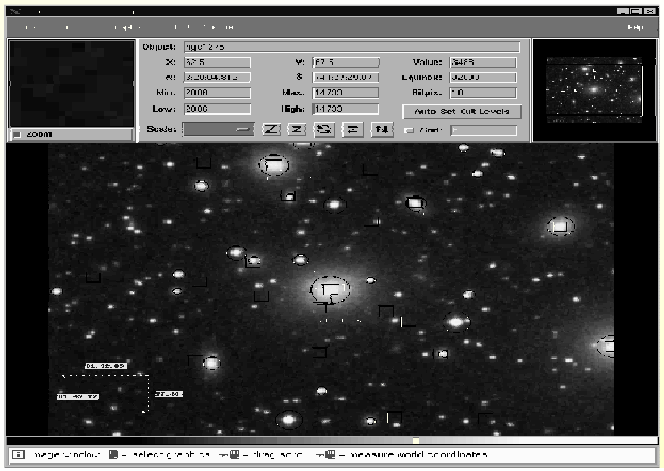
Introducción al Aprendizaje Automático y a la Minería de Datos con Weka

Índice

INTRODUCCIÓN A LA MINERÍA DE DATOS

SKYCAT: Clasificación automática de objetos del firmamento



Minería de Datos. Justificación

- **Nuevas posibilidades:** disponibilidad de grandes cantidades de datos (bancos, la web, tarjetas fidelización, Web...), potencia de cómputo
- **Nuevas necesidades:** Es complicado analizar los datos de manera manual. Necesidad de técnicas automáticas
- **Objetivo:** convertir datos en conocimiento para tomar decisiones
- MD = BBDD + estadística + aprendizaje automático

- **TAREAS EN MINERÍA DE DATOS**

Minería de Datos. Tareas

- Predicción:
  - Clasificación
  - Regresión
- Asociación
- Agrupación (clustering)

Ejemplo1. Créditos bancarios (clasificación)

- Un banco por Internet desea obtener reglas para predecir qué personas de las que solicitan un crédito no van a devolverlo.
- La entidad bancaria cuenta con una gran base de datos correspondientes a los créditos concedidos (o no) a otros clientes con anterioridad.

Ejemplo1. Datos

IDC	Años	Euros	Salario	Casa propia	Cuentas morosas	…	Devuelve el crédito
101	15	60000	2200	Si	2	…	No
102	2	30000	3500	Si	0	…	Si
103	9	9000	1700	Si	1	…	No
104	15	18000	1900	No	0	…	Si
105	10	24000	2100	No	0	…	No
…	…	…	…	…	…		…

Esquema general en predicción / clasificación

Años	Euros	Salario	Casa propia	Cuentas morosas	Crédito
10	50000	3000	Si	0	??

Años	Euros	Salario	Casa propia	Cuentas morosas	Crédito
15	60000	2200	Si	2	No
2	30000	3500	Si	0	Si
9	9000	1700	Si	1	No
15	18000	1900	No	0	Si
10	24000	2100	No	0	No
	…	…	…	…	…

Algoritmo MD

IF CM >0 **THEN** NO

IF CM =0 **Y** S>2500 **THEN** SI

Crédito = Si

Ejemplo 1. Conocimiento obtenido

- **SI** (cuentas-morosas > 0) **ENTONCES** Devuelve-crédito = no
- **SI** (cuentas-morosas = 0) **Y** ((salario > 2500) **O** (años > 10)) **ENTONCES** devuelve-crédito = si

Ejemplo 2. Determinar las ventas de un producto (Regresión)

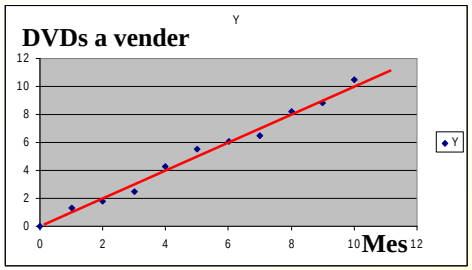
- Una gran cadena de tiendas de electrodomésticos desea optimizar el funcionamiento de su almacén manteniendo un stock de cada producto suficiente para poder servir rápidamente el material adquirido por sus clientes.

Ejemplo 2. Datos

Producto	Mes-12	…	Mes-4	Mes-3	Mes-2	Mes-1
Televisor plano	20	…	52	14	139	74
Video	11	…	43	32	26	59
Nevera	50	…	61	14	5	28
Microondas	3	…	21	27	1	49
Discman	14	…	27	2	25	12
…	…	…	…	…	…	…

Ejemplo 2. Conocimiento obtenido

- Modelo que prediga lo que se va a vender cada mes a partir de lo que se vendió en los meses anteriores (serie temporal)



Ejemplo 3. Análisis de la cesta de la compra (Asociación)

- Un supermercado quiere obtener información sobre el comportamiento de compra de sus clientes.
- Se piensa que de esta manera se puede mejorar el servicio, colocando ciertos productos juntos, etc.

Ejemplo 3. Datos de las cestas

Id	Huevos	Aceite	Pañales	Vino	Leche	Mantequilla	Salmón	Lechugas	…
1	Si	No	No	Si	No	Si	Si	Si	…
2	No	Si	No	No	Si	No	No	Si	…
3	No	No	Si	No	Si	No	No	No	…
4	No	Si	Si	No	Si	No	No	No	…
5	Si	Si	No	No	No	Si	No	Si	…
6	Si	No	No	Si	Si	Si	Si	No	…
7	No	No	No	No	No	No	No	No	…
8	Si	Si	Si	Si	Si	Si	Si	No	…
…	…	…	…	…	…	…	…	…	…

Ejemplo 3. Conocimiento obtenido

- Reglas **Si** At_1=a y At_2=b y ... **Entonces** At_n=c
  - **Si** pañales=si, **entonces** leche=si (100%, 37%)
  - **Si** huevos=si, **entonces** aceite=si (50%, 25%)
  - **Si** vino=si, **entonces** lechugas=si (33%, 12%)
- Las reglas también pueden ser:
  - **Si** At_1=a y At_2=b **Entonces** At_n=c, **At_4=D**
- (a,b) = (precisión, cobertura)
  - Precisión: veces que la regla es correcta
  - Cobertura: frecuencia de ocurrencia de la regla en los datos

Ejemplo 4. Agrupación de empleados ("clustering")

- El departamento de RRHH de una empresa desea categorizar a sus empleados en distintos grupos con el objetivo de entender mejor su comportamiento y tratarlos de manera adecuada

Ejemplo 4. Datos

Id	Sueldo	Casado	Coche	Hijos	Alq/Prop	Sindicado	Bajas	Antigüedad	Sexo
1	1000	Si	No	0	Alq	No	7	15	H
2	2000	No	Si	1	Alq	Si	3	3	M
3	1500	Si	Si	2	Prop	Si	5	10	H
4	3000	Si	Si	1	Alq	No	15	7	M
5	1000	Si	Si	0	Prop	Si	1	6	H
...	...	...	...	...		...	...	...	...

Ejemplo 4. Conocimiento obtenido

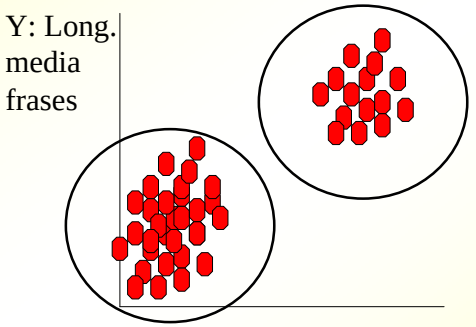
	GRUPO 1	GRUPO 2	GRUPO 3
Sueldo	1535	1428	1233
Casado (No/Si)	77%/22%	98%/2%	0%/100%
Coche	82%/18%	1%/99%	5%/95%
Hijos	0.05	0.3	2.3
Alq/Prop	99%/1%	75%/25%	17%/83%
Sindicado	80%/20%	0%/100%	67%/33%
Bajas	8.3	2.3	5.1
Antigüedad	8.7	8	8.1
Sexo (H/M)	61%/39%	25%/75%	83%/17%

Ejemplo 4. Conocimiento obtenido

- Grupo 1: sin hijos y con vivienda de alquiler. Poco sindicados. Muchas bajas
- Grupo 2: sin hijos y con coche. Muy sindicados. Pocas bajas. Normalmente son mujeres y viven en alquiler
- Grupo 3: con hijos, casados y con coche. Mayoritariamente hombres propietarios. Poco sindicados.

Idea general de agrupación

- Detectar agrupaciones naturales en los datos
- Agrupación (o "clustering") = aprendizaje no supervisado: se parte de una tabla, como en clasificación, pero sin la clase



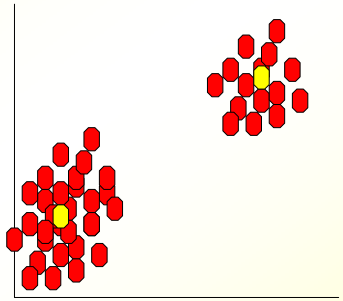
Ejemplo: clustering de libros. 2 grupos:

* Palabras y frases largas (¿filosofía?)

* Palabras y frases cortas (¿novela?)

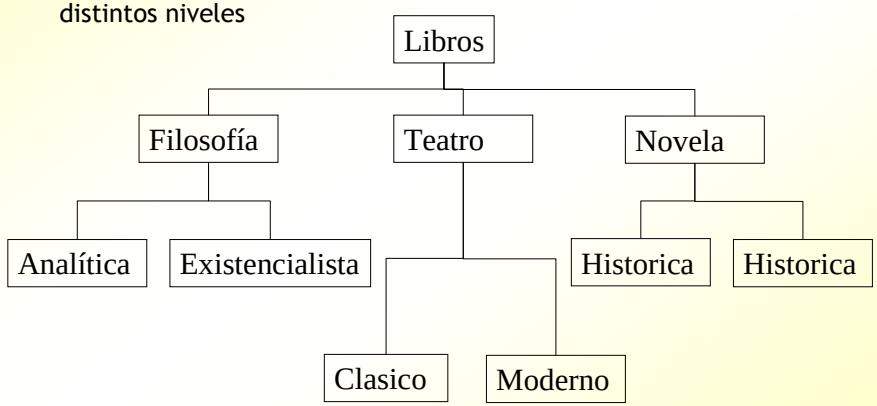
Representación de clusters

- Por sus centroides (ej: algoritmo k-medias)
- La pertenencia a un cluster puede ser probabilística (ej: algoritmo EM)



Representación de clusters

- Jerárquica (ej: algoritmo cobweb)
- Nota: las etiquetas "filosofía", "clásico", etc. aparecen sólo a título indicativo. El sistema simplemente detectaría distintos grupos a distintos niveles



Aplicaciones de Minería de Datos (técnica de carácter horizontal)

- Financieras y banca
  - Obtención de patrones de uso fraudulento de tarjetas de crédito
  - Predicción de devolución de créditos
- Análisis de mercado:
  - Análisis de cesta de la compra
  - Análisis de fidelidad de clientes. Reducción de fuga
  - Segmentación de clientes
- Seguros y salud privada: determinación de clientes potencialmente caros
- Educación: detección de abandonos
- Industria: Predicción de la demanda eléctrica, de gas, etc.

Aplicaciones II

- Medicina: diagnóstico de enfermedades (ej: diagnóstico de dolor abdominal)
- Ciencia:
  - Análisis de secuencias de proteínas
  - Predecir si un compuesto químico causa cáncer
  - Predecir si una persona puede tener potencialmente una enfermedad a partir de su DNA
  - Clasificación de cuerpos celestes (SKYCAT)
- Internet:
  - Detección de spam (SpamAssassin, bayesiano)
  - Web: asociar libros que compran usuarios en e-tiendas (amazon.com)

- **FASES EN MINERÍA DE DATOS**

Fases del proceso de extracción de conocimiento

1. Integración y recopilación de datos
2. Selección, limpieza y transformación -> Datos
3. **Minería de datos -> Patrones (ej: clasificador)**
4. Evaluación e interpretación -> Conocimiento
5. Difusión y uso -> Decisiones

Integración y recopilación

- Almacenes de datos (data warehousing, bases de datos): repositorio de información obtenido de diversas fuentes (heterogéneas), almacenada bajo un esquema unificado. En esta clase usaremos una simple tabla

IDC	Años	Euros	Salario	Casa propia	Cuentas morosas	...	Devuelve el crédito
101	15	60000	2200	Si	2	...	No
102	2	30000	3500	Si	0	...	Si
103	9	9000	1700	Si	1	...	No
104	15	18000	1900	No	0	...	Si
105	10	24000	2100	No	0	...	No

Preproceso: selección, limpieza, transformación

- Datos:
  - Valores que no se ajustan al comportamiento general (*outliers*): eliminar o dejar
  - Muestreo de datos (si hay muchos)
- Atributos:
  - Valores faltantes (*missing values*)
  - Eliminar atributos redundantes o irrelevantes (ej: sueldo y clase social)
  - Calcular nuevos atributos que sean más relevantes (area, población -> densidad de población)
  - Discretización, numerización, normalización, …

- **TIPOS DE ALGORITMOS PARA PREDICCIÓN (CLASIFICACIÓN Y REGRESIÓN)**

Datos de entrada. Ejemplo clasificación

Cielo	Temperatura	Humedad	Viento	Tenis
Sol	85	85	No	No
Sol	80	90	Si	No
Nublado	83	86	No	Si
Lluvia	70	96	No	Si
Lluvia	68	80	No	Si
Lluvia	65	70	Si	No
Nublado	64	65	Si	Si
Sol	72	95	No	No
Sol	69	70	No	Si
Lluvia	75	80	No	Si
Sol	75	70	Si	Si
Nublado	72	90	Si	Si
Nublado	81	75	No	Si
Lluvia	71	91	Si	No

Esquema general en predicción/clasificación

Cielo	Temperatura	Humedad	Viento	Tenis
Sol	85	85	No	No
Sol	80	90	Si	No
Nublado	83	86	No	Si
Lluvia	70	96	No	Si
Lluvia	68	80	No	Si
Lluvia	65	70	Si	No
Nublado	64	65	Si	Si
Sol	72	95	No	No
Sol	69	70	No	Si
Lluvia	75	80	No	Si
Sol	75	70	Si	Si
Nublado	72	90	Si	Si
Nublado	81	75	No	Si
Lluvia	71	91	Si	No

Cielo	Tempe ratura	Humedad	Viento	Tenis
Sol	60	65	No	?????

Algoritmo MD

IF Cielo = Sol **Y** Humedad <= 75

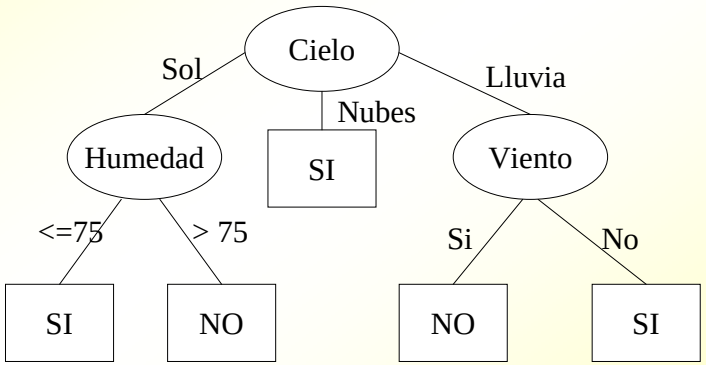
THEN Tenis = Si ...

Clase = Si

Algoritmos de clasificación / regresión (predicción)

- Árboles de decisión: ID3, C4.5 (J48), ...
- Árboles de regresión: LMT (M5), ...
- Reglas: PART, CN2, AQ, ...
- Funciones: redes de neuronas, regresión logística, máquinas de vectores de soporte (SMO), ...
- Técnicas perezosas: IB1, IBK, ...
- Técnicas Bayesianas: Naive Bayes
- Metatécnicas

Árboles de decisión (para clasificación)



Algoritmos de construcción de árboles de decisión

- El más básico es el ID3: construye árboles de decisión de manera recursiva, de la raíz hacia las hojas, seleccionando en cada momento el mejor nodo para poner en el árbol
- El C4.5 (o J48), trata con valores continuos y utiliza criterios estadísticos para impedir que el árbol se sobreadapte (que "crezca demasiado", que se aprenda los datos en lugar de generalizar)

Algoritmo ID3 simplificado

1. Detener la construcción del árbol si:
   1. Todos los ejemplos pertenecen a la misma clase
   2. Si no quedan ejemplos o atributos
2. Si no, elegir el mejor atributo para poner en ese nodo (el que minimice la entropía media)
3. Crear de manera recursiva tantos subárboles como posibles valores tenga el atributo seleccionado

Algoritmo ID3 detallado

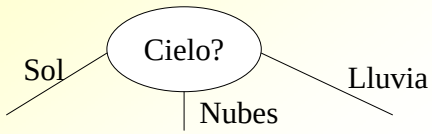
- ID3(Ejemplos, Atributo-objetivo, Atributos)

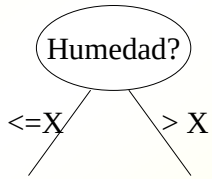
```
1. Si todos los Ejemplos son positivos, devolver un nodo etiquetado con +
2. Si todos los Ejemplos son negativos, devolver un nodo etiquetado con -
3. Si Atributos está vacío, devolver un nodo etiquetado con el valor más
   frecuente de Atributo-objetivo en Ejemplos.
4. En otro caso:
   4.1. Sea A el atributo de Atributos que MEJOR clasifica Ejemplos
   4.2. Crear Árbol, con un nodo etiquetado con A.
   4.3. Para cada posible valor v de A, hacer:
     * Añadir un arco a Árbol, etiquetado con v.
     * Sea Ejemplos(v) el subconjunto de Ejemplos con valor
       del atributo A igual a v.
     * Si Ejemplos(v) es vacío:
       - Entonces colocar debajo del arco anterior un nodo etiquetado con
         el valor más frecuente de Atributo-objetivo en Ejemplos.
       - Si no,  colocar debajo del arco anterior el subárbol
         ID3(Ejemplos(v), Atributo-objetivo, Atributos-{A}).
   4.4  Devolver Árbol
```

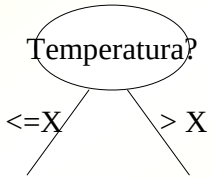
Algoritmo C4.5 simplificado

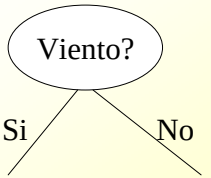
1. Detener la construcción del árbol si:
   1. Todos los ejemplos pertenecen a la misma clase
   2. Si no quedan ejemplos o atributos
   3. Si no se espera que se produzcan mejoras continuando la subdivisión
2. Si no, elegir el mejor atributo para poner en ese nodo (el que minimice la entropía media)
3. Crear de manera recursiva tantos subárboles como posibles valores tenga el atributo seleccionado

¿Qué nodo es el mejor para poner en la raíz del árbol?



