Clasificación de patrones
Introducción

Jordi Porta Zamorano

Escuela Politécnica Superior **Dept. de Lingü´ıstica Computacional**

Universidad Autónoma de Madrid **Real Academia Española**

jordi.porta@uam.es **porta@rae.es**

abril de 2005

Contenidos del curso

- Introducción
  - Ciclo de diseño
  - Tipos de aprendizaje
  - Entradas y salidas
  - Visualización de datos
- Métodos supervisados
  - Vecinos próximos
  - “Naïve Bayes”
  - Modelos lineales
  - Perceptrones y Winnow
  - Máquinas de soporte vectorial
  - Árboles y reglas de decisión
  - Metamétodos: Bagging, Boosting y Stacking

Contenidos del curso

- Validación
- Métodos no supervisados
  - Agrupamiento
    - Agrupamiento jerárquico
    - Agrupamiento dinámico
    - Otras técnicas

Referencias

[Duda et al.(2001)Duda, Hart, and Stork] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. John Wiley & Sons, 2nd edition, 2001.

[Jain et al.(1999)Jain, Murty, and Flynn] A. K. Jain, M.Ñ. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):264–323, sep. 1999.

[Jurafsky and Martin()] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An introduction to Natural Language Precessing, Computational Linguistics and Speech Recognition*.

[Manning and Schütze(1999)] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

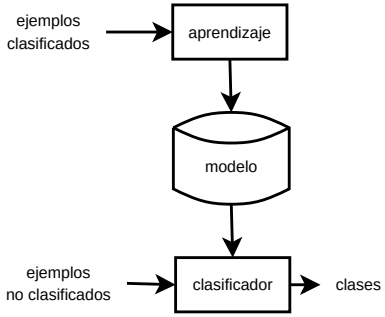
[Mitchell(1997)] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.

[Norvig and Russell(2002)] Peter Norvig and Stuart Russell. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2002.

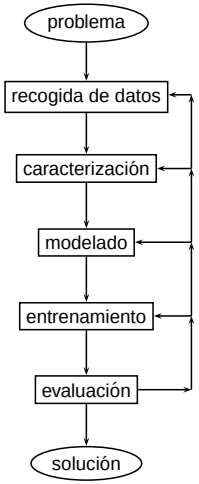
[Witten and Frank(2000)] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 2000.

Clasificadores

- basados en conocimiento *elicitado*
- basados en conocimiento extraído inductivamente:



Clasificadores: El ciclo de diseño



El ciclo de diseño: Recogida de datos

- costes de adquisición
- factibilidad del problema: empezar con un conjunto de datos “típicos”
- representatividad de los datos: cantidad y calidad

El ciclo de diseño: Caracterización de los datos

- seleccionar el conjunto de rasgos y valores que caracterizarán los datos según su:
  - facilidad de extracción
  - insensibilidad al ruido
  - utilidad en la discriminación de datos en varias categorías
- usar el *conocimiento a priori* sobre el problema

El ciclo de diseño: Modelización

- deben combinar la información de las caracterizaciones de los datos
- no hay modelos mejores que otros: dependen del problema y de los datos
- la búsqueda de un modelo es un proceso de *ensayo y error*

El ciclo de diseño: Entrenamiento

El *entrenamiento* es el proceso por el cual se determina el modelo mediante la estimación de sus parámetros. Habrá que tener en cuenta el coste (tiempo) de entrenamiento que dependerá de:

- el número de parámetros
- la complejidad del modelo
- el número de ejemplos usados

El ciclo de diseño: Evaluación del modelo

La *evaluación* del modelo obtenido con el entrenamiento consistirá básicamente en la definición de una función de error con la que podremos:

- medir la bondad del modelo o su capacidad de *generalización*:
  - *sobreajuste* (*overfitting*): el modelo clasifica bien los patrones usados durante el entrenamiento pero no los nuevos patrones. Para medirlo, los datos clasificados se dividirán en dos conjuntos:
    - uno para entrenamiento (*training-set*)
    - otro para test (*test-set*)
  - *inestabilidad*: cuando variando ligeramente el conjunto de entrenamiento varía la predicción del modelo inducido.

