor.

• Para construir un árbol de decisión hay que tener en cuenta que el espacio de árboles de decisión es demasiado grande para llevar a cabo una búsqueda sistemática, así que habrá que tomar alguna medida alternativa.

ÁRBOLES DE DECISIÓN (V)

• Para construir el árbol necesitamos un conjunto de patrones, unos atributos de entrada y uno de salida (clasificación).

• ¡¡ Si todos los ejemplos pertenecen a la misma clase no hay que construir nada!!

• En caso contrario, se selecciona un atributo, y se construye un subárbol diferente para cada uno de los diferentes valores que toma el atributo, que comprenderá a aquellos patrones que muestren ese valor del atributo.

ÁRBOLES DE DECISIÓN (VI)

- Los atributos pueden presentar más de dos valores, lo cual complica los árboles.
- Se asume que los atributos son adecuados para representar el problema, **¡¡así que es necesario un adecuado preprocesado!!**
- **¡Hay que ir con cuidado con el overfitting!**
- El atributo a seleccionar para generar el árbol se define utilizando conceptos de Teoría de la Información (habitualmente medidas de entropía).
- **¡¡Afortunadamente tenemos la instrucción *treefit* en Matlab!!**

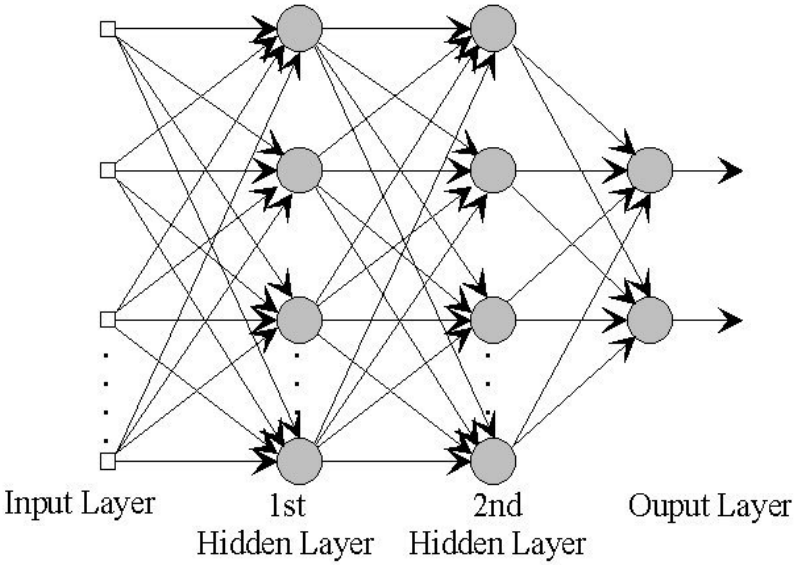
ÍNDICE

- Introducción.
- Clasificadores lineales.
- Árboles de decisión.
- **Clasificador basado en redes neuronales: El perceptrón multicapa.**

REDES NEURONALES. El MLP (I)

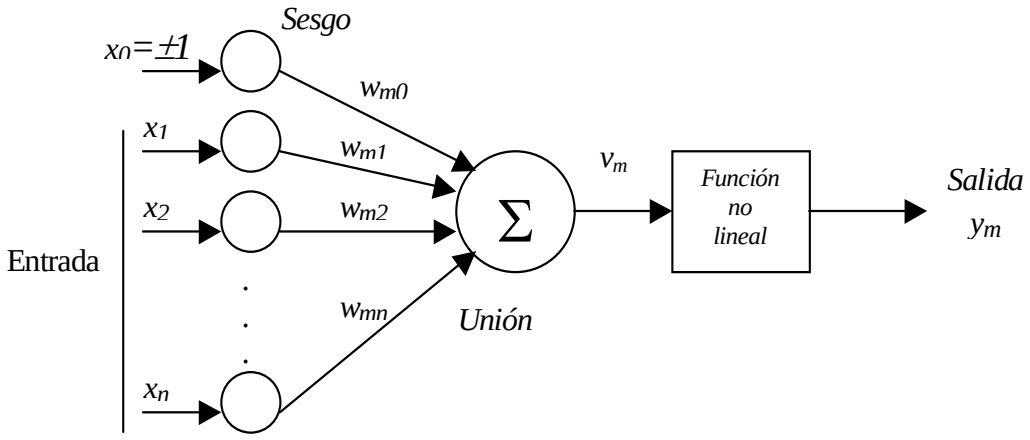
- Modelos no lineales: pérdida de sencillez a costa de una mayor exactitud.
- La naturaleza no es lineal.
- Inspirados en las redes neuronales biológicas ya que incorporan capacidad de aprendizaje.
- ¡¡ El MLP es un aproximador universal de funciones!! Mapea cualquier conjunto de entrada conexo en un conjunto de salida conexo.