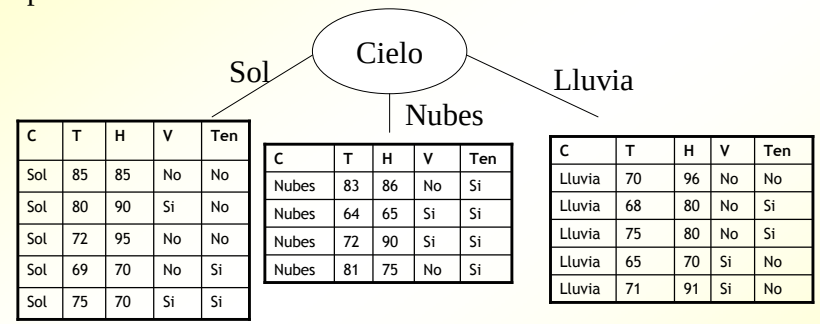






Supongamos que usamos Cielo

Cielo nos genera tres particiones de los datos, tantas como valores posibles tiene



C	T	H	V	Ten
Sol	85	85	No	No
Sol	80	90	Si	No
Sol	72	95	No	No
Sol	69	70	No	Si
Sol	75	70	Si	Si

C	T	H	V	Ten
Nubes	83	86	No	Si
Nubes	64	65	Si	Si
Nubes	72	90	Si	Si
Nubes	81	75	No	Si

C	T	H	V	Ten
Lluvia	70	96	No	No
Lluvia	68	80	No	Si
Lluvia	75	80	No	Si
Lluvia	65	70	Si	No
Lluvia	71	91	Si	No

"3 No, 2 Si" — Tendencia al "no"

"0 No, 4 Si" — Partición perfecta

"3 No, 2 Si" — Tendencia al "no"

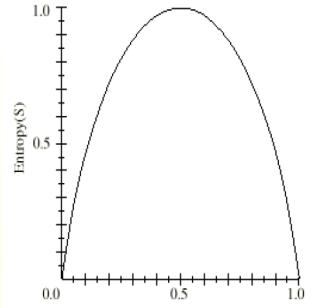
¿Cómo medimos lo bueno que es Cielo como atributo para clasificar?

- Usaremos una medida que obtenga el mejor valor cuando el atributo me obtenga particiones lo mas homogéneas posible, en media
  - Homogénea: "0 No, todo Si"; o bien "todo No, 0 Si"
  - Indecisión: "50% No, 50% Si"
- Una medida que me dice lo lejana que está una partición de la perfección es la entropía

$$H(P) = -\sum_{Ci} p_{Ci} \log_2(p_{Ci})$$

$$H(P) = -(p_{si} \log_2(p_{si}) + p_{no} \log_2(p_{no}))$$

$$p_{no} = (1 - p_{si})$$



- A mayor entropía, peor es la partición

Entropía media de Cielo

- Cielo genera tres particiones cuya entropía es:
  1. "3 No, 2 Si": H=-((3/5)*log$_2$(3/5)+(2/5)*log$_2$ (2/5) = 0.97
  2. "0 No, 4 Si": H=-((0/4)*log$_2$ (0/4)+1*log$_2$ (1))=0
  3. "3 No, 2 Si": H=-((3/5)*log$_2$ (3/5)+(2/5)*log$_2$ (2/5))= 0.97

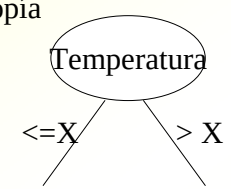
La entropía media ponderada de Cielo será:

- HP=(5/14)*0.97+(4/14)*0+(5/14)*0.97 = **0.69**
- Nota: hay 14 datos en total

¿Y si el atributo es contínuo?

Hay que partir por el valor X, donde sea mas conveniente, minimizando la entropía

Nota: solo hemos probado algunas de las posibles particiones, entre las que se encuentra la mejor



X<=70

64 – Si, 65 – No, 68 – Si, 69 – Si, 70 – Si, | 71 – No, 72 – No Si, 75 – Si Si, 80 – No, 81 – Si, 83 – Si, 85 - No

1 No, 4 Si — 4 No, 5 Si — HP = 0.89

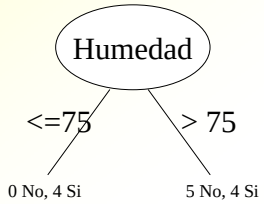
64 – Si, 65 – No, 68 – Si, 69 – Si, 70 – Si, 71 – No, 72 – No Si | 75 – Si Si, 80 – No, 81 – Si, 83 – Si, 85 - No

3 No, 5 Si — 2 No, 4 Si — HP = 0.93

64 – Si, 65 – No, 68 – Si, 69 – Si, 70 – Si, 71 – No, 72 – No Si, 75 – Si Si | 80 – No, 81 – Si, 83 – Si, 85 - No

3 No, 7 Si — 2 No, 2 Si — HP = 0.91

Caso de humedad

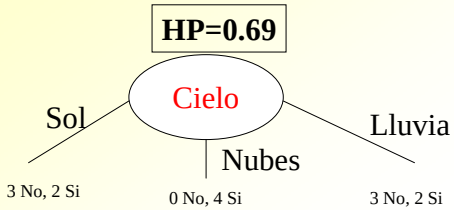


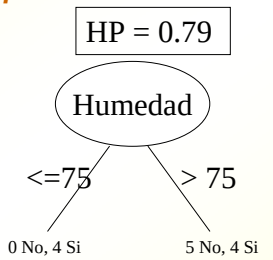
65-Si, 70-No Si Si, 75–Si, 80-Si Si, 85-No, 86-Si, 90-No Si, 91-No, 95-No, 96-Si,

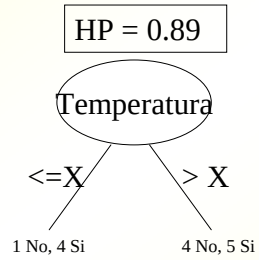
1 No, 6 Si | 4 No, 3 Si | HP = 0.79

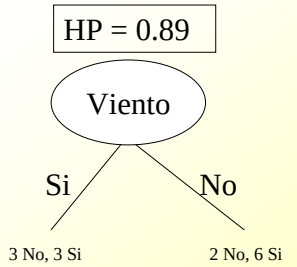
Nota: hay otras posibilidades de particiones, pero esta es la mejor

¿Qué nodo es el mejor para poner en la raíz?



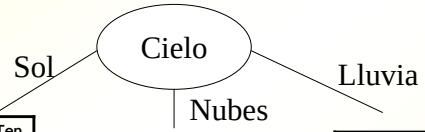






Construcción recursiva del árbol

Ahora que ya tenemos el nodo raíz, el proceso continua recursivamente: hay que construir tres subárboles con los datos que se muestran en cada rama



C	T	H	V	Ten
Sol	85	85	No	No
Sol	80	90	Si	No
Sol	72	95	No	No
Sol	69	70	No	Si
Sol	75	70	Si	Si

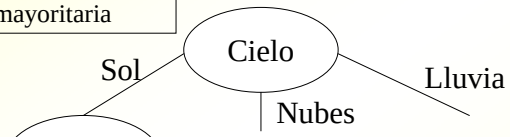
C	T	H	V	Ten
Nubes	83	86	No	Si
Nubes	64	65	Si	Si
Nubes	72	90	Si	Si
Nubes	81	75	No	Si

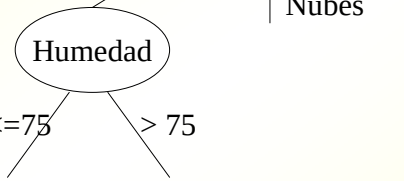
C	T	H	V	Ten
Lluvia	70	96	No	No
Lluvia	68	80	No	Si
Lluvia	75	80	No	Si
Lluvia	65	70	Si	No
Lluvia	71	91	Si	No

"3 No, 2 Si" "0 No, 4 Si" "3 No, 2 Si"

Construcción recursiva del árbol

Aquí un criterio estadístico determina que no merece la pena seguir subdividiendo y se asigna la clase mayoritaria



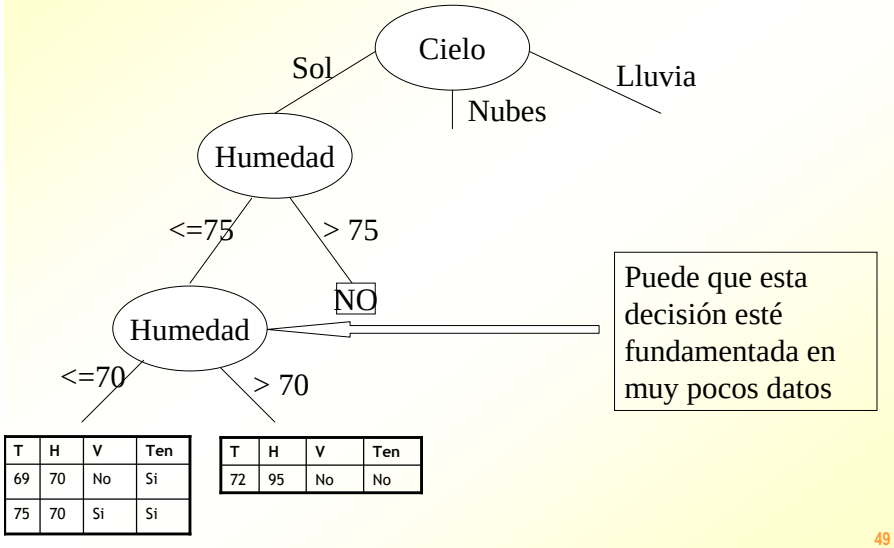


T	H	V	Ten
72	95	No	No
69	70	No	Si
75	70	Si	Si

T	H	V	Ten
85	85	No	No
80	90	Si	No

Aquí no es necesario seguir subdividiendo porque todos los datos son de la misma clase

¿Porqué no seguir subdividiendo?



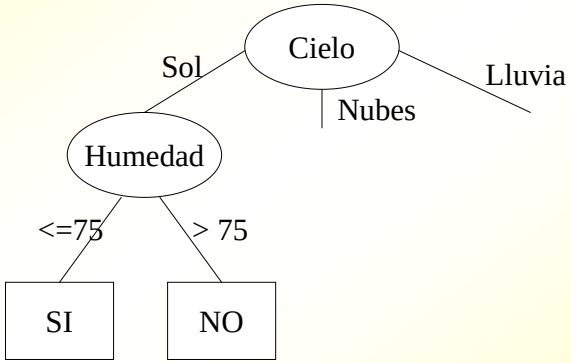
T	H	V	Ten
69	70	No	Si
75	70	Si	Si

T	H	V	Ten
72	95	No	No

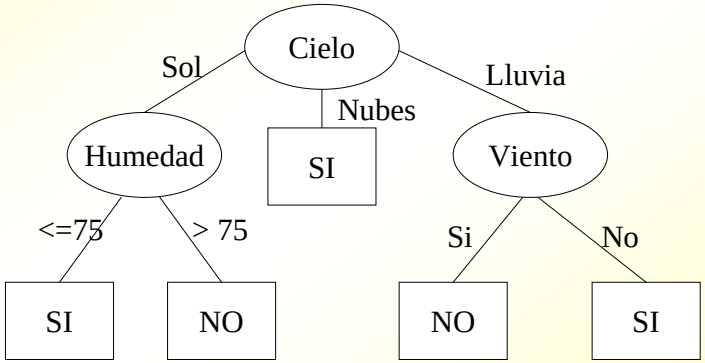
¿Porqué no seguir subdividiendo?

- ¿Hay que detener la construcción cuando tenemos “1 no, 2 si”?
- Tal vez. Cuando hay tan pocos datos (y suponiendo que haya ruido) es posible que el “1 no” haya aparecido por azar, e igualmente podriamos tener “2 no, 2 si”
- Pasamos de una situación en la que hay mayoría de “si” a otra en la que están equiparados con los “no”
- Se puede utilizar algún criterio estadístico para saber si es probable que “1 no, 2 si” se deba al azar
- Cuando se manejan pocos datos (3 en este caso), es bastante probable que las regularidades (humedad<=70 en ete caso) sean sólo aparentes y se deban al azar

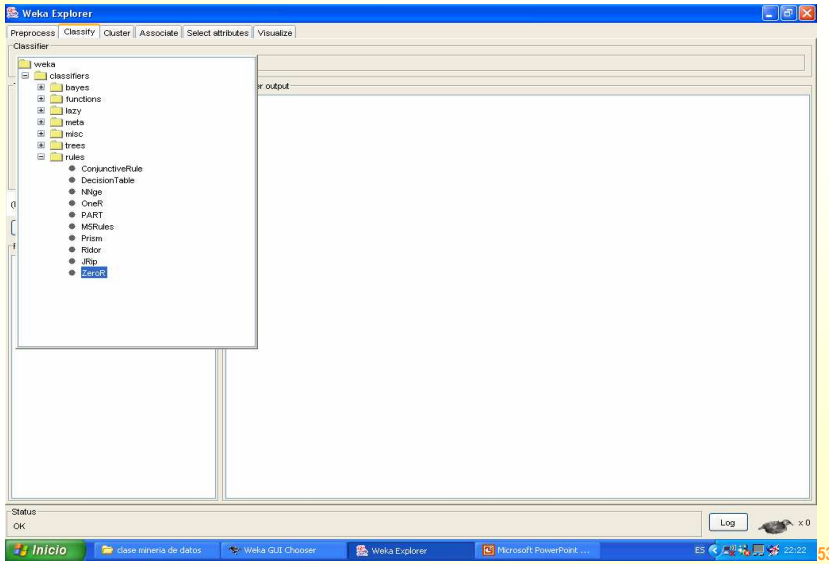
Construcción recursiva del árbol



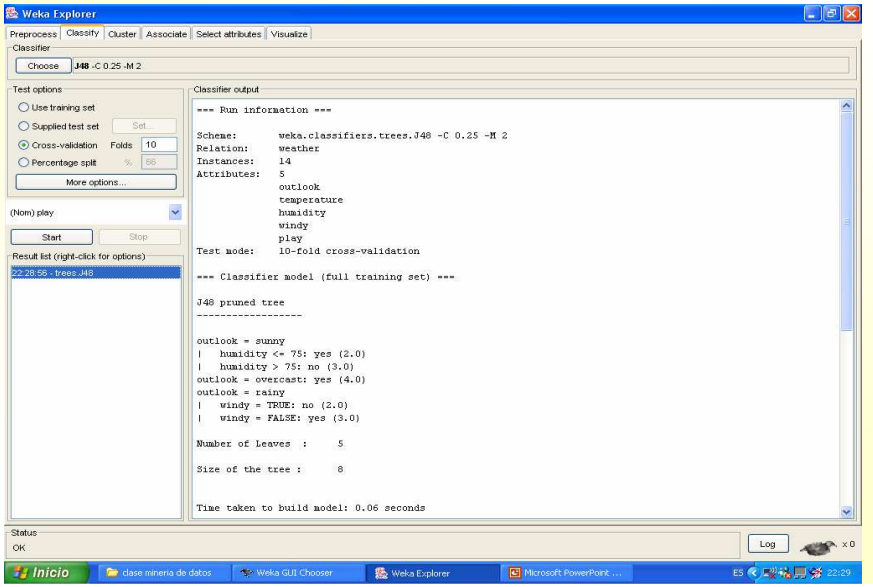
Construcción recursiva del árbol



Algoritmos de predicción (**classifiers**) en Weka

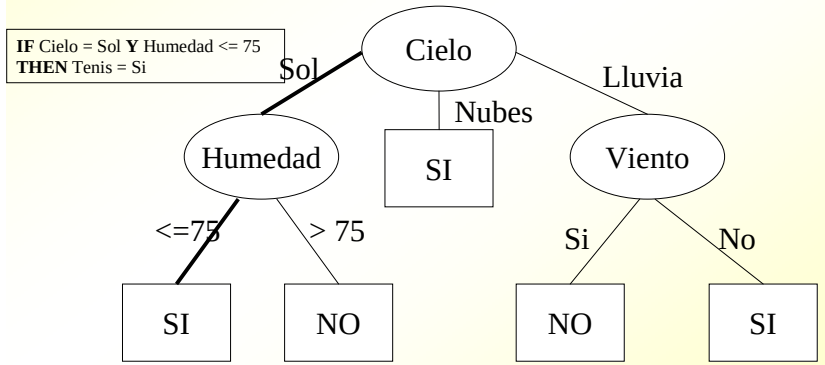


Arbol de decisión J48



Reglas (para clasificación)

- Podemos generar reglas a partir de árboles, creando una regla para cada camino que vaya de la raíz a las hojas



Reglas (para clasificación)