El ciclo de diseño: Evaluación del modelo

y además podremos:

- identificar mejoras en sus componentes
- revelar errores en los datos (*ruido*) que habrá que revisar (*data cleansing*):
  - de caracterización
  - de clasificación a priori
  - de adquisición (valores erróneos)
- identificar los *outliers* en los datos
- comparar diferentes caracterizaciones de los datos con el mismo modelo
- comparar el modelo con otros modelos

Clasificadores: Tipos de aprendizaje de modelos

Como método de aprendizaje entenderemos el algoritmo usado para la reducción del error de un modelo para clasificación, que puede ser:

- *aprendizaje supervisado*: alguien (un *tutor*) clasifica los datos para el entrenamiento y validación.
- *aprendizaje no supervisado*: asimilado a *agrupamiento* (*clustering*) no hay un tutor explícito y el clasificador construye agrupaciones o clases "naturales" con los datos que se le proporcionan. El concepto de natural depende de la técnica de agrupación y de una función de similitud entre patrones que hay que integrar en el algoritmo.
- *aprendizaje por refuerzo*: usa un *crítico* que se encarga de proporcionar *feedback* al modelo sobre si la respuesta es correcta o no. A este tipo de aprendizaje se le conoce por el "método del palo y la zanahoria" (no se tratará ningún modelo de este tipo).

Clasificadores: Entradas (caracterizaciones)

Cada ejemplo proporcionado al clasificador viene representado por un conjunto, habitualmente fijo, de pares *atributo*-valor que se han decidido, durante la etapa de caracterización, como los más relevantes para la tarea de clasificación.

El tipo de atributo pueden clasificarse de manera distinta según sean los valores que pueda tomar:

- cuantitativos (o numéricos): números reales y enteros
- cualitativos (o categóricos):
  - ordinales: existe un criterio para ordenar los valores (p. ej.: bajo $<$ normal $<$ alto).
  - nominales: no existe un criterio de ordenación (p. ej.: hombre y mujer)

La clase de un ejemplo suele darse como el valor de un atributo más.

Clasificadores: Entradas (caracterizaciones)

- Problemas:
  - atributos *inaplicables* o *irrelevantes* (número de ruedas de los vehículos de tipo barco) se marcan de alguna manera
  - valores *desconocidos*
    - de atributos cuantitativos: valor fuera del rango
    - de atributos cualitativos: blanco o un guión o un valor más en atributos nominales

    El origen de los valores desconocidos es diverso.
  - No todos los modelos aceptan valores desconocidos $\Longrightarrow$ estimación
  - modelos diseñados sólo para trabajar con atributos cuantitativos $\Longrightarrow$ codificación numérica
  - modelos diseñados sólo para trabajar con atributos cualitativos $\Longrightarrow$ discretización

Atributos: Discretización

- consiste en dividir el rango de un atributo continuo en *intervalos* y asignar la etiqueta del intervalo correspondiente
- si el nuevo atributo es ordinal sus valores respetarán el órden impuesto por los límites del intervalo
- ejemplo:

edad	5	6	6	9	...	15	16	16	17	20	...	24	25	42	50	65	...	67
conduce	0	0	0	0	...	0	1	0	1	1	...	0	1	1	1	1	...	1

¿ $[5, 15]$, $[16, 24]$, $[25, 67]$?

- algunos algoritmos usan heurísticas como:
  - intervalos de igual tamaño
  - intervalos con la misma frecuencia
  - test χ^2 para la fusión de intervalos adjacentes
  - entropía: intervalos con el menor número de clases distintas

Atributos: Codificación numérica

- codificación numérica de valores cualitativos:

rojo	1
verde	2
azul	3

(a) enumeración de valores

rojo	(1 0 0)
verde	(0 1 0)
azul	(0 0 1)

(b) vectores característicos

rojo	(0 0)
verde	(0 1)
azul	(1 0)

(c) binarización

alta	(1 1 1)
media	(1 1 0)
baja	(1 0 0)

(d) termómetro (ordinales)

- en determinados modelos conviene sustituir los 0 por -1 (*bipolarización*) para que no se anulen en los productos.

Salidas (clasificadores)

- tablas de decisión
- árboles de decisión/regresión
- reglas de clasificación/asociación/con excepciones
- parámetros de funciones
- ejemplos
- agrupaciones

Las clasificaciones resultantes pueden tener asociado un *grado de confianza* o *significación*.