REDES NEURONALES. El MLP (II)



- Capas de entrada y salida definidas por el problema.
- Capas ocultas deben ser optimizadas.
- Hay que controlar el sobreajuste y el sobreentrenamiento.
- El conocimiento se almacena en los pesos sinápticos.

PERCEPTRÓN MULTICAPA CON DOS CAPAS OCULTAS



NEURONA NO LINEAL

REDES NEURONALES. El MLP (III)

FUNCIÓN DE COSTE CUADRÁTICA

(Distribución de errores de tipo Normal)

$$J = \frac{1}{2M} \sum_{i=1}^{M} \sum_{j=1}^{N} e_j{}^2(i)$$

F.C. ENTRÓPICA

(Distribución Binomial)

$$J = \frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{N} \left((1+d_j(i)) \ln \left[\frac{1+d_j(i)}{1+o_j(i)} \right] + (1-d_j(i)) \ln \left[\frac{1-d_j(i)}{1-o_j(i)} \right] \right)$$

PROPAGACIÓN HACIA DELANTE (UNA CAPA OCULTA)

$$v_m(t) = \sum_{i=0}^{n} w_{mi}(t) \cdot x_i$$

$$y_m(t) = \varphi_m(v_m(t))$$

$$z_p(t) = \sum_{j=0}^{r} h_{pj}(t) \cdot y_j(t)$$

$$o_p(t) = \phi_p(z_p(t))$$

ϕ y φ son funciones de activación no lineales

• Existe una señal deseada que se compara con la salida obtenida por la red (aprendizaje supervisado).

• El aprendizaje se basará en la minimización de la función de coste.

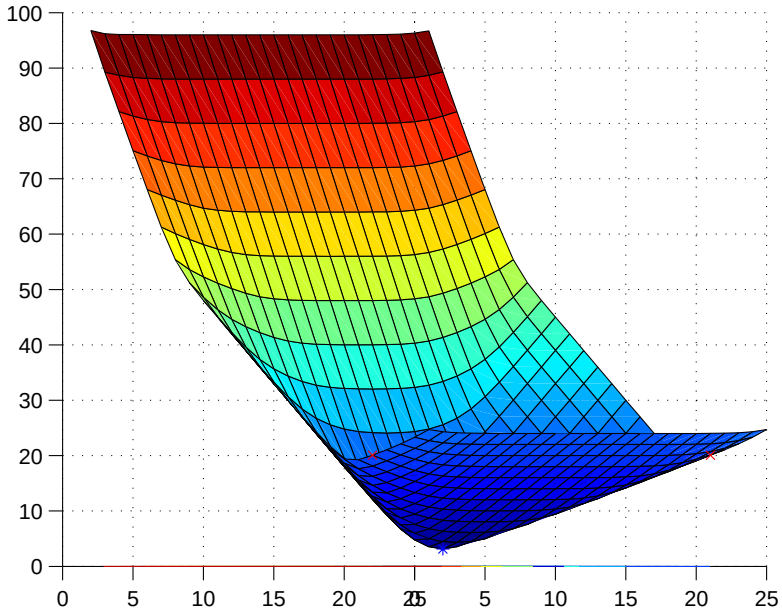
REDES NEURONALES. El MLP (IV)

Regla Delta
(*Backpropagation*)

$$\Delta\xi(t) = -\alpha \frac{\partial J}{\partial \xi(t)}$$

α: cte. Adaptación

ξ: pesos sinápticos



$$\Delta h_{pj}(t) = 2 \cdot \alpha \cdot e_p(t) \cdot \phi_p'(z_p(t)) \cdot y_j(t)$$

$$\Delta h_{p0}(t) = 2 \cdot \alpha \cdot e_p(t) \cdot \phi_p'(z_p(t)) \cdot 1$$

$$\Delta w_{mi}(t) = 2 \cdot \alpha \cdot \sum_p e_p(t) \cdot \phi_p'(z_p(t)) \cdot h_{pm}(t) \cdot \varphi_m'(v_m(t)) \cdot x_i$$

ACTUALIZACIÓN PESOS SINÁPTICOS

$$\varphi(x) = \frac{a}{1+e^{(-b\cdot x)}} \rightarrow \varphi'(x) = a\cdot b\cdot\varphi\cdot(1-\varphi)$$

(Sigmoide, entre 0 y 1)

$$\varphi(x) = a\cdot\frac{1-e^{-b\cdot x}}{1+e^{-b\cdot x}} \rightarrow \varphi'(x) = \frac{1}{2}\cdot a\cdot b\cdot(1-\varphi^2)$$

(Tangente hiperbólica, entre -1 y +1)

REDES NEURONALES. El MLP (V)

Inconvenientes del Backpropagation:

- **Saturación de las neuronas debido a la dependencia con la derivada de la función de activación.**

Solución: $\varphi_\beta(x)=\beta\cdot x+(1-\beta)\varphi(x)$ **donde** $1\geq\beta\geq 0$ ($\beta=1$ *al principio del entrenamiento*).