IF Cielo = Sol

Humedad <= 75 **THEN** Tenis = Si

ELSE IF Cielo = Sol

Humedad > 75 **THEN** Tenis = No

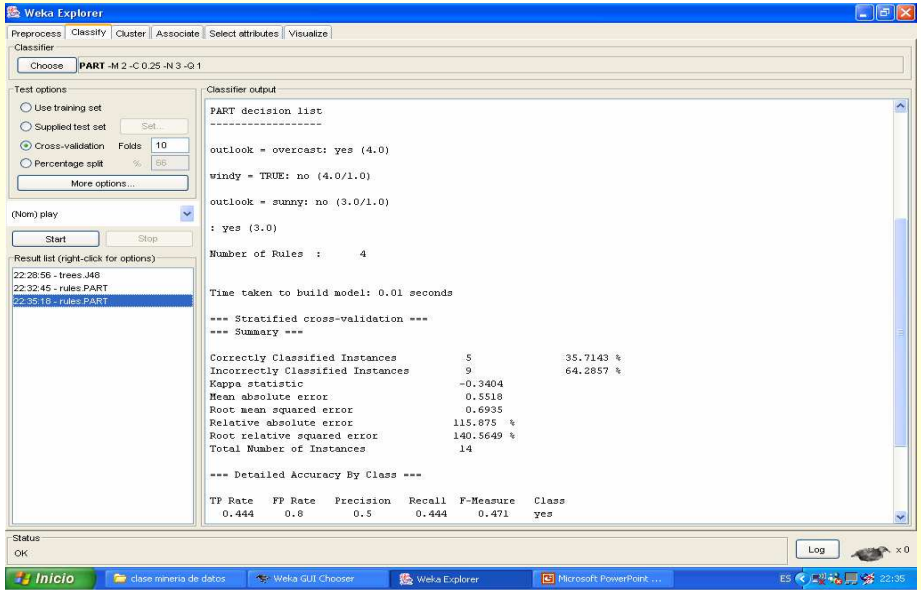
ELSE IF Cielo = Nubes **THEN** Tenis = Si

ELSE IF Cielo = Lluvia

Viento = Si **THEN** Tenis = Si

ELSE Tenis = No

Algoritmo de reglas PART



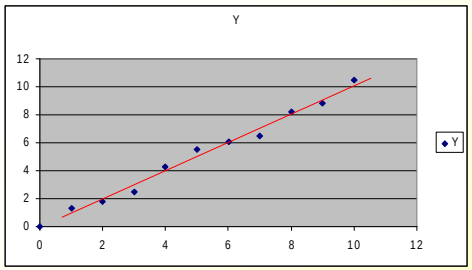
Funciones (para regresión)

X	Y
0	0
1	1,3
2	1,8
3	2,5
4	4,3
5	5,5
6	6,1
7	6,5
8	8,2
9	8,8
10	10,5

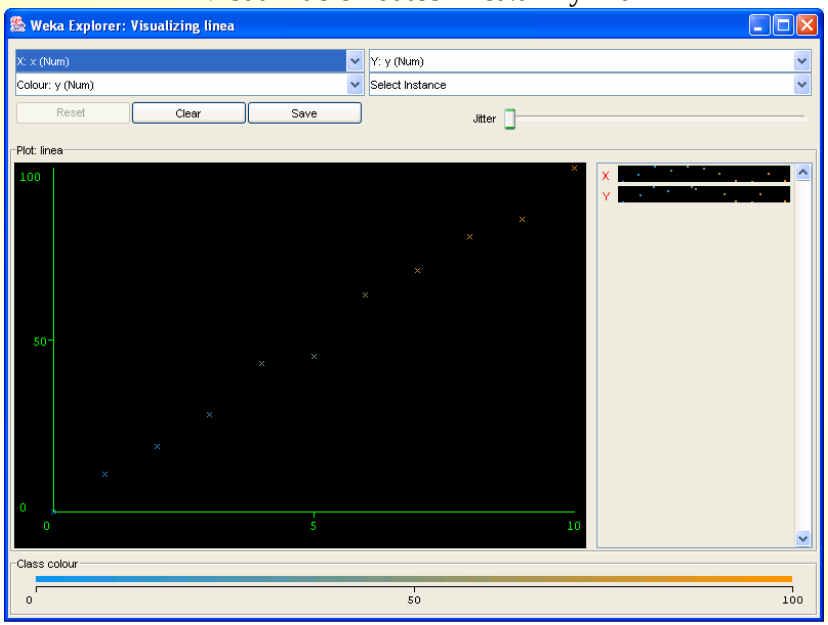
Y = 1 * X

Caso general (regresión lineal)

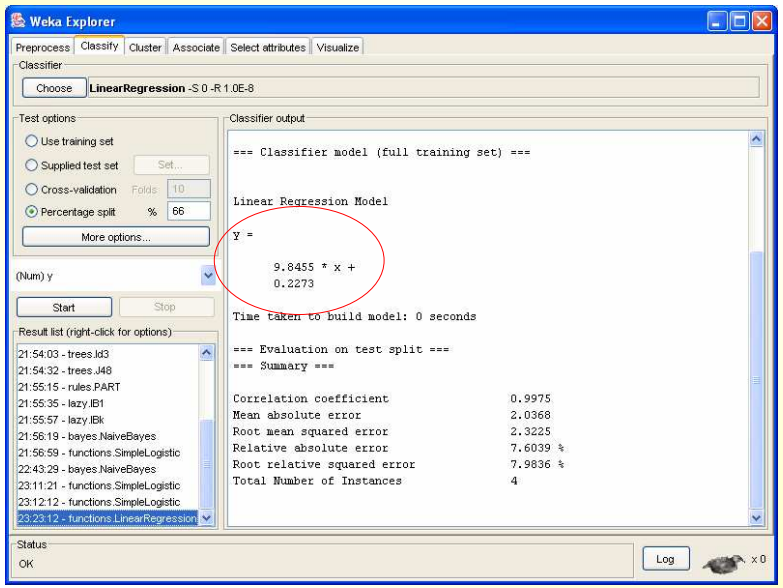
Y = A1*X1 + A2*X2 + A3*X3 + A4



Visualización datos linea.arff y=10*x

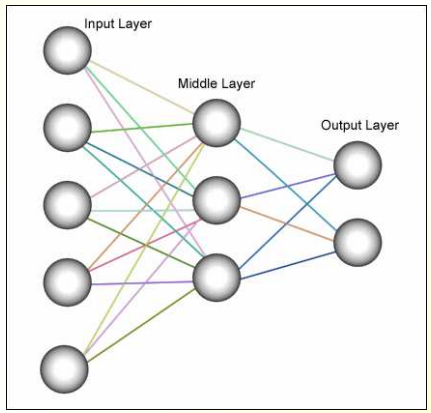


Resultados regresión lineal

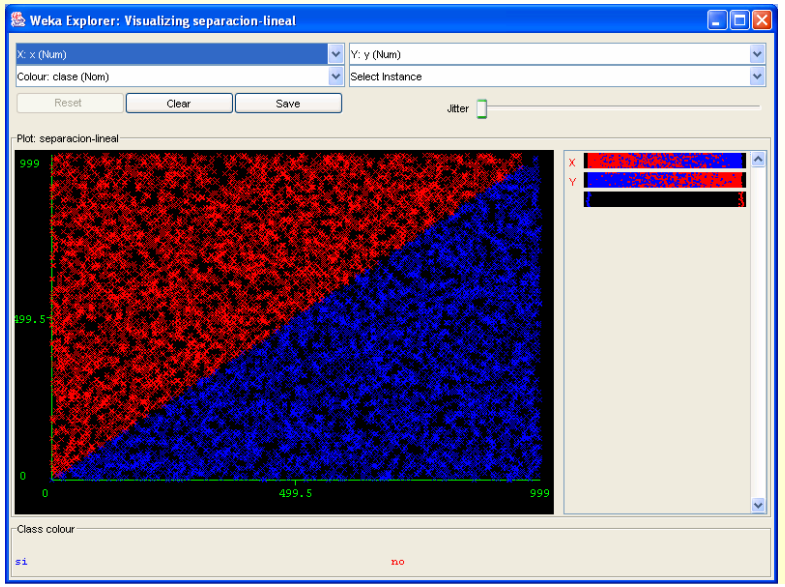


Funciones: redes de neuronas (regresión no lineal)

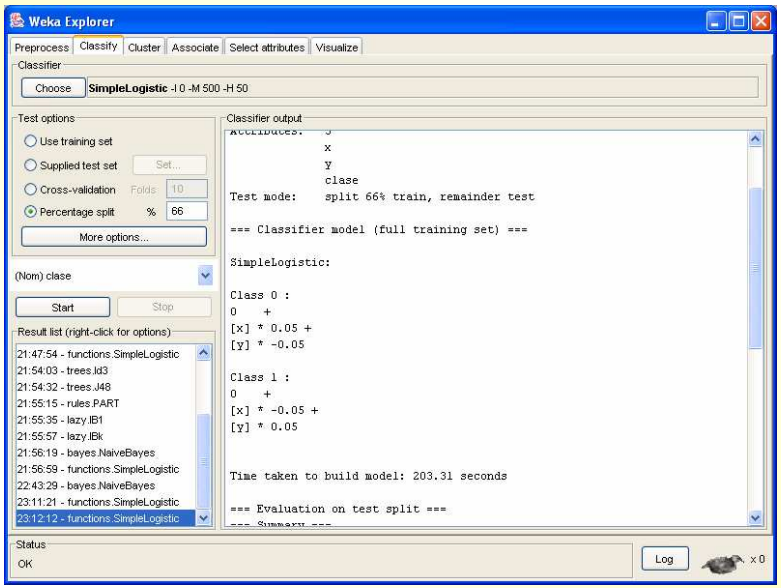
Algoritmo de retropropagación del gradiente (propagan el error hacia atrás. Calculan los pesos con el objetivo de minimizar el error)



Visualización datos separables linealmente



Algoritmo simple logistics (separación lineal)

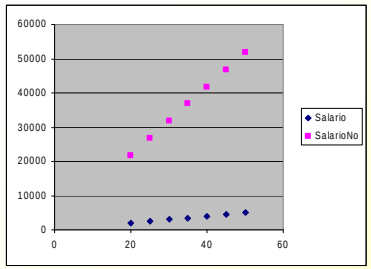


Árboles de regresión

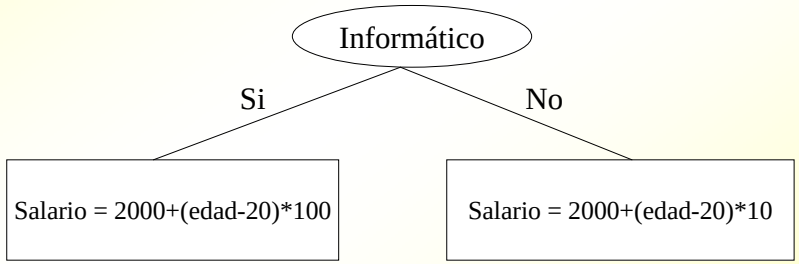
- ¿Y si tenemos atributos nominales y numéricos y queremos predecir cantidades numéricas (regresión)?
- Usar árboles de regresión: tienen funciones (regresión lineal) en las hojas

Árboles de regresión. Ejemplo

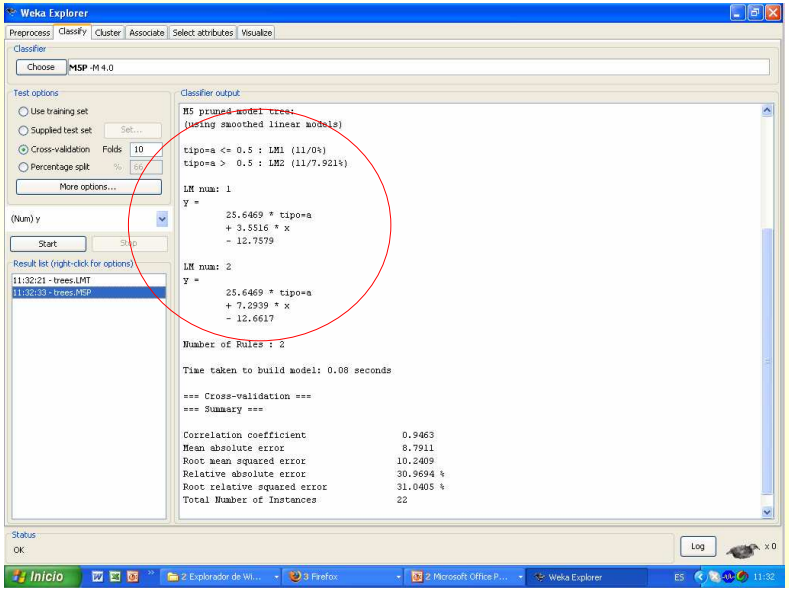
Informático	Edad	Salario
Si	20	2000
Si	25	2500
Si	30	3000
Si	35	3500
Si	40	4000
Si	45	4500
Si	50	5000
No	20	2000
No	25	2050
No	30	2100
No	35	2150
No	40	2200
No	45	2250
No	50	2300



Árboles de regresión. Ejemplo



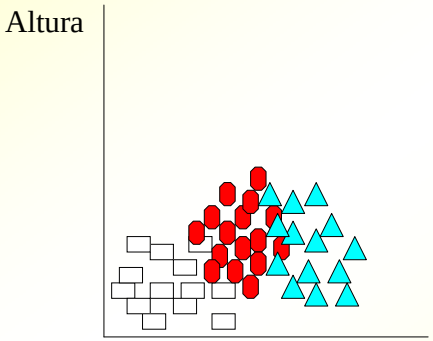
Algoritmo MSP



Técnicas "perezosas" (almacenar instancias)

- También llamadas técnicas basadas en instancias (o en ejemplos)
- En lugar de construir una estructura predictora (árbol de decisión, reglas, ...), **simplemente** se guardan las instancias (los datos) o representantes de los mismos
- Para clasificar un nuevo dato, simplemente se busca(n) la(s) instancia(s) más "parecida(s)" o cercana(s)
- Parecido a lo que hacen las personas: para resolver un nuevo problema, intentan recordar el caso más parecido que ya sepan resolver
- Ejemplo: sistema legal anglosajón

Técnicas perezosas (clasificación)



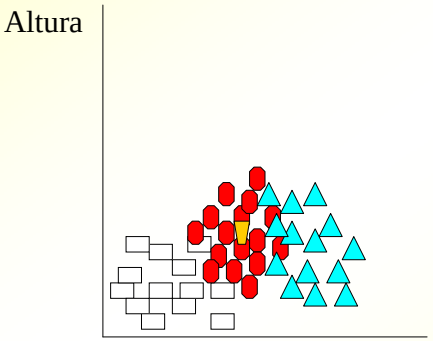
Niño

Adulto

Anciano

Se guardan todos los ejemplos

Técnicas perezosas (clasificación)



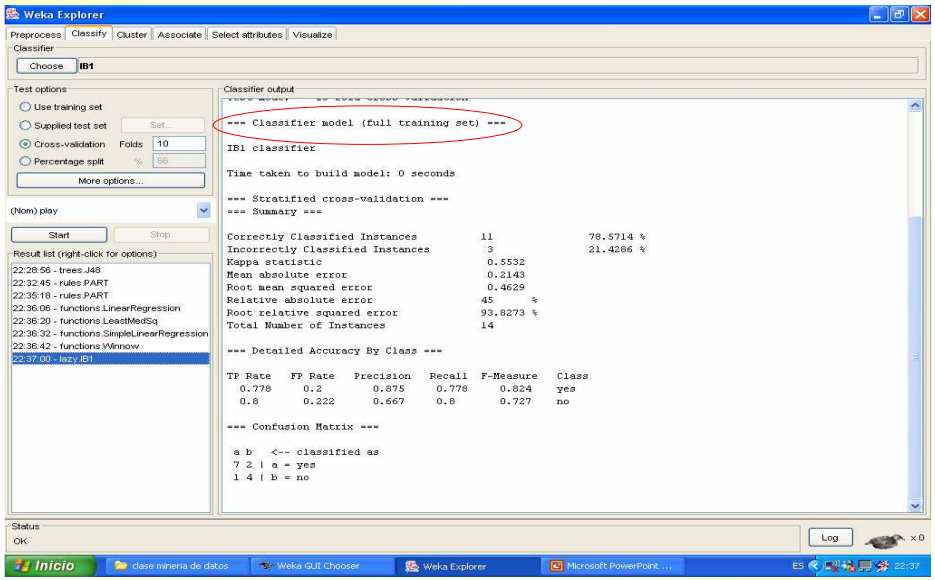
Niño

Adulto

Anciano

Se guardan todos los ejemplos

Algoritmo perezoso IB1 (1 vecino) e IBK (k vecinos)



Técnicas perezosas en regresión

- Ejemplo: predicción de la carga de electricidad según la hora y la temperatura
- Ahora se trata de un problema de regresión
- Simplemente se cogen las n instancias más cercanas y se calcula la media entre ellas
- El IBK permite hacer regresión (el IB1 no)

Técnicas Bayesianas (almacenar probabilidades)

- (sol, frío, alta, si, clase=????)
- ¿Pr(Tenis = si / cielo = sol, temperatura = frío, humedad = alta, viento = si)?
- ¿Pr(Tenis = no / cielo = sol, temperatura = frío, humedad = alta, viento = si)?
- Naive Bayes:

  Pr(si / sol, frío, alta, si) ~ pr(cielo = sol/si) * pr(humedad = alta /si) * pr(viento = si /si) * pr(si)

- Naive Bayes es técnicamente correcto sólo si se supone que los atributos son independientes (aunque funciona también en casos que no lo son, que son la mayoría)