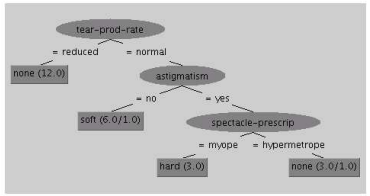
Tablas de decisión

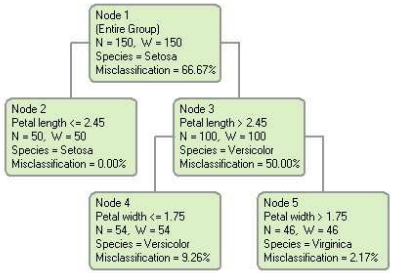
- la forma más simple de salida
- contiene los atributos (relevantes) para la clasificación
- se busca en la tabla las condiciones más apropiadas y se obtiene la clasificación

ASTIGMATISM	TEAR-PROD-RATE	CONTACT-LENSES
yes	reduced	none
no	reduced	none
yes	normal	hard
no	normal	soft

Árboles de decisión

- los *nodos internos* contienen un atributo del que salen tantas ramas como valores tenga el atributo o condiciones haya sobre los valores de atributo
- los *nodos terminales del árbol* (*hojas*) contienen la clasificación, un conjunto de clases o una distribución
- Ejemplos:





Reglas de clasificación

- toman la forma:

  antecedente o premisa $\Longrightarrow$ *consecuente o conclusión*

- la interpretación más habitual para el antecedente es la conjuntiva (AND)
- el conjunto de reglas puede interpretarse:
  - en orden (*decision list*): La primera regla, en orden, que ve cumplido su antecedente es la que clasifi ca.

    ```
    if A > 2, C < 5 then C1
    if B < 4 then C2
    ```

    (A=3, B=1, C=2) $\Longrightarrow$ C1
  - en paralelo (OR): Se intenta aplicar todas las reglas y se puede dar varias clases.

    ```
    if A > 2, C < 5 then C1
    if B < 4 then C2
    ```

    (A=3, B=1, C=2) $\Longrightarrow$ C1 y C2

Reglas de asociación

- Este tipo de reglas es formalmente igual que las reglas de decisión pero el consecuente puede ser la asociación entre atributos.
- Se usan para obtener regularidades en los datos.

  ```
  if A > 2 then B < 4
  ```

- Ejemplo

```
PredictiveApriori
==================
Best rules found:
1. tear-prod-rate=reduced 12
    ==> contact-lenses=none 12 acc:(0.99355)
2. contact-lenses=soft 5
   ==> astigmatism=no tear-prod-rate=normal 5 acc:(0.96077)
3. contact-lenses=hard 4
   ==> astigmatism=yes tear-prod-rate=normal 4 acc:(0.93425)
4. age=young contact-lenses=none 4 ==> tear-prod-rate=reduced 4
   acc:(0.93425)
...
```


Reglas con excepciones

- Son una extensión de las reglas de clasifi cacõn a las que se le añaden las condiciones de excepción en las que la clasifi cacõn es distinta. P. ej.:

  ```
  if A > 2 and B < 4 then C1 except if C = 5 then C2
  ```

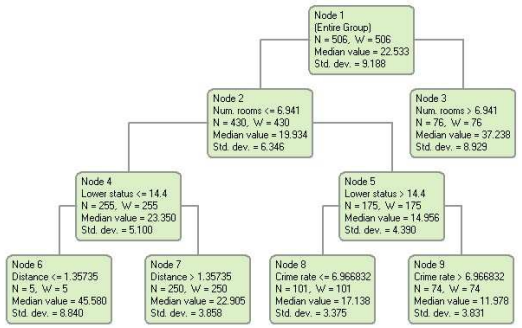
  (A=3, B=1, C=5) $\Longrightarrow$ C1

- Ejemplo:

```
RIpple DOwn Rule Learner(Ridor) rules
--------------------------------------
contact-lenses = soft  (24.0/19.0)
   Except (astigmatism = yes)
      => contact-lenses = none  (7.0/0.0) [5.0/0.0]
   Except (spectacle-prescrip = myope)
      => contact-lenses = hard  (5.0/3.0) [1.0/0.0]
   Except (tear-prod-rate = reduced)
      => contact-lenses = none  (4.0/0.0) [2.0/0.0]
```


Árbol de regresión

- es el equivalente a un árbol de decisión en problemas de predicción numérica
- en las hojas del árbol se almacenan las medias de los valores de las clases de los ejemplos de entrenamiento que han llegado por esas ramas
- comparado con una ecuación lineal obtenida por regresión, las predicciones de un árbol pueden ser más precisas en media
- ejemplo:



Representaciones basadas en ejemplos

- En métodos como el de representaciones basadas en ejemplos: En métodos como el de *vecinos próximos*, en el que la clase de un ejemplo se da por la distancia con otros objetos,
- la salida del algoritmo de aprendizaje es el mismo conjunto o un subconjunto de los ejemplos de entrenamiento.