- **Inicialización de los pesos sinápticos puede hacer caer el sistema en un mínimo local, provocar saturación de las neuronas y afectar al tiempo de convergencia.**

Solución: Algoritmo ERA (*Expanded Range Approximation*)

$$d_{ent} = \langle d \rangle + \lambda \left[d - \langle d \rangle \right]$$

- **Zonas planas de la superficie de error implican no actualización.**

Solución: Añadir término a la derivada (posibles inestabilidades) o incluir información sobre la superficie de error considerando la segunda derivada.

- **Elección de la constante de adaptación.**

Solución de compromiso o algoritmos que aceleran la convergencia lejos del mínimo y la ralentizan cerca.

REDES NEURONALES. El MLP (VI)

- **Parada del aprendizaje.**

Solución: Controlar sobreentrenamiento a partir de *Early-stopping*, validación cruzada, *v-fold, leave-one-out*, ….

- **Elección de la arquitectura.**

Solución: Controlar sobreajuste, métodos de poda y crecimiento.

- **Elección de los patrones de entrenamiento.**

Solución: Conjunto representativo.

- **Funciones de error.**

Solución: base estadística para su elección. Pueden añadirse términos para simplificar la red (términos de penalización) y ¡¡aportar conocimiento *a priori* sobre el problema!!

REDES NEURONALES. El MLP (VII)

Variantes del Backpropagation.

- **Variante de momento. Acelera la convergencia lejos del mínimo y la ralentiza cerca.**

$$\Delta w_{mi}(t) = -\alpha \cdot \nabla J + \mu \cdot \Delta w_{mi}(t-1)$$

- **Silva-Almeida (*d<1* y *u>1*).**

$$\alpha(t) = \begin{cases} \alpha(t-1)\cdot u \Longleftrightarrow \left(\nabla_{w_{ij}(t)} J\right)\cdot\left(\nabla_{w_{ij}(t-1)} J\right) > 0 \\ \alpha(t-1)\cdot d \Longleftrightarrow \left(\nabla_{w_{ij}(t)} J\right)\cdot\left(\nabla_{w_{ij}(t-1)} J\right) < 0 \end{cases}$$

- **DELTA-BAR-DELTA. Evita inestabilidades del método de Silva-Almeida y es más inmune al ruido.**

$$\alpha(t) = \begin{cases} \alpha(t-1)+u \Longleftrightarrow \left(\nabla_{w_{ij}(t)} J\right)\cdot\left(\delta_{ij}(t-1)\right) > 0 \\ \alpha(t-1)\cdot d \Longleftrightarrow \left(\nabla_{w_{ij}(t)} J\right)\cdot\left(\delta_{ij}(t-1)\right) < 0 \end{cases} \qquad 0<\theta<1$$

$$\delta_{ij}(t-1) = (1-\theta)\nabla_{w_{ij}(t-1)} J + \theta \cdot \delta_{ij}(t-2)$$

REDES NEURONALES. El MLP (VIII)

- **RPROP.**

$$\Delta w_{ij}(t) = -\alpha(t) \cdot \text{signo}\left(\nabla_{w_{ij}(t)} J\right)$$

$$\alpha(t) = \begin{cases} min\left(\alpha(t)\cdot u, \alpha_{max}\right) \Longleftrightarrow \left(\nabla_{w_{ij}(t)} J\right)\cdot\left(\nabla_{w_{ij}(t-1)} J\right) > 0 \\ max\left(\alpha(t)\cdot d, \alpha_{min}\right) \Longleftrightarrow \left(\nabla_{w_{ij}(t)} J\right)\cdot\left(\nabla_{w_{ij}(t-1)} J\right) < 0 \end{cases}$$

(u>1 y d<1)

Se evitan valores muy altos de α para evitar inestabilidades y muy bajos para evitar ralentizar demasiado la convergencia

CONCLUSIONES FINALES

- Deben utilizarse siempre métodos lo más sencillos posibles. Si un algoritmo sencillo funciona, no vale la pena complicarlo con variantes ¡¡pero si tenemos problemas sí!!
- Es fundamental realizar una buena definición del problema y un preprocesado adecuado.
- El MLP es una herramienta muy potente que debe manejarse con cautela, evitando sus inconvenientes más comunes.
- Debe llegarse a un compromiso entre estabilidad y plasticidad, y entre complejidad e interpretabilidad.
- Para decidir el método a emplear hemos de plantearnos el objetivo que queremos alcanzar y el problema a resolver.