Teorema de Bayes y Naive Bayes

- Pr(A|B) = k*Pr(B|A)*P(A)
- Pr(Tenis = si / cielo = sol, temperatura = frío, humedad = alta, viento = si)
- = k * Pr(cielo = sol, temperatura = frío, humedad = alta, viento = si / Tenis = si) * Pr(Tenis = si)
- Y si suponemos a todos los atributos independientes
- = Pr(si / ...) = k* pr(cielo = sol/si) * pr(humedad = alta /si) * pr(viento = si /si) * Pr(si)
- Esto implica que el que haga sol es independiente de la humedad (lo que no es cierto, pero suele funcionar)
- En suma, que podemos calcular Pr(si/...) a partir de:
  - pr(cielo = sol/ tenis=si) = numero de días soleados y buenos para el tenis dividido por el número de días buenos para el tenis
  - pr(humedad = alta /tenis=si)
  - pr(viento = si /tenis=si)
  - pr(tenis=si) = número de días buenos para el tenis dividido por el número de días totales

Datos de entrada

Día	Cielo	Temperatura	Humedad	Viento	Tenis
1	Soleado	Caliente	Alta	No	No
2	Soleado	Caliente	Alta	Si	No
3	Nublado	Caliente	Alta	No	Si
4	Lluvioso	Templado	Alta	No	Si
5	Lluvioso	Frio	Normal	No	Si
6	Lluvioso	Frio	Normal	Si	No
7	Nublado	Frio	Normal	Si	Si
8	Soleado	Templado	Alta	No	No
9	Soleado	Frio	Normal	No	Si
10	Lluvioso	Templado	Normal	No	Si
11	Soleado	Templado	Normal	Si	Si
12	Nublado	Templado	Alta	Si	Si
13	Nublado	Caliente	Normal	No	Si
14	Lluvioso	Templado	Alta	Si	No

Datos de entrada ordenados

Cielo	Temperatura	Humedad	Viento	Tenis
Soleado	Frio	Normal	No	Si
Soleado	Templado	Normal	Si	Si
Nublado	Frio	Normal	Si	Si
Nublado	Caliente	Alta	No	Si
Nublado	Templado	Alta	Si	Si
Nublado	Caliente	Normal	No	Si
Lluvioso	Templado	Alta	No	Si
Lluvioso	Frio	Normal	No	Si
Lluvioso	Templado	Normal	No	Si
Soleado	Caliente	Alta	No	No
Soleado	Templado	Alta	No	No
Soleado	Caliente	Alta	Si	No
Lluvioso	Frio	Normal	Si	No
Lluvioso	Templado	Alta	Si	No

Técnicas Bayesianas. Ejemplo

P(Cielo/Tenis)

Cielo	Si	No
Sol	**2/9**	3/5
Nubes	4/9	0/5
Lluvia	3/9	2/5

P(Temp/Tenis)

Temperatura	Si	No
Caliente	2/9	2/5
Templado	4/9	2/5
Frio	**3/9**	1/5

P(Hum/Tenis)

Humedad	Si	No
Alta	**3/9**	4/5
Normal	6/9	1/5

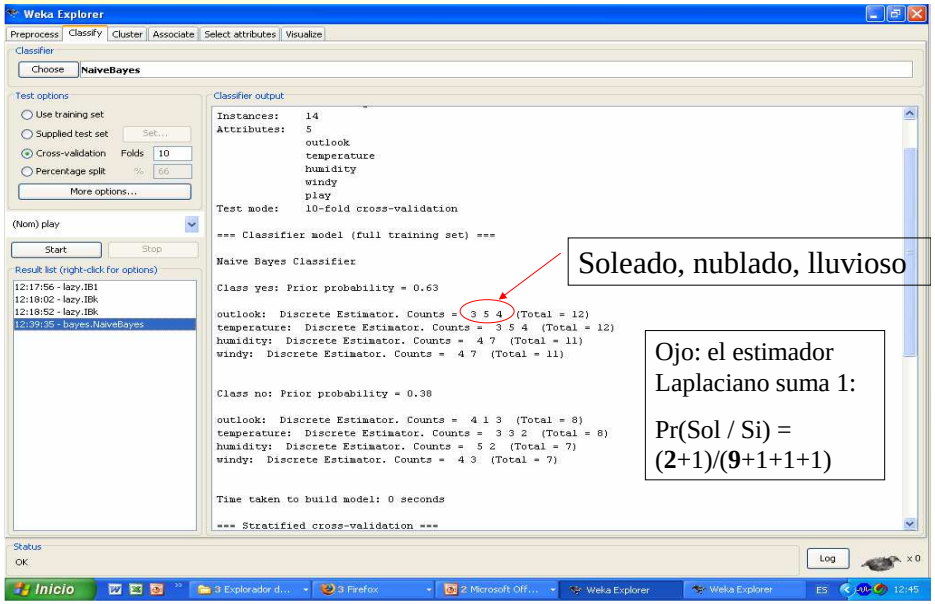
P(Tenis)

Tenis	Si	No
	9/14	5/14

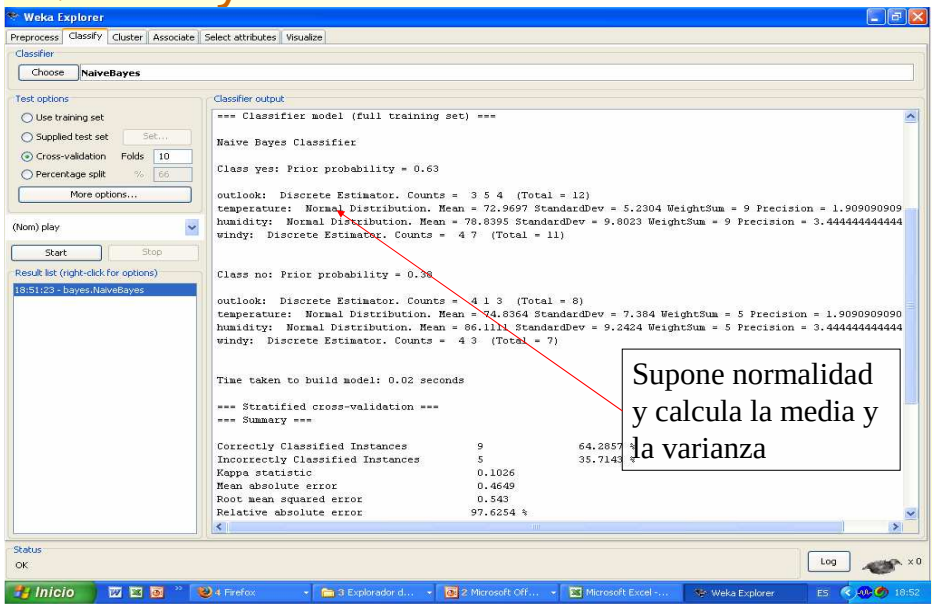
Viento	Si	No
Si	**3/9**	3/5
No	6/9	2/5

- Pr(tenis=si / sol, frío, alta, si) ~ Pr(sol/si)*Pr(frio/si)*Pr(alta/si)*Pr(vientosi/si)*Pr(tenis=si)
- Pr(tenis=no / sol, frío, alta, si) ~ Pr(sol/no)*Pr(frio/no)*Pr(alta/no)*Pr(si viento/no)*Pr(tenis=no)
- Pr(si / sol, frío, alta, si) ~ 2/9 * 3/9 * 3/9 * 3/9 * 9/14 = 0.0053
- Pr(no / sol, frío, alta, si) ~ 3/5 * 1/5 * 4/5 * 3/5 * 5/14 = **0.0206**

Algoritmo Naive Bayes



Naive Bayes con atributos numéricos



Nombres de algoritmos

- Árboles de decisión: ID3, C4.5 (J48), ...
- Árboles de regresión: LMT (M5), ...
- Reglas: PART, CN2, AQ, ...
- Funciones: redes de neuronas, regresión logística, máquinas de vectores de soporte (SMO), ...
- Técnicas perezosas: IB1, IBK, ...
- Técnicas Bayesianas: Naive Bayes

EVALUACIÓN DEL CONOCIMIENTO MINADO

Evaluación

- Una vez obtenido el conocimiento es necesario validarlo para observar su comportamiento con datos no vistos
- Ejemplo: si a un alumno se le evalúa (examen) con los mismos problemas con los que aprendió, no se demuestra su capacidad de generalización
- Solución: dividir el conjunto de datos en un subconjunto para entrenamiento (70%) y otro para test (30%)
- Problema: es posible que por azar, los datos de entrenamiento y test estén sesgados
  - Ejemplo de sesgo: Sea un problema para determinar qué tipo de personas compran aparatos de DVD. Puede ocurrir por casualidad que en los datos de entrenamiento aparezcan muchas mas mujeres que hombres. El sistema creerá que hay una correlación entre el sexo y la clase.

Evaluación: entrenamiento y test múltiples veces

- Consiste en partir el conjunto de datos totales múltiples veces y calcular el porcentaje de aciertos medio
- La idea es que los sesgos de unas y otras particiones se cancelen
- Es conveniente que las particiones sean **estratificadas**
- **Método:**
  - **Repetir múltiples veces:**
    1. **Desordenar el conjunto de datos total aleatoriamente**
    2. **Escoger los primeros 70% para entrenamiento y construir el modelo con ellos**
    3. **Escoger los últimos 30% para el test y estimar el porcentaje de aciertos**
  - **Calcular el porcentaje de aciertos medio**

Particiones estratificadas

- La proporción entre las clases que existe en el conjunto de datos original, se intenta mantener en los conjuntos de entrenamiento y test
- Ejemplo: si en el conjunto original un 65% de los datos pertenecen a la clase positiva, la estratificación intentará que esa proporción se mantenga en entrenamiento y test

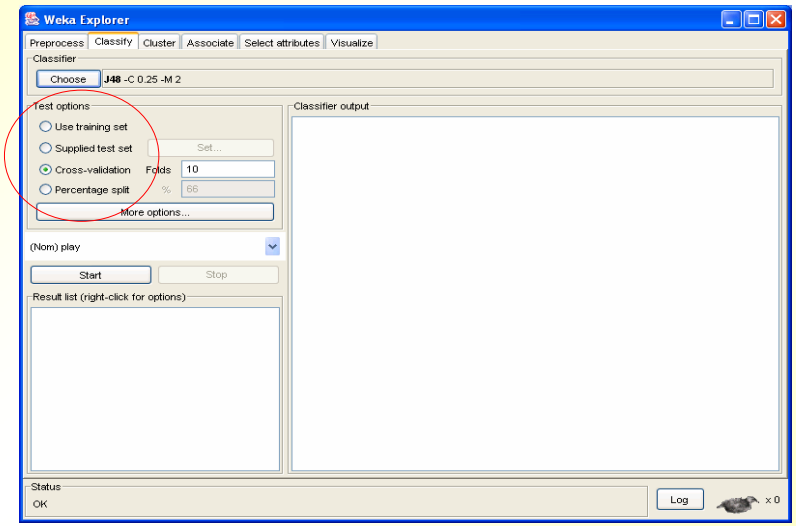
Particiones estratificadas

- La proporción entre las clases que existe en el conjunto de datos original, se intenta mantener en los conjuntos de entrenamiento y test
- Ejemplo: si en el conjunto original un 65% de los datos pertenecen a la clase positiva, la estratificación intentará que esa proporción se mantenga en entrenamiento y test

Validación cruzada

- Solución: dividir **varias veces** el mismo conjunto de datos en entrenamiento y test y calcular la media. Así es más complicado que todas las veces se produzcan sesgos
- Se divide el conjunto de datos original en k partes. Con k=3 tenemos los subconjuntos A, B, y C.
- Tres iteraciones:
  - Aprender con A, B y test con C (T1 = % aciertos con C)
  - Aprender con A, C y test con B (T2 = % aciertos con B)
  - Aprender con B, C y test con A (T3 = % aciertos con A)
  - % aciertos final T = (T1+T2+T3)/3
- El clasificador final CF se construye **con todos los datos (los tres conjuntos** A, B y C). Se supone que T es una estimación del porcentaje de aciertos de CF
- Se suele utilizar k=10

Métodos de evaluación: conjunto de entrenamiento, conjunto de test, validación cruzada, partición del conjunto de entrenamiento



Criterios básicos para evaluar

- En problemas de clasificación, si tenemos 2 clases (o M), el porcentaje de aciertos a superar es el 50% (o 100*1/M).
- De otra manera, sería mejor tirar una moneda (azar) que utilizar el clasificador para predecir
- En problemas de clasificación, si tenemos una clase con muchos más datos que otra, el porcentaje de aciertos a superar es el porcentaje de datos de la clase mayoritaria
- Ej: Sean dos clases (+ y -). Hay 90 datos + y 10 -. Un clasificador que prediga siempre + (independientemente de los atributos), ya acertará en un 90%. Hay que hacerlo mejor que eso.

Criterios básicos para evaluar. Coste

- En ocasiones el coste de fallar en una clase no es el mismo que fallar en otra
- Por ejemplo, para un clasificador de cáncer si/no, es preferible predecir que una persona tiene cáncer (sin tenerlo) que predecir que no lo tiene (teniéndolo)
- Ambos casos disminuyen el porcentaje de aciertos, pero lo primero tiene menos coste que lo segundo
- Para analizar esos casos es conveniente utilizar la matriz de confusión

Evaluación. La matriz de confusión y el coste

- Sea un problema con dos clases + y - (positivo y negativo)
- Los datos correctamente clasificados están en la diagonal, los incorrectos fuera de ella
  - El porcentaje de aciertos es (TP+TN)/(TP+TN+FN+FP)
  - El porcentaje de aciertos de + es: TP rate = TP / positivos = TP/(TP+FN)
  - El porcentaje de aciertos - es: TN rate = TN / negativos = TN/(FP+TN)

	Clasificado como +	Clasificado como -
Dato realmente +	**TP (true positive)**	FN (false negative)
Dato realmente -	FP (false positive)	**TN (true negative)**

De entre todos los datos positivos, cuantos clasificamos correctamente. Mide lo bien que acertamos en la clase +

Evaluación. La matriz de confusión y el coste

- Supongamos que en el problema de predecir cáncer si/no tenemos dos matrices de confusión. ¿Cuál es la mejor situación?
- Nótese que el % de aciertos es (90+60)/200 = 75% en los dos casos

	Clasificado como +	Clasificado como -
Dato realmente +	**TP 90**	FN 10
Dato realmente -	FP 40	**TN 60**

	Clasificado como +	Clasificado como -
Dato realmente +	**TP 60**	FN 40
Dato realmente -	FP 10	**TN 90**

Notese también que en los datos hay 100 personas con cáncer y 100 personas sin cáncer (sumar las líneas horizontales)

Evaluación. La matriz de confusión y el coste

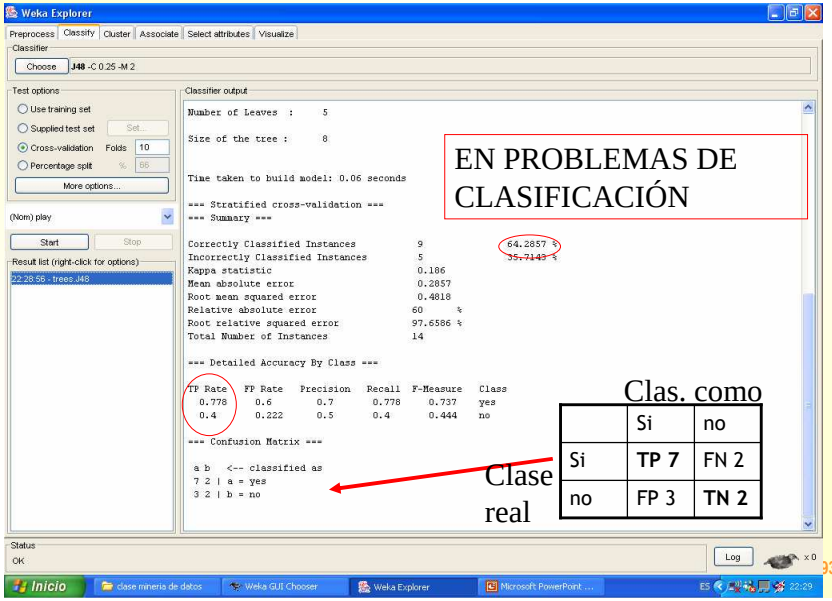
- En este caso es mejor disminuir el número de falsos negativos (pacientes que tienen cáncer, pero que el clasificador no lo detecta). O lo que es lo mismo, maximizar los TP.
- Es mejor el clasificador que nos de la matriz de la izquierda

	Clasificado como +	Clasificado como -
Dato realmente +	**TP 90**	FN 10
Dato realmente -	FP 40	**TN 60**

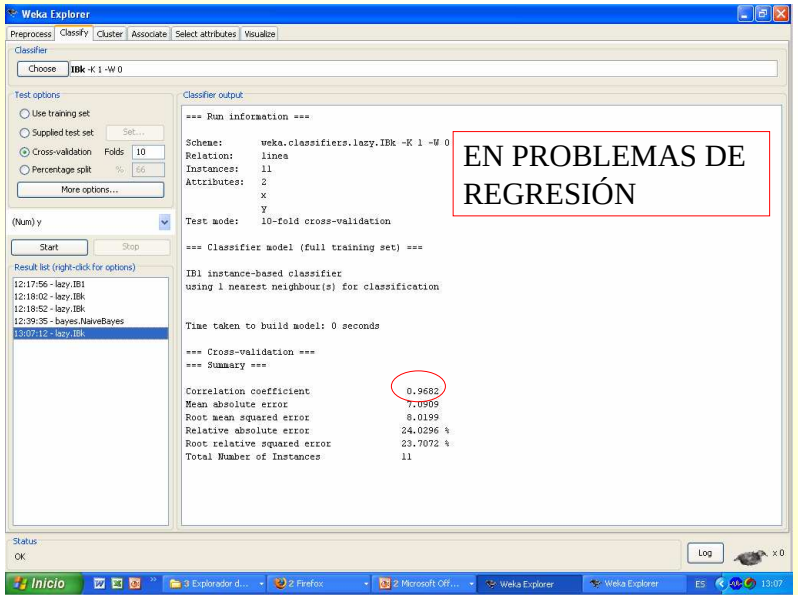
	Clasificado como +	Clasificado como -
Dato realmente +	**TP 60**	FN 40
Dato realmente -	FP 10	**TN 90**