Agrupaciones

suponiendo un conjunto de ejemplos $e_1, \ldots, e_{100}$:

- clases:

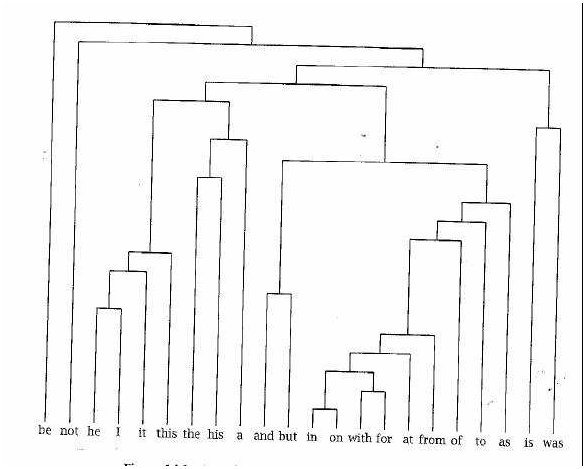
C_1: $e_1, \ldots, e_{49}$	C_1: $e_1, e_2, e_3, \ldots$
C_2: $e_{50}, \ldots, e_{100}$	C_2: $e_2, \ldots$
sin solapamiento (disjuntas)	con solapamiento

- vectores característicos:

e_1: $(1,0)$	e_1: $(0{,}8, 0{,}2)$
e_2: $(0,1)$	e_2: $(0{,}6, 0{,}4)$
$\vdots$ $\vdots$	$\vdots$ $\vdots$
sin solapamiento	con solapamiento

Agrupaciones

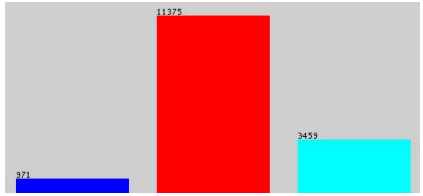
- jerarquías: un ejemplo pertenece a un grupo que está dentro de otro grupo, etc.. También son habituales los *dendrogramas*: representaciones gráfi cas de agrupaciones jerárquicas en las que la altura de una agrupación indica la similitud entre sus subgrupos.



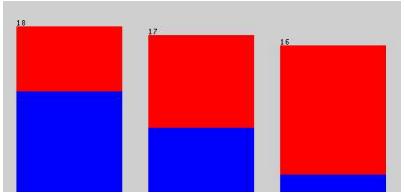
Visualización de datos

Permiten ver *distribuciones* y la existencia de *correlaciones*:

- *Histograma*:
  - gráfi ca que representa la distribución de un atributo:

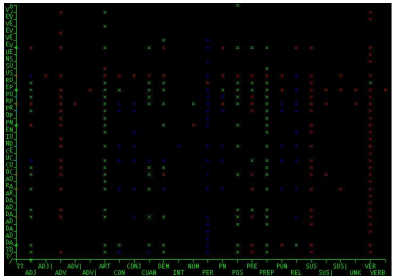


  - cada barra del histograma se puede particionar para mostrar la distribución en los valores de un segundo atributo:

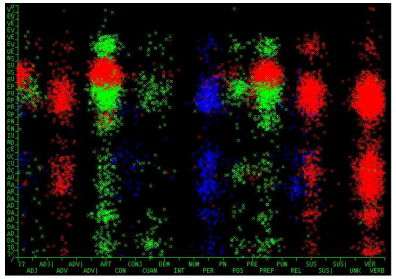


Visualización de datos

- *Diagramas de dispersión*: gráfi ca en la que cada punto representa un ejemplo situado según los valores de dos atributos. Cada punto puede representarse con un color o forma distinta según el valor de un tercer atributo (p. ej. la clase).



con superposición



con perturbación