Notese también que en los datos hay 100 personas con cáncer y 100 personas sin cáncer (sumar las líneas horizontales)

Visualización de resultados: % de aciertos, % de aciertos por clase (“true positive”), matriz de confusión



Visualización de resultados en Regresión

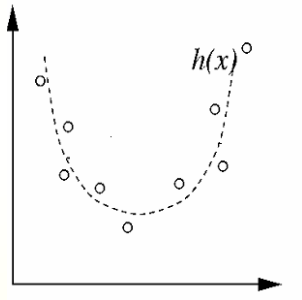


La Sobreadaptación (“overfitting”)

- Se produce sobreadaptación cuando el clasificador obtiene un alto porcentaje de aciertos en entrenamiento pero pequeño en test (es decir, no generaliza bien)
- Es decir, se está aprendiendo los datos, pero no está generalizando bien

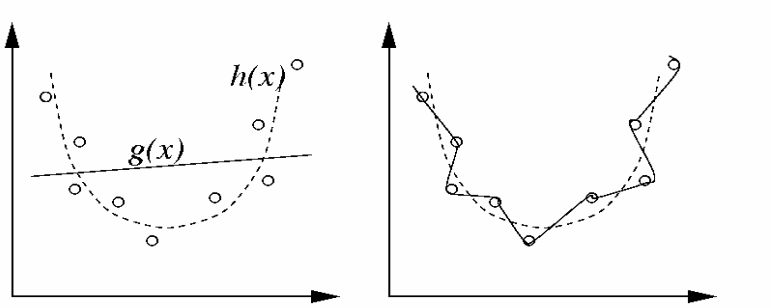
Sobreadaptación por ruido

- Supongamos que los datos están distribuidos según una parábola, pero hay algo de ruido
- Es decir, el modelo subyacente es una parábola, pero los datos muestran ligeras variaciones



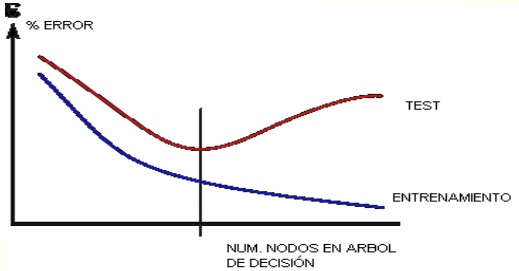
Sobreadaptación/subadaptación

- Derecha: el modelo se ha sobreadaptado al ruido porque es demasiado complejo
- Izquierda: el modelo lineal **g(x)** es demasiado simple para aproximar una parábola y subadapta los datos
- Conclusión: tiene que haber un equilibrio en la complejidad del clasificador (o del modelo en general)

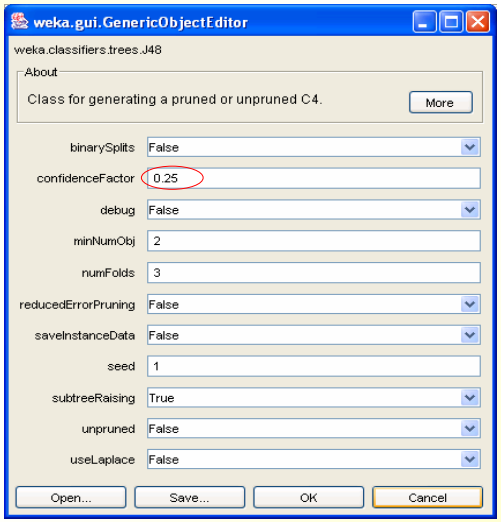


Sobreadaptación por crecimiento del clasificador

- Al principio, incrementar el tamaño del árbol de decisión disminuye el error en entrenamiento y test
- Pasada cierta complejidad del árbol, el error sigue decreciendo en entrenamiento pero crece en test
- Muchos algoritmos tienen parámetros que permiten controlar la complejidad del clasificador (en árboles de decisión, el parámetro de poda detiene el crecimiento)



Parámetro de j48 contra la sobreadaptación



Sobreadaptación. Resumen

- Factores que influyen: ruido, número de datos y complejidad del clasificador
- Ej: si hay pocos datos y permitimos gran complejidad al clasificador (“que crezca mucho”) habrá sobreadaptación (memorización)
- Ej: si hay ruido en los datos y permitimos gran complejidad al clasificador, se sobreadaptará al ruido
- Ej: pero si la complejidad del clasificador es insuficiente, habrá subadaptación

Otras medidas de evaluación: comprensibilidad

- En ocasiones es importante evaluar el conocimiento obtenido con otras medidas
- Comprensibilidad: si el conocimiento es fácilmente comprensible para un ser humano. Útil para evaluar si el conocimiento es correcto o para tomar decisiones en base al conocimiento obtenido
- Muy relacionado con el tamaño (número de reglas o nodos en el árbol de decisión)
- A veces merece la pena perder en porcentaje de aciertos (= subadaptación) para ganar en comprensibilidad (construyendo árboles de decisión más pequeños, discretizando atributos, etc.)

- **INTRODUCCIÓN A WEKA EN UN PROBLEMA DE ROBOSOCCER**

La herramienta Weka

- Página de Weka:

http://www.cs.waikato.ac.nz/ml/weka/

- El último software de Weka (con y sin máquina virtual Java):

http://www.cs.waikato.ac.nz/ml/weka/index_downloading.html

Tipos de atributos

- **Nominales** (discretos, categóricos): cielo, viento
- **Numéricos**: temperatura, humedad
- Hay atributos numéricos que son realmente nominales (ej: DNI)
- Hay atributos nominales que son realmente numéricos (ej: edad con valores “niño”, “joven”, “adulto”, “mayor”).

Formato arff. Definición de atributos

% Comentarios precedidos de %

@relation tiempo

@attribute cielo {sol, nubes, lluvia}

@attribute temperatura numeric

@attribute humedad numeric

@attribute viento {si, no}

@attribute tenis {si, no}

Formato arff. Definición de datos

@data

Sol, 85, 85, no, no

Sol, 80, 90, si, no

Nublado, 81, 86, no, si

Lluvia, 70, 96, no, si

...

Formato Arff

@relation tiempo
@attribute cielo {sol, nubes, lluvia}
@attribute temperatura numeric
@attribute humedad numeric
@attribute viento {si, no}
@attribute tenis {si, no}
@data
Sol, 85, 85, no, no
Sol, 80, 90, si, no
Nublado, 81, 86, no, si
Lluvia, 70, 96, no, si

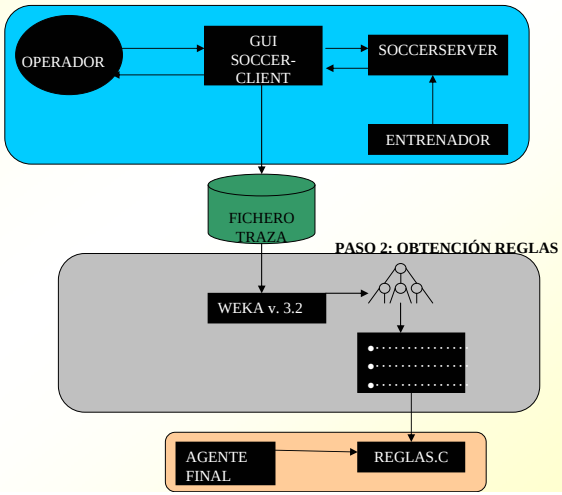
POSIBLES PASOS A REALIZAR

1. Comprender los datos:
   1. Visualización. Entender qué atributos son los mas relevantes, individualmente o por parejas
   2. Comprobar si es un dominio donde las clases están desequilibradas
2. Preproceso (normalización, muestreo, …)
3. Exploración inicial (explorer).
   1. Aplicar Zerop para ver porcentaje de aciertos a superar
   2. Ver que **tipo** de algoritmo es el mejor para nuestros datos. Obtener un resultado base (% aciertos)
   3. Encontrar modelos simplificados para entender mejor los datos
   4. Selección de atributos para simplificar el modelo y/o mejorar resultados

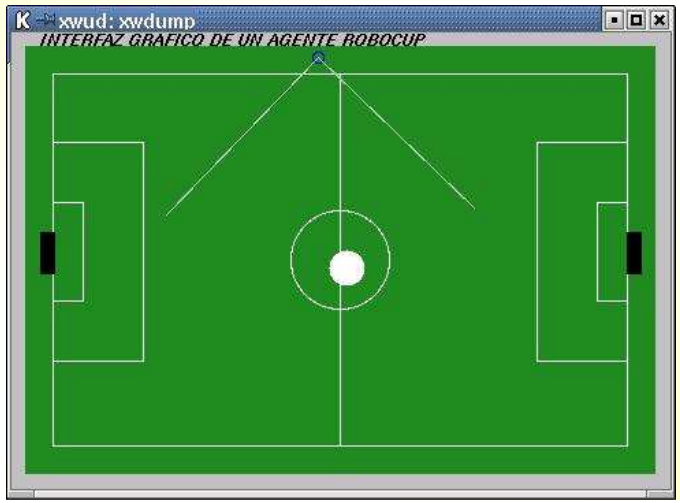
Modelización de Jugadores Humanos para la Robocup

- Se trata de obtener un modelo de una persona jugando a Robosoccer, para después programar a un agente que juegue de forma similar
- Datos para aprender: (Sensores, Acción) [lo que ve el agente en el momento *t*, lo que hace la persona en *t*]
- PFC Alberto López Cilleros

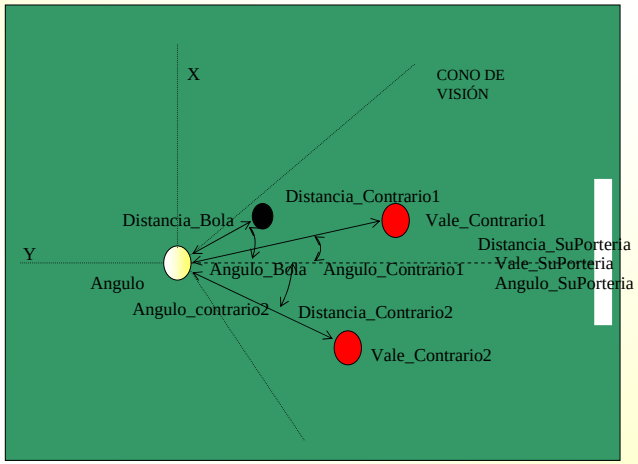
Esquema de aprendizaje



GUI Soccerclient



Atributos a utilizar



Acciones

Acciones
Avanzar rápido: **dash99**
Avanzar lento: **dash 60**
Girar 10º Derecha: **turn-right-10**
Girar 10º Izquierda: **turn-left-10**
Tirar a puerta: **kick99**
Tiro corto: **kick60**

Atributos en la robosoccer

@attribute Distancia_Bola real
@attribute Angulo_Bola real
@attribute Vale_Bola { 0, 1}
@attribute X real
@attribute Y real
@attribute Angulo real
@attribute Distancia_Contrario1 real
@attribute Angulo_Contrario1 real
@attribute Vale_Contrario1 real
@attribute Distancia_Contrario2 real
@attribute Angulo_Contrario2 real
@attribute Vale_Contrario2 { 0, 1 }
@attribute Distancia_SuPorteria real
@attribute Angulo_SuPorteria real
@attribute Vale_SuPorteria { 0, 1 }
%@attribute FueraCampo { 0, 1 }
@attribute Accion { dash99, dash60, turn10, turnmenos10, kick99, kick60 }

Datos en la robosoccer

@data

36.6,-4,1,3.26785,36.9995, 91,1000,1000,0,1000,1000,0,1000,1000,0,dash99

36.6,-4,1,3.34611,36.3996,-91,1000,1000,0,1000,1000,0,1000,1000,0,dash99

36.6,-4,1,2.78053,35.1998,-91,1000,1000,0,1000,1000,0,1000,1000,0,dash99

36.6,-4,1,2.78053,35.1998,-91,1000,1000,0,1000,1000,0,1000,1000,0,dash99

4.5,-41,0,0.61956,1.49459,-91,1000,1000,0,1000,1000,0,1000,1000,0,turnmenos10

4.5,-41,0,3.20972,1.3546,-100,1000,1000,0,1000,1000,0,1000,1000,0,turnmenos10

4.5,-41,0,2.28458,1.42641,-110,1000,1000,0,1000,1000,0,1000,1000,0,turn10

3.3,-35,1,2.37538,1.42044,-120,1000,1000,0,1000,1000,0,1000,1000,0,turn10

3.3,-25,1,3.62294,1.31465,-130,1000,1000,0,1000,1000,0,1000,1000,0,turn10

Ejemplo de conocimiento obtenido

if ((Angulo_Bola > -37)&&(Distancia_Bola > 1.2) &&(Angulo_Bola <= 24)) {dash99(memoria,puerto); break;}

if ((Angulo_Bola > 19)&&(Angulo_Bola <= 42)&&(X <= 33.9477)) {dash99(memoria,puerto);break;}

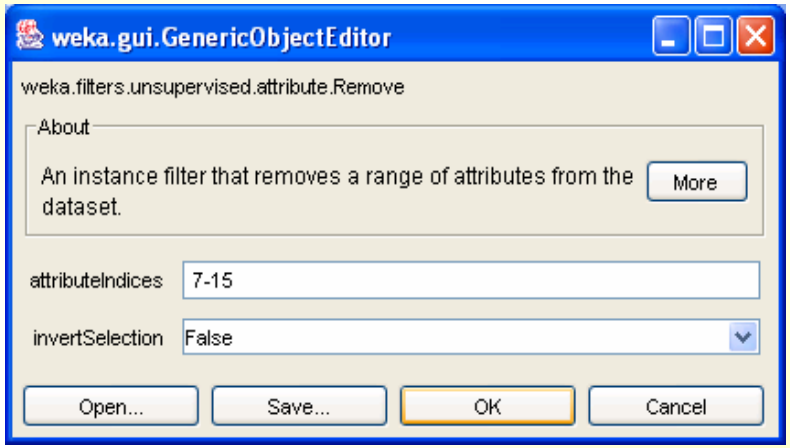
if ((Angulo_Bola > 11)) {turn10(memoria,puerto);break;}

if ((Distancia_Bola <= 0.4)&&(Angulo_Bola <= -20)) {turn10(memoria,puerto);break;}

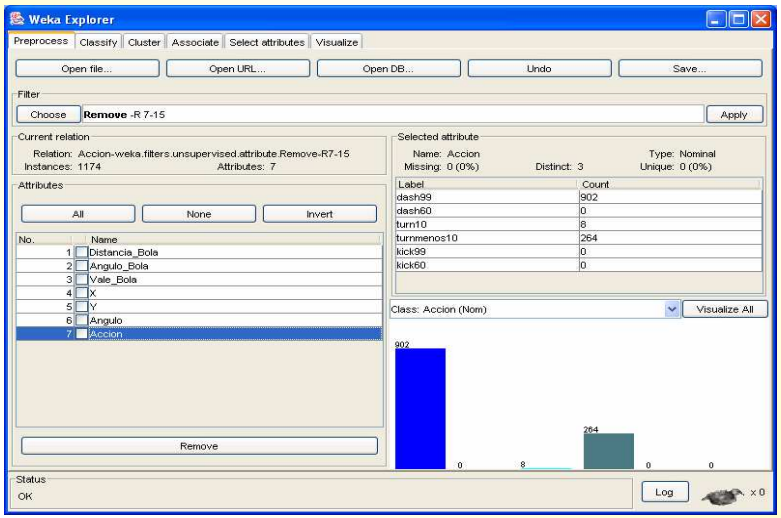
Filtros

- Supervisados: teniendo en cuenta la clase
- No supervisados: no tiene en cuenta la clase
- De atributo: Selección, borrado, creación, discretización, ...
- De instancia (datos): Selección, borrado, remuestreado, ...

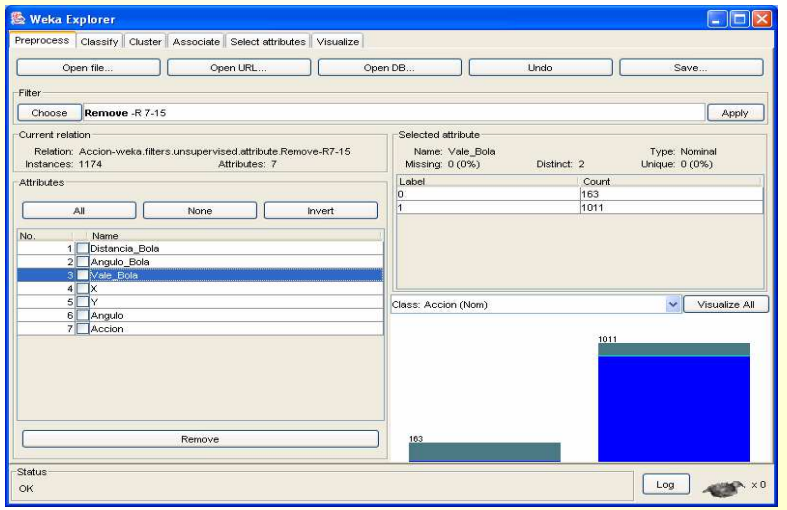
Remove (borrado atributos innecesarios)



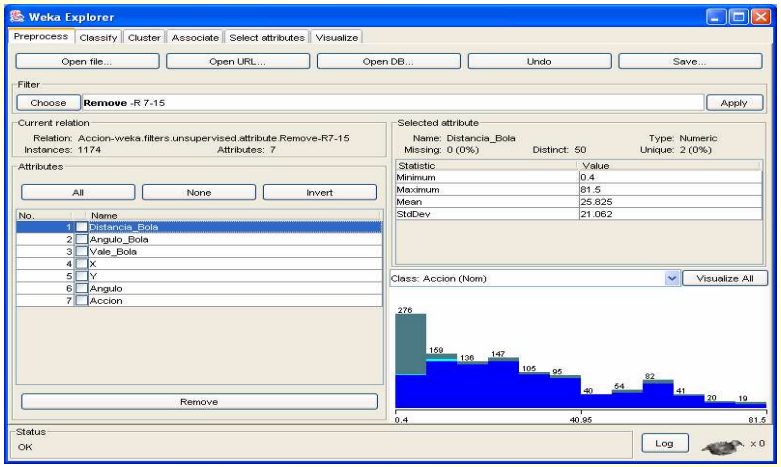
Visualización acción



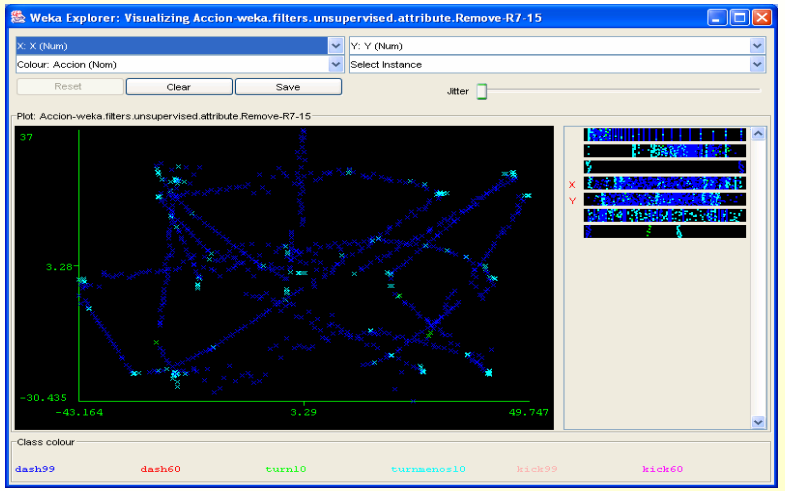
Visualización de vale_bola



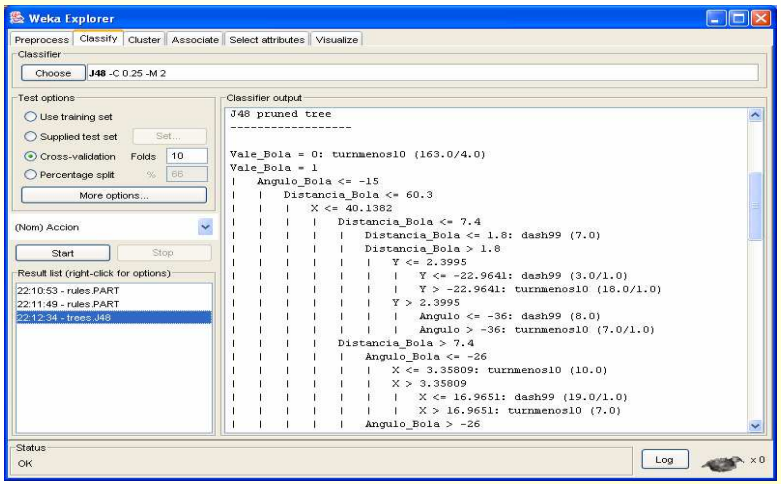
Visualización distancia_bola



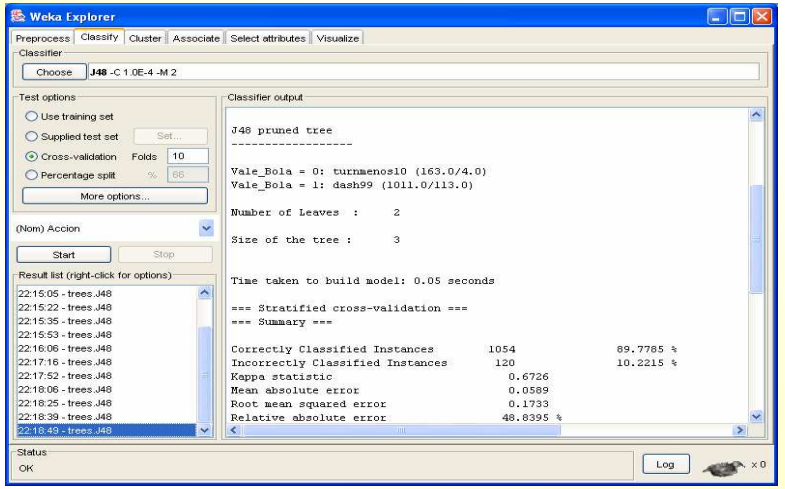
Visualización X,Y



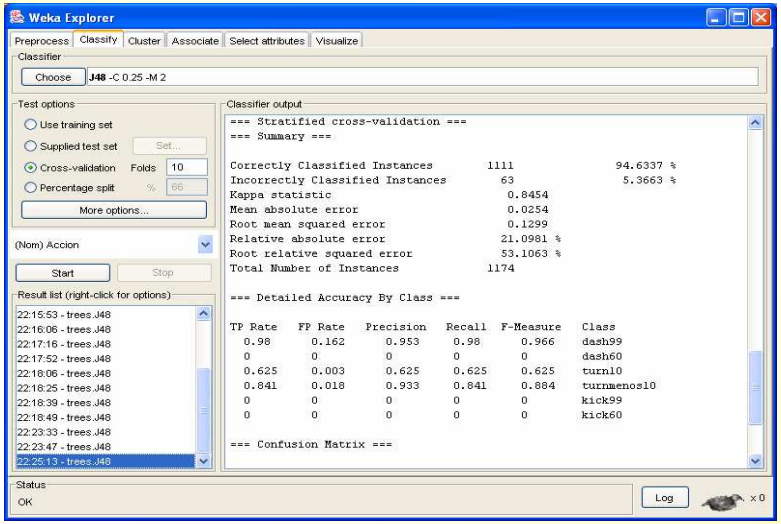
J48 aplicado a los datos



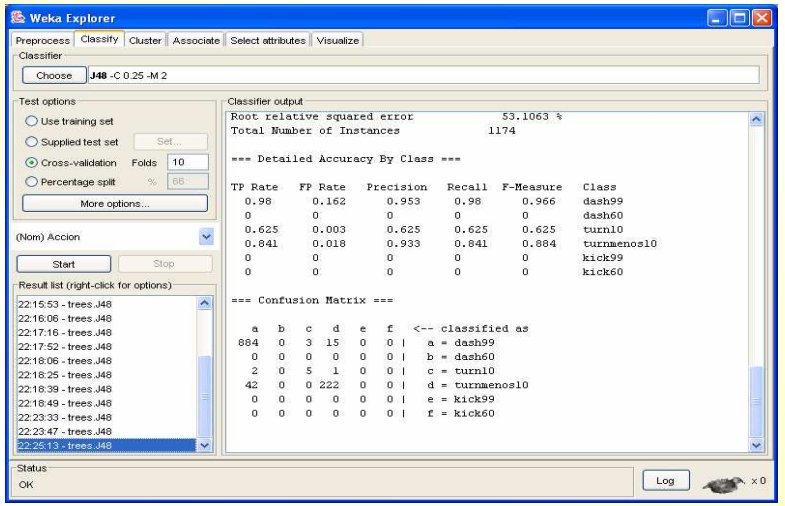
J48 Podado



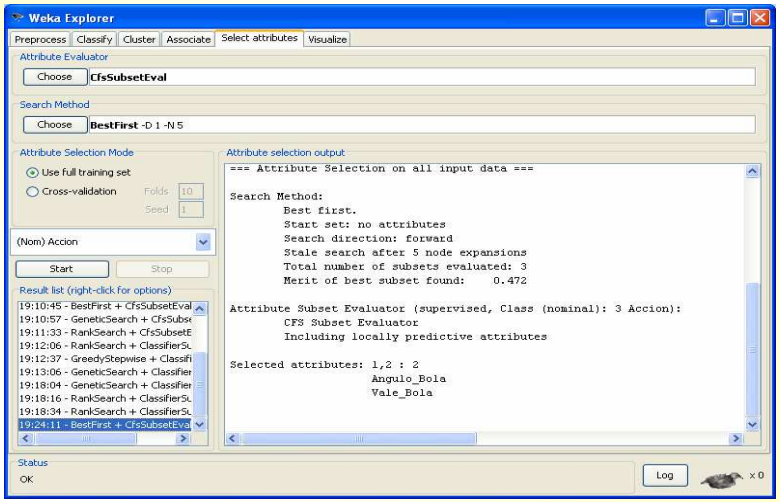
Porcentaje de aciertos



Matriz de confusión

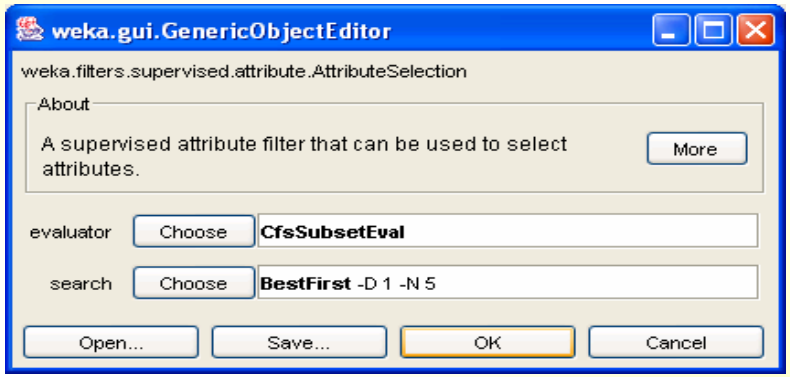


Selección de atributos (pestaña)

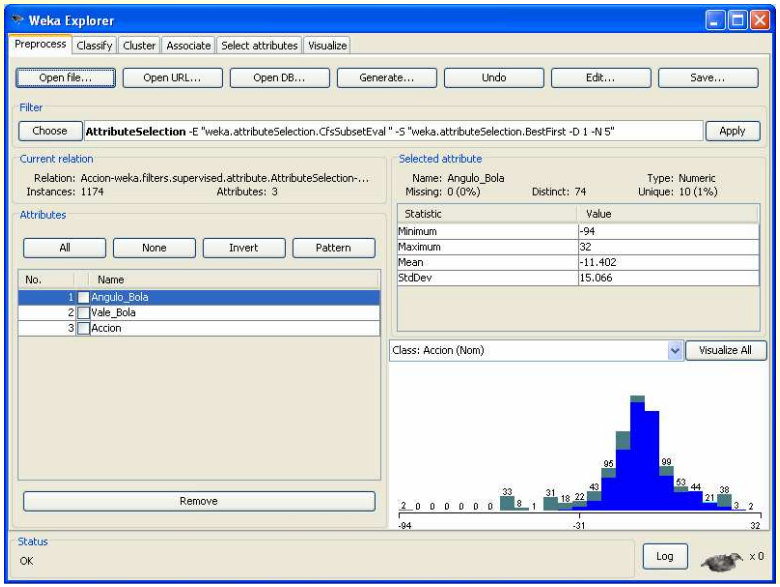


Parece seleccionar demasiado pocos atributos

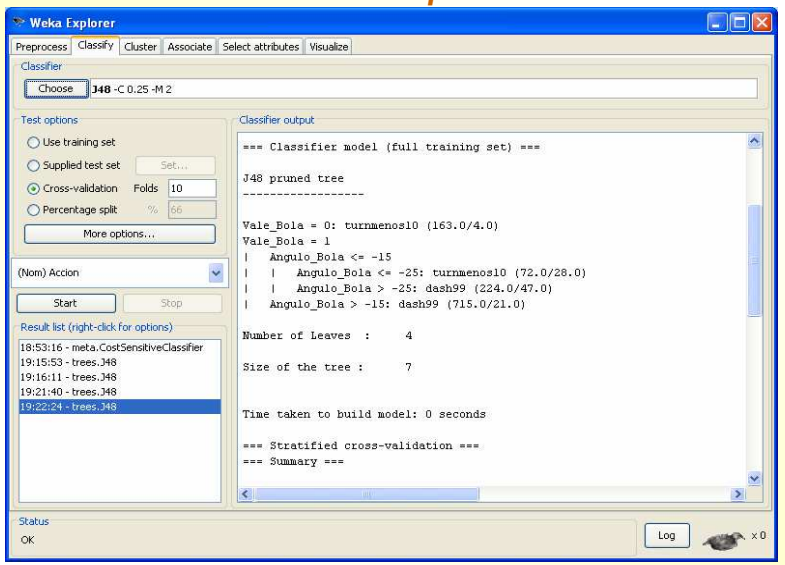
Selección de atributos (filtro attribute selection)



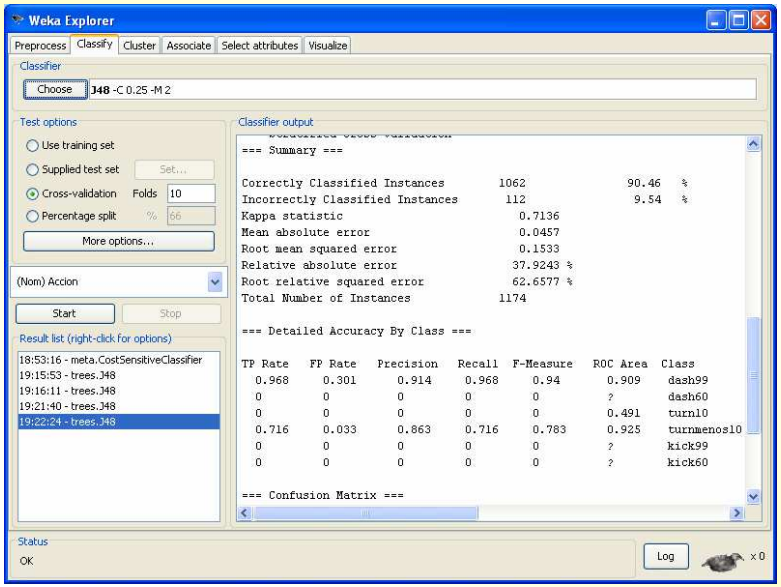
Ahora utilizamos el filtro



Y vemos el modelo que sale



Muy simple (es otra manera de simplificar el modelo)



Y con menos aciertos

Práctica

- Del tutorial:

http://www.dsic.upv.es/~cferri/weka/CursDoctorat-weka.pdf

- En este caso se trata de predecir el tipo de fármaco (drug) que se debe administrar a un paciente afectado de rinitis alérgica según distintos parámetros/variables. Las variables que se recogen en los historiales clínicos de cada paciente son:
  - Age: Edad
  - Sex: Sexo
  - BP (Blood Pressure): Tensión sanguínea.
  - Cholesterol: nivel de colesterol.
  - Na: Nivel de sodio en la sangre.
  - K: Nivel de potasio en la sangre.
- Hay cinco fármacos posibles: DrugA, DrugB, DrugC, DrugX, DrugY. Se han recogido los datos del medicamento idóneo para muchos pacientes en cuatro hospitales. Se pretende, para nuevos pacientes, determinar el mejor medicamento a probar.
- Fichero "Drug1n.arff" de:

http://www.dsic.upv.es/~cferri/weka/Datasets.rar

- O bien de uah.master@gmail.com, password=paraguas68, en Drafts

Práctica

1. Cargar los datos
2. Probar con el algoritmo ZeroR para obtener un resultado base
3. Probar con varios algoritmos de clasificación, con validación cruzada de 10. Intentar obtener el mejor porcentaje de aciertos
4. Probar a quitar atributos, para ver si el porcentaje de aciertos mejora

Práctica

- Tutorial:

http://www.dsic.upv.es/~cferri/weka/CursDoctorat-weka.pdf

- Hacer la parte 1 y 2 para repasar lo visto
- De los siguientes datos, extraer "Drug1n.arff"

http://www.dsic.upv.es/~cferri/weka/Datasets.rar

- Probar varios algoritmos hasta encontrar el que de el mejor porcentaje de aciertos con validación cruzada de 10
- Probar a quitar algún atributo para ver si se consigue mejorar el porcentaje de aciertos, o se concluye que todos los atributos son necesarios
- Hacer la parte 3.1 y 3.2 del tutorial (y si da tiempo, seguir)

- **SELECCIÓN DE ATRIBUTOS**

Selección de atributos

- Algunos atributos pueden ser **redundantes** (como "salario" y "categoría social") y hacen más lento el proceso de aprendizaje
- Otros son **irrelevantes** (como el DNI para predecir si una persona va a devolver un crédito)
- En ocasiones el exceso de atributos puede llevar a sobreaprendizaje, pues incrementa la complejidad del modelo (sobre todo si hay pocos datos)
- En ocasiones es útil tener el conocimiento de qué atributos son relevantes para una tarea
- Existen algoritmos de selección de atributos

Métodos de selección de atributos

- El método mas preciso es la búsqueda exhaustiva
- Supongamos que tenemos 4 atributos A, B, C, D
- Sería necesario comprobar la validez de todos los posibles subconjuntos (2^4=16): {A, B, C, D}, {A, B, C}, {A, B, D}, {B, C, D}, {A, C, D}, {A, B}, {A, C}, ..., {A}, {B}, {C}, {D}
- En general, el método es poco práctico: 2^n posibles subconjuntos

Métodos de selección de atributos

1. Evaluación individual de atributos (ranker u ordenación)
2. Evaluación de subconjuntos:
   - Filter: se evalúan los atributos de manera separada
   - Wrapper: se evalúan de manera conjunta

Selección de atributos. Ranker

- Dado unos atributos A_1, A_2, ..., A_n
- Se evalúa cada uno de manera independiente, calculando medidas de correlación del atributo con la clase
- Un atributo A_1 está correlacionado con la clase, si conocer su valor implica que podemos predecir la clase
- Por ejemplo, el sexo de una persona está correlacionado (de momento) con que le guste el fútbol. Su DNI no lo está
- Por ejemplo, el salario de una persona está correlacionado con el hecho de que vaya a devolver un crédito

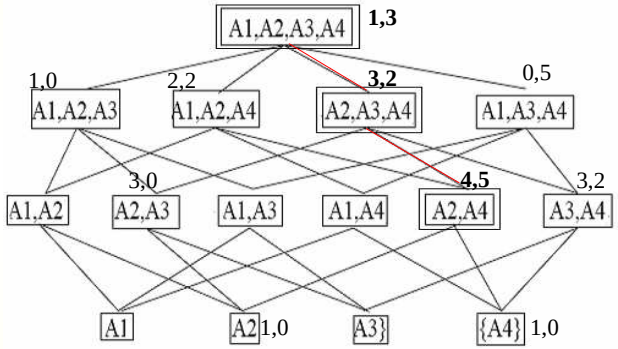
Selección de atributos. Ranker

- Se evalúa cada atributo con algún estadístico que detecte la correlación (ej: chi-cuadrado, infogain, etc.)
- Se ordenan los atributos según ese valor
- Se seleccionan los k mejores
- Método muy rápido
- Problemas:
  - No detecta atributos redundantes
  - En ocasiones no tiene sentido evaluar a los atributos por separado, sino en conjunto.
  - Ej: las aparición de las palabras “inteligencia” y “artificial” no está excesivamente correlacionado por separado con textos de informática, pero juntas su correlación se incrementa notablemente

Selección de atributos. Evaluación de subconjuntos

- Estos métodos recorren un espacio de búsqueda de subconjuntos de atributos, evaluando **subconjuntos** de atributos
- No se recorre el espacio entero, sino sólo aquellos subconjuntos más prometedores
- Se evalúa el subconjunto de manera conjunta

- Tipos:
  - Filter
  - Wrapper



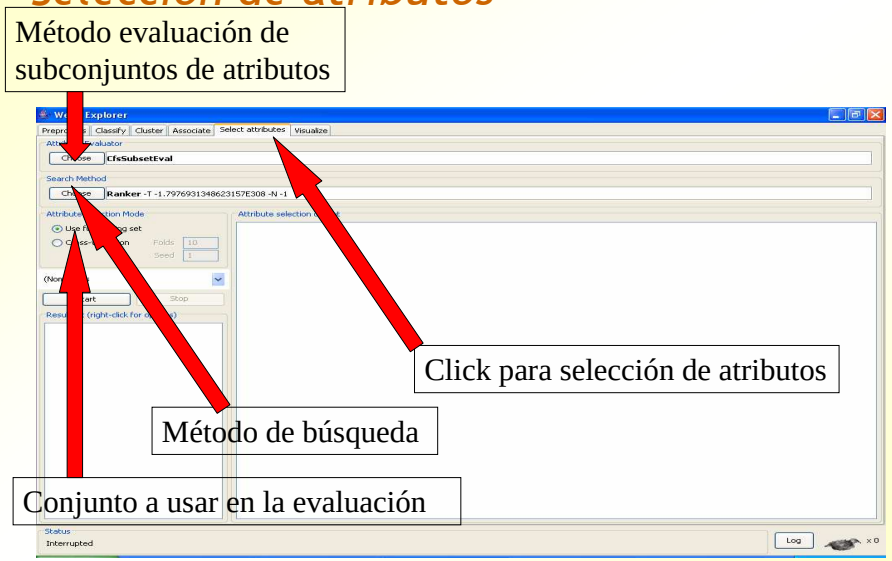
Selección de atributos. Filter

- Los métodos *filter* evalúan un subconjunto de atributos calculando:
  - La media de las correlaciones (o similar) de cada atributo con la clase
  - Descontando puntos por redundancias entre atributos
- Método rápido
- Problemas: elimina atributos redundantes, pero como ranker, puede eliminar atributos que por si solos no están correlacionados con la clase, pero con otro atributo si que lo están (ej: “inteligencia artificial”)

Selección de atributos. Wrapper

- Los métodos *Wrapper* evalúan un subconjunto de atributos ejecutando un algoritmo de minería de datos (MD) concreto, sobre un conjunto de entrenamiento
- El valor del subconjunto es el porcentaje de aciertos obtenido con esos atributos
- Son lentos
- Obtienen subconjuntos de atributos adecuados para un algoritmo de MD concreto
- Evalúan a los atributos de los subconjuntos de manera realmente conjunta

Selección de atributos



Selección atributos. Tipos

- Ranker: evaluación de atributos individuales
  - Búsqueda: **Ranker**
  - Evaluador: ChiSquare**AttributeEval**, GainRatio**AttributeEval**, InfoGain**AttributeEval**
- Evaluación de subconjuntos de atributos:
  - Búsqueda: greedy, stepwise, genetic, ...
  - Evaluador:
    - Filter: Cfs**SubsetEval**
    - Wrapper:
      - Classifier**SubsetEval**
      - Wrapper**SubsetEval**

Evaluadores de atributos (Ranker)

- ChiSquaredAttributeEval: usa el estadístico Chi-squared para evaluar el valor predictivo del atributo
- GainRatioAttributeEval: usa gainratio
- InfoGainAttributeEval: usa infogain

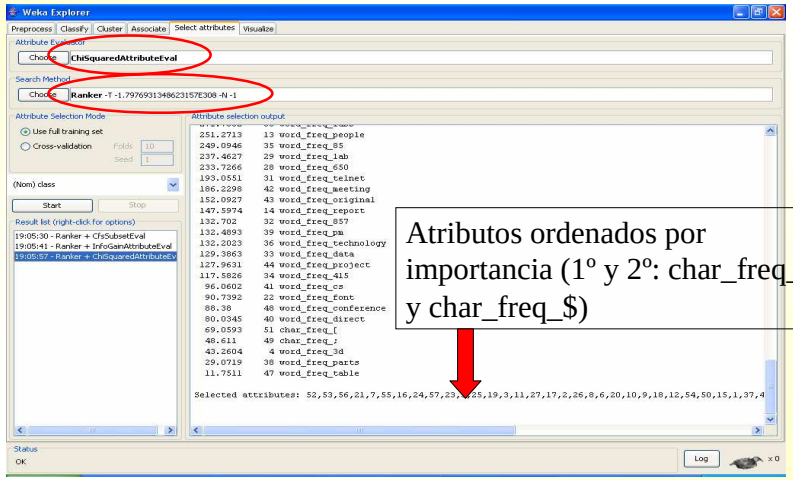
Evaluadores de subconjuntos

- FILTER: rápidos
  - CfsSubsetEval: considera el valor predictivo (correlación) de cada atributo y de su redundancia
- WRAPPER: más lentos
  - ClassifierSubsetEval: usa un clasificador para evaluar el conjunto
  - WrapperSubsetEval: classificador + validación cruzada

Métodos de búsqueda

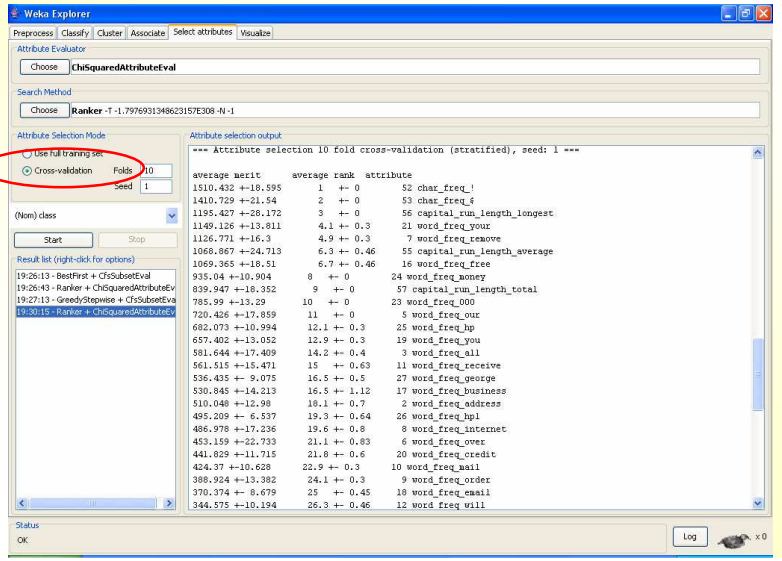
- BestFirst: Mejor primero (lento)
- ExhaustiveSearch: Búsqueda exhaustiva (muy lento)
- GeneticSearch: Búsqueda genética (rápido)
- GreedyStepWise: Escalada (muy rápido)
- RankSearch: Primero ordena los atributos y después construye el subconjunto de manera incremental, en dirección del mejor al peor, hasta que no merece la pena añadir nuevos atributos (rápido)

Selección Ranker

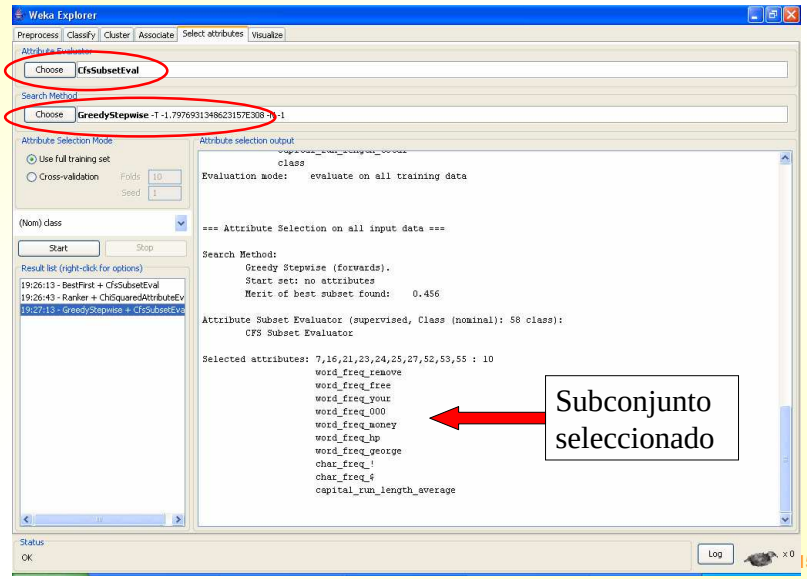


Atributos ordenados por importancia (1º y 2º: char_freq_! y char_freq_$)

Selección Ranker con validación cruzada (mérito y rango medios)



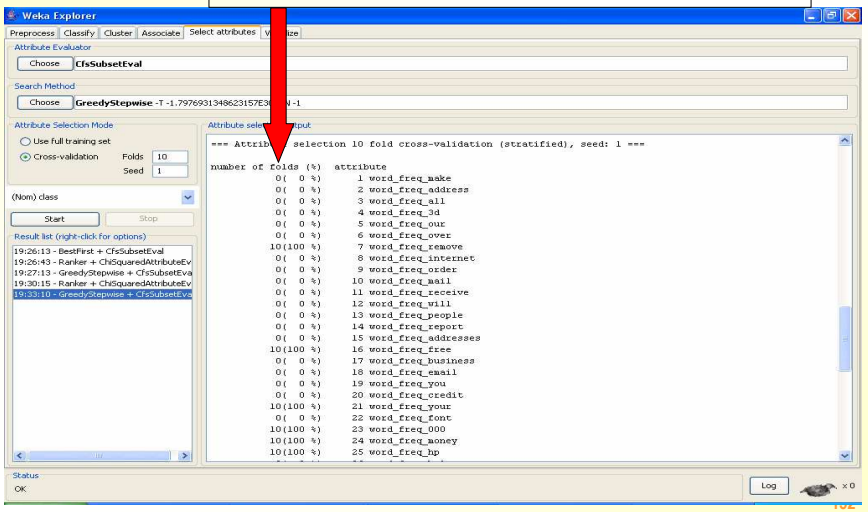
Selección Filter



Subconjunto seleccionado

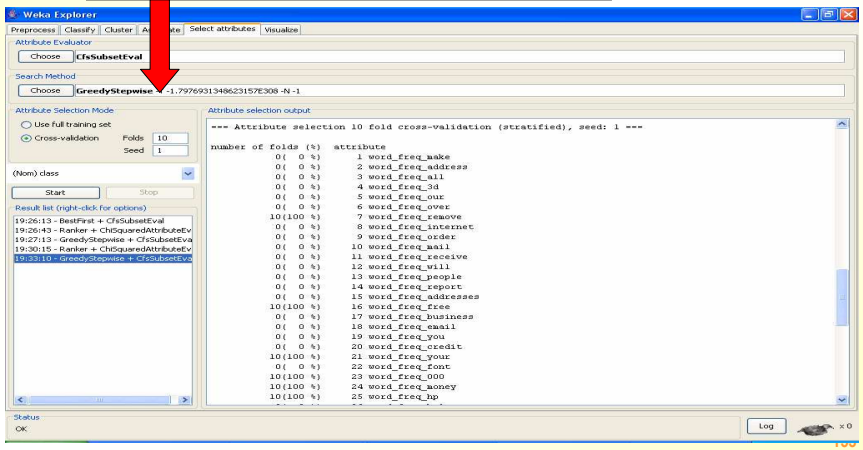
Selección Filter con validación cruzada

Número de folds en que el atributo fue seleccionado

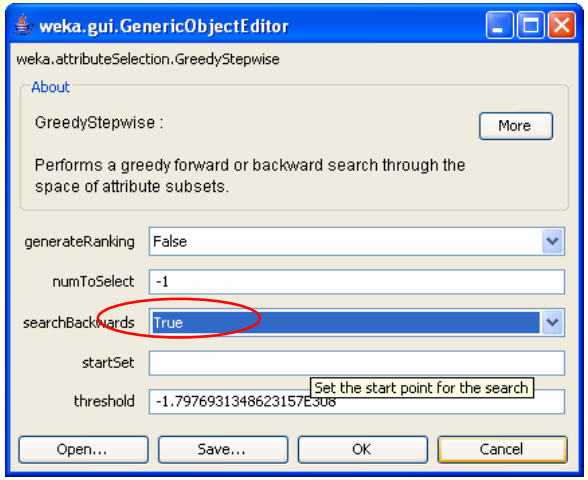


Búsqueda partiendo del conjunto total de atributos (backward en lugar de forward)

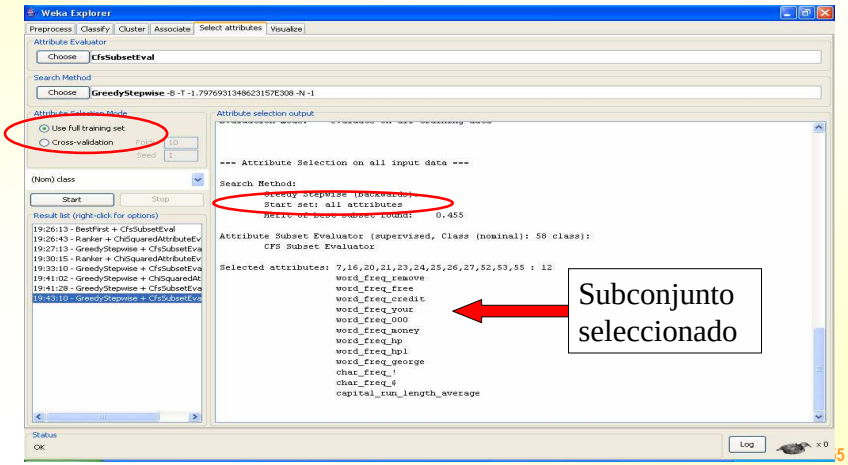
Click para parámetros de la búsqueda



Búsqueda partiendo del conjunto total de atributos (backward en lugar de forward)

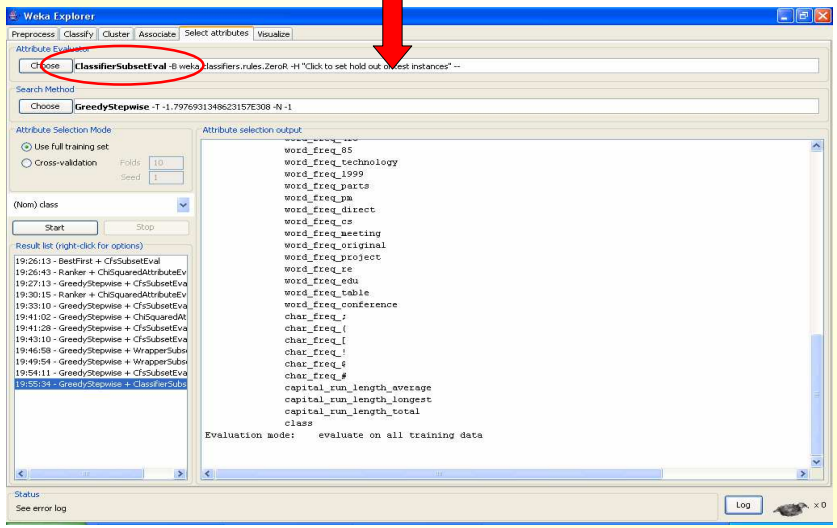


Búsqueda partiendo del conjunto total de atributos (backward en lugar de forward)

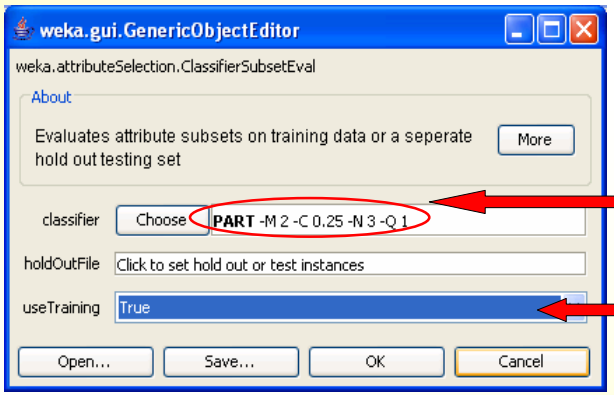


Método Wrapper

Click para parámetros de Wrapper



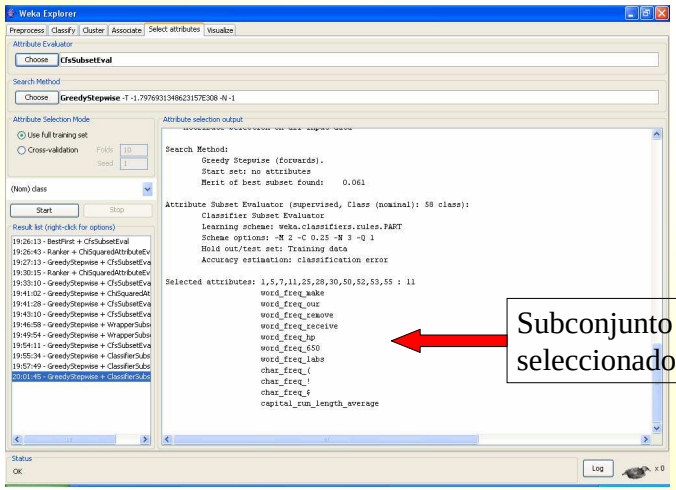
Selección parámetros Wrapper



Usaremos PART como clasificador

Usaremos el conjunto de entrenamiento para calcular los aciertos

Resultados Wrapper (¡lento!)

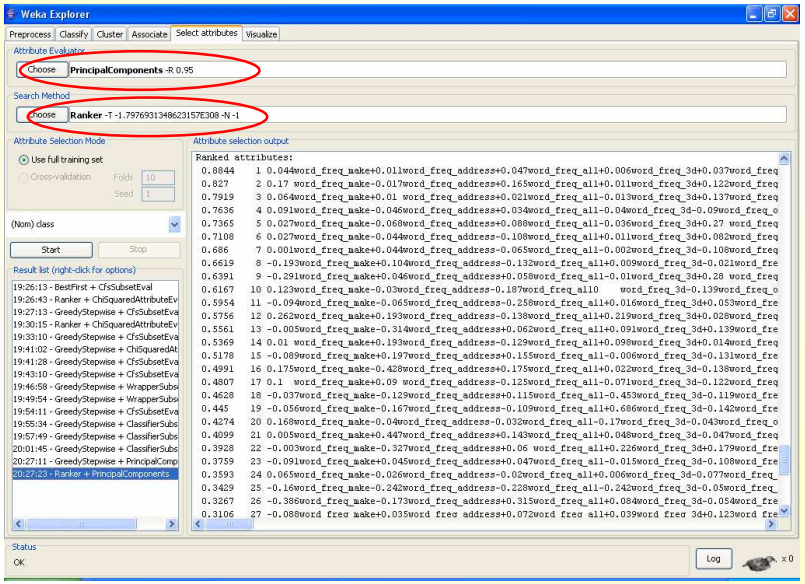


Subconjunto seleccionado

Selección con Principal Component Analysis (PCA)

- Este método construye nuevos atributos como combinación lineal de los anteriores
- Esos nuevos atributos están ordenados por importancia (varianza explicada)
- Se puede reducir la dimensionalidad escogiendo sólo algunos de los atributos

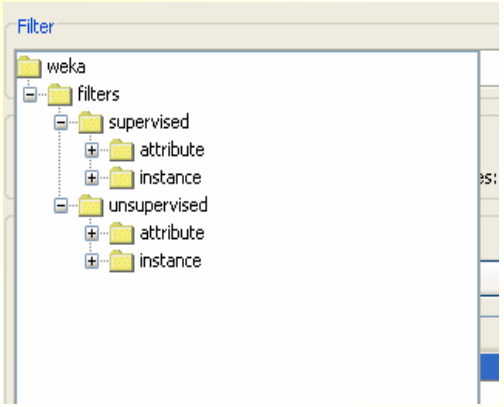
Resultados PCA



Notas selección de atributos

- Permite ver por pantalla los atributos seleccionados
- Pero no permite utilizar esa selección de atributos automáticamente para clasificar
- Para ello es necesario ir a la pestaña “preprocess” y seleccionar el filtro “Attribute Selection”

Filtros (pestaña de preproceso)

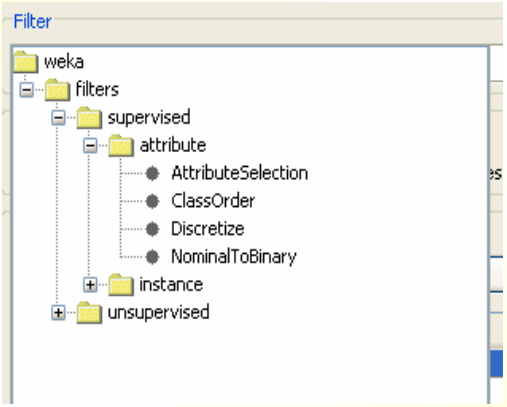


*Supervisados (tienen en cuenta la clase)

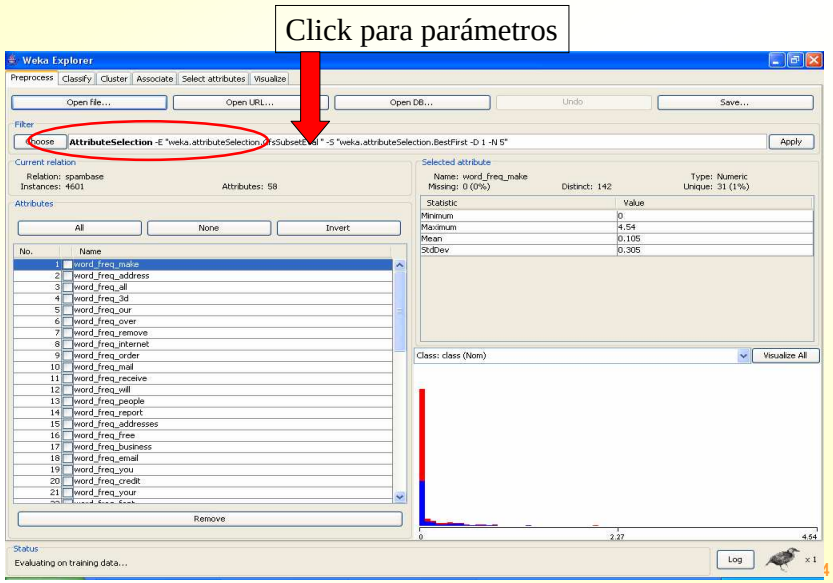
*No supervisados

De atributos y de instancias (datos)

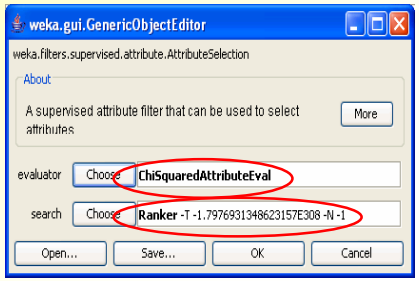
Filtro de selección de atributos

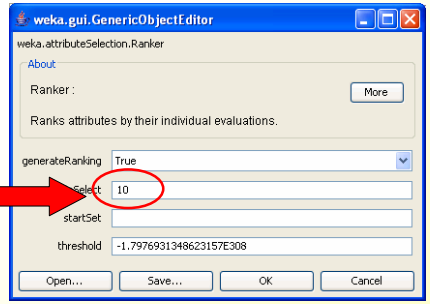


Filtro de selección de atributos



Parámetros de selección de atributos

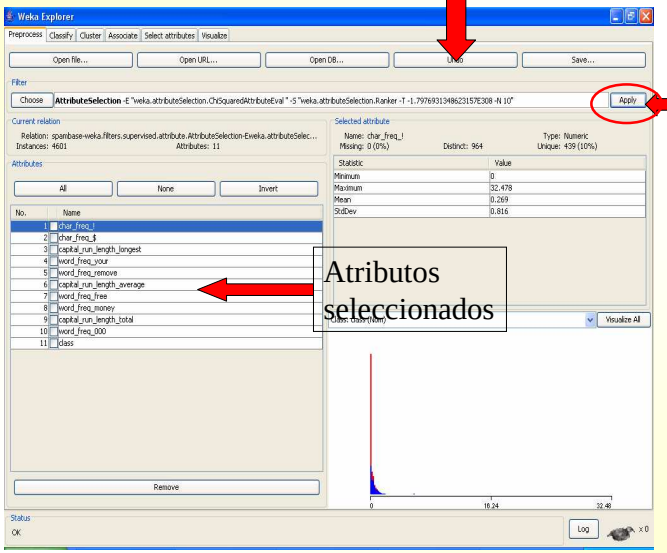




Seleccionaremos los 10 mejores atributos, tras la ordenación

Resultados de la selección

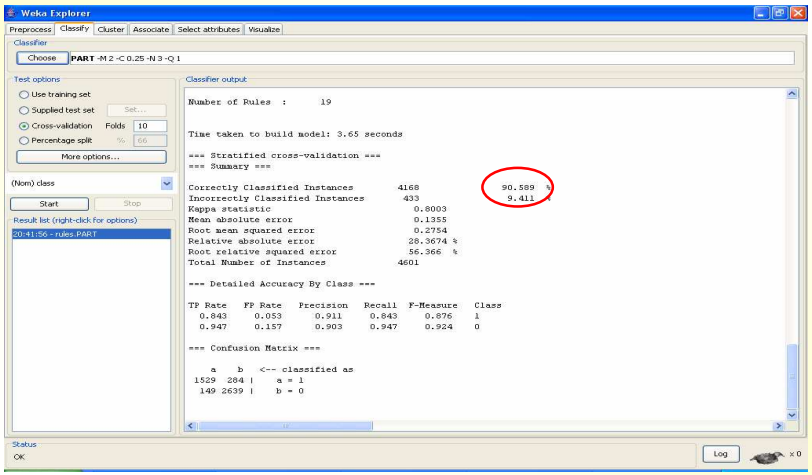
Podemos deshacer los cambios



¡Hay que pulsar Apply!

Atributos seleccionados

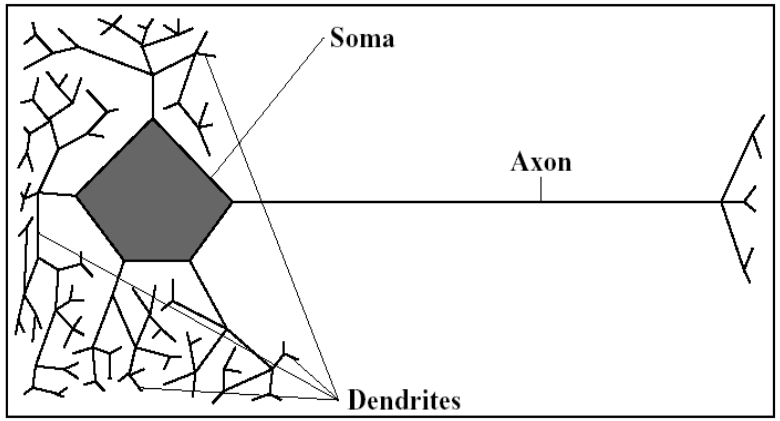
Resultados de la clasificación con los nuevos atributos



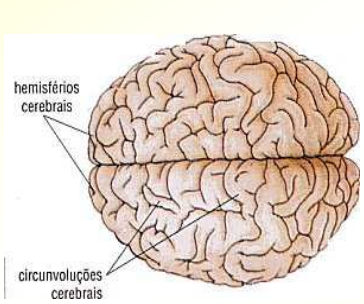
El Brain Computer Interface *(BCI)*

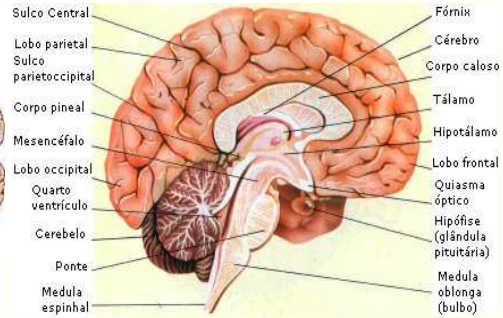
- Objetivo: comunicar personas con ordenadores mediante el pensamiento
- Ayudar a personas inmovilizadas
- Existen otros métodos (movimiento de los ojos, nervios, etc.)
- Más natural, se puede pensar en movimientos a alto nivel

Neuronas (interruptor 0/1)

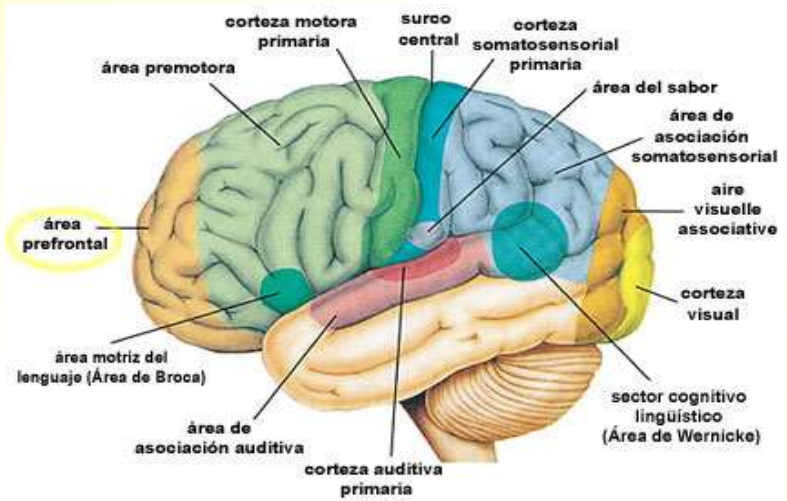


El cerebro (red de billones de neuronas)



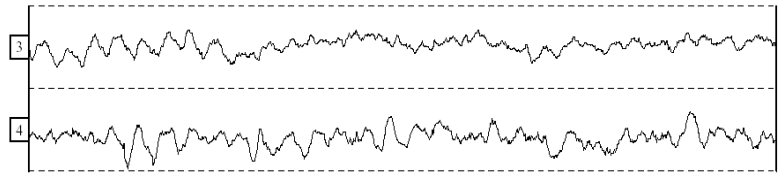


Áreas funcionales del cerebro



El electro-encefalograma (EEG)

- Cambios de potencial -> ondas electromagnéticas (muy débiles)
- Medición: invasiva o no invasiva

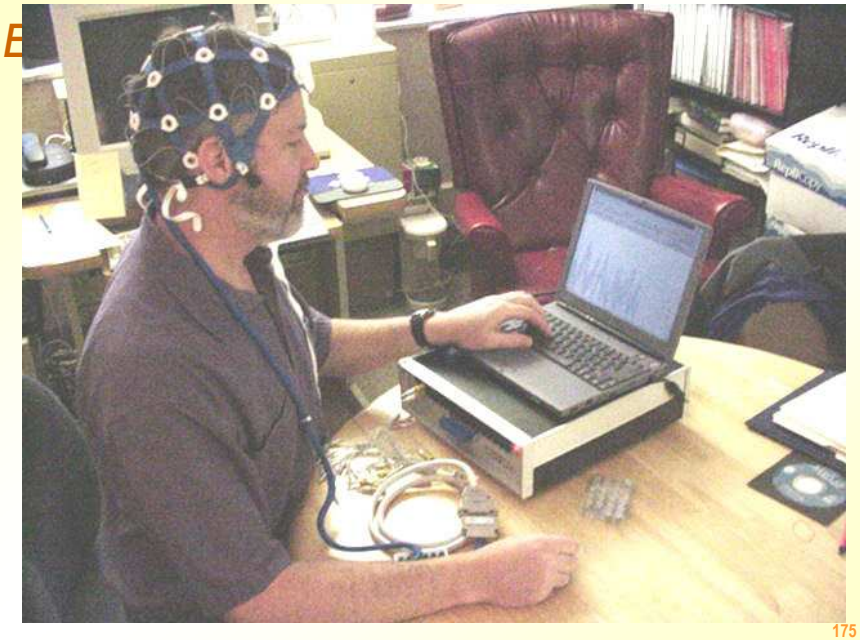


Aplicaciones del EEG

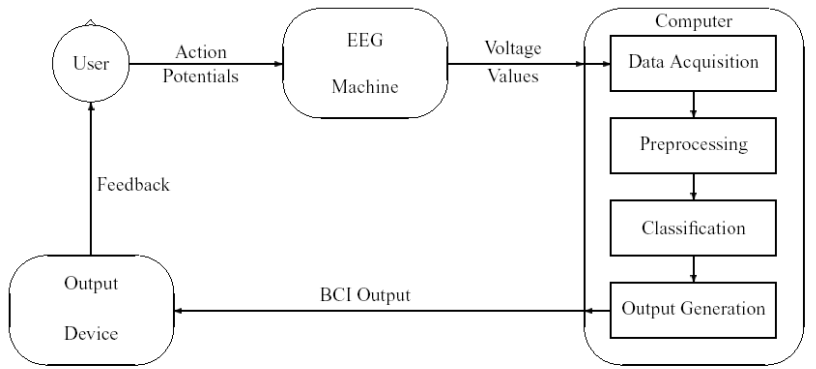
- Diagnóstico de enfermedades (epilepsia)
- Biofeedback
- El Interfaz cerebro-máquina

Frequency Band	Range
Alpha (α)	$8 - 13$ Hz
Beta (β)	$14 - 30$ Hz
Theta (θ)	$4 - 7$ Hz
Delta (δ)	$0.5 \quad 3$ Hz

E



Esquema del BCI



El spellboard